

MapReduce w Apache Spark II

9 stycznia 2019

Opis pliku z zadaniami

Wszystkie zadania na zajęciach będą przekazywane w postaci plików `.pdf`, sformatowanych podobnie do tego dokumentu. Zadania będą różnego rodzaju. Za każdym razem będą one odpowiednio oznaczone:

- Zadania do wykonania na zajęciach oznaczone są symbolem \triangle – nie są one punktowane, ale należy je wykonać w czasie zajęć.
- Punktowane zadania do wykonania na zajęciach oznaczone są symbolem \diamond – należy je wykonać na zajęciach i zaprezentować prowadzącemu.
- Zadania do wykonania w domu oznaczone są symbolem \star – są one punktowane, należy je dostarczyć w sposób podany przez prowadzącego i w wyznaczonym terminie (zwykle przed kolejnymi zajęciami).

1 Agregacja danych

5p.◇

Treść

Dla losowych danych zawierających indeks grupy oraz wartość ciągłą pochodzącą z rozkładu normalnego oblicz statystyki: liczbę przykładów, średnią oraz wariancję. Statystyki należy policzyć dla wszystkich danych oraz dla każdej grupy z osobna. Do obliczeń należy wykorzystać operacje `map`, `reduce`, oraz `reduceByKey`. Oblicz statystyki przechodząc jednokrotnie przez zbiór danych. Oblicz statystyki dla pełnego zbioru danych na dwa sposoby. W pierwszych policz je bezpośrednio na pierwotnych danych. W drugim wykorzystaj wyniki policzone dla grup. Sprawdź, czy takie podejście przyspiesza obliczania. Czy skorzystanie z metody `cache` wpływa na szybkość obliczeń? W celu monitorowania zadań można skorzystać z interfejsu webowego <http://localhost:4040/jobs/>.

W celu wygenerowania danych skorzystaj z poniższego kodu:

```
1 import scala.util.Random
2
3 val groups = Map(0 -> (0,1), 1-> (1,1), 2-> (2,2), 3->(3,3),
4               4->(4,3), 5->(5,2), 6->(6,1))
5
6
7 val n = 1000000
8
9 val random = Random
10
11 val data = sc.parallelize(for (i <- 1 to n) yield {val g =
12   random.nextInt(7); val (mu, sigma) = groups(g); (g, mu +
13   sigma*random.nextGaussian())})
14
15 //Check 10 records
16 data.take(10)
```

2 Mnożenie macierzy

5p.◇

Treść

Napisz program, który pozwoli na mnożenie dwóch macierzy przez siebie. Zakładamy, że żadna z macierzy nie mieści się w pamięci i obie należy wczytać jako dane wejściowe. Plik `result_mm.txt` zawiera oczekiwany wynik mnożenia macierzy z pliku `M.txt` przez macierz zawartą w pliku `N.txt`.

Format plików wejściowych: `wiersz kolumna wartość`

Podpowiedź:

- Zajrzyj do wykładu :)
- Wczytaj macierze podobnie jak na poprzednich zajęciach,
- Odpowiednio przemapuj wczytane dane (`map`),
- Do wykonania zadania wystarczy wykonać następujące polecenia: `join`, `map`, `reduceByKey`, `sortByKey`, `collect`, `foreach`, `println`.