Analiza złożonych struktur danych

Samuel Nowak, Bioinformatyka rok I, nr indeksu 145165
Michał Stanoch, Bioinformatyka rok I, nr indeksu 145168

Cele zadania wraz ze specyfikacją komputera pomiarowego

Celem zadania była analiza działania listy jednokierunkowej oraz BST(Binarnego drzewa poszukiwań), a także porównanie czasu wyszukiwania i dodawania elementów w różnych przypadkach dodawania danych (2 rodzaje w wypadku listy jednokierunkowej, 3 rodzaje w wypadku drzewa binarnego). Do naszych zadań należało napisanie algorytmów dodających dane do struktur oraz algorytmów wyszukujących elementy w takowych strukturach. Do kompilowania użyty został standardowy kompilator środowiska Code::Blocks.

Specyfikacja komputera pomiarowego:

OS:	Windows 10 64-bit
Procesor:	Intel® Core™ i7 - 4790k @ 4.00GHz (8 CPUs), ~4.0GHz
Pamięć RAM:	12 288 MB

Dobór wielkości testowanych struktur

Większość wielkości struktur została dobrana w taki sposób, aby łatwo można było zaobserwować wzrost czasów dodawania i szukania. Przykładowo, n_0= 1000, n_5= 10000, n_5/n_0 = 10, n_8= 30000, n_8/n_0 = 30. Niektóre wartości, takie jak n_1 (4269) zostały wybrane dla sprawdzenia działania programu dla wartości nieparzystych. Maksymalną wartością jest 40000, ponieważ dla większej ilości elementów program wyrzuca błąd.

Omówienie wyników zadania

Wyniki zostały zapisane w Tabeli 1. dla listy jednokierunkowej i w Tabeli 2. dla drzewa binarnego z dokładnością do 10^{-4} sekundy oraz zaprezentowane w formie wykresów 2.1. i 2.1.1. dla listy jednokierunkowej i 2.2. oraz 2.2.1. dla drzewa BST.

Porównując oba rodzaje struktur, możemy łatwo zauważyć różnicę w ich funkcjonowaniu dla różnych rodzajów wypełnień, od których zależny jest także czas wyszukiwania elementów. Drzewo binarne działa bardzo szybko dla wypełniania losowego oraz wypełnienia binarnego. Natomiast dla wypełnienia wzrastającego działa zdecydowanie gorzej, jest to różnica rzędu czasu wykonywania operacji 100 razy dłuższego niż w wypadku innych wypełnień, na przykład przy 7551 elementach, wzrastające wypełnienie wykonuje się w 0.1070s, a w wypełnieniu binarnym w 0.001s. Jest to najpewniej spowodowane tym, że przy wzrastających wartościach dodawanych do drzewa binarnego, w celu zachowania prawidłowej struktury drzewa binarnego wykonywane są duże ilości niepotrzebnych operacji porównania elementów. Sytuacja przedstawia się podobnie przy wyszukiwaniu danych. Lista jednokierunkowa wypada bardzo podobnie w przypadku przetasowania danych, jak i bez ich przetasowania. Różnica w czasie wykonywania operacji jest marginalna (lekka

przewaga dla przetasowania). Porównując listę z drzewem binarnym, widać, że są przystosowane do różnych typów wypełnień. Drzewo binarne wypada dobrze w momencie gdy wartości są dodawane losowo oraz binarnie, ale nie radzi sobie z wartościami wzrastającymi, w którym to przypadku lista jednokierunkowa radzi sobie dobrze. Aby podsumować wyniki zadania można stwierdzić, że lista jednokierunkowa sprawdza się bardzo dobrze dla wypełnienia wzrastającego, lecz w pozostałych kategoriach wypełnienia drzewo binarne sprawdza się dużo lepiej.

Złożoność czasowa dla listy przy wypełnianiu (zarówno z i bez przetasowania) teoretycznie wynosi O(1), i ma to przełożenie na nasze wyniki, odchylenia są marginalne.

