Laboratorium 8 - Drzewa binarne, drzewa binarne czerwono-czarne

Wygenerowano przez Doxygen 1.8.9.1

Śr, 20 maj 2015 21:48:20

Spis treści

1 Indeks hierarchiczny

1.1 Hierarchia klas

Ta lista dziedziczenia posortowana jest z grubsza, choć nie całkowicie, alfabetycznie:

2 Indeks klas

2.1 Lista klas

Tutaj znajdują się klasy, struktury, unie i interfejsy wraz z ich krótkimi opisami:

3 Indeks plików

3.1 Lista plików

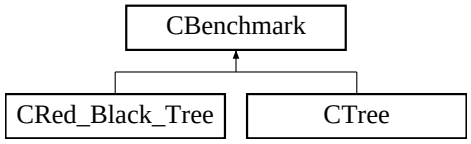
Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

4 Dokumentacja klas

4.1 Dokumentacja klasy CBenchmark

`#include <benchmark.hh>`

Diagram dziedziczenia dla CBenchmark



Metody publiczne

- virtual void start_timer ()

  definicja metody start_timer rozpoczyna pomiar czasu zapisuje dane do zmiennej performanceCountStart korzysta z metdoy StartTimer()
- virtual void stop_timer ()

  definicja metody stop_timer konczy pomiar czasu zapsiuje dane do zmiennej performanceCountEnd korzysta z metody endTimer
- virtual int put_time_to_file (int size_of_list)

  definicja metody put_time_to_file otiwra plik o nazwie 'timing.txt' zapisuje do niego ilosc elementow listy oraz czas przeprowadzenia operacji przez klasy obserwowane zamyka plik.

Metody prywatne

- LARGE_INTEGER startTimer ()
- LARGE_INTEGER endTimer ()

Atrybuty prywatne

- LARGE_INTEGER performanceCountStart
- LARGE_INTEGER performanceCountEnd

4.1.1 Opis szczegółowy

definicja klasy CBenchmark definijue stoper zliczajacy czas wykoania operacji przez inne klasy jest przykladem wzorca obserwatora obserwuje klase CSort i zlicza czas sortowania listy

4.1.2 Dokumentacja funkcji składowych

4.1.2.1 LARGE_INTEGER CBenchmark::endTimer ( ) [private]

4.1.2.2 int CBenchmark::put_time_to_file (int *size_of_list*) [virtual]

definicja metody put_time_to_file otiwra plik o nazwie 'timing.txt' zapisuje do niego ilosc elementow listy oraz czas przeprowadzenia operacji przez klasy obserwowane zamyka plik.

Parametry

size_of_list	- rozmiar listy

Zwraca

czas przeprowadzenia operacji
-1 w przypadu bledu otwarcia pliku

4.1.2.3 void CBenchmark::start_timer ( ) [virtual]

definicja metody start_timer rozpoczyna pomiar czasu zapisuje dane do zmiennej performanceCountStart korzysta z metdoy StartTimer()

4.1.2.4 LARGE_INTEGER CBenchmark::startTimer ( ) [private]

4.1.2.5 void CBenchmark::stop_timer ( ) [virtual]

definicja metody stop_timer konczy pomiar czasu zapsiuje dane do zmiennej performanceCountEnd korzysta z metody endTimer

4.1.3 Dokumentacja atrybutów składowych

4.1.3.1 LARGE_INTEGER CBenchmark::performanceCountEnd [private]

4.1.3.2 LARGE_INTEGER CBenchmark::performanceCountStart [private]

Dokumentacja dla tej klasy została wygenerowana z plików:

- benchmark.hh
- benchmark.cpp

4.2 Dokumentacja klasy CNode

definicja klasy CNode Definijue pojedynczy wezel drzewa binarnego. Jest zaprzyjazniona z klasa CTree.

`#include <node.hh>`

Metody publiczne

- CNode ()

  definicja konstruktora bezparametrycznego Ustawia wskazniki na NULL.
- ∼CNode ()

  definicja destruktora klasy CNode
- void preorder () const

  definicja metody preorder Definiuje przechodzenie przez drzewo odwiedzajac wszystkielefe galzezie, nastepnie wszystkie prawe galezie. Jest wywolywana w klasie CTree. Metoda nie zmienia stanu obiektu.

Atrybuty prywatne

- int value
- CNode ∗ up
- CNode ∗ left
- CNode ∗ right

Przyjaciele

- class CTree

4.2.1 Opis szczegółowy

definicja klasy CNode Definijue pojedynczy wezel drzewa binarnego. Jest zaprzyjazniona z klasa CTree.

