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 Next to each counter state in the model, write the number of bits that changed from the previous state. Circle the bits that changed.

4 Now, highlight the bits that changed *from zero to one*.

5 What patterns do you notice?

6 How many bits are there that change from zero to one each time?

7 How do the bits that change from zero to one relate to the bits that change from one to zero?

8 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.

9 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.

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

11 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*.)

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

13 Based on your answer to Question 9, what is the best total running time we could possibly hope for a sequence of n increment operations?

14 Based on your answer to Question 11, what is the worst possible 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												

15 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.

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

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

Hint: look at powers of two. There's no one right answer to this question.

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

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

20 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?

21 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.

22 Why is this an interesting result?

Hint: look at your answers to Questions 11–14.