Złożoność czasowa dla listy przy wyszukiwaniu rośnie $O(n^2)$. Teoretycznie lista jednokierunkowa przy wyszukiwaniu powinna mieć złożoność O(n), ale w tym konkretnym programie czas wykonywania operacji dodatkowo wzrasta najprawdopodobniej z powodu wyszukiwania wszystkich elementów w pętli.

Złożoność czasowa dla drzewa binarnego przy wypełnianiu wzrastającym wynosi O(n), widać to chociażby przy porównaniu wyników 5000/10000 elementów.

Złożoność czasowa dla drzewa binarnego przy wypełnieniu losowym i binarnym wynosi O(1). Wartości są bardzo do siebie zbliżone, a różnice są marginalne. Przy wyszukiwaniu dla wartości wypełnionych wzrastająco złożoność czasowa wynosi O(log n), co widać chociażby przy stosunku 5000 elementów / 40000 elementów.

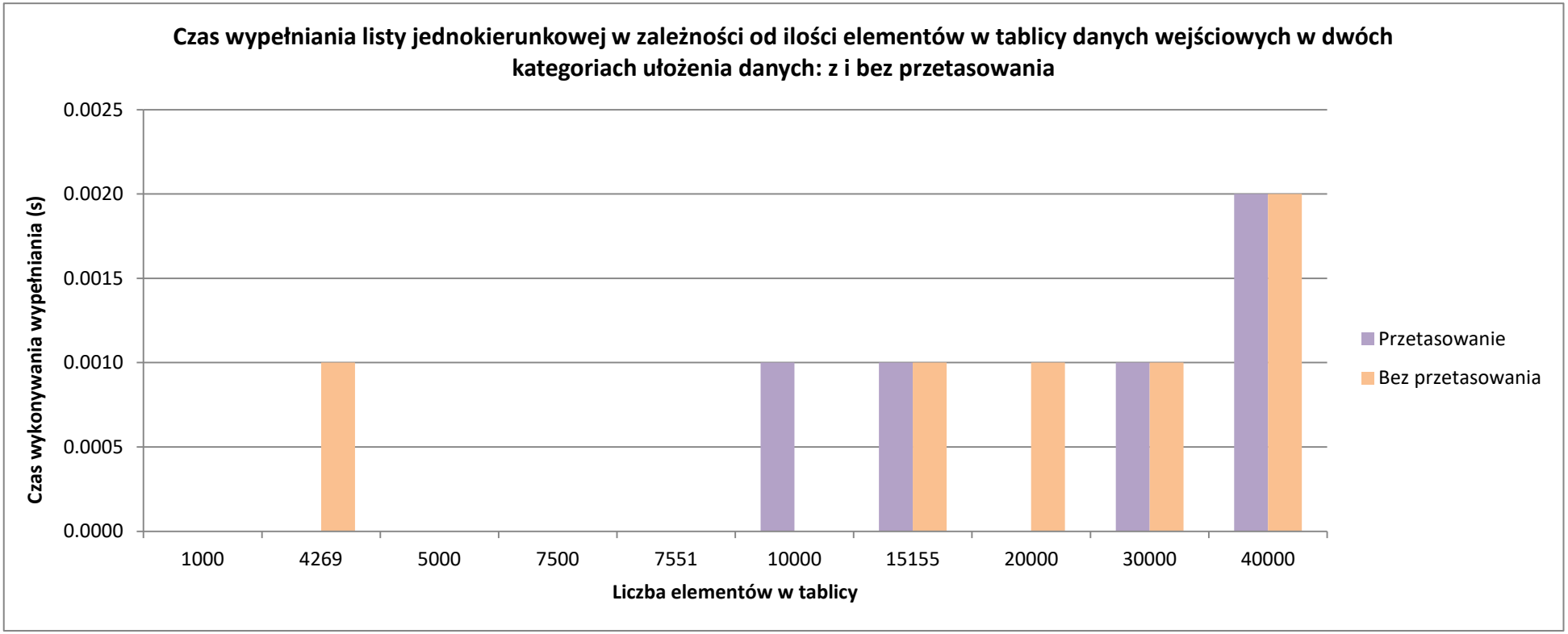
Złożoność czasowa dla wyszukiwania w drzewie binarnym przy wypełnieniu losowym i binarnym wynosi O(1), a odchylenia są minimalne.

