Řazení prvků v poli

Quick sort - Hoare

Projekt CZ.1.07/1.5.00/34.1009
DUM VY_32_INOVACE_270

Ing. Karel Johanovský

Střední průmyslová škola Jihlava

2013

Identifikační údaje

Projekt	***Inovace výuky prostřednictvím ICT***
Číslo projektu	*CZ.1.07/1.5.00/34.1009*
Číslo DUM	*VY_32_INOVACE_270*
Autor	*Ing. Karel Johanovský*
Datum vytvoření	*2. dubna 2013*
Tematický celek	*Programování a vývoj aplikací - řazení*
Téma	*Quick sort - Hoare*
Anotace	*Podpora výuky řadících algoritmů*
Metodický pokyn	*Prezentace s výkladem, časová náročnost 20 minut*
Inovace	*Podpora vjemu informací u žáka ve fázi expozice .* *a zejména ve fázi fixace získaných poznatků* *(dostupný materiál – možnost libovolného počtu opakování)*

Obsah

Shrnutí

- V minulých hodinách jsme si představili tzv. kvadratické řadící algoritmy.
- Jako první bylo bublinkové řazení, tzv.: bubble sort s několika vylepšeními: zarážka, ripple, shake a shuttle sort
- Dále jsme si ukázali řazení výběrem: select sort
- A nakonec jsme si předvedli řazení vkládáním: insert sort a jeho vylepšenou variantu binární insert sort.

Quick sort - Hoare

- To by bylo krátké připomenutí řadících algoritmů z minulých přednášek.

Quick sort - Hoare

- To by bylo krátké připomenutí řadících algoritmů z minulých přednášek.
- Dnes si ukážeme první z rychlých řazení, tzv.: quick sort.

Quick sort - Hoare

- To by bylo krátké připomenutí řadících algoritmů z minulých přednášek.

- Dnes si ukážeme první z rychlých řazení, tzv.: quick sort.
- Publikoval ho sir Charles Anthony Richard Hoare v roce 1962.

Quick sort - Hoare

- To by bylo krátké připomenutí řadících algoritmů z minulých přednášek.

- Dnes si ukážeme první z rychlých řazení, tzv.: quick sort.
- Publikoval ho sir Charles Anthony Richard Hoare v roce 1962.
- Základní myšlenkou je rozdělení řazené posloupnosti čísel na dvě přibližně stejné části.

Quick sort - Hoare

- To by bylo krátké připomenutí řadících algoritmů z minulých přednášek.

- Dnes si ukážeme první z rychlých řazení, tzv.: quick sort.
- Publikoval ho sir Charles Anthony Richard Hoare v roce 1962.
- Základní myšlenkou je rozdělení řazené posloupnosti čísel na dvě přibližně stejné části.
- V jedné části jsou čísla větší a ve druhé menší, než nějaká zvolená hodnota nazývaná pivot.

Quick sort - Hoare

- To by bylo krátké připomenutí řadících algoritmů z minulých přednášek.

- Dnes si ukážeme první z rychlých řazení, tzv.: quick sort.
- Publikoval ho sir Charles Anthony Richard Hoare v roce 1962.
- Základní myšlenkou je rozdělení řazené posloupnosti čísel na dvě přibližně stejné části.
- V jedné části jsou čísla větší a ve druhé menší, než nějaká zvolená hodnota nazývaná pivot.
- Pokud je tato hodnota zvolena dobře, jsou obě části přibližně stejně velké.

Quick sort - Hoare

- To by bylo krátké připomenutí řadících algoritmů z minulých přednášek.

- Dnes si ukážeme první z rychlých řazení, tzv.: quick sort.
- Publikoval ho sir Charles Anthony Richard Hoare v roce 1962.
- Základní myšlenkou je rozdělení řazené posloupnosti čísel na dvě přibližně stejné části.
- V jedné části jsou čísla větší a ve druhé menší, než nějaká zvolená hodnota nazývaná pivot.
- Pokud je tato hodnota zvolena dobře, jsou obě části přibližně stejně velké.
- Pokud budou obě části samostatně seřazeny, je seřazené i celé pole.

Quick sort - Hoare

- To by bylo krátké připomenutí řadících algoritmů z minulých přednášek.

