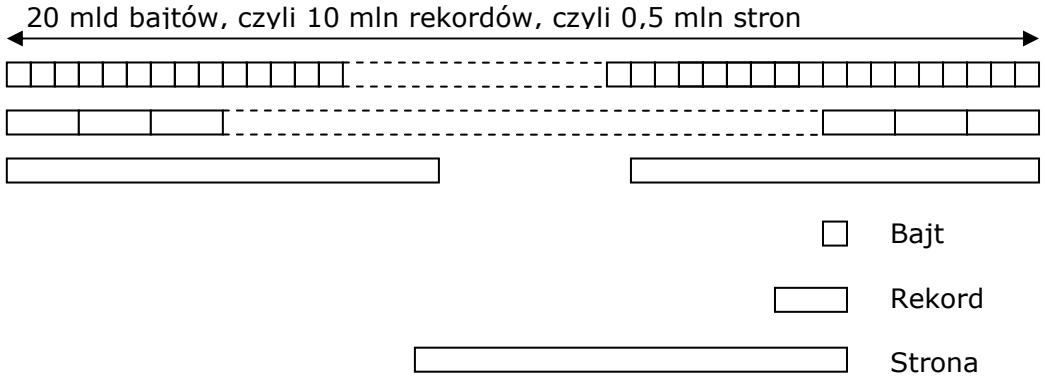
Czas sortowania EXTERNAL SORT dla PUW by wojmos ;-)

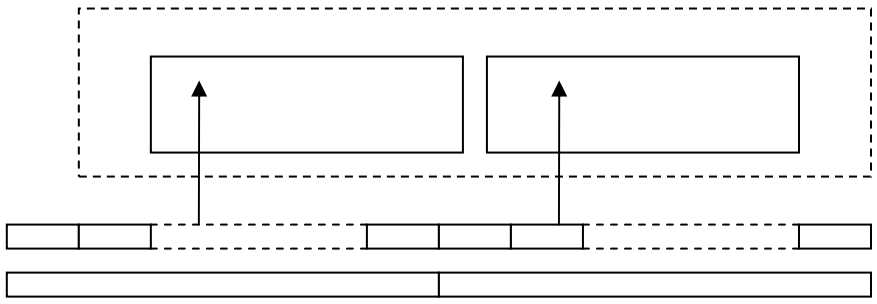
Mamy do posortowania nieuporządkowany plik z 10 mln rekordów, każdy o długości 200 bajtów. Cały plik zawiera więc 20 mld bajtów. Strona zawiera 20 rekordów.



Nasz bufor potrafi wczytać dane o wielkości 1 strony (jest to tak zwany bufor jednostronicowy). Załóżmy, że nasz automat sortujący ma 2 bufory wejściowe (czyli potrafi zapamiętać dane z 2 stron) i jeden wyjściowy. Te pierwsze używane są do pobierania danych z pliku źródłowego, zaś drugi – do zapisywani wyniku.

Algorytm działa następująco:

W przebiegu pierwszym wczytujemy kolejne pary stron (STRONA1-STRONA2, STRONA3-STRONA4 itp.). Załóżmy, że jesteśmy w pierwszym kroku, czyli:



Do bufora wejściowego 1 trafiła pierwsza strona pliku (czyli rekordy 1-20), a do bufora wejściowego 2 trafiła druga strona pliku (czyli rekordy 21-40). Bufor ma niewielką wielkość, więc jego zawartość można łatwo posortować w RAM (np. metodą QuickSort). Otrzymujemy 2 bufory wejściowe, w których są posortowane rekordy.
Teraz trzeba dokonać kolejnego sortowania – czyli z dwóch posortowanych „kawałków" (bufor1 i bufor2) ułożyć jeden kawałek posortowany (Merge-Sort). Nasz automat sortujący ma wszystkie dane w pamięci, więc po prostu będzie porównywał kolejne elementy z bufora1 i bufora2, mniejszy umieszczając w buforze wyjściowym.

