

Optimization ► Linear Programming

Lecture 11

- The Approximation Framework
- Linear Programs
- Solving Linear Programs
- Relaxations and Roundings





Optimization ► Linear Programming

Lecture 11

- The Approximation Framework
- Linear Programs
- Solving Linear Programs
- Relaxations and Roundings

Recap: Approximation Algorithms

Our Goal: Find approximate solutions to hard optimization problems

Optimization Problems:

- Minimize cost measure / maximize quality measure
- Over a space of candidate solutions

Approximation Ratio: Algorithm is an α -approximation if:

- **Minimization:** $\max_I \frac{A(I)}{\text{OPT}(I)} \leq \alpha$
- **Maximization:** $\min_I \frac{A(I)}{\text{OPT}(I)} \geq 1/\alpha$

Key Idea: **Certificates** for lower/upper bounds on OPT

- Help us prove approximation guarantees
- **Today:** Linear programming as a tool for certificates



Optimization ► Linear Programming

Lecture 11

- The Approximation Framework
- Linear Programs
- Solving Linear Programs
- Relaxations and Roundings

Recall/Review: Maximum Flow

Input: Directed graph $G = (V, E)$, source s , sink t , edge capacities $c : E \rightarrow \mathbb{R}_{\geq 0}$

Goal: Find maximum flow from s to t respecting capacities

Recall: A **flow** $f : E \rightarrow \mathbb{R}_{\geq 0}$ is valid if:

- **Capacity:** For all edges $e \in E$: $0 \leq f(e) \leq c(e)$
- **Conservation:** For all vertices $v \in V \setminus \{s, t\}$: Flow in = Flow out

Value: $\text{val}(f) = \sum_{e \text{ out of } s} f(e) - \sum_{e \text{ into } s} f(e)$

Maximum Flow as an Optimization Problem

Variables: Associate a variable f_e to each edge $e \in E$

Objective: Maximize $\sum_{e=(s,v) \in E} f_e - \sum_{e=(u,s) \in E} f_e$

Constraints:

- For all $e \in E$: $f_e \leq c_e$ (capacity)
- For all $e \in E$: $f_e \geq 0$ (non-negativity)
- For all $v \in V \setminus \{s, t\}$: $\sum_{e=(v,u) \in E} f_e = \sum_{e=(u,v) \in E} f_e$ (conservation)

All constraints are linear (in)equalities! This is a **linear program**.

A General Linear Program

Input:

- n real-valued variables $x_1, \dots, x_n$
- A linear objective function to maximize or minimize
- m linear constraints (inequalities or equalities)

Output: Values for $x_1, \dots, x_n$ that:

- Satisfy all constraints
- Maximize/minimize the objective function

Important: Only $\leq$, $\geq$, and $=$ allowed (no strict inequalities $<$ or $>$)

General LP: Formal Definition

Maximize (or Minimize): $c_1x_1 + c_2x_2 + \cdots + c_nx_n$

Subject to:

$$a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,n}x_n \leq b_1$$

$$a_{2,1}x_1 + a_{2,2}x_2 + \cdots + a_{2,n}x_n \leq b_2$$

$$\vdots$$

$$a_{m,1}x_1 + a_{m,2}x_2 + \cdots + a_{m,n}x_n \leq b_m$$

- All $a_{i,j}$, b_i , c_j are real constants
- Can also use $\geq$ or $=$ constraints
- Can add non-negativity: $x_i \geq 0$ for some/all variables

Example 1: Nutrition Planning

Problem: Design a meal plan to minimize cost while meeting nutritional goals

Available foods:

Food	Cost (per serving)	Protein (g/serving)	Sugar (g/serving)
Tofu	\$2.50	20g	1g
Rice	\$0.50	4g	1g
Beans	\$1.00	15g	3g

Requirements:

- At least 50g of protein
- At most 10g of sugar

Classic application: diet planning, resource allocation, production planning

Example 1: LP Formulation and Solution

Variables: x_1 = servings of tofu, x_2 = servings of rice, x_3 = servings of beans

