WYDZIAŁ PODSTAWOWYCH PROBLEMÓW TECHNIKI
POLITECHNIKI WROCŁAWSKIEJ

PRAKTYCZNE ZASTOSOWANIA ALGORYTMÓW PRZYBLIŻONEGO ZLICZANIA

Anna Kawecka

Praca inżynierska napisana
pod kierunkiem
dr inż. Jakuba Lemiesza

WROCŁAW 2013

Spis treści

Bibliografia

1. P. Flajolet , É. Fusy , O. Gandouet, et al., Hyperloglog: The analysis of a near-optimal cardinality estimation algorithm. Aofa '07: Proceedings of the 2007 International Conference on Analysis of Algorithms, 127-146, 2007.

2. P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. Journal of Computer and System Sciences, 31 (2): 182 - 209, 1985.

3. F. Giroire, Order Statistics and Estimating Cardinalities of massive Data Sets. AofA '05: Proceedings of the 2005 International Conference on Analysis of Algorithms, 157-166, 2005.

4. F. Giroire, Directions to use probabilistic algorithms for cardinality for DNA analysis. Journées Ouvertes Biologie Informatique Mathématiques, 2006.

5. J. Lemiesz, Podstawowe algorytmy dla sieci Ad Hoc. Praca doktorska, Politechnika Wrocławska, 2012.

6. K.-Y. Whang, B. T. Vander-Zanden, H. M. Taylor, A Linear-Time Probabilistic Counting Algorithm for Database Applications. ACM Transactions on Database Systems, 15 (2): 208 - 229, 1990.

1 Wprowadzenie do tematyki algorytmów zliczania. Opis problemu.

Problemem, dla którego opracowane zostały algorytmy przybliżonego zliczania było oszacowanie liczby różnych elementów n dla bardzo dużego zbioru elementów. Nie przyjmuje się żadnych dodatkowych założeń dotyczących struktury powtórzeń elementów w zbiorze. Dokładne rozwiązania tego problemu - zapamiętywanie kolejnych nowych elementów oraz sprawdzanie dla każdego kolejnego elementu czy już wystąpił - zużywają $O(n)$ pamięci i $O(N\log(n))$ czasu, gdzie N jest liczbą wszystkich elementów występujących w rozważanym zbiorze. [**?**]
Omawiane w tej pracy algorytmy nie dają dokładnego wyniku, jednak ich zaletami są: zmniejszenie używanej pamięci oraz zmniejszenie liczby wykonywanych operacji na element. Dla większości praktycznych zastosowań mały błąd estymacji nie ma znaczenia. Przedstawione algorytmy opierają się o pewne fakty statystyczne, które stosuje się do wartości funkcji skrótu (funkcji haszującej) dla poszczególnych elementów zbioru.
W pracy zostaną omówione trzy algorytmy:

- Probabilistic Counting, (2.1)
- HyperLogLog,(2.2)
- Minima Counting(2.3).

Celem tej pracy jest zweryfikowanie jak działają powyższe algorytmy w praktyce oraz ich porównanie. Symulacje zostaną przeprowadzone dla kilku wybranych funkcji haszujących (m.in. SHA-3) na zadanych zbiorach. Wyniki symulacji przedstawione zostaną w rozdziale 4.
W rozdziale 5 zaś omówione zostaną przykłady praktycznych zastosowań algorytmów przybliżonego zliczania, w tym wykorzystanie algorytmów przybliżonego zliczania do badania DNA (ideę tą przedstawił Giroire w pracy [**?**]).

2 Opis wybranych algorytmów

2.1 Probabilistic Counting

Algorytm Probabilistic Counting opiera się na spostrzeżeniu, że jeśli wartości wynikowe funkcji haszującej mają rozkład jednostajny, to w przybliżeniu połowa wartości $h(x)$ będzie zaczynać się od cyfry 1, jedna czwarta będzie zaczynać się od 01 itd. (1 występuje na każdej pozycji z prawdopodobieństwem 1/2). Niech $\rho(x)$ oznacza pozycję wystąpienia pierwszej jedynki w reprezentacji binarnej wartości $h(x)$. Wtedy prawdopodobieństwo, że $\rho(x) = k$ wynosi $\frac{1}{2^k}$. Niech $B = (0,0,\dots)$ będzie wektorem zer. Następnie dla każdego x należącego do badanego zbioru artość $B[\rho(x)] = 1$. Za estymator liczby różnych elementów zbioru możemy przyjąć 2^R, gdzie $R = max\{k : \forall_{i\in\{1,2,\dots,k\}} B[i] = 1\}$. Ostateczna wersja algorytmu została zmodyfikowana tak, by zmniejszyć odchylenie standardowe z σ dla pojedynczego eksperymentu do σ/m dla średniej z m eksperymentów. Otrzymanie m wartości różnych funkcji haszujących dla każdego elementu wymagałoby znajomości konstrukcji niezależnych funkcji haszujących oraz zwiększenia czasu potrzebnego na wykonanie algorytmu. Rozwiązaniem tego problemu jest uśrednianie stochastyczne, które zostało przedstawione w pracy Flajoleta ([**?**]).
Zaletą takiego podejścia jest nie tylko fakt, że potrzebna jest tylko jedna funkcja haszująca, ale również to, że nie wykonuje się wielu operacji dla każdego elementu zbioru.
Kolejna poprawka algorytmu wykorzystuje funkcje tworzące i transformatę Mellina, a związana jest z obciążeniem estymatora. Stąd