Tabele i wykresy

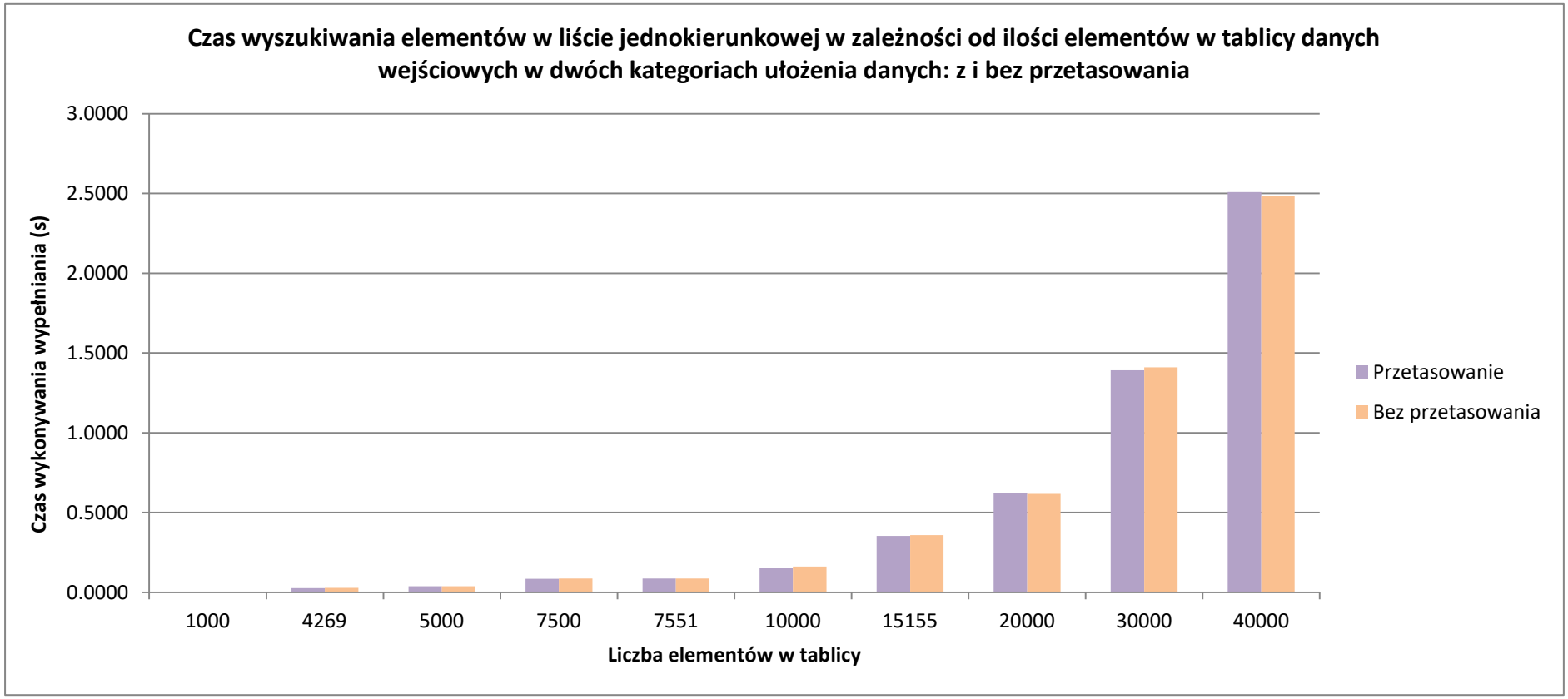
1. Lista jednokierunkowa

	Rodzaj przetasowania			
	Przetasowanie		Bez przetasowania	
Ilość elementów w tablicy danych/ liście	Czas wypełnienia	Czas wyszukiwania	Czas wypełniania	Czas wyszukiwania
1000	0.0000	0.0020	0.0000	0.0010
4269	0.0000	0.0270	0.0010	0.0280
5000	0.0000	0.0380	0.0000	0.0380
7500	0.0000	0.0850	0.0000	0.0870
7551	0.0000	0.0870	0.0000	0.0860
10000	0.0010	0.1520	0.0000	0.1610
15155	0.0010	0.3540	0.0010	0.3580
20000	0.0000	0.6200	0.0010	0.6170
30000	0.0010	1.3910	0.0010	1.4100
40000	0.0020	2.5080	0.0020	2.4820

Tabela 1. Czas wypełniania listy jednokierunkowej i wyszukiwania w niej elementów w zależności od ilości elementów w tablicy danych wejściowych/ liście w dwóch kategoriach przetasowania (podany w sekundach). Czas rzędu wielkości 0.0000 znaczy, że operacja przebiegła pomyślnie, lecz w czasie szybszym niż 10^{-4} sekundy.



Wykres 1.1. Czas wypełniania listy jednokierunkowej w zależności od ilości elementów w tablicy danych wejściowych w dwóch kategoriach ułożenia danych: z i bez przetasowania. Czas rzędu wielkości 0.0000 znaczy, że operacja przebiegła pomyślnie, lecz w czasie szybszym niż 10^{-4} sekundy.

