Materiały do wykładów z analizy algorytmów

Problem przybliżonego sumowania

Jakub Lemiesz

1 Problem przybliżonego sumowania

Analogicznie do problemu przybliżonego zliczania możemy rozpatrywać problem przybliżonego sumowania. Niech $\mathfrak{M} = (S, f)$ oznacza multizbiór, gdzie f jest funkcją $f : S \to \mathbb{N}_{\geq 1}$. Wartość funkcji $f(s)$ nazywamy licznością elementu $s \in S$. Przyjmijmy, że każe element $e \in S$ jest krotką $e = (i, \lambda_i)$, gdzie i jest unikalnym identyfikatorem elementu, a $\lambda_i \in \mathbb{R}_{>0}$ opisuje ustaloną cechę elementu i. Przyjmijmy również dla łatwości zapisu, że jeśli w zbiorze S jest n elementów, to każdy z nich ma unikalny identyfikator $i \in \{1, 2, \ldots, n\}$. W praktyce identyfikator może być dowolnym unikalnym ciągiem symboli. Wartości cech $\lambda_1, \lambda_2, \ldots, \lambda_n$ nie muszą być unikalne.

O multizbiorze $\mathfrak{M}$ możemy myśleć jak o strumieniu danych, w którym obserwowane kolejno elementy (i, λ_i) mogą się powtarzać. Wówczas problem przybliżonego sumowania unikalnych elementów można sformułować następująco. Jak z zadaną dokładnością oszacować wartość

$$\Lambda = \sum_{i=1}^{n} \lambda_i$$

w możliwie małej pamięci? W szczególności nie chcemy przechowywać w pamięci wszystkich dotychczas zaobserwowanych identyfikatorów.

2 Algorytmy przybliżonego sumowania

Idea rozważanego dalej algorytmu przybliżonego sumowania opiera się na własnościach rozkładu wykładniczego i pochodzi z pracy [3]. Oryginalne zastosowanie algorytmu było związane z efektywnym obliczaniem wartości wieloargumentowej funkcji w sieci rozproszonej. Algorytm ten umożliwia kompromis pomiędzy pamięcią a dokładnością obliczenia dla pewnej rodziny funkcji. W szczególności może być on wykorzystany do szacowania wartości sumy.

Algorytm szacowania wartości sumy został przedstawiony w poniżej zamieszczonym pseudokodzie. Algorytm otrzymuje na wejściu trzy argumenty: referencję do analizowanego multizbioru $\mathfrak{M}$, funkcję haszującą h i parametr m. Parametr m określa rozmiar tablicy przechowującej hasze wybranych elementów i kontroluje dokładność końcowego oszacowania.

Dla każdego elementu z $\mathfrak{M}$ algorytm oblicza m haszy. Wykorzystanie funkcji haszującej zapewnia substytut losowości umożliwiający przypisanie do danego elementu za każdym razem tych samych m pseudo-losowych wartości. Zapis $i^\frown k$ oznacza konkatenacje ciągów bitowych reprezentujących wartości i oraz k.

Na potrzeby analizy zakładamy, że funkcja haszująca $h : \{0,1\}^u \to 0.\{0,1\}^v$, $u, v \in \mathbb{N}_{>0}$ jest prawdziwie losowa (ang. truly random) i dla danego argumentu zwraca wartość z rozkładem jednostajnym na przedziale $[0, 1)$ niezależnie od wartości zwracanych dla innych argumentów.

Na podstawie wyliczonych haszy uaktualniana jest zawartość tablicy $\mathbf{M}$ i stosowany jest estymatora sumy postaci:

$$\bar{\Lambda} = \frac{m-1}{\sum_{k=1}^{m} \mathbf{M}[k]} \, .$$

W dalszej części uzasadniamy postać estymatora i analizujemy jego własności.

UNIQUE_SUM($\mathfrak{M}$, h, m)

Initialization:

1: set each of m positions of sketch $\mathbf{M}$ to ∞

Upon element $(i, \lambda_i) \in \mathfrak{M}$ **arrival:**

2: **for all** $k \in \{1, 2, \ldots, m\}$ **do**

3: $\quad u \leftarrow h\,(i^\frown k)$

4: $\quad \mathbf{M}[k] \leftarrow \min\left\{\mathbf{M}[k], -\frac{\ln u}{\lambda_i}\right\}$

5: **end for**

Upon request:

6: **Return:** $\bar{\Lambda} = \frac{m-1}{\sum_{k=1}^{m} \mathbf{M}[k]}$

3 Analiza algorytmu

3.1 Jedno doświadczenie losowe: $m = 1$

Załóżmy na początek, że $m = 1$, czyli że tablica $\mathbf{M}$ zawiera jedną wartość i dla każdego elementu generowany jest jeden hasz. Przyjmijmy, że $U_i \sim \mathcal{U}(0, 1)$ jest zmienną losową o rozkładzie jednostajnym na $[0, 1)$ odpowiadającą wartości hasza wygenerowanego dla elementu i w linii 3 algorytmu. Z założenia, że h jest prawdziwie losowa wynika, że $U_1, \ldots, U_n$ można traktować jak niezależne zmienne losowe. Stąd zmienne losowe $S_1, \ldots, S_n$, gdzie

$$S_i = \frac{-\ln U_i}{\lambda_i}$$

również są niezależne. Z twierdzenia o dystrybuancie odwrotnej (ang. the inverse transform sampling theorem, zobacz: [2], strona 28) wynika, że zmienna losowa S_i ma rozkład wykładniczy z parametrem λ_i, co oznaczamy w następujący sposób:

$$S_i \sim Exp(\lambda_i)\ .$$

Analizę bardzo krótkiego dowodu tego twierdzenia (jedna linijka) pozostawiamy jako ćwiczenie (zobacz: ćwiczenie 22).

