Funkcje mieszające i filtry Blooma

12 maja 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, w wypadku nie wykonania zadania w czasie zajęć lub nieobecności, zadania staje się zadaniem do wykonania w domu ($\star$).
- 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).
- Zadania programistyczne można wykonywać w dowolnym języku programowania, używając jedynie biblioteki standardowej dostępnej dla tego języka.

1 Funkcje mieszające *10p.*◇

Treść

W ramach wykładu przedstawione zostały różne przykłady funkcji mieszających. Niech $h : \{1, \ldots, M\} \to \{0, \ldots, m-1\}$. Najprostsza metoda dzielenia (ang. *division method*) oblicza wartość funkcji mieszającej następująco:

$$h(x) = x \mod m$$

Łatwo zauważyć, że jest to metoda bardzo efektywna obliczeniowo, jednak posiada wiele wad. Między innymi dla danego m istnieje tylko jedna taka funkcja. Ciekawszą propozycją są funkcje mieszające zdefiniowane następująco. Niech p będzie liczbę pierwszą, taką że $p > M$. Wtedy, dla każdego $a \in \{1, \ldots, p-1\}$ oraz dla każdego $b \in \{0, \ldots, p-1\}$, możemy zdefiniować funkcję:

$$g_{a,b}(x) = ax + b \mod p\,.$$

Dla każdej takiej funkcji g definiujemy następnie funkcję mieszającą:

$$h_{a,b}(x) = g_{a,b}(x) \mod m\,.$$

Wartości a i b należy wylosować w sposób jednostajny z podanego przedziału. Powyższe funkcje stanowią rodzinę H funkcji mieszających spełniającą następującą własność:

$$\Pr_{h \in H}(h(x) = h(y)) \leq \frac{1}{m}\,,$$

dla każdej pary różnych $x, y \in \{1, \ldots, M\}$.

W ramach tego ćwiczenia należy zbadać empirycznie jakość obu funkcji mieszających (dodatkowo można wykorzystać inne funkcje mieszające). W tym celu należy posłużyć się danymi syntetycznymi oraz danymi MSDC. W pierwszym przypadku należy wykorzystać trzy zestawy danych:

- Kolejne liczby z zakresu od 0 do $10^8 - 1$,
- Liczby parzyste z zakresu od 0 do $2 \times 10^8 - 1$,
- Liczby nieparzyste z zakresu od 0 do $2 \times 10^8 - 1$,

Należy przeprowadzić eksperyment dla m równego odpowiednio 10, 1000, 100000. W celu sprawdzenia jakości funkcji mieszających należy:

- Wyświetlić liczbę zmapowanych wartości kluczy do wartości funkcji mieszającej z zakresu od 0 do 9.
- Czas potrzebny do obliczeń.

- Policzyć entropię otrzymanego rozkładu wartości, czyli:

$$E = -\sum_{i=0}^{m-1} p_i \log p_i \,,$$

gdzie p_i jest prawdopodobieństwem empirycznym uzyskania i jako wartości funkcji mieszającej. Łatwo zauważyć, że entropia optymalnej funkcji mieszającej wynosi:

$$E^* = -\log \frac{1}{m} \,.$$

Stąd ostateczny wynik powinien przedstawiać następujące wartości, E^*, E oraz $E^* - E$.

- Błąd średniokwadratowy:

$$MSE = \frac{1}{m} \sum_{i=0}^{m-1} \left(p_i - \frac{1}{m} \right)^2 \,,$$

gdzie p_i jest zdefiniowane jak powyżej.

Następnie podobną analizę należy wykonać mapując klucze utworów muzycznych z dostarczonej na stronie tabeli faktów problemu MSDC.

Punktacja:

- poprawna implementacja: 4p.
- weryfikacja empiryczna własności funkcji mieszających na danych syntetycznych: 3p.
- weryfikacja empiryczna własności funkcji mieszających na danych MSDC: 3p.

2 Filtr Blooma *10p.*◇

Treść

Zadanie polega na zaimplementowaniu i przebadaniu filtra Blooma, przy założeniu, że obiekty pochodzą z dziedziny liczb naturalnych w podanym zakresie.

