CMPE556: Homework

Ceyhun Onur - 2018700102

March 19, 2019

We investigated the 1D ring(lattice) network in literatures and papers through the internet. There is not a clear formula for the average path length of k-neighbour lattice graphs. However it is discussed that [1] and in [2] it is roughly equal to $n/4k$. In the [2] Chapter "9.4.1 Computing the Average Path Length" the formula is derived as follow: n is vertices, and k is neighbourhood (i.e jump) which makes the average degree of the graph $2k$

$$avp \approx \frac{2k * (1 + 2 + 3 + ... + \frac{n}{2k})}{n} \tag{1}$$

$$1 + 2 + 3 + ... + x = \sum_{i=1}^{x} i = \frac{x * (x + 1)}{2} \approx \frac{x^2}{2} \tag{2}$$

Using (2) it is derived:

$$avp \approx \frac{2k * (\frac{n}{2k})^2}{2n} = \frac{n}{4k} \tag{3}$$

There is a network package called NetworkX [3] for Python. It also provides average shortest path calculation for general graphs using Dijsktra or Bellman-Ford shortest path algorithms. I also found a small script that can generate lattice(ring) graphs with n, k [4]. Generated random lattice graphs and calculated the average shortest paths with Dijkstra, Bellman-Ford, first derivation of formula (1) and the second(final) derivation of formula (3). You can find a sample run on below with visualized graphs.

```
***** Run #1*****
Number of vertices(n): 10
Number of neighbours(k): 3
Dijsktra: 1.33333333333
Bellman-ford: 1.33333333333
First estimation: 0.6
Second estimation: 0.833333333333
***** Run #2*****
Number of vertices(n): 18
Number of neighbours(k): 8
Dijsktra: 1.05882352941
Bellman-ford: 1.05882352941
First estimation: 0.888888888889
Second estimation: 0.5625
***** Run #3*****
Number of vertices(n): 14
Number of neighbours(k): 4
Dijsktra: 1.38461538462
Bellman-ford: 1.38461538462
First estimation: 0.571428571429
Second estimation: 0.875
***** Run #4*****
Number of vertices(n): 10
Number of neighbours(k): 1
Dijsktra: 2.77777777778
Bellman-ford: 2.77777777778
First estimation: 2.0
Second estimation: 2.5
```

The difference between calculations is less in big numbers:

```
***** Run #1*****
Number of vertices(n): 200
Number of neighbours(k): 9
Dijsktra: 6.03015075377
Bellman-ford: 6.03015075377
First estimation: 5.94
Second estimation: 5.55555555556
***** Run #2*****
Number of vertices(n): 2000
Number of neighbours(k): 9
Dijsktra: 56.028014007
Bellman-ford: 56.028014007
First estimation: 55.944
Second estimation: 55.5555555556
```

You can find source code in `https://github.com/ceyonur/lattice_calculator`

References

[1] Andreas Reppas, Konstantinos Spiliotis, and Constantinos Siettos. "Tuning the average path length of complex networks and its influence to the emergent dynamics of the majority-rule model". In: *Mathematics and Computers in Simulation* 109 (Mar. 2015), pp. 186–196. DOI: 10.1016/j.matcom.2014.09.005.

[2] Shai S. Shen-Orr et al. "Chapter 9 Graphs and Networks 9 . 1 Examples of Networks in the Real World". In:

[3] *NetworkX*. URL: https://networkx.github.io/.

[4] Allen B. Downey. *Think Complexity*. Green Tea Press, 2016. ISBN: 9781492040200. URL: http://greenteapress.com/complexity2/html/thinkcomplexity2004.html#chap03-1.

