Proof for Morris' Approximate Counting Algorithm

Wei Wang, HKUST(GZ)

2024-10-31

The Algorithm

1. **Initialization**: Set the counter $c = 0$.
2. **Processing Each Event**: For each new event, increment the counter c with probability $\frac{1}{2^c}$. Otherwise, leave c unchanged.
3. **Final Count**: After processing n events, the counter is X_n.

Goal

Show that:

$$\mathbb{E}[2^{x_n}] = n + 1$$

Proof by Induction

Let's define $E_n = \mathbb{E}[2^{x_n}]$.

Note that $\mathbb{E}[2^{x_n}] = \sum_{c=0}^{\infty} 2^c \mathbb{P}[X_n = c]$

> Comment:
>
> - This is a critical step, though it is just the vanilla definition of expectation.
> - X_n's range is actually only $[0, n]$, but we can *extend* the domain of it to include invalid values up to ∞ by justing assign 0 to them. Strictly speaking, we shall use a different random variable, say $Z_e n$, which has one-to-one correspondence with X_n.

1. **Base Case**:
   - Before any events $n = 0, c = 0$.
   - Thus, $E_0 = \mathbb{E}[2^{x_0}] = 2^0 = 1$
2. **Inductive Step**:
   - Assume that after $n - 1$ events, $E_{n-1} = n$.
   - Consider the n-th event:
     - **With probability** $\frac{1}{2^c}$: Increment the counter c by 1.

  - **With probability** $1 - \frac{1}{2^c}$: Keep the counter c unchanged.
- The expected value after the n-th event is:

$$\mathbb{E}[2^{x_n} \mid X_{n-1} = c] = 2^{c+1} \cdot \frac{1}{2^c} + 2^c \cdot \left(1 - \frac{1}{2^c}\right) = 2 + 2^c - 1 = 2^c + 1$$

- Taking the expectation over all possible c after $n-1$ events:

$$E_n = \mathbb{E}[2^{x_n}] = \sum_{c=0}^{\infty} \mathbb{E}[2^{x_n} \mid X_{n-1} = c] \cdot \mathbb{P}[X_{n-1} = c]$$

$$= \sum_{c=0}^{\infty} (2^c + 1) \cdot \mathbb{P}[X_{n-1} = c] = \mathbb{E}[2^{X_{n-1}}] + 1 = E_{n-1} + 1$$

  (Use the definition of $\mathbb{E}[2^{x_n}]$ **reversely**)

3. **Solving the Recurrence**:
   - Starting from $E_0 = 1$:

$$E_1 = E_0 + 1 = 2, \quad E_2 = E_1 + 1 = 3, \quad \dots, \quad E_n = n + 1$$

Conclusion

By induction, we've established that:

$$\mathbb{E}[2^{x_n}] = n + 1$$

Comment

It is actually more important to appreciate the significance of the recurrence relationship $E_n = E_{n-1} + 1$. This shows that the algorithm is also doing the **counting**, just probabilistically.

```
Deterministic Counting | Probabilistic Counting
-----------------------------------------------------------------
n <- n + 1             |  X_{n+1} <- X_{n} + 1 probabilistically
                       |  with the guarantee: E_{n+1} = E_{n} + 1
```

Or, you can think about the algorithm tracks $\log_2(n+1)$ probabilistically.

Further Readings

The original paper is [1], with an advanced analysis in [2]. See [3]'s "Algorithm" section for an *efficient bit-level* implementation of the algorithm. See [4] for the lower/upper bound results for the problem.

References

1. Robert Morris: Counting Large Numbers of Events in Small Registers. CACM 1978.

2. PHILIPPE FLAJOLET: APPROXIMATE COUNTING: A DETAILED ANALYSIS. BIT 1985.
3. Wikipedia: Approximate counting algorithm. https://en.wikipedia.org/wiki/Approximate_counting_alg
4. Jelani Nelson, Huacheng Yu: Optimal bounds for approximate counting. https://arxiv.org/abs/2010.02116