Introduction to Algorithms
Lecture 15 Linear Programming

Xue Chen
xuechen1989@ustc.edu.cn
2025 spring in

Outline

Introduction

General Paradigm

1. Alice is tired of solving algorithmic problems (dynamic program, divide & conquer, greedy method, graphs, number theory, ...)

Introduction

General Paradigm

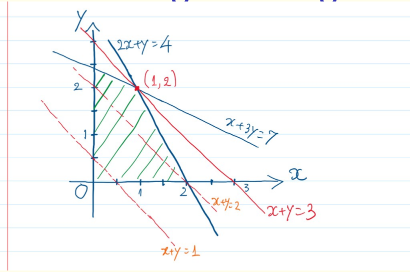
1. Alice is tired of solving algorithmic problems (dynamic program, divide & conquer, greedy method, graphs, number theory, ...)
2. Is there a general paradigm to solve all problems? ☺

Introduction

General Paradigm

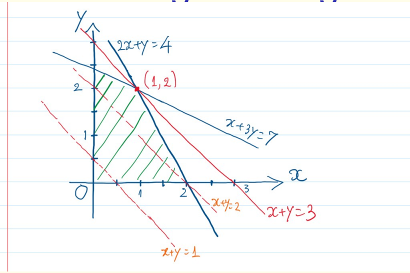
1. Alice is tired of solving algorithmic problems (dynamic program, divide & conquer, greedy method, graphs, number theory, ...)
2. Is there a general paradigm to solve all problems? ☺
3. Yes — Linear Programming!

Linear Programming



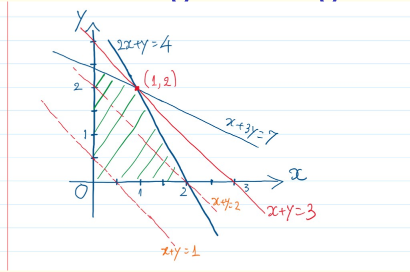
1. LP is arguably the most powerful ALGO technique

Linear Programming



1. LP is arguably the most powerful ALGO technique
2. Any problem with a poly-time ALGO will have a LP-based ALGO in poly-time — although neither the fastest nor the most intuitive

Linear Programming

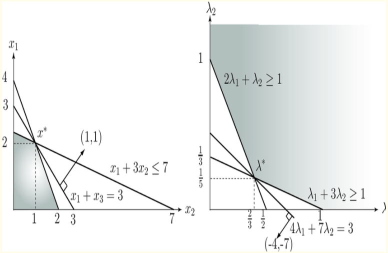


1. LP is arguably the most powerful ALGO technique
2. Any problem with a poly-time ALGO will have a LP-based ALGO in poly-time — although neither the fastest nor the most intuitive

3. Flexible and useful: management, optimization, scheduling, convex programming, ...

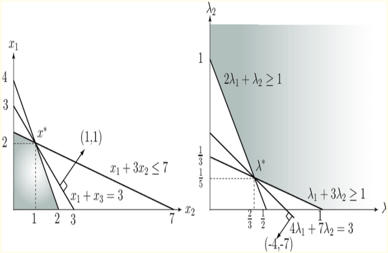
Overview

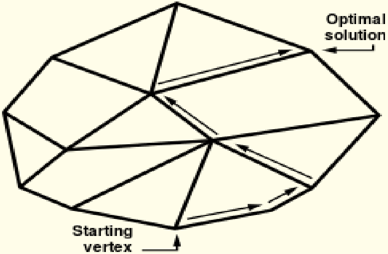
1. Define Linear Program: basic form and duality — see max-flow min-cut again



Overview

1. Define Linear Program: basic form and duality — see max-flow min-cut again

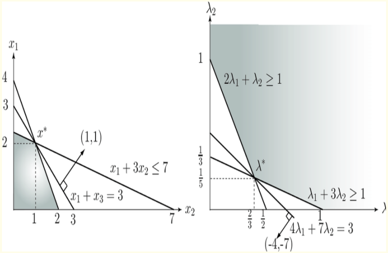


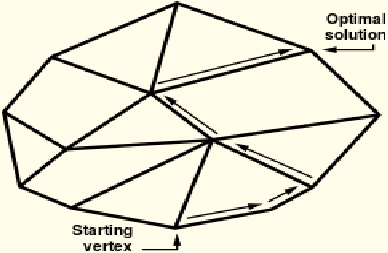


2. How to solve LP? — Simplex Method

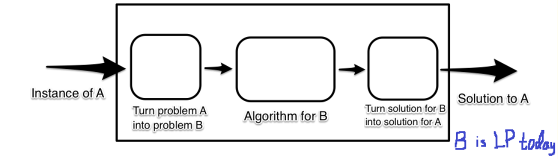
Overview

1. Define Linear Program: basic form and duality — see max-flow min-cut again





2. How to solve LP? — Simplex Method
3. Reduction: A technique that transform a problem to another



Outline

Optimization Problem

Alice can make two products A and B:

1. Profit of A is 1. Vice verse, profit of B is 6.
2. Alice could make at most 400 products per month; but at most 200 products of A and $\leqslant 300$ products B separately.

Optimization Problem

Alice can make two products A and B:

1. Profit of A is 1. Vice verse, profit of B is 6.
2. Alice could make at most 400 products per month; but at most 200 products of A and $\leqslant 300$ products B separately.
3. Help Alice maximize her profit

Optimization Problem

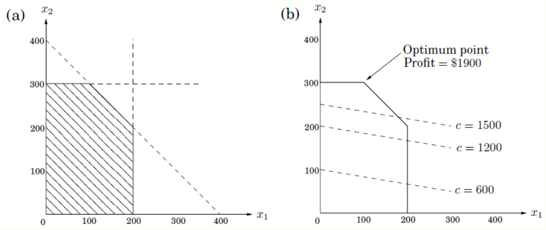
Alice can make two products A and B:

