0368.3239 Foundations of Data Mining - Assignment 1

November 11, 2017

1 The Misra Gries Sketch and Stream Order

(a) The following arrangement (from left to right) will maximize the count of x_1:

$$\underbrace{x_2,...,x_2}_{10\text{ elements}},\ \underbrace{x_3,...,x_3}_{10\text{ elements}},\ \underbrace{x_4,...,x_4}_{10\text{ elements}},\ \underbrace{x_5,...,x_5}_{10\text{ elements}},\ \underbrace{x_1,...,x_1}_{50\text{ elements}}$$

Claim: The count of x_1 in the final sketch will be 50, which is maximal for the given element frequencies and sketch size.

Proof. We will show that the final count is 50 directly by following the Misra Gries algorithm, as described in class. It is clearly the maximal count possible, since there is a total of 50 elements of this key in the stream.

n	**sketch state**
10	$\underbrace{10}_{x_2}\ \underbrace{\ }_{?}$
20	$\underbrace{10}_{x_2}\ \underbrace{10}_{x_3}$
30	$\underbrace{\ }_{?}\ \underbrace{\ }_{?}$
40	$\underbrace{10}_{x_5}\ \underbrace{\ }_{?}$
90	$\underbrace{10}_{x_5}\ \underbrace{\mathbf{50}}_{\boldsymbol{x_1}}$

After the first 10 elements the sketch will have one key x_2 with count of 10. Then, the elements of x_3 arrive, and since the size of the sketch is 2, the new key will be added with count of 10. Next, there are 10 elements of x_4. Since the sketch is full, for every element of x_4 each of the counts of x_2, x_3 will be decreased by one. Since there are exactly 10 incoming elements of x_4 and each of the counts of x_2, x_3 is also 10, the resulting sketch will be empty. Then, x_5 elements will be counted, while there is still place in the sketch for one key, which will then be filled with x_1. □

(b) Plugging into the lemma we proved in class, we are guaranteed that this estimate is smaller than the true count by at most $\frac{90-60}{2+1} = \frac{30}{3} = 10$. In other words, the actual frequency of x_1 is at least 50 (the counter value) and at most $50 + 10 = 60$.

(c) The following arrangement (from left to right) will minimize the count of x_1:

$$\underbrace{x_1, ..., x_1}_{\text{50 elements}}, \underbrace{x_2, ..., x_2}_{\text{10 elements}}, \underbrace{x_3, ..., x_3}_{\text{10 elements}}, \underbrace{x_4, ..., x_4}_{\text{10 elements}}, \underbrace{x_5, ..., x_5}_{\text{10 elements}}$$

Claim: The count of x_1 in the final sketch will be 30, which is minimal for the given element frequencies and sketch size.

Proof. First, we will show that the count of x_1 in the final sketch is 30. Similarly to subsection (a), by following the algorithm it is easy to see that the final count for x_1 will be 30:

n	**sketch state**
50	$\underbrace{50}_{x_1}\underbrace{\ }_{?}$
60	$\underbrace{50}_{x_1}\underbrace{10}_{x_2}$
70	$\underbrace{40}_{x_1}\underbrace{\ }_{?}$
80	$\underbrace{40}_{x_1}\underbrace{10}_{x_4}$
90	$\underbrace{\mathbf{30}}_{\boldsymbol{x_1}}\underbrace{\ }_{?}$

Now, we will show that this count is minimal. Assume that there is an arrangement with a smaller count for x_1, then there were at least 21 events of two types: (1) a "decrease" event while x_1 was one of the two keys in the sketch (2) a "decrease" event due to incoming x_1 element.

Notice that every "decrease" event of those types requires two non-x_1 incoming elements. For case (1) we need one element to fill the second slot in the sketch and another one to cause the decrease in counts. For case (2) we need two different elements to fill the slots in the sketch. But as are 4 non-x_1 keys of frequency 10 each, the total of non-x_1 incoming elements is exactly 40. Thus, there are at most 20 such "decrease" events possible, which is a contradiction to the initial assumption. Therefore, the minimal count for x_1 is 30. □

(d) By the same arguments as in subsection (b), this estimate is smaller than the true count by at most $\frac{90-30}{2+1} = \frac{60}{3} = 20$. Namely, the actual frequency of x_1 is at least 30 (the counter value) and at most $30 + 20 = 50$.

