Plik indeksowo-sekwencyjny

Łukasz Kołakowski 198000

Celem projektu było zaimplementowanie jednej z 3 struktur trzymającej posortowane wartości, do wyboru były b-drzewa, b+-drzewa oraz plik indeskowo-sekwencyjny. Postanowiłem zaimplementować plik indeskowo-sekwencyjny, implementacją wzorując się na materiałach wykładowych.

Metoda indeksowania:

W mojej implementacji indeks wygląda następująco: [klucz, nr strony]. Jest to klasyczna implementacja oparta na indeksach rzadkich, która oznacza że klucz trzymany w indeksie nie wskazuje tylko na 1 element, a reprezentuje wszystkie wartości na stronie. Jeden indeks rzadki trzyma elementy większe od siebie, ale nie większe niż następny rzadki wskaźnik.

1 klucz rzadki <= potencjalne klucze < 2 klucz rzadki

Algorytm działania:

Gdy chcemy dodać klucz to na początku sprawdzamy, który indeks jest nam najbliższy w tablicy indeksów rzadkich. Z niej wyczytujemy na której stronie znajduje może znajdować się miejsce na nasz klucz. Po tym czytamy daną stronę z pamięci i szukamy tam miejsca.

Jeżeli strona nie jest do końca pełna to przesuwamy wszystkie elementy i zapisujemy zaktualizowaną stronę, dopóki strona nie jest pełna nie ma możliwości dodania do strony z przepełnieniami.

Jeżeli strona jest pełna to szukamy miejsca w stronie z przepełnieniami, dopisujemy nasz rekord na końcu pliku, ale przepisujemy odpowiednie wskaźniki jak w liście jednostronnej.

Jeżeli nasz klucz jest najmniejszy ze wszystkich obecnych to zamieniamy wartość najmniejszego indeksu rzadkiego, i albo wsuwamy naszą wartość na początek strony (jeżeli strona nie jest pełna), albo wypychamy najmłodszy rekord ze strony do pliku *overflow*, a na początek strony wpisujemy nasz nowy najmniejszy.

Usuwanie rekordu polega na oznaczenia go specjalną wartością. Faktyczne usunięcie klucza nastąpi dopiero podczas reorganizacji. Uznajemy tak naprawdę że rekord cały czas istnieje, tylko nie mówimy tego użytkownikowi końcowemu.

Aktualizacja i szukanie rekordu działa podobnie do wstawiania, szukamy na początku na stornie z indeksami rzadkimi, po czym szukamy w pliku głównym a jeżeli nie istnieje to sprawdzamy plik *overflow*.

W pamięci operacyjnej buforuję jedną stronę z pliku głównego, oraz jedną stronę z pliku z przepełnieniami. Jeżeli indeks strony jaką chcę się przeczytać pokrywa się z tym, która jest zapisana w *cache'u,* nie dokonuję dodatkowego odczytu.

Cała tablica indeksów rzadkich jest trzymana w pamięci, ale przy reorganizacji zapisuję ją do pliku w partiach, i po reorganizacji czytam na nowo.

Nowe strony dodaje tylko podczas reorganizacji, jeżeli dostanę klucz o indeksie większym niż jaki jakikolwiek w pliku to trafia w odpowiednie miejsce do pliku *overflow.*

W trakcie reorganizacji, przechodzę po wszystkich kluczach zgodnie z ich kolejnością i jeżeli nie jest to klucz pusty lub nie jest on usunięty to przepisuję go do nowego pliku. Po przepisaniu wszystkich kluczy usuwam stary plik. Przy przepisywaniu kluczy pozostawiam wolne miejsce na przyszłe klucze na stronie (za to odpowiada współczynnik alfa).

Plik tekstowy:

Plik z danymi wygląda następująco: [klucz, x znaków ascii, wskaźnik na of]. Na każdą z tych wartości mogę przyjąć inną liczbę bajtów, ale przyjąłem że klucz ma 4 bajty, znaki ascii zajmują 10, a *wskaźnik na of* też jest 4 bajtowy.
Wskaźnik na of jeżeli ma wartość 0 to znaczy że nie posiada następnika, w innym przypadku wskazuje on na numer rekordu na który wskazuje. Jeżeli rekord jest usunięty po posiada on wartość „CCCCCCCCCC” (10 znaków „C”) dla łatwej widoczności usuniętego rekordu.

