Operations Research II: Algorithms
Course Summary and Future Directions

Ling-Chieh Kung

Department of Information Management
National Taiwan University

Road map

- **Summary and discussions**.
- Preview of the next course.

Summary

- In this course, we have gone through several important algorithms for solving mathematical programs.
- Let's briefly recall our memory.

Week 2: the simplex method

- Topics:
  - The standard form.
  - Basic solutions and basic feasible solutions.
  - The method (with the tableau representation).
  - Unbounded and infeasible LPs.
- The publication of the simplex method by George Dantzig in 1947 opened the whole field of Operations Research.
  - Today many commercial solver still use (some advanced version of) the simplex method.
- It runs along edges.
  - There exists **interior-point methods**, e.g., the Karmarkar's method, that moves through the interior of the feasible region.

Week 3: the branch-and-bound algorithm

- Topics:
  - Linear relaxation.
  - Branch and bound.
  - Solving the knapsack problem.
- It iteratively splits the feasible region without removing an optimal solution.
- There exists more advanced methods, e.g., the cutting-plane method, branch and cut, branch and price, and many else.
- Compared to linear programming, solving an integer program is much more time-consuming.
  - Try your best not to set "quantities" to integers.
  - Use binary and integer variables only when it is regarding the **selection** among multiple options (e.g., whether to do one thing).

Week 4: gradient descent and Newton's method

- Topics:
  - Introduction.
  - Gradient descent.
  - Newton's method.
- Important foundations for more advanced method.
  - E.g., for methods designed to solve **constrained optimization** problem.
- Note that they are by nature interior-point methods.
  - Convergence and speed of convergence are critical issues.

Week 5: a case study

- A company utilizes integer programming to determine whether to close some existing facilities.
- A **heuristic algorithm** is developed with its performance examined.
- Practical consideration for execution is always needed.

Summary

- All these algorithms are powerful.
  - A mathematical program with any number of variables and constraints is guaranteed to be solved.
  - Time complexity is always an issue.
- Some researchers work to improve existing algorithms (or developing new ones) for **general problems**.
  - Once an algorithm is improved, all programs (in that class) benefit.
- Some researchers work to develop specific algorithms for **specific problems**.
  - By considering some special properties of a problem, we should be able to do better.

Road map

- Summary and discussions.
- **Preview of the next course**.

Algorithm design

- A good algorithm does **smart search**.
- The key of being smart is to do **analysis**.
  - To obtain some necessary or sufficient conditions for an optimal solution.
  - E.g., if a linear program has an optimal solution, it has an extreme-point optimal solution.
- Do we have more properties for linear programs?

Algorithm design

- How about integer programs?
  - Linear relaxation is typically the first step.
  - Is it possible for an optimal solution to the linear relaxation be feasible (and thus optimal) to the original integer program? How to ensure this?
- How about nonlinear programs?
  - Gradient descent keeps iterating until the first-order derivative is flat enough.
  - Is it possible for an reported solution (local optimum) to be different from an optimal solution (global optimum)? How to ensure this?

Theory

- To go further, we need **theory**.
- We need to have some ways to rigorously investigate the **theoretical properties** of a mathematical program.
- Keywords:
  - Linear programming duality, shadow prices, sensitivity analysis, dual simplex method.
  - Total unimodularity, network flow.
  - Convex set, convex function, first-order condition, positive semi-definiteness, Lagrangian relaxation, Lagrangian duality, KKT condition.
  - And more.
- We will see how theories are **useful** and **beautiful**.

That's all. See you in the next course!