2 Flippers Randomized Counter

(a) We will show that $\hat{n} = p^{-1}s$ is an unbiased estimator for n.

It is enough to show that the expected increase of estimate is 1.

Proof.

$$\begin{aligned} & Pr[u \le p] \cdot (p^{-1}(s+1) - p^{-1}s) + (1 - Pr[u \le p]) \cdot 0 \\ &= \underbrace{\frac{p-0}{1-0}}_{u \sim U[0,1]} \cdot p^{-1} = p \cdot p^{-1} = 1 \end{aligned}$$

□

(b) Let X_n be a random variable of counter with input n, our estimate is $\hat{n} = p^{-1}X_n$.

Here:

$$Var[\hat{n}] = E[(\hat{n} - E[\hat{n}])^2] = E[(p^{-1}X_n - E[p^{-1}X_n])^2] = p^{-2} \cdot E[(X_n - E[X_n])^2] = p^{-2} \cdot Var[X_n]$$

Notice that X_n is a binomial distributed random variable, with $X_n \sim B(n,p)$, therefore:

$$Var[\hat{n}] = p^{-2} \cdot Var[X_n] = p^{-2} \cdot np(1-p) = n(p^{-1} - 1) = n\Big(\frac{1}{p} - 1\Big)$$

And:

$$CV = \frac{\sqrt{Var[\hat{n}]}}{E[\hat{n}]} = \frac{\sqrt{n(p^{-1}-1)}}{n} = \sqrt{\frac{1}{n}\Big(\frac{1}{p} - 1\Big)}$$

(c) The likelihood function:

$$\begin{aligned} L(s;n) &= \binom{n}{s} p^s (1-p)^{n-s} = \frac{n!}{s!(n-s!)} p^s (1-p)^{n-s} \\ l(s;n) &= \log L(s;n) = \log \frac{n!}{s!(n-s!)} + \log p^s + \log (1-p)^{n-s} \\ &= \log n! - \log s! - \log (n-s!) + s \log p + (n-s) \log (1-p) \end{aligned}$$

(d) Plugging $s = 1$, we get:

$$\begin{aligned} l(s=1;n) &= \log n! - \log 1! - \log (n-1!) + 1 \cdot \log p + (n-1) \log (1-p) \\ &= \log \frac{n!}{(n-1)!} + \log p + (n-1) \log (1-p) = \log n + \log p + (n-1) \log (1-p) \end{aligned}$$

For $p < 1$, we can compute the derivative with respect to n, and compare to zero:

$$\begin{aligned} & \frac{\partial l(s=1;n)}{\partial n} = \frac{1}{n} + \log (1-p) \\ & \frac{\partial l(s=1;n)}{\partial n} = 0 \;\Leftrightarrow\; \frac{1}{n} + \log (1-p) = 0 \\ & \Rightarrow\; \hat{n}_{MLE} = -\frac{1}{\log (1-p)} \end{aligned}$$

By taking the second derivative we can see that this is indeed a maximum:

$$\frac{\partial^2 l(s=1;n)}{\partial n^2} = -\frac{1}{n^2} < 0$$

For $p = 1$, the likelihood function is maximized by:

$$\hat{n}_{MLE} = \arg\max_n L(s=1;n) = \arg\max_n \binom{n}{1} p^1 (1-p)^{n-1} = \arg\max_n \; n \cdot 0^{n-1} = 1$$

Overall we get:

$$\hat{n}_{MLE} = \begin{cases} -\frac{1}{\log(1-p)} & \text{if } p < 1 \\ 1 & \text{if } p = 1 \end{cases}$$

