Lecture 22: Linear Programming

Jessica Sorrell

November 13, 2025
601.433/633 Introduction to Algorithms
Slides by Michael Dinitz

Introduction

Today: What, why, and juste a taste of how

- Entire course on linear programming over in AMS. Super important topic!
- Fast algorithms in theory and in practice.

Introduction

Today: What, why, and juste a taste of how

- Entire course on linear programming over in AMS. Super important topic!
- Fast algorithms in theory and in practice.

Why: Even more general than max-flow, can still be solved in polynomial time!

- Max flow important in its own right, but also because it can be used to solve many other things (max bipartite matching)
- Linear programming: important in its own right, but also even more general than max-flow.
- Can model many, many problems!

Example: Planning Your Week

168 hours in a week. How much time to spend:

- Studying ($\boldsymbol{S}$)
- Partying ($\boldsymbol{P}$)
- Everything else ($\boldsymbol{E}$)

Example: Planning Your Week

168 hours in a week. How much time to spend:

- Studying (S)
- Partying (P)
- Everything else (E)

Constraints:

- $E \geq 56$ (at least 8 hours/day sleep, shower, etc.)
- $P + E \geq 70$ (need to stay sane)
- $S \geq 60$ (to pass your classes)
- $2S + E - 3P \geq 150$ (too much partying requires studying or sleep)

Example: Planning Your Week

168 hours in a week. How much time to spend:

- Studying (S)
- Partying (P)
- Everything else (E)

Constraints:

- $E \geq 56$ (at least 8 hours/day sleep, shower, etc.)
- $P + E \geq 70$ (need to stay sane)
- $S \geq 60$ (to pass your classes)
- $2S + E - 3P \geq 150$ (too much partying requires studying or sleep)

Question: Is this possible? Is there a *feasible* solution?

Example: Planning Your Week

168 hours in a week. How much time to spend:

- Studying (S)
- Partying (P)
- Everything else (E)

Constraints:

- $E \geq 56$ (at least 8 hours/day sleep, shower, etc.)
- $P + E \geq 70$ (need to stay sane)
- $S \geq 60$ (to pass your classes)
- $2S + E - 3P \geq 150$ (too much partying requires studying or sleep)

Question: Is this possible? Is there a *feasible* solution?

- Yes! $S = 80$, $P = 20$, $E = 68$

Example: Planning Your Week

168 hours in a week. How much time to spend:

- Studying (S)
- Partying (P)
- Everything else (E)

Constraints:

- $E \geq 56$ (at least 8 hours/day sleep, shower, etc.)
- $P + E \geq 70$ (need to stay sane)
- $S \geq 60$ (to pass your classes)
- $2S + E - 3P \geq 150$ (too much partying requires studying or sleep)

Question: Is this possible? Is there a *feasible* solution?

- Yes! $S = 80$, $P = 20$, $E = 68$

Question: Suppose “happiness” is $2P + 3E$. Can we find a feasible solution maximizing this?

Linear Programming

Input (a "linear program"):

- $\boldsymbol{n}$ variables $\boldsymbol{x_1, \ldots, x_n}$ (take values in $\mathbb{R}$)
- $\boldsymbol{m}$ *non-strict linear inequalities* in these variables (constraints)
  - E.g.: $\mathbf{3x_1 + 4x_2 \leq 6}$, $\quad \mathbf{0 \leq x_1 \leq 3} \quad \mathbf{x_2 - 3x_3 + 2x_7 = 17}$
  - Not allowed (examples): $\mathbf{x_2 x_3 \geq 5}$, $\quad \mathbf{x_4 < 2}$, $\quad \mathbf{x_5 + \log x_2 \geq 4}$
- Possibly a *linear* objective function
  - $\mathbf{\max 2x_3 - 4x_5}$, $\quad \mathbf{\min \frac{5}{2} x_4 + x_2}$, $\quad \ldots$

Goals:

- Feasibility: Find values for $\boldsymbol{x}$'s that satisfy all constraints
- Optimization: Find feasible solutions maximizing/minimizing objective function

Both achievable in polynomial time, reasonably fast!

Planning your week as an LP

Variables: $\boldsymbol{P}, \boldsymbol{E}, \boldsymbol{S}$

Planning your week as an LP

Variables: $\boldsymbol{P}, \boldsymbol{E}, \boldsymbol{S}$

$$\mathbf{max} \quad \mathbf{2}\boldsymbol{P} + \boldsymbol{E}$$

Planning your week as an LP

Variables: $\boldsymbol{P}, \boldsymbol{E}, \boldsymbol{S}$

$$\begin{aligned} \textbf{max} \quad & \mathbf{2}\boldsymbol{P} + \boldsymbol{E} \\ \text{subject to} \quad & \boldsymbol{E} \geq \mathbf{56} \\ & \boldsymbol{S} \geq \mathbf{60} \\ & \mathbf{2}\boldsymbol{S} + \boldsymbol{E} - \mathbf{3}\boldsymbol{P} \geq \mathbf{150} \\ & \boldsymbol{P} + \boldsymbol{E} \geq \mathbf{70} \end{aligned}$$