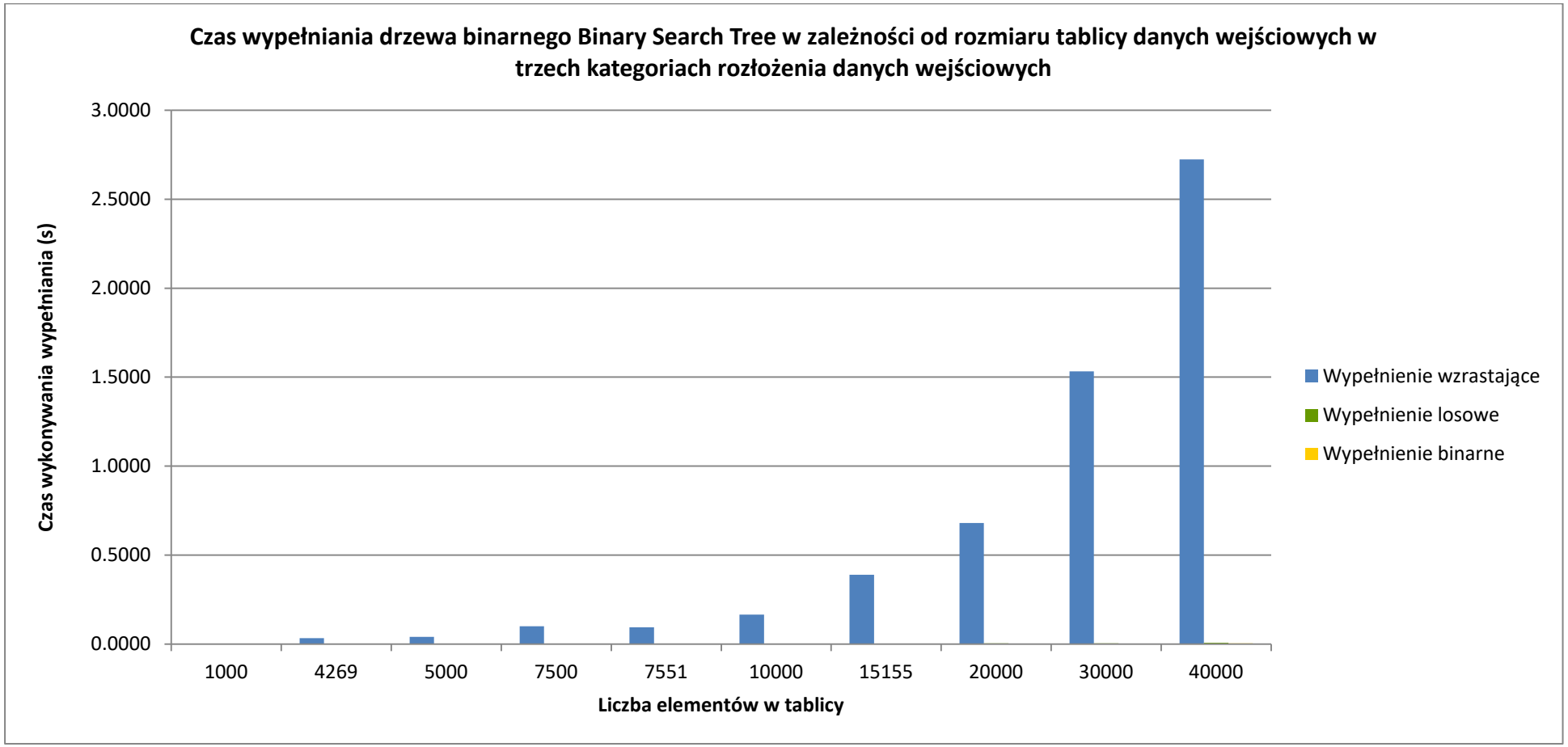


Wykres 1.2. Czas wypełniania listy jednokierunkowej w zależności od ilości elementów w tablicy danych wejściowych w dwóch kategoriach ułożenia danych: z i bez przetasowania. Czas rzędu wielkości 0.0000 znaczy, że operacja przebiegła pomyślnie, lecz w czasie szybszym niż 10^{-4} sekundy.

