Knowledge Graph Visualization for Understanding Ideas

Allen Zhu

Abstract

Knowledge graphs provide information on entities such as objects, events and historical figures, and how they are connected to one another. In this paper we discuss the application of visualization techniques to knowledge graphs and highlight the insights that are made possible. We also propose a browser-based method for exploring knowledge graphs based on self-organizing maps. We developed a browser-based framework called Desic to apply this technique to the Knowledge Web project, and to evaluate its performance.

1. Introduction

Knowledge graphs represent entities such as objects, events and historical figures, and how they are connected to one another.

Knowledge graphs have significant overlap with other aspects of network theory, but differ in terms of emphasis. Knowledge graphs are related to semantic networks, except that entities have physical meaning, and may contain social networks, although interactions with non-social entities are considered as well. The generality of knowledge graphs poses significant challenges, but offers significant rewards.

The generality of knowledge graphs allows for several important uses. First, knowledge graphs can be used to disambiguate user input, because "Kevin Bacon" can refer to a famous actor, a music producer, a senator from Ohio, and an Olympic athlete.

Second, knowledge graphs are a simple and effective way to define entities in a structured and systematic manner, by interpreting connections as properties of the entity. For example, actor Kevin Bacon can be defined by his connections to his hometown, family, college, filmography, and so on.

Third, knowledge graphs provide broader context for how knowledge and innovations develop and create social change.

Fourth, knowledge graphs allow users to understand the creation of ideas. Often, the pieces for technological breakthroughs existed long before someone put them together. Knowledge graphs also demonstrate the social impact of ideas, revealing changes in society and thought.

By randomly exploring knowledge graphs, students and other users may find serendipitous trends or combine ideas in a new way.

2. Application of Self-Organizing Maps

Given the connectionist nature of our work, it is only appropriate that the visualization algorithm for the knowledge graph should be computed by interpreting it as a neural network. Topological mappings are common in the brain, and the brain likely computes information based on neural connectivity. If so, our algorithm is a model of how the brain actually works, and should therefore be optimal for conveying information to humans.

Our method is based on the Self-Organizing Map (SOM), introduced by Kohonen [1] which remains a popular tool for visualizing high-dimensional data. SOM is applied in applications ranging over biology [2], neurology [3], data mining [3], and robotics [4]. A conventional SOM is an unsupervised neural network that performs a topographic mapping from input data in $\mathbb{R}^n$ to a two- or three-dimensional rectangular or hexagonal grid.

During the training phase, all neurons compete with each other for the input signals. The winner and its neighbors update their connection weights.

The algorithm takes as parameters a neighborhood function h, which decreases the response of neurons based on their distance from the winner; an adaptation schedule α, which decreases neuron adaptation over time; and the number of input samples. Common choices are

$$h(dist) = e^{-\frac{dist^2}{diameter}} \text{ and } \alpha(t) = e^{-\frac{t}{t_{max}}}.$$

Algorithm 1: SOM

input: Neural network N
output: layout of G in R^2

for each input sample x:
  find the nearest neuron n with $\min\|x - n_i\|$
  for all neurons n_i with $dist(n, n_i) \geq r$:
    $\overrightarrow{n_i} = \overrightarrow{n_i} + h(dist)\alpha(\vec{v} - \overrightarrow{n_i})$
end

2.1. Optimized SOM for Graph Layout

To adapt SOM for graph drawing, we input the knowledge graph as the neural network. We wish for the knowledge graph to approximate $\mathbb{R}^2$, so we initialize each node with a random vector and provide random vectors as training samples. Similar approaches may be found in [5-7]; however, our algorithm works with general graphs, including disconnected graphs, generalizes to different edge types (different relationships between nodes) and other auxiliary data, has a highly-tuned time performance.

We specifically designed the algorithm to have an effectively linear performance on scale-free graphs. The inner function (find) is implemented by a breadth-first search in O(V) time, and creates a hashmap that allows for O(1) lookup in the for loop. The number of iterations is therefore parameterized by the diameter of the graph; specifically, the harmonic sum

$$\sum_{r=1}^{D=diameter} \left\lceil \frac{1000}{r} \right\rceil = 1000 H_D \sim \log(D).$$

Thus, the theoretic time complexity of our algorithm is $O(V \log D)$. In addition, many networks based on real datasets are thought to scale-free [8-10], which are highly connected with a diameter that grows at a rate below *log V* [11-12]. Consequently, the time complexity may also be expressed as $O(V \log\log V)$. In practice, D is so small as to be thought constant, and performance is effectively linear with respect to the number of vertices.

