Politechnika Wrocławska

Projekt 1

Sortowanie
Projektowanie i Analiza Algorytmów

1. Wstęp

Sortowanie danych jest kluczowym zagadnieniem w informatyce, wykorzystywanym do uporządkowania elementów w zbiorze w określonej kolejności. Wśród wielu algorytmów sortowania trzy z nich wyróżniają się zarówno ze względu na swoją popularność, jak i efektywność w różnych sytuacjach.

2. Opis badanych algorytmów

Merge Sort:

Merge Sort to algorytm sortowania dziel i zwyciężaj. Działa on poprzez podział listy na mniejsze części, posortowanie ich, a następnie scalenie posortowanych podlist w celu uzyskania całkowicie posortowanej listy.

Złożoność obliczeniowa:

- Dla przypadku średniego i najgorszego:
  - Czas: O(n log n)
  - Pamięć: O(n)

Quick Sort:

Quick Sort to inny algorytm sortowania dziel i zwyciężaj, który działa poprzez wybór elementu rozdzielającego (pivot), a następnie umieszczanie elementów mniejszych od pivota po lewej stronie, a większych po prawej. Następnie ten sam proces jest stosowany rekurencyjnie do lewej i prawej części tablicy.

Złożoność obliczeniowa:

- Dla przypadku średniego:
  - Czas: O(n log n)
  - Pamięć: O(log n)
- Dla najgorszego przypadku (gdy pivot jest zawsze skrajnym elementem lub tablica jest już posortowana):
  - Czas: O(n^2)
  - Pamięć: O(log n)

Bucket Sort:

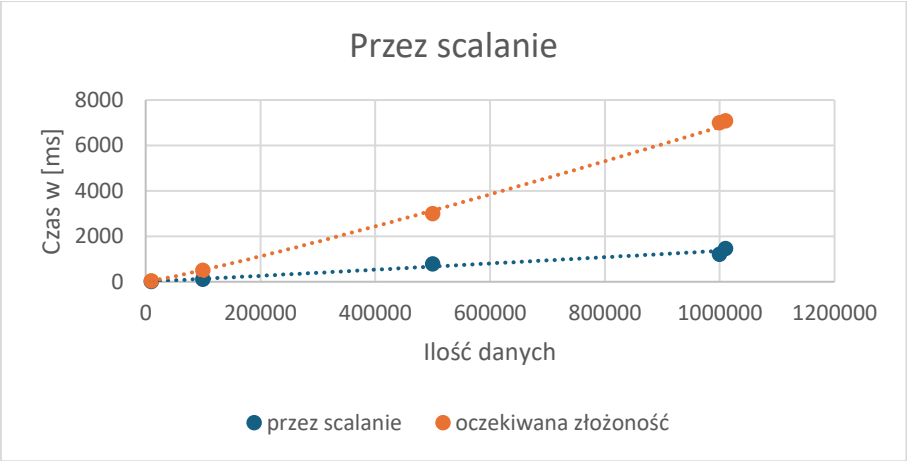
Bucket Sort to algorytm sortowania, który sortuje elementy poprzez podzielenie ich na "kubełki", a następnie posortowanie każdego kubełka osobno. Jest to skuteczne, gdy elementy są równomiernie rozłożone na całym zakresie.

Złożoność obliczeniowa:

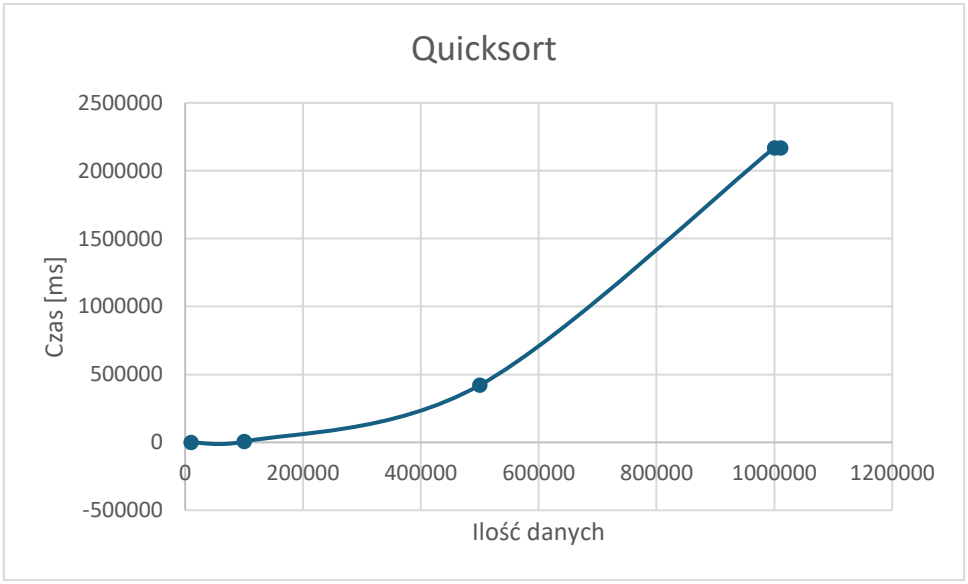
- Dla przypadku średniego:
  - Czas: O(n + k), gdzie k to liczba kubełków
  - Pamięć: O(n + k)
- Dla najgorszego przypadku (gdy wszystkie elementy trafią do jednego kubełka):
  - Czas: O(n^2)
  - Pamięć: O(n + k)

3. Badania

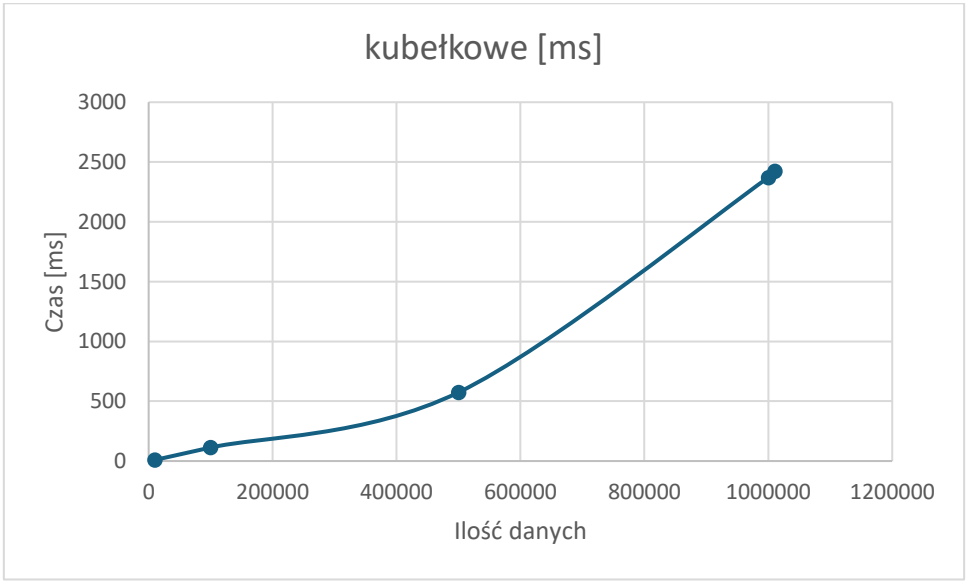
Ilość danych	przez scalanie [ms]	oczekiwana złożoność [ms]
10000	13	40
100000	108	500
500000	789	3 000
1000000	1212	7 000
1010292	1458	7 081,543



