Univerzita Karlova
Přírodovědecká fakulta

Základy programování

#45 Setřídění posloupnosti metodou Shaker Sort

David Šklíba 3. ročník BGEKA Jablonec nad Nisou 2021

Zadání

Z textového souboru načtěte vstupní data představovaná reálnými čísly. Proveďte vzestupné/sestupné setřídění této množiny (metoda setřídění vstupním parametrem algoritmu). Setříděnou množinu uložte zpět do textového souboru.

Třídící algoritmy

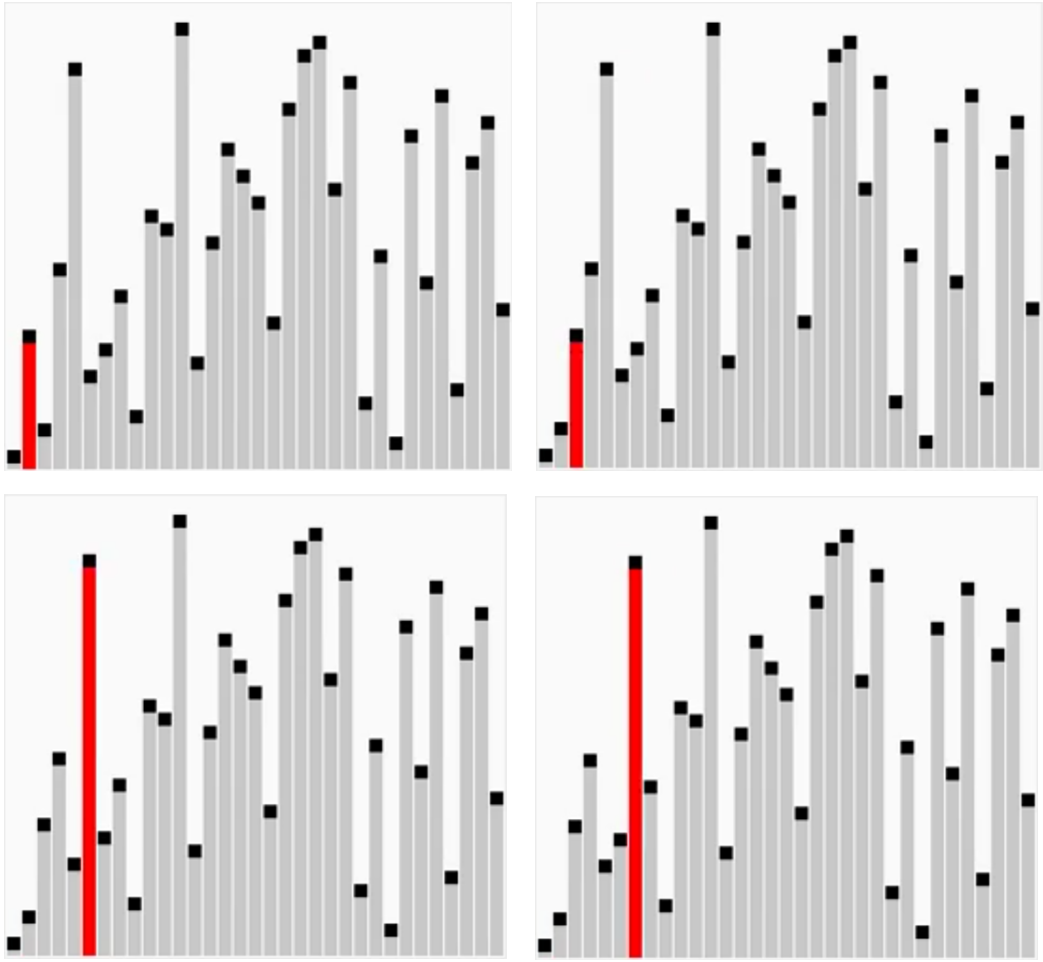
Cílem této skupiny algoritmů je setřídit obsah seznamu či posloupnosti podle stanoveného kritéria. V případě číselné posloupnosti je kritériem seřazení vzestupné nebo sestupné. Rozlišujeme vnitřní a vnější třídící algoritmy. Zatímco u vnitřních třídících algoritmů se prvky tříděné posloupnosti nacházejí uvnitř paměti počítače, u vnějších třídících algoritmů jsou data uložena na nějakém periferním zařízení (Yang a kol. 2011).