Figures

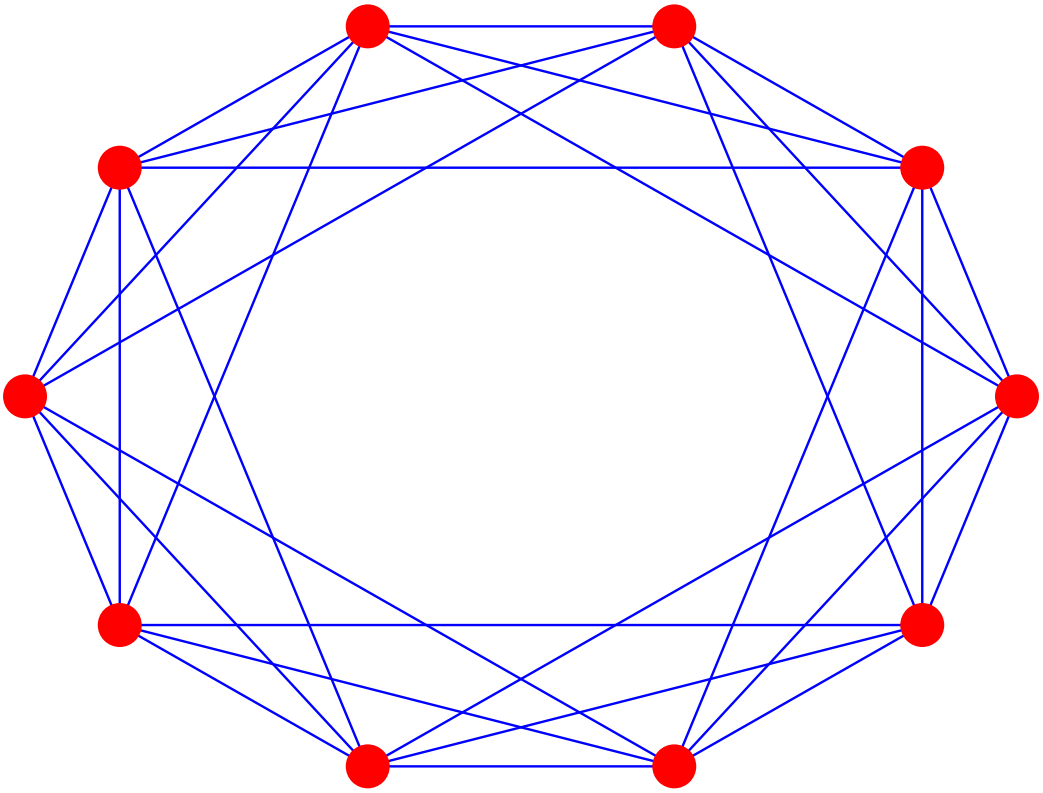

Figure 1: n=10, k=3

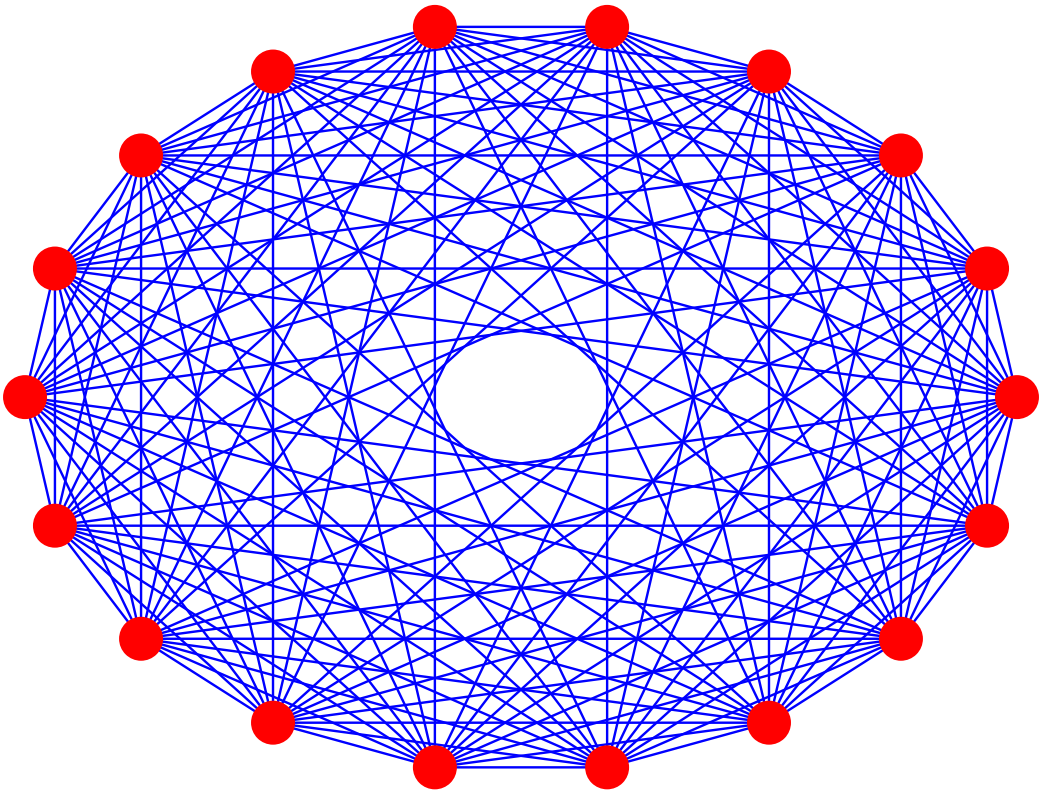

Figure 2: n=18, k=8

