15-112: Fundamentals of Programming and Computer Science
Competitive Analysis

Michael Rosenberg
Kosbie, Section J
mmrosenb@andrew.cmu.edu
April 15, 2015

Summary

The growth of the field of network science has coincided with a growth in software dedicated to its analysis. Many of these are libraries and packages available in certain programming languages, but a few of these are stand-alone executable applications. It is important to consider how this division and other deeper design choices have influenced the usability and power of these applications. I will be looking at three particular modules/applications: *Krackplot*, *NetworkX*, and *iGraph*.

Krackplot

- *Krackplot* is a social network visualization tool created by David Krackhardt, a Professor of Organizations at the Heinz School of Public Policy and Management[1]. Krackplot is an application that uses a standard graphical user interface where one can move around nodes, assign attributes to nodes, establish edges between nodes, establish several metrics on the graph, and print a given graph. See figure 1 example of the kind of output that one can produce through an adjacency matrix.

- **Pros:**

  - The visual customization suite is quite robust. the ability to add an endless number of characteristics to nodes and edges and to manipulate the shapes, colors, and mechanisms of nodes and edges provides a level of flexibility that social network analysts need.

  - Because it does not need any background in scripting and programmatic techniques to make a network in this application, it provides a level of accessibility that is not found

[1]Dr. David Krackhardt - Professor in the Heinz School of Public Policy and Management. David Krackhardt, 2006. Web. 14 Apr. 2015. ¡https://www.andrew.cmu.edu/user/krack/about.shtml¿.

in libraries that are dedicated in programming languages.

```
Manuel  0.398084  0.307494  Nancy
Stuart  0.283429  0.357614  Nancy Charles Manuel
Charles 0.358046  0.633273  Nancy Stuart Manuel
Donna   0.709291  0.234592  Nancy Manuel
Sharon  0.025000  0.300659  Stuart
Fred    0.186973  0.025000  Stuart
Bob     0.092337  0.148022  Fred Sharon Stuart
Harold  0.470881  0.956775  Charles
Wynn    0.157854  0.861091  Charles
Carol   0.319828  0.975000  Charles
Kathy   0.763889  0.309772  Tanya Nancy Donna
Nancy   0.529119  0.435072  Donna
Tanya   0.783908  0.423681  Nancy Kathy Donna
Susan   0.975000  0.264209  Tanya Kathy Donna
```

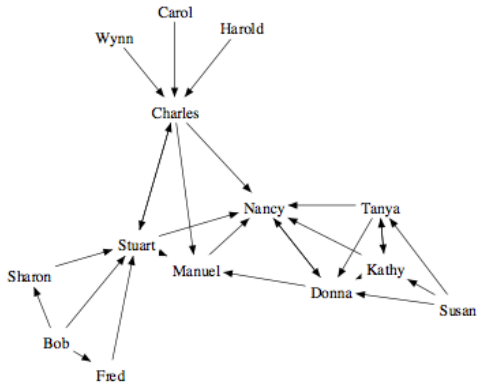

Figure 1: An example of the graph output produced by Krackplot.

- **Cons:**
  - This is primarily a network visualization tool. As such, the metric-producing functions in the application are lacking. Not only are the metrics lacking in robustness, but they also are measures that are arguably outdated by the field's current standards.
  - It is meant to be primarily for small networks due to the graphic limits of visualizing large networks. Hence, it doesn't quite have the power to represent the modern networks created by "big data."
- **The Takeaway:** The visualization tools and diversity of customization options are the aspects that I would like in my front-end for my project. However, We could make a significantly more powerful network visualization tool if we could establish better and more robust metrics

in the back-end for the project.

NetworkX

- *NetworkX* is a library in Python. It is widely used in network analysis due to both the accessibility of Python and the expansiveness of the library. One can build various different classes of graphs (DiGraph, MultiGraph, Spectral Graph, etc.), use built-in advanced algorithms to produce metrics and find special structural properties of graphs (k-cores, cliques, subgraphs), and visualize graphs. There are even parts of this library that can draw three-dimensional graphs.
- **Pros:**
  - The pure power of this package is overwhelmingly useful for persons trying to analyze very large networks. The structure of the graph objects is essentially a nested dictionary of various node and edge dictionaries. This efficiency in data structure allows for millions of nodes and millions of edges to be represented visually and algorithmically. There is simply no other package that has this much power.
  - The fact that various algorithms (measuring clustering coefficients, calculating assortativity or homophily, and finding maximum flows through Ford-Fulkerson and Edmonds-Karp) are built in to this library is very convenient for a user who wants to quickly analyze a given network dataset.
- **Cons:**
  - It assumes a familiarity with Python and Python data structures that may not be as known among various researchers interested in network theory. Due to the broad number of fields that interact with network theory, we cannot assume that every researcher who deals with these topics will know this programming language.
  - While it does contain some drawing algorithms, like many non-design-oriented graphics packages, the methods used to draw the graphs can be rather clunky. This is very

frustrating for the user who is simply interested in the visualization aspect of network science rather than the analysis

- **The Takeaway:** I want to have an application that has the level of power and complexity that NetworkX provides for the network science community. However, by taking away the need to understand how to program, I can add greater accessibility to tools and functions of network science.

---

iGraph

- *iGraph* is a library of network analysis tools that can be accessed and programmed in R, Python, and C. Due to my personal familiarity with the R version of the library, I will discuss this version in this section. This package contains several methods and objects that allow us to build a variety of different graphs studied network science, such as the Barabasi-Albert scale-free graph and the Erdos̈-Renyi random graph. it also contains several mathematical algorithms, such as graph composition and graph complement, that make it more than just a network visualization tool.

- **Pros:**

  - It's availability on R makes it a very accessible network analysis tool for statisticians who focus their work on Bayesian Network Analysis. The syntax and readability of R make it a preferable language among researchers, and thus this package has a major reach in academia.
  - Similar to *NetworkX*, it carries a lot of diverse algorithms that make it a very powerful tool for a network scientist who is trying to quickly retrieve information about the network.

- **Cons:**

  - The visualization suite is not as strong as our previous two options. It can produce 2D-graphs of our networks, but due to the low power of R, it can only do this for small

numbers of vertices and edges. This is unfortunate because the actual methods for making these graphs in R are considerably less clunky than making these graphs using *NetworkX*.

  - Some of the more advanced algorithms featured in *NetworkX* are not built-in to iGraph. While the library has a wide variety of mathematically intriguing graphs, algorithms such as Ford-Fulkerson and Shortest Augmenting Path are not immediately available, which then puts extra work on the user who needs information that is provided by these algorithms

- **The Takeaway:** In terms of its reach and its position on a rather accessible language, iGraph provides a strong introductory platform for new network researchers. However, it would be better to have a networks interface that provided them with a stronger visualization suite and more powerful algorithms on the onset in order to empower the researcher with the benefits of studying ideas in network science.

Conclusion

Overall, each of these modules/applications provide the user with strong tools and methods in order to learn more about networks. However, each of these modules/applications have certain ways of falling short of providing these desired properties: accessibility, algorithmic power, and dynamic visualization. Hence, when going farther into my term project, I hope to consider how my design choices will effect these three desired properties. Emphasizing these three properties in design will allow me to accomplish my ultimate goal: create an interface that empowers the researcher with more than just nodes and edges.