Advanced Algorithms

Nicole Megow (Universität Bremen)

SoSe 2025

Introduction to Linear Programming

Lecture 10

So far in this course

We have focused on exploiting problem structure to design tailored algorithms for specific combinatorial optimization problems:

- Maximum (weighted) matching
- Maximum s-t-flow and minimum s-t-cut
- Minimum-cost flows
- Scheduling

So far in this course

We have focused on exploiting problem structure to design tailored algorithms for specific combinatorial optimization problems:

- Maximum (weighted) matching
- Maximum s-t-flow and minimum s-t-cut
- Minimum-cost flows
- Scheduling

Wouldn't it be great to have a more general machinery for solving (certain) optimization problems? A black box?

So far in this course

We have focused on exploiting problem structure to design tailored algorithms for specific combinatorial optimization problems:

- Maximum (weighted) matching
- Maximum s-t-flow and minimum s-t-cut
- Minimum-cost flows
- Scheduling

Wouldn't it be great to have a more general machinery for solving (certain) optimization problems? A black box?

$\longrightarrow$ Linear Programming

Linear Program

Definition

A linear program (LP) is a model of an optimization problem over real-valued variables with finitely many linear constraints and a linear objective function.

Linear Program

Definition

A linear program (LP) is a model of an optimization problem over real-valued variables with finitely many linear constraints and a linear objective function.

- A linear function: $c_1 \cdot x_1 + c_2 \cdot x_2 + c_3 \cdot x_3 + \cdots + c_n x_n$, where c_i are parameters (constants), x_i are variables.
- A linear constraint: $a_1 x_1 + a_2 x_2 + \cdots + a_n x_n \leq b$, where a_i and b are parameters.

Linear Program

Definition

A linear program (LP) is a model of an optimization problem over real-valued variables with finitely many linear constraints and a linear objective function.

- A linear function: $c_1 \cdot x_1 + c_2 \cdot x_2 + c_3 \cdot x_3 + \cdots + c_n x_n$,
  where c_i are parameters (constants), x_i are variables.
- A linear constraint: $a_1 x_1 + a_2 x_2 + \cdots + a_n x_n \leq b$,
  where a_i and b are parameters.
- Arbitrary linear inequalities with $\geq, \leq$ and $=$.

Linear Program

Definition

A linear program (LP) is a model of an optimization problem over real-valued variables with finitely many linear constraints and a linear objective function.

- A linear function: $c_1 \cdot x_1 + c_2 \cdot x_2 + c_3 \cdot x_3 + \cdots + c_n x_n$, where c_i are parameters (constants), x_i are variables.
- A linear constraint: $a_1 x_1 + a_2 x_2 + \cdots + a_n x_n \leq b$, where a_i and b are parameters.
- Arbitrary linear inequalities with $\geq, \leq$ and $=$.
- Objective: maximum or minimum.

Linear Program

Definition

A linear program (LP) is a model of an optimization problem over real-valued variables with finitely many linear constraints and a linear objective function.

Typical LP:

$$
\begin{array}{rl}
\min & c_1x_1 + c_2x_2 + \ldots + c_nx_n \\
\text{s.t.} & a_{1,1}x_1 + a_{1,2}x_2 + \ldots + a_{1,n}x_n \geq b_1 \\
& a_{2,1}x_1 + a_{2,2}x_2 + \ldots + a_{2,n}x_n \geq b_2 \\
& \ldots \\
& a_{m,1}x_1 + a_{m,2}x_2 + \ldots + a_{m,n}x_n \geq b_n \\
& \forall i\colon x_i \geq 0, x_i \in \mathbb{R}
\end{array}
$$

Compact form:

$$
\begin{array}{rl}
\min & c^T x \\
\text{s.t.} & Ax \geq b \\
& x \geq 0
\end{array}
$$

First Example: Pizza and Lasagne

Ingredients:

	Pizza	Lasagne	available
Tomatoes	2	3	18
Cheese	4	3	24

First Example: Pizza and Lasagne

Ingredients:

	Pizza	Lasagne	available
Tomatoes	2	3	18
Cheese	4	3	24

Profit: Pizza 8 €, Lasagne 7 €

First Example: Pizza and Lasagne

Ingredients:

	Pizza	Lasagne	available
Tomatoes	2	3	18
Cheese	4	3	24

Profit: Pizza 8 €, Lasagne 7 €

Task: Determine an optimal producible number of pizza and lasagne to maximize total profit.

First Example: Pizza and Lasagne

Ingredients:

	Pizza	Lasagne	available
Tomatoes	2	3	18
Cheese	4	3	24

Profit: Pizza 8 €, Lasagne 7 €

Task: Determine an optimal producible number of pizza and lasagne to maximize total profit.

LP Model

$x_1 =$ number of produced pizzas

$x_2 =$ number of produced lasagne

First Example: Pizza and Lasagne

Ingredients:

	Pizza	Lasagne	available
Tomatoes	2	3	18
Cheese	4	3	24

Profit: Pizza 8 €, Lasagne 7 €

Task: Determine an optimal producible number of pizza and lasagne to maximize total profit.

