(CS602) Applied Algorithms 23 Jan, 2014

Lecture 7: Count-Min and Count sketches

Lecturer: Nutan Limaye *Scribe: Nutan Limaye*

In the last class we studied two different randomized approximation algorithm for computing all frequency moments using $O(\frac{1}{\varepsilon} \cdot \log\left(\frac{1}{\delta}\right) \cdot \log m)$ space and additive approximation error. We fully analyzed the first algorithm. Recall that the algorithm used a sketch which is known as Count-Min sketch, which was developed by Cormode and Muthukrishnana [2]. Towards the end of the last class, we presented the second algorithm and we were in the process of analysing the second algorithm. The second algorithm was developed by Charikar, Chen and Farach-Colton [1]. It also uses a sketch, which is known as Count sketch.

Both these sketching algorithms have found a plethora of applications in varied areas of computer science including compressed sensing, natural language processing, databses, and networking. (See for example: `https://sites.google.com/site/countminsketch/`)

We will finish the analysis of the second algorithm in this class.

7.1 Recalling the Count sketch algorithm

Let us start by recalling the algorithm we presented last time, the notation we introduced, and the Lemmas we proved.

Pick $h_1, h_2, \ldots, h_t$ uniformly randomly from pairwise independent family of functions $\mathcal{F} = \{f : [m] \to [k]\}$ and pick $g_1, g_2, \ldots, g_t$ uniformly randomly from pairwise independent family of functions $\mathcal{G} = \{g : [m] \to \{\pm\}\}$;
for $i = 1$ *to* t **do**
  for $j = 1$ *to* k **do**
    $C[i][h_i(j)] \leftarrow 0$;
  end
end
while *there exists* x*, an input element* **do**
  for $i = 1$ *to* t **do**
    $C[i][h_i(x)] \leftarrow C[i][h_i(x)] + g_i(x)$;
  end
end
On query a, output $\text{Median}_{1 \le i \le t} \{g_i(a) C[i][h_i(a)]\}$;

Like in the previous section, let for $j \in [n] \setminus \{a\}$, $Y_{i,j}$ denote the excess in the counter $C[i][h_i(a)]$.

Then we have, $Y_{i,j} = \begin{cases} f_j & \text{if } h_i(a) = h_i(j) \text{ and } g_i(j) = +1 \quad \text{(with probability } 1/2k\text{)} \\ -f_j & \text{if } h_i(a) = h_i(j) \text{ and } g_i(j) = -1 \quad \text{(with probability } 1/2k\text{)} \\ 0 & \text{otherwise} \qquad\qquad\qquad\qquad \text{(with probability } 1 - 1/k\text{)} \end{cases}$

Let $Y_i := \sum_j Y_{i,j}$. The output of the algorithm and the Y_is are related in the following way: Y_i counts the total error terms in $C[i][h_i(a)]$. Therefore, $C[i][h_i(a)] = Y_i + g_i(a) \cdot f_a$.

In order to bound the error in the output, let us first compute the expected value of Y_i and variance of Y_i.

Lemma 6.2.1. For every $1 \leq i \leq t$, $\mathbb{E}(Y_i) = 0$ and $\mathbb{V}ar(Y_i) = \frac{\sum_{j \in [m] \setminus \{a\}} f_j^2}{k}$.

The proof of this emma was presented in the last lecture note.

7.2 Analysis of the algorithm

We now use Lemma 6.2.1 to analyze the performance of the algorithm. Let $\hat{\hat{f}}_a$ denote the output of the algorithm. Therefore, $\hat{\hat{f}}_a = \text{Median}\{\hat{\hat{f}}_{a,i}\}$, where $\hat{\hat{f}}_{a,i} = g_i(a) \times (Y_i + g_i(a) \cdot f_a)$. We will first compute the expectation and variance of $\hat{\hat{f}}_{a,i}$.