1. Profit of A is 1. Vice verse, profit of B is 6.
2. Alice could make at most 400 products per month; but at most 200 products of A and $\leqslant 300$ products B separately.
3. Help Alice maximize her profit
4. Consider x_1 and $x_2 \in \mathbb{R}$ denote number of products A and B separately

$$
\begin{aligned}
\max \quad & x_1 + 6x_2 \\
\text{subject to } & x_1 \leqslant 200 \\
& x_2 \leqslant 300 \\
& x_1 + x_2 \leqslant 400 \\
& x_1 \geqslant 0 \\
& x_2 \geqslant 0.
\end{aligned}
$$

Discussion

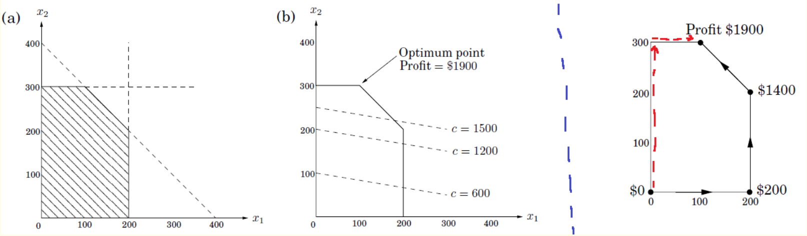
$$\begin{aligned} \max \quad & x_1 + 6x_2 \\ \text{subject to } & x_1 \leqslant 200 \\ & x_2 \leqslant 300 \\ & x_1 + x_2 \leqslant 400 \\ & x_1 \geqslant 0 \\ & x_2 \geqslant 0. \end{aligned}$$



1. Each constraint gives a halfspace
2. The feasible region is cut by 5 halfspaces
3. Objective value $x_1 + 6x_2 = c$ is a line $\Rightarrow$ maximization equals moving it as far as possible

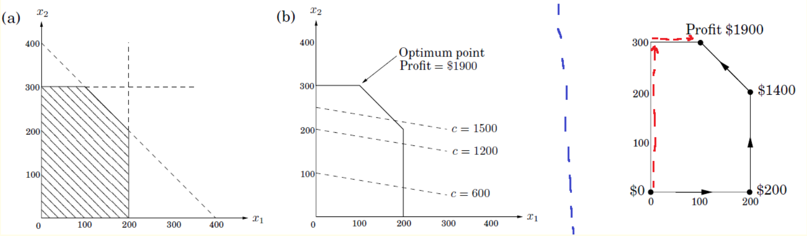
Solving Linear Programs

1. Since we are looking at the highest intersection between $x_1 + 6x_2 = c$ and the feasible region, optimum point is a vertex
2. Simplex Method: starts at $(0, 0)$ and repeatedly looks for an adjacent vertex of better value



Solving Linear Programs

1. Since we are looking at the highest intersection between $x_1 + 6x_2 = c$ and the feasible region, optimum point is a vertex
2. Simplex Method: starts at $(0, 0)$ and repeatedly looks for an adjacent vertex of better value



3. Stop after reaching a vertex without better neighbor
4. Why does this local test imply global optimality? Simple Convex Geometry

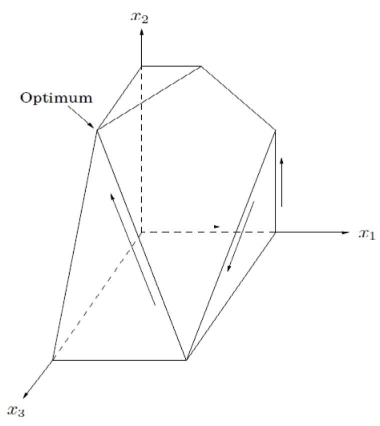
Example of 3D

1. Suppose Alice has product C whose profit is 13
2. Let x_3 denote number of product C
3. Product B and Product C are very time-consuming — $x_2 + 3x_3 \leqslant 600$

$$\begin{aligned} \max \quad & x_1 + 6x_2 + 13x_3 \\ \text{subject to } & x_1 \leqslant 200 \\ & x_2 \leqslant 300 \\ & x_1 + x_2 + x_3 \leqslant 400 \\ & x_2 + 3x_3 \leqslant 600 \\ & x_1, x_2, x_3 \geqslant 0 \end{aligned}$$

3D polyhedron

$$\begin{aligned} \max \quad & x_1 + 6x_2 + 13x_3 \\ \text{subject to } & x_1 \leqslant 200 \\ & x_2 \leqslant 300 \\ & x_1 + x_2 + x_3 \leqslant 400 \\ & x_2 + 3x_3 \leqslant 600 \\ & x_1, x_2, x_3 \geqslant 0 \end{aligned}$$



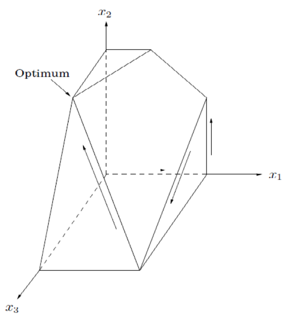
Simplex method may find:

$$\begin{array}{ccccc} (0,0,0) \to & (200,0,0) \to & (200,200,0) \to & (200,0,200) \to & (0,300,100) \\ \text{value } 0 \to & 200 \to & 1400 \to & 2800 \to & 3100 \end{array}$$

Outline

Duality

$$\begin{aligned} \max \quad & x_1 + 6x_2 + 13x_3 & \\ \text{subject to } & x_1 \leqslant 200 & (1) \\ & x_2 \leqslant 300 & (2) \\ & x_1 + x_2 + x_3 \leqslant 400 & (3) \\ & x_2 + 3x_3 \leqslant 600 & (4) \\ & x_1, x_2, x_3 \geqslant 0 & \end{aligned}$$

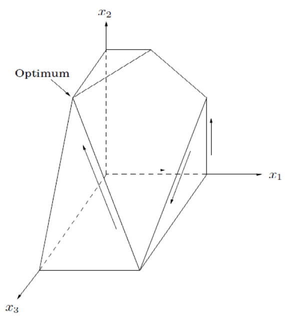


Question

Why is 3100 optimal?

