Struktury Baz Danych Projekt 2 - Sprawozdanie z indeksowej organizacji plików z użyciem struktury B-drzewa

Kamil Szabłowski, 193248

December 10, 2024

1 Wprowadzenie

Zadaniem projektu była implementacja jednego z wybranych algorytmów organizacji plików indeksowych. Zaimplementowanym przeze mnie algorytmem jest algorytm *B-drzewa*.

Użytym w projekcie typem rekordu pliku są ciągi 10 *liczb* z dodatkowycm kluczem, będącym liczbą natruralną większą od 0. Kryterium sortowania rekordu jest wartość klucza. Za *liczbę* będącą elementem rekordu uznaję 32-bitową liczbę całkowita ze znakiem. Poniżej przedstawione został przykład takiego rekordu:

1	23	−45	67	12	−8	34	0	29	−16	5

Pierwszy element powyższego rekordu to kluczz wynoszący 1, reszta stanowi ciąg 10 liczb.

Rozmiar rekordu wynosi w takim razie 44 bajtów (11 liczb całkowitych 4-bajtowych). Przyjęty rozmiar bloku dyskowego pliku danych na potrzeby tego eksperymentu wynosi 364 bajtów.

2 Opis implementacji

Użytą w tym projekcie metodą organizacji pliku jest użycie struktury B-drzewa.

Alogorytmy wykorzystane do wstawianie, usuwania i wyszukiwania rekordów w B-drzewie są analogiczne do tych przedstawionych na wykładzie. Modyfikacja rekordu polega na usunięciu starego rekordu i dodaniu nowego rekordu z nowym kluczem, chyba że nie dokonujemy zmiany klucza, wtedy modyfikowana jest wartość rekordu bezpośrednio w pliku z danymi, po uwczesnym wyszukaniu klucza.

W mojej implementacji na stronę B-drzewa zapisaną do pliku składają się:

- Wskaźnik na stronę rodzica (liczba całkowita, 4B)
- Liczba rekordów w stronie (liczba całkowita, 4B)

- Lewy skrajny wskaźnik na stronę, z rekordami mniejszymi od wszystkich innych należących do obecnej strony (liczba całkowita, 4B)
- Od 1 do $2d$ węzłów, gdzie na dany węzeł składa się:
  - Wartość klucza rekordu (liczba całkowita, 4B)
  - Offset rekordu o danym kluczu w pliku z danymi (liczba całkowita, 4B)
  - Wskaźnik na stronę z rekordami większymi od klucza w tym węźle, ale mniejszymi od klucza w następnym węźle (liczba całkowita, 4B)

Nie zależnie od ilości węzłów w pliku indeksowym B-drzewa, zapisana (lub odczytana) strona z dysku ma zawsze rozmiar $4 + 4 + 4 + 2d * 12 = 12 + 24d$ bajtów.

Strona w pamięci w programie jest prawie identyczna jak strona zapisana na dysku, z tą różnicą, że lewy skrajny wskaźnik na stronę jest zastąpiony dodatkowym węzłem 0, w którym poza wartością wskaźnika na stronę-dziecko, wartość klucza i rekordu ustawiona jest na NULL

Jeśli chodzi o plik z danymi, to podobnie jak w przypadku pierwszego projektu, rekordy są zapisywane binarnie, jeden po drugim, bez żadnych dodatkowych informacji.

2.1 Opis buforowania

3 Specyfikacja plików testowych

Plikiem testowym jest plik tekstowy (.txt) o określonej poniżej strukturze.

3.1 Struktura pliku testowego

Plik testowy składa się z komend oddzielonych znakime nowej linii.

Komendy dostępne do użycia w pliku testowym są analogiczne co do komend dostępnych w interfejsie tekstowym. Lista dostępnych komend może zostać wyświetlona przy użyciu komendy `help`. Dodatkowe komendy do debuggowania programu są pokazane po użyciu komendy `help debug`.

