Algorytmy i struktury danych - Sortowanie

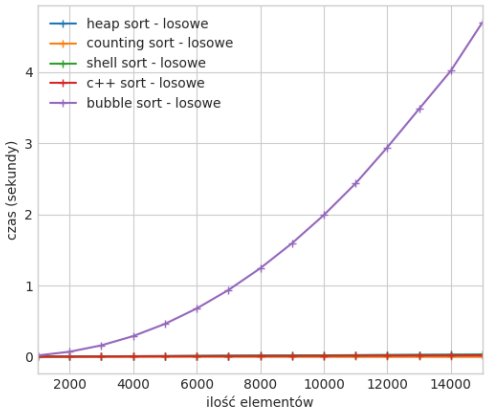
Dariusz Max Adamski

Poznań, 20.03.2018

W tym sprawozdaniu porównywana będzie efektywność różnych algorytmów sortujących, w odniesieniu do ich złożoności obliczeniowej, zużycia pamięci i czasu wykonywania. Brane pod uwagę będą także przypadki krańcowe, czyli dane wejściowe, dla których algorytm może mieć zdecydowanie dłuższy czas wykonywania, niż dla ciągu złożonego z liczb losowych, generowanych zgodnie z równomiernym rozkładem prawdopodobieństwa.

Dla odniesienia, na wykresach zostały także umieszczone czasy dla algorytmu sortowania z biblioteki standardowej języka C++.

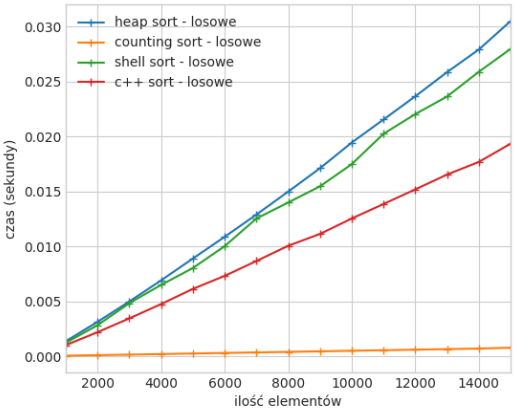
Pierwszymi porównywanymi algorytmami będą sortowanie bąbelkowe (BS), sortowanie stogowe (HS), sortowanie przez zliczanie (CS) oraz sortowanie Shella (ShS). Czasy wykonywania są przedstawione na wykresach 1, 2 i 3.



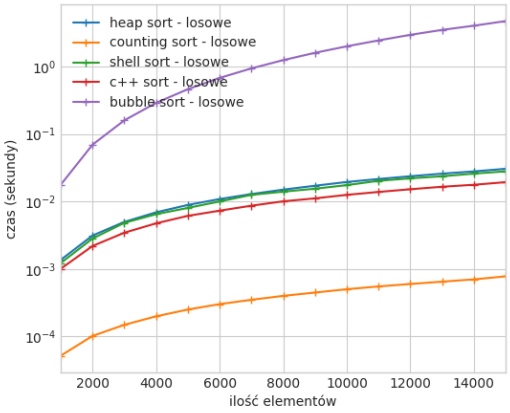
Rysunek 1: Czas sortowania losowych danych

Sortowanie bąbelkowe to algorytm o złożoności obliczeniowej $O(n^2)$ [1]. Dla danych losowych, w porównaniu z algorytmami HS, CS i ShS, sortowanie bąbelkowe zajmuje najwięcej czasu. Ponieważ, przy tablich o liczbie elementów większej niż 1000, skala czasu bardzo się wydłuża, algorytmy o złożoności $O(n \log n)$ zostały porównane na wykresie 2.

