Fundamental Algorithms
Chapter 7: Linear Programming

Sevag Gharibian

Universität Paderborn

WS 2019

Outline

1. Definitions
2. Applications
3. Duality theory
4. Solving LPs

References

- CLRS Chapter 29
- Convex Optimization (Boyd and Vandenberghe):
  `https://web.stanford.edu/~boyd/cvxbook/`
- Luca Trevisan lecture notes:
  `http://theory.stanford.edu/~trevisan/cs261/lecture15.pdf`

Motivation

- Studied shortest paths, matchings, network flow, etc.
- What if I told you that many such problems can all be solved via a more general, unified framework?

Motivation

- Studied shortest paths, matchings, network flow, etc.
- What if I told you that many such problems can all be solved via a more general, unified framework?

That framework is *Linear Programming (LP)*.

Motivation

- Studied shortest paths, matchings, network flow, etc.
- What if I told you that many such problems can all be solved via a more general, unified framework?

That framework is *Linear Programming (LP)*.

LPs:

- ...are useful in everything from industrial optimization problems to approximating NP-complete problems.

Motivation

- Studied shortest paths, matchings, network flow, etc.
- What if I told you that many such problems can all be solved via a more general, unified framework?

That framework is *Linear Programming (LP)*.

LPs:

- ...are useful in everything from industrial optimization problems to approximating NP-complete problems.
- ...have long history of algorithms for them (Simplex Method, Ellipsoid Method, Interior Point Methods).

Motivation

- Studied shortest paths, matchings, network flow, etc.
- What if I told you that many such problems can all be solved via a more general, unified framework?

That framework is *Linear Programming (LP)*.

LPs:

- . . . are useful in everything from industrial optimization problems to approximating NP-complete problems.
- . . . have long history of algorithms for them (Simplex Method, Ellipsoid Method, Interior Point Methods).
- . . . can be generalized further to SDPs, cone programs, etc. Here, we focus on LPs.

Example

- Konditorei X produces 3 types of cake: Kirschtorte, Mohnkuchen, Sachertorte.
- X sells a whole cake of each type for 30,20,40 EUR, respectively.
- Assume the production of each cake requires:

Cake	Flour	Cocoa Powder	Butter
Kirschtorte	2	3	1
Mohnkuchen	3	0	2
Sachertorte	2	4	2

- X's suppliers provide, per day, 90 units flour, 70 units cocoa, 80 units butter

Example

- Konditorei X produces 3 types of cake: Kirschtorte, Mohnkuchen, Sachertorte.
- X sells a whole cake of each type for 30,20,40 EUR, respectively.
- Assume the production of each cake requires:

Cake	Flour	Cocoa Powder	Butter
Kirschtorte	2	3	1
Mohnkuchen	3	0	2
Sachertorte	2	4	2

- X's suppliers provide, per day, 90 units flour, 70 units cocoa, 80 units butter
- Q: What is max profit X can make in one day, given above constraints?

Example

- Konditorei X produces 3 types of cake: Kirschtorte, Mohnkuchen, Sachertorte.
- X sells a whole cake of each type for 30,20,40 EUR, respectively.
- Assume the production of each cake requires:

Cake	Flour	Cocoa Powder	Butter
Kirschtorte	2	3	1
Mohnkuchen	3	0	2
Sachertorte	2	4	2

- X's suppliers provide, per day, 90 units flour, 70 units cocoa, 80 units butter
- Q: What is max profit X can make in one day, given above constraints?

$$\text{maximize} \quad 30K + 20M + 25S \qquad \text{(profit)}$$

Example

- Konditorei X produces 3 types of cake: Kirschtorte, Mohnkuchen, Sachertorte.
- X sells a whole cake of each type for 30,20,40 EUR, respectively.
- Assume the production of each cake requires:

Cake	Flour	Cocoa Powder	Butter
Kirschtorte	2	3	1
Mohnkuchen	3	0	2
Sachertorte	2	4	2

- X's suppliers provide, per day, 90 units flour, 70 units cocoa, 80 units butter
- Q: What is max profit X can make in one day, given above constraints?

$$\begin{array}{lrcll} \text{maximize} & 30K + 20M + 25S & & & \text{(profit)} \\ \text{subject to} & 2K + 3M + 2S & \leq & 90 & \text{(flour constraint)} \end{array}$$

Example

- Konditorei X produces 3 types of cake: Kirschtorte, Mohnkuchen, Sachertorte.
- X sells a whole cake of each type for 30,20,40 EUR, respectively.
- Assume the production of each cake requires:

Cake	Flour	Cocoa Powder	Butter
Kirschtorte	2	3	1
Mohnkuchen	3	0	2
Sachertorte	2	4	2

- X's suppliers provide, per day, 90 units flour, 70 units cocoa, 80 units butter
- Q: What is max profit X can make in one day, given above constraints?

$$\begin{array}{lrcll} \text{maximize} & 30K + 20M + 25S & & & \text{(profit)} \\ \text{subject to} & 2K + 3M + 2S & \leq & 90 & \text{(flour constraint)} \\ & 3K + 4S & \leq & 70 & \text{(cocoa constraint)} \end{array}$$

Example

- Konditorei X produces 3 types of cake: Kirschtorte, Mohnkuchen, Sachertorte.
- X sells a whole cake of each type for 30,20,40 EUR, respectively.
- Assume the production of each cake requires:

Cake	Flour	Cocoa Powder	Butter
Kirschtorte	2	3	1
Mohnkuchen	3	0	2
Sachertorte	2	4	2

- X's suppliers provide, per day, 90 units flour, 70 units cocoa, 80 units butter
- Q: What is max profit X can make in one day, given above constraints?

$$
\begin{array}{lrcll}
\text{maximize} & 30K + 20M + 25S & & & \text{(profit)} \\
\text{subject to} & 2K + 3M + 2S & \leq & 90 & \text{(flour constraint)} \\
 & 3K + 4S & \leq & 70 & \text{(cocoa constraint)} \\
 & K + 2M + 2S & \leq & 80 & \text{(butter constraint)}
\end{array}
$$

Example

- Konditorei X produces 3 types of cake: Kirschtorte, Mohnkuchen, Sachertorte.
- X sells a whole cake of each type for 30,20,40 EUR, respectively.
- Assume the production of each cake requires:

Cake	Flour	Cocoa Powder	Butter
Kirschtorte	2	3	1
Mohnkuchen	3	0	2
Sachertorte	2	4	2

- X's suppliers provide, per day, 90 units flour, 70 units cocoa, 80 units butter
- Q: What is max profit X can make in one day, given above constraints?

$$
\begin{array}{lrcll}
\text{maximize} & 30K + 20M + 25S & & & \text{(profit)} \\
\text{subject to} & 2K + 3M + 2S & \leq & 90 & \text{(flour constraint)} \\
 & 3K + 4S & \leq & 70 & \text{(cocoa constraint)} \\
 & K + 2M + 2S & \leq & 80 & \text{(butter constraint)} \\
 & K, M, S & \geq & 0 & \text{(negative cakes = bad)}
\end{array}
$$

Outline

1. Definitions
2. Applications
3. Duality theory
4. Solving LPs

Standard form linear program (LP)

Input:

- (Cost function) $c_1, \ldots, c_n \in \mathbb{R}$.
- (Constraints) $a_{ij} \in \mathbb{R}$ for $i \in [m], j \in [n]$, and $b_1, \ldots, b_m \in \mathbb{R}$.

Primal standard form LP:

$$
\begin{array}{llcll}
\text{maximize} & \sum_{j=1}^{n} c_j x_j & & & \text{(objective function)} \\
\text{subject to} & & & & \text{(constraints)} \\
& \sum_{j=1}^{n} a_{ij} x_j & \leq & b_i & \text{for } i = 1, 2, \ldots, m \\
& x_j & \geq & 0 & \text{for } j = 1, 2, \ldots, n.
\end{array}
$$

Standard form linear program (LP)

Input:

- (Cost function) $c_1, \ldots, c_n \in \mathbb{R}$.
- (Constraints) $a_{ij} \in \mathbb{R}$ for $i \in [m], j \in [n]$, and $b_1, \ldots, b_m \in \mathbb{R}$.

Primal standard form LP:

$$
\begin{array}{llcll}
\text{maximize} & \sum_{j=1}^{n} c_j x_j & & & \text{(objective function)} \\
\text{subject to} & & & & \text{(constraints)} \\
& \sum_{j=1}^{n} a_{ij} x_j & \leq & b_i & \text{for } i = 1, 2, \ldots, m \\
& x_j & \geq & 0 & \text{for } j = 1, 2, \ldots, n.
\end{array}
$$

Equivalent Linear Algebraic formulation:

$$
\begin{array}{llcll}
\text{maximize} & c^T x & & & \text{(objective function)} \\
\text{subject to} & Ax & \leq & b & \text{(constraints)} \\
& x & \geq & 0 &
\end{array}
$$

for matrix $A \in \mathbb{R}^{m \times n}$ and column vectors $c, b \in \mathbb{R}^n$.

What does this geometrically mean?

Suppose LP has $n = 2$ variables, i.e. optimize over 2D plane $\mathbb{R}^2$.

Constraints

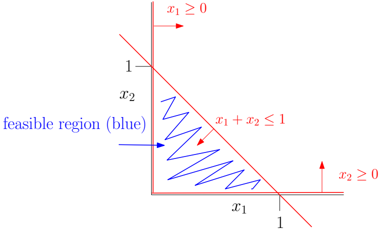
- Restrict optimization over *subset* of $\mathbb{R} \times \mathbb{R}$, called feasible region.

What does this geometrically mean?