4.2.2 Dokumentacja konstruktora i destruktora

4.2.2.1 CNode::CNode ( ) [inline]

definicja konstruktora bezparametrycznego Ustawia wskazniki na NULL.

wskaznik na prawego syna

4.2.2.2 CNode::~CNode ( )

definicja destruktora klasy CNode

4.2.3 Dokumentacja funkcji składowych

4.2.3.1 void CNode::preorder ( ) const

definicja metody preorder Definiuje przechodzenie przez drzewo odwiedzajac wszystkielefe galzezie, nastepnie wszystkie prawe galezie. Jest wywolywana w klasie CTree. Metoda nie zmienia stanu obiektu.

4.2.4 Dokumentacja przyjaciół i funkcji związanych

4.2.4.1 friend class CTree [friend]

4.2.5 Dokumentacja atrybutów składowych

4.2.5.1 CNode* CNode::left [private]

wskaznik na rodzica

4.2.5.2 CNode* CNode::right [private]

wskaznik na lewego syna

4.2.5.3 CNode* CNode::up [private]

przechowywane dane

4.2.5.4 int CNode::value [private]

Dokumentacja dla tej klasy została wygenerowana z plików:

- node.hh
- node.cpp

4.3 Dokumentacja klasy CRed_Black_Node

definicja klasy CRed_Black_Node definicja wezla drzewa czerwono-czarnego. Klasa zaprzyjazniona z klasa CRed_Black_Tree

`#include <Red_black_node.hh>`

Metody publiczne

- CRed_Black_Node ()

  definicja konstruktora bezparametrycznego ustaiwa wskazniki na NULL.
- ~CRed_Black_Node ()

  definicja destruktora klasy

Atrybuty prywatne

- CRed_Black_Node * up
- CRed_Black_Node * left
- CRed_Black_Node * right
- int value
- char color

Przyjaciele

- class CRed_Black_Tree

4.3.1 Opis szczegółowy

definicja klasy CRed_Black_Node definicja wezla drzewa czerwono-czarnego. Klasa zaprzyjazniona z klasa CRed_Black_Tree

4.3.2 Dokumentacja konstruktora i destruktora

4.3.2.1 CRed_Black_Node::CRed_Black_Node ( ) [inline]

definicja konstruktora bezparametrycznego ustaiwa wskazniki na NULL.

informacja o kolorze wezla

4.3.2.2 CRed_Black_Node::~CRed_Black_Node ( )

definicja destruktora klasy

4.3.3 Dokumentacja przyjaciół i funkcji związanych

4.3.3.1 friend class CRed_Black_Tree [friend]

4.3.4 Dokumentacja atrybutów składowych

4.3.4.1 char CRed_Black_Node::color [private]

[rzechowywana wartosc

4.3.4.2 CRed_Black_Node* CRed_Black_Node::left [private]

wskaznik na rodzica

4.3.4.3 CRed_Black_Node* CRed_Black_Node::right [private]

wskaznik na lewego syna

4.3.4.4 CRed_Black_Node* CRed_Black_Node::up [private]

4.3.4.5 int CRed_Black_Node::value [private]

wskaznik na prawego syna

Dokumentacja dla tej klasy została wygenerowana z plików:

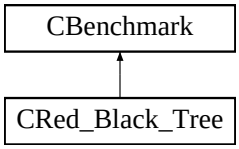
- Red_black_node.hh
- Red_black_node.cpp

4.4 Dokumentacja klasy CRed_Black_Tree

definicja klasy CRed_Black_Tree Definiuje Drzewo czerwono - czarne Posiada wartownika na lisciach. Drzewo czerwono-czarne spelnia nastepujace wlasnosci 1) Każdy węzeł jest czerwony lub czarny. 2) Korzeń jest czarny. 3) Każdy liść jest czarny (Można traktować nil jako liść). 4) Jeśli węzeł jest czerwony, to jego synowie muszą być czarni. 5) Każda ścieżka z ustalonego węzła do liścia liczy tyle samo czarnych węzłów.

`#include <Red_black_tree.hh>`

Diagram dziedziczenia dla CRed_Black_Tree



Metody publiczne

- CRed_Black_Tree ()

  definicja konstruktora bezparametrycznego Ustawia wszystkie wskazniki wezla drzewa na wartownika
- ∼CRed_Black_Tree ()

  definicja destruktora klasy
- CRed_Black_Node ∗ find (int k)

  definicja metody find
- CRed_Black_Node ∗ min (CRed_Black_Node ∗tmp)

  definicja metody min
- CRed_Black_Node ∗ max (CRed_Black_Node ∗r)

  definicja metody max
- void rotate_left (CRed_Black_Node ∗prev)

