Projektowanie Algorytmów i Metody Sztucznej Inteligencji			
Autor: Kacper Brodziak Nr indeksu: 258978	Grupa: Y03-51f	Rok ak: 2	Semestr: 4
	Termin: 13:15 – 15:00	Oddano: 17.05.2022	Ocena:
Nr projektu: 2	Temat: Algorytmy sortowania		

1. Wstęp

Tematem projektu z jakim przyszło nam się zmierzyć są algorytmy sortowania. W zależności od wyboru poziomu skomplikowania zadania musiałem zdecydować się na selekcję trzech z czterech dostępnych algorytmów sortowania. W moim przypadku wybór padł na sortowania: przez scalanie, quicksort oraz kubełkowe. Po wyborze sortować należało zaimplementować algorytmy oraz przeprowadzić analizę efektywności na podanym zborze danych. Zbiorem danych, na którym pracujemy jest okrojona baza filmów „IMDb Largest Review Dataset" z kaggle.com. Plik zawiera tytuł oraz ocenę filmu. Celem zadania projektowego jest przygotowanie danych zawierających odpowiednio: 10 000, 100 000, 500 000, 1 000 000, maksymalną ilość danych z pliku. Następnie na każdym zestawie przeprowadzić analizę efektywności sortowania z wykorzystaniem zaimplementowanych algorytmów, dodatkowo dla każdego zestawy musimy podać czas sortowania, średnią wartość oraz medianę rankingu.

2. Sortowanie przez scalanie

Sortowanie przez scalanie (ang. merge sort) jest to rekurencyjny algorytm sortowania danych, który koncentruje się tylko i wyłącznie na metodzie „dziel i zwyciężaj". W każdym kroku działania tablica zostaje podzielona na dwie części, do momentu powstania jednoelementowych. Ponieważ zbiór jednoelementowy jest już posortowany to możemy przejść od scalania. Po wystąpieniu przypadku bazowego algorytm porównuje dwie tablice, po czym je scala, powodując ułożenie elementów w odpowiedniej kolejności. Złożoność algorytmu wynosi n*log(n), więc jest wydajniejszy od chociażby sortowania bąbelkowego, którego złożoność jest kwadratowa. Zaletami algorytmu są m.in.: prostota implementacji, wydajność, stabilność, czas sortowania jest proporcjonalny do złożoności bez względu na rodzaj danych na wejściu. Jedyną zauważalną wadą algorytmu jest potrzeba posiadania dodatkowego obszaru pamięci przechowującego kopiec podtablic do scalania.

3. Sortowanie szybkie (quicksort)

Sortowanie szybkie podobnie jak merge sort jest algorytmem rekurencyjnym. W każdym kolejnym kroku sortowania zostaje wybrany jeden element nazwany pivotem, służący do podziału tablicy. Algorytm porównuje elementy tablicy z wybranym punktem odniesienia i tworzy 2 nowe tablice, zawierające mniejsze i większe elementy. Następnie pivot zostawiamy, ponieważ znajduje się on już na swojej ostatecznej pozycji. Następnie algorytm powtarza kroki do momentu posortowania całości tablicy. Algorytm został wynaleziony w 1960 roku przez Sir Chales Antony Richard Hoare. Quicksort jest powszechnie spotykany i wdrażany do systemów informatycznych ze względu na olbrzymią ilość zalet jakie ze sobą niesie, m.in.:

- złożoność czasowa jest rzędu O(nlog(n))

- przeciwnie do sortowania przez scalanie, algorytm nie potrzebuje dodatkowej tablicy

- łatwość w implementacji, dobrze z różnymi typami danych

4. Sortowanie kubełkowe

Sortowanie kubełkowe (ang. bucket sort) jest najczęściej stosowanym algorytmem sortowania w przypadku, gdy liczby w zadanym przedziale są rozłożone jednostajnie, ma on wówczas złożoność O(n). W przypadku ogólnym pesymistyczna złożoność obliczeniowa tego algorytmu wynosi O(n^2). Idea działania algorytmu rozpoczyna się od podzielenia zadanego przedziału liczb na n podprzedziałów (tzw. Kubełków) o równej długości. Następnie należy przypisać liczby z sortowanej tablicy do odpowiednich kubełków. Kolejnym etapem jest sortowanie liczby w niepustych kubełkach. Ostatnim etapem jest wypisanie po kolei zawartości niepustych kubełków. Zalety algorytmu:

- złożoność czasowa rzędu O(n)

- algorytm nie potrzebuje dodatkowych tablic

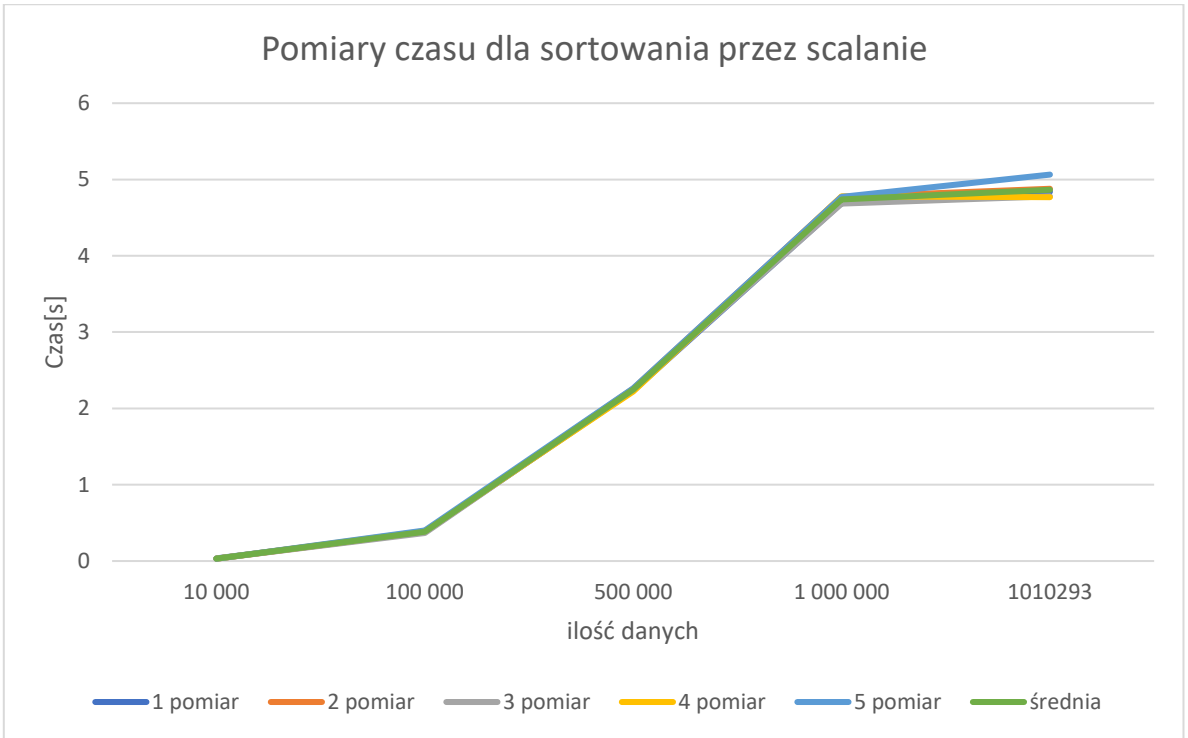
- jest łatwy w implementacji

W przypadku sortowania kubełkowego należy pamiętać o dwóch ważnych wadach tego rodzaju sortowania. Wady te skupiają się na konieczności równomiernego rozłożenia danych, aby złożoność była liniowa oraz problemem jest konieczność dokładnej znajomości rozstępu zbioru.