(e) For $n = 1$, s can be either 0 or 1. Consider the likelihood function in subsection (c) for $s = 0$:

$$L(s=0;n) = \binom{n}{0} p^0 (1-p)^{n-0} = (1-p)^n$$

Since $p \in (0, 1]$, then $(1-p) \in [0, 1)$ and:

$$\hat{n}_{MLE} = \arg\max_n (1-p)^n = 0$$

Plugging this result and the MLE we calculated in subsection (d), we get:

$$E[\hat{n}_{MLE}] = (1-p) \cdot 0 + p \cdot -\frac{1}{\log(1-p)} = -\frac{p}{\log(1-p)}$$

$$\Rightarrow Bias[\hat{n}_{MLE}] = E[\hat{n}_{MLE}] - n = -\frac{p}{\log(1-p)} - 1$$

Computing the variance:

$$E[\hat{n}^2_{MLE}] = (1-p) \cdot 0^2 + p \cdot \frac{1}{\log^2(1-p)} = \frac{p}{\log^2(1-p)}$$

$$Var[\hat{n}_{MLE}] = E[\hat{n}^2_{MLE}] - E[\hat{n}_{MLE}]^2 = \frac{p}{\log^2(1-p)} - \frac{p^2}{\log^2(1-p)} = \frac{p-p^2}{\log^2(1-p)}$$

Finally, we can calculate the MSE:

$$\begin{aligned} MSE &= Var[\hat{n}_{MLE}] + Bias[\hat{n}_{MLE}]^2 \\ &= \frac{p-p^2}{\log^2(1-p)} + \Big(-\frac{p}{\log(1-p)} - 1\Big)^2 \\ &= \frac{p-p^2}{\log^2(1-p)} + \frac{p^2}{\log^2(1-p)} + \frac{2p}{\log(1-p)} + 1 \\ &= \frac{p}{\log^2(1-p)} + \frac{2p}{\log(1-p)} + 1 \end{aligned}$$

(f) Flipper counters are composable. Two Flipper counters s_1, s_2 with parameter p can be merged into a Flipper counter s by addition, e.g. $s = s_1 + s_2$

Since both s_1 and s_2 are binomial distributed with parameter p, then so is their sum:

$$s_1 \sim B(n_1, p),\; s_2 \sim B(n_2, p) \;\Rightarrow\; s = s_1 + s_2 \sim B(n_1 + n_2, p)$$

Since:

$$\begin{aligned} P[s=k] &= \sum_{i=0}^{k} \binom{n_1}{i} p^i (1-p)^{n_1-i} \binom{n_2}{k-i} p^{k-i} (1-p)^{n_2-k+i} \\ &= \sum_{i=0}^{k} \binom{n_1}{i} \binom{n_2}{k-i} p^k (1-p)^{n_1+n_2-k} = \binom{n_1+n_2}{k} p^k (1-p)^{n_1+n_2-k} \end{aligned}$$

(g) Table 1 summaries the expected counter size and CV for the three versions of Morris counters we saw in class. We shall compare it with Flipper counters, in those terms.

Table 1: expected counter size and CV of Morris counters

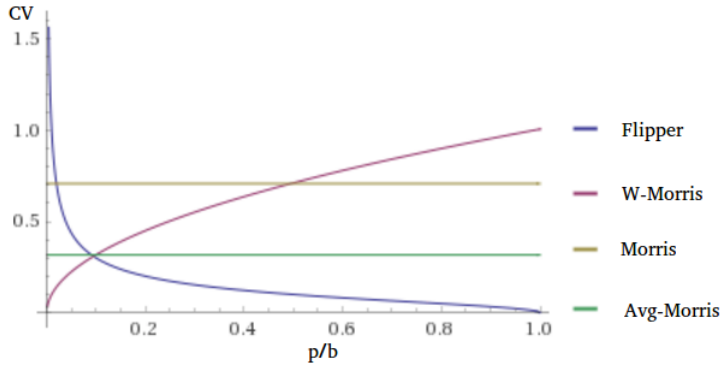
	expected counter size	**CV**
Morris counter	$\log\log n$	$\sqrt{1/2}$
Averaging Morris counter	$k \log\log n$	$\sqrt{1/2k}$
Weigthed Morris counter	$\leq \log\log n + 2\log_2 1/\sqrt{b}$	$\leq\sqrt{b(1+1/n)}$

For Flipper counters, the expected counter size is $\log_2(ps)$, which is dependent on the choice of p. Concretely, for smaller values of p, we will get a lower space requirements. Yet, there is an order of magnitude difference from the expected size of Morris counters, which use only $O(\log_2 \log_2 n)$ bits.