Algorithm 2: SOM for graph layout

input: Graph *G*
output: layout of *G* in $\mathbb{R}^2$

r := diameter(G)
iterations := $\lceil 1000/r \rceil$

while $r > 0$:
  generate a random vector $\vec{v}$
  find node n with $\min(\|\vec{v} - \vec{n}\|)$
  for all nodes n_i with $dist(n, n_i) \geq r$:
    $\vec{n_i} = \vec{n_i} + h(dist)\alpha(\vec{v} - \vec{n_i})$

  if(*iterations* = 0)
    decrement r
    iterations = $\lceil 1000/r \rceil$
  else
    decrement iterations

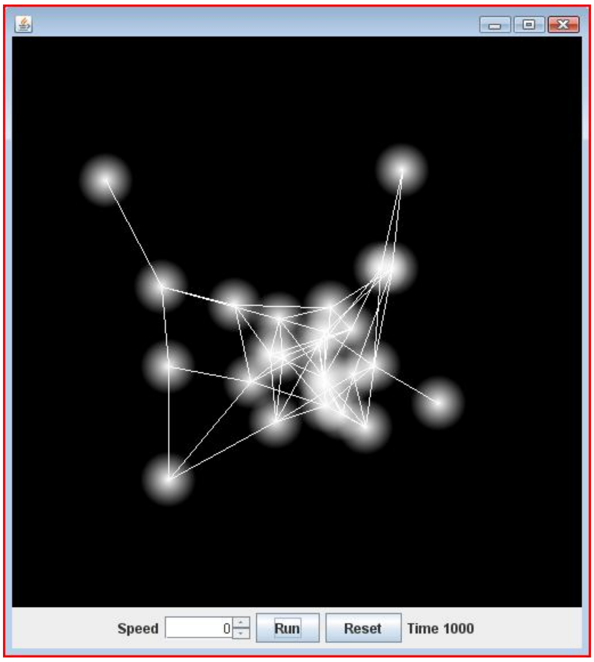

Figure 1: SOM layout of a random 25-node graph

2.2. SOM for Spherical Graph Layout

Unfortunately, SOM in a Euclidean geometry suffers from the border effect: nodes located toward the boundary of the space have fewer neighbors than those located near the center, and thus the weight vectors of the nodes tend to collapse toward the center of the input space [13]. Our two-dimensional self-organizing maps often utilized less than half of the available space (Figure 1). This problem was exacerbated by the fact that the algorithm does not enforce any vertex separation, causing vertices to be clustered in the center.

To avoid these problems, we first tested the conscience mechanism of [14], which adds a frequency bias B_n to the winner selection process based on winning frequency.

$$B_n = \gamma \left(\frac{1}{N} - F_n \right)$$

F_n is the winning frequency of node n, initialized for all nodes as $1/N$, and γ amplifies the frequency difference, initialised as a large factor. The overall effect is to prevent a few nodes from representing too much of the input data of random vectors, creating a more even distribution and thus reducing the number of iterations required. We found that the frequency bias further improved performance, but did not satisfactorily solve the border effect, because too high of a bias created an essentially uniform layout, which adversely affected the quality of our visualization.

We explored other workarounds by Kohonen [15] and others [16], but ultimately concluded that the fundamental problem is that Euclidean planar or three-dimensional graph layout is unsuitable for knowledge graphs: finite Euclidean geometry gives an excessive and mistaken emphasis on central nodes versus outlying nodes. Instead, we chose to map the knowledge graph onto the surface of a sphere. For display on a screen, we provide a perspective projection, which the user can rotate to view all the nodes. Spherical layout provides further benefits as well:

- Higher information density
- A natural fisheye view, allowing users to examine their selection in detail while maintaining global context
- Greater interactivity and engagement
- Compatibility with a future geographic mode of exploration
- Provides the sense that knowledge is never-ending

Implementing this change in topology required a few changes. First, the vectors must be three-dimensional. Second, we updated our procedure for generating random vectors, because generating random vectors based on uniform distributions $\theta \in [0,2\pi), \varphi \in [0,\pi]$ results in a non-uniform distribution. Third, we changed the distance metric from the Euclidean norm to the geodesic arc, and normalized the updated vectors to the sphere. Finally, we implemented a matrix rotation and projection for the user interface.

