CSC 373: Algorithm Design and Analysis
Lecture 19

Allan Borodin

March 4, 2013

Announcements and Outline

Announcements

- Term test today in tutorial rooms

Today's outline

- Integer Programming (IP) and Linear Programming (LP)
- LP relaxation of an IP and rounding to an integral solution.

How to read chapter 7 of text

- Chapter 7 is a mixture of linear programming and max flow concepts and the relation between these topics.
- The chapter nicely motivates LP and discusses the simplex method and duality.
- We will postpone and de-emphasize the discussion of solving LPs and duality.
- Instead we will mainly be considering IP/LP rounding

Expressing problems as IPs and LPs: motivating examples

- The text starts off with two specific examples, the first one being a minimization problem where a carpet company wishes to minimize its costs while meeting different monthly damands d_i for the carpets.
- The carpet example is an example of an IP.

For each month, non-negative integer variables are introduced to represent the number of workers employed w_i, the number of carpets made in regular time x_i and overtime o_i, the number of workers hired is h_i and fired f_i at the start of month i, and the number of carpets that are in storage s_i at the end of month i.

- This results in 72 variables, 6 for each month.
- There are labour costs (for regular and overtime), regulations as to limits of overtime, HR costs to hire and fire, and storage costs.
- This sounds like a rather daunting task for management to meet demands, satisfy all the constraints and minimize total annual cost.

Motivating example continued

- But it turns out that the problem can be expressed as the minimization of a linear function (in these integer variables) subject to a set of linear equations and linear inequalities.
- For example,
  - The number of carpets in storage at the end of month i is ensured by the constraint $s_i = s_{i-1} + x_i - d_i$.
  - The number of carpets that can be produced overtime is given by the constraint $o_i \leq 6w_i$ which follows from the limited overtime (30%) and the number (20) of carpets a worker can produce in regular time during a month.
- One can then try to solve this with an IP solver. But a more efficient approach is to let the variables be non-negative rational numbers and then use an LP solver.
- In this application rounding up the rational solution values would probably not significantly change the costs.
- In other applications there is no way (say assuming P $\neq$ NP) to round rational solution values without significantly impacting the objective function.

Representing the max flow problem as an LP

- We will now adopt the more traditional formulation of flows where all flows are non-negative.

- To represent this problem as an LP, we introduce non-negative rational variables f_e for each edge $e \in E$. The input is the flow network including the capacities c_e for each edge.

- The goal now is to maximize $\sum_{(s,u)\in E} f_{su}$
  subject to:
  1. $0 \leq f_e \leq c_e$ for all $e \in E$ (capacity constraint)
  2. $\sum_{(w,u)\in E} f_{wu} = \sum_{(u,z)\in E} f_{uz}$ for all $u \neq s, t$ (flow in $=$ flow out)

- In this case, the desired optimal flows $\{f_e\}$ are allowed to be rational.

IP and LP relaxations

- With these two motivating examples, we now begin to study one of the most widely used and successful algorithmic paradigm(s) for optimization: integer programming (IP) and linear programming (LP).
- We will be discussing the LP relaxation of IPs and rounding such LPs to obtain IP solutions.
- We start with some examples and then briefly discuss some LP theory.
- We will mainly be treating LP solvers as a black box.

Complexity status

- While there are problems which are directly represented by LPs (e.g. max flow), I will focus on NP-hard problems which are (in most cases) naturally represented by IPs.

- Indeed solving IPs is an NP-hard problem although there are many heuristics and special cases that are solvable in practice and sometimes in theory.

- LPs are efficiently solvable both in practice and theoretically.
  - There poly-time algorithms for solving them although these algorithms tend not to be as efficient as simpler combinatorial methods.
  - Not known if there is a strongly poly-time algorithm.
  - All analyzed implementations of the simplex method can take exponential time in worst case analysis but "in practice" simplex works well and with respect to "smoothed analysis", the method is polynomial time.

