Big Data
Lista zadań

Jacek Cichoń, WPPT PWr, 2018/19

1 Wstęp

Zadanie 1 — Zainstaluj język Scala na swoim komputerze i pobaw się w konsoli REPL podstawowymi klasami, i obiektami tego języka. Rozszyfruj i zapamiętaj skrót REPL.

Zadanie 2 — Zacznij czytać książkę M. Oderskiego pt. *Programming in Scala*.

Zadanie 3 — Oprogramuj w języku Scala następujące funkcje:

1. `gcd(x:Int,y:Int):Int` (największy wspólny dzielnik) oraz `lcm` (najmniejsza wspólna wielokrotność). Ustalmy, że $\gcd(0,0) = \mathrm{lcm}(0,0) = 0$ oraz $\gcd(a,b) = \gcd(|a|,|b|)$ i $\mathrm{lcm}(a,b) = \mathrm{lcm}(|a|,|b|)$;
2. funkcję $\tau(n) = |\{k : 1 \leq k \leq n \wedge k|n\}|$;
3. funkcję $\sigma(n) = \sum\{k : 1 \leq k \leq n \wedge k|n\}$;
4. funkcję $\sigma(n,a) = \sum\{k^a : 1 \leq k \leq n \wedge k|n\}$;
5. funkcję Eulera phi zdefiniowaną wzorem

$$\phi(n) = |\{k \in \{1,\ldots,n\} : \gcd(k,n) = 1\}|$$

   Spróbuj w tym celu zastosować (mocno nieefektywną w tym przypadku) metodę `count` do zakresu Range(1,n+1). Sprawdź, czy na pewno otrzymasz $\phi(1) = 1$;
6. Sprawdź poprawność napisanych funkcji obliczając
   `Range(1,101).filter(x => 100 % x==0).map(x=>Euler(x)).sum`
   w konsoli REPL. Powinno wyjść 100.
7. Poznaj prosty dowód tego, że $\sum_{k|n} \phi(k) = n$ dla każdej liczby naturalnej $n \geq 1$.
8. Funkcję `isPrime(n:Int):Boolean` sprawdzającą, czy podana liczba jest pierwsza.
9. Sprawdź, czy Range(1,1000).filter(isPrime) = 168

Zadanie 4 — Zaimplementuj klasę Complex liczb zespolonych. Klasa ta ma posiadać metodę toString (deklaracja jej powinna następująca: `override def toString() : String`. Oprogramuj metodę dodawania liczb zepolonych (`def + (that:Complex) : Complex` = { ... }), mnożenia oraz wyznaczania modułu.

Zadanie 5 — Znajdź źródła swojej ulubionej książki. Zapisz je w formacie utf-8.

1. Zaimportuj bibliotekę io.Source (`import scala.io.Source`)
2. Skorzystaj z polecenia `Source.fromFile(source, "UTF-8")` do wczytania książki, zamień plik na łańcuch (`mkString`) i następnie podziel na wyrazy (split("\\s+")). Możesz to zrobić jednym poleceniem.
3. Usuń z tej listy stop-words (możesz ja znaleźć na stronie
   `https://pl.wikipedia.org/wiki/Wikipedia:Stopwords`
   (coś w stylu `Book.filterNot(Stop.contains(_))`)
4. Przekształć zbudowaną listę słów w kolekcję par typu (String,Int) postaci (słowo,1)
5. Pogrupuj listę par (coś w stylu `Filtered.groupBy(x=>x._1)`)
6. Zredukuj słowa (coś w rodzaju `Groupped.mapValues(x=>x.length)`)
7. Posortuj według drugiego parametru (np. `reduced.toSeq.sortWith((x,y)=>x._2>y._2)`)
8. Zapisz wynik do pliku. Uwaga: możesz skorzystać z obiektu `PrintWriter` z bibliotek `java.io`

9. Wyświetl kilkadziesiąt pierwszych elementów. Usuń z niej kilkanaście początkowych elementów i zapisz listę do pliku tekstowego.
10. Zbuduj chmurę wyrazów (word cloud) z otrzymanej listy. Możesz skorzystać np. z serwisu `http://www.wordclouds.com/`

Celem tego zadania jest wygenerowanie mniej więcej takiego obrazka (dla książki "Pan Tadeusz"):

Zadanie 6 — To jest kontynuacja poprzedniego zadania.

1. Podziel swoją książkę na rozdziały.
2. Każdy rozdział potraktuj jako dokumenty.
3. Podziel dokumenty na słowa. Wyznacz indeksy TF.IDF wszystkich słów we wszystkich rozdziałach
4. Zbuduj chmury wyrazów dla wszystkich rozdziałów i jedną chmurę dla całego dokument.