Lemma 7.2.1. *For every* $1 \leq i \leq t$, $\mathbb{E}(\hat{\hat{f}}_{a,i}) = f_a$ *and* $\mathbb{V}ar(\hat{\hat{f}}_{a,i}) = \frac{\sum_{j \in [m] \setminus \{a\}} f_j^2}{k}$

Proof.

$$\begin{aligned}
\mathbb{E}(\hat{\hat{f}}_{a,i}) &= \mathbb{E}(g_i(a) \times (Y_i + g_i(a) \cdot f_a)) \\
&= \mathbb{E}(g_i(a)Y_i + f_a) && (\text{As } g_i(a)^2 = 1) \\
&= \mathbb{E}(g_i(a)Y_i) + f_a \\
&= f_a + \sum_{j=1}^{k} \mathbb{E}(g_i(a)Y_{i,j}) \\
&= f_a + \sum_{j=1}^{k} \frac{+f_j}{4k} + \frac{-f_j}{4k} + \frac{+f_j}{4k} + \frac{-f_j}{4k} \\
&= 0
\end{aligned}$$

Remark 7.2.2. *Note that as an intermediate step, the above proof also proved that* $\mathbb{E}(g_i(a)Y_i) = 0$. *We will use this fact in the analysis of the variance.*

$$\begin{aligned}
\mathbb{E}(\hat{\hat{f}}_{a,i}^2) &= \mathbb{E}\left((g_i(a)Y_i + f_a)^2\right) && (\text{From Step 2 above}) \\
&= f_a^2 + \mathbb{E}\left(g_i(a)^2 Y_i^2 + 2 f_a g_i(a) Y_i\right) \\
&= f_a^2 + \mathbb{E}\left(Y_i^2\right) + 2 f_a \mathbb{E}(g_i(a)Y_i) && (\text{As } g_i(a)^2 = 1) \\
&= f_a^2 + \mathbb{E}\left(Y_i^2\right) + 0 && (\text{By Remark 7.2.2}) \\
&= f_a^2 + \frac{\sum_{j \in [m] \setminus \{a\}} f_j^2}{k} && (\text{By proof of Lemma 6.2.1})
\end{aligned}$$

Therefore, we get $\mathbb{V}ar(\hat{\hat{f}}_{a,i}) = \frac{\sum_{j \in [m] \setminus \{a\}} f_j^2}{k}$ □

By using Chebyshev's inequality, we get $\Pr\left[|\hat{\hat{f}}_{a,i} - f_a| \geq \varepsilon\sqrt{F_2}\right] \leq \frac{\sum_{j\in[m]\setminus\{a\}} f_j^2}{k\varepsilon^2 F_2}$, where $F_2 = \sum_{j\in[m]} f_j^2$. By choosing k appropriately, we can bound this probability by $1/3$, that is $\Pr\left[|\hat{\hat{f}}_{a,i} - f_a| \geq \varepsilon\sqrt{F_2}\right] \leq 1/3$. Now by the standard median trick, by choosing $t = O\left(\log \frac{1}{\delta}\right)$, we get $\Pr\left[|\hat{\hat{f}}_{a,i} - f_a| \geq \varepsilon\sqrt{F_2}\right] \leq \delta$ for any constant $\delta > 0$.

7.3 Comparison of the two algorithms

Recall that in the last lecture note we proved that $\Pr\left[|\hat{f}_{a,i} - f_a| \geq \varepsilon F_1\right] \leq \delta$ for any constant $\delta > 0$. That is, in the Count-Min algorithm, the additive approximation error was proportional to F_1 where as here it is proportional to $\sqrt{F_2}$. However, the dependence on ε is $1/\varepsilon$ for Count-Min and $1/\varepsilon^2$ for Count sketch.

References

[1] Finding frequent items in data streams. *Theoretical Computer Science*, 312(1):3 – 15, 2004. ¡ce:title¿Automata, Languages and Programming¡/ce:title¿.

[2] Graham Cormode and S. Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. *J. Algorithms*, 55(1):58–75, 2005.

7.4 Exercises

Exercise 1. *In the algorithm presented here, say the last line is changed from*

$$\text{“On query } a\text{, output Median}_{1\leq i\leq t}\{g_i(a)C[i][h_i(a)]\}\text{”}$$

to

$$\text{“On query } a\text{, output } \frac{\sum_{i=1}^{t}\{g_i(a)C[i][h_i(a)]\}}{t}$$

The rest of the algorithm is kept as it is. Analyze the performance of this modified algorithm.