  definicja metody rotate_left Wykonuje rotacje w lewo. zmienia this z prev, ustawiajac this jako lewego syna. Ustawia rodzica (up) na rodzica wezla this. Ustawia lewego syna this jako prawy syn prev.
- void rotate_right (CRed_Black_Node ∗prev)

  definicja metody rotate_right Wykonyje rotacje w prawo. zmienia this z prev, ustawiajac this jako prawego syna. Ustawia rodzica (up) na rodzica wezla this. Utawia prawego syna this jako lewy syn prav.
- void add (int k)

  definicja metody add Tworzy Wezel drzewa czerwono czarnego o kluczu k i dodaje do go drzewa zgodnie z regulal drzewa binarnego. Koloruje go na czerwono. Przywraca wlasciwosc dzewa czerwno czarnego zgodnie z przypadkami 1) Gdy ojciec i wuj sa czerwoni, zmien ich kolory. 2) Gdy lewy ojeciec i jego lewy syn sa czeroni, wykonaj obrot w prawo dziadka i ojca oraz zamien ich kolory. 3) Gdy lewy ojciec i jego prawy syn sa czerwoni, wykonaj obrot w lewo ojca i syna oraz wykonaj 2). 4) Gdy prawy ojciec i jego prawy syn sa czerwoni, wykonaj obrot w lewo dziadka i ojca, zamien ich kolory 5) Gdy prawy ojciec i jego prawy syn sa czerwoni, wykonaj obrot w prawo ojca i syna oraz wykonaj 4)
- void fill_random (int x)

  definicja metody fill_random dziala analogicznie jak metoda klasy CTree
- int benchmark_add ()

  definicja metody benchmark_add Dzia³a analogicznie jak metoda klasy CTree
- void find_random (int quantity)

  definicja metody find_random Dzia³a analogicznie jak metoda klasy CTree
- int benchmark_find ()

  definicja metody benchmark_find Dziala analogicznie jak metoda klasy CTree

Atrybuty prywatne

- CRed_Black_Node S
- CRed_Black_Node ∗ root

4.4.1 Opis szczegółowy

definicja klasy CRed_Black_Tree Definiuje Drzewo czerwono - czarne Posiada wartownika na lisciach. Drzewo czerwono-czarne spelnia nastepujace wlasnosci 1) Każdy węzeł jest czerwony lub czarny. 2) Korzeń jest czarny. 3) Każdy liść jest czarny (Można traktować nil jako liść). 4) Jeśli węzeł jest czerwony, to jego synowie muszą być czarni. 5) Każda ścieżka z ustalonego węzła do liścia liczy tyle samo czarnych węzłów.

4.4.2 Dokumentacja konstruktora i destruktora

4.4.2.1 CRed_Black_Tree::CRed_Black_Tree ( )

definicja konstruktora bezparametrycznego Ustawia wszystkie wskazniki wezla drzewa na wartownika

wskaznik na korzen drzewa czerwono-czarnego.

4.4.2.2 CRed_Black_Tree::∼CRed_Black_Tree ( )

definicja destruktora klasy

4.4.3 Dokumentacja funkcji składowych

4.4.3.1 void CRed_Black_Tree::add (int *k*)

definicja metody add Tworzy Wezel drzewa czerwono czarnego o kluczu k i dodaje do go drzewa zgodnie z regulal drzewa binarnego. Koloruje go na czerwono. Przywraca wlasciwosc dzewa czerwno czarnego zgodnie z przypadkami 1) Gdy ojciec i wuj sa czerwoni, zmien ich kolory. 2) Gdy lewy ojeciec i jego lewy syn sa czeroni, wykonaj obrot w prawo dziadka i ojca oraz zamien ich kolory. 3) Gdy lewy ojciec i jego prawy syn sa czerwoni, wykonaj obrot w lewo ojca i syna oraz wykonaj 2). 4) Gdy prawy ojciec i jego prawy syn sa czerwoni, wykonaj obrot w lewo dziadka i ojca, zamien ich kolory 5) Gdy prawy ojciec i jego prawy syn sa czerwoni, wykonaj obrot w prawo ojca i syna oraz wykonaj 4)

Parametry

k	dodawany klucz do drzewa czerwono-czarnego

4.4.3.2 int CRed_Black_Tree::benchmark_add ( )

definicja metody benchmark_add Dzia³a analogicznie jak metoda klasy CTree

4.4.3.3 int CRed_Black_Tree::benchmark_find ( )

definicja metody benchmark_find Dziala analogicznie jak metoda klasy CTree

4.4.3.4 void CRed_Black_Tree::fill_random (int *x*)

definicja metody fill_random dziala analogicznie jak metoda klasy CTree

Parametry

x	- ilosc elementow pseudolosowych

4.4.3.5 CRed_Black_Node ∗ CRed_Black_Tree::find (int *k*)

definicja metody find

Parametry

k	szukany klucz

Zwraca

wskaznik na wezel przechowujacy klucz Przeszukuje drzewo czerwono-czarne.

