Dodawanie i szukanie w drzewie binarnym

218582

16 maja 2016

1 Wstęp

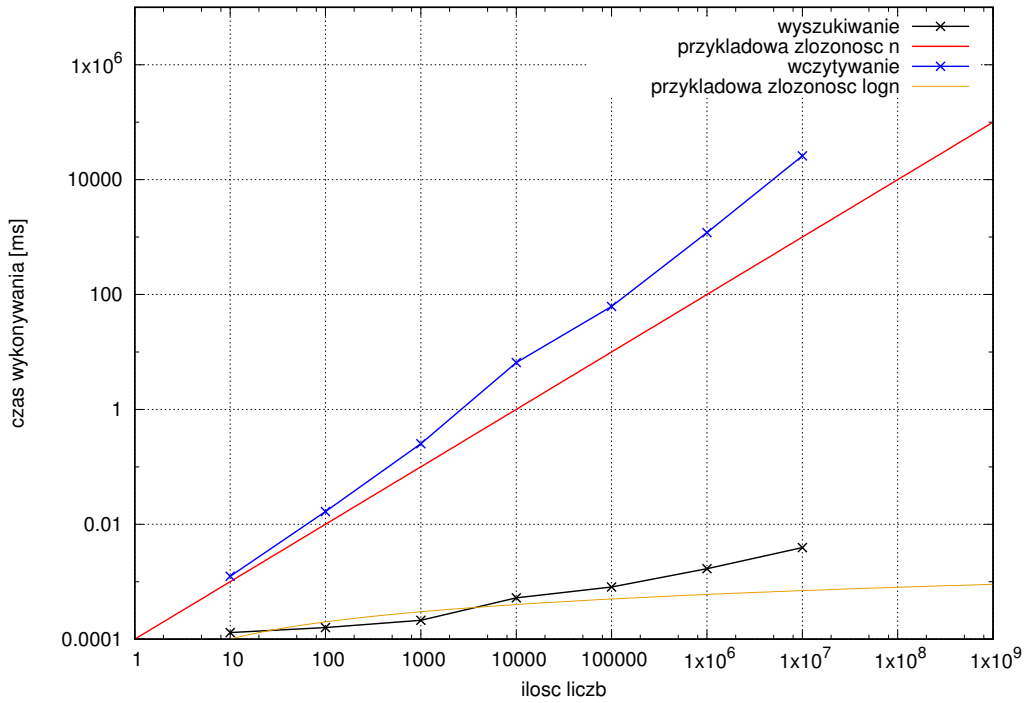
Celem zadania było zmierzenie czasu wyszukiwania klucza w drzewie czerwono-czarnym. Oczekiwana złożoność wyszukiwania w obecnie zaimplementowanym drzewie binarnym to $O(log(n))$, a dodawania - $O(n)$. Drzewo jest balansowane tak, aby nie osiągnięto przypadków skrajnych, w których złożoność wyszukiwania wyniosłaby $O(n)$.

Porównano drzewo czerwono-czarne do drzewa BST bez balansowania.

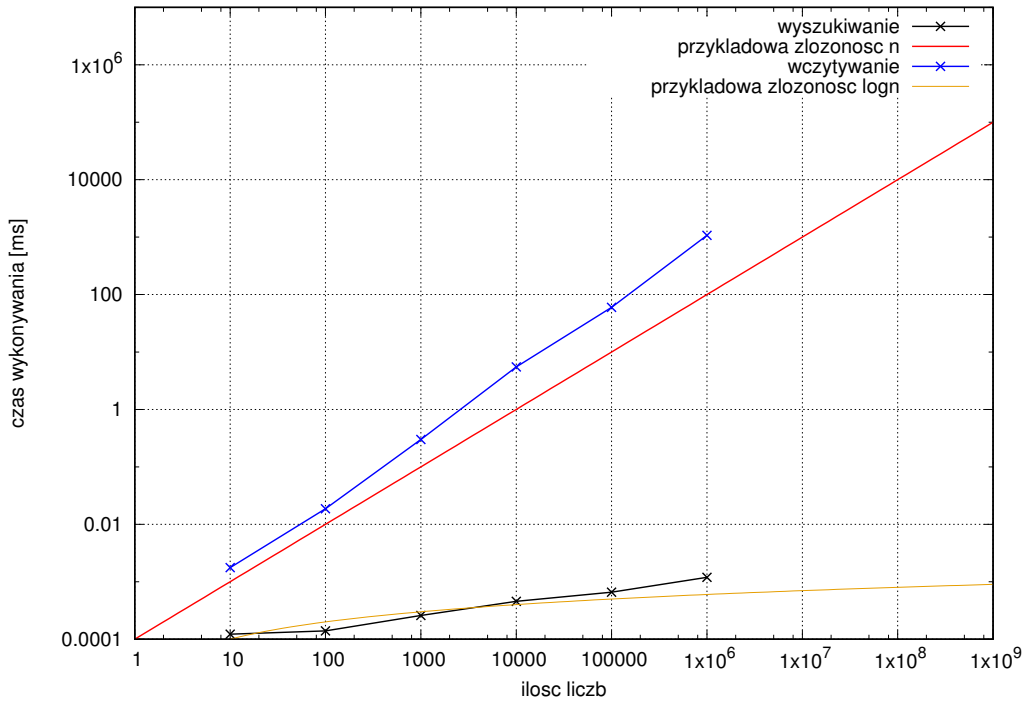
Losowość elementów wpisywanych do drzewa zapewnia użycie funkcji `clock()` z biblioteki `ctime` w `srand()`, która zwraca inną wartość dla każdego taktu procesora.

2 Wyniki

Zgodnie ze standardową procedurą, przeprowadzono testy dla kolejnych rozmiarów problemów. Wyniki prezentuje wykres 1.



Rysunek 1: Działanie wyszukiwania i wpisywania elementów do drzewa niezbalansowanego



Rysunek 2: Działanie wyszukiwania i wpisywania elementów do drzewa czerwono-czarnego

3 Wnioski

Zgodnie z wykresem na Rys. 1, można zauważyć, że dla mniejszych rozmiarów złożoność operacji wyszukiwania i dodawania elementów do binarnego drzewa przeszukiwań pokrywa się z teoretyczną. Jednakże im większa ilość elementów tym złożoność empiryczna zaczyna bardziej odbiegać od teoretycznej. Wynika to z faktu, że nie jest zatrzymywany rozrost drzewa - drzewo nie jest balansowane.

Jak widać na wykresie na Rys. 2, złożoność wyszukiwania w drzewie, którego nadmierny rozrost niektórych gałęzi jest zatrzymywany i tworzone jest drzewo zbalansowane (tutaj - czerwono-czarne), pozostaje w zgodzie ze złożonością teoretyczną niezależnie od ilości elementów. Również złożoność dodawania elementów nie odbiega znacząco od złożoności teoretycznej.

Balansowanie drzewa daje znaczącą poprawę złożoności wyszukiwania.