Overview

Complete the following problems from the textbook. For every *programming exercise* (indicated with an asterisk), do it by writing a short program that calculates the answer for you, and turn in both your code and your answer. I recommend that you use a programming environment that does the matrix multiplication for you. `numpy` within Python or the language R are both great choices.

For every non-programming exercise, you should turn in a carefully written solution showing your work and explaining your answer. Specifically, I very highly recommend using LaTeX, which is the standard tool in theoretical computer science for communicating technical material, and it is valuable to get the practice with it. If you have no experience with LaTeX at all, Getting Something out of LaTeX is a fine place to start. LaTeX Tutorials — A Primer is a more detailed tutorial. Text Processing using LaTeX is a great reference for the language. I choose to prepare my LaTeX documents using the Emacs text editor, but you may choose another text editor or environment if you prefer.

Other word processors do have equation editors that you can use if you prefer that approach. Do *not* submit a raw text submission where you "fake" the equations by using normal ASCII characters.

If you have diagrams that you would like to submit, hand-drawn supplements are acceptable (but they too must be submitted electronically). You may find it easiest to print your solution, draw in your diagrams, scan the printout, and submit the resulting PDF. The better approach, of course, is to generate the diagrams electronically. The native LaTeXsoltion TikZ produces lovely pictures, though it has something of a learning curve associated with it. There's also Graphviz, which is a specialized tool for generating pretty graphs. Finally, you can also use any standard GUI drawing tool such as Dia, and save what you create as a PDF. All of the above are installed on the department systems.

Collaboration policy: You may collaborate on the homework assignments to the extent of formulating ideas as a group, but you may not collaborate in the actual writing of solutions. *In particular, you may not work from notes taken during collaborative sessions.* You *must* cite all sources, including others in the class from whom you obtained ideas. You may not consult any materials from any previous offerings of this course or from any other similar course offered elsewhere. You are required to completely understand any solution that you submit, and, in case of any doubt, you must be prepared to orally explain your solution to me. *If you have submitted a solution that you cannot verbally explain to me, then you have violated this policy.*

Part 1

- Exercise 5.1.1*
- Exercise 5.1.2*
- Exercise 5.1.3
- Exercise 5.1.6
- Exercise 5.1.7

Part 2

- Exercise 5.3.1*
- Exercise 5.4.1(c)
- Exercise 5.4.3. This is a great open-ended problem, and I'd encourage you to think about it in a variety of ways, but to make grading more feasible, I'm going to specify this problem more carefully. Consider having two spam farms, each of which resemble Figure 5.16 in the text. Each spam farm has its own target page, its own set of accessible pages, and its own set of "own" pages. Take the picture in 5.16 and duplicate it, effectively, for the second spam farm. The number of accessible pages and "own" pages may be different for the second farm than for the first. Connect the two farms together in this one way only: Have the two farms pool their "own" pages. In other words, each target page links to all of the "own" pages in both farms, and all "own pages" in both farms link back to both target pages. in each farm link not only to its own target page, but also to the target page in the other farm. The two target pages do *not* link to each other. Repeat the analysis from section 5.4.2 for this spam farm, and interpret the results.
- Exercise 5.5.1*