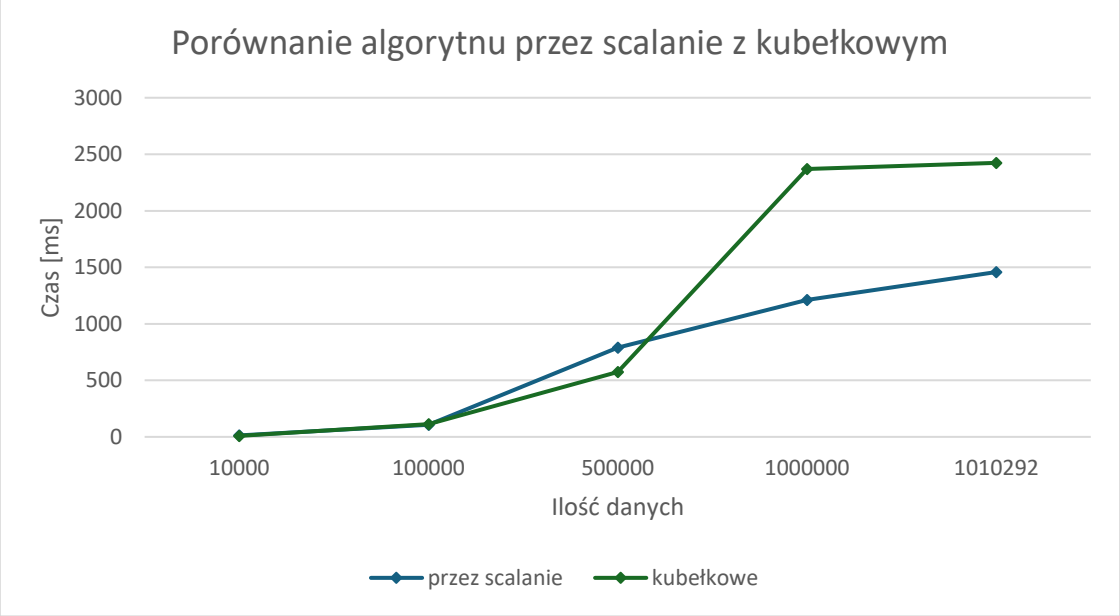
Ilość danych	quicksort [ms]
10000	64
100000	5433
500000	421000
1000000	2169000
1010292	2168000



Ilość danych	kubełkowe [ms]
10000	9
100000	113
500000	574
1000000	2370
1010292	2424



	Czas sortowania [ms]				
Ilość danych	przez scalanie	quicksort	kubełkowe	Mediana	Średnia
10000	13	64	9	5	5,46045
100000	108	5433	113	6	6,06404
500000	789	421000	574	7	6,6553
1000000	1212	2169000	2370	7	6,63408
1010292	1458	2168000	2424	7	6,63638



4. Wnioski

Podczas analizy algorytmów sortowania, takich jak Merge Sort i Bucket Sort, zwrócono uwagę na ich efektywność i złożoność obliczeniową. Zarówno Merge Sort, wykorzystujący podejście "dziel i zwyciężaj", jak i Bucket Sort, sortujący elementy poprzez podział na "kubełki", wykazują stabilną wydajność w różnych przypadkach.
Merge Sort, ze złożonością czasową O(n log n) dla przypadku średniego i najgorszego, oraz złożonością pamięciową O(n), wydaje się być szczególnie przydatny w przypadku dużej ilości danych do posortowania, zachowując jednocześnie stabilność wydajności.
Bucket Sort, charakteryzujący się czasem działania O(n + k) dla przypadku średniego, gdzie k to liczba kubełków, może być wydajną opcją, szczególnie gdy dane są równomiernie rozłożone na całym zakresie.
Jednakże podczas badania Quick Sorta, zaobserwowano potencjalne błędy pomiarów, które mogą wynikać z kilku czynników:

- **Wybór pivotu**: Mimo że pivot wybierany był na środku przy względnie ułożonych danych mogło to doprowadzić do przypadku skrajnego w którym złożoność obliczeniowa wynosi O(n^2).
- **Charakterystyka danych**
- **Implementacja algorytmu**: Błędne wyniki mogą wynikać również z niedokładnej lub nieoptymalnej implementacji samego algorytmu Quick Sorta.