Zauważmy, że wartość przechowywana w $\mathbf{M}$ po przejściu całego multizbioru odpowiada najmniejsze z wartości $S_1, \ldots, S_n$ (patrz: linia 4 algorytmu). Niech

$$M = \min\{S_1, \ldots, S_n\}$$

i zauważmy, że ponieważ $S_1, \ldots, S_n$ są niezależnymi zmiennymi losowymi, wykorzystując wzór na dystrybuantę zmiennej o rozkładzie wykładniczym mamy

$$\Pr(M \geq x) = \prod_{i=1}^{n} \Pr(S_i \geq x) = e^{-(\lambda_1 + \ldots + \lambda_n)x} = e^{-\Lambda x}\ .$$

Zatem M również ma rozkład wykładniczy (zobacz: ćwiczenie 23):

$$M \sim Exp(\Lambda).$$

Ponieważ M ma rozkład wykładniczy z parametrem Λ wiemy, że $\mathbb{E}(M) = 1/\Lambda$. Moglibyśmy zatem spróbować zdefiniować estymator wartości Λ jako $\bar{\Lambda} = 1/M$. Niestety, łatwo sprawdzić, że wartość oczekiwana tak zdefiniowanego estymatora $\bar{\Lambda}$ jest nieograniczona:

$$\mathbb{E}[1/M] = \int_0^\infty \frac{1}{x} \Lambda e^{-\Lambda x} dx\ .$$

3.2 Średnia z niezależnych doświadczeń losowych: $m \geq 3$

Przyjmijmy teraz, że $m \geq 3$, czyli że dla każdego elementu w linii 3 algorytmu generowane są $m \geq 3$ hasze. Każdy z tych haszy jest związany z innym doświadczeniem losowym, takim jak doświadczenie losowe opisane w sekcji 3.1. Wszystkie te doświadczenia losowe są niezależne (inna wartość k na wejściu do funkcji haszującej w linii 3 algorytmu). Zatem każda z wartości $\mathbf{M}_k$ przechowywanych w tablicy $\mathbf{M} = (\mathbf{M}_1, \ldots, \mathbf{M}_m)$ odpowiada zmiennej losowej

$$M_k \sim Exp(\Lambda) \ ,$$

a zmienne $M_1, \ldots, M_m$ są niezależne.

Można pokazać (zobacz: ćwiczenie 24 i 25), że suma m niezależnych zmiennych o rozkładzie wykładniczym z tym samym parametrem Λ ma rozkład gamma:

$$G_m := \sum_{i=1}^{m} M_i \sim \Gamma(m, \Lambda) \ ,$$

gdzie $m \in \{1, 2, 3, \ldots\}$ i $\Lambda \in (0, \infty)$ są parametrami rozkładu. Można pokazać, że dla $m \geq 2$ estymator

$$\bar{\Lambda}_m := \frac{m-1}{G_m}$$

jest nieobciążonym estymatorem Λ ćwiczenie 26):

$$\mathbb{E}[\bar{\Lambda}_m] = \Lambda$$

oraz, że dla $m \geq 3$ mamy ćwiczenie 27):

$$\mathbb{SE}\left[\bar{\Lambda}_m\right] = \sqrt{\mathbb{V}\mathrm{ar}\left(\bar{\Lambda}_m / \Lambda\right)} = \frac{1}{\sqrt{m-2}} \ .$$

Estymator $\bar{\Lambda}_m$ przypomina średnią harmoniczną estymacji z m niezależnych eksperymentów, takich jak opisane w sekcji 3.1 (w średniej harmonicznej mielibyśmy m zamiast $m-1$ w liczniku).

Estymator $\bar{\Lambda}_m$ można również wykorzystać do szacowania liczby unikalnych elementów n, a także średniej wartości $\frac{\Lambda}{n}$ czy wariancji (zobacz: ćwiczenie 28).

Dla przykładu, ustalając $\lambda_i = 1$ dla wszystkich $i \in \{1, \ldots, n\}$, można pokazać, że $\bar{\Lambda}_m$ jest estymatorem największej wiarygodności dla parametru $\Lambda = n$, gdzie n oznacza w tym przypadku liczbę różnych elementów w multizbiorze $\mathfrak{M}$ (zobacz [1]).

Literatura

[1] C. Baquero, P. S. Almeida, and R. Menezes. Fast estimation of aggregates in unstructured networks. In *Proceedings of the 2009 Fifth International Conference on Autonomic and Autonomous Systems*, pages 88–93, 2009.

[2] L. Devroye. *Non-Uniform Random Variate Generation*. Springer-Verlag, New York, NY, USA, 1986.

[3] D. Mosk-Aoyama and D. Shah. Computing separable functions via gossip. In *Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, PODC '06, pages 113–122, 2006.