CO327 Deterministic OR Models (2021-Spring)

How to solve LP, IP and QP: use computer!

Who will solve it by hand????

Instructor: Andersen Ang

Combinatorics and Optimization, U.Waterloo, Canada

msxang@uwaterloo.ca, where $\mathbf{x} = \lfloor \pi \rfloor$

Homepage: angms.science

First draft: Janurary 2, 2021 Last update: June 12, 2021

LP, LIP and QP

- LP in standard form

$$\begin{aligned} \max_{\mathbf{x}} \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned}$$

- IP in Knapscak standard form

$$\begin{aligned} \max_{\mathbf{x}} \quad & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \\ & \mathbf{x} \in \mathbb{N}^m \end{aligned}$$

- QP with linear constraints

$$\begin{aligned} \max_{\mathbf{x}} \quad & \frac{1}{2}\mathbf{x}^\top \mathbf{Q}\mathbf{x} - \mathbf{p}^\top \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned}$$

Methods

- You already know
  - Graphical method
  - Simplex method
  - LP relaxation
  - Brute-force / Complete enumeration
  - Cutting plane
  - Branch-and-Bound

- Advanced methods for LP
  - Ellipsoidal method
  - Interior point method

- Advanced method for LIP
  - Simulated Annealing
  - Genetic Algorithm
  - Ant colony and particle swarm

- Advanced methods for QP
  - Gradient descent
  - Coordinate descent
  - Newton's iteration

- NOT IN THIS COURSE!

This course: modeling not algorithm

1. This course is named Deterministic OR Models
   - Not "Linear programming"
   - Not "Integer programming"

   We focus on modelling, not the methodology on how to solve them.

2. You have enough knowledge on how to solve them from the prerequisites.

3. In reality, no one solve optimization problems by hands, uses computer!
   - In this course you are allowed to solve all problems by computer.
   - There will be computer assignments.

But still, let's "review" these methods in a high level point of view.

Graphical method

- Easy but only for toy problems.
- What is it
  - draw the feasible region
  - locate the extreme points
  - test the objective value at these points.
- The mathematics: if there is a sol., then

  Sol. of LP is often located at the corner point / edge.

  Why : to be discussed in the theory lecture.

Simplex method

- The first LP solver that works in practice.
  - Average runtime complexity: polynomial time
  - Worst case complexity: $\mathcal{O}(2^n)$, non-polynomial time, $n = \#$ corners.

- The mathematics: if there is a sol., then

  Move along the corner edge will ultimately reach the sol.

- Primal and dual Simplex
  - Primal simplex method = simplex method on the primal variable.
  - Dual simplex method = simplex method on the dual variable.
  - Use which: depends on m, n.

Relaxation

- The idea
  - Relax the original problem by ignoring some constraints.
  - Solve the (simpler) relaxed problem
  - Use the sol. of the relaxed problem for the original problem.
- LP relaxation of IP: relax, solve, round.
- Relaxation gap:
  - If IP is a min problem, the optimal objective value for LP relaxation $\leq$ that of IP.
  - If IP is a max, the optimal objective value for LP relaxation $\geq$ that of IP.
- If LP relaxation is infeasible, then so is LIP.
- Integrality gap and when will relaxation work
  - Not work: if the integrality gap is large
  - Work: if the convex hull of the integer polyhedron = the feasible region of the LP, or no integrality gap.
  - TU (Total Unimodular) matrix
  - More on theory section of LIP.

Brute-force / Complete enumeration

- Idea: try all possible sol.
- Let n = number of binary variables.
  Number of possible sol. $= 2^n$.
- Suppose that we could evaluate 1 billion (1,000,000,000, 1 Giga) solutions per second.
- Time for computing the sol.:
  - $n = 30$: 1 second.
  - $n = 50$: 11.6 days.
  - $n = 70$: 31000 years.
- Fact: only a (tiny) fraction of the solutions actually need to be examined.
  - Branch-and-bound: eliminate subtree.
  - Cutting plane: cut away the redundant solution space.

Branch-and-bound method

- Idea: search the enumeration tree, cut enumeration when possible.
- What: Bounding, branching, and fathoming.
- When to cut subtree (fathom it)
  - The sol. is integer (there is no need to go further) or
  - There is no feasible sol. or
  - The best sol. in the subtree cannot be as good as the best available sol. (the incumbent).
- Runtime: still exponential (Non-polynomial). But okay if problem size not too large.

Cutting plane method

- Idea: cut off parts of the feasible region of the LP relaxation, so that the optimal integer solution becomes an extreme point and therefore can be found by the simplex method.
- What
  1. Solve the LP relaxation. Get x^*
  2. If x^* integral stop, else find a valid inequality that will exclude x^*
  3. Go to step 1.

Solvers: Computer programming

IF this is a course on linear programming that focus heavily on the theory:

- We will study deeply the simplex method, ellipsoidal method, interior point method
- Implement these method
- Test the implementation
- Compare with commercially available solvers (e.g. Gurobi)

But we will not do these: we will **just call the computer to solve it!!**
So what's more important: **learn the tool – know how to code to use computer**

"Write your own code" VS "just call solvers"

	Write your own code	just call solvers
Effort	Tedious, high effort	Easy, simple
Problem specific?	Specific to your problem	General purpose solver
Speed	Can be fastest in the world	Can be slow
Who will do this	Mostly academic	Mostly industry

When to write your own code:

- If general purpose solver is not good enough to solve your problem
  - Not fast enough / too slow
  - Too expensive to run / computational cost too high
- Your problem **has specific structure**, and the general purpose solver doesn't fully utilize such information for its advantage
- An inconvenient truth: many algorithms developed are in fact **reinventing the wheel**.

Some famous solvers

- Commercial software (with formal maintenance: more reliable)
  - MATLAB: `intlinprog`, `linprog`, `quadprog`
  - CPLEX
  - AMPL
  - Gurobi
- Free software (no formal maintenance, can be error prone)
  - SciPy library of Python
  - GNU Octave
  - Scilab
  - JuMP in Julia
- We will stick with MATLAB (because UW has the license!)
- For a longer list of optimization software, see
  `https://en.wikipedia.org/wiki/List_of_optimization_software`

Now what: MATLAB time!