Planning your week as an LP

Variables: P, E, S

$$
\begin{aligned}
\textbf{max} \quad & 2P + E \\
\text{subject to} \quad & E \geq 56 \\
& S \geq 60 \\
& 2S + E - 3P \geq 150 \\
& P + E \geq 70 \\
& P + S + E = 168 \\
& P \geq 0 \\
& S \geq 0 \\
& E \geq 0
\end{aligned}
$$

Planning your week as an LP

Variables: $\boldsymbol{P}, \boldsymbol{E}, \boldsymbol{S}$

$$
\begin{aligned}
\textbf{max} \quad & 2P + E \\
\text{subject to} \quad & E \geq 56 \\
& S \geq 60 \\
& 2S + E - 3P \geq 150 \\
& P + E \geq 70 \\
& P + S + E = 168 \\
& P \geq 0 \\
& S \geq 0 \\
& E \geq 0
\end{aligned}
$$

When using an LP to model your problem, need to be sure that *all* aspects of your problem included!

Operations Research-style Example

Four different manufacturing plants for making cars:

	labor	materials	pollution
Plant 1	2	3	15
Plant 2	3	4	10
Plant 3	4	5	9
Plant 4	5	6	7

Operations Research-style Example

Four different manufacturing plants for making cars:

	labor	materials	pollution
Plant 1	2	3	15
Plant 2	3	4	10
Plant 3	4	5	9
Plant 4	5	6	7

- Need to produce at least **400** cars at plant 3 (labor agreement)
- Have **3300** total hours of labor, **4000** units of material
- Environmental law: produce at most **12000** pollution
- Make as many cars as possible

OR example as an LP

Four different manufacturing plants for making cars:

	labor	materials	pollution
Plant 1	2	3	15
Plant 2	3	4	10
Plant 3	4	5	9
Plant 4	5	6	7

Variables:

OR example as an LP

Four different manufacturing plants for making cars:

	labor	materials	pollution
Plant 1	2	3	15
Plant 2	3	4	10
Plant 3	4	5	9
Plant 4	5	6	7

Variables: x_i = # cars produced at plant i, for $i \in \{1, 2, 3, 4\}$

OR example as an LP

Four different manufacturing plants for making cars:

	labor	materials	pollution
Plant 1	2	3	15
Plant 2	3	4	10
Plant 3	4	5	9
Plant 4	5	6	7

Variables: x_i = # cars produced at plant i, for $i \in \{1, 2, 3, 4\}$

Objective:

OR example as an LP

Four different manufacturing plants for making cars:

	labor	materials	pollution
Plant 1	2	3	15
Plant 2	3	4	10
Plant 3	4	5	9
Plant 4	5	6	7

Variables: x_i = # cars produced at plant i, for $i \in \{1, 2, 3, 4\}$

Objective: max $x_1 + x_2 + x_3 + x_4$

OR example as an LP

Four different manufacturing plants for making cars:

	labor	materials	pollution
Plant 1	2	3	15
Plant 2	3	4	10
Plant 3	4	5	9
Plant 4	5	6	7

Variables: x_i = # cars produced at plant i, for $i \in \{1, 2, 3, 4\}$

Objective: **max** $x_1 + x_2 + x_3 + x_4$

Constraints:

OR example as an LP

Four different manufacturing plants for making cars:

	labor	materials	pollution
Plant 1	2	3	15
Plant 2	3	4	10
Plant 3	4	5	9
Plant 4	5	6	7

Variables: x_i = # cars produced at plant i, for $i \in \{1, 2, 3, 4\}$

Objective: max $x_1 + x_2 + x_3 + x_4$

Constraints:

$$x_3 \geq 400$$

OR example as an LP

Four different manufacturing plants for making cars:

	labor	materials	pollution
Plant 1	2	3	15
Plant 2	3	4	10
Plant 3	4	5	9
Plant 4	5	6	7

Variables: x_i = # cars produced at plant i, for $i \in \{1, 2, 3, 4\}$

Objective: $\max \; x_1 + x_2 + x_3 + x_4$

Constraints:

$$\begin{aligned} x_3 &\geq 400 \\ 2x_1 + 3x_2 + 4x_3 + 5x_4 &\leq 3300 \end{aligned}$$

OR example as an LP

Four different manufacturing plants for making cars:

	labor	materials	pollution
Plant 1	2	3	15
Plant 2	3	4	10
Plant 3	4	5	9
Plant 4	5	6	7

Variables: x_i = # cars produced at plant i, for $i \in \{1, 2, 3, 4\}$

Objective: $\max\ x_1 + x_2 + x_3 + x_4$

Constraints:

$$\begin{aligned} x_3 &\geq 400 \\ 2x_1 + 3x_2 + 4x_3 + 5x_4 &\leq 3300 \\ 3x_1 + 4x_2 + 5x_3 + 6x_4 &\leq 4000 \end{aligned}$$

OR example as an LP

Four different manufacturing plants for making cars:

	labor	materials	pollution
Plant 1	2	3	15
Plant 2	3	4	10
Plant 3	4	5	9
Plant 4	5	6	7