- Dnes si ukážeme první z rychlých řazení, tzv.: quick sort.
- Publikoval ho sir Charles Anthony Richard Hoare v roce 1962.
- Základní myšlenkou je rozdělení řazené posloupnosti čísel na dvě přibližně stejné části.
- V jedné části jsou čísla větší a ve druhé menší, než nějaká zvolená hodnota nazývaná pivot.
- Pokud je tato hodnota zvolena dobře, jsou obě části přibližně stejně velké.
- Pokud budou obě části samostatně seřazeny, je seřazené i celé pole.
- Obě menší části se pak REKURZIVNĚ řadí stejným postupem.

Hoareho quick sort v JAVĚ

- Naše řešení naprogramujeme jako funkci, která převezme pole a meze ve kterých má řadit a seřadí jej.

```
public static void QuickSortHoare(int l, int r, int[] pole) {
  int i = l;
  int j = r;
  int pivot = pole[(l + r) / 2];
  do {
    while (pole[i] < pivot) i++;
    while (pole[j] > pivot) j--;
    if (i <= j) {
      int tmp = pole[i];
      pole[i] = pole[j];
      pole[j] = tmp;
      i++;
      j--;
    }
  } while (i < j);
  if ((j - l) > 0) QuickSortHoare(l, j, pole);
  if ((r - i) > 0) QuickSortHoare(i, r, pole);
}
```

Použití

- Poté naši funkci vezmeme a použijeme v metodě main, kterou jsme vytvořili v první přednášce.
- Pozor meze musí být platné pozice v poli.

```
import java.util.Random;
public class SortingAlg {
  public static void main(String[] args) {
    int velikost = 100;
    Random rd = new Random();
    int pole[] = new int[velikost];
    for (int i = 0; i < pole.length; i++) {
      pole[i] = rd.nextInt(velikost);
      System.out.print(pole[i] + "\t");
    }
    QuickSortHoare(0, pole.length-1, pole);
    for (int i = 0; i < pole.length; i++) {
      System.out.print(pole[i] + "\t");
    }
  }
}
```

Složitost - vzorce

- Nyní se podívejme na složitost algoritmu.

Složitost - vzorce

- Nyní se podívejme na složitost algoritmu.
- Nejprve tu máme fázi půlení, těch bude:
  - $log_2(N)$

Složitost - vzorce

- Nyní se podívejme na složitost algoritmu.
- Nejprve tu máme fázi půlení, těch bude:
  - $log_2(N)$
- A v každé fázi musíme projít celé pole a přehazovat prvky, pokud je to nutné:
  - N