Suppose LP has $n = 2$ variables, i.e. optimize over 2D plane $\mathbb{R}^2$.

Constraints

- Restrict optimization over *subset* of $\mathbb{R} \times \mathbb{R}$, called feasible region.
- Example: Consider constraints $x_1 \geq 0$, $x_2 \geq 0$, $x_1 + x_2 \leq 1$.

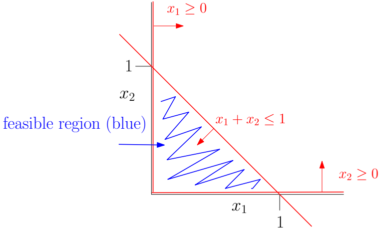


What does this geometrically mean?

Suppose LP has $n = 2$ variables, i.e. optimize over 2D plane $\mathbb{R}^2$.

Constraints

- Restrict optimization over *subset* of $\mathbb{R} \times \mathbb{R}$, called feasible region.
- Example: Consider constraints $x_1 \geq 0$, $x_2 \geq 0$, $x_1 + x_2 \leq 1$.



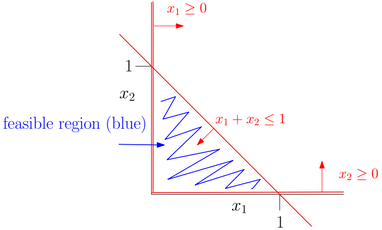
- Each constraint partitions R^2 into pair of *halfspaces*, i.e. "left" and "right" of each "dividing line" (formally, each hyperplane).

What does this geometrically mean?

Suppose LP has $n = 2$ variables, i.e. optimize over 2D plane $\mathbb{R}^2$.

Constraints

- Restrict optimization over *subset* of $\mathbb{R} \times \mathbb{R}$, called feasible region.
- Example: Consider constraints $x_1 \geq 0$, $x_2 \geq 0$, $x_1 + x_2 \leq 1$.



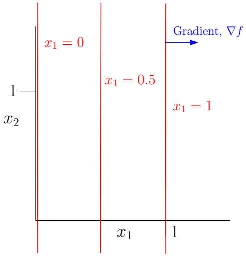
- Each constraint partitions R^2 into pair of *halfspaces*, i.e. "left" and "right" of each "dividing line" (formally, each hyperplane).
- Feasible region is intersection of these halfspaces (formally, convex polyhedron).

What about objective function?

- Objective function $f(x_1, x_2) = c_1 x_1 + c_2 x_2$ is linear by definition.

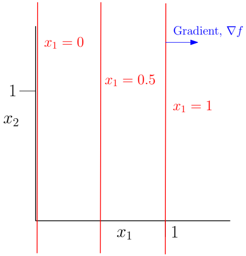
What about objective function?

- Objective function $f(x_1, x_2) = c_1x_1 + c_2x_2$ is linear by definition.
- Example: $f(x_1, x_2) = x_1$.
  - Know *slope* of line (representing objective function).
  - Don't know its *offset* from origin.



What about objective function?

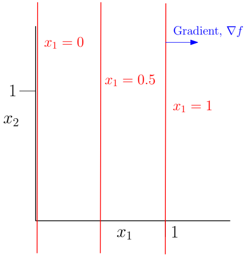
- Objective function $f(x_1, x_2) = c_1x_1 + c_2x_2$ is linear by definition.
- Example: $f(x_1, x_2) = x_1$.
  - Know *slope* of line (representing objective function).
  - Don't know its *offset* from origin.



  - Observe: As $f(x_1, x_2) = x_1$ grows, offset moves to right.

What about objective function?

- Objective function $f(x_1, x_2) = c_1x_1 + c_2x_2$ is linear by definition.
- Example: $f(x_1, x_2) = x_1$.
  - Know *slope* of line (representing objective function).
  - Don't know its *offset* from origin.



  - Observe: As $f(x_1, x_2) = x_1$ grows, offset moves to right.
- Formally, direction of movement given by *gradient* of f,

$$\nabla f = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2} \right).$$

- In this example: $\nabla f = (1, 0)$, hence the blue vector above.

Putting it all together

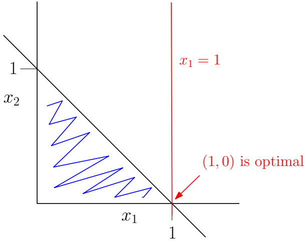
Optimizing our LP corresponds to (in our example):

- Move vertical line representing f as far right as possible, *while ensuring* it has non-empty intersection with feasible region.

Putting it all together

Optimizing our LP corresponds to (in our example):

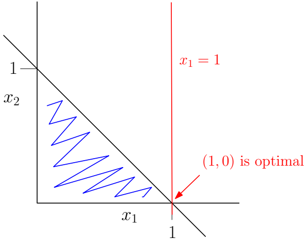
- Move vertical line representing f as far right as possible, *while ensuring* it has non-empty intersection with feasible region.
- Last point of intersection is optimal solution:



Putting it all together

Optimizing our LP corresponds to (in our example):

- Move vertical line representing f as far right as possible, *while ensuring* it has non-empty intersection with feasible region.
- Last point of intersection is optimal solution:



- Sanity check: Convince yourself that $(x_1, x_2) = (1, 0)$ is indeed optimal for:

$$\begin{array}{lrcl} \text{maximize} & x_1 & & \\ \text{subject to} & x_1 + x_2 & \leq & 1 \\ & x_1, x_2 & \geq & 0. \end{array}$$

Boundary case

What if we drop a constraint?

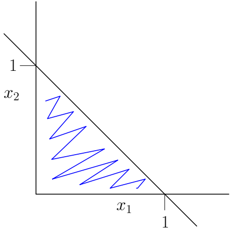
$$\begin{array}{lrcl} \text{maximize} & x_1 & & \\ \text{subject to} & x_1 + x_2 & \leq & 1 \\ & x_1, x_2 & \geq & 0 \end{array}$$

$$\begin{array}{lrcl} \text{maximize} & x_1 & & \\ \text{subject to} & x_1, x_2 & \geq & 0 \end{array}$$

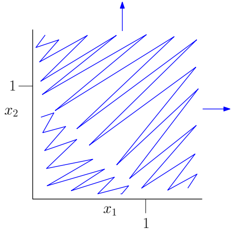
Boundary case

What if we drop a constraint?

$$\begin{array}{lrcl} \text{maximize} & x_1 & & \\ \text{subject to} & x_1 + x_2 & \leq & 1 \\ & x_1, x_2 & \geq & 0 \end{array}$$



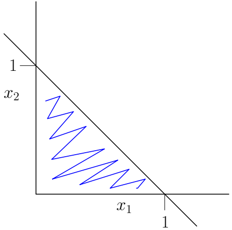
$$\begin{array}{lrcl} \text{maximize} & x_1 & & \\ \text{subject to} & x_1, x_2 & \geq & 0 \end{array}$$



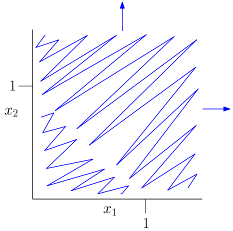
Boundary case

What if we drop a constraint?

$$\begin{array}{lrcl} \text{maximize} & x_1 & & \\ \text{subject to} & x_1 + x_2 & \leq & 1 \\ & x_1, x_2 & \geq & 0 \end{array}$$



$$\begin{array}{lrcl} \text{maximize} & x_1 & & \\ \text{subject to} & x_1, x_2 & \geq & 0 \end{array}$$



- Can move the vertical objective function line as far right as we like!
- Optimal value is ∞, i.e. the LP is *unbounded*.

Outline

1 Definitions

2 Applications

3 Duality theory

4 Solving LPs

Application 1: Shortest paths

- Recall single-source shortest paths problem:
  - Given graph $G = (V, E)$ with real edge costs, and a source vertex $s \in V$, find smallest weight path to all other $v \in V$.
- Solved by Bellman-Ford in $O(mn)$ time.
- Can also be phrased as LP! Let's consider scaled down problem:

Application 1: Shortest paths

- Recall single-source shortest paths problem:
  - Given graph $G = (V, E)$ with real edge costs, and a source vertex $s \in V$, find smallest weight path to all other $v \in V$.
- Solved by Bellman-Ford in $O(mn)$ time.
- Can also be phrased as LP! Let's consider scaled down problem:

Single-*pair* shortest path problem (SPSP)

Given graph $G = (V, E)$ with real edge costs, source and sink vertices $s, t \in V$, respectively, find smallest weight path P from s to t in G.

Application 1: Shortest paths

Single-*pair* shortest path problem (SPSP)

Given graph $G = (V, E)$ with real edge costs, source and sink vertices $s, t \in V$, respectively, find smallest weight path P from s to t in G.

Application 1: Shortest paths

Single-*pair* shortest path problem (SPSP)

Given graph $G = (V, E)$ with real edge costs, source and sink vertices $s, t \in V$, respectively, find smallest weight path P from s to t in G.

Claim: The following LP optimally solves SPSP.

For each $v \in V$, introduce variable d_v.

$$\begin{array}{llcll} \text{maximize} & d_t & & & \\ \text{subject to} & d_v & \leq & d_u + w(u,v) & \forall (u,v) \in E \\ & d_s & = & 0 & \end{array}$$

Application 1: Shortest paths

Single-*pair* shortest path problem (SPSP)

Given graph $G = (V, E)$ with real edge costs, source and sink vertices $s, t \in V$, respectively, find smallest weight path P from s to t in G.

Claim: The following LP optimally solves SPSP.