Output komendy `help`:

Output komendy `help`

```
List of available commands
TIP: Most commands can be used by using
first letters of each word in the command
Example: 'am 5' is the same as 'addmulti 5'
--------------------------------------------------------------------
help                                        Show this help message
help debug                                  Show help relating to debug
setcompensation [true/false]                Toggle compensation
clear                                       Clear all the files
rand [n]                                    Insert n random records
update [key] [value] [newKey: OPTIONAL]     Update record with given key
insert [key] [value]                        Insert record into the file
search [key]                                Search for record with key
print [group] [all]                         Prints all records in db
loadtest [filename]                         Loads test file
```

Output komendy `help debug`:

Output komendy `help debug`

```
List of available debug commands
--------------------------------------------------------------------
dblockstats             Prints block stats
dforceflush             Forces a flush of the files
dgetrecord [n]          Gets a record at offset (data file)n
```

Przykład poprawnie ustrukturyzowanego pliku testowego:

Plik: `text.txt`

```
clear
insert 1 5
insert 2 10
insert 3 15
insert 4 20
insert 5 25
delete 2
search 3
search 2
dblockstats
print all group
print
```

Poniższy plik testowy wykona dokładnie to samo co powyższy:

Plik: `text.txt`

```
c
i 1 5
i 2 10
i 3 15
i 4 20
i 5 25
d 2
s 3
s 2
dbs
p a g
p
```

Tak spreparowany plik może bezproblemu zostać załadowany do interfejsu tekstowego.

3.2 Załadowanie i uruchomienie pliku testowego

Aby załadować plik testowy z dysku, wystarczy w interfejsie tekstowym użyć komendy `loadtest nazwa_pliku.txt`, gdzie zamiast *nazwa_pliku.txt* należy wpisać ścieżkę do docelowego pliku testowego, który ma zostać załadowany. Ścieżka może być bezwzględna, lub względna - zależna od katalogu w którym został uruchomiony interfejs tekstowy.

4 Sposób prezentacji wyników działania programu

Po uruchomieniu programu, organizowany plik indeksowy orazz z danymi jest pusty i nie zawiera żadnych rekordów ani stron.

Aby dodać rekordy do bazy danych należzy użyć jednej z nastepujących komend:

- `insert [key] [value]` - próbuje dodać rekord o danym kluczu i wartości do bazy danych
- `rand [n]` - próbuje dodać n losowych rekordów do bazy danych

Analogicznie, usunięcie rekordu z bazy danych odbywa się przy użyciu komendy `delete [key]`.

Wyszukiwanie rekordu w bazie danych odbywa się przy użyciu komendy `search [key]`.

Modyfikacja rekordu w bazie danych odbywa się przy użyciu komendy `update [key] [value] [new`

Po użyciu komendy `print` bez żadnych parametrów, program wypisuje zawartość bazy danych w formie rekordów, od najmniejszego do największego klucza w następującym formacie: Przykład:

Output komendy `print`

```
(1) 10 10 10 10 10 10 10 10 10 10
(2) 20 20 20 20 20 20 20 20 20 20
(3) 30 30 30 30 30 30 30 30 30 30
(4) 40 40 40 40 40 40 40 40 40 40
(5) 50 50 50 50 50 50 50 50 50 50
```

Liczba w nawiasie to klucz rekordu, podczas gdy liczby po spacji to wartości rekordu, czyli liczby należące do ciągu.

Po użyciu komendy `print` z parametrem `group`, program wypisuje zawartość bazy danych w formie rekordów, ale tym razem grupując rekordy znajdujące się na jednej stronie B-drzewa.