Zadanie 7 — Załóżmy, że mamy dostęp do bazy zakupów klientów w sieci hurtowni środków chemicznych z poprzedniego roku. W ciągu roku 10^7 klientów odwiedza ją 10 razy i za każdym razem kupuje średnio 10 różnego typu produktów z puli 200 dostępnych typów produktów. Załóżmy że znaleźliśmy w tej bazie danych dwóch klientów którzy zakupili choć raz ten sam koszyk produktów. Czy jest to czysty przypadek?

2 Funkcje haszujące

Zadanie 8 — Załóżmy, że $\mathcal{H} = \{h_1, \ldots, h_n\}$ jest rodziną k-niezależnych funkcji haszujących ze zbioru D w zbiór skończony R, czyli, że dla dowolnych $x_1, \ldots, x_k \in D$ oraz $y_1, \ldots, y_k \in R$ mamy

$$\Pr_{h \leftarrow \mathcal{H}}(h(x_1) = y_1 \wedge \ldots \wedge h(x_k) = y_k) = \frac{1}{|R|^k} .$$

Pokaż, że dla dowolnego $m \leq k$ jest ona również m-niezależna.

Zadanie 9 — Rozważmy funkcje haszującą zadaną wzorem $h(x) = x \mod 21$. Stosujemy ją do liczb podzielnych przez pewną stałą c. Dla jakich stałych c jest to odpowiednia funkcja haszująca, czyli dla jakich stałych c można się spodziewać, że rozkład załadowania kubełków $\{0, \ldots, 20\}$ będzie jednostajny?

Zadanie 10 — Znajdź wzór na rząd elementu $k \in \{0, \ldots, n-1\}$ w grupie $C_n = (\{0, \ldots, n-1\}, \oplus_n)$? Jaki jest związek tego zadnia z poprzednim zadaniem?

Zadanie 11 — Mamy n kubełków. Rzucamy do niech k kul.

1. Oszacuj k taki aby z dużym prawdopodobieństwem doszło do 3-kolizji, czyli aby a jakimś kubełku znalazły się 3 kulki.
2. Sprawdź eksperymentalnie otrzymany wynik
3. Uogólnij zadanie na a - kolizje
4. Wyznacz w podobny sposób wartość oczekiwaną liczby pustych urn. Kiedy (dla jakich k) ta liczba staje się mniejsza od 1.

Zadanie 12 — Dwóch studentów ma dzban wypełniony 8 litrami napoju. Mają do dyspozycji dzbanek o pojemności 5 litrów oraz drugi dzbanek o pojemności 3 litrów. Chcą podzielić się równo napojem. Jak mogą to zrobić? Zagadnie to można potraktować jako system przepisujący o stanie początkowym $(8, 0, 0)$. Możemy to zadanie próbować rozwiązać tak: losowo wybieramy dwie różne liczby i, j ze zbioru $\{1, 2, 3\}$ i próbujemy przelać napój z i-tego pojemnika do j-tego pojemnika; iterujemy to błądzenie losowe tak długo aż dojdziemy do stanu $(4, 4, 0)$. Jednak jest to kiepskie rozwiązanie - algorytm taki wpada bardzo często w pętle. Zastosuj technikę śledzenia przebiegu do unikania zapętleń. Oprogramuj to w języku Scala.

3 Model MapReduce

Zadanie 13 — Wymień jakie aspekty działania systemu MapReduce są poza zasięgiem programisty. Które elementy kontroluje programista?

Zadanie 14 — Wracamy do zadania „Word Count". Przepisz algorytmy które już napisałaś w języku Scala do programów działających w paradygmacie Map Reduce

Zadanie 15 — Zaprojektuj algorytm MapReduce który dostaje bardzo duży zbiór liczb całkowitych i produkuje na wyjściu jednocześnie:

1. Największą liczbę, najmniejszą liczbę.
2. Średnią wszystkich liczb.
3. Ten sam zbiór ale bez powtórzeń.
4. Liczbę różnych elementów bez powtórzeń.

Zadanie 16 — Niech $x \oplus y = x + y + 1$ oraz $x \otimes y = xy + x + y$ dla $x, y \in \mathbb{R}$. Pokaż, że są to działania łączne i przemienne na $\mathbb{R}$. Wskazówka: Spróbuj to zrobić z minimalną liczbą rachunków; rozważ funkcje $f(x) = x + 1$ oraz $g(x) = x - 1$; zauważ, że $g \circ f = Id$.

Zadanie 17 — Podaj kilka przykładów działań nieprzemiennych. Podaj kilka przykładów działań które nie są łączne.

