Patryk Miszke 193249

Implementacja scalania z użyciem metody wielkich buforów

1.Wstęp teoretyczny

Metoda scalania z wielkimi buforami sprawdza się świetnie, gdy dane wejściowe są na tyle duże, że nie mieszczą się w pamięci operacyjnej. Wykorzystuje ona dysk do sortowania i dzieli się na dwa główne etapy:

- **Tworzenie serii (runs)**: Dane są odczytywane porcjami (blokami), sortowane w pamięci, a następnie zapisywane na dysk jako posortowane serie.

- **Scalanie serii (merge)**: Posortowane serie są wczytywane, scalane i zapisywane z powrotem na dysk, aż pozostanie jedna finalna posortowana seria.

Bufory w pamięci są wykorzystywane do przechowywania porcji danych odczytywanych z pliku oraz tymczasowego przechowywania danych podczas scalania.

Taśma logiczna jest odpowiednikiem posortowanej serii w etapie tworzenia i scalania. W przypadku tej metody będzie ich $\frac{N}{n \cdot b}$, gdzie N - łączna liczba rekordów, n - liczba buforów, b - ilość rekordów w pojedynczej stronie/bloku.

2.Specyfika pliku testowego

Plik testowy to binarny plik zawierający dane w postaci ciągu liczb zmiennoprzecinkowych (*double*). Każda para liczb reprezentuje jedną współrzędną w rekordzie. Blok posiada wiele rekordów, każdy z nich to 3 współrzędne w układzie kartezjańskim (x1,y1,x2,y2,x3,y3), a zatem 6 liczb zmiennoprzecinkowych. W pliku wyjściowym rekordy po sortowaniu są wypisane względem wielkości pola trójkąta jaki tworzą (rosnąco). Plik jest zapisywany i odczytywany w formacie binarnym, co zapewnia szybki dostęp i minimalizuje rozmiar pliku.

3.Prezentacja wyników

N, b, n - zostały wyjaśnione w pkt. 1

Disk_oper – łączna liczba operacji zapisu i odczytu

Sorting_ph – liczba faz sortowania, zatem tworzenie run oraz scalanie ich

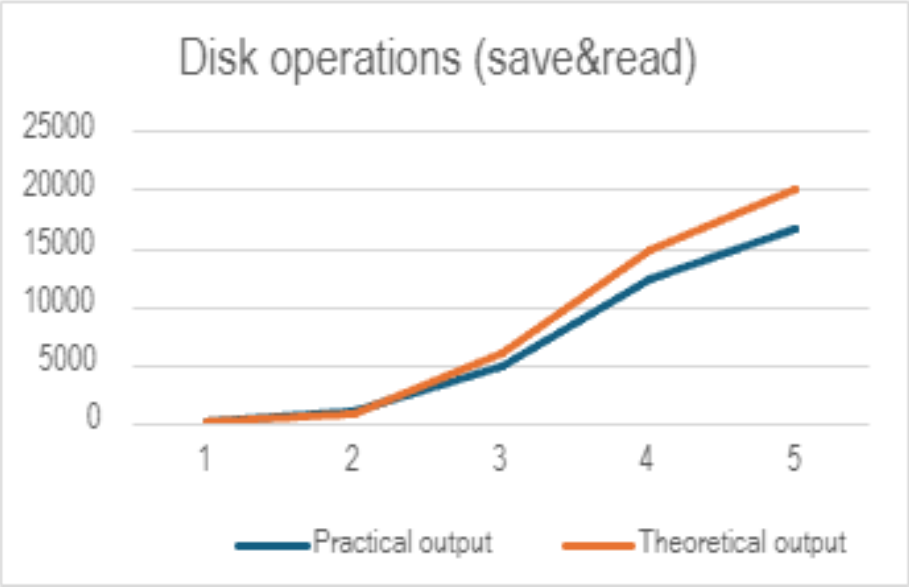
Runes – liczba utworzonych run w procesie ich tworzenia