For each $v \in V$, introduce variable d_v.

$$\begin{array}{lrcll} \text{maximize} & d_t & & & \\ \text{subject to} & d_v & \leq & d_u + w(u,v) & \forall (u,v) \in E \\ & d_s & = & 0 & \end{array}$$

Observations:

Application 1: Shortest paths

Single-*pair* shortest path problem (SPSP)

Given graph $G = (V, E)$ with real edge costs, source and sink vertices $s, t \in V$, respectively, find smallest weight path P from s to t in G.

Claim: The following LP optimally solves SPSP.

For each $v \in V$, introduce variable d_v.

$$\begin{array}{llcll} \text{maximize} & d_t & & & \\ \text{subject to} & d_v & \leq & d_u + w(u,v) & \forall (u,v) \in E \\ & d_s & = & 0 & \end{array}$$

Observations:

- No $d_v \geq 0$ constraints for all $v \in V$?

Application 1: Shortest paths

Single-*pair* shortest path problem (SPSP)

Given graph $G = (V, E)$ with real edge costs, source and sink vertices $s, t \in V$, respectively, find smallest weight path P from s to t in G.

Claim: The following LP optimally solves SPSP.

For each $v \in V$, introduce variable d_v.

$$\begin{array}{lrcll} \text{maximize} & d_t & & & \\ \text{subject to} & d_v & \leq & d_u + w(u,v) & \forall (u,v) \in E \\ & d_s & = & 0 & \end{array}$$

Observations:

- No $d_v \geq 0$ constraints for all $v \in V$?
- Equality constraint (i.e. $d_s = 0$)?

Application 1: Shortest paths

Single-*pair* shortest path problem (SPSP)

Given graph $G = (V, E)$ with real edge costs, source and sink vertices $s, t \in V$, respectively, find smallest weight path P from s to t in G.

Claim: The following LP optimally solves SPSP.

For each $v \in V$, introduce variable d_v.

$$\begin{array}{llll} \text{maximize} & d_t & & \\ \text{subject to} & d_v \leq & d_u + w(u,v) & \forall (u,v) \in E \\ & d_s = & 0 & \end{array}$$

Observations:

- No $d_v \geq 0$ constraints for all $v \in V$?
- Equality constraint (i.e. $d_s = 0$)?
- Why are we *maximizing* d_t?

Application 1: Shortest paths

$$
\begin{array}{lrcl}
\text{maximize} & d_t & & \\
\text{subject to} & d_v & \leq & d_u + w(u,v) \quad \forall (u,v) \in E \\
 & d_s & = & 0
\end{array}
$$

Observation: No $d_v \geq 0$ constraints for all $v \in V$?

Application 1: Shortest paths

$$\begin{array}{llcll} \text{maximize} & d_t & & & \\ \text{subject to} & d_v & \leq & d_u + w(u,v) & \forall (u,v) \in E \\ & d_s & = & 0 & \end{array}$$

Observation: No $d_v \geq 0$ constraints for all $v \in V$?

- Optimal solution might require $d_t < 0$! Can you give an example?

Application 1: Shortest paths

$$
\begin{array}{lllll}
\text{maximize} & d_t & & & \\
\text{subject to} & d_v & \leq & d_u + w(u,v) & \forall (u,v) \in E \\
 & d_s & = & 0 &
\end{array}
$$

Observation: No $d_v \geq 0$ constraints for all $v \in V$?

- Optimal solution might require $d_t < 0$! Can you give an example?
- Ex. G has only one edge, (s,t), of weight $w(s,t) = -1$.
- Recall: Standard form for LPs required variables to be non-negative...

Application 1: Shortest paths

$$\begin{array}{lrcll} \text{maximize} & d_t & & & \\ \text{subject to} & d_v & \leq & d_u + w(u,v) & \forall (u,v) \in E \\ & d_s & = & 0 & \end{array}$$

Observation: No $d_v \geq 0$ constraints for all $v \in V$?

- Optimal solution might require $d_t < 0$! Can you give an example?
- Ex. G has only one edge, (s,t), of weight $w(s,t) = -1$.
- Recall: Standard form for LPs required variables to be non-negative...
- Solution: For all $v \in V$, rewrite $d_v = d_{v_1} - d_{v_2}$ with $d_{v_1}, d_{v_2} \geq 0$.

Application 1: Shortest paths

$$\begin{array}{lrcll} \text{maximize} & d_t & & & \\ \text{subject to} & d_v & \leq & d_u + w(u,v) & \forall (u,v) \in E \\ & d_s & = & 0 & \end{array}$$

Observation: Equality constraint (i.e. $d_s = 0$)?

Application 1: Shortest paths

$$
\begin{array}{lrcl}
\text{maximize} & d_t & & \\
\text{subject to} & d_v & \leq & d_u + w(u,v) \quad \forall (u,v) \in E \\
 & d_s & = & 0
\end{array}
$$

Observation: Equality constraint (i.e. $d_s = 0$)?

- Recall: Standard form for LPs allowed only inequalities...

Application 1: Shortest paths

$$\begin{array}{lrcll} \text{maximize} & d_t & & & \\ \text{subject to} & d_v & \leq & d_u + w(u,v) & \forall (u,v) \in E \\ & d_s & = & 0 & \end{array}$$

Observation: Equality constraint (i.e. $d_s = 0$)?

- Recall: Standard form for LPs allowed only inequalities...
- Solution:
  - Replace $d_s = 0$ with two constraints:
    1. $d_s \geq 0$
    2. $-d_s \geq 0$

Application 1: Shortest paths

$$\begin{array}{llcll} \text{maximize} & d_t & & & \\ \text{subject to} & d_v & \leq & d_u + w(u,v) & \forall (u,v) \in E \quad (**) \\ & d_s & = & 0 & \end{array}$$

Observation: Why are we *maximizing* d_t?

Application 1: Shortest paths

$$\begin{array}{llcll} \text{maximize} & d_t & & & \\ \text{subject to} & d_v & \leq & d_u + w(u,v) & \forall (u,v) \in E \quad (**) \\ & d_s & = & 0 & \end{array}$$

Observation: Why are we *maximizing* d_t?

- LP essentially encodes recursive dynamic programming approach, i.e. cheapest path to t has cost

$$\min_{(u,t)\in E} d_u + w(u,t).$$

Application 1: Shortest paths

$$
\begin{array}{llcll}
\text{maximize} & d_t & & & \\
\text{subject to} & d_v & \leq & d_u + w(u,v) & \forall (u,v) \in E \quad (**) \\
 & d_s & = & 0 &
\end{array}
$$

Observation: Why are we *maximizing* d_t?

- LP essentially encodes recursive dynamic programming approach, i.e. cheapest path to t has cost
$$\min_{(u,t)\in E} d_u + w(u,t).$$
- Inequality (**) uses $\forall$ over neighbors of t to "simulate" $\min_{(u,t)\in E}$.

Application 1: Shortest paths

$$\begin{array}{lrcll} \text{maximize} & d_t & & & \\ \text{subject to} & d_v & \leq & d_u + w(u,v) & \forall (u,v) \in E \quad (**) \\ & d_s & = & 0 & \end{array}$$

Observation: Why are we *maximizing* d_t?

- LP essentially encodes recursive dynamic programming approach, i.e. cheapest path to t has cost
$$\min_{(u,t)\in E} d_u + w(u,t).$$
- Inequality (**) uses $\forall$ over neighbors of t to "simulate" $\min_{(u,t)\in E}$.
- Hence, maximizing d_t ensures optimal solution satisfies (with equality):
$$d_t = \min_{(u,t)\in E} d_u + w(u,t).$$

Application 1: Shortest paths

$$\begin{array}{lrcll} \text{maximize} & d_t & & & \\ \text{subject to} & d_v & \leq & d_u + w(u,v) & \forall (u,v) \in E \quad (**) \\ & d_s & = & 0 & \end{array}$$

Observation: Why are we *maximizing* d_t?

- LP essentially encodes recursive dynamic programming approach, i.e. cheapest path to t has cost
$$\min_{(u,t)\in E} d_u + w(u,t).$$
- Inequality (**) uses $\forall$ over neighbors of t to "simulate" $\min_{(u,t)\in E}$.
- Hence, maximizing d_t ensures optimal solution satisfies (with equality):
$$d_t = \min_{(u,t)\in E} d_u + w(u,t).$$

Application 2: Network Flow

- The Max Flow problem is, *by definition*, an LP!
- Given a flow network (G, s, t, c) for capacity function $c : E \mapsto \mathbb{R}^+$, the following LP yields max flow value:

$$\text{maximize} \quad \sum_{v \in V} f(s, v)$$

Application 2: Network Flow

- The Max Flow problem is, *by definition*, an LP!
- Given a flow network (G, s, t, c) for capacity function $c : E \mapsto \mathbb{R}^+$, the following LP yields max flow value:

$$\begin{array}{llllll} \text{maximize} & \sum_{v \in V} f(s,v) & & & & \\ \text{subject to} & f(u,v) & \leq & c(u,v) & \forall u,v \in V & \text{(capacity constraint)} \end{array}$$

Application 2: Network Flow

- The Max Flow problem is, *by definition*, an LP!
- Given a flow network (G, s, t, c) for capacity function $c : E \mapsto \mathbb{R}^+$, the following LP yields max flow value:

$$\begin{array}{llllll}
\text{maximize} & \sum_{v \in V} f(s, v) & & & & \\
\text{subject to} & f(u, v) & \leq & c(u, v) & \forall u, v \in V & \text{(capacity constraint)} \\
 & f(u, v) & = & -f(v, u) & \forall u, v \in V & \text{(skew symmetry)}
\end{array}$$