In subsection (b) we found that the CV of Flipper counters is $\sqrt{\frac{1}{n}(\frac{1}{p}-1)}$, which is dependent on the number of incoming elements n. Assuming that $n >> k$, it is significantly lower than the CVs of the original and the averaging Morris counters.

For both Flipper counters and weighted Morris counters, the selection of p and b controls the trade off between smaller variance and bigger counter size. There are choices of b and p for which the CV of the weighted Morris counter is lower than this of the Flipper counter, and vice versa. In general, the decrease in variation is more rapid for the Flipper counter then for the weighter Morris counter. And the opposite is true for the expected counter size.

This is illustrated in the plot below, of the counter CVs as a function of p / b, for $n = 10^4$ and $k = 5$:



To conclude, Flipper counters have the advantage of low variance, with the drawback of relatively big counter size. In contrast, Morris counters are highly efficient in space terms, but we had to reduce their variance to get the final low-variance weighted version.

3 Reservoir Sampling (Vitters)

(a) .

Algorithm: Merge of two reservoir samples

Input : two reservoir samples S_1 and S_2 of size k, of sets of elements E_1, E_2, where $|E_1| = n_1, |E_2| = n_2$

Output: a uniform sample of size k of $E_1 \cup E_2$

Initialize an empty reservoir S of size k
if $|S_1| + |S_2| \leq k$ **then**
  Insert all the elements of S_1 and S_2 into S
else
  Set $c \leftarrow \frac{kn_1}{n_1+n_2}$
  Denote by $\{c\} := c - \lfloor c \rfloor$ (the fractional part of c)
  Set m_1 to:

$$\begin{cases} \lfloor c \rfloor + 1 & \text{with probability } \{c\} \\ \lfloor c \rfloor & \text{with probability } 1 - \{c\} \end{cases}$$

  Set $m_2 \leftarrow k - m_1$
  $M_1 \leftarrow m_1$ randomly selected elements from S_1
  $M_2 \leftarrow m_2$ randomly selected elements from S_2
  Insert all the elements of M_1 and M_2 into S
end
return S

(b) *Proof.* If $|S_1| + |S_2| \leq k$, then the inclusion probability of every element $e \in S_1 \cup S_2$ in S is 1. Furthermore, $n_1 \leq k$ and $n_2 \leq k$ and by the correctness of Reservoir sampling it holds that:

$$\forall e \in E_i\,, i = 1, 2 \,:\, P[e \in S] = P[e \in S_i] \cdot P[e \in S \,|\, e \in S_i] = \min\left\{1, \frac{k}{n_i}\right\} \cdot 1 = 1$$

Now, consider the case of $|S_1| + |S_2| > k$ and let $e \in S_1 \cup S_2$. If $e \in S_1$, then:

$$\begin{aligned} P[e \in S] = P[e \in M_1] = \frac{m_1}{|S_1|} &= \{c\} \cdot \frac{\lfloor c \rfloor + 1}{|S_1|} + (1 - \{c\}) \cdot \frac{\lfloor c \rfloor}{|S_1|} \\ &= \frac{\lfloor c \rfloor + \{c\}}{|S_1|} = \frac{\lfloor c \rfloor + c - \lfloor c \rfloor}{|S_1|} = \frac{c}{|S_1|} = \frac{1}{|S_1|} \cdot \frac{kn_1}{n_1 + n_2} \end{aligned}$$

Similarly, if $e \in S_2$, then:

$$P[e \in S] = P[e \in M_2] = \frac{m_2}{|S_2|} = \frac{k - m_1}{|S_2|} = \frac{k - c}{|S_2|} = \frac{k}{|S_2|} \cdot \left(1 - \frac{n_1}{n_1 + n_2}\right) = \frac{1}{|S_2|} \cdot \frac{kn_2}{n_1 + n_2}$$

