MATH 408 D100 ASSIGNMENT #1

Due: Friday, September 21st (in class)

Reading

Chapter 23 of Vanderbei's *Linear Programming: Foundations and Extensions (4th edition)*. SFU Library link.

From the textbook, Chapter 1 and Sections 2.1 and 2.2. Sections 1.3 briefly introduces computational complexity, while Section 1.5 explores connection between discrete optimization and number theory. If you are not familiar with those subjects, then you may find these sections quite challenging.

Chapter 1 of Applegate, Bixby, Chvátal and Cook's *The Traveling Salesman Problem : A Computational Study*. SFU library link.

Please make sure you understand the translation of combinatorial optimization problems into integer programs, for instance knapsack, set covering and travelling salesman. Try to convince yourself that it is straightforward to model most natural constraints using mixed integer programs. LP is already powerful, but doesn't easily capture integer variables or constraints of the form "A or B", "satisfying k of n constraints" or even "takes a value from a specified discrete set".

Problems for Math 408 and Math 708

0. (Not graded.) Download and install `AMPL` following the instructions given in class. (A limited version is available at `http://www.ampl.com/try-ampl/download-a-free-demo/`, but it may not be able to handle larger problems later in the course.) You can use either the command-line or graphical (Integrated Development Environment) version. Work through the diet examples in Chapter 2 of the `AMPL` book: `http://ampl.com/resources/the-ampl-book/chapter-downloads/`, so that you are familiar with the software.

1. Take your nine digit student id number and add 10 to each digit to get a sequence of nine numbers $a_1, a_2, \dots, a_9$ between 10 and 19. Take $b_1, b_2, \dots, b_9$ to be the first nine digits of π. Your personal knapsack problem is:

$$\text{Maximize} \sum_{i=1}^{9} b_i x_i \quad \text{subject to} \sum_{i=1}^{9} a_i x_i \leq \frac{1}{2}\sum_{i=1}^{9} a_i \quad \text{and } x_i \in \{0,1\} \text{ for } i = 1, \dots, 9.$$

Solve this integer program using `AMPL`, either via the command line interface or the `AMPL IDE` graphical interface. Use the `Gurobi` solver.

Please submit your answer to problem 1 directly to the teaching assistant by e-mail (`xla97` at `sfu` dot `ca`). All file names should begin: `math_408_1187_name_hw1_q1` where `name` is your family name. Submit the relevant .dat, .mod and a .pdf file showing your output in a single e-mail. If you would like to submit additional questions via e-mail, you may do so, but only if they are *typeset*. (Do **not** include documents that are produced by a scanner.) If you wish to do this, include your remaining answers either in a single .pdf file named: `math_408_1187_name_hw1_qall`, or with one .pdf file per question names `math_408_1187_name_hw1_q2` (or `q3`, `q4`, `q5`, `q6`, `q7`, `q8` as appropriate). Here again, `name` is substituted with your own family name.

E-mail submissions are due 10 minutes before class, at 9:20 a.m.

2. Textbook Exercise 1.1.

3. Consider a situation where your model includes binary variables $x_1, x_2, \dots, x_9$, each representing the potential purchase of a particular investment. Show how to represent each of the following constraints as a single linear constraint:

   a. You cannot choose all of them.

   b. You must choose at least two of them.

   c. If you choose investment 6, you must also choose investment 4.

d. If you choose investment 3, you may not choose investment 8.

e. You must choose either both investments 7 and 9, or neither of them.

f. You must choose either one of the first 3 investments, or two of the last 3 investments.

4. Textbook Exercise 1.13.

5. Textbook Exercise 1.15.

Additional Problems for Math 708

6. Suppose that you are building a model which contains variables x_1 and x_2, and you have that $0 \leq x_1, x_2 \leq C$. Show how you can use an additional binary variables to represent the functions $u = \max\{x_1, x_2\}$ and $v = |x_1 - x_2|$.

7. Textbook Exercise 1.19.

8. Consider the problem of allocating storage (memory) dynamically in a computer. Model the memory as a simple array indexed by the positive integers. Suppose we are given a series of n requests to use an array of size s_i from arrival time r_i to departure time d_i. We would like to find the minimum memory size that will accommodate these requests (and a way to do it). Formulate this problem as an integer program.