Bubble sort
Samotný Shaker sort je vylepšením právě tohoto algoritmu. Jeho název vznikl z představy čísel jako bublinek. Menší číslo symbolizuje lehčí bublinku, která rychleji stoupá k hladině. Dochází k porovnání dvou hodnot, a tedy posouvání té menší. Pokud je první hodnota posloupnosti vlevo menší než hodnota vpravo, vymění si spolu místo. Původně první hodnota posloupnosti si takto vyměňuje místo do té doby, než narazí na menší hodnotu. V ten moment se přesunu ujímá menší hodnota, takto algoritmus pokračuje, dokud hodnota nedosáhne konce posloupnosti, potom se začíná zase od prvního členu.
(Stančík 2015a)

Shaker sort
Algoritmus známý také pod názvy Shake sort nebo Cocktail sort je oboustranný Bubble sort. Dokud nejnižší hodnota nedosáhne konce, algoritmus funguje stejně jako Bubble sort. Po dosažení konce, probíhá řazení opačným směrem. Pokud je hodnota vpravo větší než hodnota vlevo, vzájemně si vymění místo. Dosažením prvního členu posloupností se směr řazení opět obrací. (Stančík 2015b)

Grafické znázornění algoritmu

- řazení vzestupně ve směru zleva doprava



Obr. 1: Ilustrace koktejlového řazení. (Zdroj: https://cs.wikipedia.org/wiki/Koktejlové_řazení)

Pseudokód

```
# Definování funkce load_numbers(file)
# Pokus o otevření vstupního souboru file
# Vytvoření lokální proměnné a_string
# Ukládání obsahu vstupního souboru po řádcích do proměnné a_string
# Ukončení programu, pokud je proměnná a_string prázdná
# Změna datové struktury textového řetězce na seznam v lokální proměnné a_list
# Vytvoření seznamu list_to_be_sorted
# Převedení každého prvku seznamu a_list do datového typu float a uložení do
  seznamu list_to_be_sorted
# Ukončení programu, pokud prvek nelze převést do datového typu float
# Ukončení programu, pokud soubor neexistuje
```

```
#   Ukončení programu, pokud soubor nelze otevřít
#   Vrácení seznamu list_to_be_sorted funkcí
#
#   Definování funkce compare(a, b, finished, is_descendant)
#   Pokud je is_descendant = True
#   Pokud je a > b
#   Funkce vrátí a, b, False
#   V opačném případě funkce vrátí b, a, finished
#   Pokud is_descendant != True
#   Pokud a < b
#   Funkce vrátí a, b, False
#   V opačném případě funkce vrátí b, a, finished
#
#   Definování funkce shaker_sort(list_to_be_sorted, method)
#   Pro i v rozsahu délky seznamu list_to_be_sorted - 1 do 0 s krokem -1
#   Vytvoření lokální proměnné finished = True
#   Vytvoření lokální proměnné is_descendant = False
#   Pokud method = desc
#   is_descendant = True
#   Pro j v rozsahu i
#   list_to_be_sorted[j], list_to_be_sorted[j+1], finished =
    compare(list_to_be_sorted[j+1], list_to_be_sorted[j], finished, is_descendant)
#   Pro j v rozsahu od I do 0 s krokem -1
#   list_to_be_sorted[j-1], list_to_be_sorted[j], finished =
    compare(list_to_be_sorted[j], list_to_be_sorted[j-1], finished, is_descendant)
#   Pokud finished = True
#   Funkce vrátí seznam list_to_be_sorted
#
#   Definování funkce save_sort(sorted_list, file)
#   Vytvoření lokální proměné num_string
#   Pro každé číslo v seznamu sorted_list
#   Převedení čísla do datového typu textového řetezce a přidání mezery za něj
#   Připojení čísla do proměnné num_string
#   Otevření výstupního souboru
#   Uložení proměnné num_string do výstupního souboru
#
#   Vytvoření globální proměnné file = název textového souboru
#   Vytvoření globální proměnné numbers = load_numbers(file)
#   Vytvoření globální proměnné sorted_list = shaker_sort(numbers, “desc”)
#   Spuštění funkce save_sort(sorted_list, název výstupního souboru)
```

Dokumentace

Popis programu
Program načte vstupní textový soubor s číselnou posloupností a ověří jeho validitu. Pomocí algoritmu Shaker sort ji buďto vzestupně nebo sestupně seřadí. Seřazenou posloupnost nakonec uloží do výstupního textového souboru.

