Współczesne technologie programowania

Sprawozdanie

Analiza złożoności obliczeniowej i profilowanie algorytmów

Adam Bemski 22061

Wojciech Pełka 220090

1 Cel zadania

Celem zadania jest dokonanie analizy złożoności obliczeniowej algorytmu QuickSort, na podstawie różnych zbiorów danych dla przypadku optymistycznego i pesymistycznego.

Na zadanie składa się :

- Przygotowanie programu implementującego algorytm szybkiego sortowania, który posłuży jako narzędzie do eksperymentalnej analizy złożoności.
- Analityczna ocena złożoności obliczeniowej wraz z uzasadnieniem doboru zbiorów testowych

2 Wstęp teoretyczny

Algorytm sortowania szybkiego jest oparty o zasadę „Dziel i zwyciężaj”. Zasada ta polega na tym, że problem zostaje podzielony na mniejsze podproblemy (Dziel) a następnie problemy te zostają rozwiazywane (Zwyciężaj).

Algorytm QuickSort działa następująco:

1. Zbiór testowy dzielimy na dwie części zwanymi lewą i prawą partycją. Podziału dokonujemy wybierając element ze zbioru zwany piwotem.
2. Do lewej partycji przenoszone są elementy nie większe od wybranego piwota a do prawej partycji przenoszone są elementy nie mniejsze od wybranego piwota.
3. Następnie wykonuje się ten sam algorytm dla części lewej i prawej.
4. Rekurencja kończy się, gdy kolejny fragment uzyskany z podziału zawiera pojedynczy element.

Algorytm sortowania szybkiego jest uznawany za najszybszy algorytm sortujący – jego złożoność obliczeniowa dla przypadku optymistycznym sięga rzędu $O(n \log n)$. Należy mieć jednak na uwadze, że dla przypadku pesymistycznego złożoność algorytmu może spaść nawet do złożoności $O(n^2)$.

Z wersją optymistyczną mamy do czynienia, gdy za każdym razem jako element rodzielający wybrana zostaje mediana z sortowanego aktualnie fragmentu tablicy, czyli gdy każdy podział daje równe podzbiory danych. Wówczas równianie rekurencyjne wygląda nasępująco

$$T(n) = 2T(\frac{n}{2}) + O(n)$$

Co daje nam złożonośc obliczeniową:

$$T(n) = O(n \log n)$$

Z wersją pesymistyczną mamy do czynienia wtedy, gdy zbiór testowy jest posortowany rosnąco lub malejąco. Objawia się to wtedy każdorazowym wyborem elementu najmniejszego, bądź największego w sortowanym fragmencie zbioru. Wówczas równianie rekurencyjne wygląda nasępująco:

$$T(n) = T(n-1) + O(n) = \sum_{k=1}^{n} O(k) = O(\sum_{k=1}^{n} k)$$

Co daje nam złożonośc obliczeniową:

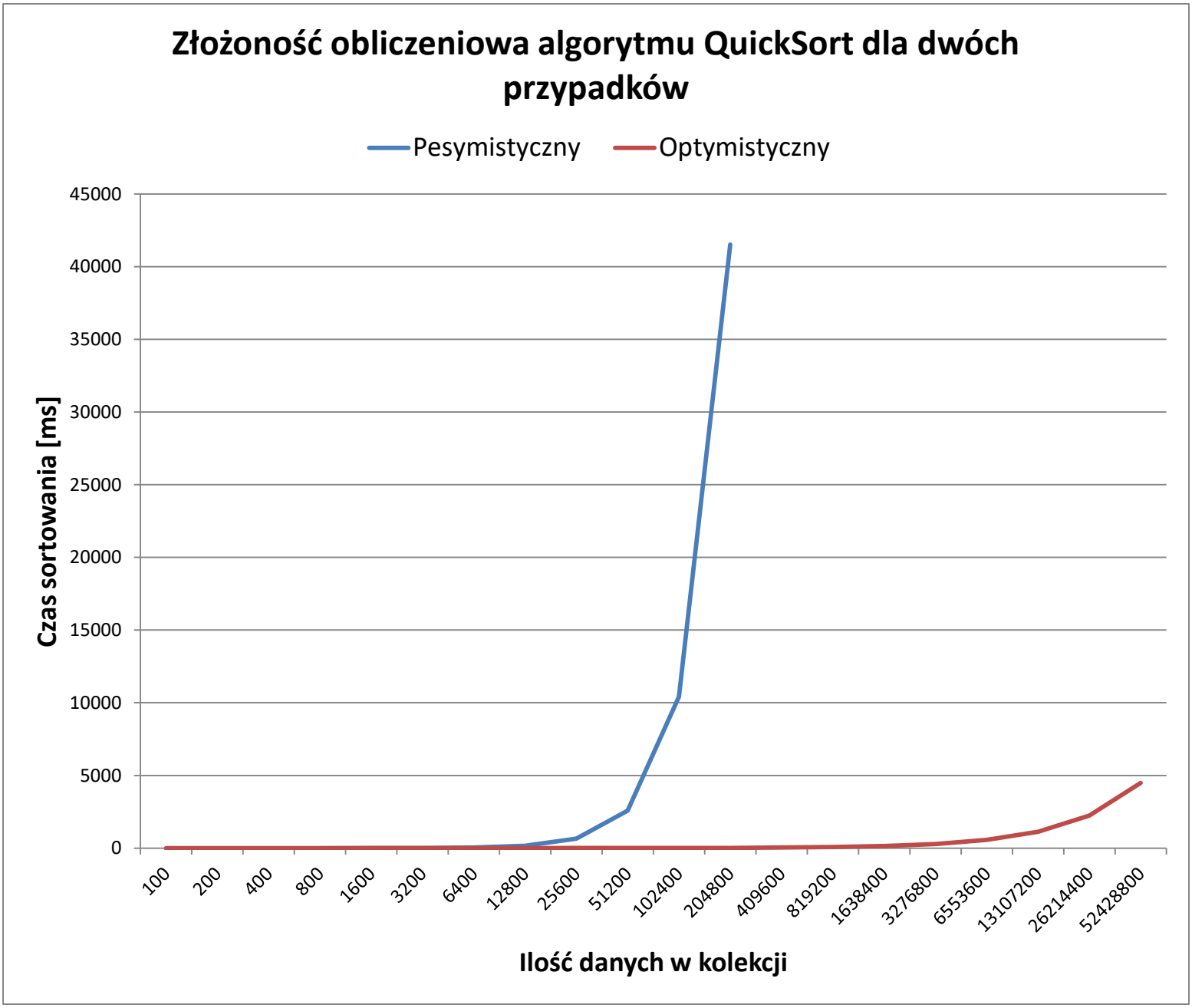
$$T(n) = O(n^2)$$

3 Wyniki

Dla przypadku optymistycznego dane generowane są za pomocą funkcji pseudolosującej. Dla przypadku pesymistyczneg, w zbiorze testowym zostały umieszczone dane już posortowane. Ilość danych w obu typach zbiorów jest każdorazowo zwiększania w celu wyznaczenia czasu sortowania w zależności od liczności zbioru. Niestety ilość danych w zbiorze pesymistycznym musiała zostać ograniczona do około 200 tysięcy, ze względu na bardzo długi czas wykonywania sortowania.

Dane testowe

Ilość danych	**Czas w milisekundach**	
	Przypadek optymistyczny	**Przypadek pesymistyczny**
100	0	0
200	0	0
400	0	0
800	0	0
1600	0	3
3200	0	10
6400	0	41
12800	1	162
25600	2	647
51200	4	2585
102400	9	10415
204800	17	41517
409600	36	
819200	74	
1638400	144	
3276800	285	
6553600	561	
52428800	4482	



4 Wnioski

Zgodnie z założeniami czas sortowania wariantu pesymistycznego jest znacznie większy od czasu wariantu optymistycznego. Na powyższym wykresie można zauażyć, że im większy rozmiar tablicy sortowanej tym różnica między czasem sortowania przypadku pesymistycznego a czasem wariantu optymistycznego jest znacząco większa.