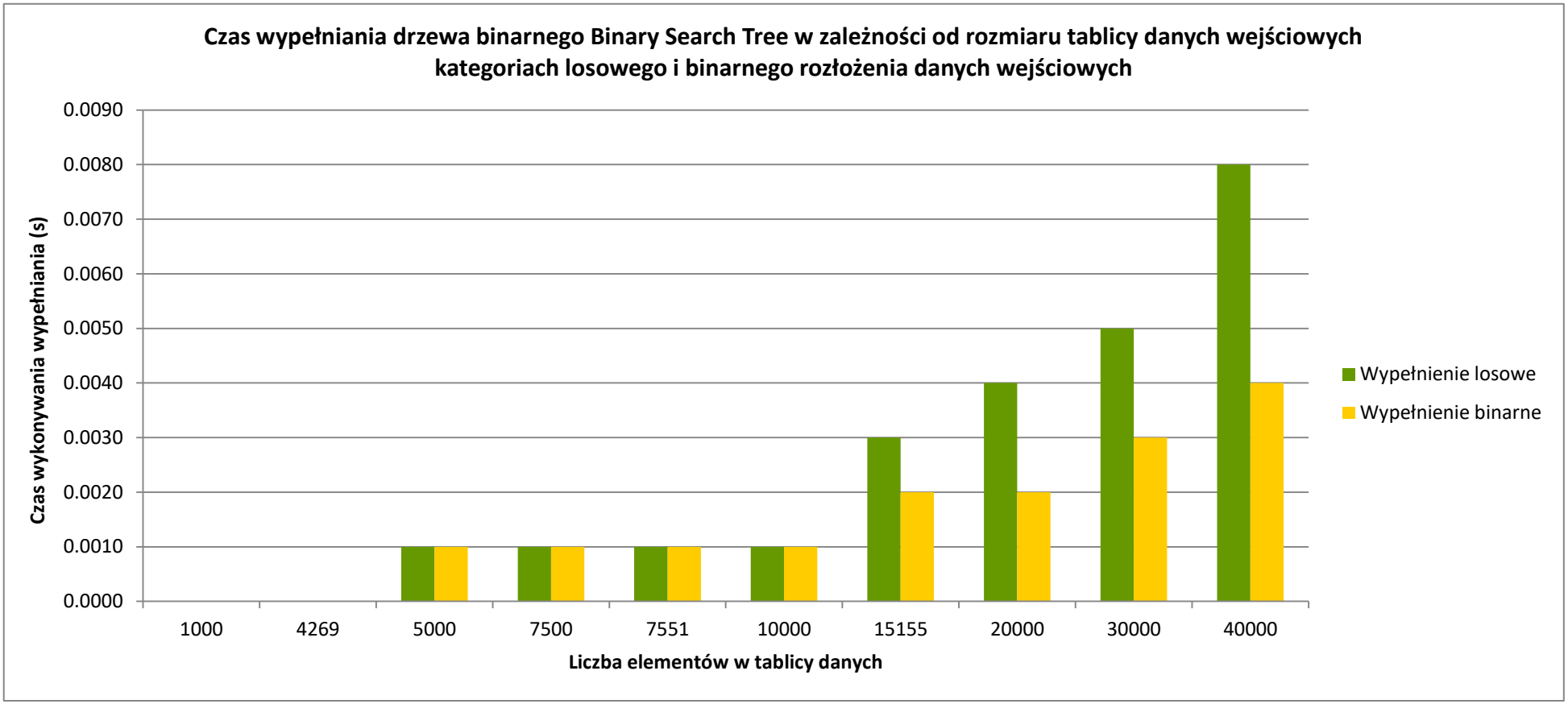
2. Drzewo binarne Binary Search Tree

	Rodzaj wypełnienia					
	Wypełnienie wzrastające		Wypełnienie losowe		Wypełnienie binarne	
Ilość elementów w tablicy danych/ drzewie	Czas wypełnienia	Czas wyszukiwania	Czas wypełniania	Czas wyszukiwania	Czas dla wypełnienia binarnego	Czas wyszukiwania
1000	0.0010	0.0010	0.0000	0.0000	0.0000	0.0000
4269	0.0330	0.0540	0.0000	0.0010	0.0000	0.0010
5000	0.0410	0.0620	0.0010	0.0010	0.0010	0.0010
7500	0.1010	0.1450	0.0010	0.0010	0.0010	0.0010
7551	0.0950	0.1440	0.0010	0.0010	0.0010	0.0010
10000	0.1660	0.2540	0.0010	0.0010	0.0010	0.0010
15155	0.3900	0.5830	0.0030	0.0020	0.0020	0.0020
20000	0.6810	1.0200	0.0040	0.0030	0.0020	0.0030
30000	1.5320	2.2980	0.0050	0.0050	0.0030	0.0050
40000	2.7240	4.1620	0.0080	0.0080	0.0040	0.0060

