STRUKTURY BAZ DANYCH PROJEKT

Sortowanie przez scalanie z wykorzystaniem wielkich buforów.

Jędrzej Jasiniecki 197993

1. Opis Metody:

❖ Informacje ogólne:

W ramach tego projektu napisałem program sortujący plik przy pomocy algorytmu sortowania zewnętrznego metodą scalania z użyciem wielkich buforów. Metoda ta ma na celu minimalizację liczby operacji dyskowych przez maksymalne wykorzystanie dostępnej pamięci RAM. Metoda ta jest konieczna gdy sortowany plik jest za duży żeby go zmieścić w całości w pamięci RAM.

❖ Opis algorytmu:

Na wejściu dostajemy ilość dostępnych buforów (n), ilość rekordów (N), oraz współczynnik blokowania (b), czyli ilość rekordów na stronę.
Algorytm składa się z dwóch etapów:

- **Tworzenie serii**
  Dzielimy plik na N/(n*b) posortowanych serii.
  1,.Do pamięci RAM wczytujemy n stron z pliku które tworzą blok danych mieszczący n*b rekordów.
  2.Następnie blok ten sortujemy za pomocą *QuickSort*
  3. Posortowaną serię zapisujemy na dysk.
  Powtarzamy kroki 1-3 aż skończą się dane w pliku wejściowym
- **Scalanie serii**
  Mając N/n*b posortowanych serii teraz będziemy je scalać aż dostaniemy posortowany plik.
  1. Wyznaczamy n-1 buforów do scalania oraz 1 bufor wyjściowy
  2.Inicjalizujemy bufory do scalania, wczytując do nich pierwszą stronę z każdej serii

3. Następnie inicjalizujemy kopiec minimalny i wrzucamy tam pierwszy rekord z dostępnych plików, dzięki czemu zawsze mamy dostęp do najmniejszego elementu
4. Dopóki kopiec nie jest pusty to: wrzucamy rekord ze szczytu kopca do buforu wyjściowego. Jeżeli bufor wyjściowy jest pełen to zapisujemy go do pliku, następnie do kopca dokładamy następny rekord z serii z której rekord został przesłany do bufora wyjściowego i jeżeli przejrzeliśmy całą stronę w buforze scalania to pobieramy następną.
5.Gdy liczba serii jest większa od n-1 to powtarzamy scalanie wielofazowo. Pliki wyjściowe z jednej fazy stają się plikami wejściowymi dla kolejnej. PRoces kończy się, gdy na dysku pozostanie tylko jeden plik.

❖ Liczba zastosowanych taśm

Etap 1: Używane są 2 taśmy (taśma z danymi wejściowymi i taśma wyjściowa do zapisu kolejnych serii)
Etap 2: używane jest n taśm (n-1 taśm do czytania serii i 1 taśma wyjściowa)

❖ Pliki testowe

Rekordy składają się z 3 doubli: masa (M),ciepło właściwe (c) , oraz różnica temperatur (T), a sortujemy po cieple (Q) ze wzoru:
$Q = T * c * M$
Jeden rekord zajmuje 24 b.

2. Opis eksperymentu:

Eksperyment będzie przeprowadzany na następujących danych:

- n(liczba buforów) = 10
- b(współczynnik blokowania) = 100

❖ Obliczenia teoretyczne:

Wzory:

- $Liczba\ stron\ =\ [N/b]$
- $Liczba\ serii\ =\ [N/(n\ *\ b)]$
  *UZASADNIENIE: bo każda seria składa się z n*b rekordów*
- $Liczba\ faz\ scalania\ =\ [log_n(N/(n\ *\ b))]$

  *UZASADNIENIE: w każdej fazie mamy N/(n*b) serii, z których dostajemy n-1 zmergowanych serii, a to robimy aż dostaniemy 1 plik, i takich operacji będzie właśnie* $[log_{n-1}(N/(n\ *\ b))]$ *, dla dużych buforów można przybliżać do* $[log_n(N/(n\ *\ b))]$

