Communications Network Design

A/Prof. Matthew Roughan

March 4, 2009

lecture01 (24 pages)

- **Introduction**

  What, When, Where, How, Who, Why

- **A Brief History of Networking**

  > ”Those who do not study history are doomed to repeat it.”
  >
  > Georges Santayana

lecture02 (38 pages)

- **Computer Networks**

  Computer networks are a recent invention (in human history), but they have been around for longer than some of you may think. In this lecture we consider the underlying drivers in computer networks, and how this subject fits with the ongoing development of those networks.

lecture03 (45 pages)

- **Internet Design Principles**

  Most of this course is about mathematical optimization of networks. However, the term design covers a lot more than just deciding the capacity of links, etc. We need to know some of these more general principles

  - so you understand basic terminology
  - so you can appreciate what parts of a network are under your control and what forms “constraints”

lecture04 (39 pages)

- **Network Optimization: Goals and Constraints**

  What are the typical optimization goals (e.g., cost, performance, reliability) for network operators? Where are the costs in networks? What are the constraints (technological, and non-tech.) they operate under?

  Example 1: Single link problem

lecture05 (11 pages)

- **Optimization: 1000 foot view**

  Its helpful for us to talk a little about optimization techniques before we start. We also presenta little notation.

  Example 1: Three node network

lecture06 (43 pages)

- **Routing**

  A common approach to routing uses shortest-paths. The canonical algorithm for solving shortest-path routing is Dijkstra's.

  Example 1: Dijkstra

lecture07 (44 pages)

- **Routing (continued)**

  We continue the algorithmic viewpoint by considering an alternative to Dijkstra called the Floyd-Warshall algorithm. Also we consider routing implementation: OSPF, IS-IS, and some miscellaneous issues such as load balancing. Finally we will look into the distributed Bellman-Ford dynamic programming algorithm as implemented in RIP.

  Example 1: Floyd-Warshall

  Example 2: Distance vector: Bellman-Ford

lecture08 (51 pages)

- **Routing (continued)**

  The simple routing considered so far has fixed distances, but if we consider a more queueing view of networks, then packets are delayed when a link is heavily loaded, and so this increases delays. Minimum delay routing forms a non-linear, **convex** optimization problem with **separable** costs. We present two simple gradient descent methods for solution of such problems including the Frank Wolfe method.

  Example 1: cost and load calculation

  Example 2: linear, separable costs

  Example 3: convex functions

  Example 4: queueing model

  Example 5: descent method 1

- **Traffic Engineering**

  Modern IGP routing protocols are almost all based on simple SPF algorithms with linear costs, but real costs are non-linear. It works fine most of the time, but when congestion occurs, there is a problem. Traffic engineering is the process of rebalancing traffic loads on a network to avoid congestion.

lecture09 (24 pages)

- **The Network Design Problem**

  In this lecture we consider a new optimization problem, the network design problem, where we can choose the network links (in contrast to routing where we only chose the routes across a given network). In this lecture we present some basics such as **star-like** topologies, **ring** topologies and the **travelling salesman's problem**.

  Example 1: topologies

  Example 2: travelling salesman's problem

  Example 3: minimal cost star

lecture10 (24 pages)

- **Concave costs**

  When costs are concave, the network design problem has properties like single path routing. A common example is linear costs. Also we present a simple heursitic approach.

  Example 1: single path routing

  Example 2: linear costs example

lecture11 (30 pages)

- **Multicommodity flow problems**

  In this section we consider a special case of the network design with linear separable costs, but note that this is still NP-hard, so we need a heursitic solution. The first we try is Minoux's greedy method.

  Example 1: Minoux's Method

  Example 2: Minoux's Method

lecture12 (38 pages)

- **Budget constraint model and branch and bound**

  Branch and bound is a standard technique for solving integer programs, by relaxing the problem to the non-integer problem to find bounds, and using these to prune a tree of the possible solutions (rather than evaluating all possible solutions).

  Example 1: branching example

  Example 2: branch and bound: simple program

lecture13 (48 pages)

- **Branch and bound (cont)**

  The simple branch and bound solution shown previously is rather naive. It doesn't take advantage of the structure of the problem. We show how branch and bound can be applied to the budget constraint model, by showing the relationship with the **knapsack problem**. The useful result we get is the **Dionne-Florian** lower bound, which can be used in bounding.

  Example 1: branch and bound on budget constrained network design problem

lecture14 (31 pages)

- **Randomized algorithms: simulated annealing**

  It is often the case that we optimize against a non-convex objective function. In these cases we often use heuristics such as gradient descent, but they can become stuck in a local minimum. **Simulated annealing** allows our search to "bounce" out of such a point, by including some randomization in its search. We present here the **Metropolis** algorithm for simulated annealing.

  Example 1: one step of simulated annealing

  Example 2: travelling salesman's problem

lecture15 (33 pages)

- **Randomized algorithms: genetic algorithms**

  Genetic algorithms (sometimes called evolutionary computing) work by analogy to Darwin's theory of evolution. We generate populations of solutions, and allow them to "evolve" towards fit solutions (solutions that minimize our objective). GAs have advantages in flexibility: they can even be applied when the objective function isn't known.

  Example 1: genetic algorithm: travelling salesman's problem

lecture16 (41 pages)

- **Tree-like networks**

  Tree-like networks, and algorithms for their design: minimum spanning tree problem, spanning trees and spanning tree protocol, greedy methods (Kruskal's and Prim's methods).

  Example 1: spanning tree

  Example 2: Kruskal

  Example 3: Prim

lecture17 (47 pages)

- **Advanced tree-like network design**

  Tree-like networks, and some more advanced algorithms. Starting with **cutsets** we get **Gomory-Hu** and **Gusfield's** methods.

  Example 1: Gomory-Hu Algorithm

  Example 2: Gusfield's Algorithm

lecture18 (21 pages)

- **Tree-like networks implementations**

  We look into one example where tree-like network design is important: the design of Ethernet LANs. This leads onto consideration of the Internet as a larger "Network of networks".

lecture19 (32 pages)

- **Networks of networks**

  The Internet is a network of networks. Most of the problems we have considered up to this point concern a single network. There are many interesting problems when we consider how these networks interconnect.

lecture20 (29 pages)

- **BGP**

  BGP (the Border Gateway Protocol) version 4 is the defacto inter-domain routing protocol.

lecture21 (47 pages)

- **Input data**

  One of the key problems in network design is how we get the input data for network design. In particular traffic matrices are one of the key inputs, but are not always easy to measure.

lecture22 (19 pages)

- **Network Design without complete information**

  What can we do where the critical input data (e.g. the traffic matrix) is missing or incomplete? The answer is to optimize with respect to all possible traffic matrices. There are several possible algorithms: oblivious routing and Valiant network design.

lecture23 (4 pages)

- **Revision**