Minimize: $2.5x_1 + 0.5x_2 + x_3$ (total cost in dollars)

Subject to:

$$20x_1 + 4x_2 + 15x_3 \geq 50 \quad (\text{protein requirement})$$

$$1x_1 + 1x_2 + 3x_3 \leq 10 \quad (\text{sugar limit})$$

$$x_1, x_2, x_3 \geq 0 \quad (\text{non-negativity})$$

Optimal solution: $x_1^* = 0$, $x_2^* = 2.5$, $x_3^* = 2.5$ with cost \$3.75
(No tofu, 2.5 servings each of rice and beans)

Example 2: Linear Regression

Input: Points $(x_1, y_1), \dots, (x_n, y_n)$ in the plane

Goal: Find a line $y = ax + b$ that minimizes the **L1 distance** to the points:

$$\sum_{i=1}^n |y_i - (a \cdot x_i + b)|$$

Remark

In a Machine Learning class you might have seen a similar problem with L2 distance (i.e. least squares):

$$\sum_i (y_i - (ax_i + b))^2$$

Example 2: LP Formulation

Variables: a, b (line parameters)

Minimize: $\sum_{i=1}^n |y_i - (a \cdot x_i + b)|$

Subject to: No constraints!

What is wrong with this formulation?

Example 2: Modeling Absolute Value

Key Idea: For each point i , introduce variable z_i to represent $|y_i - (ax_i + b)|$

We need: $z_i \geq |y_i - (ax_i + b)|$

This is equivalent to:

$$\begin{aligned} z_i &\geq y_i - (ax_i + b) \\ z_i &\geq -(y_i - (ax_i + b)) = ax_i + b - y_i \end{aligned}$$

When we minimize $\sum_i z_i$, each z_i will equal $|y_i - (ax_i + b)|$ (not larger)

Example 2: LP Formulation

Variables: a, b (line parameters) and $z_1, \dots, z_n$ (helper variables)

Minimize: $z_1 + z_2 + \dots + z_n$

Subject to: For all $i = 1, \dots, n$:

$$z_i \geq y_i - ax_i - b$$

$$z_i \geq -y_i + ax_i + b$$

This shows how to model certain non-linear functions using LPs by introducing helper variables!

Example 3: Min Cut

Input: Directed graph $G = (V, E)$, source s , sink t

Goal: Find minimum capacity cut separating s from t

- A **cut** (S, T) partitions V into two sets with $s \in S$, $t \in T$
- **Capacity** = sum of capacities of edges crossing from S to T

Example 3: Min Cut as an Integer Program

Variables: $x_v \in \{0, 1\} \quad \forall v \in V$
 $y_e \in \{0, 1\} \quad \forall e \in E$

Minimize: $\sum_{e \in E} c_e \cdot y_e$

Subject to: $x_s = 0, \quad x_t = 1$
 $x_v \leq x_u + y_{uv} \quad \forall (u, v) \in E$

Interpretation:

- $x_v = 0$ if $v \in S$
- $x_v = 1$ if $v \in T$
- $y_e = 1$ if edge crosses from S to T
- $y_e = 0$ otherwise

If $x_u = 0$ and $x_v = 1$, then $y_{uv} \geq 1$, so $y_{uv} = 1$ (edge is cut)

Objective: minimize total capacity of cut edges

This program needs integrality constraints to work correctly, making it an Integer Program (IP)!

Solving general IPs is NP-hard (it's not hard to see that SAT can be modeled as an IP).