Zadanie 18 — Pokaż, że operacje $\min(x, y)$ i $\max(x, y)$ są przemienne i łączne. Czy operacja $s(x, y) = \frac{x+y}{2}$ jest łączna?

Zadanie 19 — Wymień jakie aspekty działania systemu MapReduce są poza zasięgiem programisty. Które elementy kontroluje programista?

Zadanie 20 — Zaprojektuj algorytm MapReduce który dostaje bardzo duży zbiór liczb całkowitych i produkuje na wyjściu:

1. Największą liczbę.
2. Średnią wszystkich liczb.
3. Ten sam zbiór ale bez powtórzeń.
4. Liczbę różnych elementów bez powtórzeń.

Zadanie 21 — Zaprojektuj algorytm MapReduce, który wyznacza złączenie dwóch relacji o schemacie R(A,B,C) i S(X,Y,Z) według połączenia B=X oraz C=Y, czyli wyznacz tabelę

$$\{(A, Y) : (\exists B, C)(R(A, B, C) \wedge S(B, C, Y)\} .$$

Zadanie 22 — (Odwrócenie grafu) Dany jest graf w postaci listy sąsiadów: $[w, [w_{i,1}, w_{i,2}, \ldots, w_{i,n_i}]]$ zapisany w zbiorze tekstowym, np.

```
[
  [1, [3,4,5]],
  [2, [1,3]],
  [3, [4,5]],
  [4, [1,2]],
  [5, [4,5]]
]
```

Zastosuj technologię MapReduce do zbudowania grafu z odwróconymi linkami.

Zadanie 23 — Niech $F : ((\mathbb{N} \setminus \{0\}) \times \mathbb{R})^2 \to (\mathbb{N} \setminus \{0\}) \times \mathbb{R}$ będzie funkcją określoną wzorem

$$F([c_1, x_1], [c_2, x_2]) = [c_1 + c_2, \frac{c_1 x_1 + c_2 x_2}{c_1 + c_2}]$$

1. Pokaż, że F jest działaniem przemiennym i łącznym.
2. Oznaczmy przez $\oslash$ działanie $x \oslash y = F(x, y)$. Znajdź zwartą formułę dla

$$[c_1, x_1] \oslash [c_2, x_2] \oslash \ldots \oslash [c_n, x_n] .$$

3. Zastosuj tę własność funkcji do zastosowania combainera dla problemu wyznaczania średniej i wariancji.

Zadanie 24 — Zastosuj metodę map-reduce do wyznaczenia średniej geometrycznej i harmonicznej.

Zadanie 25 — Zastosuj metodę map-reduce do wyznaczenia wszystkich anagramów występujących w zbiorze tekstowym.

Zadanie 26 — Multizbiorem o skończonym nośniku Ω nazywamy funkcję $F : \Omega \to \mathbb{N}$. Dla $F, G : \Omega \to \mathbb{N}$ określamy $(F \cup G)(\omega) = \max\{F(\omega), G(\omega)\}$, $(F \cap G)(\omega) = \min\{F(\omega), G(\omega)\}$, $(F \setminus G)(\omega) = \max\{F(\omega) - G(\omega), 0\}$. Zaprojektuj map-reduce algorytm do wyznaczania tych trzech operacji. Algorytm na wejściu dostaje listę elementów zbioru

$$\{(1, \omega, F(\omega)) : \omega \in \Omega \wedge F(\omega) > 0\} \cup \{(2, \omega, G(\omega)) : \omega \in \Omega \wedge G(\omega) > 0\}$$

Zadanie 27 — (To nie jest zadanie na MapReduce) W pliku `TwoCollisions.csv`, do którego link znajduje się na stronie wykładu, w każdej linijce znajduje się `(NumerHotelu, NumerDnia,NumerOsoby)`. Znajdź takie osoby, które w dwóch różych dniach znajdowały się się w tym samym hotelu.

Zadanie 28 — Zastosuj dwukrotnie MapReduce do swojej książki którą używałeś do zadania z Word - Count zrób listę pięciu najczęściej powiązanych wyrazów z danym słowem. Przez powiązane słowa rozumiemy słowa występujące obok siebie (po usunięciu stop-words). Oczywiście oprogramować masz to zadanie w paradygmacie MapReduce.
Spróbuj zastosować otrzymaną listę do wygenerowania losowego paragrafu ze swojej książki.