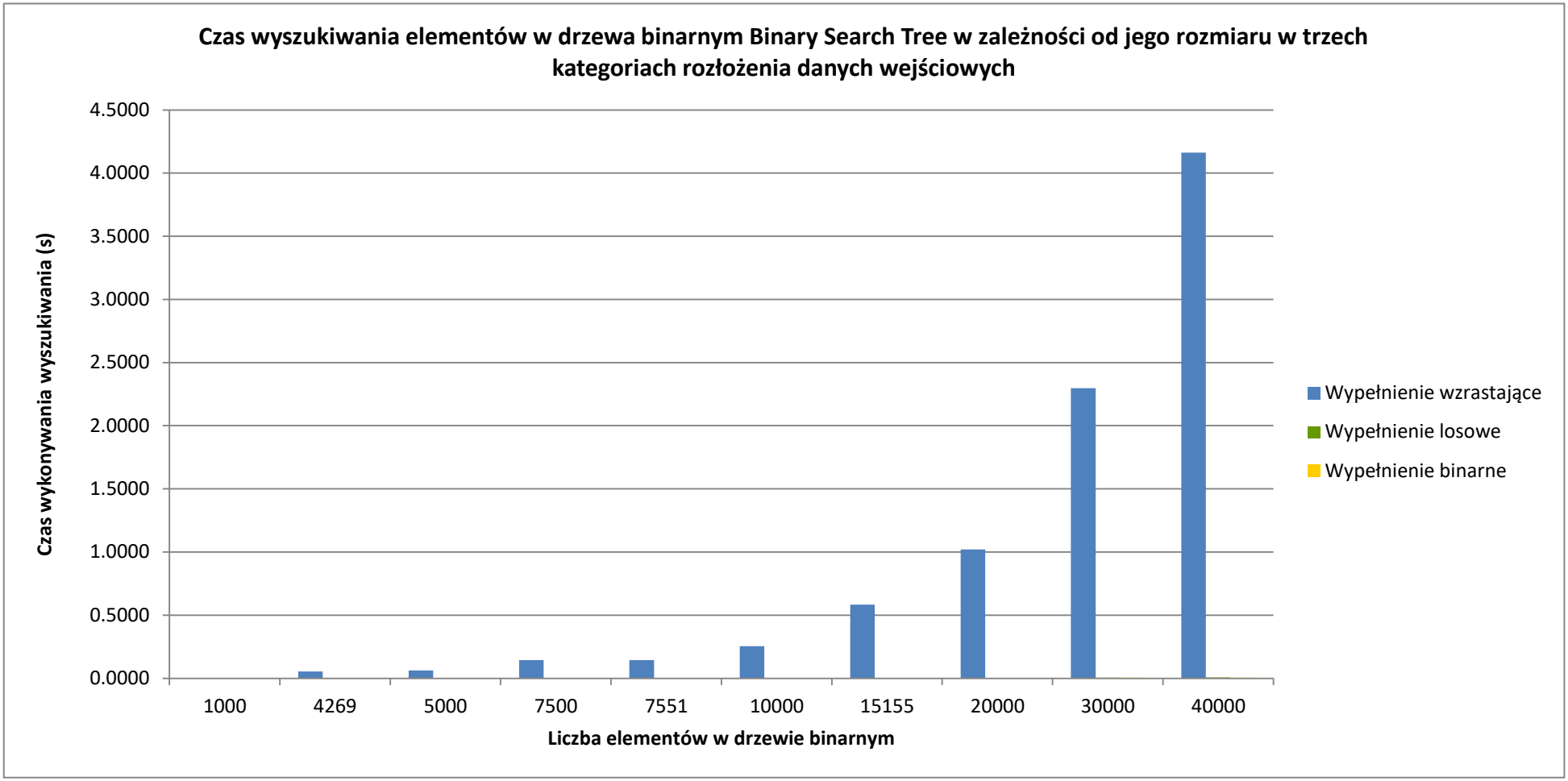
Tabela 2. Czas wypełniania drzewa BST i wyszukiwania w nim elementów w zależności od ilości elementów tablicy danych wejściowych/ drzewie w trzech kategoriach wypełnienia drzewa (podany w sekundach). Czas rzędu wielkości 0.0000 znaczy, że operacja przebiegła pomyślnie, lecz w czasie szybszym niż 10^{-4} sekundy.