Therefore, by the correctness of Reservoir sampling:

$$\forall e \in E_i\,, i = 1, 2 \,:\, P[e \in S] = P[e \in S_i] \cdot P[e \in S \,|\, e \in S_i] = \min\left\{1, \frac{k}{n_i}\right\} \cdot \frac{1}{|S_i|} \cdot \frac{kn_i}{n_1 + n_2}$$

If $k \geq n_i$, then $|S_i| = n_i$ and

$$P[e \in S] = 1 \cdot \frac{1}{n_i} \cdot \frac{kn_i}{n_1 + n_2} = \frac{k}{n_1 + n_2}$$

If $k < n_i$, then $|S_i| = k$ and

$$P[e \in S] = \frac{k}{n_i} \cdot \frac{1}{k} \cdot \frac{kn_i}{n_1 + n_2} = \frac{k}{n_1 + n_2}$$

Overall, we showed that the final inclusion probability of each element $e \in E_1 \cup E_2$ in S is $\min\{1, \frac{k}{n_1+n_2}\}$. □

4 Bloom filter with deletion

(a) We assume that there are only insertions of elements that are not currently stored in the data structure and deletions of elements that are currently stored.

To support insertions and deletions, we will modify the Bloom filter structure as follows. Instead of a single bit, every entry of the Bloom filter structure will contain a counter of length $l > 1$ bits. The sketch API is described below, where x is a key and S_1, S_2 are two structures of the same size and set of hash functions:

Method: Initialize()

```
1 Declare counter array S of size m;
2 for i ← 1 to m do
3 |  S[i] ← 0;
4 end
```

Method: Insert(x)

```
1 for i ← 1 to k do
2 |  S[h_i(x)] ← S[h_i(x)] + 1;
3 end
```

Method: MembershipQuery(x)

```
1 for i ← 1 to k do
2 |  if S[h_i(x)] == 0 then
3 |  |  return False;
4 |  end
5 end
6 return True;
```

Method: Delete(x)

```
1 for i ← 1 to k do
2 |  S[h_i(x)] ← S[h_i(x)] − 1;
3 end
```

Method: Merge(S_1, S_2)

```
1 Declare counter array S of size m;
2 for i ← 1 to m do
3 |  S[i] ← S_1[i] + S_2[i];
4 end
```

Due to our assumptions, there are only deletions of stored elements. Since an insertion increments all hit-by-a-hash-function counters by 1, and deletion of an element decreases the same counters that were hit by its insertion, it is not possible that a counter will be decremented while it is set to 0.

Furthermore, an element is being inserted only if it was not in the structure before. As long as this element is not deleted, all of its counters will be set to at least 1, meaning that the membership query will return True. Therefore, it is not possible to get false positives.

(b) Denote by l the length of each counter, in bits. First, we will bound the probability that one counter overflows.

For every $i \in 1, ..., m * k$, let X_i be an indicator for whether the counter j was hit by the i'th mapping of an element to a counter with one of the k hash functions.

Considering the size n of the Bloom filter, the probability of hitting a counter j is

$$Pr[X_i = 1] = \frac{1}{n}$$

and the expected number of hits of counter j is:

$$\mu = E[\sum_{i=1}^{mk} X_i] = \sum_{i=1}^{mk} E[X_i] = \sum_{i=1}^{mk} P[X_i = 1] = \frac{mk}{n}$$

An overflow occurs when the number of times a particular counter was hit is greater than 2^l. Thus, the probability that counter j overflows is

$$Pr[\text{counter } j \text{ overflows}] = Pr[\sum_{i=1}^{mk} X_i > 2^l] = Pr[\sum_{i=1}^{mk} X_i \geq 2^l + 1] = Pr[\mu \sum_{i=1}^{mk} X_i \geq \mu(2^l + 1)]$$

Since $2^l > 0$, we can apply Chernoff bound to have:

$$Pr[\text{counter } j \text{ overflows}] \leq \exp\left\{-\frac{2^l \ln(1 + 2^l)\mu}{2}\right\}$$