Variables: x_i = # cars produced at plant i, for $i \in \{1, 2, 3, 4\}$

Objective: $\max \; x_1 + x_2 + x_3 + x_4$

Constraints:

$$\begin{aligned} x_3 &\geq 400 \\ 2x_1 + 3x_2 + 4x_3 + 5x_4 &\leq 3300 \\ 3x_1 + 4x_2 + 5x_3 + 6x_4 &\leq 4000 \\ 15x_1 + 10x_2 + 9x_3 + 7x_4 &\leq 12000 \end{aligned}$$

OR example as an LP

Four different manufacturing plants for making cars:

	labor	materials	pollution
Plant 1	2	3	15
Plant 2	3	4	10
Plant 3	4	5	9
Plant 4	5	6	7

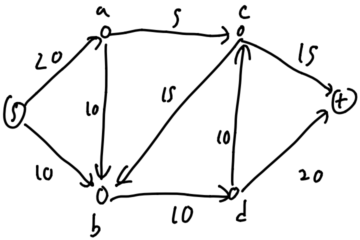
Variables: x_i = # cars produced at plant i, for $i \in \{1, 2, 3, 4\}$

Objective: max $x_1 + x_2 + x_3 + x_4$

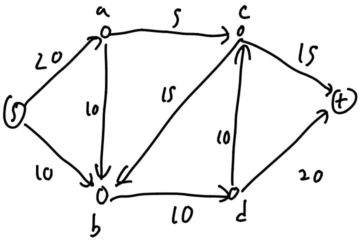
Constraints:

$$\begin{aligned} x_3 &\geq 400 \\ 2x_1 + 3x_2 + 4x_3 + 5x_4 &\leq 3300 \\ 3x_1 + 4x_2 + 5x_3 + 6x_4 &\leq 4000 \\ 15x_1 + 10x_2 + 9x_3 + 7x_4 &\leq 12000 \\ x_i &\geq 0 \qquad \forall i \in \{1, 2, 3, 4\} \end{aligned}$$

Max Flow as LP

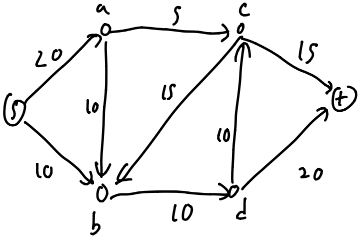


Max Flow as LP



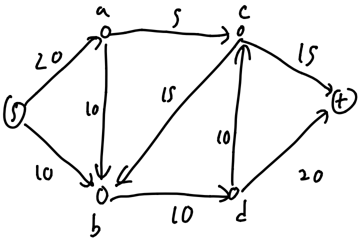
Variables:

Max Flow as LP



Variables: $f(e)$ for all $e \in E$

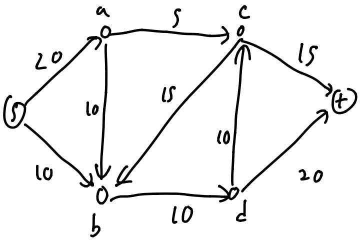
Max Flow as LP



Variables: $f(e)$ for all $e \in E$

Objective:

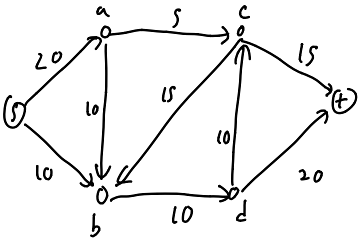
Max Flow as LP



Variables: $f(e)$ for all $e \in E$

Objective: $\max \sum_v f(s,v) - \sum_v f(v,s)$

Max Flow as LP

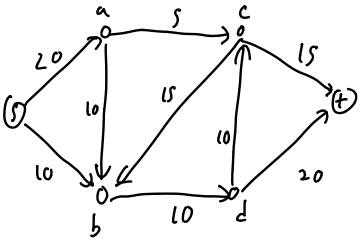


Variables: $f(e)$ for all $e \in E$

Objective: $\max \sum_v f(s,v) - \sum_v f(v,s)$

Constraints:

Max Flow as LP



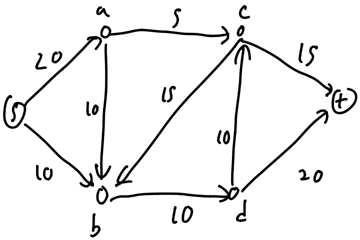
Variables: $f(e)$ for all $e \in E$

Objective: $\max \sum_v f(s,v) - \sum_v f(v,s)$

Constraints:

$$\sum_v f(v,u) - \sum_v f(u,v) = 0 \qquad \forall u \in V \setminus \{s,t\}$$

Max Flow as LP



Variables: $f(e)$ for all $e \in E$

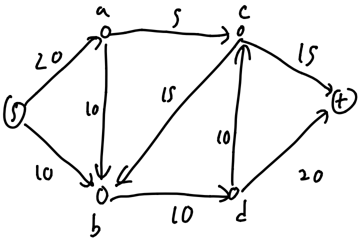
Objective: $\max \sum_v f(s,v) - \sum_v f(v,s)$

