CS4234: Optimization Algorithms

MiniProject Ideas

Mini-Project 1: Social Networks

Social networks such as Facebook have become immensely popular with users—but they are also immensely popular with researchers, providing a fascinating dataset that yields insights on how people interact with each other. In this mini-project, you will examine some properties of social networks.

We can view a social network as simply a graph, where nodes represent people, and edges represent links between people. For this mini-project, your goal is to take the largest dataset you can, and try to derive interesting conclusions about the graph.

To get some ideas about what types of questions you might ask (and what types of algorithms you might try), you may find some of the sections of the book "Networks, Crowds, Markets" by Easley and Kleinberg to be of use. You can find it at: `http://www.cs.cornell.edu/home/kleinber/networks-book/`.

For data, the following site contains many interesting data sets: `http://snap.stanford.edu/data/index.html`. You can choose a data set from this site that focuses on social networks.

The following are some questions you might answer in your project.

- What sort of graph is a social network graph? Does a social network graph look more like a random graph (e.g., the graph $G(n, p)$ that connects every two nodes with probability p), or more like a preferential attachment graph? Or is there another better model for a social network graph?

- Recall how we used gradient descent to determine parameters to best fit our model of e-mail spam to real data. Could you use a similar technique to fit a model for a social network graph? (This might be tricky.)

- To answer those questions, first you need to decide what aspcects of the social network graph are you trying to model? The goal of a model is to find a synthetic graph (i.e., one you can can construct easily) that looks a lot like many different social networks in some way. But does it mean for a graph to "look like" a social network graph?

  You might look at the average node degree, or the distribution of node degrees, and see which model most looks like a social network. Or you might look at the number of triangles in a graph. Or you might look at the diameter of the graph. Or you might look at how well connected the graph is (e.g., the expansion or conductance of the graph, or some estimate based on how long it takes a random walk to go from a node u to node v).

- Thus, you might try to implement efficient algorithms for counting triangles or finding the diameter of very large graphs. Can you find the approximate diameter of a graph more quickly than the real diameter? (See, for example: `http://arxiv.org/pdf/1207.3622v1.pdf`.) What about approximate triangle counting?

- How many connected components are there in a social network graph? Or is the graph connected? How well connected is the graph? If I choose two random people, what is the min-cut between them? What is the average distance between them? How many people would I have to remove to shatter the graph into a large number of distinct components?

- What if I want to attack a social network by hacking a few accounts. How do I choose the best accounts to hack if I want to influence as many people as possible? What does it mean to influence people? Do I want to maximize the number of people who have a friend who is hacked? Do I want to maximize the number of people who are within two hops of a hacked friend? Do I want to maximize the number of pairs of people who have a hacked friend on the shortest path connecting them? Think of an interesting notion of "influence" and devise the best algorithm you can to hack the network.

In your mini-project, you do not have to answer all of these questions. Try to choose a few interesting questions and focus on finding good answers.