Po użyciu komendy `print` z parametrem `all`, program wypisuje zawartość bazy danych w pełnej formie. Przykład:

Output komendy `print all`

```
RecordOffset: 0 | PageOffset: 0 | ParentPageOffset: 1
| LeftPagePtr: -1 | RightPagePtr: -1
(1) 1 1 1 1 1 1 1 1 1 1

RecordOffset: 1 | PageOffset: 0 | ParentPageOffset: 1
| LeftPagePtr: -1 | RightPagePtr: -1
(2) 2 2 2 2 2 2 2 2 2 2

RecordOffset: 2 | PageOffset: 1 | ParentPageOffset: -1
| LeftPagePtr: 0 | RightPagePtr: 2
(3) 3 3 3 3 3 3 3 3 3 3

RecordOffset: 3 | PageOffset: 2 | ParentPageOffset: 1
| LeftPagePtr: -1 | RightPagePtr: -1
(4) 4 4 4 4 4 4 4 4 4 4

RecordOffset: 4 | PageOffset: 2 | ParentPageOffset: 1
| LeftPagePtr: -1 | RightPagePtr: -1
(5) 5 5 5 5 5 5 5 5 5 5

-----------------------------------
Total records: 5
Total pages count: 3
Height: 2
-----------------------------------
```

Poza wyświetleniem klucza i wartości rekordu, program wypisuje również informacje o stronach B-drzewa, takie jak:

- `RecordOffset` - offset rekordu w pliku z danymi
- `PageOffset` - offset strony w pliku indeksowym
- `ParentPageOffset` - offset strony rodzica w pliku indeksowym
- `LeftPagePtr` - offset strony dziecka z rekordami mniejszymi od klucza w obecnym węźle
- `RightPagePtr` - offset strony dziecka z rekordami większymi od klucza w obecnym węźle

Wartość -1 w polach `LeftPagePtr`, `RightPagePtr` oraz `ParentPageOffset` oznacza, że dany wskaźnik nie wskazuje na nic, czyli węzeł nie ma dzieci, bądź rodzica (tylko korzeń).

Dodatkowo wyświetlana jest informacja o całkowitej liczbie rekordów, liczbie stron oraz wysokości drzewa.

Po użyciu komendy `print all group` program wypisuje zawartość bazy danych w pełnej formie, ale tym razem grupując rekordy po stronach B-drzewa. Przykład:

Output komendy `print all group`

```
-------- Page 0 ----------
RecordOffset: 0 | PageOffset: 0 | ParentPageOffset: 1
| LeftPagePtr: -1 | RightPagePtr: -1
(1) 1 1 1 1 1 1 1 1 1 1

RecordOffset: 1 | PageOffset: 0 | ParentPageOffset: 1
| LeftPagePtr: -1 | RightPagePtr: -1
(2) 2 2 2 2 2 2 2 2 2 2

-------- Page 1 ----------
RecordOffset: 2 | PageOffset: 1 | ParentPageOffset: -1
| LeftPagePtr: 0 | RightPagePtr: 2
(3) 3 3 3 3 3 3 3 3 3 3

-------- Page 2 ----------
RecordOffset: 3 | PageOffset: 2 | ParentPageOffset: 1
| LeftPagePtr: -1 | RightPagePtr: -1
(4) 4 4 4 4 4 4 4 4 4 4

RecordOffset: 4 | PageOffset: 2 | ParentPageOffset: 1
| LeftPagePtr: -1 | RightPagePtr: -1
(5) 5 5 5 5 5 5 5 5 5 5

----------------------------------
Total records: 5
Total pages count: 3
Height: 2
----------------------------------
```

Jak widać powyżej, rekordy są grupowane po stronach B-drzewa, co w przypadku większej ilości rekordów może spowodować, że rekordy nie są wyświetlone w kolejności rosnącej klucza.

5 Eksperyment

5.1 Konstrukcja eksperymentu

Do przeprowadzenia eksperymentu wykorzystane zostały funkcje udostępniane prze interfejs tekstowy programu.