Rekord jest pusty jeżeli posiada klucz o indeksie 0 oraz wartość 10 znaków „C”), rekord jest usunięty jeżeli posiada nie zerowy klucz i też warość 10 znaków „C”.

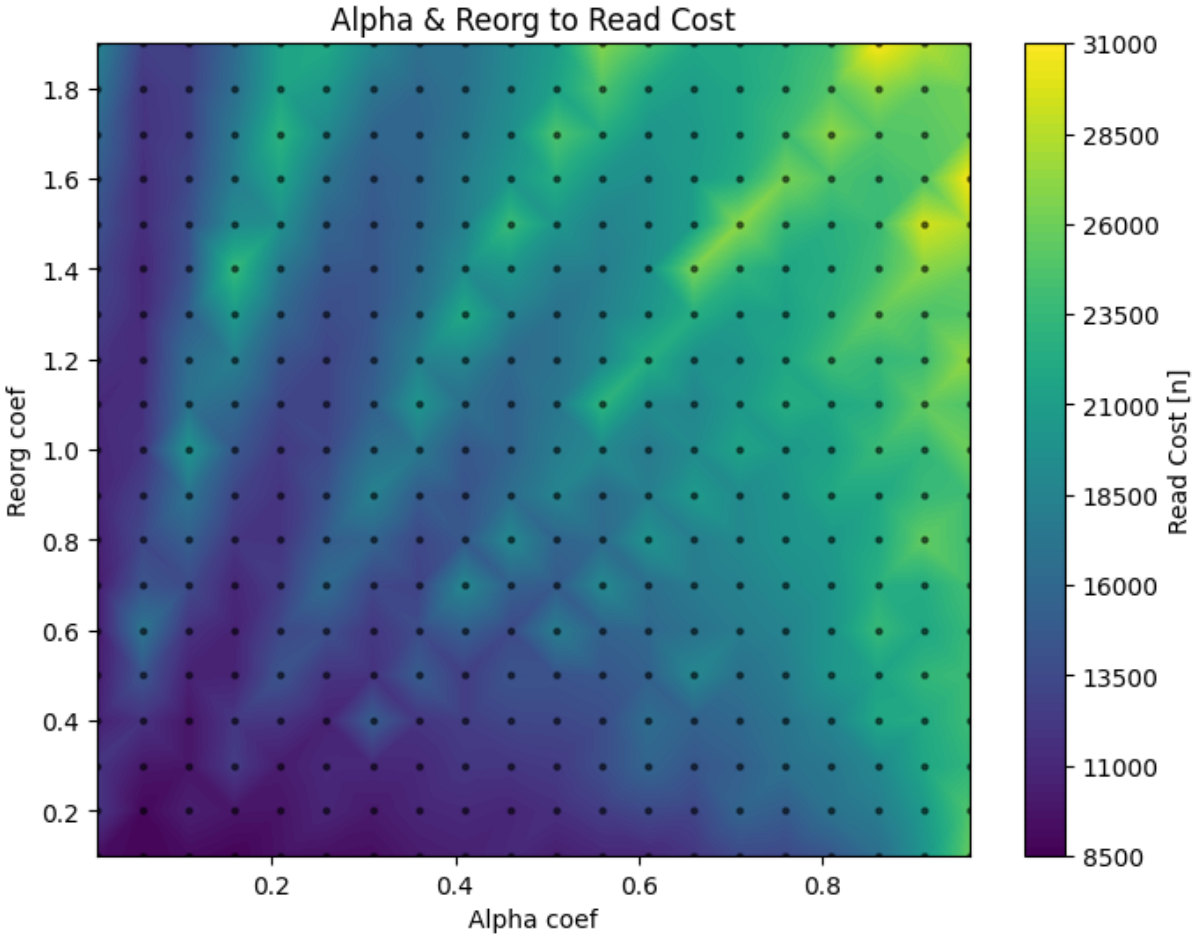
Z tego wynikają ograniczenia, użytkownik nie powinien być w stanie dodać klucza o wartości 0 jak i wskazanej frazy 10 znaków „C”.