4.4.3.6 void CRed_Black_Tree::find_random (int *quantity*)

definicja metody find_random Dzia³a analogicznie jak metoda klasy CTree

4.4.3.7 CRed_Black_Node * CRed_Black_Tree::max (CRed_Black_Node * *r*)

definicja metody max

Zwraca

wezel o najwiekszym kluczu

4.4.3.8 CRed_Black_Node * CRed_Black_Tree::min (CRed_Black_Node * *tmp*)

definicja metody min

Zwraca

wezel o najmniejszym kluczu

4.4.3.9 void CRed_Black_Tree::rotate_left (CRed_Black_Node * *prev*)

definicja metody rotate_left Wykonuje rotacje w lewo. zmienia this z prev, ustawiajac this jako lewego syna. Ustawia rodzica (up) na rodzica wezla this. Ustawia lewego syna this jako prawy syn prev.

Parametry

prev	obracany wezel nadrzedny przy zalozeniu, ze obracanym wezlem podrzednym jest this.

4.4.3.10 void CRed_Black_Tree::rotate_right (CRed_Black_Node * *prev*)

definicja metody rotate_right Wykonyje rotacje w prawo. zmienia this z prev, ustawiajac this jako prawego syna. Ustawia rodzica (up) na rodzica wezla this. Utawia prawego syna this jako lewy syn prav.

Parametry

prev	obracany wezel nadrzedny przy zalozeniu, ze obracanym wezlem podrzednym jest this.

4.4.4 Dokumentacja atrybutów składowych

4.4.4.1 CRed_Black_Node* CRed_Black_Tree::root [private]

wartownik

4.4.4.2 CRed_Black_Node CRed_Black_Tree::S [private]

Dokumentacja dla tej klasy została wygenerowana z plików:

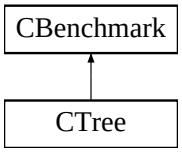
- Red_black_tree.hh
- Red_black_tree.cpp

4.5 Dokumentacja klasy CTree

definicja klasy drzewo Definiuje binarne drzewo zbudowane z wezlow galezi.

`#include <tree.hh>`

Diagram dziedziczenia dla CTree



Metody publiczne

- CTree ()

  definicja konstruktora bezparametrycznego Ustawia korzen na NULL.
- ∼CTree ()

  definicja destruktora klasy CTree wywoluje rekurencujnie destruktr klasy CNode dopuki nie usunie korzenia. Ustawia korzen na NULL.
- void add (int k)

  definicja metdy add Tworzy nowy wezel, nastepnie dodaje go zgodnie z regula drzewa binarnego. Dodawany element zawsze staje sie lisciem.
- void preorder () const

  definicja metody preorder wywoluje metode preorder klasy CNode Wyswietla kolejno lewa galez drzewa, a nastepnie prawa.
- CNode ∗ find (int k)

  definicja metody find
- CNode ∗ get_min ()

  definicja metody get_min
- CNode ∗ get_max ()

  definicja metody get_max
- void fill_random (int x)

  definicja metody fill_random Dodaje do drzewa liczby pseudolowe w zakresie 1-10. Metoda obsewowana przez klase CBenchmark Zapisuje do pliku timing.txt czas dodawania elementow.
- int benchmark_add ()

  definicja metody benchmark_add Wywoluje metode fill_random zawiera petle benchmarkujace w zakresie 1-10, 10-100, ... , 10 000.
- void find_random (int quantity)

  definicja metody find_random
- int benchmark_find ()

  definicja metody benchmark_find Wywoluje metode find_random. Zawiera petle benchmarkujace w zakresie 1-10, 10-100, ... , 10 000 000.

Atrybuty prywatne

- CNode ∗ root

4.5.1 Opis szczegółowy

definicja klasy drzewo Definiuje binarne drzewo zbudowane z wezlow galezi.

4.5.2 Dokumentacja konstruktora i destruktora

4.5.2.1 CTree::CTree ( )

definicja konstruktora bezparametrycznego Ustawia korzen na NULL.

korzen drzewa

4.5.2.2 CTree::∼CTree ( )

definicja destruktora klasy CTree wywoluje rekurencujnie destruktr klasy CNode dopuki nie usunie korzenia. Ustawia korzen na NULL.

4.5.3 Dokumentacja funkcji składowych

4.5.3.1 void CTree::add (int *k*)

definicja metdy add Tworzy nowy wezel, nastepnie dodaje go zgodnie z regula drzewa binarnego. Dodawany element zawsze staje sie lisciem.