Duality

$$
\begin{aligned}
\max \quad & x_1 + 6x_2 + 13x_3 & \\
\text{subject to } & x_1 \leqslant 200 & (1)\\
& x_2 \leqslant 300 & (2)\\
& x_1 + x_2 + x_3 \leqslant 400 & (3)\\
& x_2 + 3x_3 \leqslant 600 & (4)\\
& x_1, x_2, x_3 \geqslant 0 &
\end{aligned}
$$



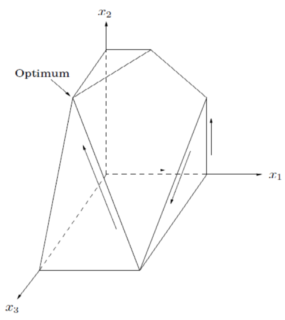
Question

Why is 3100 optimal?

1. Duality: $(2) \times 1 + (3) \times 1 + (4) \times 4 \leqslant 3100$.

Duality

$$\begin{aligned} \max \quad & x_1 + 6x_2 + 13x_3 \\ \text{subject to } & x_1 \leqslant 200 && (1) \\ & x_2 \leqslant 300 && (2) \\ & x_1 + x_2 + x_3 \leqslant 400 && (3) \\ & x_2 + 3x_3 \leqslant 600 && (4) \\ & x_1, x_2, x_3 \geqslant 0 \end{aligned}$$



Question

Why is 3100 optimal?

1. Duality: $(2) \times 1 + (3) \times 1 + (4) \times 4 \leqslant 3100$.
2. In general, any $y_1, y_2, \ldots, y_4$ such that

$$(1) \times y_1 + (2) \times y_2 + \cdots + (4) \times y_4 \geqslant x_1 + 6x_2 + 13x_3$$

gives a upper bound $200y_1 + 300y_2 + 400y_3 + 600y_4$.

More details

$OPT \leqslant 3100$ in this example:

$$\max \quad x_1 + 6x_2 + 13x_3$$
$$\begin{aligned} x_1 &\leq 200 & (1) \\ x_2 &\leq 300 & (2) \\ x_1 + x_2 + x_3 &\leq 400 & (3) \\ x_2 + 3x_3 &\leq 600 & (4) \\ x_1, x_2, x_3 &\geq 0 \end{aligned}$$

Why?

$(2) + (3) + 4 \cdot (4) \quad \Rightarrow \quad x_1 + 6x_2 + 13x_3 \leqslant 300 + 400 + 4 \cdot 600!$

More details

$OPT \leqslant 3100$ in this example:

$$\begin{aligned} \max \quad & x_1 + 6x_2 + 13x_3 & \\ & x_1 \le 200 & (1) \\ & x_2 \le 300 & (2) \\ & x_1 + x_2 + x_3 \le 400 & (3) \\ & x_2 + 3x_3 \le 600 & (4) \\ & x_1, x_2, x_3 \ge 0 & \end{aligned}$$

Why?

$(2) + (3) + 4 \cdot (4) \quad \Rightarrow \quad x_1 + 6x_2 + 13x_3 \leqslant 300 + 400 + 4 \cdot 600!$

1. Given Constraint (2),(3),(4), we can not have $x_1 + 6x_2 + 13x_3 > 3100$.
2. Since $(0, 300, 100)$ achieves 3100, it is optimal.

$$
\begin{array}{lrl}
\text{Multiplier} & \max \quad x_1 + 6x_2 + 13x_3 & \\
y_1 & x_1 \leq 200 & (1) \\
y_2 & x_2 \leq 300 & (2) \\
y_3 & x_1 + x_2 + x_3 \leq 400 & (3) \\
y_4 & x_2 + 3x_3 \leq 600 & (4) \\
 & x_1, x_2, x_3 \geq 0 &
\end{array}
$$

1. Any linear combination $y_1 \cdot (1) + y_2 \cdot (2) + y_3 \cdot (3) + y_4 \cdot (4) \geqslant x_1 + 6x_2 + 13x_3$ provides a upper bound $y_1 \cdot 200 + y_2 \cdot 300 + y_3 \cdot 400 + y_4 \cdot 600$ on the objective!

Multiplier		
	$\max \quad x_1 + 6x_2 + 13x_3$	
y_1	$x_1 \leq 200$	(1)
y_2	$x_2 \leq 300$	(2)
y_3	$x_1 + x_2 + x_3 \leq 400$	(3)
y_4	$x_2 + 3x_3 \leq 600$	(4)
	$x_1, x_2, x_3 \geq 0$	

1. Any linear combination $y_1 \cdot (1) + y_2 \cdot (2) + y_3 \cdot (3) + y_4 \cdot (4) \geqslant x_1 + 6x_2 + 13x_3$ provides a upper bound $y_1 \cdot 200 + y_2 \cdot 300 + y_3 \cdot 400 + y_4 \cdot 600$ on the objective!
2. OBS: Minimize $y_1 \cdot 200 + y_2 \cdot 300 + y_3 \cdot 400 + y_4 \cdot 600$ like a linear program again

$$\min\ y_1 \cdot 200 + y_2 \cdot 300 + y_3 \cdot 400 + y_4 \cdot 600$$

$$\begin{aligned}
\text{subject to } & y_1 + y_3 \geqslant 1 && //\text{constraint on } x_1 \\
& y_2 + y_3 + y_4 \geqslant 6 && //\text{constraint on } x_2 \\
& y_3 + 3y_4 \geqslant 13 && //\text{constraint on } x_3 \\
& y_1, y_2, y_3, y_4 \geqslant 0
\end{aligned}$$

Multiplier		
	$\max \quad x_1 + 6x_2 + 13x_3$	
y_1	$x_1 \leq 200$	(1)
y_2	$x_2 \leq 300$	(2)
y_3	$x_1 + x_2 + x_3 \leq 400$	(3)
y_4	$x_2 + 3x_3 \leq 600$	(4)
	$x_1, x_2, x_3 \geq 0$	