Constraints:

$$\sum_v f(v,u) - \sum_v f(u,v) = 0 \qquad \forall u \in V \setminus \{s,t\}$$

$$f(e) \le c(e) \qquad \forall e \in E$$

Max Flow as LP



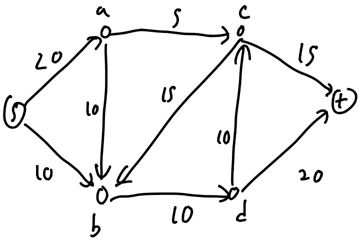
Variables: $f(e)$ for all $e \in E$

Objective: $\max \sum_v f(s,v) - \sum_v f(v,s)$

Constraints:

$$\begin{aligned}
\sum_v f(v,u) - \sum_v f(u,v) &= 0 & \forall u \in V \setminus \{s,t\} \\
f(e) &\le c(e) & \forall e \in E \\
f(e) &\ge 0 & \forall e \in E
\end{aligned}$$

Max Flow as LP



Variables: $f(e)$ for all $e \in E$

Objective: $\max \sum_v f(s,v) - \sum_v f(v,s)$

Constraints:

$$\sum_v f(v,u) - \sum_v f(u,v) = 0 \qquad \forall u \in V \setminus \{s,t\}$$

$$f(e) \le c(e) \qquad \forall e \in E$$

$$f(e) \ge 0 \qquad \forall e \in E$$

So can solve max-flow and min-cut (slower) by using generic LP solver

Multicommodity Flow

Generalization of max-flow with multiple commodities that can't mix, but use up same capacity

Multicommodity Flow

Generalization of max-flow with multiple commodities that can't mix, but use up same capacity

Setup:

- Directed graph $G = (V, E)$
- Capacities $c : E \to \mathbb{R}_{\geq 0}$
- k source-sink pairs $\{(s_i, t_i)\}_{i \in [k]}$

Goal: send flow of commodity i from s_i to t_i, max total flow sent across all commodities

Multicommodity Flow

Generalization of max-flow with multiple commodities that can't mix, but use up same capacity

Setup:

- Directed graph $G = (V, E)$
- Capacities $c : E \to \mathbb{R}_{\geq 0}$
- k source-sink pairs $\{(s_i, t_i)\}_{i \in [k]}$

Goal: send flow of commodity i from s_i to t_i, max total flow sent across all commodities

Variables:

Multicommodity Flow

Generalization of max-flow with multiple commodities that can't mix, but use up same capacity

Setup:

- Directed graph $G = (V, E)$
- Capacities $c : E \to \mathbb{R}_{\geq 0}$
- k source-sink pairs $\{(s_i, t_i)\}_{i \in [k]}$

Goal: send flow of commodity i from s_i to t_i, max total flow sent across all commodities

Variables: $f_i(e)$ for all $e \in E$ and for all $i \in [k]$.
Flow of commodity i on edge e

Multicommodity Flow

Generalization of max-flow with multiple commodities that can't mix, but use up same capacity

Setup:

- Directed graph $G = (V, E)$
- Capacities $c : E \to \mathbb{R}_{\geq 0}$
- k source-sink pairs $\{(s_i, t_i)\}_{i \in [k]}$

Goal: send flow of commodity i from s_i to t_i, max total flow sent across all commodities

Variables: $f_i(e)$ for all $e \in E$ and for all $i \in [k]$.
Flow of commodity i on edge e

Objective:

Multicommodity Flow

Generalization of max-flow with multiple commodities that can't mix, but use up same capacity

Setup:

- Directed graph $G = (V, E)$
- Capacities $c : E \to \mathbb{R}_{\geq 0}$
- k source-sink pairs $\{(s_i, t_i)\}_{i \in [k]}$

Goal: send flow of commodity i from s_i to t_i, max total flow sent across all commodities

Variables: $f_i(e)$ for all $e \in E$ and for all $i \in [k]$.
Flow of commodity i on edge e

Objective: $\max \sum_{i=1}^{k} \left(\sum_v f_i(s_i, v) - \sum_v f_i(v, s_i) \right)$

Multicommodity Flow

Generalization of max-flow with multiple commodities that can't mix, but use up same capacity

Setup:

- Directed graph $G = (V, E)$
- Capacities $c : E \to \mathbb{R}_{\geq 0}$
- k source-sink pairs $\{(s_i, t_i)\}_{i \in [k]}$

Goal: send flow of commodity i from s_i to t_i, max total flow sent across all commodities

Variables: $f_i(e)$ for all $e \in E$ and for all $i \in [k]$. Flow of commodity i on edge e

Objective: $\max \sum_{i=1}^{k} \left(\sum_v f_i(s_i, v) - \sum_v f_i(v, s_i)\right)$

Constraints:

Multicommodity Flow

Generalization of max-flow with multiple commodities that can't mix, but use up same capacity

Setup:

- Directed graph $G = (V, E)$
- Capacities $c : E \to \mathbb{R}_{\geq 0}$
- k source-sink pairs $\{(s_i, t_i)\}_{i \in [k]}$

Goal: send flow of commodity i from s_i to t_i, max total flow sent across all commodities

Variables: $f_i(e)$ for all $e \in E$ and for all $i \in [k]$.
Flow of commodity i on edge e

Objective: $\max \sum_{i=1}^{k} \left(\sum_v f_i(s_i, v) - \sum_v f_i(v, s_i)\right)$

Constraints:

$$\sum_v f_i(v, u) - \sum_v f_i(u, v) = 0 \qquad \forall i \in [k],\ \forall u \in V \setminus \{s_i, t_i\}$$

Multicommodity Flow

Generalization of max-flow with multiple commodities that can't mix, but use up same capacity

Setup:

- Directed graph $\boldsymbol{G = (V, E)}$
- Capacities $\boldsymbol{c : E \to \mathbb{R}_{\geq 0}}$
- $\boldsymbol{k}$ source-sink pairs $\boldsymbol{\{(s_i, t_i)\}_{i \in [k]}}$

Goal: send flow of commodity $\boldsymbol{i}$ from $\boldsymbol{s_i}$ to $\boldsymbol{t_i}$, max total flow sent across all commodities

Variables: $\boldsymbol{f_i(e)}$ for all $\boldsymbol{e \in E}$ and for all $\boldsymbol{i \in [k]}$.
Flow of commodity $\boldsymbol{i}$ on edge $\boldsymbol{e}$

Objective: $\boldsymbol{\max \sum_{i=1}^{k} \left(\sum_v f_i(s_i, v) - \sum_v f_i(v, s_i)\right)}$

Constraints:

$$\boldsymbol{\sum_v f_i(v,u) - \sum_v f_i(u,v) = 0 \qquad \forall i \in [k],\ \forall u \in V \setminus \{s_i, t_i\}}$$

$$\boldsymbol{\sum_{i=1}^{k} f_i(e) \leq c(e) \qquad \forall e \in E}$$

Multicommodity Flow

Generalization of max-flow with multiple commodities that can't mix, but use up same capacity

Setup:

- Directed graph $G = (V, E)$
- Capacities $c : E \to \mathbb{R}_{\geq 0}$
- k source-sink pairs $\{(s_i, t_i)\}_{i \in [k]}$

Goal: send flow of commodity i from s_i to t_i, max total flow sent across all commodities

Variables: $f_i(e)$ for all $e \in E$ and for all $i \in [k]$.
Flow of commodity i on edge e

Objective: $\max \sum_{i=1}^{k} \left(\sum_v f_i(s_i, v) - \sum_v f_i(v, s_i)\right)$

Constraints:

$$\sum_v f_i(v, u) - \sum_v f_i(u, v) = 0 \qquad \forall i \in [k],\ \forall u \in V \setminus \{s_i, t_i\}$$

$$\sum_{i=1}^{k} f_i(e) \leq c(e) \qquad \forall e \in E$$

$$f_i(e) \geq 0 \qquad \forall e \in E,\ \forall i \in [k]$$

Concurrent Flow

Multicommodity flow, but:

- Also given *demands* $d : [k] \to \mathbb{R}_{\geq 0}$
- Question: Is there a multicommodity flow that sends at least $d(i)$ commodity-i flow from s_i to t_i for all $i \in [k]$?

Concurrent Flow

Variables: $f_i(e)$ for all $e \in E$ and for all $i \in [k]$.

Multicommodity flow, but:

- Also given *demands* $d : [k] \to \mathbb{R}_{\geq 0}$
- Question: Is there a multicommodity flow that sends at least $d(i)$ commodity-i flow from s_i to t_i for all $i \in [k]$?

Concurrent Flow

Multicommodity flow, but:

- Also given *demands* $d : [k] \to \mathbb{R}_{\geq 0}$
- Question: Is there a multicommodity flow that sends at least $d(i)$ commodity-i flow from s_i to t_i for all $i \in [k]$?

Variables: $f_i(e)$ for all $e \in E$ and for all $i \in [k]$.

Constraints:

Concurrent Flow

Multicommodity flow, but:

- Also given *demands* $d : [k] \to \mathbb{R}_{\geq 0}$
- Question: Is there a multicommodity flow that sends at least $d(i)$ commodity-i flow from s_i to t_i for all $i \in [k]$?

Variables: $f_i(e)$ for all $e \in E$ and for all $i \in [k]$.

Constraints:

$$\sum_v f_i(v,u) - \sum_v f_i(u,v) = 0 \qquad \forall i \in [k],\ \forall u \in V \setminus \{s_i, t_i\}$$

$$\sum_{i=1}^{k} f_i(e) \leq c(e) \qquad \forall e \in E$$

$$f_i(e) \geq 0 \qquad \forall e \in E,\ \forall i \in [k]$$

Concurrent Flow

Multicommodity flow, but:

- Also given *demands* $d : [k] \to \mathbb{R}_{\geq 0}$
- Question: Is there a multicommodity flow that sends at least $d(i)$ commodity-i flow from s_i to t_i for all $i \in [k]$?