Application 2: Network Flow

- The Max Flow problem is, *by definition*, an LP!
- Given a flow network (G, s, t, c) for capacity function $c : E \mapsto \mathbb{R}^+$, the following LP yields max flow value:

$$
\begin{array}{llllll}
\text{maximize} & \sum_{v\in V} f(s,v) & & & & \\
\text{subject to} & f(u,v) & \leq & c(u,v) & \forall u,v \in V & \text{(capacity constraint)} \\
 & f(u,v) & = & -f(v,u) & \forall u,v \in V & \text{(skew symmetry)} \\
 & \sum_{v\in V} f(u,v) & = & 0 & \forall u \in V \setminus \{s,t\} & \text{(flow conservation)}
\end{array}
$$

Application 2: Network Flow

- The Max Flow problem is, *by definition*, an LP!
- Given a flow network (G, s, t, c) for capacity function $c : E \mapsto \mathbb{R}^+$, the following LP yields max flow value:

$$
\begin{array}{llcll}
\text{maximize} & \sum_{v\in V} f(s,v) & & & \\
\text{subject to} & f(u,v) & \leq & c(u,v) & \forall u,v \in V & \text{(capacity constraint)} \\
 & f(u,v) & = & -f(v,u) & \forall u,v \in V & \text{(skew symmetry)} \\
 & \sum_{v\in V} f(u,v) & = & 0 & \forall u \in V \setminus \{s,t\} & \text{(flow conservation)}
\end{array}
$$

Application 3: Multi-commodity flow (MCF)

- Like Max Flow, except instead of 1 commodity to route through network (e.g. water), have k commodities which share the network.
- Like Max Flow, given $G = (V, E)$ and capacities $c(u, v) \geq 0$.

Application 3: Multi-commodity flow (MCF)

- Like Max Flow, except instead of 1 commodity to route through network (e.g. water), have k commodities which share the network.
- Like Max Flow, given $G = (V, E)$ and capacities $c(u, v) \geq 0$.
- Unlike Max Flow, each commodity K_i specified via $K_i = (s_i, t_i, d_i)$:
  - s_i and t_i are source/sink for K_i, respectively.
  - d_i is the total *demand* for K_i which must be met, i.e.

$$\sum_{v \in V} f_i(s_i, v) = d_i \text{ for all } i \in [k].$$

Application 3: Multi-commodity flow (MCF)

- Like Max Flow, except instead of 1 commodity to route through network (e.g. water), have k commodities which share the network.
- Like Max Flow, given $G = (V, E)$ and capacities $c(u, v) \geq 0$.
- Unlike Max Flow, each commodity K_i specified via $K_i = (s_i, t_i, d_i)$:
  - s_i and t_i are source/sink for K_i, respectively.
  - d_i is the total *demand* for K_i which must be met, i.e.

$$\sum_{v \in V} f_i(s_i, v) = d_i \text{ for all } i \in [k].$$

  - Above, f_i is flow for commodity i, so that aggregate flow f satisfies

$$f(u, v) = \sum_{i=1}^{k} f_i(u, v).$$

Application 3: Multi-commodity flow (MCF)

- Like Max Flow, except instead of 1 commodity to route through network (e.g. water), have k commodities which share the network.
- Like Max Flow, given $G = (V, E)$ and capacities $c(u, v) \geq 0$.
- Unlike Max Flow, each commodity K_i specified via $K_i = (s_i, t_i, d_i)$:
  - s_i and t_i are source/sink for K_i, respectively.
  - d_i is the total *demand* for K_i which must be met, i.e.

$$\sum_{v \in V} f_i(s_i, v) = d_i \text{ for all } i \in [k].$$

  - Above, f_i is flow for commodity i, so that aggregate flow f satisfies

$$f(u, v) = \sum_{i=1}^{k} f_i(u, v).$$

- Q: Possible to route all k commodities through network, while meeting demand constraints but (2) not violating capacity constraints?

Application 3: Multi-commodity flow (MCF)

Q: Can you guess the LP for MCF?

Application 3: Multi-commodity flow (MCF)

Q: Can you guess the LP for MCF?

$$\text{maximize} \qquad ?$$

Application 3: Multi-commodity flow (MCF)

Q: Can you guess the LP for MCF?

$$
\begin{array}{llcll}
\text{maximize} & ? & & & \\
\text{subject to} & \sum_{i=1}^{k} f_i(u,v) & \leq & c(u,v) & \forall u,v \in V \qquad \text{(capacity)}
\end{array}
$$

Application 3: Multi-commodity flow (MCF)

Q: Can you guess the LP for MCF?

$$
\begin{array}{llcll}
\text{maximize} & ? & & & \\
\text{subject to} & \sum_{i=1}^{k} f_i(u,v) & \leq & c(u,v) & \forall u,v \in V & \text{(capacity)} \\
 & f_i(u,v) & = & -f_i(v,u) & \forall i \in [k], u,v \in V & \text{(skew symmetry)}
\end{array}
$$

Application 3: Multi-commodity flow (MCF)

Q: Can you guess the LP for MCF?

$$
\begin{array}{llcll}
\text{maximize} & ? & & & \\
\text{subject to} & \sum_{i=1}^{k} f_i(u,v) & \leq & c(u,v) & \forall u,v \in V & \text{(capacity)} \\
& f_i(u,v) & = & -f_i(v,u) & \forall i \in [k], u,v \in V & \text{(skew symmetry)} \\
& \sum_{v \in V} f_i(u,v) & = & 0 & \forall i \in [k], u \in V \setminus \{s,t\} & \text{(flow conservation)}
\end{array}
$$

Application 3: Multi-commodity flow (MCF)

Q: Can you guess the LP for MCF?

$$
\begin{array}{llcll}
\text{maximize} & ? & & & \\
\text{subject to} & \sum_{i=1}^{k} f_i(u,v) & \leq & c(u,v) & \forall u,v \in V & \text{(capacity)} \\
& f_i(u,v) & = & -f_i(v,u) & \forall i \in [k], u,v \in V & \text{(skew symmetry)} \\
& \sum_{v\in V} f_i(u,v) & = & 0 & \forall i \in [k], u \in V \setminus \{s,t\} & \text{(flow conservation)} \\
& \sum_{v\in V} f_i(s_i,v) & = & d_i & \forall i \in [k] & \text{(demand)}
\end{array}
$$

Application 3: Multi-commodity flow (MCF)

Q: Can you guess the LP for MCF?

$$
\begin{array}{llcllr}
\text{maximize} & ? & & & & \\
\text{subject to} & \sum_{i=1}^{k} f_i(u,v) & \leq & c(u,v) & \forall u,v \in V & \text{(capacity)} \\
 & f_i(u,v) & = & -f_i(v,u) & \forall i \in [k], u,v \in V & \text{(skew symmetry)} \\
 & \sum_{v \in V} f_i(u,v) & = & 0 & \forall i \in [k], u \in V \setminus \{s,t\} & \text{(flow conservation)} \\
 & \sum_{v \in V} f_i(s_i,v) & = & d_i & \forall i \in [k] & \text{(demand)}
\end{array}
$$

Q: What about objective function?

- Recall defined MCF as decision problem (answer is YES or NO).
- (Recall: Possible to route all k commodities through network, while meeting demand constraints but (2) not violating capacity constraints?)

Application 3: Multi-commodity flow (MCF)

Q: Can you guess the LP for MCF?

$$
\begin{array}{llcll}
\text{maximize} & ? & & & \\
\text{subject to} & \sum_{i=1}^{k} f_i(u,v) \leq c(u,v) & & \forall u,v \in V & \text{(capacity)} \\
& f_i(u,v) = -f_i(v,u) & & \forall i \in [k], u,v \in V & \text{(skew symmetry)} \\
& \sum_{v \in V} f_i(u,v) = 0 & & \forall i \in [k], u \in V \setminus \{s,t\} & \text{(flow conservation)} \\
& \sum_{v \in V} f_i(s_i,v) = d_i & & \forall i \in [k] & \text{(demand)}
\end{array}
$$

Q: What about objective function?

- Recall defined MCF as decision problem (answer is YES or NO).
- (Recall: Possible to route all k commodities through network, while meeting demand constraints but (2) not violating capacity constraints?)
- Once we have flows f_i satisfying all constraints, we know the answer is YES. Hence, we don't "need" objective function.
- Geometrically, MCF only asks if LP feasible region is *non-empty*.

Application 3: Multi-commodity flow (MCF)

Q: Can you guess the LP for MCF?

$$
\begin{array}{llcll}
\text{maximize} & ? & & & \\
\text{subject to} & \sum_{i=1}^{k} f_i(u,v) & \leq & c(u,v) & \forall u,v \in V & \text{(capacity)} \\
& f_i(u,v) & = & -f_i(v,u) & \forall i \in [k], u,v \in V & \text{(skew symmetry)} \\
& \sum_{v \in V} f_i(u,v) & = & 0 & \forall i \in [k], u \in V \setminus \{s,t\} & \text{(flow conservation)} \\
& \sum_{v \in V} f_i(s_i,v) & = & d_i & \forall i \in [k] & \text{(demand)}
\end{array}
$$

Q: What about objective function?

- Recall defined MCF as decision problem (answer is YES or NO).
- (Recall: Possible to route all k commodities through network, while meeting demand constraints but (2) not violating capacity constraints?)
- Once we have flows f_i satisfying all constraints, we know the answer is YES. Hence, we don't "need" objective function.
- Geometrically, MCF only asks if LP feasible region is *non-empty*.
- Hence, can set objective function to 0.

