Projektowanie algorytmów i metody sztucznej inteligencji

Dawid Marszałkiewicz 218665

24 kwietnia 2016

1 Zadanie

Zmierzyć czas odczytu z drzewa binarnego po wypełnieniu go N elementami, gdzie $N = \{100, 1000, 10000, 1000000, 10000000, 100000000\}$

2 Analiza zadania

2.1 Drzewo binarne - implementacja

Drzewo binarne zostało zaimplementowane na tablicy dynamicznej. Korzeń zostal zapisany jako pierwszy element (zerowy pusty). Indeks lewego syna obliczany jest ze wzoru $2 \cdot indeksOjca$, natomiast prwaego syna definiuje wzór $2 \cdot indeksOjca + 1$. Analogicznie $\frac{indeksSyna}{2}$ opisuje indeks ojca.

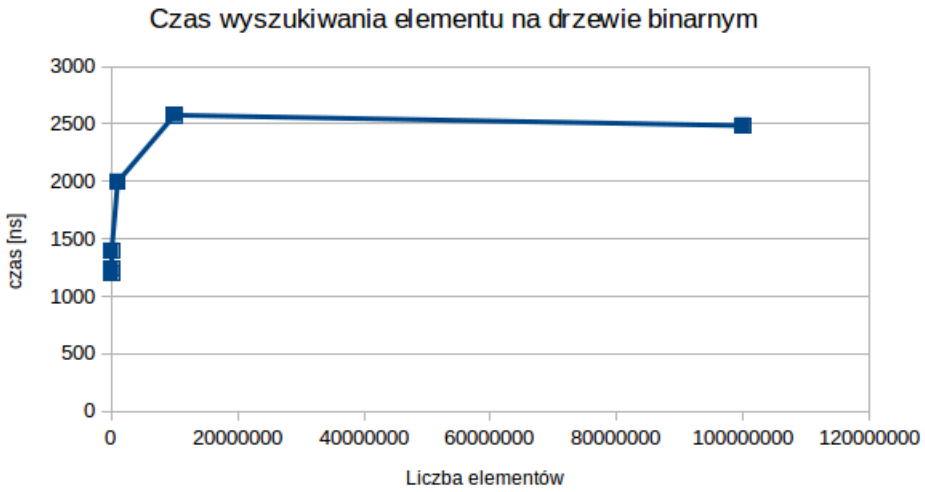
2.1.1 Równoważenie drzewa

W przypadku tego zadania równoważenie drzewa wykonywane jest w następujący sposób. Do tablicy pomocniczej zapisywane są kolejne odwiedzane elementy w przejściu *in-order*, w ten sposób powstaje posortowana tablica. Następnie środkowy element zostaje korzeniem. Z elementów na lewo od niego tworzone jest rekurencyjnie lewe poddrzewo, z elementów na prawo - prawe poddrzewo.

2.2 Tabela z średnimi wynikami wyszukiwania

l. zapisanych elementów	czas
	[ns]
100	1202,5
1000	1243,2
10000	1396,9
1000000	1997,7
10000000	2577,1
100000000	2486,4

2.3 Wykres czasów zapisu n-tego elementu



3 Wnioski

- Zgodnie z wykresem w punkcie 2.3 wyszukiwanie w binarnym drzewie poszukiwań ma złożoność obliczeniową $O(logn)$. Wynika to z organizacji drzewa (mniejsze elementy po lewej stronie, większe po prawej) oraz z maksymalnej wysokości drzewa, która dla n elementowej struktury wynosi $log_2 n$.

- Kosztem utrzymywania odpowiedniego porządku w strukturze otrzymujemy dostęp do elementów w krótkim czasie.

- Równoważenie drzewa w sposób z punktu 2.1.1 sprawia, że za każdym razem zostaje posortowane całe drzewo - złożoność obliczeniowa $O(n)$. Lepszym rozwiązaniem jest użycie samoorganizującego drzewa czerwono-czarnego ze złożonością obliczeniową dodawania elementów $O(logn)$.