Badanie złożoności obliczeniowej sortowania szybkiego w przypadku pesymistycznym

Szymon Leśniak

28 maja 2014

1 Wstęp

W algorytmie sortowania szybkiego (*quicksort*) wybiera się jeden z elementów tablicy (nazywany *pivot*), który jest umieszczany na właściwe jemu miejsce. Na powstałych po jego bokach podtablicach wywołuje się rekurencyjnie ten algorytm. W średnim przypadku jego złożoność wynosi $O(n \log n)$, jednak działa szybciej od innych algorytmów tego samego rzędu.

Największy wpływ na czas wykonania algorytmu ma każdorazowo wybór elementu pivot. W najstarszych implementacjach każdorazowo był nim skrajny element tablicy. Powoduje to jednak, że przy częściowo lub całkowicie posortowanej tablicy złożoność dąży do $O(n^2)$. Obranie na stałe innego pivota powoduje, że do wywołania przypadku pesymistycznego trzeba innego rozłożenia elementów.

Znalezienie tego rozłożenia jest jednak możliwe, co otwiera drogę do przeciążenia komputerów podczas złośliwych ataków. Zaproponowaną poniżej metodą rozwiązania problemu jest losowy wybór pivota, niezależny dla każdego wywołania algorytmu *quicksort*. Możliwe jest także zastosowanie hybrydowych algorytmów sortowania.

2 Opis badania

Sortowanym pojemnikiem użytym w badaniu był `vector` liczb typu `int`, pochodzący z STL C++. Implementacji dokonano przy użyciu systemu operacyjnego Ubuntu 13.10 oraz kompilatora g++ 4.8.1.

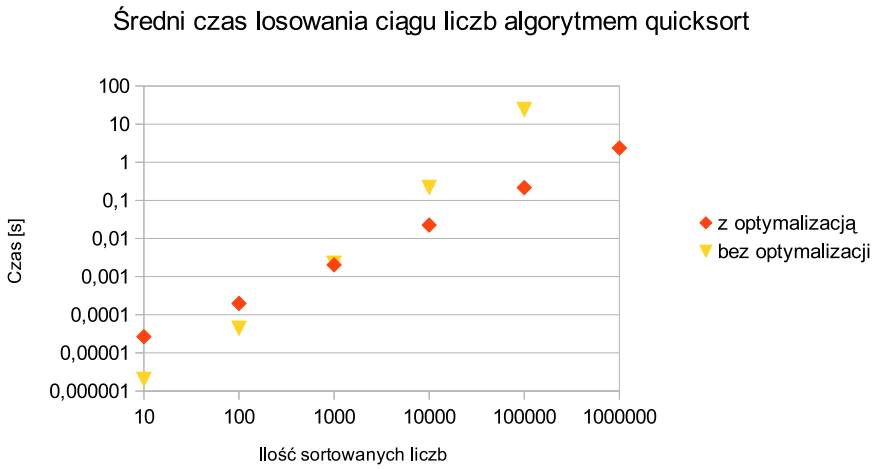
Przeprowadzone zostały dwie serie badań: dla stałego i ruchomego elementu pivot. Realizację przypadku pesymistycznego zapewniono przez nakazanie sortowania już posortowanych tablic. W tym celu pliki ze wzorcowo posortowanymi tablicami wykorzystano dwukrotnie: zarówno jako źródło, jak i wzór. Wynik był średnią z czasu wielokrotnego sortowania.

3 Wyniki badania

Wyniki są przedstawione w poniżej tabeli oraz na wykresie (skala log-log)

Tablica 1: Czasy działania algorytmów sortowania (w sekundach)

Ilość liczb	**Ilość próbek**	**z optymalizacją**	**bez optymalizacji**
10	500	$2{,}66{\cdot}10^{-5}$	$2{,}03{\cdot}10^{-6}$
100	100	0,000198192	$4{,}43{\cdot}10^{-5}$
1000	50	0,00203221	0,00224721
10000	25	0,0225887	0,219349
100000	15	0,216304	24,32
1000000	5	2,3648	



4 Wnioski

Zadany ciąg liczb rzeczywiście sprowadzał algorytm *quicksort* do przypadku pesymistycznego – przy stałym pivocie zależność czasu działania od wielkości problemu jest wyraźnie kwadratowa.

Teoretycznej, oczekiwanej zależności nie udało się uzyskać dla algorytmu z optymalizacją. Ewidentną tego przyczyną jest czas potrzebny na losowanie liczby. Tłumaczy to fakt, że odchylenie od wykresu typu $n \log n$ jest największe dla najmniejszego zestawu danych.