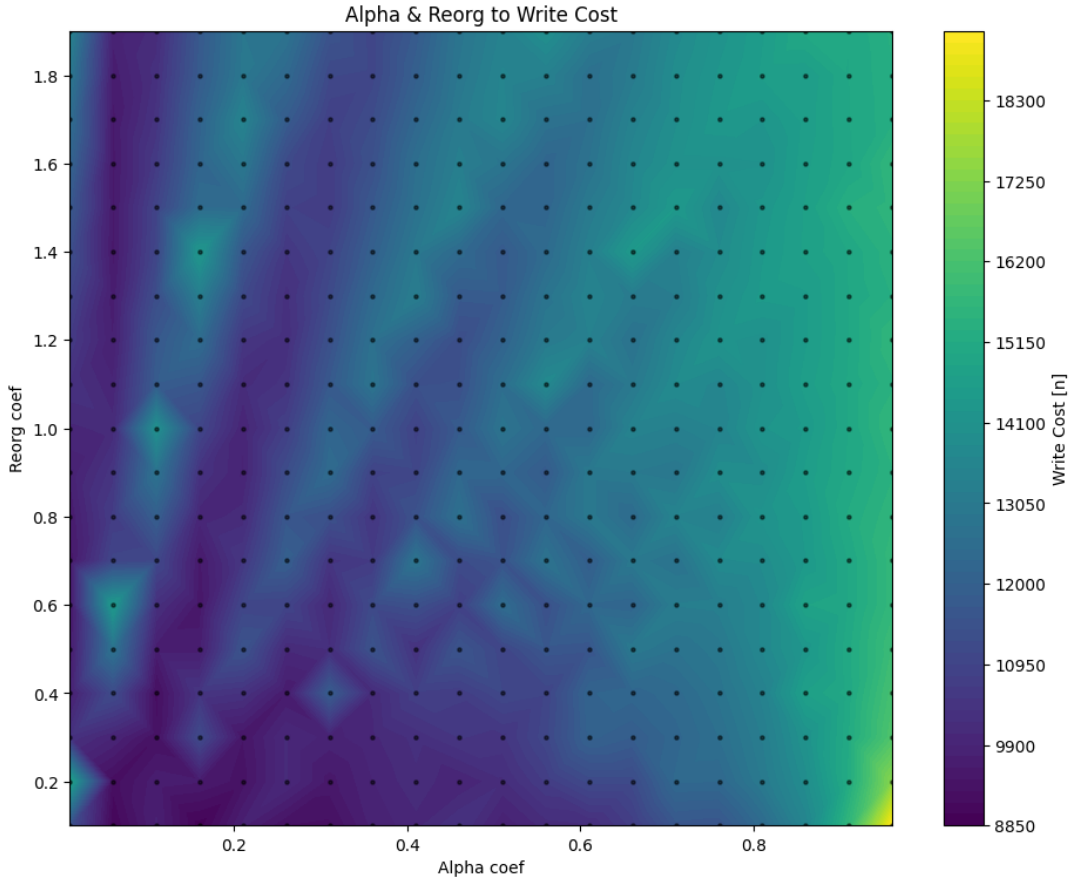
Przeprowadzony eksperyment

Postanowiłem sprawdzić zależność współczynnika *alfa* oraz współczynnika reorganizacji względem liczby operacji zapisu i odczytu. Rozbieżności i fluktuacje na wykresach mogą brać się z tego, że za każdym razem dodajemy inne rekordy losowe i może się zdarzyć, że losowo generowane rekordy zmuszą plik do reorganizacji.

Eksperyment 1:

Liczba rekordów	Współczynnik blokowania	Alfa	Współczynnik reorganizacji	Całkowity rozmiar rekordu
75000	30	5% - 100%	5% - 190%	18 bajtów


Alpha & Reorg to Read Cost
Reorg coef
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.2
0.4
0.6
0.8
Alpha coef
31000
28500
26000
23500
21000
18500
16000
13500
11000
8500
Read Cost [n]


Alpha & Reorg to Write Cost
Reorg coef
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.2
0.4
0.6
0.8
Alpha coef
18300
17250
16200
15150
14100
13050
12000
10950
9900
8850
Write Cost [n]

Objaśnienie eksperymentu 1:

Grafiki przedstawiają liczby odczytu i zapisu w zależności od współczynnika alfa i współczynnika reorganizacji. Kolorem zaznaczona jest liczba odczytów i zapisów, i jest ona mierzona w tych punkach zaznaczanych czarnym kolorem, wartości pomiędzy punktami są przybliżone liniowo.