Algorithm 3: Desic algorithm for graph layout

input: Graph *G*
output: layout of *G* on the sphere

$r \coloneqq$ diameter(G)
iterations := $[1000/r]$
$\lambda \coloneqq 10000$

while $r > 0$:
    generate a random vector $\vec{v}$
    find node n with $min(\|\vec{v} - \vec{n}\| - B_n)$
    for all nodes n_i with $dist(n, n_i) \geq r$:
        $\overrightarrow{n_i} = normalized[\overrightarrow{n_i} + h(dist)\alpha(\vec{v} - \overrightarrow{n_i})]$

    if(*iterations* = 0)
        decrement r
        iterations = $[1000/r]$
    else decrement iterations

3. Desic

We developed a framework named Desic (for the fact that edges are represented as geodesic arcs on the sphere) to explore graphs generated from real data.

We decided to build a web application to reach a larger audience of students and to reduce the friction of sharing ideas and journeys. In the past we have prototyped ideas using the Java and Flash platforms, but we ultimately chose to use solely Javascript and HTML5 canvas because of (1) increased accessibility from not relying on plugins, (2) more accessible code, and (3) the open specifications align with the ideal of open access to knowledge.

We found that the majority of browsers currently in use can draw up 10,000 nodes with an acceptable frame rate (at least 24 FPS) with efficient processing. Since individual nodes reach the limits of screen resolution beyond that, larger graphs will require a modification to the algorithm to support clustering. Ultimately, Desic was able to embed large knowledge graphs of thousands of nodes in real time. Source code is available from the author.

4. Knowledge Web

The Knowledge Web seeks to make learning exciting and accessible and counters the tendency in modern education toward specialized learning and thinking. The project generously provided data processed from James Burke’s *Connections* series, which explores the idiosyncratic relationships between technology and social change.

The *Connections* dataset validates many of the assumptions we made for Algorithm 3. The general structure was scale-free with an exponential degree distribution; the average vertex degree was 7 while Isaac Newton had 98 adjacencies. As we had previously assumed, such graphs would have a very small diameter. The diameter of the *Connections* dataset was 10, which is vastly below log (1718).

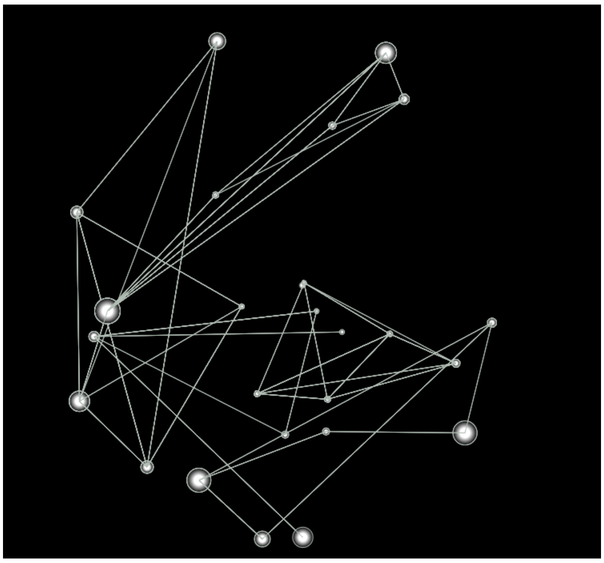

Figure 2. Desic visualization of a 5-component, 25-node graph. Note that Desic uses rotation to provide a better sense of depth than can be seen here

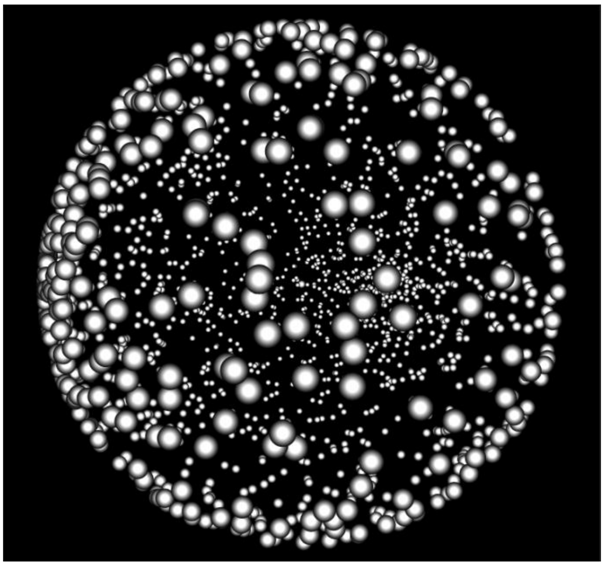

Figure 3. Desic visualization of the full 1718-node *Connections* graph, rendered in-browser (edges not shown)