Filtr Blooma jest probabilistyczną strukturą, która w sposób efektywny pamięciowo przechowuje (przybliżoną) informację o przynależności obiektów do zbioru S, będącego podzbiorem bardzo licznej dziedziny. Filtr Blooma jest parametryzowany poprzez liczbę funkcji mieszających k oraz rozmiar m, który musi być odpowiednio większy od zakładanej liczności n zbioru S.

Należy zaimplementować metodę dodawania elementu do zbioru oraz sprawdzania, czy dany element znajduje się w zbiorze. Porównaj filtr Blooma z filtrem referencyjnym, który wykorzystuje dokładną reprezentację zbioru (np. tabelę mieszającą). Policz dla danej instancji filtra Blooma liczbę obiektów prawdziwie pozytywnych (TP), prawdziwie negatywnych (TN), fałszywie pozytywnych (FP) oraz fałszywie negatywnych (FN). Sprawdź, czy zaimplementowany filtr Blooma spełnia gwarancje teoretyczne. W tym celu zweryfikuj Twoją wiedzę z treści prezentowanych na wykładzie. Ponadto zweryfikuj jego złożoność pamięciową i czas potrzebny do obliczeń. W tym celu wypisz zajmowaną pamięć dla obu filtrów oraz czas potrzebny do ich utworzenia oraz czas filtrowania danych z zakresu od 0 do $10^8 - 1$.

W celu przygotowania danych (struktury `set` reprezentującej zbiór wykorzystywany do filtrowania) można skorzystać z dowolnej reprezentacji zbioru i wylosować n wartości z podanego zakresu r, np.:

```
int n = 1_000_000;
int r = 100_000_000;

Random random = new Random();
HashSet<Integer> set = new HashSet<Integer>(n);
while(set.size() != n) {
  set.add(random.nextInt(r));
}
```

Przykładowy kod weryfikujący jakość filtra prezentuje się następująco (`filter` to filtr Blooma lub filtr referencyjny):

```
int TP = 0, FP = 0, TN = 0, FN = 0;
  for(int i = 0; i < r; i++) {
    if(filter.contains(i) && set.contains(i)) {
      TP++;
    } else if(!filter.contains(i) && !set.contains(i)) {
      TN++;
    } else if(!filter.contains(i) && set.contains(i)) {
      FN++;
```

```
9       } else if(filter.contains(i) && !set.contains(i)) {
10        FP++;
11      }
12    }
13    System.out.println("TP = " + String.format("%6d", TP) + "\tTPR
          = " + String.format("%1.4f", (double) TP/ (double) (TP+FN)
        ));
14    System.out.println("TN = " + String.format("%6d", TN) + "\tTNR
          = " + String.format("%1.4f", (double) TN/ (double) (TN+FP)
        ));
15    System.out.println("FN = " + String.format("%6d", FN) + "\tFNR
          = " + String.format("%1.4f", (double) FN/ (double) (TP+FN)
        ));
16    System.out.println("FP = " + String.format("%6d", FP) + "\tFPR
          = " + String.format("%1.4f", (double) FP/ (double) (TN+FP)
        ));
```

Punktacja:

- poprawna implementacja: 5p.
- efektywna implementacja: 3p.
- weryfikacja własności teoretycznych: 2 p.

3 Filtr Blooma dla danych MSDC *5p.*◇

Treść

Wykorzystaj filtr Blooma do przechowywania informacji na temat utworów odsłuchanych przez każdego użytkownika. Porównaj to podejście z filtrem referencyjnym, który wykorzystuje dokładną reprezentację zbioru (np. tabelę mieszającą). Policz dla filtra Blooma liczbę obiektów prawdziwie pozytywnych (TP), prawdziwie negatywnych (TN), fałszywie pozytywnych (FP) oraz fałszywie negatywnych (FN). W tym celu dla każdego użytkownika wylosuj liczbę 100 identyfikatorów utworów muzycznych i sprawdź odpowiedź filtru Blooma i filtru referencyjnego. Ponadto zweryfikuj jego złożoność pamięciową i czas potrzebny do obliczeń i porównaj z odpowiednimi wielkościami otrzymanymi dla filtra referencyjnego.

Zbiór danych zawierający tabelę faktów problemu MSDC jest dostępny na stronie. W celu realizacji tego zadania wykorzystaj kod z poprzednich zadań.

Punktacja:

- poprawna implementacja: 3p.
- weryfikacja jakości: 2p.