Now, we want to bound the probability that any counter overflows. Using the union bound and the identity $\ln(1 + x) \leq x$, we get:

$$\begin{aligned} Pr[\text{any counter overflows}] &\leq \sum_{j=1}^{n} Pr[\text{counter } j \text{ overflows}] \\ &\leq n \cdot \exp\left\{-\frac{2^l \ln(1 + 2^l)\mu}{2}\right\} \leq n \cdot \exp\left\{-\frac{\ln^2(1 + 2^l)\mu}{2}\right\} \end{aligned}$$

We want that the probability of an overflow will be at most δ. Hence, we shall require

$$n \cdot \exp\left\{-\frac{\ln^2(1 + 2^l)\mu}{2}\right\} \leq \delta$$

and get a bound on the length l:

$$\begin{aligned} & -\frac{\ln^2(1 + 2^l)\mu}{2} \leq \ln\left(\frac{\delta}{n}\right) \\ \Rightarrow \quad & \ln^2(1 + 2^l) \geq \frac{2}{\mu} \ln\left(\frac{n}{\delta}\right) \\ \Rightarrow \quad & 2^l \geq \exp\left\{\sqrt{\frac{2}{\mu} \ln\left(\frac{n}{\delta}\right)}\right\} - 1 \\ \Rightarrow \quad & l \geq \log_2\left(\exp\left\{\sqrt{\frac{2}{\mu} \ln\left(\frac{n}{\delta}\right)}\right\} - 1\right) = \log_2\left(\exp\left\{\sqrt{\frac{2n}{mk} \ln\left(\frac{n}{\delta}\right)}\right\} - 1\right) \end{aligned}$$

5 Maximum likelihood estimator

(a) Given k independent samples $s_1, ..., s_k \sim Exp[n]$, in class we derived the MLE for estimating n:

$$\hat{n}_{MLE} = \frac{k}{\sum_{i=1}^{k} s_i}$$

Let $\theta = g(n) = \frac{1}{n}$, we will show that $\hat{\theta}_{MLE} = \frac{\sum_{i=1}^{k} s_i}{k}$

Proof. For $n > 1$, g is a one-to-one function, therefore:

$$L(s_1, ..., s_k \,;\, n) = L(s_1, ..., s_k \,;\, g^{-1}(g(n)))$$

And both are maximized by $\hat{n}_{MLE}$. Hence, the RHS is maximized by $g(\hat{n}_{MLE})$ and:

$$\begin{aligned}
&\hat{n}_{MLE} = g^{-1}(g(\hat{n}_{MLE})) = g^{-1}(\widehat{g(n)}_{MLE}) \\
&\Rightarrow \widehat{g(n)}_{MLE} = g(\hat{n}_{MLE}) = \frac{1}{\hat{n}_{MLE}} \\
&\Rightarrow \hat{\theta}_{MLE} = \frac{1}{\hat{n}_{MLE}} = \frac{\sum_{i=1}^{k} s_i}{k}
\end{aligned}$$

□

Let $\lambda = f(n) = n^2$, we will show that $\hat{\lambda}_{MLE} = \left(\frac{k}{\sum_{i=1}^{k} s_i}\right)^2$

Proof. Although f is a many-to-one function, for the given domain it is single valued (since $n > 0$). Therefore:

$$L'(s_1, ..., s_k \,;\, \lambda) = \sup_{n:f(n)=\lambda} L(s_1, ..., s_k \,;\, n) = L(s_1, ..., s_k \,;\, \hat{n}_{MLE})$$

Namely, $\hat{n}_{MLE}$ maximizes the two likelihood functions, therefore:

$$\hat{\lambda}_{MLE} = f(\hat{n}_{MLE}) = \hat{n}_{MLE}^2 = \left(\frac{k}{\sum_{i=1}^{k} s_i}\right)^2$$

□

(b) Following the definition in class, the sufficient statistic is a function independent of the parameter, that is used for estimating some function of the parameter.

Therefore, considering the parameter n, the sufficient statistics for $\frac{1}{n}$ and n^2 are the same as the sufficient statistic for n, which is: $\sum_{i=1}^{k} s_i$