Algorytmy obsługi tablicy dynamicznie rozszerzalnej - badania

218582

6 marca 2016

1 Wstęp

Celem pracy na laboratorium było utworzenie klasy tablicy dynamicznie powiększającej się. Algorytm w sytuacji, gdy dana tablica jest za mała, by zmieścić nowe dane, tworzy większą tablicę, kopiuje ze starej dane, dodaje nowe i zastępuje starą tablicę nową, większą.

Istnieją różne sposoby powiększania tablicy - można ją powiększać o jedną komórkę za każdym razem, zwiększać półtorakrotnie, dwukrotnie itd. Zbadano trzy wyżej wymienione metody.

2 Program

Program pobiera ze standardowego wejścia liczby i wpisuje je kolejno do tablicy, którą rozszerza wybranym algorytmem w miarę potrzeby. Przykład wywołania:

```
$ cat plik_z_danymi | ./program
```

Pliki z danymi utworzono przy użyciu opracowanego wcześniej programu generującego pseudolosowe liczby całkowite z zadanego zakresu.

Na wyjściu program wyświetla czas działania algorytmu, oraz informację o ilości wpisanych liczb i zaalokowanych pól tablicy, które pozwalają na kontrolę poprawności działania. Przykładowe wyjście:

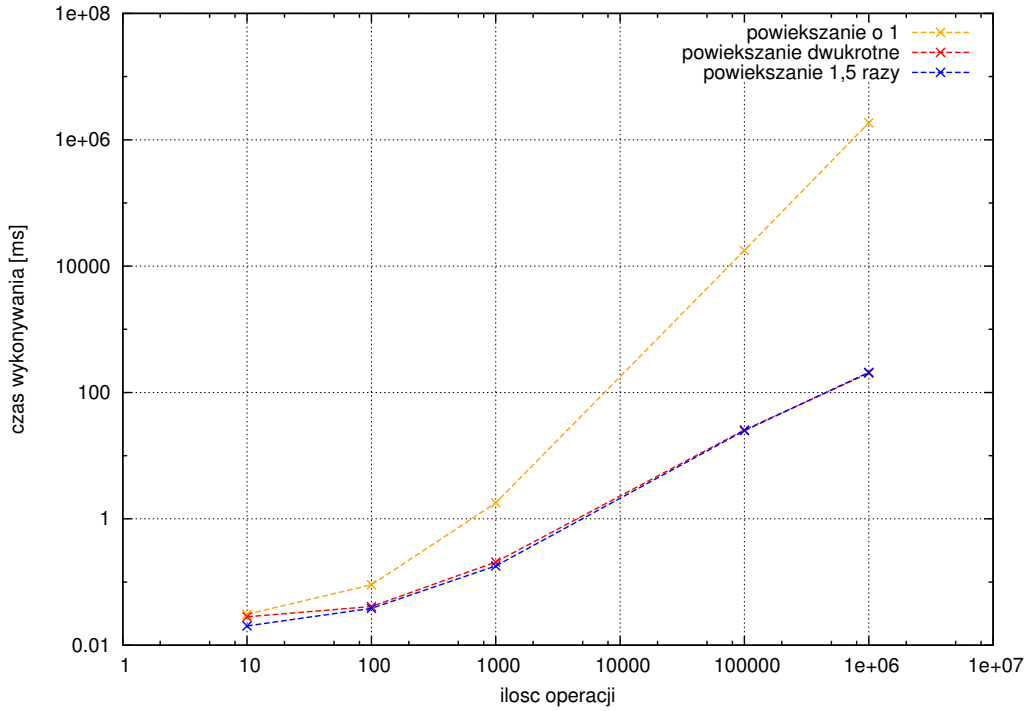
```
-----------------------------------------------------

Operacja wykonana w 0.261694 ms.
Elementow:          1000
Wielkosc tablicy:   1234
```

3 Wyniki

Tablica 1: Czasy wykonywania się algorytmów

	Czas [ms]		
Ilość liczb	Powiększanie o 1	Powiększanie 1,5 raza	Powiększanie 2 razy
10	0,031516	0,020153	0,028121
100	0,090258	0,038362	0,040921
1000	1,78643	0,179709	0,207396
100000	17784,4	24,9265	25,5722
1000000	1,85616E+06	208,014	203,177



Rysunek 1: Porównanie czasu działania algorytmów powiększania tablicy

Pominięto sprawdzenie działania algorytmów dla następującej ilości elementów: 10^9, z powodu szacowanego czasu wykonywania się czynności na poziomie setek lat, lub szacowanej wymaganej pamięci przekraczającej dostępną. Dodano sprawdzenie czasu działania dla 100 elementów.

4 Podsumowanie

Zbyt częste kopiowanie tablicy przy dużej ilości danych powoduje znaczny wzrost czasu działania funkcji. Sprawdza się jednak, gdy mamy wpisać relatywnie niewielką ilość danych, gdyż nie zajmujemy niepotrzebnie pamięci. Algorytmy tworzące miejsce w tablicy „na zapas" działają znacznie krócej, ale zajmują więcej pamięci, zwykle nie wykorzystując jej w pełni. Należy więc dobierać algorytm powiększania tablicy zależnie od wymagań.