Schemat eksperymentu wyglądał następująco:

1. Ustawienie parametru D B-drzewa zgodnie z przeprowadzanym eksperymentem i kompilacja programu
2. Wprowadzenie komend ręcznie do interfejsu tekstowego
3. Spisanie informacji o każdej operacji do pliku tekstowego
4. Wygenerowanie wykresów na podstawie danych eksperymentalnych
5. Analiza otrzymanych wyników

5.2 Przygotowania do eksperymentu

Zmiana parametru D B-drzewa odbywa się poprzez zmianę wartości stałej zdefiniowanej w pliku `defines.h` jako `BTREE_D_FACTOR`.

Eksperyment polegający na wstawieniu N losowych rekordów do pliku, i spisaniu ich wyników został przeprowadzony przez wpisanie kolejnych komend:

```
clear
rand N
dblockstats
print all group
```

Gdzie N to liczba rekordów, którą chcemy wstawić do pliku.

Na potrzeby tego eksperymentu, N przyjmowało wartości od $N = \{250, 1000, 2500, 10000, 25000\}$, a współczynnik D B-drzewa przyjmował wartości $D = \{2, 4, 6, 10, 20, 50\}$.

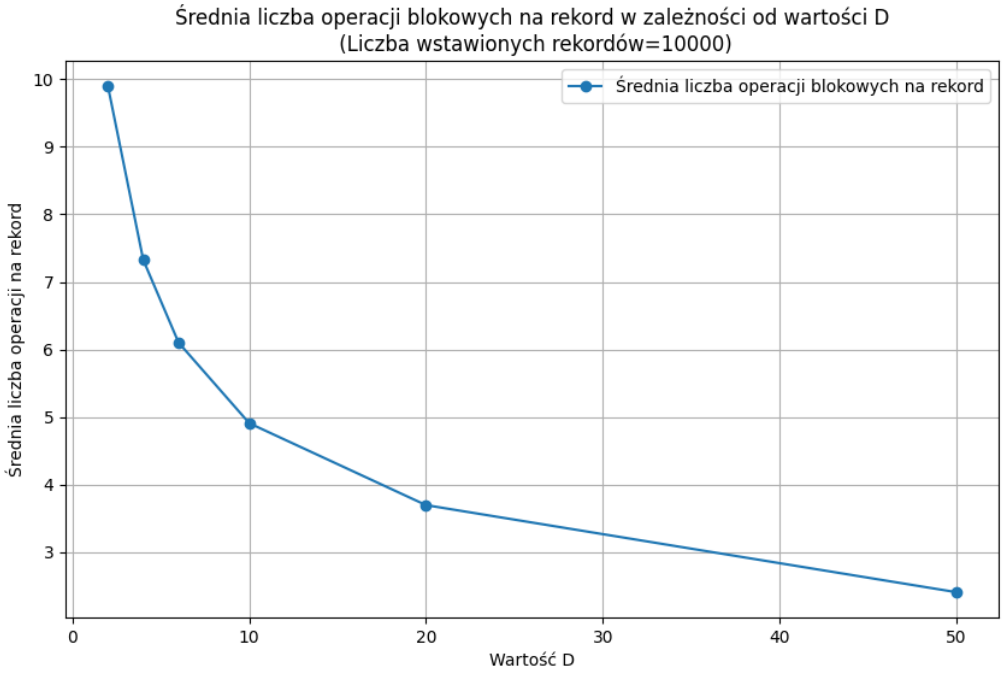
Wyniki eksperymentu zostały zapisane do pliku tekstowego `eksperyment.txt`. Wyniki te następnie zostały użyte do utwożenia wykresów i tabel, które zostały przedstawione w kolejnej sekcji.