Parametry

k	dodawana wartosc

4.5.3.2 int CTree::benchmark_add ( )

definicja metody benchmark_add Wywoluje metode fill_random zawiera petle benchmarkujace w zakresie 1-10, 10-100, ... , 10 000.

4.5.3.3 int CTree::benchmark_find ( )

definicja metody benchmark_find Wywoluje metode find_random. Zawiera petle benchmarkujace w zakresie 1-10, 10-100, ... , 10 000 000.

4.5.3.4 void CTree::fill_random (int *x*)

definicja metody fill_random Dodaje do drzewa liczby pseudolowe w zakresie 1-10. Metoda obsewowana przez klase CBenchmark Zapisuje do pliku timing.txt czas dodawania elementow.

Parametry

x	ilosc elementow pseudolosowych

4.5.3.5 CNode ∗ CTree::find (int *k*)

definicja metody find

Parametry

k	szukany klucz

Zwraca

wskaznik na wezel o szukanym kluczu.

4.5.3.6 void CTree::find_random (int *quantity*)

definicja metody find_random

Parametry

quantity	ilosc szukanych elementow Przeszukuje drzewo binarne. Metoda obserwowana przez klase CBenchmark. Zapisuje do pliku timing.txt czas przeszuiwania drzewa.

4.5.3.7 CNode * CTree::get_max ( )

definicja metody get_max

Zwraca

wezel o najwiekszym elemencie.

4.5.3.8 CNode * CTree::get_min ( )

definicja metody get_min

Zwraca

wezel o najmniejszym kluczu

4.5.3.9 void CTree::preorder ( ) const

definicja metody preorder wywoluje metode preorder klasy CNode Wyswietla kolejno lewa galez drzewa, a nastepnie prawa.

4.5.4 Dokumentacja atrybutów składowych

4.5.4.1 CNode* CTree::root [private]

Dokumentacja dla tej klasy została wygenerowana z plików:

- tree.hh
- tree.cpp

5 Dokumentacja plików

5.1 Dokumentacja pliku benchmark.cpp

```
#include "benchmark.hh"
#include <windows.h>
#include <fstream>
#include <iostream>
```

5.2 Dokumentacja pliku benchmark.hh

```
#include <windows.h>
```

Komponenty

- class CBenchmark

5.3 Dokumentacja pliku main.cpp

```
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <windows.h>
#include "tree.hh"
#include "benchmark.hh"
#include "node.hh"
#include "Red_black_node.hh"
#include "Red_black_tree.hh"
```

Funkcje

- int main ()

5.3.1 Dokumentacja funkcji

5.3.1.1 int main ( )

5.4 Dokumentacja pliku node.cpp

```
#include <iostream>
#include <cstdlib>
#include "tree.hh"
#include "benchmark.hh"
```

5.5 Dokumentacja pliku node.hh

```
#include "benchmark.hh"
```

Komponenty

- class CNode

  definicja klasy CNode Definijue pojedynczy wezel drzewa binarnego. Jest zaprzyjazniona z klasa CTree.

5.6 Dokumentacja pliku Red_black_node.cpp

```
#include "Red_black_node.hh"
#include <iostream>
```

5.7 Dokumentacja pliku Red_black_node.hh

```
#include <iostream>
```

Komponenty

- class CRed_Black_Node

  definicja klasy CRed_Black_Node definicja wezla drzewa czerwono-czarnego. Klasa zaprzyjazniona z klasa CRed_Black_Tree

5.8 Dokumentacja pliku Red_black_tree.cpp

```
#include <iostream>
#include "Red_black_tree.hh"
#include "Red_black_node.hh"
```

5.9 Dokumentacja pliku Red_black_tree.hh

```
#include "Red_black_node.hh"
#include "benchmark.hh"
```

Komponenty

- class CRed_Black_Tree

  definicja klasy CRed_Black_Tree Definiuje Drzewo czerwono - czarne Posiada wartownika na lisciach. Drzewo czerwono-czarne spelnia nastepujace wlasnosci 1) Każdy węzeł jest czerwony lub czarny. 2) Korzeń jest czarny. 3) Każdy liść jest czarny (Można traktować nil jako liść). 4) Jeśli węzeł jest czerwony, to jego synowie muszą być czarni. 5) Każda ścieżka z ustalonego węzła do liścia liczy tyle samo czarnych węzłów.

5.10 Dokumentacja pliku tree.cpp

```
#include "tree.hh"
#include <iostream>
```

5.11 Dokumentacja pliku tree.hh

```
#include "benchmark.hh"
#include "node.hh"
```

Komponenty

- class CTree

  definicja klasy drzewo Definiuje binarne drzewo zbudowane z wezlow galezi.