$$\hat{n} := \frac{1}{\varphi} m 2^A,$$

gdzie A jest średnią arytmetyczną $R_1, \dots, R_m$, $\varphi := \frac{e^\gamma}{\sqrt{2}} \prod_{j=1}^{\infty} (\frac{2j+1}{2j})^{\varepsilon(j)} \approx 0.7735$, γ oznacza stałą Eulera, a $\varepsilon(j) = \pm 1$ wskazuje parzystość liczby wystąpień 1 w reprezentacji binarnej j.
Tak skonstruowany estymator ma następujące właściwości asymptotyczne (względem n):

- $\frac{E(\hat{n})}{n} = 1 + \varepsilon(m) + \delta_1(m,n) + o(1)$
- $SE(\hat{n}) = \eta(m) + \delta_2(m,n) + o(1)$

gdzie δ_1, δ_2 są pewnymi funkcjami oscylacyjnymi takimi, że $|\delta_1(m,n)| < 10^{-5}$, $|\delta_2(m,n)| < 10^{-5}$ oraz wraz ze wzrostem m: $\varepsilon(m) \sim \frac{0.31}{m}$ i $\eta(m) \sim \frac{0.78}{\sqrt{m}}$

2.2 HyperLogLog

Pewnym rozwinięciem metody Probabilistic Counting jest HyperLogLog. Zamiast pamiętać cały wektor B ustalany jest licznik $C = max_{x \in M} \rho(x)$. Również dla tej metody stosuje się uśrednianie stochastyczne, stąd potrzebnych jest m takich liczników $C_1, C_2, \ldots, C_m$. Główna różnica między opisywaną metodą a Probabilistic Counting polega na użyciu średniej harmonicznej zamiast arytmetycznej. Tak skonstruowany estymator jest postaci:

$$\hat{n} := \frac{m}{\alpha_m} \frac{m}{\sum_{j=1}^{m} 2^{-C_j}},$$

gdzie $\alpha_m = \int_0^\infty (log(\frac{2+u}{1+u}))^m du = 2log2 + O(\frac{1}{m})$. Estymator ten ma mniejszą wariancję niż estymator uzyskany metodą Probabilistic Counting.

Dla $m \geq 16$ zachodzą następujące właściwości asymptotyczne (względem n):

- $\frac{E(\hat{n})}{n} = 1 + \delta_1(n) + o(1)$
- $SE(\hat{n}) = \frac{\beta_m}{\sqrt{m}} + \delta_2(n) + o(1)$

gdzie $|\delta_1(n)| < 10^{-5}$, $|\delta_2(n)| < 10^{-5}$ oraz β_m jest pewną stałą.

2.3 Minima Counting

Wadą opisanych algorytmów Probabilistic Counting i HeyperLogLog, jest niewątpliwie fakt, że oba estymatory są obciążone. Wady tej nie posiada ostatni z przedstawianych w tej pracy algorytmów: algorytm opierający się na statystyce pozycyjnej. Podobnie jak dla poprzednich algorytmów każdy element zbioru M przekształcamy przy pomocy funkcji haszującej. Wartości te traktowane są dalej jako losowe liczby rzeczywiste z przedziału $[0,1]$. Dla minimum ($U_{[1:n]}$) zachodzi:

$$E[U_{[1:n]}] = \int_0^1 xn(1-x)^{n-1}dx = \frac{1}{n+1}.$$

Można zatem otrzymać aproksymację $\hat{n}$ z powyższego wzoru. Ważny jest fakt, że zaobserwowane minimum jest niewrażliwe na strukturę powtórzeń w zbiorze. Tak stworzony algorytm zużywa stałą pamięć oraz tylko jedną zmienną przechowującą minimum. Podejście to ma jednak wady: po pierwsze odchylenie standardowe dla $U_{1:n}$ jest tego samego rzędu co wartość oczekiwana, a po drugie odwrotność $U_{1:n}$ ma nieskończoną wartość oczekiwaną ([**?**], [**?**]). Żeby uniknąć tych problemów wystarczy użyć drugiej lub dalszej statystyki pozycyjnej. Dla praktycznych zasotsowań Giroire w swojej pracy sugeruje statystykę $U_{3:n}$.