Zadanie 29 — Niech (G, E) będzie grafem skierowanym. Dla $g \in G$ oznaczamy $inDeg(g) = |\{y \in G : (y, g) \in E\}$ oraz $outDeg(g) = |\{y \in G : (g, y) \in E\}$. Zaprojektuj w MapReduce procedury służące do wyznaczania listy $\{(g, inDeg(g), outDeg(g)) : g \in G\}$. Napisz również procedurę do wyznaczania średnich stopni $\frac{1}{n}\sum\{inDeg(g) : g \in G\}$ oraz $\frac{1}{n}\sum\{outDeg(g) : g \in G\}$.
Zastosuj opracowane algorytmy to wyznaczenia średnich stopni dla grafu „Stanford web graph", który możesz znaleźć na stronie http://snap.stanford.edu/data/web-Stanford.html

Zadanie 30 — Niech (G, E) będzie grafem. Współczynnikiem klasteryzacji węzła $g \in G$ nazywamy liczbę

$$c(g) = \frac{2}{|N_g|(|N_g| - 1)}|N_g \cap E| ,$$

gdzie $N_g = \{a : \{g, a\} \in E\}$. Zaprojektuj w MapReduce procedury służące do wyznaczania listy $\{(g, c(g), outDeg) : g \in G\}$. Napisz również procedurę do wyznaczania średniej wartości $\frac{1}{n}\sum\{c(g) : g \in G\}$
Zastosuj opracowane algorytmy to wyznaczenia średnich stopni dla grafu „Stanford web graph" (musisz przerobić ten graf na graf nieskierowany).

Zadanie 31 — Mamy n serwerów (wartości liczbowe sprawdzaj dla $n = 3000$). Prawdopodobieństwo zepsucia się jednego serwera podczas wykonania zadania map-reduce przez pojedynczy serwer w czasie T wynosi p (przyjmij dla obliczeń numerycznych, że $p = \frac{1}{3000}$). Przyjmij, że zdarzenia polegające na zepsuciu się serwera w kolejnych slotach czasowych o długości T są niezależne.

1. Oblicz prawdopodobieństwo poprawnego zakończenia całego zadania

2. Dzielimy teraz serwery na 3 grupy po 1000 serwerów. Każde zadania wykonujemy na trzech serwerach (po jednym z każdej grupy). Czas wykonania zadania wynosie teraz $3T$. Jakie jest teraz prawdopodobieństwo poprawnego zakończenia zadania?

Zadanie 32 — Załóżmy, że dwie macierze A, B rozmiaru $2n \times 2n$ zostały podzielone na bloki

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

rozmiaru $n \times n$

1. Pokaż, że
$$A \cdot B = \begin{bmatrix} A_{11} \cdot B_{11} + A_{12} \cdot B_{21} & A_{11} \cdot B_{12} + A_{12} \cdot B_{22} \\ A_{21} \cdot B_{11} + A_{22} \cdot B_{21} & A_{21} \cdot B_{12} + A_{22} \cdot B_{22} \end{bmatrix}$$
2. Spróbuj sformułować i udowodnić bardziej ogólny fakt.

Zadanie 33 — Nadal zajmujemy się wybraną przez Ciebie książką podzieloną na rozdziały. Teraz dla każdego słowa oraz każdego rozdziału wyznacz współczynniki TF.IDF.

1. Dla każdego rozdziału wyznacz 20 wyrazów o najwyższymwpółczynniki TF.IDF
2. Napisz funkcję realizującą następujące zadanie: wprowadzasz do niej słowo, a ona zwraca listę najbardziej pasujących rozdziałów (czyli posortuj rozdziały wg parametryu FT.IDF.

4 Podobieństwo tekstów

Zadanie 34 — Pokaż, że funkcja $d(A, B) = |A \triangle B|$ jest metrykę na przestrzeni niepustych skończonych podzbiorów ustalonego zbioru X.

Zadanie 35 — Niech $f : [0, \infty) \to [0, \infty)$ będzie funkcją rosnącą i wklęsłą.

1. Pokaż, że dla $a, b \geq 0$ mamy $f(a + b) \leq f(a) + f(b)$.
Wskazówka: Zauważ, że możemy założyć, że $a + b > 0$; następnie zauważ, że $a = (a + b)\frac{a}{a+b}$ oraz $b = (a + b)\frac{b}{a+b}$ i zastosuj nierówność Jensena dla funkcji wklęsłych.
2. Załóżmy dodatkowo, że $f(0) = 0$. Niech d będzie metryką na zbiorze X. Pokaż, że funkcja $\rho(x, y) = f(d(x, y))$ jest również metryką na zbiorze X.
3. Pokaż, że jeśli $\epsilon \in (0, 1)$ oraz d jest metryką na zbiorze X, to funkcja $\rho(x, y) = d(x, y)^\epsilon$ jest metryką na zbiorze X.
4. Pokaż, że jeśli d jest metryką na zbiorze X, to funkcja $\rho(x, y) = \frac{d(x,y)}{1+d(x,y)}$ jest metryką na zbiorze X.