Operacja ta zostanie powtórzona dla wszystkich par stron (par takich jest oczywiście 250000). W jej wyniku otrzymamy plik wyjściowy, który składa się z posortowanych „ciągów" danych o długości 2 stron. Ciąg taki skrypt nazywa sekwensem. W tej chwili sekwensów jest 250000, a każdy liczy sobie 2 strony (czyli 40 rekordów).

Zastanówmy się, ile odczytów stron potrzeba do takiego przebiegu? Oczywiście 500000 – bo każda strona musi być „przeczytana".

Drugi krok algorytmu będzie podobny. Algorytm będzie teraz porównywał sekwensy. Innymi słowy: pierwszy rekord sekwensu1 z pierwszym rekordem sekwensu 2.

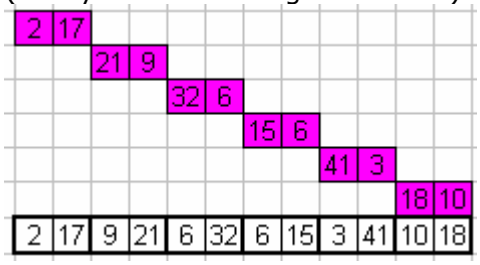
Oczywiście bufor może jednorazowo przechować jedynie 1 stronę (bo ma taką wielkość), ale to nie szkodzi, ponieważ w odpowiedniej chwili po prostu wczyta następną.

Posłużmy się na chwilę prostszym przykładem, aby zobrazować to działanie:

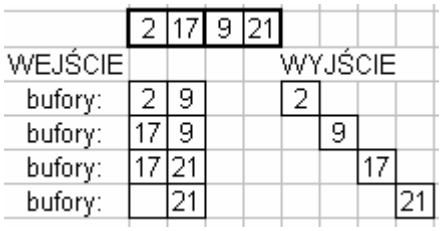
Załóżmy że nasz plik wygląda tak:

2	17	21	9	32	6	15	6	41	3	18	10

Dla uproszczenia przyjmijmy, że każda kratka to jeden rekord, a przy okazji, że nasza strona ma pojemność dokładnie jednego rekordu.
Tak wygląda przebieg pierwszego sortowania (zaznaczyłem wartości wczytywane jednocześnie do buforów wejściowych) i wygenerowane sekwensy (każdy z nich ma długość 2 stron):



Dalszy schemat postępowania pokazuje jedynie pierwsze scalanie drugiego przebiegu (czyli sortowanie sekwensu 2-17 i 2-21):



Jak widać, w pierwszej operacji, do bufora wczytaliśmy pierwsze strony obu sekwensów (czyli 2 i 9). Z porównania wynika, że 2 jest mniejsze (więc zostaje umieszczone w buforze wyjściowym). Algorytm potrzebuje teraz zastąpić mniejszą wartość (czyli 2) wartością kolejnej strony tego samego sekwensu. W naszym przypadku strona ta zawiera wartość 17. Ponieważ 9<17, więc tym razem do bufora wyjściowego wylatuje 9, a automat doczytuje 21 (kolejna strona z tego sekwensu, do którego należała „wyrzucona na wyjście" liczba).

Ja widać, po kolejnym przejściu algorytmu, nasz plik będzie się składał z następujących sekwensów:

2	9	17	21	6	6	15	32	3	10	18	41

Ile było odczytów stron? Oczywiście tyle, ile jest ich w pliku (każda musi być przeczytana i umieszczona na wyjściu). Łatwo zauważyć, że liczna ta jest stała w każdym przebiegu, bo co prawda spada dwukrotnie liczba sekwensów, ale za to są on dwukrotnie dłuższe.

Nawiasem mówiąc, jeśli nie ma sekwensu „do sparowania", to po prostu zostają „puste miejsca". Kolejne kroki naszego przykładu to:

2	6	6	9	15	17	21	32	3	10	18	41

2	3	6	6	9	10	15	17	18	21	32	41

Po tej krótkiej dygresji wracam do „dużego" przykładu:

Powstają trzy pytania:

1. Ile operacji wczytania strony ma miejsce podczas każdego przebiegu?
2. Ile operacji zapisania strony ma miejsce podczas każdego przebiegu?
3. Ile przebiegów zostanie przeprowadzonych?