Example 3: LP Relaxation

Idea: Relax integrality constraints to $[0, 1]$ instead of $\{0, 1\}$

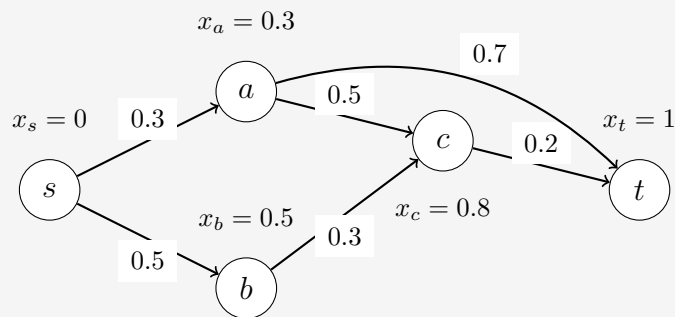
Variables: $x_v \in [0, 1] \quad \forall v \in V$
 $y_e \in [0, 1] \quad \forall e \in E$

Minimize: $\sum_{e \in E} c_e \cdot y_e$

Subject to: $x_s = 0, \quad x_t = 1$
 $x_v \leq x_u + y_{uv} \quad \forall (u, v) \in E$

This is a linear program! But the solution might not correspond to a min cut.

Example 3: LP Solution on a Small Graph



Solution values:

$x : s = 0, a = 0.3,$
 $b = 0.5, c = 0.8,$
 $t = 1$

$y : sa = 0.3,$
 $sb = 0.5,$
 $ac = 0.5,$
 $bc = 0.3,$
 $ct = 0.2,$
 $at = 0.7$

Example 3: “Rounding” an LP

Question: How do we convert a fractional LP solution to an actual cut? We “round” it!

Algorithm: Random Threshold for Min Cut LP

1. Solve LP relaxation to get x_v^* for all $v \in V$
2. Choose threshold $\tau \in [0, 1]$ uniformly at random
3. Set $S = \{v \in V : x_v^* < \tau\}$ and $T = V \setminus S$
4. Return cut (S, T)

Lemma:

The algorithm always produces a valid s - t cut.

Proof

Since $x_s^* = 0$ and $x_t^* = 1$, for any threshold $\tau \in [0, 1]$:

- $x_s^* = 0 < \tau \leq 1$, so $s \in S$
- $x_t^* = 1 \not< \tau$, so $t \notin S$, thus $t \in T$

Example 3: “Rounding” an LP

Lemma:

For any edge $(u, v) \in E$:

$$\Pr[\text{edge } (u, v) \text{ is cut}] = \max(0, x_v^* - x_u^*) \leq y_{uv}^*$$

Proof

Recall: $S = \{v : x_v^* < \tau\}$. Edge (u, v) crosses from S to T iff $u \in S$ and $v \in T$:

$$x_u^* < \tau \leq x_v^*$$

Since τ is uniform on $[0, 1]$:

$$\Pr[\text{edge cut}] = \Pr[x_u^* < \tau \leq x_v^*] = \max(0, x_v^* - x_u^*) \leq y_{uv}^*$$

(The max accounts for the case when $x_u^* \geq x_v^*$, where the probability is 0.)

The inequality follows from the LP constraint $x_v^* \leq x_u^* + y_{uv}^*$, we get $y_{uv}^* \geq x_v^* - x_u^*$. □

Example 3: “Rounding” an LP

Lemma:

The expected cost of the randomized cut is at most OPT_{LP} .

Proof

By linearity of expectation:

$$\begin{aligned} E[\text{cost}] &= E \left[\sum_{e \in E} c_e \cdot \mathbb{1}[\text{edge } e \text{ cut}] \right] \\ &= \sum_{e \in E} c_e \cdot \Pr[\text{edge } e \text{ cut}] \end{aligned}$$

By the previous lemma, $\Pr[\text{edge } e \text{ cut}] \leq y_e^*$ for all $e \in E$. Thus:

$$E[\text{cost}] \leq \sum_{e \in E} c_e \cdot y_e^* = \text{OPT}_{\text{LP}}$$

Where OPT_{LP} is the optimal value of the LP relaxation. □

Example 3: Putting It Together

LP Relaxation Property: $\text{OPT}_{\text{LP}} \leq \text{OPT}$

The LP has a larger feasible set than the IP (every integer solution is also LP-feasible), so the LP optimum can only be smaller (we're minimizing).