LPs in standard form (for a minimization problem)

The standard form for a minimization problem

$$
\begin{aligned}
\text{minimize} \quad & \sum_{j=1}^{d} c_j \cdot x_j \\
\text{subject to} \quad & \sum_{j=1}^{d} A_{ij} \cdot x_j \geq b_i \qquad && \text{for each } i = 1, \ldots, m \\
& x_j \geq 0 && \text{for each } j = 1, \ldots, d
\end{aligned}
$$

Using boldface to denote vectors, we can write this standard form more compactly as follows:

$$
\begin{aligned}
\text{minimize } & \mathbf{c} \cdot \mathbf{x} \\
\text{s.t. } & A\mathbf{x} \geq \mathbf{b} \\
& \mathbf{x} \geq 0
\end{aligned}
$$

LPs in standard form (for a maximization problem)

The standard form for a maximization problem

$$\begin{aligned}
\text{maximize} \quad & \sum_{j=1}^{d} c_j \cdot x_j \\
\text{subject to} \quad & \sum_{j=1}^{d} A_{ij} \cdot x_j \leq b_i && \text{for each } i = 1, \ldots, m \\
& x_i \geq 0 && \text{for each } i = 1, \ldots, d
\end{aligned}$$

We can write this standard form more compactly as follows:

$$\begin{aligned}
\text{maximize} \quad & \mathbf{c} \cdot \mathbf{x} \\
\text{s.t.} \quad & A\mathbf{x} \leq \mathbf{b} \\
& \mathbf{x} \geq 0
\end{aligned}$$

Notes

- The previous examples had equalities and not just inequalities but it is easy to see how an equality can be converted into two inequalities.
- Integer programs (IPs) have additional constraints $x_i \in \mathbb{Z}$.
- "Duality" uses standard form.
- Minimization problems (in standard form) for which all a_{ij} and b_i are non negative are called covering problems.
- Maximization problems (in standard form) for which all a_{ij} and b_i are non negative are called packing problems.
- There is another basic form called slack form where slack variables are used to turn all inequalities into equalities and that is a convenient form for the Simplex method that is often used to solve LPs.

Overview of the IP formulation for vertex cover and its LP relaxation

- We discuss weighted vertex cover as a $\{0, 1\}$ IP.
- We will discuss its LP relaxation and "naïve" and why this provides a 2-approximation.
- What is the integrality gap of this IP/LP relaxation?
- Adding some additional inequalities (say corresponding to odd cycles) does not help.
- In general there will be many IP formulations for a given problem. An integrality gap pertains to one (or a class of) IP/LP relaxations.

An IP for weighted vertex cover

IP formula for weighted vertex cover

$$
\begin{aligned}
\text{minimize} \quad & \sum_{j=1}^{n} w_j \cdot x_j & \\
\text{s.t.} \quad & x_i + x_j \geq 1 & \text{for each edge } (i,j) \in E \\
& x_i \in \{0,1\} & \text{for each } i = 1, \ldots, n
\end{aligned}
$$

- The intended meaning is that $x_i = 1$ iff vertex v_i is in the cover.
- The LP relaxation is to relax the integrality condition $x_i \in \{0,1\}$ to $x_i \geq 0$.
- In this problem it follows that an optimal LP solution also satisfies $x_i \leq 1$.

Rounding an LP optimal solution

- Suppose x^* is an LP optimum.
- We can apply a "naïve" rounding (naive in the sense that the rounding ignores the input) to the fractional solution by setting
$$x_i' = 1 \text{ iff } x_i^* \geq 1/2$$
- Claim: x' is an integral solution to the IP and hence $V' = \{v_i \mid x_i' = 1\}$ is a vertex cover. Why?

Claim

The weight of the cover V' is at most twice the weight of an optimal cover.

Proof.

- Because the LP is a relaxation, it also allows IP solutions. Thus, $OPT_{LP} \leq OPT_{IP}$.
- Then we have $w(V') \leq 2 \cdot OPT_{LP} \leq 2 \cdot OPT_{IP}$.