Programátorská dokumentace

Jakub Endlicher

18. února 2025

Obsah

1 Úvod

Tento program slouží jako dokumentace k programu **AlgoVis** (Algorithm Visualizer), který vizualizuje, jak pracují třídící algoritmy *bubble sort*, *selection sort* a *insertion sort*.

2 Uživatelská dokumentace

2.1 Použití programu

Po instalaci a spuštění programu uživatel vidí 2 zaškrtávací okénka a 2 tlačítka.
Ihned po spuštění může uživatel zmáčknout tlačítko *Start sorting*, které spustí třídící algoritmy a ihned začne třídit náhodně vygenerovaná data, kde třídící algoritmy budou spuštěny postupně jeden po druhém.

Pokud uživatel chce vložit vlastní data k třídení, stačí zakliknout okénko *Data from file* a v otevřeném okně vybrat textovou složku, která obsahuje hodnoty k setřídění ve formátu hodnot oddělených mezerou.

Program také dává uživateli možnost spustit třídící algoritmy naráz pomocí tlačítka *Run async*, kde bude třízení probíhat v jednu chvíli u všech tří algoritmů.

Poslední tlačítko uživateli ukáže graf všech tří algoritmů a čas, jak dlouho jednotlivým algoritmům trvá setřídít určitý dataset.

2.2 Vysvětlení změřených metrik

Změřené metriky se u každého algoritmu nachází v levém horním rohu. Konkrétně se jedná o počet porovnání a počet vyměnění dvou prvků.

Obě tyto metriky jsou velmi důležíté, jelikož časová složitost obou je konstatní, tak můžeme brát jejich součet jako celkovou časovou složitost, kde nižší číslo je lepší.

Na co je třeba dávat si pozor je, že při spuštění naráz to vypadá, že *bubble sort* algoritmus je rychlejší než *selection sort* algoritmus. Avšak není tomu tak, jde pouze o implementační rozohdnutí, díky kterému to takto vypadá.

Pokud uživatel spustí funkci na vykreslení grafu časů, vidí, který algoritmus je nejrychlejší. Zde se bere v potaz jediná metrika, a to reálný změřený čas trvání jednotlivých algoritmů.

2.3 Formát vstupních dat

Data je nutné uložit do textové složky s koncovkou .txt, ve které budou uložena jednotlivá čísla oddělená libovolným počtem mezer, případně nových řádek. Tento soubor může být uložen kdekoli na disku uživatele.

3 Programátorská dokumentace

3.1 Anotace

Vytvoření programu na vizualizaci průběhu třídících algoritmů. Program vygeneruje přípdně načte nesetříděná data, ty setřídí a během tohoto třídění bude zobrazovat každý jednotlivý krok. V průběhu třízení bude rovněž zobrazovat metriku počtu výměn prvků a počtu porovnání prvků.

3.2 Využité knihovny

- **Tkinter** - Tato knihovna je využita pro implementaci grafického rozhraní jak na vykreslování třízení, tak na ovládání programu uživatelem.
- **Random a Copy** - Tyto knihovny jsou využity pro práci s daty, kde konkrétně random je využito pro generování náhodných vstupních dat a copy se využívá ay bylo možné dát stejný dataset všem algoritmům
- **Asyncio** - Tato knihovna se využívá pro asynchronní spuštění programu.
- **Matplotlib a Time** - Tyto knihovny slouží k vytvoření grafu porovnání všech tří algoritmů. Time se využívá na meření času jako takového a Matplotlib k vytvoření grafu.