Odpowiedź na pierwsze pytanie jest łatwa – N, czyli tyle, ile stron liczy dany plik (co jest logiczne, bo przecież wszystkie muszą być odczytane). Innymi słowy – w naszym przykładzie będzie to ZAWSZE 500000. Ile jest zapisów? Oczywiście tyle samo.
WNIOSEK: w takim przypadku liczba operacji = 2N (gdzie N – liczba stron w pliku)

Odpowiedź na pytanie trzecie jest trudniejsza. Za „zerowym" przejściem, długość sekwensu wynosi 1 stronę. Za każdym razem długość sekwensu ulega podwojeniu. Innymi słowy, trzeba zbadać za którym podwojeniem, długość sekwensu będzie równa liczbie stron pliku (lub od niej większa). Innymi słowy, wzór na liczbę przebiegów wynosi:

$x = \lceil \log_2 N \rceil + 1$, gdzie N jest liczbą stron (nie: rekordów) w pliku.

Operator $\lceil a \rceil$ to nic innego jak tzw. „sufit" czyli najmniejsza liczba całkowita równa lub mniejsza od a (czyli zaokrąglenie w górę).
Warto też przypomnieć twierdzenie o zamianie podstawy logarytmu (na kalkulatorze mamy zwykle log, czyli $\log_{10}$.

$$\log_2 N = \frac{\log N}{\log 2} = \frac{\ln N}{\ln 2}$$

W naszym przypadku:

$$x = \lceil \log_2 N \rceil + 1 = x = \lceil \log_2 N \rceil + 1 \approx \lceil 18{,}4 \rceil + 1 = 19 + 1 = 20$$

Jaki jest wiec sumaryczny czas całej operacji?

T = ilość przebiegów * czas przebiegu
T = 20 * 2N* czas_operacji_IO

W naszym przypadku:

T = 20*1000000*0,025s = 138,8 GODZIN

To nadal dużo, trzeba więc zastosować jakiś mechanizm przyspieszania.

Najprościej zastosować większy bufor (zwany buforem wielostronicowym). Zauważmy, że bufor o dwukrotnie większej pojemności oznacza taką samą sytuację, jak dwukrotnie mniejsza liczba stron.
W związku z tym, przy k-razy większym buforze wejściowym zmniejszy się liczba przebiegów:

$$x = \left\lceil \log_2 \frac{N}{k} \right\rceil + 1$$

N – liczba stron
k – ilość stron, które mieszczą się jednocześnie w buforze

W naszym przypadku dwukrotnie większego bufora będzie liczba przebiegów wynosi 18. Przy 10 krotnie większym buforze – 17 itp.

Sortowanie wielobuforowe polega na powiększeniu liczby buforów wejściowych (w metodzie klasycznej mamy oczywiście tylko 2 wejściowe). Oczywiste jest, że jeśli mamy 3 bufory, to wówczas długość sekwensu rośnie w każdym przebiegu 3 razy (a nie 2), zaś liczba sekwensów spada 3 krotnie.

Innymi słowy, kiedy mamy m buforów, wzór na liczbę przejść algorytmu wygląda następująco:

$$x = \lceil \log_m N \rceil + 1$$

I ostatecznie, wzór na przypadek m buforów, z których każdy mieści k stron:

$$x = \left\lceil \log_m \frac{N}{k} \right\rceil + 1$$

N – liczba stron
k – ilość stron, które mieszczą się jednocześnie w buforze
m – liczba buforów

Na zakończenie dodam tylko, że w naszych rozważaniach nie uwzględniliśmy czasu sortowania rekordów na jednej stronie, ale ponieważ dzieje się to w pamięci RAM – to jest on zaniedbywanie mały.

Z punktu widzenia algorytmów, opisane sortowanie to MERGE-SORT (skrypt tłumaczy to dość niezręcznie jako „dostawianie", podczas gdy poprawna nazwa to sortowanie przez **scalanie**)