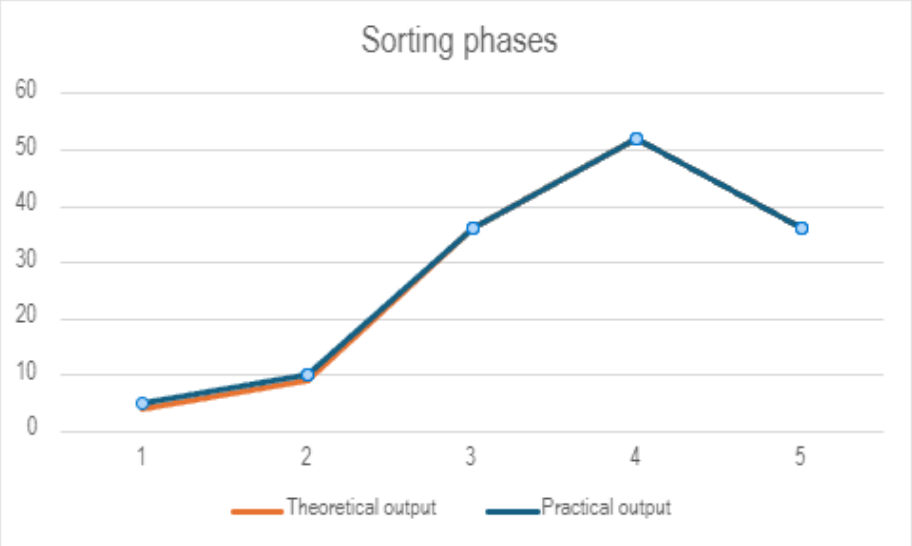
Dane wynikowe praktyczne:

N	b	n	Disk_oper	Sorting_ph	Runes
100	3	15	174	5	3
1000	5	25	1019	10	8
10000	10	30	4811	36	34
50000	20	50	12500	52	50
100000	30	100	16731	36	34

Dane wynikowe teoretyczne:

N	b	n	Disk_oper	Sorting_ph	Runes
100	3	15	134	4	3
1000	5	25	800	9	8
10000	10	30	6000	36	34
50000	20	50	15000	52	50
100000	30	100	20001	36	34





4.Wnioski eksperymentu

- **Liczby faz**:
  Przy mniejszej liczbie rekordów: Widać więcej faz w wynikach praktycznych, co może sugerować nadmiarowe scalanie. Może to wynikać z nieoptymalnego zarządzania buforami lub dodatkowych operacji, np. scalania zbyt małych serii.

  Przy większej liczbie rekordów: Liczba faz sortowania (zarówno praktyczna, jak i teoretyczna) jest **niższa w proporcji do liczby rekordów**, ponieważ większe serie mogą być tworzone i scalane dzięki lepszym proporcjom między buforami (n) i rozmiarem bloków (b).

- **Operacje dyskowe**:
  W obu tabelach liczba operacji dyskowych rośnie niemal liniowo z liczbą rekordów (N).

  Dla większych wartości N różnica między wynikami teoretycznymi a praktycznymi jest większa. Możliwe powody:
  - **Błędy zaokrągleń**: Teoretyczne operacje mogłyby zakładać idealne podziały na bloki.
  -**Zachowanie implementacyjne**: Praktyczne zapisy mogły obsługiwać kilka bloków jednocześnie, co skutkowało mniejszą liczbą operacji niż obliczona teoretycznie.

- **Możliwe poprawki**:
  Optymalizacja zarządzania buforami:
  -Sprawdzenie, czy buforów nie można lepiej wykorzystać, by uniknąć scalania zbyt małych serii.
  -Lepsze skalowanie algorytmu dla małych danych.

  Analiza, czy algorytm poprawnie rozpoznaje sytuacje, gdy kilka bloków może zostać zapisanych jako jedna operacja (zamiast odnotowywać każdą z nich osobno).