5. Future Work

More experimental results are needed to test these results on different real-world knowledge graphs. We plan to generate visualizations on successively larger graphs. Our vision is to eventually allow users to see the entire article structure of knowledge graphs such as Wikipedia (4 million nodes) and Freebase (22 million nodes).

User experience may also be improved by taking advantage of semantic information, such as references to geographic locations, past dates, and different types of relationships between nodes. Alternative user interfaces including maps and timelines marked with nodes would complement our visualization very well and could even serve as filters to view nodes only in the relevant times or locations.

6. Conclusion

This paper discusses the potential of knowledge graphs. We have also developed an algorithm to visualize knowledge graphs, which has been implemented in the Desic framework. Through Desic, an effectively linear graph drawing algorithm, we have shown that exploration of entire graphs is possible.

7. Acknowledgements

We thank Patrick McKercher, James Zaun and James Burke from the Knowledge Web project (k-web.org) for providing valuable mentorship, particularly in arguing for the necessity of better, more accessible tools for exploring knowledge graphs.

8. References

[1] Xu, R. and Wunsch, D., “Survey of Clustering Algorithms.” *IEEE Transactions on Neural Networks*, 16(3) 645-678, 2005.

[2] Haemaelaeinen, M., Hari, R., Ilmoniemi, R. and J. Knuutila, J., “Magnetoencephalography - theory, instrumentation, and applications to noninvasive studies of the working human brain.” Reviews of Modern Physics, 65(2) 413, 1993.

[3] Berkhin, P. “Survey of clustering data mining techniques.” Technical report, Accrue Software, San Jose, CA, 2002.

[4] Mucientes, M., Iglesias, R., Regueiro, C.V., et. al. “A fuzzy temporal rule-based velocity controller for mobile robotics,” *Fuzzy Sets Syst*. 134, pp. 83–99, 2003.

[5] Chang, R., VLSI “Circuit Placement with Rectilinear Modules using Three-Layer Force-Directed Self-Organizing Maps.” *IEEE Transactions on Neural Networks*, 8(5) 1049-1064, 1997.

[6] Bonabeau, E. and Henaux, F., Self-organizing maps for drawing large graphs. *Information Processing Letters*, 67(4):177–184, 1998.

[7] Meyer, B., “Self-organizing graphs-a neural network perspective of graph layout.” *Proceedings of the 6th International Symposium on Graph Drawing*, pp. 246–262, 1998.

[8] Pastor-Satorras, R. and Vespignani, A., “Epidemic Spreading in Scale-Free Networks.” *Phys. Rev. E*, 86(14): 3200-3203, 2001.

[9] Goh, K.I., Kahng, B. and Kim, D., “Universal Behavior of Load Distribution in Scale-Free Networks.” *Phys. Rev. E*, 87(27): 278701, 2001.

[10] Barabasi, A.-L., Albert, R. and Jeong, H., “Scale-free characteristics of random networks: The topology of the World Wide Web,” *Physica A*, 281, 69–77. 2000.

[11] Cohen, R. and Havlin, S., “Scale-Free Networks Are Ultrasmall.” *Phys. Review. E*, 90(5): 058701, 2003.

[12] Bollobas, B. and Riordan, O., “The diameter of a scale-free random graph.” Preprint, Department of Mathematical Sciences, University of Memphis, 2002.

[13] Kohonen, T., *Self-Organizing Maps*. New York: Springer-Verlag, 2001.

[14] Desieno, D., “Adding a conscience to competitive learning.” *Proc. ICNN'88, Int. Conf. on Neural Networks*, pp. 117-124, 1988.

[15] Kohonen, T. “Things You Haven’t Heard about the Self-Organizing Map.” *Proc. ICNN'93, Int. Conf. on Neural Networks*, pp. 1147-1156, 1993.

[16] Wand, M.P. and Jones M.C., *Kernel Smoothing*. London: Chapman and Hall, 1994.