3.3 Stažení a spuštění programu

3.3.1 Stažení programu

1. Otevřete terminál nebo příkazový řádek.
2. Klonujte repozitář z GitHubu pomocí následujícího příkazu:

```
git clone https://github.com/Skelda/Zapoctak
```

3. Přejděte do adresáře projektu:

```
cd Zapoctak
```

4. Nainstalujte potřebné závislosti pomocí následujícího příkazu:

```
pip install -r requirements.txt
```

3.3.2 Spuštění programu

1. Spusťte hlavní skript pro spuštění aplikace:

```
python main.py
```

2. V okně aplikace vyberte, zda chcete spustit třídicí algoritmy současně (zaškrtněte políčko "Run async") nebo jeden po druhém.
3. Vyberte, zda chcete importovat data z vašeho počítače (zaškrtněte políčko "Data from file") nebo ne, v takovém případě budou data náhodně generována.
4. Spusťte třídění kliknutím na tlačítko "Start sorting".
5. Tlačítko "Show Sorting Times"zobrazí graf porovnávající výkon všech tří třídicích algoritmů. Algoritmy budou třídit pole náhodných dat o velikosti od 100 do 1000 prvků. Poznámka: Tato data nejsou spojena s daty použitými ve vizualizátoru a budou vždy náhodně generována.

3.4 Struktura programu

3.4.1 Hlavní soubor

main.py - Hlavní skript pro spuštění aplikace.

- **initializeScreen()** - Inicializuje okno Tkinter a spustí hlavní funkci.
- **main()** - Hlavní funkce pro spuštění vizualizací třídicích algoritmů.
- **start_sorting()** - Spustí proces třídění na základě výběru uživatele.
- **show_sorting_times()** - Zobrazí graf časů třídění.

3.4.2 Adresář Sorting_algorithms

Sorting_algorithms - Adresář obsahující implementace třídicích algoritmů a funkce pro vizualizaci.

- **bubble.py** - Obsahuje implementaci a vizualizaci Bubble Sort.
  - **bubbleSortStep()** - Provede jeden krok algoritmu Bubble Sort.
  - **drawBubble()** - Kreslí vizualizaci Bubble Sort.
  - **drawRectangle()** - Kreslí obdélník reprezentující prvek v seznamu.
- **selection.py** - Obsahuje implementaci a vizualizaci Selection Sort.
  - **selectionSortStep()** - Provede jeden krok algoritmu Selection Sort.
  - **drawSelection()** - Kreslí vizualizaci Selection Sort.

  - **drawRectangle()** - Kreslí obdélník reprezentující prvek v seznamu.

- **insertion.py** - Obsahuje implementaci a vizualizaci Insertion Sort.
  - **insertionSortStep()** - Provede jeden krok algoritmu Insertion Sort.
  - **drawInsertion()** - Kreslí vizualizaci Insertion Sort.
  - **drawRectangle()** - Kreslí obdélník reprezentující prvek v seznamu.
- **__init__.py** - Inicializuje modul Sorting_algorithms.

3.4.3 Soubory pro spuštění

run_normal.py - Skript pro spuštění vizualizací třídicích algoritmů synchronně.

- **main()** - Hlavní funkce pro spuštění vizualizací třídicích algoritmů synchronně.
- **bubble_loop()** - Vizualizuje algoritmus Bubble Sort.
- **selection_loop()** - Vizualizuje algoritmus Selection Sort.
- **insertion_loop()** - Vizualizuje algoritmus Insertion Sort.

run_async.py - Skript pro spuštění vizualizací třídicích algoritmů asynchronně.

- **startAsyncMain()** - Spustí hlavní funkci pomocí asyncio.
- **main()** - Hlavní funkce pro spuštění vizualizací třídicích algoritmů asynchronně.
- **bubble_loop()** - Vizualizuje algoritmus Bubble Sort asynchronně.
- **selection_loop()** - Vizualizuje algoritmus Selection Sort asynchronně.
- **insertion_loop()** - Vizualizuje algoritmus Insertion Sort asynchronně.

3.4.4 Skript pro měření rychlosti

time_actual_sorting.py - Skript pro měření a vykreslení výkonu třídicích algoritmů.

- **bubbleSort()** - Třídí data pomocí algoritmu Bubble Sort.
- **selectionSort()** - Třídí data pomocí algoritmu Selection Sort.
- **insertionSort()** - Třídí data pomocí algoritmu Insertion Sort.
- **sorting_times()** - Měří časy třídění pro různé algoritmy.
- **plot_sorting_times()** - Vykresluje časy třídění pro různé algoritmy.