Algorithms: Amortized Analysis

Model 1: Incrementing a binary counter

	5	4	3	2	1	0
0	0	0	0	0	0	0
1	0	0	0	0	0	1
2	0	0	0	0	1	0
3	0	0	0	0	1	1
4	0	0	0	1	0	0
5	0	0	0	1	0	1
6	0	0	0	1	1	0
7	0	0	0	1	1	1
8	0	0	1	0	0	0
9	0	0	1	0	0	1
10	0	0	1	0	1	0
11	0	0	1	0	1	1
12	0	0	1	1	0	0
13	0	0	1	1	0	1
14	0	0	1	1	1	0
15	0	0	1	1	1	1

Model 1 shows a binary counter, stored as an array of bits with the 2^i place stored at index i, undergoing a sequence of increment operations. The indices are shown at the top, and the number represented by each state of the binary counter is shown at the left.

Note that the array is drawn with index 0 at the right side instead of the left!

1 How many bits differ between the counter in state 0 and state 1?

2 How many bits differ between states 1 and 2? Between 2 and 3? Between 3 and 4?

3 In general, describe which bits need to flip to go from state n to state $n + 1$.

4 Write pseudocode to perform an increment operation, given an array of bits b as an input.

You do not need to worry about overflowing the array.

5 If we assume that changing the value of a bit takes 1 time step, what is the best-case runtime of your algorithm when given a counter representing some number n? Express your answer using big-Θ notation.

6 Give an example of a best-case input for your algorithm.

7 What is the worst-case runtime of your algorithm when given a counter representing some number n? Express your answer in terms of n, using big-Θ notation.

Careful! n is the *number represented by* the bits, not the number of bits.

8 Give an example of a worst-case input for your algorithm.

9 Based on your answer to Question 5, what is the best-case total running time for a sequence of n increment operations?

10 Based on your answer to Question 7, what is the worst-case total running time for a sequence of n increment operations?

STOP

Model 2: Total cost of repeated increments

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
cost of $(n-1) \to n$		1	2	1	3												
cumulative cost	0	1	3	4	7												

11 Start by filling in the missing values in the table above. Each value in the second row counts the number of bit flips needed to increment a binary counter from $(n-1)$ to n, and each value in the third row is the sum of all the values in the second row so far.

12 How many bit flips are needed, in total, to start at 0 and repeatedly increment a binary counter until reaching 16?

13 Look at the third row and compare it to the first row. What patterns do you notice?

14 Make a conjecture: how many total bit flips will be needed to increment from 0 to 32?

15 In general, how many bit flips do you think will be needed to increment up to 2^k?

16 Generalize your conjecture to give an *upper bound* on the total number of bit flips needed to increment from 0 to any n (not necessarily a power of 2). That is, can you say anything about how big the entries in the third row can get, relative to n?

17 Based on your conjecture, if we repeatedly increment a binary counter from 0 up to n, how long does each increment take *on average*? Express your answer using big-O notation.

18 Why is this an interesting result?

Hint: look at your answers to Questions 7–10.

STOP

Model 3: The accounting method

$$0000 \xrightarrow{\$\$} 000\underset{\$}{1} \xrightarrow{\$\$} 00\underset{\$}{1}0 \xrightarrow{\$\$} 00\underset{\$}{1}\underset{\$}{1} \xrightarrow{\$\$} 0\underset{\$}{1}00 \xrightarrow{\$\$} 0\underset{\$}{1}0\underset{\$}{1} \xrightarrow{\$\$} 0\underset{\$}{1}\underset{\$}{1}0 \xrightarrow{\$\$} 0\underset{\$}{1}\underset{\$}{1}\underset{\$}{1} \xrightarrow{\$\$} \underset{\$}{1}000$$

Now, let's actually prove your conjecture from Question 16. Imagine that we are providing a binary counter service. Anyone can ask us to create a binary counter for them, which starts out with the value zero. They can then ask us to increment their counter whenever they want (they can also ask us to tell them the current value). Every time we flip a bit, it costs us $1. How much should we charge our customers for an increment operation to make sure that we can at least break even?

Perhaps it should be called `countr.com`... drat, that domain is already taken.

19 Explain why charging $1 for an increment operation is not enough. That is, if we charge only $1 for every increment operation, it will not be enough to cover our costs and we will eventually go bankrupt.

The model shows what happens if we charge $2 per increment operation. The two $$ signs over each arrow represent the $2 paid by a customer every time they want to increment their counter. In between the arrows are various states of the counter *along with extra money we have saved*.

20 Consider the first arrow in the diagram,

$$0000 \xrightarrow{\$\$} 000\underset{\$}{1}.$$

Explain what is happening with the money. Why is there $1 left over? Why do you think we save it under the 1?

21 Now explain the second step,

$$000\underset{\$}{1} \xrightarrow{\$\$} 00\underset{\$}{1}0.$$

Why is there $1 left over again? What do you think we did with the $1 that was stored under the rightmost 1? What did we do with the $2 paid by the customer?

22 Now explain

$$00\underset{\$}{1}0 \xrightarrow{\$\$} 00\underset{\$}{1}\underset{\$}{1} \xrightarrow{\$\$} 0\underset{\$}{1}00.$$

23 Can you explain why charging \$2 per increment operation will ensure that we always have enough money to cover our costs?

24 This actually constitutes a proof of your conjecture from Question 16. Explain why.