Upper bound: By combining the lemmas:

$$E[\text{cost}] \leq \text{OPT}_{\text{LP}} \leq \text{OPT}$$

Lower bound: The expected cost is always at least OPT because any cut produced by the algorithm is a valid integer solution, so by definition of the IP: $\text{cost} \geq \text{OPT}$, thus $E[\text{cost}] \geq \text{OPT}$.

Conclusion: $\text{OPT} \leq E[\text{cost}] \leq \text{OPT}$, therefore $\text{OPT}_{\text{LP}} = \text{OPT}$!

The LP relaxation is tight, and any τ achieving the expected cost finds a min cut.



Optimization ► Linear Programming

Lecture 11

- The Approximation Framework
- Linear Programs
- Solving Linear Programs
- Relaxations and Roundings

Why Should We Care About LPs?

Three key reasons:

1. Efficiently solvable in theory

- LPs can be solved in polynomial time
- Contrast with Integer Programs: NP-hard!

2. Efficiently solvable in practice

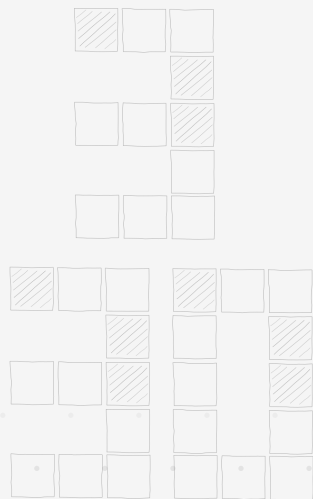
- Powerful software packages widely used in industry and research
- Examples: CPLEX, Gurobi, GLPK, HiGHS

3. Rich and fundamental theory

- Deep connections to optimization, geometry, game theory, economics
- Powerful tool for algorithm design (as we've seen!)

Remark

We won't focus on LP theory or algorithms in this course. For the rich theory and practice of LPs, see courses like ORF 307 (Optimization).



Optimization ► Linear Programming

Lecture 11

- The Approximation Framework
- Linear Programs
- Solving Linear Programs
- Relaxations and Roundings

The Relaxation and Rounding Framework

Recall Our Goal: Find approximate solutions to hard optimization problems

Key Idea: Certificates for lower/upper bounds on OPT

- Help us prove approximation guarantees
- We use LP relaxations as certificates

General Approach:

1. Model problem as an Integer Program (IP)
2. Relax to a Linear Program (LP) by allowing fractional values
3. Solve LP to get OPT_{LP} (polynomial time!)
4. OPT_{LP} provides a certificate: $\text{OPT}_{\text{LP}} \leq \text{OPT}$
5. Round LP solution to get integral solution
6. Compare rounded solution quality to OPT_{LP}
7. By proxy, this tells us about quality w.r.t. OPT

Recall/Review: Vertex Cover

Input: Undirected graph $G = (V, E)$

Goal: Find a minimum-size subset $S \subseteq V$ such that every edge has at least one endpoint in S

Formally: For every edge $(u, v) \in E$: $u \in S$ or $v \in S$ (or both)

What we know:

- Vertex Cover is NP-hard
- Last lecture: 2-approximation algorithm (greedy maximal matching approach)
- Today: Another 2-approximation using LP relaxation

Vertex Cover: Integer Program vs. LP Relaxation

Integer Program (IP)

Variables: $x_v \in \{0, 1\}$ for each $v \in V$

Minimize: $\sum_{v \in V} x_v$

Subject to:

- For every edge $(u, v) \in E$:

$$x_u + x_v \geq 1$$

- $x_v \in \{0, 1\}$ for all v

LP Relaxation

Variables: $x_v \in [0, 1]$ for each $v \in V$

Minimize: $\sum_{v \in V} x_v$

Subject to:

- For every edge $(u, v) \in E$:

$$x_u + x_v \geq 1$$

- $0 \leq x_v \leq 1$ for all v

Remark

Key observation: $\text{OPT}_{\text{LP}} \leq \text{OPT}$ (LP has larger feasible set)

We will round the LP solution to get an integral solution and use OPT_{LP} as our certificate/lower bound.