1. Any linear combination $y_1 \cdot (1) + y_2 \cdot (2) + y_3 \cdot (3) + y_4 \cdot (4) \geqslant x_1 + 6x_2 + 13x_3$ provides a upper bound $y_1 \cdot 200 + y_2 \cdot 300 + y_3 \cdot 400 + y_4 \cdot 600$ on the objective!
2. OBS: Minimize $y_1 \cdot 200 + y_2 \cdot 300 + y_3 \cdot 400 + y_4 \cdot 600$ like a linear program again

$$\min \; y_1 \cdot 200 + y_2 \cdot 300 + y_3 \cdot 400 + y_4 \cdot 600$$

$$\begin{aligned} \text{subject to } & y_1 + y_3 \geqslant 1 && //\text{constraint on } x_1 \\ & y_2 + y_3 + y_4 \geqslant 6 && //\text{constraint on } x_2 \\ & y_3 + 3y_4 \geqslant 13 && //\text{constraint on } x_3 \\ & y_1, y_2, y_3, y_4 \geqslant 0 \end{aligned}$$

3. Weak duality: This minimum value is not less than the maximum value of the original

LP of max flow

Given a (s,t)-network, let f_e denote flow on edge e:

$$\max \sum_{e=(s,v)} f_e$$

$$\text{s.t. } f_e \leqslant c_e \qquad \forall e$$

$$\sum_{e:(u,v)} f_e = \sum_{e:(v,w)} f_e \qquad \forall v \neq s,t$$

Consider its dual: y_e for the 1st type constraint and z_v for the 2nd type

$$\min \sum_e y_e \cdot c_e$$

$$\text{s.t. } y_e + z_v \geqslant 1 \qquad \forall e = (s,v)$$

$$y_e - z_u \geqslant 0 \qquad \forall e = (u,t)$$

$$y_e + z_v - z_u \geqslant 0 \qquad \forall e = (u,v)$$

$$y_e \geqslant 0$$

Max Flow and Min Cut

$$\begin{aligned}
\min \sum_e y_e \cdot c_e & \\
\text{s.t. } y_e + z_v \geqslant 1 & \qquad \forall e = (s, v) \\
y_e - z_v \geqslant 0 & \qquad \forall e = (v, t) \\
y_e + z_v - z_u \geqslant 0 & \qquad \forall e = (u, v)
\end{aligned}$$

Theorem

This is equivalent to (s, t)-min-cut!

1. For every cut $(S, \overline{S})$, construct y_e and z_v such that $\sum_e y_e c_e = CUT(S, \overline{S})$.
2. For any feasible solution $(y_e, z_v)_{e,v}$, what is the cut corresponding to them?

Outline

General Forms

1. m constraints on n variables: $x_1, x_2, \ldots, x_n$
2. At most one linear objective (could be empty): Given $c_1, \ldots, c_n \in \mathbb{R}$, either $\max c_1 x_1 + \ldots + c_n x_n$ or $\min c_1 x_1 + \ldots + c_n x_n$

General Forms

1. m constraints on n variables: $x_1, x_2, \ldots, x_n$
2. At most one linear objective (could be empty): Given $c_1, \ldots, c_n \in \mathbb{R}$, either $\max c_1 x_1 + \ldots + c_n x_n$ or $\min c_1 x_1 + \ldots + c_n x_n$
3. m linear constraints: i-th constraint could be

$$\begin{aligned} & a_{i,1}x_1 + a_{i,2}x_2 + \cdots + a_{i,n}x_n \leqslant b_i \\ \text{or} \quad & a_{i,1}x_1 + a_{i,2}x_2 + \cdots + a_{i,n}x_n = b_i \\ \text{or} \quad & a_{i,1}x_1 + a_{i,2}x_2 + \cdots + a_{i,n}x_n \geqslant b_i \end{aligned}$$

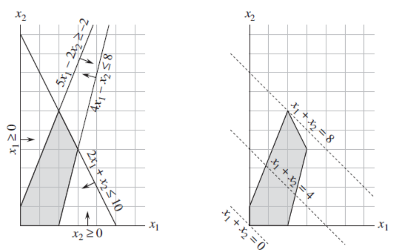
General Forms

1. m constraints on n variables: $x_1, x_2, \ldots, x_n$
2. At most one linear objective (could be empty): Given $c_1, \ldots, c_n \in \mathbb{R}$, either $\max c_1 x_1 + \ldots + c_n x_n$ or $\min c_1 x_1 + \ldots + c_n x_n$
3. m linear constraints: i-th constraint could be

$$
\begin{aligned}
& a_{i,1}x_1 + a_{i,2}x_2 + \cdots + a_{i,n}x_n \leqslant b_i \\
\text{or} \quad & a_{i,1}x_1 + a_{i,2}x_2 + \cdots + a_{i,n}x_n = b_i \\
\text{or} \quad & a_{i,1}x_1 + a_{i,2}x_2 + \cdots + a_{i,n}x_n \geqslant b_i
\end{aligned}
$$

Example:

$$
\begin{array}{llcrcl}
\text{maximize} & x_1 & + & x_2 & & \\
\text{subject to} & & & & & \\
& 4x_1 & - & x_2 & \le & 8 \\
& 2x_1 & + & x_2 & \le & 10 \\
& 5x_1 & - & 2x_2 & \ge & -2 \\
& & x_1, x_2 & & \ge & 0 .
\end{array}
$$



Standard Form

It is much easier to work with.

Standard form

A linear program of n variables and m constraints is in this form:

1. Input: $c \in \mathbb{R}^n, b \in \mathbb{R}^m, A \in \mathbb{R}^{m \times n}$
2. Object: $\max c_1 x_1 + \cdots + c_n x_n$
3. Constraints: $\sum_j A_{i,j} x_j \leqslant b_i, \forall i$ and $x_j \geqslant 0, \forall j$

Standard Form

It is much easier to work with.

Standard form

A linear program of n variables and m constraints is in this form:

1. Input: $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$
2. Object: $\max c_1 x_1 + \cdots + c_n x_n$
3. Constraints: $\sum_j A_{i,j} x_j \leqslant b_i, \forall i$ and $x_j \geqslant 0, \forall j$

In linear algebra, $\max c^\top x$ subject to $Ax \leqslant b$ and $x \geqslant 0$

Standard Form

It is much easier to work with.