Widać że duża wartość współczynnika alfa negatywnie wpływa na liczbę operacji, ponieważ kosz reorganizacji jest duży, bo zostawiamy strony z dużą liczbą wolnego miejsca co powoduje powstanie więcej stron niż jak mamy mały ten współczynnik.

Widać także linie zaznaczone zimniejszym kolorem co oznacza na pewne optimum między wartościami alfa i reorganizacji.

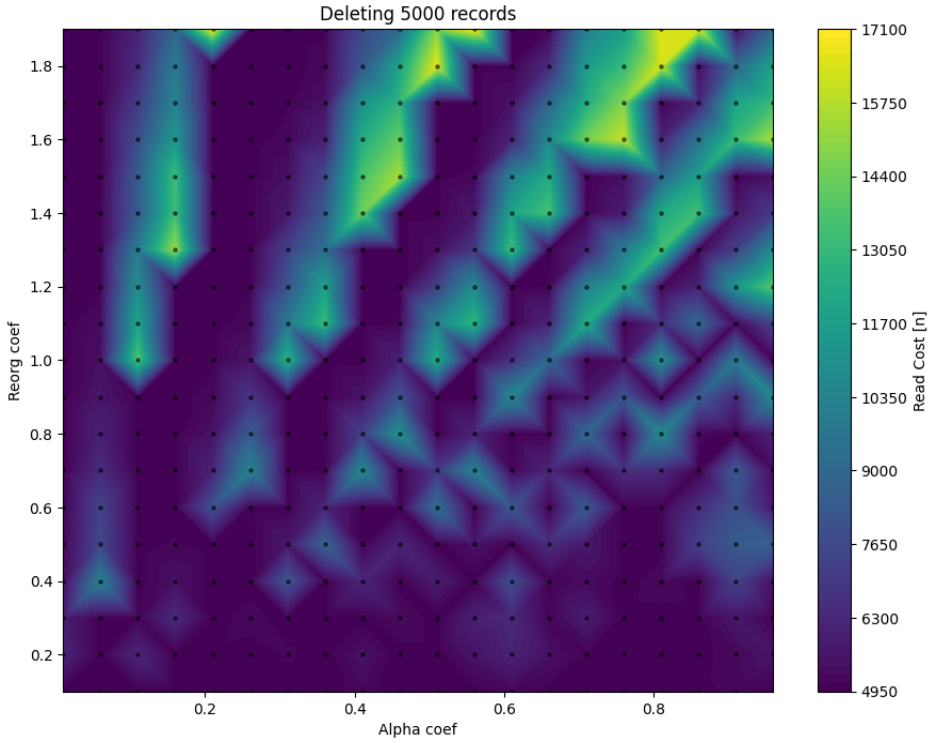
Średnia liczba operacji na dodanie klucza: 2.56 odczytów i 1.77 zapisów