Application 3: Multi-commodity flow

Final LP:

$$
\begin{array}{llcllr}
\text{maximize} & 0 & & & & \\
\text{subject to} & \sum_{i=1}^{k} f_i(u,v) & \leq & c(u,v) & \forall u,v \in V & \text{(capacity)} \\
 & f_i(u,v) & = & -f_i(v,u) & \forall i \in [k], u,v \in V & \text{(skew symmetry)} \\
 & \sum_{v \in V} f_i(u,v) & = & 0 & \forall i \in [k], u \in V \setminus \{s,t\} & \text{(flow conservation)} \\
 & \sum_{v \in V} f_i(s_i,v) & = & d_i & \forall i \in [k] & \text{(demand)}
\end{array}
$$

Application 3: Multi-commodity flow

Final LP:

$$
\begin{array}{llllll}
\text{maximize} & 0 & & & & \\
\text{subject to} & \sum_{i=1}^{k} f_i(u,v) & \leq & c(u,v) & \forall u,v \in V & \text{(capacity)} \\
 & f_i(u,v) & = & -f_i(v,u) & \forall i \in [k], u,v \in V & \text{(skew symmetry)} \\
 & \sum_{v \in V} f_i(u,v) & = & 0 & \forall i \in [k], u \in V \setminus \{s,t\} & \text{(flow conservation)} \\
 & \sum_{v \in V} f_i(s_i,v) & = & d_i & \forall i \in [k] & \text{(demand)}
\end{array}
$$

- The only polynomial-time algorithm know for MCF is via LPs.
- In this sense, LPs “seem” strictly more powerful than network flow algorithms.

Application 3: Multi-commodity flow

Final LP:

$$
\begin{array}{lrcll}
\text{maximize} & 0 & & & \\
\text{subject to} & \sum_{i=1}^{k} f_i(u,v) & \leq & c(u,v) & \forall u,v \in V & \text{(capacity)} \\
& f_i(u,v) & = & -f_i(v,u) & \forall i \in [k], u,v \in V & \text{(skew symmetry)} \\
& \sum_{v \in V} f_i(u,v) & = & 0 & \forall i \in [k], u \in V \setminus \{s,t\} & \text{(flow conservation)} \\
& \sum_{v \in V} f_i(s_i,v) & = & d_i & \forall i \in [k] & \text{(demand)}
\end{array}
$$

- The only polynomial-time algorithm know for MCF is via LPs.
- In this sense, LPs “seem” strictly more powerful than network flow algorithms.
- Indeed, linear programming is P-complete! (Roughly, any algorithm in P can be reduced to solving an LP.)

Application 3: Multi-commodity flow

Final LP:

$$
\begin{array}{llcll}
\text{maximize} & 0 & & & \\
\text{subject to} & \sum_{i=1}^{k} f_i(u,v) & \leq & c(u,v) & \forall u,v \in V & \text{(capacity)} \\
& f_i(u,v) & = & -f_i(v,u) & \forall i \in [k], u,v \in V & \text{(skew symmetry)} \\
& \sum_{v \in V} f_i(u,v) & = & 0 & \forall i \in [k], u \in V \setminus \{s,t\} & \text{(flow conservation)} \\
& \sum_{v \in V} f_i(s_i,v) & = & d_i & \forall i \in [k] & \text{(demand)}
\end{array}
$$

- The only polynomial-time algorithm know for MCF is via LPs.
- In this sense, LPs "seem" strictly more powerful than network flow algorithms.
- Indeed, linear programming is P-complete! (Roughly, any algorithm in P can be reduced to solving an LP.)

- If we demand *integer* flow, i.e. $f_i(u,v) \in \mathbb{Z}$, then MCF becomes NP-complete.

But there is more black magic to come...

Outline

1. Definitions
2. Applications
3. Duality theory
4. Solving LPs

I haven't told you yet how to actually *solve* an LP.

- But do we need to actually solve the LP to *provably* get the optimal solution?

I haven't told you yet how to actually *solve* an LP.

- But do we need to actually solve the LP to *provably* get the optimal solution?
- Remarkably, no... Can:
  - *Guess* a solution $x = \{x_i\}$.
  - If x is optimal, can use *duality theory* to prove this.

I haven't told you yet how to actually *solve* an LP.

- But do we need to actually solve the LP to *provably* get the optimal solution?
- Remarkably, no. . . Can:
  - *Guess* a solution $x = \{x_i\}$.
  - If x is optimal, can use *duality theory* to prove this.
- Yields powerful method for proving analytic bounds on optimization problems in math proofs. (Where in this course have we used this idea, at least indirectly?)

Intuition

Let's return to our LP example, slightly rewritten below:

$$
\begin{aligned}
\text{maximize} \quad & x_1 & & & (1)\\
\text{subject to} \quad & x_1 + x_2 & \leq & \; 1 & (2)\\
& -x_1 & \leq & \; 0 & (3)\\
& -x_2 & \leq & \; 0 & (4)
\end{aligned}
$$

- Recall: Optimal solution was $(x_1, x_2) = (1, 0)$, with value 1.
- Claim: Can *prove* no solution can do better.

Intuition

Let's return to our LP example, slightly rewritten below:

$$
\begin{array}{lrcll}
\text{maximize} & x_1 & & & (1)\\
\text{subject to} & x_1 + x_2 & \leq & 1 & (2)\\
 & -x_1 & \leq & 0 & (3)\\
 & -x_2 & \leq & 0 & (4)
\end{array}
$$

- Recall: Optimal solution was $(x_1, x_2) = (1, 0)$, with value 1.
- Claim: Can *prove* no solution can do better.
  - Adding inequalities (2) and (4) yields bound $x_1 \leq 1$.
  - Thus, objective function upper bounded by 1, and $(1, 0)$ is indeed optimal.

Intuition

Let's return to our LP example, slightly rewritten below:

$$
\begin{array}{llcll}
\text{maximize} & x_1 & & & (1)\\
\text{subject to} & x_1 + x_2 & \leq & 1 & (2)\\
 & -x_1 & \leq & 0 & (3)\\
 & -x_2 & \leq & 0 & (4)
\end{array}
$$

- Recall: Optimal solution was $(x_1, x_2) = (1, 0)$, with value 1.
- Claim: Can *prove* no solution can do better.
  - Adding inequalities (2) and (4) yields bound $x_1 \leq 1$.
  - Thus, objective function upper bounded by 1, and $(1, 0)$ is indeed optimal.
- Even better: Can generalize this idea to get tight upper bound on optimal value.

Intuition

Let's return to our LP example, slightly rewritten below:

$$
\begin{array}{lrcll}
\text{maximize} & x_1 & & & (1)\\
\text{subject to} & x_1 + x_2 & \leq & 1 & (2)\\
 & -x_1 & \leq & 0 & (3)\\
 & -x_2 & \leq & 0 & (4)
\end{array}
$$

- Recall: Optimal solution was $(x_1, x_2) = (1, 0)$, with value 1.
- Claim: Can *prove* no solution can do better.
  - Adding inequalities (2) and (4) yields bound $x_1 \leq 1$.
  - Thus, objective function upper bounded by 1, and $(1, 0)$ is indeed optimal.
- Even better: Can generalize this idea to get tight upper bound on optimal value.
- Idea:
  - To each constraint, assign a "dual" variable y_i.
  - "Do a minimization" over linear combinations of y_i to get upper bound on objective function.
  - This minimization *itself* an LP, called *dual LP*.

I did say there was more black magic to come, no?

Primal-dual pair

LPs come in pairs, known as the primal (left) and dual (right) LP:

$$
\begin{array}{llll}
\max & \sum_{j=1}^{n} c_j x_j & & \\
\text{s.t.} & \sum_{j=1}^{n} a_{ij} x_j \leq & b_i & \forall i \in [m] \\
& x_j \geq & 0 & \forall j \in [n]
\end{array}
\qquad\qquad
\begin{array}{llll}
\min & \sum_{i=1}^{m} b_i y_i & & \\
\text{s.t.} & \sum_{i=1}^{m} a_{ij} y_i \geq & c_j & \forall j \in [n] \\
& y_i \geq & 0 & \forall j \in [m]
\end{array}
$$

Primal-dual pair

LPs come in pairs, known as the primal (left) and dual (right) LP:

$$
\begin{array}{llcll}
\max & \sum_{j=1}^{n} c_j x_j & & & \\
\text{s.t.} & \sum_{j=1}^{n} a_{ij} x_j & \leq & b_i & \forall i \in [m] \\
 & x_j & \geq & 0 & \forall j \in [n]
\end{array}
\qquad
\begin{array}{llcll}
\min & \sum_{i=1}^{m} b_i y_i & & & \\
\text{s.t.} & \sum_{i=1}^{m} a_{ij} y_i & \geq & c_j & \forall j \in [n] \\
 & y_i & \geq & 0 & \forall j \in [m]
\end{array}
$$

Our example:

$$
\begin{array}{lrcl}
\max & x_1 & & \\
\text{s.t.} & x_1 + x_2 & \leq & 1 \\
 & -x_1 & \leq & 0 \\
 & -x_2 & \leq & 0
\end{array}
$$

Primal-dual pair

LPs come in pairs, known as the primal (left) and dual (right) LP:

$$\begin{array}{llll} \max & \sum_{j=1}^{n} c_j x_j & & \\ \text{s.t.} & \sum_{j=1}^{n} a_{ij} x_j \leq b_i & \forall i \in [m] \\ & x_j \geq 0 & \forall j \in [n] \end{array} \qquad \begin{array}{llll} \min & \sum_{i=1}^{m} b_i y_i & & \\ \text{s.t.} & \sum_{i=1}^{m} a_{ij} y_i \geq c_j & \forall j \in [n] \\ & y_i \geq 0 & \forall j \in [m] \end{array}$$