Algorytmy sortowania stogowego, przez zliczanie i Shella to algorytmy o złożoności obliczeniowej $O(n \log n)$

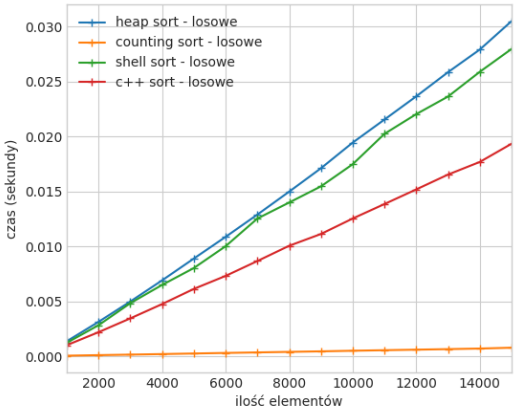


Rysunek 2: Czas sortowania losowych danych



Rysunek 3: Czas sortowania losowych danych ()

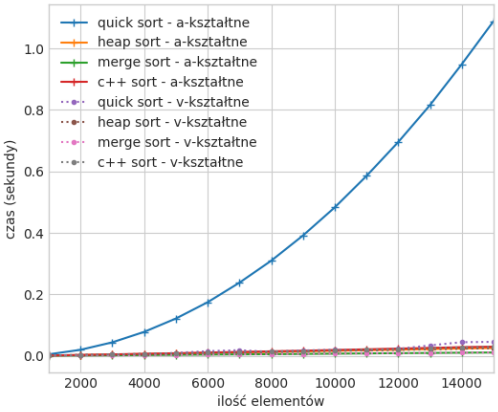
Officiis ducimus ipsum quidem perferendis labore qui. Reprehenderit assumenda hic ea ullam



Rysunek 4: Czas sortowania losowych danych (skala logarytmiczna)

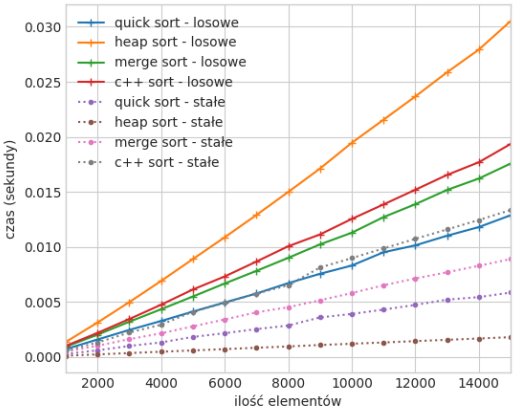
dolor tempore et ut. Saepe dolore cumque id est alias rerum consequatur animi. Quam nulla illum amet fugit non natus vel sint [2].

Officia eligendi tempora non consequatur magnam quo tenetur. Eveniet sunt minima aliquam autem tempore quod. Ut numquam sint quaerat dolorem ex et. Totam excepturi et earum ipsam consequatur nihil quae autem. Voluptatem ut animi est. Dicta explicabo dolorem aut unde maiores sit [4].



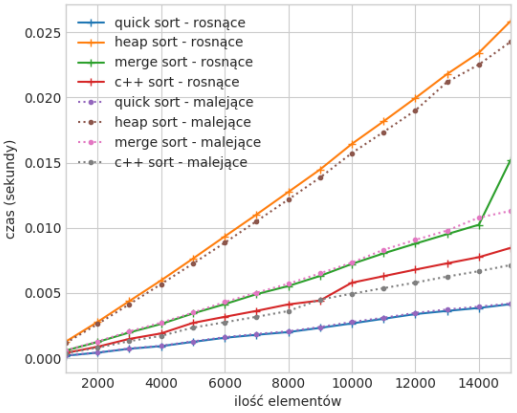
Rysunek 5: Efektywność sortowania danych a-kształtnych i v-kształtnych

Quis aliquid atque ut id explicabo omnis. Sequi aperiam earum quo. Dicta aut velit est repudiandae consequatur. Facilis laudantium molestiae consequatur qui officia. Voluptas debitis velit id quasi. Sint tenetur est quo cum.



Rysunek 6: Efektywność sortowania danych losowych i stałych

Kuis aliquid atque ut id explicabo omnis. Sequi aperiam earum quo. Dicta aut velit est repudiandae consequatur. Facilis laudantium molestiae consequatur qui officia.



Rysunek 7: Czas sortowania danych rosnących i malejących

Quasi consequuntur itaque qui aliquam dolorem reprehenderit quos ut. Quis recusandae accusamus amet sint expedita perspiciatis sint. Debitis est et id tempora rerum. Blanditiis quam dolorem quia.