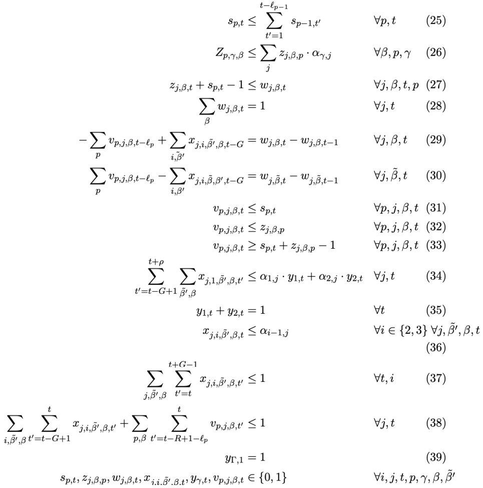
LP Model

$x_1 =$ number of produced pizzas

$x_2 =$ number of produced lasagne

$$\begin{aligned} \max \quad & 8x_1 + 7x_2 & & \text{(profit)} \\ s.t. \quad & 2x_1 + 3x_2 \leq 18 & & \text{(tomato)} \\ & 4x_1 + 3x_2 \leq 24 & & \text{(cheese)} \\ & x_1, x_2 \geq 0 & & \end{aligned}$$

LPs can be arbitrarily complex



$$s_{p,t} \leq \sum_{t'=1}^{t-\ell_{p-1}} s_{p-1,t'} \qquad \forall p, t \qquad (25)$$

$$Z_{p,\gamma,\beta} \leq \sum_{j} z_{j,\beta,p} \cdot \alpha_{\gamma,j} \qquad \forall \beta, p, \gamma \qquad (26)$$

$$z_{j,\beta,t} + s_{p,t} - 1 \leq w_{j,\beta,t} \qquad \forall j, \beta, t, p \qquad (27)$$

$$\sum_{\beta} w_{j,\beta,t} = 1 \qquad \forall j, t \qquad (28)$$

$$-\sum_{p} v_{p,j,\beta,t-\ell_p} + \sum_{i,\tilde{\beta}'} x_{j,i,\tilde{\beta}',\beta,t-G} = w_{j,\beta,t} - w_{j,\beta,t-1} \qquad \forall j, \beta, t \qquad (29)$$

$$\sum_{p} v_{p,j,\beta,t-\ell_p} - \sum_{i,\beta'} x_{j,i,\tilde{\beta},\beta',t-G} = w_{j,\tilde{\beta},t} - w_{j,\tilde{\beta},t-1} \qquad \forall j, \tilde{\beta}, t \qquad (30)$$

$$v_{p,j,\beta,t} \leq s_{p,t} \qquad \forall p, j, \beta, t \qquad (31)$$

$$v_{p,j,\beta,t} \leq z_{j,\beta,p} \qquad \forall p, j, \beta, t \qquad (32)$$

$$v_{p,j,\beta,t} \geq s_{p,t} + z_{j,\beta,p} - 1 \qquad \forall p, j, \beta, t \qquad (33)$$

$$\sum_{t'=t-G+1}^{t+\rho} \sum_{\tilde{\beta}',\beta} x_{j,1,\tilde{\beta}',\beta,t'} \leq \alpha_{1,j} \cdot y_{1,t} + \alpha_{2,j} \cdot y_{2,t} \qquad \forall j, t \qquad (34)$$

$$y_{1,t} + y_{2,t} = 1 \qquad \forall t \qquad (35)$$

$$x_{j,i,\tilde{\beta}',\beta,t} \leq \alpha_{i-1,j} \qquad \forall i \in \{2,3\}\ \forall j, \tilde{\beta}', \beta, t \qquad (36)$$

$$\sum_{j,\tilde{\beta}',\beta} \sum_{t'=t}^{t+G-1} x_{j,i,\tilde{\beta}',\beta,t'} \leq 1 \qquad \forall t, i \qquad (37)$$

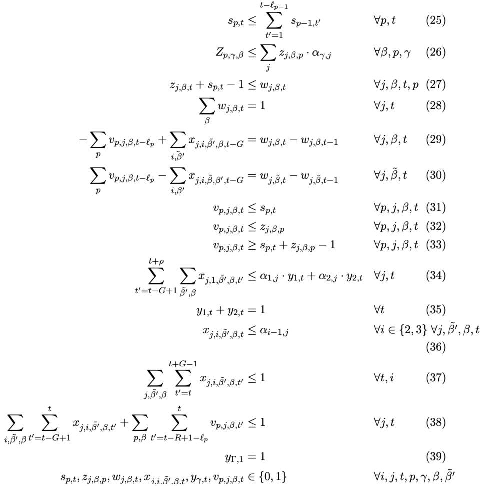
$$\sum_{i,\tilde{\beta}',\beta} \sum_{t'=t-G+1}^{t} x_{j,i,\tilde{\beta}',\beta,t'} + \sum_{p,\beta} \sum_{t'=t-R+1-\ell_p}^{t} v_{p,j,\beta,t'} \leq 1 \qquad \forall j, t \qquad (38)$$

$$y_{\Gamma,1} = 1 \qquad (39)$$

$$s_{p,t}, z_{j,\beta,p}, w_{j,\beta,t}, x_{i.i.\tilde{\beta}'.\beta.t}, y_{\gamma,t}, v_{p,j,\beta,t} \in \{0,1\} \qquad \forall i, j, t, p, \gamma, \beta, \tilde{\beta}'$$

Verifying feasibility of a given solution is easy, but proving optimality is much harder.