5. Wyniki

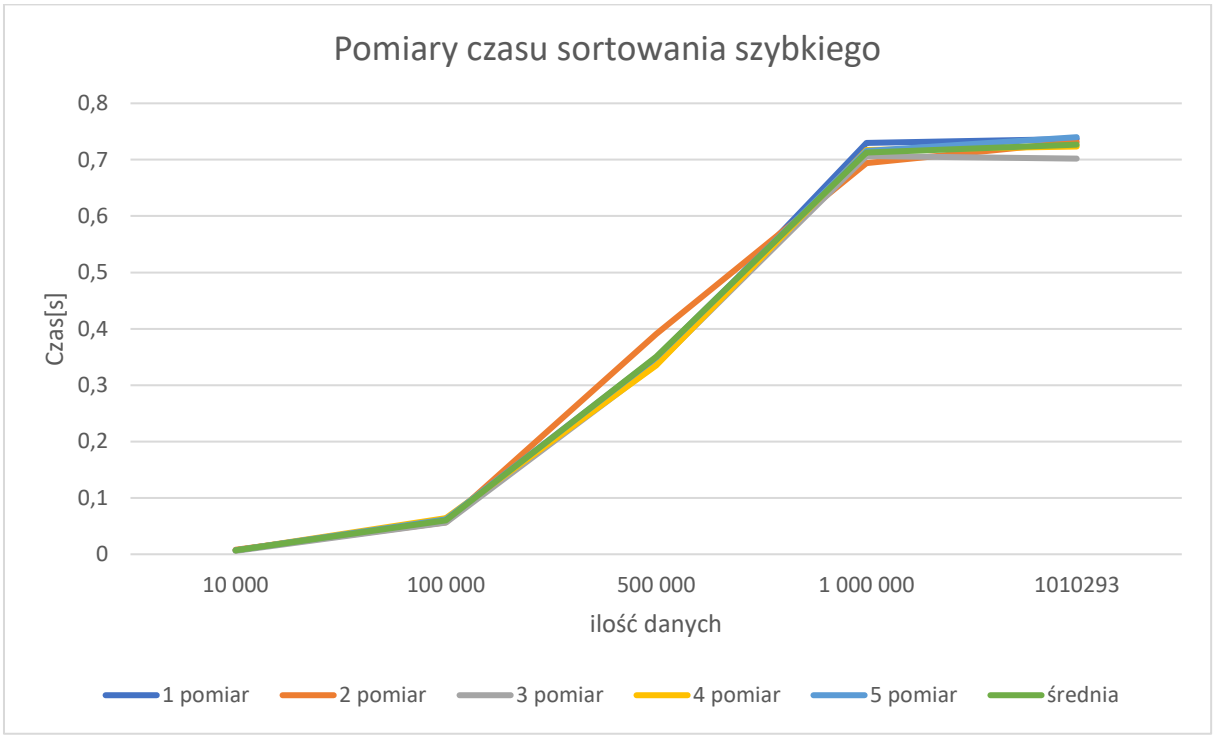
a) Sortowanie przez scalanie

Sortowanie przez scalanie - czas sortowania dla 5 pomiarów					
ilość danych	10 000	100 000	500 000	1 000 000	1010293
czas1 [s]	0,035005	0,381072	2,235273	4,691209793	4,83835
czas2 [s]	0,030503	0,379085	2,260615	4,77039814	4,878474
czas3 [s]	0,032018	0,368082	2,249329	4,682766438	4,779006
czas4 [s]	0,031	0,39271	2,21951	4,778953552	4,76593
czas5 [s]	0,030001	0,40175	2,265085	4,77285552	5,063882
średnia ocena	5,460446	6,064038	6,65528	6,634075145	6,636372
mediana	5	6	7	7	7



