Why don't I know anyone who knows anyone with COVID-19?

William John Holden

March 25, 2020

Abstract

You have an 85% chance of not knowing anyone who knows anyone with COVID-19. This estimate is computed from a repeatable simulation in Julia based on a U.S. population of 330 million, log-normal social connectivity with $\hat{\mu} = 6.2$ and $\hat{\sigma} = 0.68$, and the current case count of 44 183.

1 Introduction

A friend of mine asks a question along the lines of "given the six degrees to Kevin Bacon effect, why do I not know anyone *who knows anyone* with COVID-19?" This is a fair question that deserves analysis.

"Six degrees of Kevin Bacon" is a widely understood effect of highly connected networks. The idea is that you can name any actor/actress, then name another actor/actress with whom they appeared with, and repeat the process until you reach Kevin Bacon. If we were to represent all actors in a large graph we might say the *radius* of the graph (from Kevin Bacon) is no more than six. Equivalently, the shortest path (measured in units of "co-starring") from any actor to Kevin Bacon is at most six.

In 2016, Facebook reported that the *average* distance between any random pair of Facebook users is only 3.47 [1]. This result seems too incredible to be true, even if we accept the Kevin Bacon thing.

COVID-19 has had a profound global impact this year, yet many of us (including myself) are surprised that we have no direct or even indirect connection to this pandemic. Just how large is the social bubble around COVID-19 cases?

2 Social bubbles

For a small population we could simply generate a random social network, run the Floyd-Warshall algorithm, and count the 0's, 1's, and 2's in the resulting distance matrix. Unfortunately for us, finding all-pairs shortest paths is computationally infeasible at just 10 000

[1] https://research.fb.com/blog/2016/02/three-and-a-half-degrees-of-separation/

vertices. To simulate a country of more than 1 000 000 people we need a less exhaustive approach.

We will explore the graph with something akin to a breadth-first search. In this paper, a "bubble" is the size of the social network with r degrees of separation. At $r = 0$, a person's bubble has unit size (the bubble only contains the person themselves). At $r = 1$, the bubble includes all of the people the person *directly* knows. At $r = 2$, the bubble also includes all "friends of friends" with a radius of at most 2. The simulation will:

1. Begin with an empty set S. S is the set of all people in a COVID-19 bubble, which we will expand to a radius of 2.

2. Randomly generate a set of integers that will represent COVID-19 cases (bubbles of radius $r = 0$). Add each of these integers to S.

3. For each case u, randomly generate a set of integers v that represent the patients' friendships (bubble of radius $r = 1$). Add all integers v to S.

4. For each friend v, randomly generate a second set of integers w of the friends' friends (bubble of radius $r = 2$). Add all integers w to S.

5. Find the size of S. This gives us an estimate of how large the combined bubbles around COVID-19 cases are. The ratio of the size of S divided by the population size is the probability that you know someone who knows someone with COVID-19.

3 Parameters

3.1 U.S. population

We still need to decide on a few parameters before we get started. First, we are going to bound the integers that represent people from $[1, n]$. What is n? n should be the size of the population. The U.S. has a national population of about 330 million people (United States Census Bureau, 2019), so we set $n = 330\,000\,000$ [2].

3.2 Connectivity

Next, how many friends do we give each person? This is more difficult to answer. McCormick, Salganik, and Zheng (2013) estimate that the average person has a mean of 611 friends [3]. They model the mean "degree" as a log-normal distribution with $\hat{\mu} = 6.2$ and $\hat{\sigma} = 0.68$.

Julia does not have a built-in `rlnorm` function [4]. We can improvise with $e^{\mu + \sigma \texttt{randn()}}$.

```
julia> using Random, StatsBase

julia> prng = MersenneTwister(2020);
```

[2] https://www.census.gov/quickfacts/fact/table/US/PST045219

[3] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3666355/

[4] https://svn.r-project.org/R/trunk/src/nmath/rlnorm.c

```
julia> network = randn(prng, 8, 8) |> x -> x * 0.68 |> x -> x .+ 6.2 |> x ->
    exp.(x) |> x -> round.(x) |> x -> Int.(x)
8×8 Array{Int64,2}:
  524  790  911  574  1462   179   517   264
  836  741  487  485  2008   178  1565   269
  396  572  153  538   505   476   831   302
  190  762  682  858   916  1013   399   145
  268  260  679  452   868   594   343  1203
  872  306  519  423   886   361   115  1216
 1109  769  805  597   834   459   554   262
  349  990  688  557   330   318   217   811

julia> mean(network)
617.84375
```

The above pipeline generates an 8×8 matrix of random numbers. These numbers are centered at 0 and have a standard deviation of 1. Multiply each by 0.68 to change the standard deviation, then add 6.2 to uniformly shift the mean. Take the element-wise exponent e^x and round the result to the nearest integer. Looks reasonable, I suppose. We can consolidate this logic as a function and plot the outputs on a histogram.

```
friends(rng) = Int(round(exp(6.2 + (0.68 * randn(rng)))))
using Plots
histogram([friends(prng) for i=1:10000])
```

The above call to `histogram` generates the plot shown in figure 1.