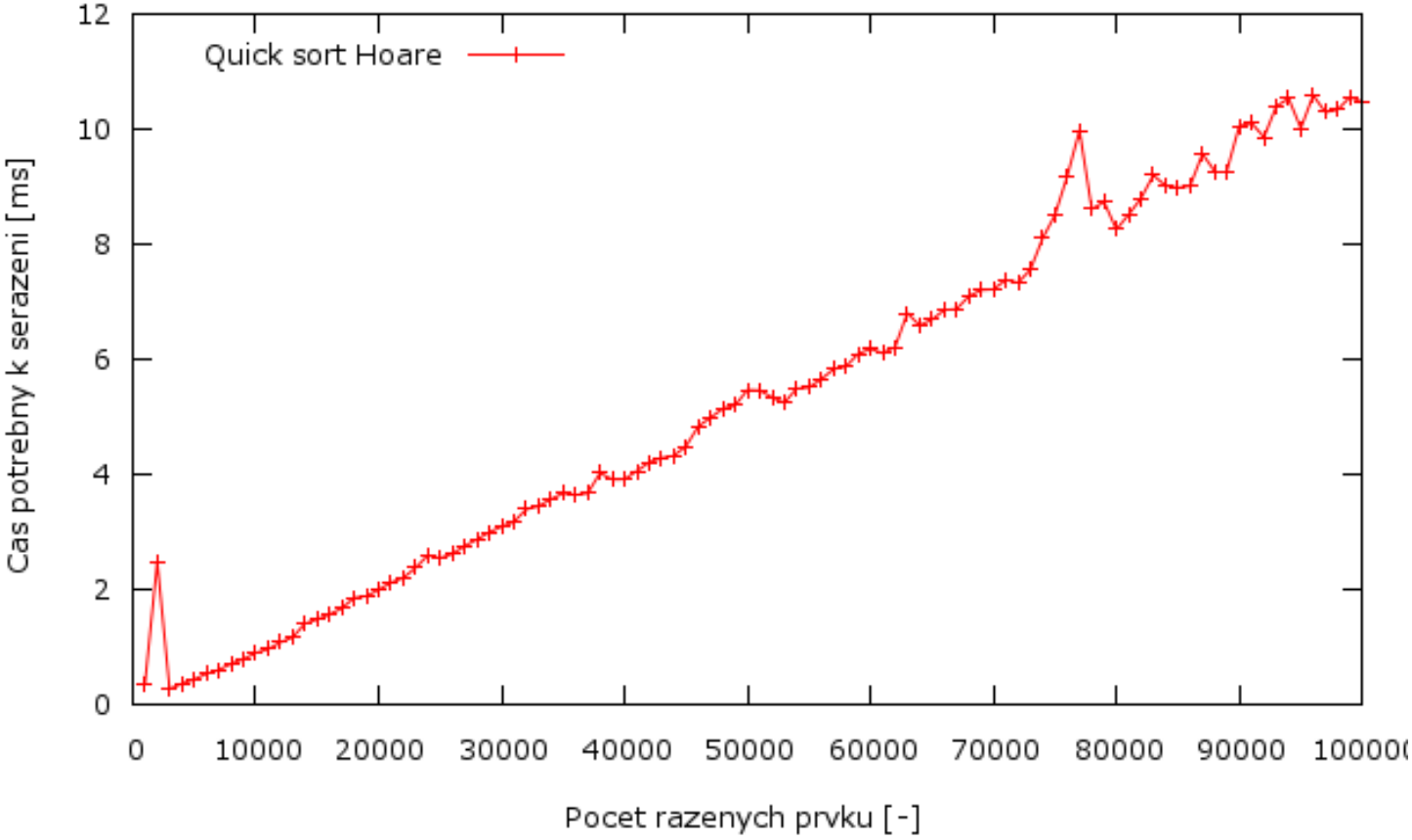
Složitost - vzorce

- Nyní se podívejme na složitost algoritmu.
- Nejprve tu máme fázi půlení, těch bude:
  - $log_2(N)$
- A v každé fázi musíme projít celé pole a přehazovat prvky, pokud je to nutné:
  - N
- Dohromady je to tedy celkem: $N * log_2(N)$, nic méně rychlost tohoto algoritmu je velmi závislá na volbě pivota.

Složitost - vzorce

- Nyní se podívejme na složitost algoritmu.
- Nejprve tu máme fázi půlení, těch bude:
  - $log_2(N)$
- A v každé fázi musíme projít celé pole a přehazovat prvky, pokud je to nutné:
  - N
- Dohromady je to tedy celkem: $N * log_2(N)$, nic méně rychlost tohoto algoritmu je velmi závislá na volbě pivota.
- Pokud je pivot volen špatně, může i quick sort degradovat na složitost: N^2, ale to nejhorší možný případ.

Složitost - graf



Obrázek : Složitost Hoareho quick sort

Závěr - co jsme se dozvěděli?

Závěr - co jsme se dozvěděli?

- Zopakovali jsme si princip řadícího algoritmu bubble sort, select sort a insert sort.

Závěr - co jsme se dozvěděli?

- Zopakovali jsme si princip řadícího algoritmu bubble sort, select sort a insert sort.
- Vysvětlili jsme si Hoareho variantu quick sort.

Závěr - co jsme se dozvěděli?

- Zopakovali jsme si princip řadícího algoritmu bubble sort, select sort a insert sort.
- Vysvětlili jsme si Hoareho variantu quick sort.
- Ukázali jsme si jeho realizaci v jazyce Java.

Závěr - co jsme se dozvěděli?

- Zopakovali jsme si princip řadícího algoritmu bubble sort, select sort a insert sort.
- Vysvětlili jsme si Hoareho variantu quick sort.
- Ukázali jsme si jeho realizaci v jazyce Java.
- Spočítali jsme jeho složitost.

Reference


KNUTH, Donald Ervin.
Art of Computer Programming, Volume 3: Sorting and Searching.
Reading, Massachusetts: Addison-Wesley, 1998.
ISBN 0-201-89685-0.


Quick Sort: Sorting Algorithm Animation. [online].
[cit. 2013-03-31]. Dostupné z:
http://www.sorting-algorithms.com/quick-sort