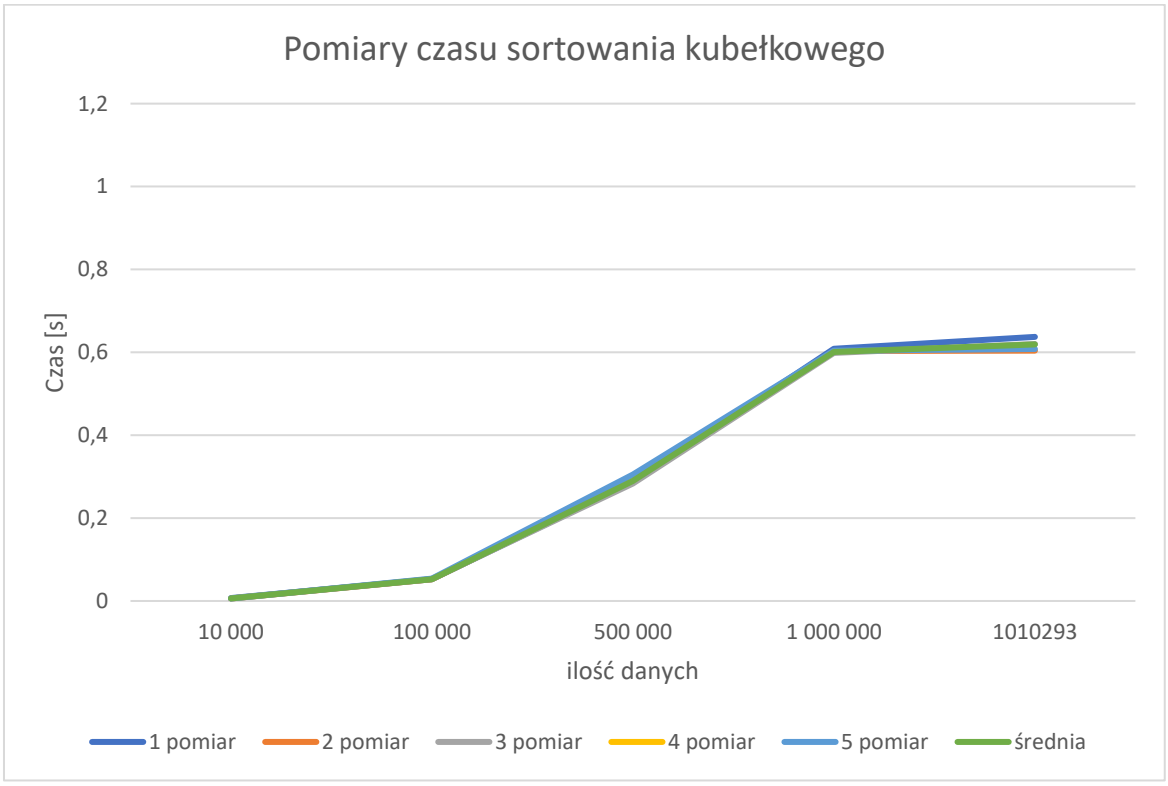
b) Sortowanie szybkie

Sortowanie szybkie - czas sortowania dla 5 pomiarów					
ilość danych	10 000	100 000	500 000	1 000 000	1010293
czas1 [s]	0,00701332	0,061014	0,3391571	0,729586363	0,73677206
czas2 [s]	0,00801945	0,057007	0,3907342	0,694180727	0,7315197
czas3 [s]	0,00600076	0,056014	0,336972	0,706488132	0,7019453
czas4 [s]	0,00698781	0,064017	0,3351457	0,717518806	0,72337532
czas5 [s]	0,00701475	0,061697	0,3490915	0,715422153	0,73992658
średnia ocena	5,46044604	6,064038	6,6552797	6,634075145	6,63637195
mediana	5	6	7	7	7



c) Sortowanie kubełkowe

Sortowanie kubełkowe - czas sortowania dla 5 pomiarów					
ilość danych	10 000	100 000	500 000	1 000 000	1010293
czas1 [s]	0,007004	0,052017	0,286587	0,6081371	0,637162
czas2 [s]	0,006001	0,052023	0,289427	0,6030421	0,604028
czas3 [s]	0,006001	0,053012	0,283067	0,5985234	0,62002
czas4 [s]	0,00599	0,052005	0,282559	0,5912225	0,625796
czas5 [s]	0,006021	0,054012	0,305144	0,6038139	0,607928
średnia ocena	5,460446	6,064038	6,65528	6,6340751	6,636372
mediana	5	6	7	7	7



6. Podsumowanie

Zaimplementowanie algorytmy bardzo sprawnie radzą sobie z sortowaniem dużych ilości danych. Natomiast widoczne jest, że sortowanie przez scalanie działa znacznie wolniej od 2 pozostałych sortowań (ok. 6,7 razy wolniej od sortowania szybkiego). Dla małych baz danych nie robi to dużej różnicy, lecz jeśli mamy do przesortowania miliony danych to już staje się do dla nas ogromnym problemem.

7. Literatura

https://en.wikipedia.org/wiki/Quicksort

https://en.wikipedia.org/wiki/Merge_sort

https://en.wikipedia.org/wiki/Bucket_sort

8. Odnośnik do repozytorium

https://github.com/brodzi4k/desktop-tutorial