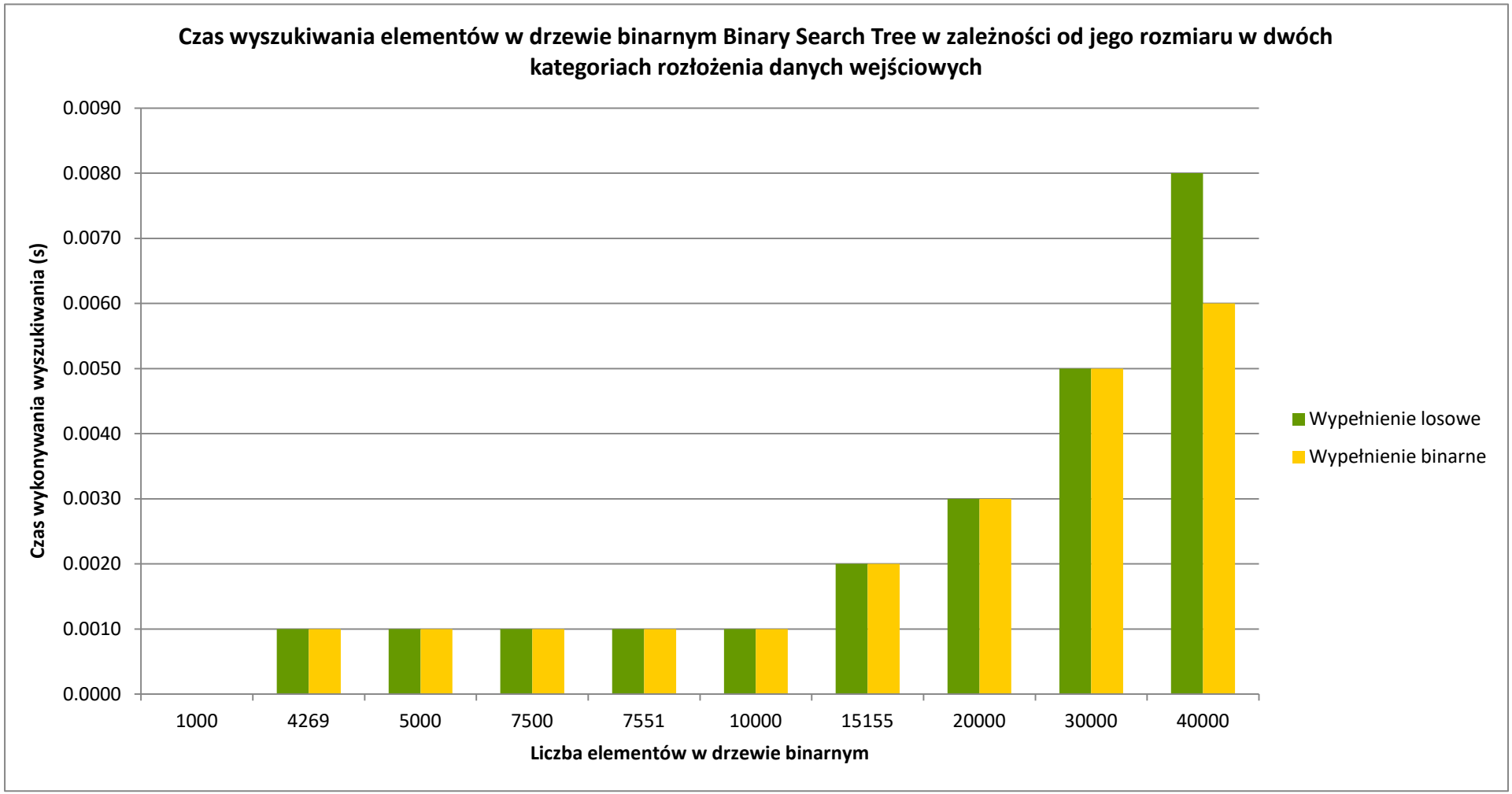
Wykres 2.1. Czas wypełniania drzewa binarnego Binary Search Tree w zależności od rozmiaru tablicy danych wejściowych w trzech kategoriach rozłożenia danych wejściowych. Wartości dla wypełnień losowego i binarnego są ledwie widoczne przy osi X (dla wyraźniejszej reprezentacji danych dla wypełnień losowego i binarnego, patrz: wykres 2.1.1.).



Wykres 2.1.1. Czas wypełniania drzewa binarnego Binary Search Tree w zależności od rozmiaru tablicy danych wejściowych w kategoriach losowego i binarnego rozłożenia danych wejściowych.



Wykres 2.2. Czas wyszukiwania elementów w drzewa binarnym Binary Search Tree w zależności od jego rozmiaru w trzech kategoriach rozłożenia danych wejściowych. Wartości dla wypełnień losowego i binarnego są ledwie widoczne przy osi X. (dla wyraźniejszej reprezentacji danych dla wypełnień losowego i binarnego, patrz: wykres 2.2.1.).



Wykres 2.2.1. Czas wyszukiwania elementów w drzewa binarnym Binary Search Tree w zależności od jego rozmiaru w kategoriach losowego i binarnego rozłożenia danych wejściowych.