5.3 Wyniki eksperymentu

5.3.1 Liczba operacji blokowych

Tabela 1: Średnia liczba operacji blokowych dla różnych wartości D, przy wstawianiu 10000 rekordów

D	Odczyty	Zapisy	Średnia liczba operacji na rekord	Wysokość
2	68451	30452	9.8982	7
4	49086	24137	7.3252	5
6	39877	21114	6.1009	4
10	30853	18198	4.9080	4
20	21809	15135	3.6959	3
50	12230	11838	2.4075	3



Wykres 1: Średnia liczba operacji blokowych przypadających na pojedynczy rekord

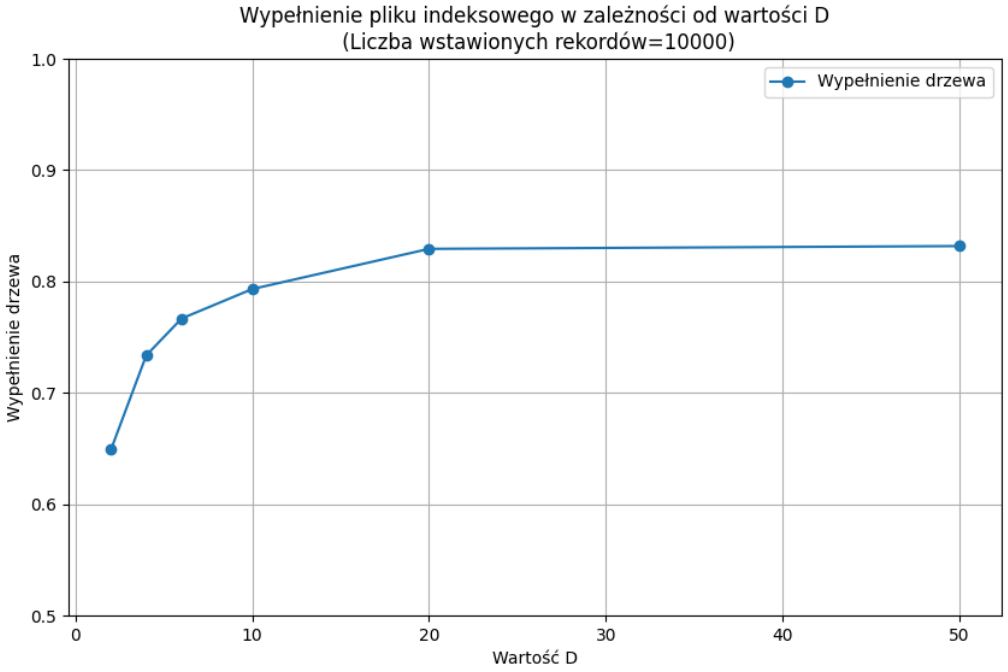
Jak widać z powyższego wykresu, zwiększenie wartości D B-drzewa powoduje zmniejszenie liczby odczytów i zapisów blokowych przypadających na pojedynczy rekord, co świadczy o lepszej efektywności działania B-drzewa.

Z tabelki można również przeczytać, że zwiększenie wartości D B-drzewa powoduje zmniejszenie wysokości drzewa. Biorąc to pod uwagę można stwierdzić, że średnia liczba operacji blokowych na B-drzewie jest odwrotnie proporcjonalna do wysokości tego drzewa.

5.3.2 Wypełnienie B-drzewa na dysku

Tabela 2: Wypełnienie pliku indeksowego B-drzewa na dysku dla różnych wartości D, przy wstawianiu 10000 rekordów

D	Rozmiar rekordu (B)	Całkowity rozmiar drzewa(B)	Wypełnienie drzewa (%)
2	119904	184800	64.88
4	119952	163512	73.36
6	119964	156468	76.67
10	119928	151200	79.32
20	119952	144648	82.93
50	119964	144228	83.18



Wykres 2: Wypełnienie pliku indeksowego na dysku

Wyniki można odczytać tak, że na jeden rekord faktycznie zapisany w strukturze logicznej B-drzea przypada X% rekordu zapisanego na dysku.