Our example:

$$\begin{array}{lrcl} \max & x_1 & & \\ \text{s.t.} & x_1 + x_2 & \leq & 1 \\ & -x_1 & \leq & 0 \\ & -x_2 & \leq & 0 \end{array} \qquad \begin{array}{lrcl} \min & y_1 & & \\ \text{s.t.} & y_1 - y_2 & \geq & 1 \\ & y_1 - y_3 & \geq & 0 \\ & y_1, y_2, y_3 & \geq & 0 \end{array}$$

Primal-dual pair

LPs come in pairs, known as the primal (left) and dual (right) LP:

$$\begin{array}{lrcll} \max & \sum_{j=1}^{n} c_j x_j & & & \\ \text{s.t.} & \sum_{j=1}^{n} a_{ij} x_j & \leq & b_i & \forall i \in [m] \\ & x_j & \geq & 0 & \forall j \in [n] \end{array} \qquad \begin{array}{lrcll} \min & \sum_{i=1}^{m} b_i y_i & & & \\ \text{s.t.} & \sum_{i=1}^{m} a_{ij} y_i & \geq & c_j & \forall j \in [n] \\ & y_i & \geq & 0 & \forall j \in [m] \end{array}$$

Our example:

$$\begin{array}{lrcl} \max & x_1 & & \\ \text{s.t.} & x_1 + x_2 & \leq & 1 \\ & -x_1 & \leq & 0 \\ & -x_2 & \leq & 0 \end{array} \qquad \begin{array}{lrcl} \min & y_1 & & \\ \text{s.t.} & y_1 - y_2 & \geq & 1 \\ & y_1 - y_3 & \geq & 0 \\ & y_1, y_2, y_3 & \geq & 0 \end{array}$$

Q: Can you give dual solution with dual objective function value 1?

Formalization primal vs dual

$$\begin{array}{lrcll} \max & \sum_{j=1}^{n} c_j x_j & & & \\ \text{s.t.} & \sum_{j=1}^{n} a_{ij} x_j & \leq & b_i & \forall i \in [m] \\ & x_j & \geq & 0 & \forall j \in [n] \end{array}$$

$$\begin{array}{lrcll} \min & \sum_{i=1}^{m} b_i y_i & & & \\ \text{s.t.} & \sum_{i=1}^{m} a_{ij} y_i & \geq & c_j & \forall j \in [n] \\ & y_i & \geq & 0 & \forall j \in [m] \end{array}$$

Formalization primal vs dual

$$\begin{array}{llcll} \max & \sum_{j=1}^{n} c_j x_j & & & \\ \text{s.t.} & \sum_{j=1}^{n} a_{ij} x_j & \leq & b_i & \forall i \in [m] \\ & x_j & \geq & 0 & \forall j \in [n] \end{array} \qquad \begin{array}{llcll} \min & \sum_{i=1}^{m} b_i y_i & & & \\ \text{s.t.} & \sum_{i=1}^{m} a_{ij} y_i & \geq & c_j & \forall j \in [n] \\ & y_i & \geq & 0 & \forall j \in [m] \end{array}$$

Let P and D denote primal and dual LP, respectively.

Let p^* and d^* denote optimal solutions for P and D, respectively.

Intuitively: Designed D so that d^* yields upper bound on p^*. Let's prove this!

Formalization primal vs dual

$$\begin{array}{llcll} \max & \sum_{j=1}^n c_j x_j & & & \\ \text{s.t.} & \sum_{j=1}^n a_{ij} x_j & \leq & b_i & \forall i \in [m] \\ & x_j & \geq & 0 & \forall j \in [n] \end{array} \qquad \begin{array}{llcll} \min & \sum_{i=1}^m b_i y_i & & & \\ \text{s.t.} & \sum_{i=1}^m a_{ij} y_i & \geq & c_j & \forall j \in [n] \\ & y_i & \geq & 0 & \forall j \in [m] \end{array}$$

Let P and D denote primal and dual LP, respectively.

Let p^* and d^* denote optimal solutions for P and D, respectively.

Intuitively: Designed D so that d^* yields upper bound on p^*. Let's prove this!

Theorem (Weak duality)

For any primal feasible $x = \{x_j\}$ and dual feasible $y = \{y_i\}$,

$$\sum_{j=1}^n c_j x_j \leq \sum_{i=1}^m b_i y_i.$$

Formalization primal vs dual

$$\begin{array}{llcll} \max & \sum_{j=1}^{n} c_j x_j & & & \\ \text{s.t.} & \sum_{j=1}^{n} a_{ij} x_j & \leq & b_i & \forall i \in [m] \\ & x_j & \geq & 0 & \forall j \in [n] \end{array} \qquad \begin{array}{llcll} \min & \sum_{i=1}^{m} b_i y_i & & & \\ \text{s.t.} & \sum_{i=1}^{m} a_{ij} y_i & \geq & c_j & \forall j \in [n] \\ & y_i & \geq & 0 & \forall j \in [m] \end{array}$$

Theorem (Weak duality)

For any primal feasible $x = \{x_j\}$ and dual feasible $y = \{y_i\}$,

$$\sum_{j=1}^{n} c_j x_j \leq \sum_{i=1}^{m} b_i y_i.$$

Formalization primal vs dual

$$\begin{array}{llcll} \max & \sum_{j=1}^{n} c_j x_j & & & \\ \text{s.t.} & \sum_{j=1}^{n} a_{ij} x_j & \leq & b_i & \forall i \in [m] \\ & x_j & \geq & 0 & \forall j \in [n] \end{array}$$

$$\begin{array}{llcll} \min & \sum_{i=1}^{m} b_i y_i & & & \\ \text{s.t.} & \sum_{i=1}^{m} a_{ij} y_i & \geq & c_j & \forall j \in [n] \\ & y_i & \geq & 0 & \forall j \in [m] \end{array}$$

Theorem (Weak duality)

For any primal feasible $x = \{x_j\}$ and dual feasible $y = \{y_i\}$,

$$\sum_{j=1}^{n} c_j x_j \leq \sum_{i=1}^{m} b_i y_i.$$

Proof.

$$\sum_{j=1}^{n} c_j x_j \leq \sum_{j=1}^{n} \left(\sum_{i=1}^{m} a_{ij} y_i \right) x_j = \sum_{i=1}^{m} y_i \left(\sum_{j=1}^{n} a_{ij} x_j \right) \leq \sum_{i=1}^{m} b_i y_i.$$

□

Formalization primal vs dual

Theorem (Weak duality)

For any primal feasible $x = \{x_j\}$ *and dual feasible* $y = \{y_i\}$,

$$\sum_{j=1}^{n} c_j x_j \leq \sum_{i=1}^{m} b_i y_i.$$

Corollary

For any primal and dual LPs P and D, respectively, $p^* \leq d^*$.

Formalization primal vs dual

Theorem (Weak duality)

For any primal feasible $x = \{x_j\}$ and dual feasible $y = \{y_i\}$,

$$\sum_{j=1}^{n} c_j x_j \leq \sum_{i=1}^{m} b_i y_i .$$

Corollary

For any primal and dual LPs P and D, respectively, $p^* \leq d^*$.

Corollary

If you can guess primal solution x and dual solution y with matching objective function values $p = d$, then guaranteed x is optimal! (No need to explicitly solve either LP.)

- Q: Is it always true that $p^* = d^*$?

Formalization primal vs dual

Theorem (Weak duality)

For any primal feasible $x = \{x_j\}$ and dual feasible $y = \{y_i\}$,

$$\sum_{j=1}^{n} c_j x_j \leq \sum_{i=1}^{m} b_i y_i.$$

Corollary

For any primal and dual LPs P and D, respectively, $p^* \leq d^*$.

Corollary

If you can guess primal solution x and dual solution y with matching objective function values $p = d$, then guaranteed x is optimal! (No need to explicitly solve either LP.)

- Q: Is it always true that $p^* = d^*$? Yes! Called *strong duality*.

Returning to Max Flow

Primal LP:

$$
\begin{array}{lrcll}
\text{maximize} & \sum_{v \in V} f(s,v) & & & \\
\text{subject to} & f(u,v) & \leq & c(u,v) & \forall u,v \in V & \text{(capacity constraint)} \\
& f(u,v) & = & -f(v,u) & \forall u,v \in V & \text{(skew symmetry)} \\
& \sum_{v \in V} f(u,v) & = & 0 & \forall u \in V \setminus \{s,t\} & \text{(flow conservation)}
\end{array}
$$

Recall: Max flow is bounded by min capacity across any $s-t$ cut in G.

- Claim: This is precisely what the *dual* LP for Max Flow says.

Returning to Max Flow

Primal LP:

$$
\begin{array}{lrcll}
\text{maximize} & \sum_{v\in V} f(s,v) & & & \\
\text{subject to} & f(u,v) & \leq & c(u,v) & \forall u,v\in V & \text{(capacity constraint)}\\
& f(u,v) & = & -f(v,u) & \forall u,v\in V & \text{(skew symmetry)}\\
& \sum_{v\in V} f(u,v) & = & 0 & \forall u\in V\setminus\{s,t\} & \text{(flow conservation)}
\end{array}
$$

Recall: Max flow is bounded by min capacity across any $s-t$ cut in G.

- Claim: This is precisely what the *dual* LP for Max Flow says.
- Unfortunately, dual of our current primal LP is messy.
- Idea: First rewrite LP in an equivalent, but “simpler” way, bringing us into standard form.

