The main concern we had with this design was that there is the possibility of creating ongoing loops in the paths. Depending on the graph, these loops could repeat themselves over and over again and decrease the probability of producing a correct Hamiltonian Path.

Take, for instance, the Adleman graph:



Possible loops here include: 1-2-7-2-7-2-7…

2-3-6-7-2-3-6-7-2-3-6-7…

and others

 We were afraid that the presence of these loops might make the odds of getting a correct answer so small that this might not be worth doing. We are in the process of writing a simulation program in Python. While we don’t have full results yet we have successfully run a simulation of the Adleman graph. At this point it seems that the odds are indeed small, but they are not necessarily worse than the odds we found for the “column method,” for which we would need approximately 35,000,000 cells to be 99% confident of producing a correct answer.

 This number was found by first calculating that the column method could produce 146 different answers and then assuming that each answer is equally likely.

To have a probability greater than or equal to .99 of at least 1 success we need n results such that n= ln(.01)/ln(1-1/146)