Projektowanie algorytmów i metod sztucznej inteligencji
Laboratorium 8- Sprawozdanie

Wojciech Makuch

6 Zadanie

Implementacja drzewa binarnego oraz drzewa czerwono-czarnego. Zbadanie złożoności obliczeniowej metody wstawiania oraz przeszukiwania.

7 Wstep

Binarne drzewo przeszukiwań – struktura danych przechowująca dane oraz 2 wskaźniki na elementy następne zwane prawym oraz lewym synem. Koszt wykonania podstawowych operacji w drzewie BST jest proporcjonalny do wysokości drzewa h, ponieważ operacje wykonywane są wzdłuż drzewa. Jeżeli drzewo jest zrównoważone, to jego wysokość bliska jest logarytmowi dwójkowemu liczby węzłów. Złożoność obliczeniowa wynosi wtedy $O(log_2 n)$. Z drugiej strony drzewo skrajnie niezrównoważone ma wysokość porównywalną z liczbą węzłów (w skrajnym przypadku drzewa zdegenerowanego do listy wartości te są równe: h=n), z tego powodu koszt pesymistyczny wzrasta do $O(n)$.

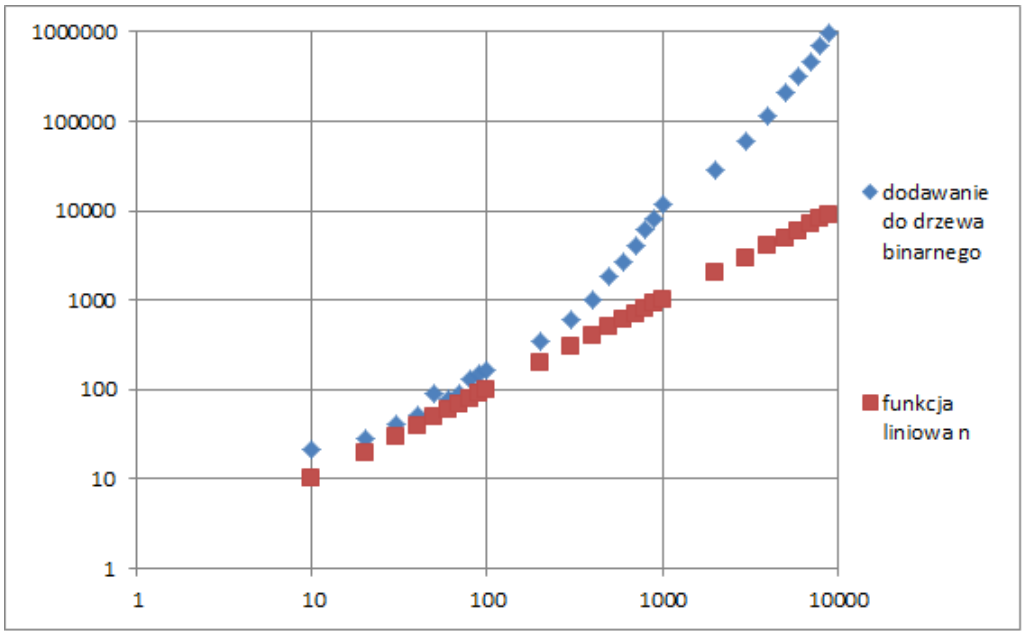
8 Drzewo binarne - implementacja

Utworzono klasę węzła posiadającą wskaźniki na ojca, lewego syna i prawego syna oraz klasę drzewa składającego się z węzłów przechowującą wskaźnik na korzeń. Zaimplementowano podstawowe metody takie jak dodawanie elementu zgodnie z zasadą drzewa binarnego, wyświetlanie zawartości drzewa zgodnie z metodą preorder oraz przeszukiwanie drzewa w celu znalezienia węzła o zadanym kluczy. Ponadto klasa jest obserwowana przez klasę benchmarkujące oraz posiada metody zawieracjące pętle benchmarkujące.

9 Drzewo czerowno-czarne

Drzewo czerwono-czarne – drzewo binarne dążące do zrównoważenie się, dzięki czemu głębokość jest mniejsza niż w zwykłym drzewie binarnym. Zachowuje własności:

1. Każdy węzeł jest czerwony lub czarny.
2. Korzeń jest czarny.
3. Każdy liść jest czarny (Można traktować nil jako liść).
4. Jeśli węzeł jest czerwony, to jego synowie muszą być czarni.
5. Każda ścieżka z ustalonego węzła do liścia liczy tyle samo czarnych węzłów



Rysunek 1: Dodawanie do drzewa binarnego.