- Całkowita ilość zapisów i odczytywania z dysku:
  $(koszt) = koszt\ etapu\ 1 + koszt\ etapu\ 2$.
  gdzie:
  $koszt\ etapu\ 1 = 2 * (N/b)$, ponieważ każdą stronę musimy pobrać i zapisać
  $koszt\ etapu\ 2 = log_n(N/(n * b)) * (2 * N/b)$, ponieważ w każdej fazie każdą stronę musimy pobrać i zapisać
  $KOSZT\ CAŁKOWITY = 2 * N/b + 2 * N/b * log_n(N/(n * b)) =$
  $= 2 * N/b\,(1 + log_n(N/n * b) = 2 * N/b(log_n n + log_n(N/n * b) =$
  $= 2 * N/b * (log_n N/b) = 2 * N/(b * log_2 n) * log_2(N/b)$

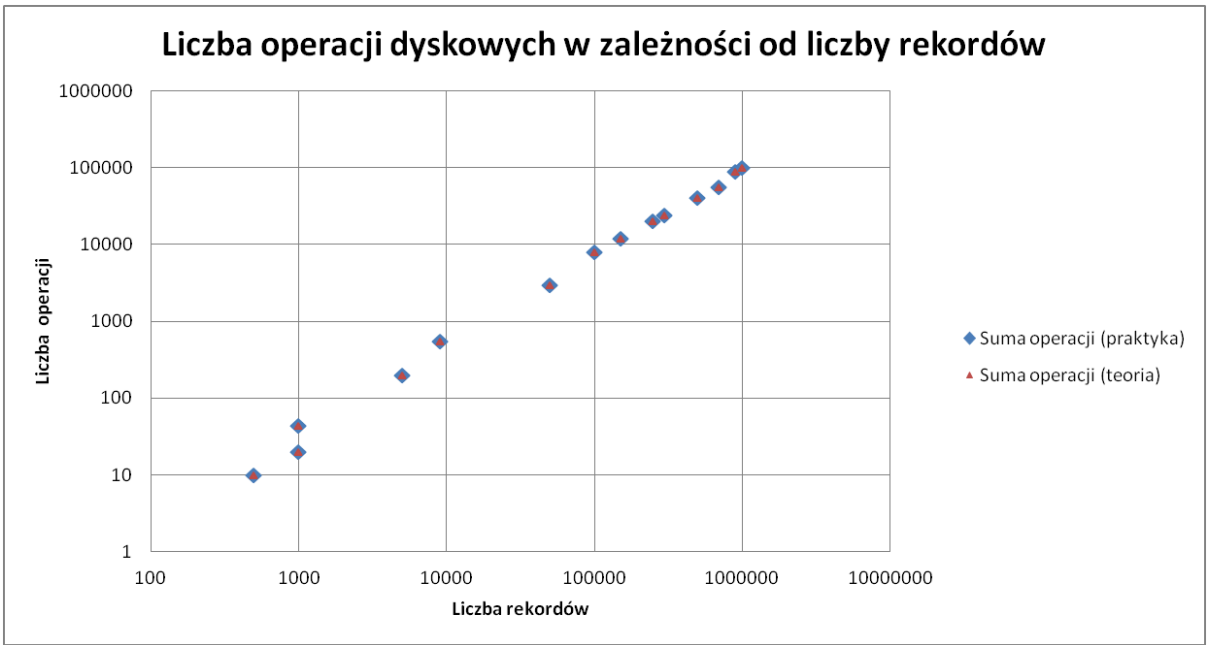
- Dla porównania będę również podawał dokładny koszt i liczbę faz (bez przybliżenia) czyli :
  $Liczba\ faz\ scalania = log_{n-1}(N/(n * b))$
  $Koszt\ Całkowity = 2 * N/b * (1 + log_{n-1}(N/n * b))$