Standard form

A linear program of n variables and m constraints is in this form:

1. Input: $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$
2. Object: $\max c_1 x_1 + \cdots + c_n x_n$
3. Constraints: $\sum_j A_{i,j} x_j \leqslant b_i, \forall i$ and $x_j \geqslant 0, \forall j$

In linear algebra, $\max c^\top x$ subject to $Ax \leqslant b$ and $x \geqslant 0$

Claim

Any general LP with n variables and m constraints could be reduced to a standard LP of $O(n)$ variables and $O(m)$ constraints

Standard Form

It is much easier to work with.

Standard form

A linear program of n variables and m constraints is in this form:

1. Input: $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$
2. Object: $\max c_1 x_1 + \cdots + c_n x_n$
3. Constraints: $\sum_j A_{i,j} x_j \leqslant b_i, \forall i$ and $x_j \geqslant 0, \forall j$

In linear algebra, $\max c^\top x$ subject to $Ax \leqslant b$ and $x \geqslant 0$

Claim

Any general LP with n variables and m constraints could be reduced to a standard LP of $O(n)$ variables and $O(m)$ constraints

Question: what does reduced mean?

Outline

Introduction

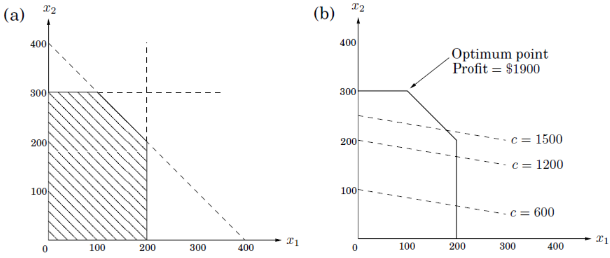
Main question: Give a LP in the standard form, how to find the optimal solution efficiently?

2D Example

Objective function $\max x_1 + 6x_2$

Constraints

$$x_1 \le 200$$
$$x_2 \le 300$$
$$x_1 + x_2 \le 400$$
$$x_1, x_2 \ge 0$$

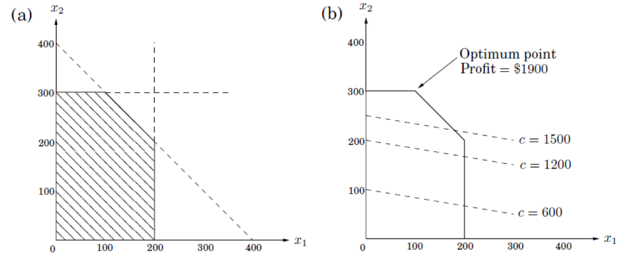


Introduction

Main question: Give a LP in the standard form, how to find the optimal solution **efficiently**?

2D Example

Objective function	$\max x_1 + 6x_2$
Constraints	$x_1 \leq 200$
	$x_2 \leq 300$
	$x_1 + x_2 \leq 400$
	$x_1, x_2 \geq 0$



3 cases in general:

1. Feasible region is empty
2. Unbounded value
3. $OPT < \infty$ exists

Find OPT

When we discuss the optimal solution, it means Case 3: $OPT < \infty$ exists.

Find OPT

When we discuss the optimal solution, it means Case 3: $OPT < \infty$ exists.

Claim

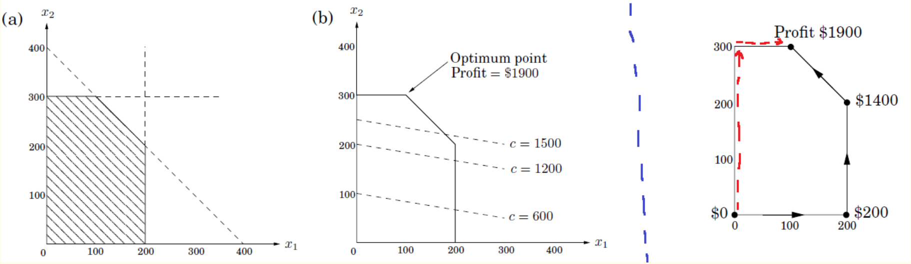
OPT is achieved at a vertex of the feasible region; exceptions are Case 1 and Case 2.

Find OPT

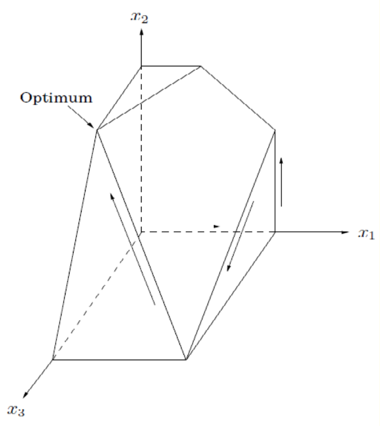
When we discuss the optimal solution, it means Case 3: $OPT < \infty$ exists.

Claim

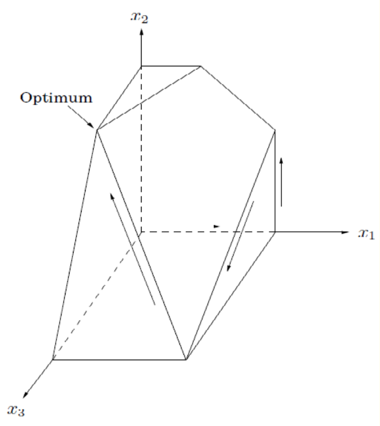
OPT is achieved at a vertex of the feasible region; exceptions are Case 1 and Case 2.