10 Drzewa czerwono-czarne - implementacja

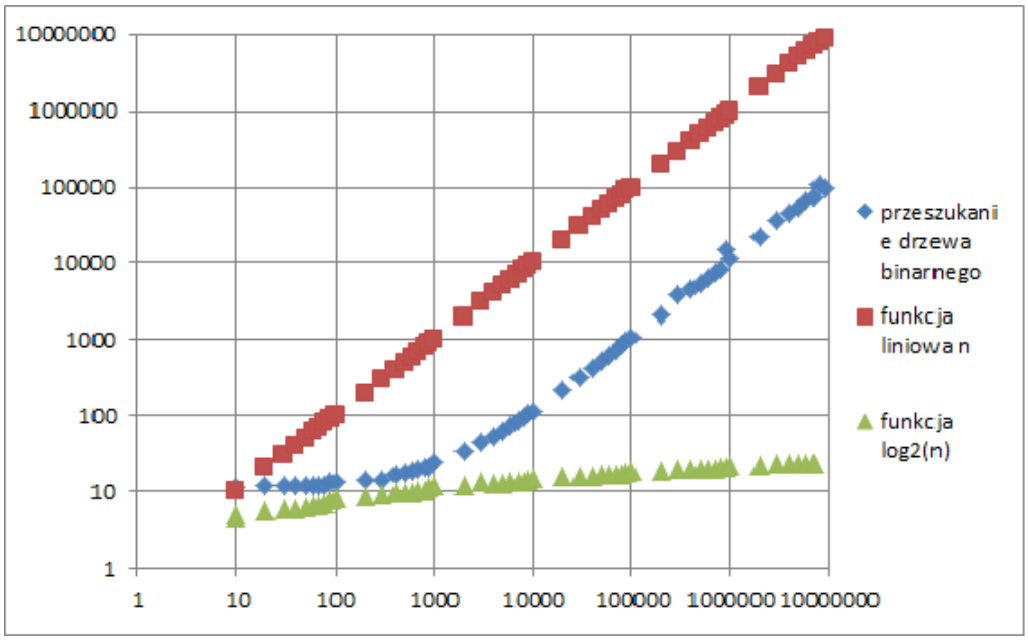
Implementacje drzewa czerwono – czarnego wykonano analogicznie do drzewa binarnego. Klasa węzeł zawiera dodatkowe pole informujące o kolorze. Ponadto zastosowano wartownika informujący kiedy następują liście nie przechowujące żadnej wartości. Zaimplementowano metody rotacji w prawo oraz rotacji w lewo w wykorzystane w metodzie dodawania elementów, aby zrównoważyć drzewo. Metoda wstawiania elementu do drzewa czerwono – czarnego jest rozbudowaniem metody wstawiania do zwykłego drzewa binarnego sprawdzająca 5 przypadków i wykonująca następujące kroki:

1. Gdy ojciec i wuj sa czerwoni, zmien ich kolory.
2. Gdy lewy ojeciec i jego lewy syn sa czeroni, wykonaj obrot w prawo dziadka i ojca oraz zamien ich kolory.
3. Gdy lewy ojciec i jego prawy syn sa czerwoni, wykonaj obrot w lewo ojca i syna oraz wykonaj 2).
4. Gdy prawy ojciec i jego prawy syn sa czerwoni, wykonaj obrot w lewo dziadka i ojca, zamien ich kolory
5. Gdy prawy ojciec i jego prawy syn sa czerwoni, wykonaj obrot w prawo ojca i syna oraz wykonaj 4)

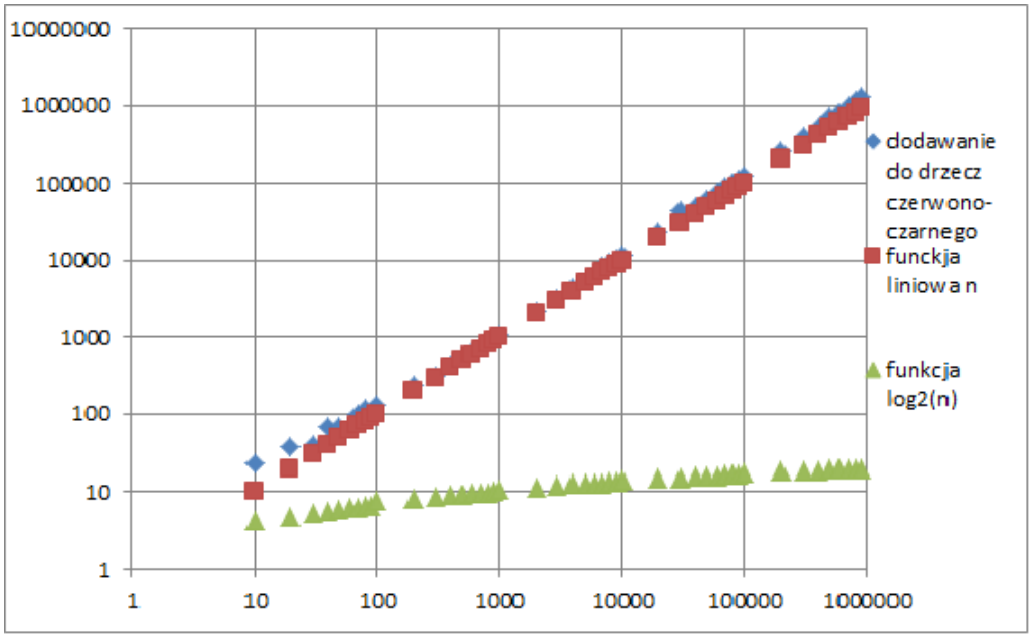
Ponadto klasa zawiera metody benchmarkujące analogicznie jak w drzewie binarnym.