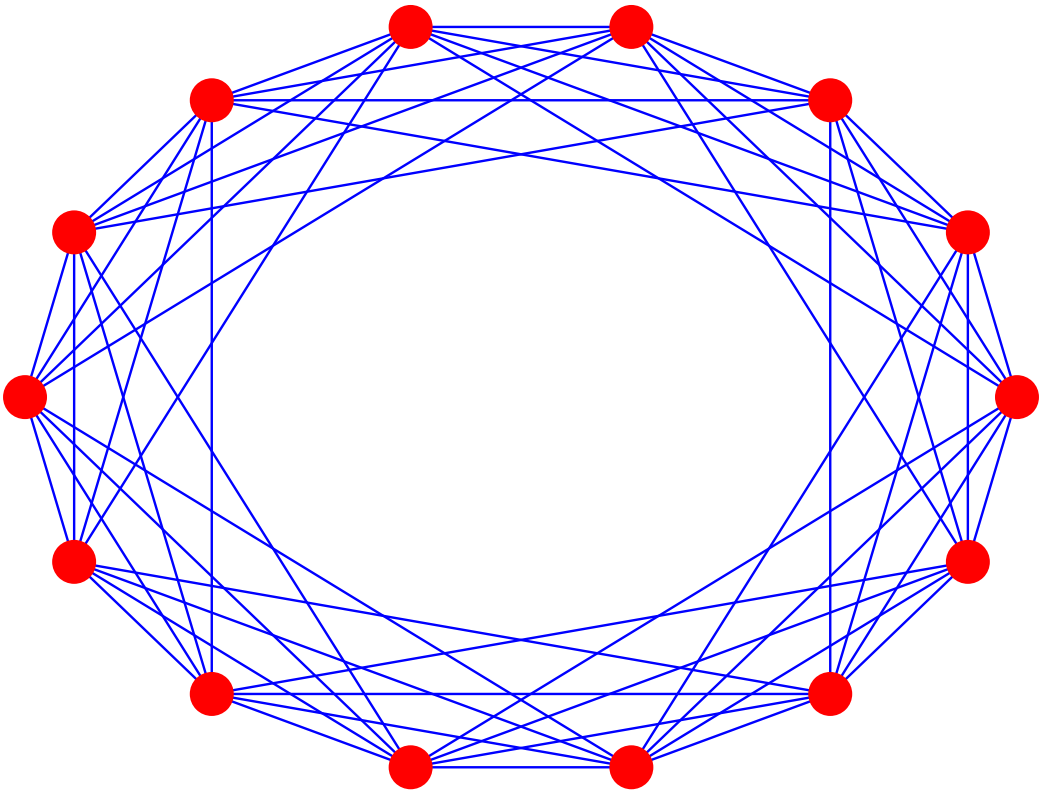

Figure 3: n=14, k=4

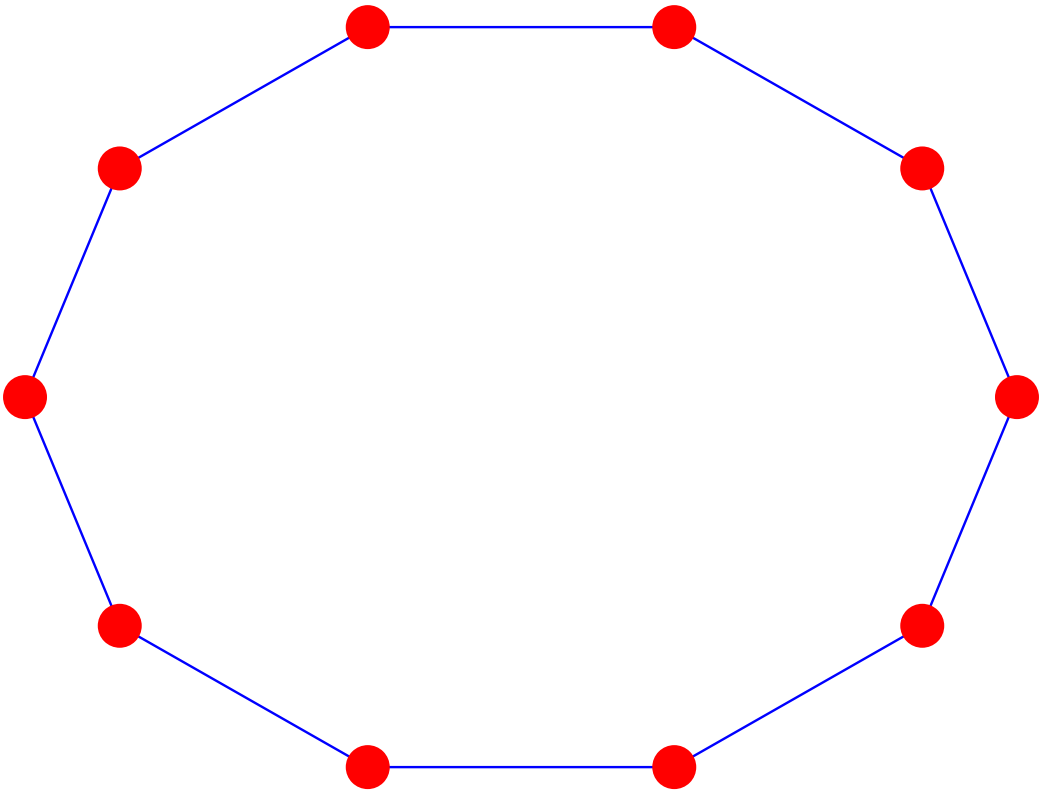

Figure 4: n=10, k=1