3D Example



3D Example



Questions

1. How to implement it?
2. Why is it correct? — LP duality
3. Running Time?

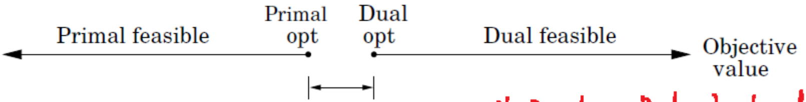
Primal and Dual

Primal LP:

$$\max \ \mathbf{c}^T\mathbf{x}$$
$$\mathbf{A}\mathbf{x} \le \mathbf{b}$$
$$\mathbf{x} \ge 0$$

Dual LP:

$$\min \ \mathbf{y}^T\mathbf{b}$$
$$\mathbf{y}^T\mathbf{A} \ge \mathbf{c}^T$$
$$\mathbf{y} \ge 0$$



This duality gap is zero if Primal or Dual opt is bounded

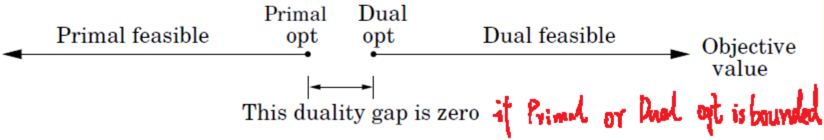
Primal and Dual

Primal LP:

$$\max\ \mathbf{c}^T\mathbf{x}$$
$$\mathbf{A}\mathbf{x} \le \mathbf{b}$$
$$\mathbf{x} \ge 0$$

Dual LP:

$$\min\ \mathbf{y}^T\mathbf{b}$$
$$\mathbf{y}^T\mathbf{A} \ge \mathbf{c}^T$$
$$\mathbf{y} \ge 0$$

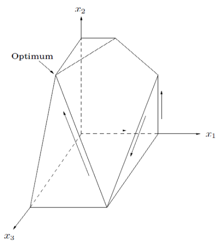


Duality THM

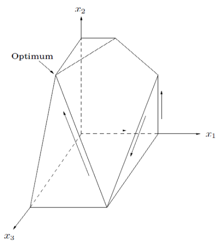
If a LP has a bounded optimum, then so does it dual. Moreover, the two optimum values coincide.

Example: Max-flow Min-cut theorem. See the simplex algorithm for a full version proof.

Notation for Simplex

Generalize 3D to n-dimensional space 

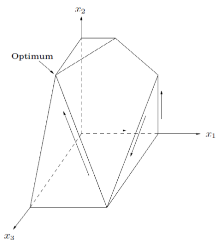
Notation for Simplex



Generalize 3D to *n*-dimensional space

1. Define vertices: Pick a subset of constraints. If $\exists$ a unique point that satisfies them with equality, and this point is feasible, call it a vertex

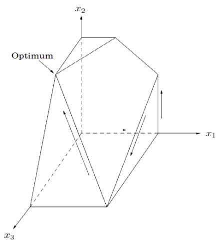
Notation for Simplex



Generalize 3D to n-dimensional space

1. Define vertices: Pick a subset of constraints. If $\exists$ a unique point that satisfies them with equality, and this point is feasible, call it a vertex
2. OBS: Each vertex is specified by n constraints
3. Two vertices are neighbors if they share $n-1$ defining inequalities in common

Notation for Simplex



Generalize 3D to *n*-dimensional space

1. Define vertices: Pick a subset of constraints. If $\exists$ a unique point that satisfies them with equality, and this point is feasible, call it a vertex
2. OBS: Each vertex is specified by *n* constraints
3. Two vertices are neighbors if they share $n-1$ defining inequalities in common
4. Rough Description of Simplex:

   let v be any vertex of the feasible region
   while there is a neighbor v' of v with better objective value:
       set $v = v'$

Formal Description

$$\max c^\top x \text{ subject to } Ax \leqslant b, x \geqslant 0$$

1. Basic idea is to shift x to $\vec{0}$ such that the job becomes easier

   — because $\vec{0}$ is OPT iff $c_i \leqslant 0$ for all c

Formal Description

$$\max c^\top x \text{ subject to } Ax \leqslant b, x \geqslant 0$$

1. Basic idea is to shift x to $\vec{0}$ such that the job becomes easier
   — because $\vec{0}$ is OPT iff $c_i \leqslant 0$ for all c
2. O.w. $\exists c_i > 0$. Let us adjust switch x_i from 0 to > 0 — but how much?

Formal Description

$$\max c^\top x \text{ subject to } Ax \leqslant b, x \geqslant 0$$

1. Basic idea is to shift x to $\vec{0}$ such that the job becomes easier — because $\vec{0}$ is OPT iff $c_i \leqslant 0$ for all c
2. O.w. $\exists c_i > 0$. Let us adjust switch x_i from 0 to > 0 — but how much?
3. Since $x_j = 0$ for $j \neq i$, consider the tight constraint of $x_i = \min_{k:A_{k,i}>0} b_k/A_{k,i}$
4. Adjust $x_i := \min_{k:A_{k,i}>0} b_k/A_{k,i}$

Formal Description

$$\max c^\top x \text{ subject to } Ax \leqslant b, x \geqslant 0$$

1. Basic idea is to shift x to $\vec{0}$ such that the job becomes easier — because $\vec{0}$ is OPT iff $c_i \leqslant 0$ for all c
2. O.w. $\exists c_i > 0$. Let us adjust switch x_i from 0 to > 0 — but how much?
3. Since $x_j = 0$ for $j \neq i$, consider the tight constraint of $x_i = \min_{k: A_{k,i} > 0} b_k / A_{k,i}$
4. Adjust $x_i := \min_{k: A_{k,i} > 0} b_k / A_{k,i}$
5. Question: How to reduce to $\vec{0}$?
6. Set $x_0 = x$ and consider $\max c^\top x'$ s.t. $Ax' \leqslant b - Ax_0, x' + x_0 \geqslant 0$

Formal Description

$$\max c^\top x \text{ subject to } Ax \leqslant b, x \geqslant 0$$

1. Basic idea is to shift x to $\vec{0}$ such that the job becomes easier — because $\vec{0}$ is OPT iff $c_i \leqslant 0$ for all c
2. O.w. $\exists c_i > 0$. Let us adjust switch x_i from 0 to > 0 — but how much?
3. Since $x_j = 0$ for $j \neq i$, consider the tight constraint of $x_i = \min_{k: A_{k,i} > 0} b_k / A_{k,i}$
4. Adjust $x_i := \min_{k: A_{k,i} > 0} b_k / A_{k,i}$
5. Question: How to reduce to $\vec{0}$?
6. Set $x_0 = x$ and consider $\max c^\top x'$ s.t. $Ax' \leqslant b - Ax_0, x' + x_0 \geqslant 0$
7. Formally, for each tight constraint $b_i = A_i \cdot x$, introduce $y_i = b_i - A_i \cdot x$ and rewrite the other constraint into y!