Variables: $f_i(e)$ for all $e \in E$ and for all $i \in [k]$.

Constraints:

$$\sum_v f_i(v,u) - \sum_v f_i(u,v) = 0 \qquad \forall i \in [k],\ \forall u \in V \setminus \{s_i, t_i\}$$

$$\sum_{i=1}^{k} f_i(e) \leq c(e) \qquad \forall e \in E$$

$$f_i(e) \geq 0 \qquad \forall e \in E,\ \forall i \in [k]$$

$$\sum_v f_i(s_i,v) - \sum_v f_i(v,s_i) \geq d(i) \qquad \forall i \in [k]$$

Maximum Concurrent Flow

If answer is no: how much do we need to scale down demands so that there is a multicommodity flow?

Maximum Concurrent Flow

If answer is no: how much do we need to scale down demands so that there is a multicommodity flow?

Variables:

- $f_i(e)$ for all $e \in E$ and for all $i \in [k]$.
- λ

Objective: $\max \lambda$

Constraints:

$$\begin{aligned}
\sum_v f_i(v,u) - \sum_v f_i(u,v) &= 0 && \forall i \in [k],\ \forall u \in V \setminus \{s_i, t_i\} \\
\sum_{i=1}^{k} f_i(e) &\le c(e) && \forall e \in E \\
f_i(e) &\ge 0 && \forall e \in E,\ \forall i \in [k] \\
\sum_v f_i(s_i,v) - \sum_v f_i(v,s_i) &\ge \lambda d(i) && \forall i \in [k]
\end{aligned}$$

Shortest $s - t$ path

Very surprising LP!

Variables: d_v for all $v \in V$: shortest-path distance from s to v

$$
\begin{aligned}
\textbf{max} \quad & d_t \\
\text{subject to} \quad & d_s = 0 \\
& d_v \le d_u + \ell(u, v) && \forall (u, v) \in E
\end{aligned}
$$

Shortest $s - t$ path

Very surprising LP!

Variables: d_v for all $v \in V$: shortest-path distance from s to v

$$\begin{aligned} \textbf{max} \quad & d_t \\ \text{subject to} \quad & d_s = 0 \\ & d_v \le d_u + \ell(u,v) && \forall (u,v) \in E \end{aligned}$$

Correctness Theorem: Let $\vec{d}^*$ denote the optimal LP solution. Then $d_t^* = d(s,t)$

Shortest $s - t$ path

Very surprising LP!

Variables: d_v for all $v \in V$: shortest-path distance from s to v

$$\begin{aligned} \textbf{max} \quad & d_t \\ \text{subject to} \quad & d_s = 0 \\ & d_v \leq d_u + \ell(u, v) \qquad\qquad \forall (u, v) \in E \end{aligned}$$

Correctness Theorem: Let $\vec{d}^*$ denote the optimal LP solution. Then $d_t^* = d(s, t)$

Proof Sketch: $\geq$: Let $d_v = d(s, v)$ for all $v \in V$. Feasible $\implies d_t^* \geq d_t = d(s, t)$.

Shortest $s - t$ path

Very surprising LP!
Variables: d_v for all $v \in V$: shortest-path distance from s to v

$$
\begin{aligned}
\textbf{max} \quad & d_t \\
\text{subject to} \quad & d_s = 0 \\
& d_v \leq d_u + \ell(u, v) & \forall (u, v) \in E
\end{aligned}
$$

Correctness Theorem: Let $\vec{d}^*$ denote the optimal LP solution. Then $d_t^* = d(s, t)$
Proof Sketch: $\geq$: Let $d_v = d(s, v)$ for all $v \in V$. Feasible $\implies d_t^* \geq d_t = d(s, t)$.

$\leq$: Let $P = (s = v_0, v_1, \ldots, v_k = t)$ be shortest $s \to t$ path.
Prove by induction: $d_{v_i}^* \leq d(s, v_i)$ for all i

Shortest $s - t$ path

Very surprising LP!

Variables: d_v for all $v \in V$: shortest-path distance from s to v

$$\begin{aligned} \textbf{max} \quad & d_t & \\ \text{subject to} \quad & d_s = 0 & \\ & d_v \leq d_u + \ell(u, v) & \forall (u, v) \in E \end{aligned}$$

Correctness Theorem: Let $\vec{d}^*$ denote the optimal LP solution. Then $d_t^* = d(s, t)$

Proof Sketch: $\geq$: Let $d_v = d(s, v)$ for all $v \in V$. Feasible $\implies$ $d_t^* \geq d_t = d(s, t)$.