❖ TEST 1

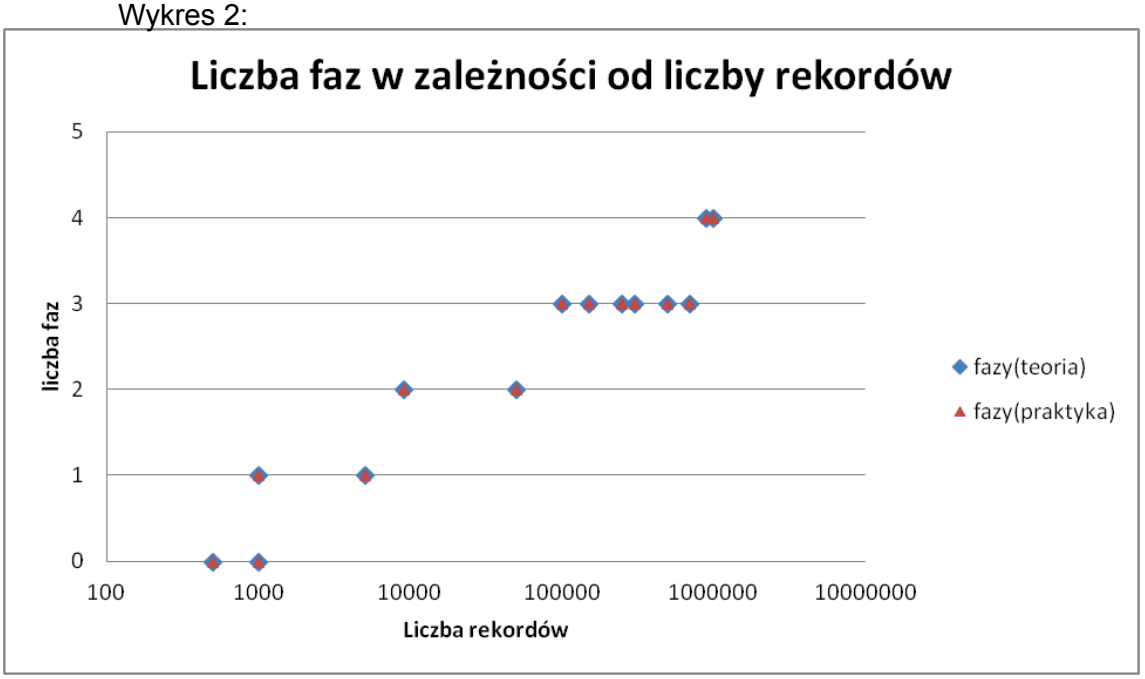
dla n= 10 i b=100
Dane 1:

	A	B	C	D	E	F	G	H	I	J
1	n=	10								
2	b=	100								
3										
4	licz. Rekordów	licz. Stron	liczb. Serii	fazy	odczyty(praktyka)	zapisy(praktyka)	suma operacji I/O	koszt teoria przyb.	koszt teoria dok.	fazy(praktyka)
5	500	5	1	0	5	5	10	7	10	0
6	1000	10	1	0	10	10	20	20	20	0
7	1001	11	2	1	22	22	44	21	44	1
8	5000	50	5	1	100	100	200	170	200	1
9	9001	91	10	2	273	273	546	352	546	2
10	50000	500	50	2	1500	1500	3000	2699	3000	2
11	100000	1000	100	3	4000	4000	8000	6000	8000	3
12	150000	1500	150	3	6000	6000	12000	9529	12000	3
13	250000	2500	250	3	10000	10000	20000	16990	20000	3
14	300000	3000	300	3	12000	12000	24000	20863	24000	3
15	500000	5000	500	3	20000	20000	40000	36990	40000	3
16	700000	7000	700	3	28000	28000	56000	53832	56000	3
17	900000	9000	900	4	45000	45000	90000	71177	90000	4
18	1000001	10001	1001	4	50005	50005	100010	80001	100010	4

Wykres 1:



Wykres 2:



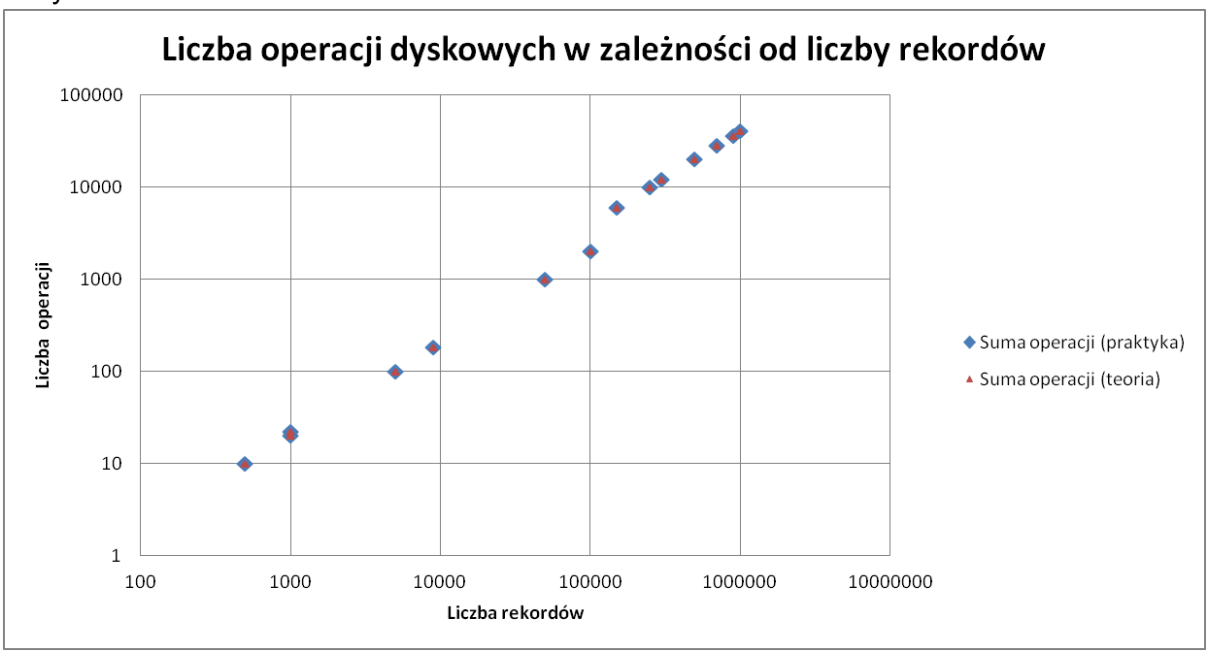
❖ TEST 2

dla n = 1000 i b =100

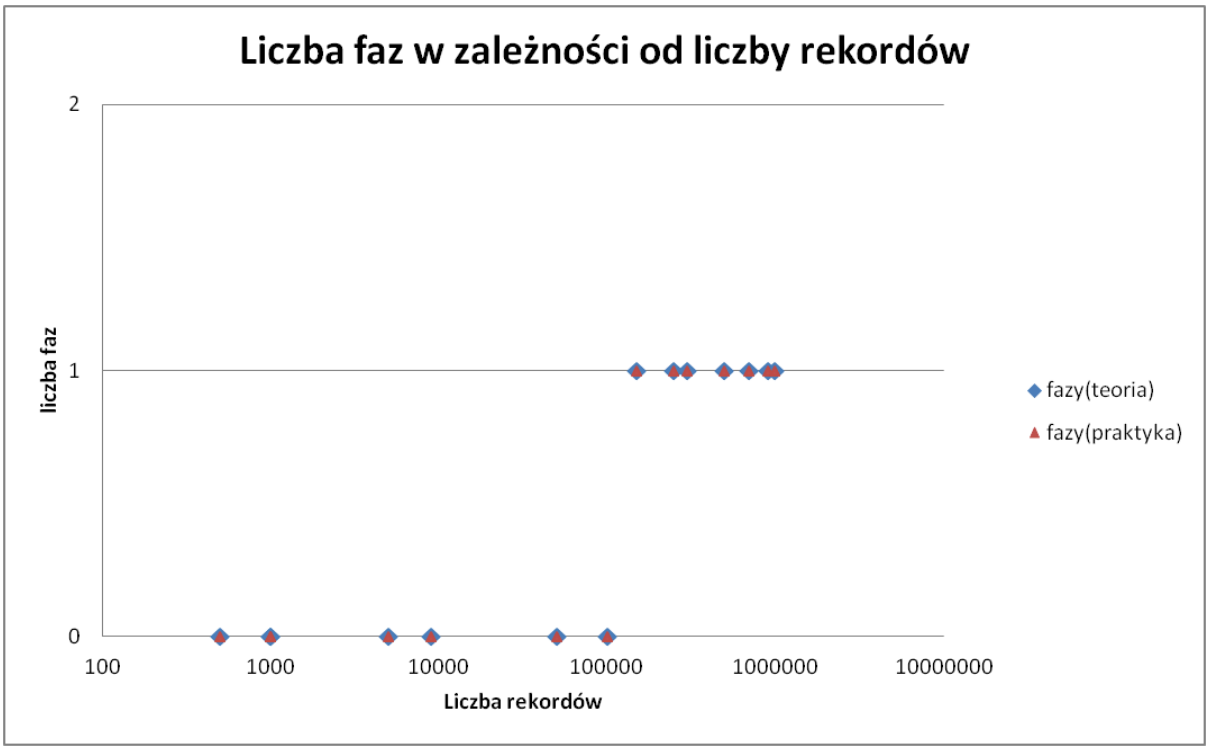
Dane 2:

	A	B	C	D	E	F	G	H	I	J
1	n=	1000								
2	b=	100								
3										
4	licz. Rekordów	licz. Stron	liczb. Serii	fazy	odczyty(praktyka)	zapisy(praktyka)	suma operacji I/O	koszt teoria przyb.	koszt teoria dok.	fazy(praktyka)
5	500	5	1	0	5	5	10	3	10	0
6	1000	10	1	0	10	10	20	7	20	0
7	1001	11	1	0	11	11	22	7	22	0
8	5000	50	1	0	50	50	100	57	100	0
9	9001	91	1	0	91	91	182	118	182	0
10	50000	500	1	0	500	500	1000	900	1000	0
11	100000	1000	1	0	1000	1000	2000	2000	2000	0
12	150000	1500	2	1	3000	3000	6000	3177	6000	1
13	250000	2500	3	1	5000	5000	10000	5664	10000	1
14	300000	3000	3	1	6000	6000	12000	6955	12000	1
15	500000	5000	5	1	10000	10000	20000	12330	20000	1
16	700000	7000	7	1	14000	14000	28000	17944	28000	1
17	900000	9000	9	1	18000	18000	36000	23726	36000	1
18	1000001	10001	11	1	20002	20002	40004	26667	40004	1

Wykres 3:



Wykres 4:



❖ Omówienie wykresów:

Na zebranych wynikach, przedstawionych na wykresach 2 i 4, zauważyć można, że kluczowy wpływ na wydajność sortowania ma liczba dostępnych buforów n. Wykresy faz dla obu eksperymentów (n=10 i n=1001) rosną logarytmicznie, jednak w zupełnie innej skali.

Dla małej liczby buforów (n=10), liczba faz rośnie w wyraźny sposób "schodkowy". Każdy stopień na wykresie pojawia się w momencie, gdy liczba serii początkowych przekracza kolejną potęgę ilości wektorów mergujących w 2 fazie algorytmu = n-1. Przykładowo, dla N=9000 (n-1=9) wystarcza 1 faza scalania, ale już dla N=9001 (n-1=10) konieczne są 2 fazy.W przypadku dużej liczby buforów (n=1001)**,** i n-1 =100- jest tak duża, że dla całego badanego zakresu N (do 1 miliona rekordów) liczba serii początkowych nigdy nie przekroczyła 1000. W efekcie, algorytm zawsze mieścił się w jednej fazie scalania.

Potwierdzają to dane o operacjach dyskowych. Wykres I/O dla n=10 pokazuje że liczba operacji jest ściśle związana z liczbą faz, przy wzroście ilości faz liczba operacji dyskowych gwałtownie rośnie . Każdy skok w liczbie faz (np. z 1 na 2) wymusza dodatkowy, pełny cykl odczytu i zapisu wszystkich danych, co powoduje gwałtowny, skokowy wzrost całkowitej liczby operacji I/O. Natomiast dla n=1001 wykres I/O rośnie niemal idealnie liniowo, ponieważ liczba faz pozostaje stała w całym badanym zakresie.

Różnice pomiędzy zaobserwowaną ilością operacji a ilością teoretyczną przybliżoną wynika właśnie z tego przybliżenia

Porównując teoretyczną liczbę z zaobserwowaną w programie, można stwierdzić pełną zgodność. Zarówno liczba faz, jak i całkowita liczba operacji dyskowych uzyskana przez program, była identyczna z wartościami obliczonymi. Potwierdza to poprawność implementacji algorytmu.

3. Podsumowanie

Sortowanie przez scalanie z użyciem wielkich buforów jest wysoce efektywną metodą sortowania zewnętrznego, przeznaczoną do obsługi dużych zbiorów danych przy dostępie blokowym.

Kluczową zaletą algorytmu jest jego skalowalność: jak wykazano w przeprowadzonych eksperymentach, zwiększenie liczby dostępnych buforów (n) pozwala na tworzenie znacznie dłuższych serii początkowych w Etapie 1. To bezpośrednio przekłada się na drastyczne zmniejszenie liczby faz scalania w Etapie 2, które są najbardziej kosztowną operacją całego procesu.

Potwierdza to wysoką przydatność tej metody we współczesnych systemach komputerowych. Dzisiejszy sprzęt dysponuje dużą ilością pamięci operacyjnej, co algorytm ten potrafi efektywnie wykorzystać do zminimalizowania "wąskiego gardła" wydajności, jakim jest dostęp do pamięci masowej (dysku)