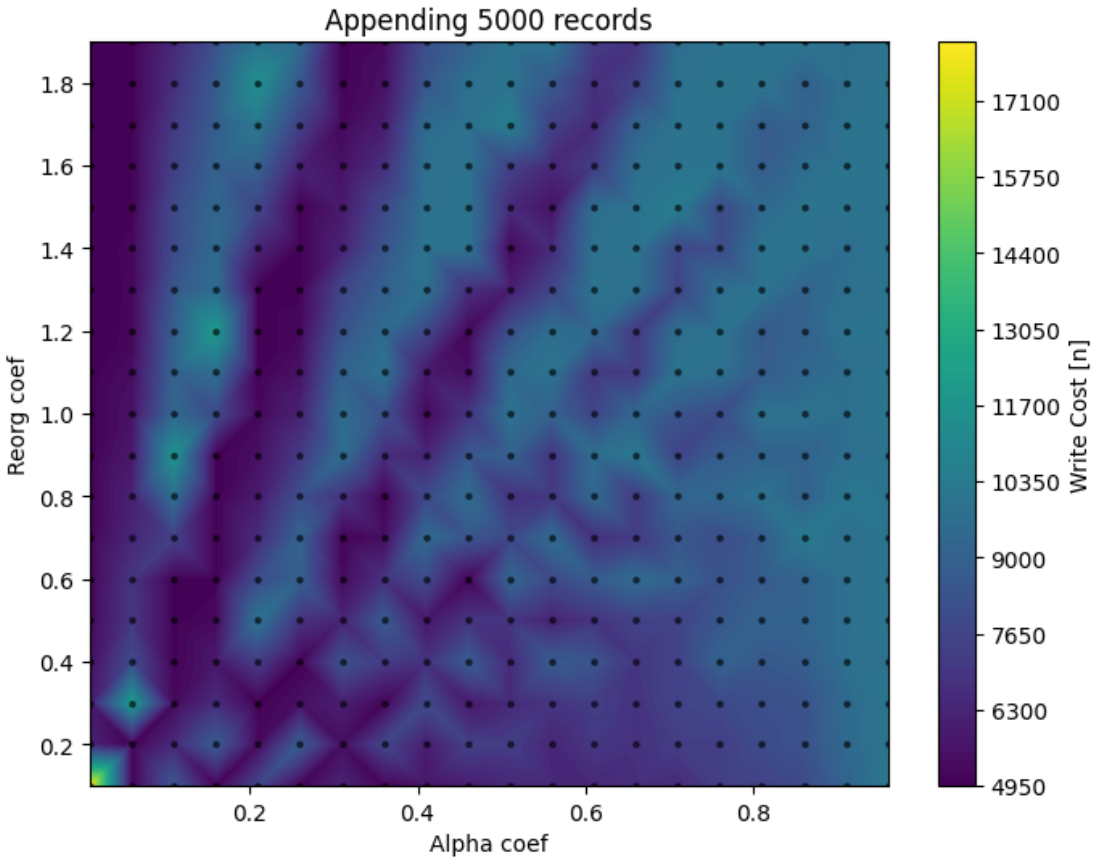
Wyższe wartości średnie odczytu i zapisu wynikają, że do liczby operacji doliczana jest reorganizacja pliku która na pewno nastąpiła.

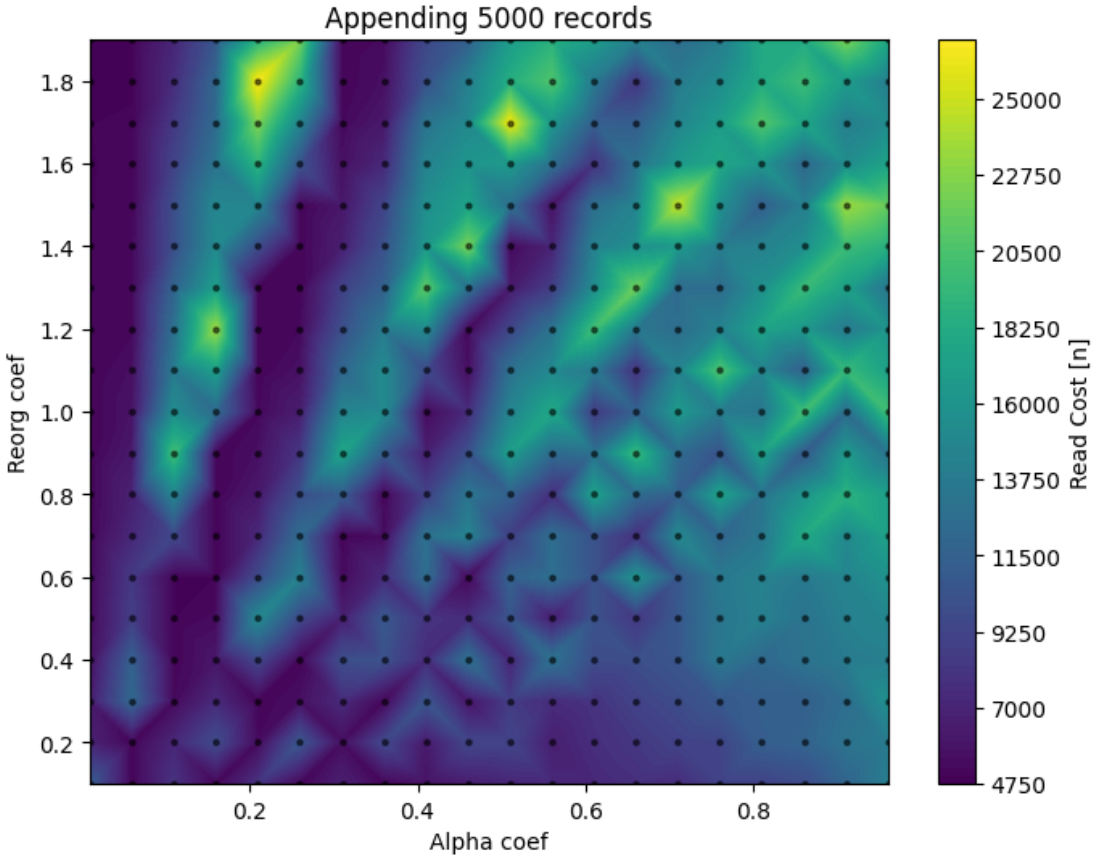
Eksperyment 2:

Początkowa liczba rekordów	Liczba dodawanych rekordów	Liczba usuwanych rekordów	Współczynnik blokowania	Alfa	Współczynnik reorganizacji	Całkowity rozmiar rekordu
7500	5000	5000	30	5% - 100%	5% - 190%	18 bajtów

W drugim eksperymencie sprawdziłem jak mają się operacje dodawania u usuwania do wyżej wskazanych danych.




Appending 5000 records
Reorg coef
Alpha coef
Write Cost [n]
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.2
0.4
0.6
0.8
17100
15750
14400
13050
11700
10350
9000
7650
6300
4950


Appending 5000 records
Reorg coef
Alpha coef
Read Cost [n]
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.2
0.4
0.6
0.8
25000
22750
20500
18250
16000
13750
11500
9250
7000
4750

Objaśnienie eksperymentu 2:

Prezentacja wyników jest analogiczna ja w eksperymencie pierwszym, nie załączyłem liczby zapisów do usuwania rekordu ponieważ, wiadomo że jest on stały równy 1 dla każdego usuniętego rekordu, wynika to z implementacji algorytmu.

Na wykresie przedstawiający liczbę zapisów ciemny kolor oznacza wartości mniejsze niż domyślne 5000 co wskazuje na działanie *cache'u* który umożliwia nam zaoszczędzenie operacji. Jasne pola pewnie wynikają z tego że w pliku nie została wykonana reorganizacja i są tam długie łańcuchy rekordów w *overflow*, przez które nasz algorytm musiał przejść.

Widać także, że dodawanie do istniejącego już pliku ma łagodniejsze zmiany związane z liczbą odczytu. Wynika to pewnie z tego że wystąpiła reorganizacja po której nie jest wymaganie szukanie wartości.

W moim algorytmie widać że w tych ciemnych polach liczba operacji dyskowych jest bliska 2 (jeden zapis i odczyt) na każdą operację jaką chcemy wykonać. Wynika to z tego że jeżeli posiadam wolne miejsce na stronie głównej, to tam dodam rekord, co powoduje że szukanie w czasie liniowym w sekcji *overflow* nie następuje.

Średnie liczby operacji dodawanie kolejnych rekordów:
odczytów na dodanie: **1.5**
zapisów na dodanie: **1.05**

Średnie liczby operacji na usuwanie rekordów:
odczytów na usunięcie: **1.4**
zapisów na usunięcie: **1.0**

Wnioski

Możliwe usprawnienia:

Jeżeli klucz jest większy niż jakikolwiek z aktualnych w pliku, to stworzyć nową stronę i zaktualizować tablicę indeksów. Podniosło by to trochę liczbę rekordów po których nastąpiła by reorganizacja pliku.

Optymalne wartości:

Z moich eksperymentów wynika, że najbardziej optymalne wartości alfy są w przedziałach od 10% do 40% wielkości strony. Reorganizacja strony też powinna następować w zakresie od 10% do 30% wielkości pliku *overflow* względem pliku głównego.

Możliwe jest też dobranie większych wartości współczynników alfa i reorganizacji, ale wymaga zależy to od większe liczby zmiennych, a ich wartości i tak są gorsze od tych wskazanych wyżej. Jedynym sensownym kandydatem wydaje się być niska alfa na poziomie 5% i reorganizacja w dużym zakresie od 10% do 200%.