$\leq$: Let $P = (s = v_0, v_1, \ldots, v_k = t)$ be shortest $s \to t$ path.
Prove by induction: $d_{v_i}^* \leq d(s, v_i)$ for all i
Base case: $i = 0$ ✓

Shortest $s - t$ path

Very surprising LP!
Variables: d_v for all $v \in V$: shortest-path distance from s to v

$$\begin{aligned} \max \quad & d_t \\ \text{subject to} \quad & d_s = 0 \\ & d_v \leq d_u + \ell(u, v) & \forall (u, v) \in E \end{aligned}$$

Correctness Theorem: Let $\vec{d}^*$ denote the optimal LP solution. Then $d_t^* = d(s,t)$
Proof Sketch: $\geq$: Let $d_v = d(s,v)$ for all $v \in V$. Feasible $\implies d_t^* \geq d_t = d(s,t)$.

$\leq$: Let $P = (s = v_0, v_1, \ldots, v_k = t)$ be shortest $s \to t$ path.
Prove by induction: $d_{v_i}^* \leq d(s, v_i)$ for all i
Base case: $i = 0$ ✓
Inductive step: $d_{v_i}^* \leq d_{v_{i-1}}^* + \ell(v_{i-1}, v_i) \leq d(s, v_{i-1}) + \ell(v_{i-1}, v_i) = d(s, v_i)$

Algorithms for LPs

Geometry

To get intuition: think of LPs *geometrically*

- Space: $\mathbb{R}^{\boldsymbol{n}}$ (one dimension per variable
- Linear constraint: halfspace (one side of a hyperplane)
- Feasible region: intersection of halfspaces. *Convex Polytope* (usually just called a *polytope*)

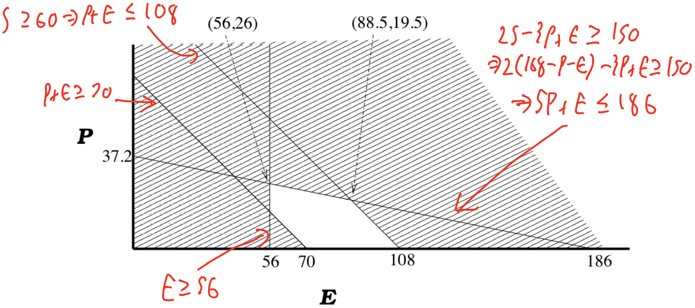
Geometry

To get intuition: think of LPs *geometrically*

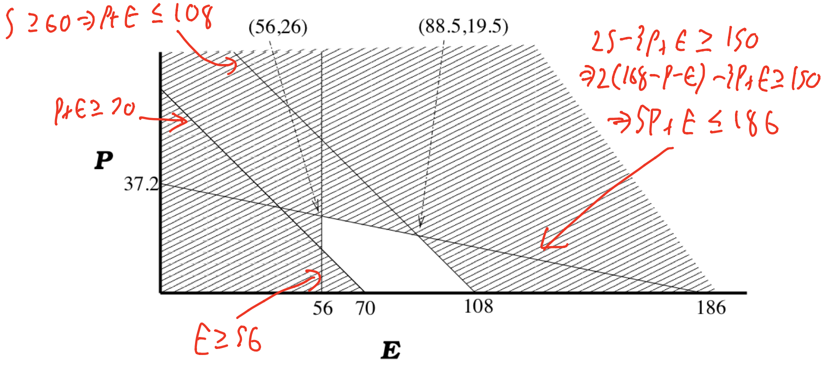
- Space: $\mathbb{R}^n$ (one dimension per variable
- Linear constraint: halfspace (one side of a hyperplane)
- Feasible region: intersection of halfspaces. *Convex Polytope* (usually just called a *polytope*)

Example: planning your week

- **3** variables S, P, E so $\mathbb{R}^3$
- But $S + P + E = 168 \implies S = 168 - P - E$
- Make this substitution, get $\mathbb{R}^2$



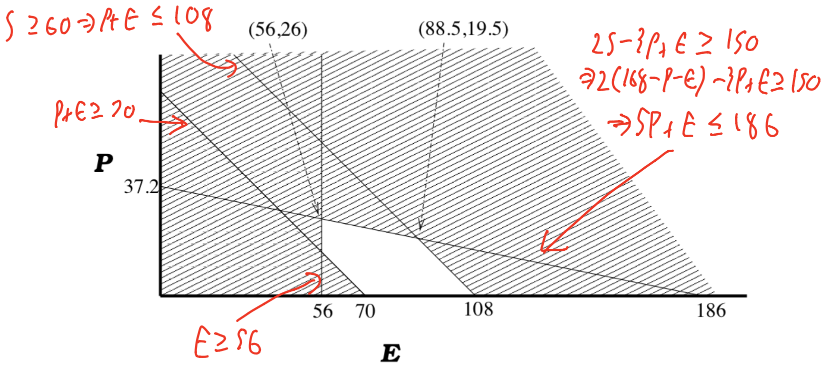
Geometry (cont'd)



Objective: feasible solution "furthest" along specified direction

- **max P: (56, 26)**
- **max $2P + E$: (88.5, 19.5)**

Geometry (cont'd)



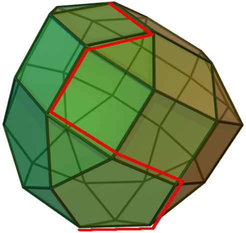
Objective: feasible solution "furthest" along specified direction