Funkce načítající textový soubor
Na začátku se funkce load_numbers(file) pokusí otevřít soubor file s posloupností reálných čísel. V případě, že soubor neexistuje nebo není ve formátu textového souboru, program skončí chybovou hláškou. Obsah souboru načtený ve formátu textového řetězce je následně rozdělen po členech posloupnosti, které jsou uloženy do seznamu a_list. Každé položce tohoto seznamu je změněn datový typ na float, pokud to u nějakého členu není možné, program skončí chybovou hláškou. Nakonec je float objekt uložen do seznamu list_to_be_sorted, který pak funkce vrací.

Funkce porovnávající 2 čísla mezi sebou v závislosti na vstupním parametru
Pokud je argument is_descendant funkce compare(a, b, finished, is_descendant) pravdivý, a > b, tak funkce vrátí a, b, False, jinak vrátí b, a, finished. Pokud argument is_descendant není pravdivý, a < b, tak funkce vrátí a, b, False, v opačném případě vrátí b, a, finished. Argumenty a a b jsou reálná čísla, argument finished je booleovská proměnná značící fázi seřazené posloupnosti, pokud je posloupnost seřazená finished = True. Pokud je parametr is_descendant = True, posloupnost je seřazována sestupně, při jeho jiné hodnotě je seřazování prováděno vzestupně.

Funkce seřazující seznam reálných čísel metodou Shaker sort
Do parametru list_to_be_sorted funkce shaker_sort(list_to_be_sorted, method) vstupuje seznam reálných čísel k seřazení podle metody specifikované v parametru method, pokud method = „desc“, potom je posloupnost seřazována sestupně, při jiných hodnotách parametru method, je posloupnost seřazována vzestupně. Lokální proměnná finished je ze začátku nastavená na hodnotu True, v případě, kdy v iteraci následujícího cyklu nedojde k úplnému seřazení, proměnná finished změní svou hodnotu na False. Funkce vrátí seznam seřazených čísel v momentě, kdy oba vnitřní for cykly neprovedou změnu pořadí prvků, hodnota proměnné finished tedy zůstane True.

Funkce ukládající setříděnou posloupnost do textového souboru
Do funkce save_sort(sorted_list, file) jako první argument vstupuje setříděná posloupnost ve formátu seznamu. Druhý argument určuje, kam bude uložen výstup. Každé číslo je převedeno do datového typu textového řetězce a s přidanou mezerou připojeno do lokální proměnné num_string. Ta je potom uložena do textového souboru file.

Popis vstupních a výstupních dat

Vstupními daty musí být textový soubor obsahující členy posloupnosti oddělené mezerou. Reálná čísla, pro jejichž množinu je program vytvořen, se mohou vyskytovat na více řádcích. Výstupem je potom jednořádkový textový soubor obsahující setříděnou posloupnost čísel oddělených mezerou.

Problematická místa a možnosti vylepšení

Tento algoritmus patří i přes vylepšenou variantu oproti jeho předchůdci mezi nejpomalejší řadící algoritmy, a to kvůli velkému množství zápisů do paměti. Rychlejších výsledků dosahuje při větším částečném setřídění vstupního souboru. Časová složitost je $O(N^2)$, nelze tedy použít pro rozsáhlá data. Kvůli těmto nevýhodám se v praxi nevyužívá na rozdíl od skupiny algoritmů, do které patří Quick sort, Merge sort nebo Heap sort.

Seznam literatury

STANČÍK, P. (2015a): Bubble sort. https://www.algoritmy.net/article/3/Bubble-sort (6. 2. 2021).

STANČÍK, P. (2015b): Shaker sort. https://www.algoritmy.net/article/40270/Shaker-sort (6. 2. 2021).

YANG, Y., YU, P., GAN, Y. (2011): Experimental study on the five sort algorithms. In: MACE 2011 (ed.): Second International Conference on Mechanic Automation and Control Engineering. Inner Mongolia, China, 1314–1317.