11 Zlozonosci obliczeniowe usyskane

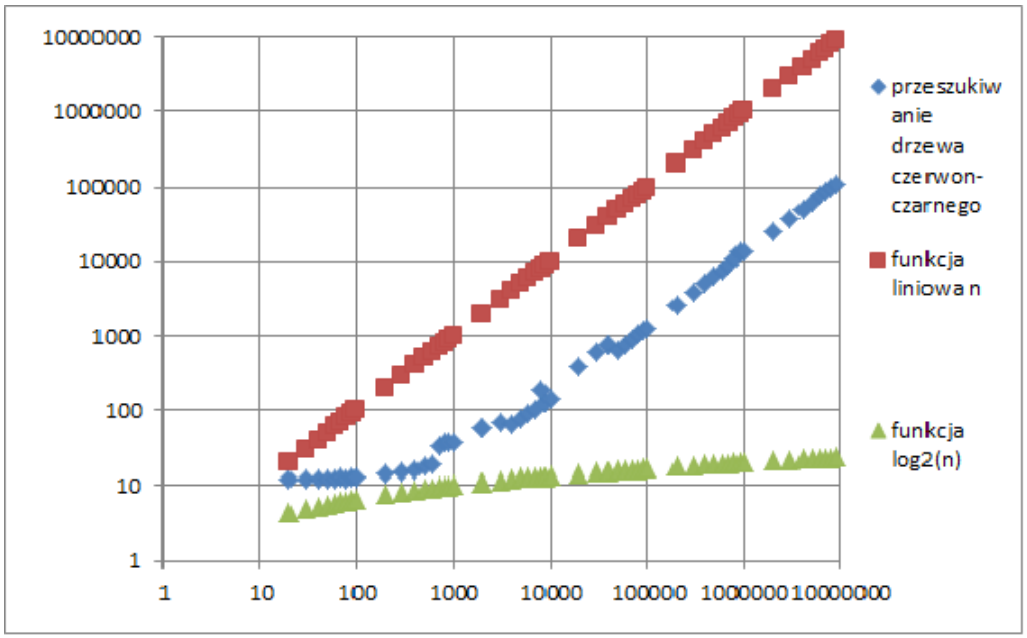
Na rysunkach poniżej przedstawiono uzyskane wyniki złożoność obliczeniowych. Analizując je po kolei można dojść do wniosku, że wstawiania do drzewa binarnego ma pesymistyczną złożoność obliczeniowa $O(n)$. Wykres nieco ucieka od tej wartości, co może być spowodowane niedokładną implementacją algorytmu(np. brak wartownika przy zwykłym drzewie binarnym). Najlepsze wyniki dają algorytmy wyszukiwania w drzewie binarnym oraz binarnym czerwono –czarnym, ponieważ tam działają tylko operatory porównania. Uzyskana złożoność obliczeniowa wynosi $O(log_2 n)$, w przypadku pesymistycznym $O(n)$. Algorytm wstawiania do drzewa czerwono-czarnego ma posiada lepszą złożoność obliczeniową, niż w przypadku zwykłego drzewa binarnego, ale wynosi ona w przypadku pesymistycznym również $O(n)$.



Rysunek 2: Przszeszukiwanie drzewa binarnego.



Rysunek 3: Dodawanie do drzewa czerwono-czarnego.



Rysunek 4: Przeszukiwanie drzewa czerwono-czarnego.

12 Komentarz

Do utworzenia dokumentacji wykorzystano system Doxygen. Funkcja pomiaru czasu dla systemu Windows pobrana ze strony dr. J. Mierzwy. Program skompilowano w środowisku Code::Blocks. Do stworzenia wykresu posłużono się pakietem MS Excel, sprawozdanie napisano używając systemu LaTeX.