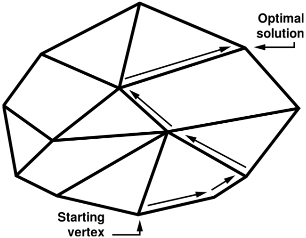
- **max P: (56, 26)**
- **max $2P + E$: (88.5, 19.5)**

Main theorem: optimal solution is always at a "corner" (also called a "vertex")

Simplex Algorithm [Dantzig 1940's]

Initialize $\vec{x}$ to an arbitrary corner
while(a neighboring corner $\vec{x}'$ of $\vec{x}$ has better objective value) {
    $\vec{x} \leftarrow \vec{x}'$
}
return $\vec{x}$





Simplex Analysis

Theorem: Simplex returns the optimal solution.

Simplex Analysis

Theorem: Simplex returns the optimal solution.

Proof Sketch:

- Objective linear $\implies$ optimal solution at a corner
- Feasible set convex + linear objective $\implies$ any local opt is global opt

$\implies$ Once simplex terminates, at global opt

Simplex Analysis

Theorem: Simplex returns the optimal solution.

Proof Sketch:

- Objective linear $\implies$ optimal solution at a corner
- Feasible set convex + linear objective $\implies$ any local opt is global opt

$\implies$ Once simplex terminates, at global opt

Problem: Exponential number of corners!

Simplex Analysis

Theorem: Simplex returns the optimal solution.

Proof Sketch:

- Objective linear $\implies$ optimal solution at a corner
- Feasible set convex + linear objective $\implies$ any local opt is global opt

$\implies$ Once simplex terminates, at global opt

Problem: Exponential number of corners!

- Slow in theory

Simplex Analysis

Theorem: Simplex returns the optimal solution.

Proof Sketch:

- Objective linear $\implies$ optimal solution at a corner
- Feasible set convex + linear objective $\implies$ any local opt is global opt

$\implies$ Once simplex terminates, at global opt

Problem: Exponential number of corners!

- Slow in theory
- Fast in practice!
  - Much of AMS LP course really about simplex: traditionally favorite algorithm of people who want to actually solve LPs

Simplex Analysis

Theorem: Simplex returns the optimal solution.

Proof Sketch:

- Objective linear $\Longrightarrow$ optimal solution at a corner
- Feasible set convex + linear objective $\Longrightarrow$ any local opt is global opt

$\Longrightarrow$ Once simplex terminates, at global opt

Problem: Exponential number of corners!

- Slow in theory
- Fast in practice!
  - Much of AMS LP course really about simplex: traditionally favorite algorithm of people who want to actually solve LPs
- Some theory to explain discrepancy ("smoothed analysis")

Ellipsoid Algorithm [Khachiyan 1980]

First polytime algorithm!

Designed to just solve feasibility question $\implies$ can also solve optimization

Ellipsoid Algorithm [Khachiyan 1980]

First polytime algorithm!

Designed to just solve feasibility question $\implies$ can also solve optimization

- Start with ellipsoid E containing feasible region P (if it exists)
- Let x be center of E
- While(x not feasible)
  - Find a hyperplane H through x such that all of P on one side
  - Let E' be the half-ellipsoid of E defined by H
  - Find a new ellipsoid $\hat{E}$ containing E' so that $vol(\hat{E}) \leq \left(1 - \frac{1}{n}\right) vol(E)$
  - Let $E = \hat{E}$ and let x be center of $\hat{E}$

Analysis

Extremely complicated!

Geometry of ellipsoids: can always find an ellipsoid containing a half-ellipsoid with at most $(1 - 1/n)$ of the volume of the original

- After t iterations, volume drops by $\left(1 - \frac{1}{n}\right)^t$ factor
- Absurdly useful inequality: $1 + x \leq e^x$
- $\left(1 - \frac{1}{n}\right)^t \leq (e^{-1/n})^t = e^{-t/n}$
- Crucial fact: if volume "too small", P must be empty. Let v a volume below which we can conclude P is empty.
- Then suffices to find t such that $(e^{-t/n}) Vol(E) \leq v$, so taking $t \geq n \log(Vol(E)/v)$ suffices

$\Longrightarrow$ Polynomial time!

Analysis

Extremely complicated!

Geometry of ellipsoids: can always find an ellipsoid containing a half-ellipsoid with at most $(1 - 1/n)$ of the volume of the original

- After t iterations, volume drops by $\left(1 - \frac{1}{n}\right)^t$ factor
- Absurdly useful inequality: $1 + x \leq e^x$
- $\left(1 - \frac{1}{n}\right)^t \leq (e^{-1/n})^t = e^{-t/n}$
- Crucial fact: if volume “too small”, P must be empty. Let v a volume below which we can conclude P is empty.
- Then suffices to find t such that $(e^{-t/n}) Vol(E) \leq v$, so taking $t \geq n \log(Vol(E)/v)$ suffices

$\Longrightarrow$ Polynomial time!

In practice: horrible.

Interior Point Methods (Karmarkar's Algorithm)

Fast in both theory and practice!