From LP Solution to Vertex Cover

Question: The LP gives us fractional values x_v^* . How do we get an actual vertex cover?

Algorithm: LP Rounding for Vertex Cover

1. Solve the LP relaxation to get x_v^* for all $v \in V$
2. Let $S = \{v \in V : x_v^* \geq 1/2\}$
3. Return S as the vertex cover

Key Questions:

- Is S a valid vertex cover?
- How large is S compared to the optimal vertex cover?

Goal: Compare $|S|$ to OPT_{LP} , which by proxy gives us a bound on $|S|$ vs. OPT

Analysis: Validity

Lemma:

$S = \{v : x_v^* \geq 1/2\}$ is a valid vertex cover.

Proof

Consider any edge $(u, v) \in E$.

By the LP constraints: $x_u^* + x_v^* \geq 1$

Since $x_u^* + x_v^* \geq 1$, at least one of x_u^* or x_v^* must be $\geq 1/2$.

Therefore, at least one of u or v is in S .

Since this holds for every edge, S is a valid vertex cover. □

Analysis: Approximation Ratio

Lemma:

$|S| \leq 2 \cdot \text{OPT}$, where OPT is the size of the optimal vertex cover.

Proof

Let x_v^* be the optimal LP solution, and let $\text{OPT}_{\text{LP}} = \sum_v x_v^*$ denote the value of this optimal LP solution. Now, we bound the size of S in terms of OPT_{LP} . Recall that $S = \{v : x_v^* \geq 1/2\}$. For each vertex $v \in S$, we have $x_v^* \geq 1/2$, which means $2x_v^* \geq 1$. Therefore:

$$|S| = \sum_{v \in S} 1 \leq \sum_{v \in S} 2x_v^* \leq 2 \sum_{v \in V} x_v^* = 2 \cdot \text{OPT}_{\text{LP}}$$

Recall that $\text{OPT}_{\text{LP}} \leq \text{OPT}$, since the LP relaxation has a larger feasible set than the IP.

Combining these two inequalities, we obtain $|S| \leq 2 \cdot \text{OPT}_{\text{LP}} \leq 2 \cdot \text{OPT}$. □

Conclusion: This algorithm is a **2-approximation** for Vertex Cover.

Why This Approach?

We already knew a 2-approximation for Vertex Cover. Why care about this one?

- **Simplicity:** Once you understand LP relaxations, this approach is conceptually clean and systematic
- **Generality:** The LP relaxation framework extends to many other problems where ad-hoc techniques may not work

Example: Weighted Vertex Cover

Given a graph $G = (V, E)$ where each vertex v has a weight $w_v > 0$, find a vertex cover S that minimizes $\sum_{v \in S} w_v$.

Exercise: Show that the same LP relaxation approach (with appropriately weighted objective) gives a 2-approximation for weighted Vertex Cover.

Hint: The LP relaxation is:

$$\begin{aligned} \text{Minimize: } & \sum_{v \in V} w_v \cdot x_v \\ \text{Subject to: } & x_u + x_v \geq 1 \quad \forall (u, v) \in E \\ & 0 \leq x_v \leq 1 \quad \forall v \in V \end{aligned}$$

Round the same way: $S = \{v : x_v^* \geq 1/2\}$.

Summary: LP Relaxation Technique

General Strategy:

1. Model NP-hard problem as an Integer Program
2. Relax integrality constraints to get an LP
3. Solve the LP in polynomial time
4. LP optimum provides a **certificate** for lower/upper bound on OPT
5. Round the fractional solution to an integral one
6. Analyze approximation ratio by comparing to LP certificate

For Vertex Cover:

- LP relaxation: $\text{OPT}_{\text{LP}} \leq \text{OPT}$ (certificate!)
- Simple rounding ($x_v^* \geq 1/2$) gives 2-approximation
- Proof: $|S| \leq 2 \cdot \text{OPT}_{\text{LP}} \leq 2 \cdot \text{OPT}$

LPs provide certificates for proving approximation guarantees!