Lesson 10 Proving the run-time of the doubling algorithm

# Learning goal

Derive the run-time of algorithms by numerical investigation and by mathematical proof.

# Agenda

1. Deriving the run-time of the exponential-increment algorithm for the Hiker Bridge problem
2. Work on Assignment 5.

## Deriving run-time of the doubling algorithm for the Hiker-Bridge problem

In the proof below, we’ll need the formula for geometric series we learned last year:

Bridge

Hiker start

2s = gain of 1

s

4s

8s = gain of 4

16s

32s = gain of 16

For simplicity, let’s assume s = 1. Let k = the number of steps away from us the bridge is.

|  |  |  |
| --- | --- | --- |
| Movement (right or left) | Final position relative to start | Gain since the last good move |
| -1 | -1 |  |
| 2 | **1** | 1 – 0 = **1 = 40** |
| -4 | -3 |  |
| 8 | **5** | 5 – 1 = **4= 41** |
| -16 | -11 |  |
| 32 | **21** | 21 – 5 = **16= 42** |
| -64 | -43 |  |
| 128 | **85** | 85 – 21 = **64 = 43** |

We’ve reached the bridge when the total gains add up to (or exceed) k.

Let T = the number of two-way trips it takes us to reach the bridge.

In the diagram, T = 3 because we needed 3 back-and-forths to reach the bridge.

In other words, the number of two-way trips needed to find the bridge is the value of T when

1 + 4 + 42 + 43 +… 4T ≥ k

Whatever T is (we’ll find it in a moment), the number of *one*-*way* trips will be 2T.

Then the total distance we walk will be

d(k) = 1 + 2 + 22 + 23 +… + 22T-1 + 22T = , using the red formula for geometric series with *a* = 2 and *n* = 2T.

Once we find T and simplify that expression, we’ll apply Big-O notation to it, and that will be the worst-case running time for the doubling algorithm.

Now let’s figure out T using the equation 1 + 4 + 42 + 43 +… 4T = k

1 + 4 + 42 + 43 +… 4T = , again using the red formula for geometric series with *a* = 4.

4T+1 = 3k + 1

22(T+1) = 3k + 1

Since we want to solve for T, we need to get it out of the exponent. We can do this by taking

log-base-2 of both sides. This gives us

2(T+1) = log2 (3k+1)

T =

So the number of one-way trips is 2T, which equals log2 (3k+1) – 2

At last we’re ready to substitute this value of 2T into the blue distance formula above.

The total distance traveled is

= 2 log (3k+1) – 2 + 1 – 1

= 2 log (3k+1) – 1  – 1

= 2 log (3k+1)/2 – 1

= , (since 2log P = P for any value of P)

Thus, the total distance the hiker walks as a function of k is

But this is a linear function of k! Thus, the doubling algorithm runs in O(k) time, which crushes the O(k2 ) run-time of the constant-growth algorithm.