Rewriting the primal LP

Primal LP:

$$
\begin{array}{lrcll}
\text{maximize} & \sum_{v\in V} f(s,v) & & & \\
\text{subject to} & f(u,v) & \leq & c(u,v) & \forall u,v\in V & \text{(capacity constraint)}\\
 & f(u,v) & = & -f(v,u) & \forall u,v\in V & \text{(skew symmetry)}\\
 & \sum_{v\in V} f(u,v) & = & 0 & \forall u\in V\setminus\{s,t\} & \text{(flow conservation)}
\end{array}
$$

Rewriting the primal LP

Primal LP:

$$
\begin{array}{llllll}
\text{maximize} & \sum_{v\in V} f(s,v) & & & & \\
\text{subject to} & f(u,v) & \leq & c(u,v) & \forall u,v \in V & \text{(capacity constraint)} \\
 & f(u,v) & = & -f(v,u) & \forall u,v \in V & \text{(skew symmetry)} \\
 & \sum_{v\in V} f(u,v) & = & 0 & \forall u \in V \setminus \{s,t\} & \text{(flow conservation)}
\end{array}
$$

Equivalent LP: Let Ω denote the set of all simple paths from s to t in G.

$$
\begin{array}{lllll}
\text{maximize} & \sum_{p\in\Omega} x_p & & & \\
\text{subject to} & & & & \\
 & \sum_{p\in\Omega \text{ containing } (u,v)} x_p & \leq & c(u,v) & \forall (u,v) \in E \\
 & x_p & \geq & 0 & \forall p \in \Omega
\end{array}
$$

Rewriting the primal LP

Primal LP:

$$\begin{array}{lrcll} \text{maximize} & \sum_{v\in V} f(s,v) & & & & \\ \text{subject to} & f(u,v) & \leq & c(u,v) & \forall u,v\in V & \text{(capacity constraint)} \\ & f(u,v) & = & -f(v,u) & \forall u,v\in V & \text{(skew symmetry)} \\ & \sum_{v\in V} f(u,v) & = & 0 & \forall u\in V\setminus\{s,t\} & \text{(flow conservation)} \end{array}$$

Equivalent LP: Let Ω denote the set of all simple paths from s to t in G.

$$\begin{array}{lrcll} \text{maximize} & \sum_{p\in\Omega} x_p & & & \\ \text{subject to} & & & & \\ & \sum_{p\in\Omega \text{ containing } (u,v)} x_p & \leq & c(u,v) & \forall (u,v)\in E \\ & x_p & \geq & 0 & \forall p\in\Omega \end{array}$$

- In this new view, each x_p denotes flow along path p.
- Clearly, such flow along any p is limited by the bottleneck edge (u,v) of p.
- Taking dual of this new LP will yield much nicer dual.

Sanity check

Primal LP:

$$\begin{array}{llllll}
\text{maximize} & \sum_{v\in V} f(s,v) & & & & \\
\text{subject to} & f(u,v) & \leq & c(u,v) & \forall u,v\in V & \text{(capacity constraint)} \\
& f(u,v) & = & -f(v,u) & \forall u,v\in V & \text{(skew symmetry)} \\
& \sum_{v\in V} f(u,v) & = & 0 & \forall u\in V\setminus\{s,t\} & \text{(flow conservation)}
\end{array}$$

Q: How many constraints are in the LP above?

Equivalent LP: Let Ω denote the set of all simple paths from s to t in G.

$$\begin{array}{lllll}
\text{maximize} & \sum_{p\in\Omega} x_p & & & \\
\text{subject to} & & & & \\
& \sum_{p\in\Omega \text{ containing } (u,v)} x_p & \leq & c(u,v) & \forall (u,v)\in E \\
& x_p & \geq & 0 & \forall p\in\Omega
\end{array}$$

Q: How many constraints are in the LP above?

Sanity check

Primal LP:

$$\begin{array}{llcllr}
\text{maximize} & \sum_{v\in V} f(s,v) & & & & \\
\text{subject to} & f(u,v) & \leq & c(u,v) & \forall u,v\in V & \text{(capacity constraint)}\\
 & f(u,v) & = & -f(v,u) & \forall u,v\in V & \text{(skew symmetry)}\\
 & \sum_{v\in V} f(u,v) & = & 0 & \forall u\in V\setminus\{s,t\} & \text{(flow conservation)}
\end{array}$$

Q: How many constraints are in the LP above?

Equivalent LP: Let Ω denote the set of all simple paths from s to t in G.

$$\begin{array}{llcll}
\text{maximize} & \sum_{p\in\Omega} x_p & & & \\
\text{subject to} & & & & \\
 & \sum_{p\in\Omega \text{ containing } (u,v)} x_p & \leq & c(u,v) & \forall (u,v)\in E\\
 & x_p & \geq & 0 & \forall p\in\Omega
\end{array}$$

Q: How many constraints are in the LP above?

- A: In the worst case, exponential in n. (Hint: Consider two binary trees glued together at leaves.)
- Does it matter that our new formulation is too big to write down?

Sanity check

Primal LP:

$$\begin{array}{llcll} \text{maximize} & \sum_{v\in V} f(s,v) & & & \\ \text{subject to} & f(u,v) & \leq & c(u,v) & \forall u,v\in V \quad \text{(capacity constraint)} \\ & f(u,v) & = & -f(v,u) & \forall u,v\in V \quad \text{(skew symmetry)} \\ & \sum_{v\in V} f(u,v) & = & 0 & \forall u\in V\setminus\{s,t\} \quad \text{(flow conservation)} \end{array}$$

Q: How many constraints are in the LP above?

Equivalent LP: Let Ω denote the set of all simple paths from s to t in G.

$$\begin{array}{llcll} \text{maximize} & \sum_{p\in\Omega} x_p & & & \\ \text{subject to} & & & & \\ & \sum_{p\in\Omega \text{ containing } (u,v)} x_p & \leq & c(u,v) & \forall (u,v)\in E \\ & x_p & \geq & 0 & \forall p\in\Omega \end{array}$$

Q: How many constraints are in the LP above?

- A: In the worst case, exponential in n. (Hint: Consider two binary trees glued together at leaves.)
- Does it matter that our new formulation is too big to write down?
- Yes, if you plan to solve the LP in practice via a solver.

Sanity check

Primal LP:

$$\begin{array}{llcccl}
\text{maximize} & \sum_{v\in V} f(s,v) & & & & \\
\text{subject to} & f(u,v) & \leq & c(u,v) & \forall u,v\in V & \text{(capacity constraint)}\\
& f(u,v) & = & -f(v,u) & \forall u,v\in V & \text{(skew symmetry)}\\
& \sum_{v\in V} f(u,v) & = & 0 & \forall u\in V\setminus\{s,t\} & \text{(flow conservation)}
\end{array}$$

Q: How many constraints are in the LP above?

Equivalent LP: Let Ω denote the set of all simple paths from s to t in G.

$$\begin{array}{llcccl}
\text{maximize} & \sum_{p\in\Omega} x_p & & & \\
\text{subject to} & & & & \\
& \sum_{p\in\Omega \text{ containing } (u,v)} x_p & \leq & c(u,v) & \forall (u,v)\in E\\
& x_p & \geq & 0 & \forall p\in\Omega
\end{array}$$

Q: How many constraints are in the LP above?

- A: In the worst case, exponential in n. (Hint: Consider two binary trees glued together at leaves.)
- Does it matter that our new formulation is too big to write down?
- Yes, if you plan to solve the LP in practice via a solver.
- No, if all you want to do is look at the dual to extract theoretical bounds on primal value. (Our goal.)

Dual LP for Network Flow

Equivalent LP: Let Ω denote the set of all simple paths from s to t in G.

$$
\begin{array}{llll}
\text{maximize} & \sum_{p\in\Omega} x_p & & \\
\text{subject to} & & & \\
& \sum_{p\in\Omega \text{ containing } (u,v)} x_p \leq & c(u,v) & \forall (u,v)\in E \\
& x_p \geq & 0 & \forall p\in\Omega
\end{array}
$$

Dual LP for Network Flow

Equivalent LP: Let Ω denote the set of all simple paths from s to t in G.

$$\begin{array}{lrcll} \text{maximize} & \sum_{p\in\Omega} x_p & & & \\ \text{subject to} & & & & \\ & \sum_{p\in\Omega \text{ containing } (u,v)} x_p & \leq & c(u,v) & \forall (u,v)\in E \\ & x_p & \geq & 0 & \forall p \in \Omega \end{array}$$

Dual LP: For each $(u,v)\in E$, add dual variable y_{uv}.

$$\begin{array}{lrcll} \text{minimize} & \sum_{(u,v)\in E} c(u,v) y_{uv} & & & \\ \text{subject to} & & & & \\ & \sum_{(u,v)\in p} y_{uv} & \geq & 1 & \forall p\in\Omega \quad (*) \\ & y_{uv} & \geq & 0 & \forall (u,v)\in E \end{array}$$

Dual LP for Network Flow

Equivalent LP: Let Ω denote the set of all simple paths from s to t in G.

$$\begin{array}{lrcll} \text{maximize} & \sum_{p\in\Omega} x_p & & & \\ \text{subject to} & & & & \\ & \sum_{p\in\Omega \text{ containing } (u,v)} x_p & \leq & c(u,v) & \forall (u,v)\in E \\ & x_p & \geq & 0 & \forall p \in \Omega \end{array}$$

Dual LP: For each $(u,v)\in E$, add dual variable y_{uv}.

$$\begin{array}{lrcll} \text{minimize} & \sum_{(u,v)\in E} c(u,v) y_{uv} & & & \\ \text{subject to} & & & & \\ & \sum_{(u,v)\in p} y_{uv} & \geq & 1 & \forall p\in\Omega \quad (*) \\ & y_{uv} & \geq & 0 & \forall (u,v)\in E \end{array}$$

- Claim: There exists dual feasible solution for each $s-t$ cut in G.