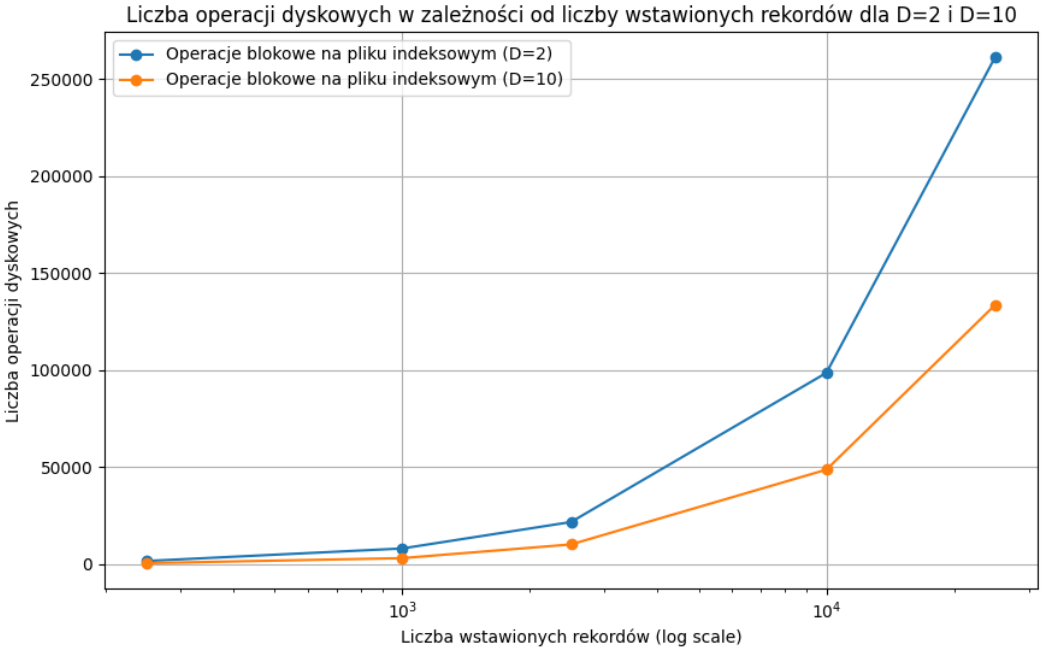
Z powyższego wykresu można zauważyć, że zwiększenie wartości parametru D B-drzewa powoduje również zwiększenie poziomu wypełnienia pliku indeksowego na dysku. Wzrost ten jest znaczący dla małych wartości D, ale dla większych wartości D, dochodzi do stagnacji wypełnienia pliku na dysku.

Wynika z tego, że optymalne wartości parametru D dla B-drzewa nie powinny być bardzo małe, gdyż wtedy plik indeksowy na dysku jest wypełniony bardziej nieefektywnie, niż dla dużych waartości.

5.3.3 Ilość dodawanych rekordów

Tabela 3: Liczba operacji blokowych dla różnych ilości rekordów

Liczba rekordów	D = 2	D = 10
250	1598	472
1000	8045	3067
2500	21675	10126
10000	98545	48649
25000	261268	133576



Wykres 3: Liczba operacji blokowych w zależności od ilości rekordów

Jak widać na powyższym wykresie, zzarówno liczba dodawanych rekordów jak i parametr D ma wpływ na liczbę wykonywanych operacji blokowych na B-drzewie.

Zwiększenie liczby rekordów powoduje szybszy od liniowego wzrost liczby operacji blokowych, co można wyjaśnić faktem, że dla większej ilości rekordów w B-drzewie, istnieje możliwość że wysokość drzewa wzrosła, a operacje na większym drzewie wymagają przynajmniej o jedną więcej operację blokową na operacje na rekordzie.

Można też zauważyć, że dla większego D=10, liczba operacji blokowych jest mniejsza od D=2, ale mimo to kształt obu wykresów jest bardzo podobny.

5.4 Podsumowanie