3.3 Bubble size at $n = 0$

We now have a cap on the population to simulate (330 million) and a `friends` function to randomly generate a realistic number of relations between them. The final parameter we need is to decide on how large a bubble we should begin with. This is easy: as of 25 March 2020, the official coronavirus confirmed case count for the U.S. was 44 183 (Centers for Disease Control and Prevention, 2020) [5].

4 Simulation

```
S = Set()

n = 330000000 # U.S. population estimate
c = 44183 # official U.S. COVID-19 count as of 25 March 2020

```

[5] https://www.cdc.gov/coronavirus/2019-ncov/cases-updates/cases-in-us.html

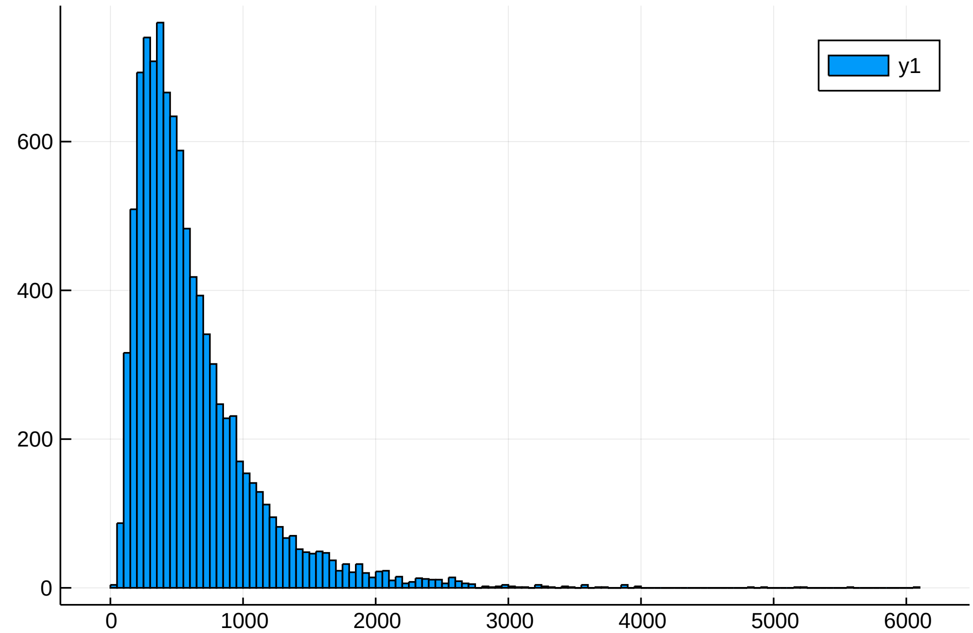

Figure 1: Estimated degree per person

```
friends(rng) = Int(round(exp(6.2 + (0.68 * randn(rng)))))

using Random
prng = MersenneTwister(2020)

# Begin constructing social bubbles to represent confirmed cases.
bubble0 = rand(prng, 1:n, c)
union!(S, bubble0)
println("Degree 0: size of S = ", length(S)) # Almost c (no duplicates)

# Expand the bubble by one degree. These are the friends and families of
# persons directly impacted.
bubble1 = [rand(prng, 1:n, friends(prng)) for i in bubble0]
foreach(x -> union!(S, x), bubble1)
println("Degree 1: size of S = ", length(S)) # Predictably close to c * 611.

# Anonymously expand the bubble to the second degree. The bubble now has all
# of the COVID-19 patients, their friends and families, and all of those
# peoples' friends and families.
foreach(y -> union!(S, rand(prng, 1:n, y)), map(length, bubble1))
println("Degree 2: size of S = ", length(S)) # Much smaller than c * 611^2!
```

5 Result

The output of this program may surprise you. The size of the bubble at $r = 0$ is (obviously) almost 44 183. The size of the bubble at $r = 1$ is almost $44\,183 \times 611$ (the average number of friends). However, the size of the bubble at $r = 2$ is nowhere near $44\,183 \times 611^2 = 16\,494\,441\,743$. By degree 2, even with completely random associations and no clustering (which we know is not realistic), the program outputs:

```
Degree 0: size of S = 44181
Degree 1: size of S = 26210074
Degree 2: size of S = 50302654
```

Only 50 million. The social bubbles around the 44 thousand COVID-19 patients in the U.S., a country of 330 million people with average social connectivity around 610 people, contain 50 million people.

So why don't you know anyone who knows anyone with COVID-19?

```
julia> length(S) / n
0.15243228484848484
```

It is because you are among the 85% of the country outside of the COVID-19 bubbles.

6 Note

I am not a professional researcher and I welcome your constructive feedback. Feel free to contact me at `wjholden@gmail.com` or `https://twitter.com/wjholdentech`.