Dual LP for Network Flow

Equivalent LP: Let Ω denote the set of all simple paths from s to t in G.

$$\begin{array}{lrcll} \text{maximize} & \sum_{p\in\Omega} x_p & & & \\ \text{subject to} & & & & \\ & \sum_{p\in\Omega \text{ containing } (u,v)} x_p & \leq & c(u,v) & \forall (u,v)\in E \\ & x_p & \geq & 0 & \forall p\in\Omega \end{array}$$

Dual LP: For each $(u,v)\in E$, add dual variable y_{uv}.

$$\begin{array}{lrcll} \text{minimize} & \sum_{(u,v)\in E} c(u,v) y_{uv} & & & \\ \text{subject to} & & & & \\ & \sum_{(u,v)\in p} y_{uv} & \geq & 1 & \forall p\in\Omega \quad (*) \\ & y_{uv} & \geq & 0 & \forall (u,v)\in E \end{array}$$

- Claim: There exists dual feasible solution for each $s-t$ cut in G.
- Construction: Consider any partition S, T of V, for $s\in S$, $t\in T$.
  - For each cut edge (u,v), i.e. $u\in S$, $t\in T$, set $y_{uv}=1$.
  - For all other edges, set $y_{uv}=0$.

Dual LP for Network Flow

Equivalent LP: Let Ω denote the set of all simple paths from s to t in G.

$$\begin{array}{lrcll} \text{maximize} & \sum_{p\in\Omega} x_p & & & \\ \text{subject to} & & & & \\ & \sum_{p\in\Omega \text{ containing } (u,v)} x_p & \leq & c(u,v) & \forall (u,v)\in E \\ & x_p & \geq & 0 & \forall p\in\Omega \end{array}$$

Dual LP: For each $(u,v)\in E$, add dual variable y_{uv}.

$$\begin{array}{lrcll} \text{minimize} & \sum_{(u,v)\in E} c(u,v) y_{uv} & & & \\ \text{subject to} & & & & \\ & \sum_{(u,v)\in p} y_{uv} & \geq & 1 & \forall p\in\Omega \quad (*) \\ & y_{uv} & \geq & 0 & \forall (u,v)\in E \end{array}$$

- Claim: There exists dual feasible solution for each $s-t$ cut in G.
- Construction: Consider any partition S, T of V, for $s\in S$, $t\in T$.
  - For each cut edge (u,v), i.e. $u\in S$, $t\in T$, set $y_{uv}=1$.
  - For all other edges, set $y_{uv}=0$.
- Objective function value is precisely capacity across S vs T cut.
- Q: Why is this solution feasible?

Dual LP for Network Flow

Equivalent LP: Let Ω denote the set of all simple paths from s to t in G.

$$\begin{array}{lrcll} \text{maximize} & \sum_{p\in\Omega} x_p & & & \\ \text{subject to} & & & & \\ & \sum_{p\in\Omega \text{ containing } (u,v)} x_p & \leq & c(u,v) & \forall (u,v)\in E \\ & x_p & \geq & 0 & \forall p \in \Omega \end{array}$$

Dual LP: For each $(u,v)\in E$, add dual variable y_{uv}.

$$\begin{array}{lrcll} \text{minimize} & \sum_{(u,v)\in E} c(u,v) y_{uv} & & & \\ \text{subject to} & & & & \\ & \sum_{(u,v)\in p} y_{uv} & \geq & 1 & \forall p\in\Omega \quad (*) \\ & y_{uv} & \geq & 0 & \forall (u,v) \in E \end{array}$$

- Claim: There exists dual feasible solution for each $s-t$ cut in G.
- Construction: Consider any partition S, T of V, for $s\in S$, $t \in T$.
    - For each cut edge (u,v), i.e. $u\in S$, $t\in T$, set $y_{uv}=1$.
    - For all other edges, set $y_{uv}=0$.
- Objective function value is precisely capacity across S vs T cut.
- Q: Why is this solution feasible?
- A: Each path $p\in\Omega$ takes *some* cut edge to pass from S to T, i.e. $(*)$ satisfied.

In other words

- By weak duality, capacity across any $s - t$ cut upper bounds max flow value.

In other words

- By weak duality, capacity across any $s-t$ cut upper bounds max flow value.
- But by strong duality, optimum primal and dual values must match.

In other words

- By weak duality, capacity across any $s - t$ cut upper bounds max flow value.
- But by strong duality, optimum primal and dual values must match.
- From this, one can re-obtain the Max-Flow Min-Cut Theorem! (Which said the Max Flow value equals the Min Cut value.)

In other words

- By weak duality, capacity across any $s - t$ cut upper bounds max flow value.
- But by strong duality, optimum primal and dual values must match.
- From this, one can re-obtain the Max-Flow Min-Cut Theorem! (Which said the Max Flow value equals the Min Cut value.)

Outline

1. Definitions
2. Applications
3. Duality theory
4. Solving LPs

What if we want to solve an LP?

Observations:

- Any primal feasible solution lower bounds p^*.
  - Implies solving primal LP is in NP.

What if we want to solve an LP?

Observations:

- Any primal feasible solution lower bounds p^*.
  - Implies solving primal LP is in NP.
- Any dual feasible solution upper bounds d^*, and hence p^* (by weak duality).
  - Implies refuting candidate optimal LP values is in co-NP.

What if we want to solve an LP?

Observations:

- Any primal feasible solution lower bounds p^*.
  - Implies solving primal LP is in NP.
- Any dual feasible solution upper bounds d^*, and hence p^* (by weak duality).
  - Implies refuting candidate optimal LP values is in co-NP.
- Conclusion: Solving LPs is in NP $\cap$ co-NP.
- But is it also in P?

Long history

Dates:

- (1827) Fourier proposes method for solving systems of linear inequalities

Long history

Dates:

- (1827) Fourier proposes method for solving systems of linear inequalities
- (1939-ish) Kantorovich and Koopmans independently study more general LPs. They would later share Nobel prize in Economics (1975).

Long history

Dates:

- (1827) Fourier proposes method for solving systems of linear inequalities
- (1939-ish) Kantorovich and Koopmans independently study more general LPs. They would later share Nobel prize in Economics (1975).
- (1941) Hitchcock gives solution very similar to simplex method.
- (1947) Dantzig discovers simplex method for solving LPs.

Long history

Dates:

- (1827) Fourier proposes method for solving systems of linear inequalities
- (1939-ish) Kantorovich and Koopmans independently study more general LPs. They would later share Nobel prize in Economics (1975).
- (1941) Hitchcock gives solution very similar to simplex method.
- (1947) Dantzig discovers simplex method for solving LPs.
- (1979) Khachiyan discovers poly-time algorithm using ellipsoid method.

Long history

Dates:

- (1827) Fourier proposes method for solving systems of linear inequalities
- (1939-ish) Kantorovich and Koopmans independently study more general LPs. They would later share Nobel prize in Economics (1975).
- (1941) Hitchcock gives solution very similar to simplex method.
- (1947) Dantzig discovers simplex method for solving LPs.
- (1979) Khachiyan discovers poly-time algorithm using ellipsoid method.
- (1984) Karmarkar discovers poly-time algorithm using interior-point method.

Long history

Dates:

- (1827) Fourier proposes method for solving systems of linear inequalities
- (1939-ish) Kantorovich and Koopmans independently study more general LPs. They would later share Nobel prize in Economics (1975).
- (1941) Hitchcock gives solution very similar to simplex method.
- (1947) Dantzig discovers simplex method for solving LPs.
- (1979) Khachiyan discovers poly-time algorithm using ellipsoid method.
- (1984) Karmarkar discovers poly-time algorithm using interior-point method.

Efficiency notes:

- Ellipsoid method rarely used in practice, numerically unstable.

Long history

Dates:

- (1827) Fourier proposes method for solving systems of linear inequalities
- (1939-ish) Kantorovich and Koopmans independently study more general LPs. They would later share Nobel prize in Economics (1975).
- (1941) Hitchcock gives solution very similar to simplex method.
- (1947) Dantzig discovers simplex method for solving LPs.
- (1979) Khachiyan discovers poly-time algorithm using ellipsoid method.
- (1984) Karmarkar discovers poly-time algorithm using interior-point method.

Efficiency notes:

- Ellipsoid method rarely used in practice, numerically unstable.
- In practice, one uses simplex method or interior-point method.

Long history

Dates:

- (1827) Fourier proposes method for solving systems of linear inequalities
- (1939-ish) Kantorovich and Koopmans independently study more general LPs. They would later share Nobel prize in Economics (1975).
- (1941) Hitchcock gives solution very similar to simplex method.
- (1947) Dantzig discovers simplex method for solving LPs.
- (1979) Khachiyan discovers poly-time algorithm using ellipsoid method.
- (1984) Karmarkar discovers poly-time algorithm using interior-point method.

Efficiency notes:

- Ellipsoid method rarely used in practice, numerically unstable.
- In practice, one uses simplex method or interior-point method.
- Simplex method is, in its current known variants, not poly-time.

Long history

Dates:

- (1827) Fourier proposes method for solving systems of linear inequalities
- (1939-ish) Kantorovich and Koopmans independently study more general LPs. They would later share Nobel prize in Economics (1975).
- (1941) Hitchcock gives solution very similar to simplex method.
- (1947) Dantzig discovers simplex method for solving LPs.
- (1979) Khachiyan discovers poly-time algorithm using ellipsoid method.
- (1984) Karmarkar discovers poly-time algorithm using interior-point method.

Efficiency notes:

- Ellipsoid method rarely used in practice, numerically unstable.
- In practice, one uses simplex method or interior-point method.
- Simplex method is, in its current known variants, not poly-time.

In practice:

- Many LP solvers available.
- Ex: CVX (implemented in Matlab), which can do LPs and a whole lot more.

Have a great break!