Zadanie 36 — Wybierzmy dwa losowe m-elementowe podzbiory A, B n elementowego zbioru X. Jaka jest wartość oczekiwana podobieństwa Jacckarda $J(A, B)$?

Zadanie 37 — Korzystając z Twierdzenia o Gęstości Liczb Pierwszych (Prime Numbers Theorem) oszacuj liczbę liczb pierwszych z przedziału $[2^{64}, 2^{64} + 1000]$ i następnie wyznacz te liczby.

Zadanie 38 — (Twierdzenie Steinhausa) Niech d będzie metryką na zbiorze X. Ustalmy element $a \in X$ i zdefiniujmy funkcję

$$\rho(x, y) = \frac{2d(x, y)}{d(x, a) + d(y, a) + d(x, y)}$$

Celem tego zadania jest pokazanie, że ρ jest metryką na zbiorze X.

1. Pokaż najpierw, że jeśli $0 < p \leq q$ oraz $r \geq 0$ to $\frac{p}{q} \leq \frac{p+r}{q+r}$.
2. Wprowadź oznaczenia $p = d(x, y)$, $q = d(x, y) + d(x, a) + d(y, a)$ oraz $r = d(x, z) + d(y, z) - d(x, y)$ i zastosuj obserwację z poprzedniego punktu do pokazania nierówności trójkąta dla funkcji ρ.

Zadanie 39 — Zastosuj Twierdzenie Steinhausa do metryki $d(X, Y) = |X \triangle Y|$ na zbiorze skończonych podzbiorów zbioru Ω do pokazania, że funkcja $d(X, Y 0 = 1 - S(X, Y)$ (odległość Jaccarda) jest metryką.

Zadanie 40 — Załóżmy, że S jest takim podobieństwem obiektów przestrzeni Ω, że istnieje rodzina funkcji haszujących $\mathcal{H}$ oraz prawdopodobieństwo na rodzinie $\mathcal{H}$ takie, że dla dowolnych dwóch obiektów $A, B \in \Omega$ mamy

$$P_h[h(A) = h(B)] = S(A, B)$$

Pokaż, że wtedy funkcja $d(A, B) = 1 - S(A, B)$ jest metryką na zbiorze Ω.

Zadanie 41 — Uzupełnij szczegóły dowodu tego, że jeśli $\Omega = \{\omega_i : 1 \leq N\}$, π jest losową permutacją zbioru $\{1, \ldots, N\}$ (wybieraną zgodnie z rozkładem jednostajnym), oraz $h_\pi(X) = \min\{k : \omega_{\pi(k)} \in X\}$ dla $X \subseteq \Omega$ to

$$P_\pi[h_\pi(A) = h_\pi(B)] = S(A, B)\ .$$

Zadanie 42 — Napisz procedurę o specyfikacji `jaccard(f1:String,f2:String,k:Integer):Double`, która dla plików o nazwach `f1`, `f2` wyznacza ich `k`-gramy i następnie wylicza ich odległość Jaccarda. Przed wyznaczeniem k-gramów pliki powinne być oczyszczone (minimum to usunięcie znaków nowej linii, tabulatorów oraz podwójnych spacji)

1. Zastosuj tę procedurę do kilku wariantów swoich plików z algorytmami (zastosuj 4-gramy)
2. Zastosuj tę procedurę do porównania kolejnych rozdziałów analizowanej w Zadaniu 6 książki (zastosuj 7-gramy)

Zadanie 43 — Zastosuj metodę minhash do poprzedniego zadania. Twoja procedura powinna zależeć od paramertu H który określa liczbę funkcji haszujących stosowanych do budowania sygnatury tekstu.

1. Przetestuj tę procedurę na danych z poprzedniego zadania hla $H \in \{50, 100, 250\}$ - porównaj aproksymację odległości Jaccarda z jej dokładnymi wartościami.

Pamiętaj o wygenerowaniu wspólnej rodziny funkcji haszujących dla wszystkich analizowanych tekstów.

Zadanie 44 — Napisz procedurę służącą do wyznaczania sygnatur kosinusowych plików tekstowych korzystających z 1024 losowych wektorów z $\mathbb{R}^n$ (n tutaj oznacza moc wspólnego zbioru słów występujących w badanych dokumentach). Dokumenty reprezentowane mają być przez wektor częstotliwości słów. Zastosuj tę metodę do plików z Zadanie 6.

c.d.n.
Powodzenia,
Jacek Cichoń