Initial LP:	*Current vertex:* $\{④, ⑤\}$ (origin). *Objective value:* 0.
$$\begin{aligned} &\max\ 2x_1 + 5x_2 \\ 2x_1 - x_2 &\leq 4 \quad ① \\ x_1 + 2x_2 &\leq 9 \quad ② \\ -x_1 + x_2 &\leq 3 \quad ③ \\ x_1 &\geq 0 \quad ④ \\ x_2 &\geq 0 \quad ⑤ \end{aligned}$$	*Move:* increase x_2. ⑤ is released, ③ becomes tight. Stop at $x_2 = 3$. New vertex $\{④, ③\}$ has local coordinates (y_1, y_2): $$y_1 = x_1, \quad y_2 = 3 + x_1 - x_2$$

Initial LP: $$\begin{aligned} &\max\ 2x_1 + 5x_2 \\ 2x_1 - x_2 &\leq 4 \quad (1) \\ x_1 + 2x_2 &\leq 9 \quad (2) \\ -x_1 + x_2 &\leq 3 \quad (3) \\ x_1 &\geq 0 \quad (4) \\ x_2 &\geq 0 \quad (5) \end{aligned}$$	*Current vertex:* {(4), (5)} (origin). *Objective value:* 0. *Move:* increase x_2. (5) is released, (3) becomes tight. Stop at $x_2 = 3$. New vertex {(4), (3)} has local coordinates (y_1, y_2): $$y_1 = x_1, \quad y_2 = 3 + x_1 - x_2$$
Rewritten LP: $$\begin{aligned} &\max\ 15 + 7y_1 - 5y_2 \\ y_1 + y_2 &\leq 7 \quad (1) \\ 3y_1 - 2y_2 &\leq 3 \quad (2) \\ y_2 &\geq 0 \quad (3) \\ y_1 &\geq 0 \quad (4) \\ -y_1 + y_2 &\leq 3 \quad (5) \end{aligned}$$	*Current vertex:* {(4), (3)}. *Objective value:* 15. *Move:* increase y_1. (4) is released, (2) becomes tight. Stop at $y_1 = 1$. New vertex {(2), (3)} has local coordinates (z_1, z_2): $$z_1 = 3 - 3y_1 + 2y_2, \quad z_2 = y_2$$

Initial LP: $$\begin{aligned} &\max\ 2x_1+5x_2 \\ 2x_1-x_2 &\le 4 \quad ① \\ x_1+2x_2 &\le 9 \quad ② \\ -x_1+x_2 &\le 3 \quad ③ \\ x_1 &\ge 0 \quad ④ \\ x_2 &\ge 0 \quad ⑤ \end{aligned}$$	***Current vertex:*** {④, ⑤} (origin). ***Objective value:*** 0. ***Move:*** increase x_2. ⑤ is released, ③ becomes tight. Stop at $x_2 = 3$. New vertex {④, ③} has local coordinates (y_1, y_2): $$y_1 = x_1, \quad y_2 = 3 + x_1 - x_2$$
Rewritten LP: $$\begin{aligned} &\max\ 15+7y_1-5y_2 \\ y_1+y_2 &\le 7 \quad ① \\ 3y_1-2y_2 &\le 3 \quad ② \\ y_2 &\ge 0 \quad ③ \\ y_1 &\ge 0 \quad ④ \\ -y_1+y_2 &\le 3 \quad ⑤ \end{aligned}$$	***Current vertex:*** {④, ③}. ***Objective value:*** 15. ***Move:*** increase y_1. ④ is released, ② becomes tight. Stop at $y_1 = 1$. New vertex {②, ③} has local coordinates (z_1, z_2): $$z_1 = 3 - 3y_1 + 2y_2, \quad z_2 = y_2$$
Rewritten LP: $$\begin{aligned} &\max\ 22-\tfrac{7}{3}z_1-\tfrac{1}{3}z_2 \\ -\tfrac{1}{3}z_1+\tfrac{5}{3}z_2 &\le 6 \quad ① \\ z_1 &\ge 0 \quad ② \\ z_2 &\ge 0 \quad ③ \\ \tfrac{1}{3}z_1-\tfrac{2}{3}z_2 &\le 1 \quad ④ \\ \tfrac{1}{3}z_1+\tfrac{1}{3}z_2 &\le 4 \quad ⑤ \end{aligned}$$	***Current vertex:*** {②, ③}. ***Objective value:*** 22. ***Optimal:*** all $c_i < 0$. Solve ②, ③ (in original LP) to get optimal solution $(x_1, x_2) = (1, 4)$.

Wrap UP

1. Duality Theorem: Those tight constraints are the support of multipliers y and coefficients are in the last c
— in the above example, $\frac{7}{3} \cdot (2) + \frac{1}{3} \cdot (3) = 2x_1 + 5x_2$

Wrap UP

1. Duality Theorem: Those tight constraints are the support of multipliers y and coefficients are in the last c
— in the above example, $\frac{7}{3} \cdot (2) + \frac{1}{3} \cdot (3) = 2x_1 + 5x_2$
2. How to start? Solve

$$\min t \text{ subject to } Ax = (1-t)b, t \geqslant 0, x \geqslant 0$$

with start vertex $t = 1, x = \vec{0}$.

