COMP 251 : Assignment 4

Xin Rui Li
260870605

3. Consider $n = 3$

Cost for flipping the index at $k=2$ : $2^2 = 4$

Cost for flipping the index at $k=1$ : $2^1 = 2$

Cost for flipping the index at $k=0$ : $2^0 = 1$

| Binary Bits | | | Accumulated Cost |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 (+1) |
| 0 | 1 | 0 | 4 (+1 +2) |
| 0 | 1 | 1 | 5 (+1) |
| 1 | 0 | 0 | 12 (+4 +2 +1) |
| 1 | 0 | 1 | 13 (+1) |
| 1 | 1 | 0 | 16 (+1 +2) |
| 1 | 1 | 1 | 17 (+1) |

Next increment is going to take $2^3 + 2^2 + 2^1 + 2^0$.

<u>Aggregating Method</u>

| Bit | Frequency flipped | Time |
|---|---|---|
| 0 | 1 | 1 |
| 1 | 1/2 | 2 |
| 2 | 1/4 | 4 |
| 3 | 1/8 | 8 |
| ... | ... | ... |
| $i \geq k$ | 0 | 0 |

The accumulated cost is then defined by:

$$\sum_{i=0}^{k-1} \frac{n}{2^i} \cdot 2^i = n \cdot \sum_{i=0}^{k-1} \frac{2^i}{2^i}$$

$\Rightarrow n \cdot \sum_{i=0}^{k-1} 1 = n \cdot k$ where $k$ is the number of bits and $n$ is the number of increments.

Since $k \leq \log(n)$, then $nk \leq n \log(n)$.

So the accumulated cost for $n$ increments is $O(n \log(n))$ such that for each increment, the amortized cost is $O(\log(n))$. $\square$

4. $K'$ is the smallest integer such that $2^{k'} << n$ for which it is safe to reset the binary counter to 0.

$$0 \underbrace{1\ 1\ 1 \ldots 1\ 1}_{k'-1} + 1 = 1 \underbrace{0\ 0\ 0 \ldots 0\ 0}_{k'-1}$$

a) Cost of a reset operation:

There are $k$ bits flips required and the cost for each flip depends on its index such that:

$$\sum_{i=0}^{k'-1} 2^i = \frac{1 - 2^{k'}}{1 - 2} = -(1 - 2^{k'}) = 2^{k'} - 1$$

We can assign an amortized cost of $2 \cdot 2^i = 2^{i+1}$ for each increment where we can use $2^i$ to flip the bit from 0 to 1 and store $2^i$ to flip it back to 0 later.

Since for each bit, we are storing an additional $2^i$, when it needs to reset all bits to zero, there will be:

$$\sum_{i=0}^{k'-1} 2^i = 2^{k'} - 1 \text{ Stored.}$$

Since this is equal to the cost of reset, the credit will never go negative.

In conclusion, for $n$ operations, we have a maximum total cost of $n \cdot 2^{k'+1}$ (since $2^{i+1} \leq 2^{k'+1}$). Since $2^{k'+1}$ is a constant, the increment operation is $O(n)$.

Therefore, for each increment operation, it takes $O(1)$.
Similarly, since the cost of reset is a constant, the reset operation takes $O(1)$. $\square$