Podobnie jak dla dwóch algorytmów zaprezentowanych wcześniej, również dla tego algorytmu można zastosować uśrednianie stochastyczne, co poprawi dokładność estymacji. W tym przypadku m eksperymentów symuluje się dzieląc $(0,1)$ na m równych przedziałów. Dla każdej wartości $h(x)$ liczba $\lfloor mh(x) - mh(x) \rfloor$ należy do i-tego przedziału gdzie $i = \lfloor mh(x) \rfloor + 1$. Algorytm wybiera k-tą statystykę pozycyjną z każdego przedziału, przy czym, jeżeli w którymś przedziale znajduje się mniej niż k wartości wtedy $U^{(i)}_{k:n} = 1$. Za estymator liczności $\hat{n}$ można przyjąć: $\hat{n} = \sum_{i=1}^{m} \frac{k-1}{U^{(i)}_{(k:n)}}$.

Asymptotycznie (względem n) zachodzi: $SE(\hat{n}) = \frac{1}{\sqrt{k-2}} \frac{1}{\sqrt{m}}$.

3 Implementacja algorytmów

3.1 Probabilistic Counting

3.2 HyperLogLog

3.3 Minima Counting

4 Porównanie eksperymentalne algorytmów

5 Przykłady praktycznych zastosowań

5.1 Duże bazy danych

Liczba unikalnych wartości w kolumnie jest jedną z podstawowych informacji używanych do optymalizacji zapytań oraz przy projektowaniu bazy danych. Dla optymalizacji zapytań oszacowana liczba wartości używana jest przy wyborze minimalnego pod względem kosztu planu wykonywania kwerendy. W konstrukcji bazy danych natomiast stosuje się ją do konfiguracji dostępu, który daje minimalny czas odpowiedzi dla zestawu transakcji użytkownika. Zatem, oszacowanie liczności różnych wartości w kolumnie ma kluczowe znaczenie dla uzyskania dobrej wydajności bazy danych. Jednak pojawianie się powtórzeń utrudnia otrzymywanie dynamiczne liczności w kolumnie przy każdej operacji dodania lub usunięcia krotki. Problem ten można zmniejszyć utrzymując indeksy kolumn, jednak w wielu praktycznych zastosowaniach, nie tworzy się indeksów dla wszystkich kolumn. Rozwiązaniem tego problemu jest zastosowanie algorytmów zliczania proponowane w pracach „Probabilistic counting algorithms for data base applications" ([**?**]), „A Linear-Time Probabilistic Counting Algorithm for Database Applications"([**?**]).

5.2 Analiza DNA

Bardzo ciekawy przykład wykorzystania algorytmów przybliżonego zliczania został podany przez Giroire w pracy [**?**]. Algorytmy zliczania użyte zostały do analizy korelacji w ludzkim genomie. Korelacja mierzona jest poprzez liczbę wystąpień różnych podciągów o ustalonej długości k we fragmencie DNA rozmiaru N. Fragment z niewieloma różnymi podciągami jest bardziej skorelowany niż fragment tego samego rozmiaru, w którym występuje więcej różnych podciągów. Giroire odpowiada w swojej pracy na trzy pytania:

- Czy w genomie występują wszystkie teoretycznie możliwe „słowa" (4^k) czy też niektóre wzorce nie pojawiają się?
- Czy genom jest jednorodny czy też niektóre jego fragmenty są bardziej skorelowane niż inne?
- Jaka jest częstotliwość występowania różnych „słów" w genomie (rozumianych jako sekwencja czytana „od początku").

5.3 Zliczanie obiektów

Algorytmy zliczania znajdują również zastosowanie w problemie aproksymacji liczby osób (lub dowolnych obiektów mobilnych) - możliwe jest monitorowanie ruchu oraz zagęszczenia np. na trasach narciarskich czy imprezach masowych. Zastosowanie algorytmów zliczania pozwala oszacować dużą liczbę osób przy minimalnym zużyciu zasobów, a więc umożliwia implementację na słabych urządzeniach wykorzystując niewielki obszar pamięci. Dodatkowo sposób zapamiętywania informacji spełnia wymagania ochrony danych osobowych.

Podsumowanie