Wrap UP

1. Duality Theorem: Those tight constraints are the support of multipliers y and coefficients are in the last c
   — in the above example, $\frac{7}{3} \cdot (2) + \frac{1}{3} \cdot (3) = 2x_1 + 5x_2$
2. How to start? Solve

$$\min t \text{ subject to } Ax = (1-t)b, t \geqslant 0, x \geqslant 0$$

   with start vertex $t = 1, x = \vec{0}$.
3. Degenerate: linear dependent
4. Unbounded: adjust x_i since $c_i > 0$; but $A_{k,i} \leqslant 0$ for all k

Wrap UP

1. Duality Theorem: Those tight constraints are the support of multipliers y and coefficients are in the last c
— in the above example, $\frac{7}{3} \cdot (2) + \frac{1}{3} \cdot (3) = 2x_1 + 5x_2$
2. How to start? Solve

$$\min t \text{ subject to } Ax = (1-t)b, t \geqslant 0, x \geqslant 0$$

with start vertex $t = 1, x = \vec{0}$.
3. Degenerate: linear dependent
4. Unbounded: adjust x_i since $c_i > 0$; but $A_{k,i} \leqslant 0$ for all k
5. Running time — fast in practical but $\binom{n+m}{n}$ in the worst

Wrap UP

1. Duality Theorem: Those tight constraints are the support of multipliers y and coefficients are in the last c
   — in the above example, $\frac{7}{3} \cdot (2) + \frac{1}{3} \cdot (3) = 2x_1 + 5x_2$
2. How to start? Solve

$$\min t \text{ subject to } Ax = (1-t)b, t \geqslant 0, x \geqslant 0$$

   with start vertex $t = 1, x = \vec{0}$.
3. Degenerate: linear dependent
4. Unbounded: adjust x_i since $c_i > 0$; but $A_{k,i} \leqslant 0$ for all k
5. Running time — fast in practical but $\binom{n+m}{n}$ in the worst
6. Strong poly-time algorithms: Interior method and Ellipsoid method
   — in fact, m could be exponential as long as it finds a violated constraint in $n^{O(1)}$

Outline

Shortest Paths

Description

Given a weighted directed graph $G = (V, E)$ and (s, t), find the shortest path from s to t

1. Basic idea: let d_v denote the shortest distance from s to v.

Shortest Paths

Description

Given a weighted directed graph $G = (V, E)$ and (s, t), find the shortest path from s to t

1. Basic idea: let d_v denote the shortest distance from s to v.
2. Two types of constraint: (1) $d_v \leqslant d_u + w(u, v)$ for any edge (u, v); (2) $d_v \geqslant 0$
3. Question: What is our objective function?

Min Cost Flow

Description

Given a directed graph $G = (V, E)$ with capacity c and cost a, find a min-cost d-flow from s to t

Min Cost Flow

Description

Given a directed graph $G = (V, E)$ with capacity c and cost a, find a min-cost d-flow from s to t

$$
\begin{aligned}
\text{minimize} \quad & \sum_{(u,v)\in E} a(u,v) f_{uv} \\
\text{subject to} \quad & \\
& f_{uv} \le c(u,v) \quad \text{for each } u, v \in V , \\
& \sum_{v\in V} f_{vu} - \sum_{v\in V} f_{uv} = 0 \quad \text{for each } u \in V - \{s,t\} \\
& \sum_{v\in V} f_{sv} - \sum_{v\in V} f_{vs} = d , \\
& f_{uv} \ge 0 \quad \text{for each } u, v \in V .
\end{aligned}
$$

Min Cost Flow

Description

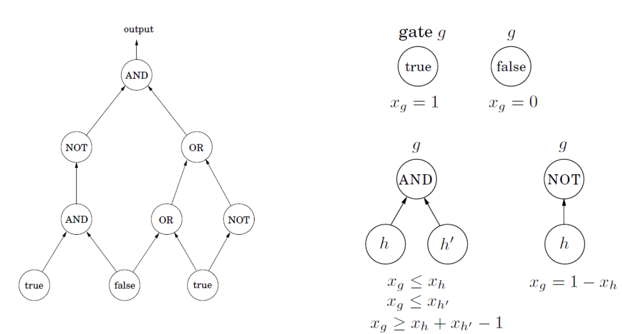
Given a directed graph $G=(V,E)$ with capacity c and cost a, find a min-cost d-flow from s to t

$$
\begin{aligned}
\text{minimize} \quad & \sum_{(u,v)\in E} a(u,v) f_{uv} \\
\text{subject to} \quad & \\
& f_{uv} \le c(u,v) \quad \text{for each } u,v\in V, \\
& \sum_{v\in V} f_{vu} - \sum_{v\in V} f_{uv} = 0 \quad \text{for each } u\in V-\{s,t\} \\
& \sum_{v\in V} f_{sv} - \sum_{v\in V} f_{vs} = d, \\
& f_{uv} \ge 0 \quad \text{for each } u,v\in V.
\end{aligned}
$$

Question: How to compute a max-flow with min-cost?

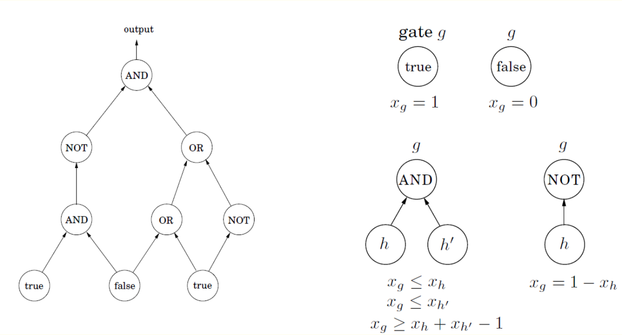
Circuit Evaluation

1. Given a boolean circuit, a DAG with following gates
2. Input gates have in-degree 0, with value True or False
3. AND and OR gates of degree 2, NOT gates of degree 1
4. One of the gate is outputting



Circuit Evaluation

1. Given a boolean circuit, a DAG with following gates
2. Input gates have in-degree 0, with value True or False
3. AND and OR gates of degree 2, NOT gates of degree 1
4. One of the gate is outputting



Claim

All problems that can be solved in polynomial time admit a poly-size LP.

— LP is the most powerful algorithm tool

Summary

1. LP is a powerful tool to solve many problems in poly-time (even though not the fastest)
2. Flexible with a geometric interpretation
3. Applications in analysis, data science, machine learning, ...

Questions?