LPs can be arbitrarily complex



$$s_{p,t} \leq \sum_{t'=1}^{t-\ell_{p-1}} s_{p-1,t'} \qquad \forall p, t \tag{25}$$

$$Z_{p,\gamma,\beta} \leq \sum_{j} z_{j,\beta,p} \cdot \alpha_{\gamma,j} \qquad \forall \beta, p, \gamma \tag{26}$$

$$z_{j,\beta,t} + s_{p,t} - 1 \leq w_{j,\beta,t} \qquad \forall j, \beta, t, p \tag{27}$$

$$\sum_{\beta} w_{j,\beta,t} = 1 \qquad \forall j, t \tag{28}$$

$$-\sum_{p} v_{p,j,\beta,t-\ell_p} + \sum_{i,\tilde{\beta}'} x_{j,i,\tilde{\beta}',\beta,t-G} = w_{j,\beta,t} - w_{j,\beta,t-1} \qquad \forall j, \beta, t \tag{29}$$

$$\sum_{p} v_{p,j,\beta,t-\ell_p} - \sum_{i,\beta'} x_{j,i,\tilde{\beta},\beta',t-G} = w_{j,\tilde{\beta},t} - w_{j,\tilde{\beta},t-1} \qquad \forall j, \tilde{\beta}, t \tag{30}$$

$$v_{p,j,\beta,t} \leq s_{p,t} \qquad \forall p, j, \beta, t \tag{31}$$

$$v_{p,j,\beta,t} \leq z_{j,\beta,p} \qquad \forall p, j, \beta, t \tag{32}$$

$$v_{p,j,\beta,t} \geq s_{p,t} + z_{j,\beta,p} - 1 \qquad \forall p, j, \beta, t \tag{33}$$

$$\sum_{t'=t-G+1}^{t+\rho} \sum_{\tilde{\beta}',\beta} x_{j,1,\tilde{\beta}',\beta,t'} \leq \alpha_{1,j} \cdot y_{1,t} + \alpha_{2,j} \cdot y_{2,t} \qquad \forall j, t \tag{34}$$

$$y_{1,t} + y_{2,t} = 1 \qquad \forall t \tag{35}$$

$$x_{j,i,\tilde{\beta}',\beta,t} \leq \alpha_{i-1,j} \qquad \forall i \in \{2,3\}\ \forall j, \tilde{\beta}', \beta, t \tag{36}$$

$$\sum_{j,\tilde{\beta}',\beta} \sum_{t'=t}^{t+G-1} x_{j,i,\tilde{\beta}',\beta,t'} \leq 1 \qquad \forall t, i \tag{37}$$

$$\sum_{i,\tilde{\beta}',\beta} \sum_{t'=t-G+1}^{t} x_{j,i,\tilde{\beta}',\beta,t'} + \sum_{p,\beta} \sum_{t'=t-R+1-\ell_p}^{t} v_{p,j,\beta,t'} \leq 1 \qquad \forall j, t \tag{38}$$

$$y_{\Gamma,1} = 1 \tag{39}$$

$$s_{p,t}, z_{j,\beta,p}, w_{j,\beta,t}, x_{i.i.\tilde{\beta}'.\beta.t}, y_{\gamma,t}, v_{p,j,\beta,t} \in \{0,1\} \qquad \forall i, j, t, p, \gamma, \beta, \tilde{\beta}'$$

Verifying feasibility of a given solution is easy, but proving optimality is much harder.

There is good news...!

Fundamental Result: Solving LPs is in P

Theorem

An optimal vector x^* to an LP can be computed in polynomial time with respect to the LP encoding size.

- Any problem that we can model as a polynomial-size LP can be solved in polynomial time. (black box)

Fundamental Result: Solving LPs is in P

Theorem

An optimal vector x^* to an LP can be computed in polynomial time with respect to the LP encoding size.

- Any problem that we can model as a polynomial-size LP can be solved in polynomial time. (black box)
- If our model has no feasible solution, the solver will inform us about this also in polynomial time.

Fundamental Result: Solving LPs is in P

Theorem

An optimal vector x^* to an LP can be computed in polynomial time with respect to the LP encoding size.

- Any problem that we can model as a polynomial-size LP can be solved in polynomial time. (black box)
- If our model has no feasible solution, the solver will inform us about this also in polynomial time.
- For optimization problems that arise from NP-hard problems, it is unlikely that there is a (polynomial-size) LP formulation.

Fundamental Result: Solving LPs is in P

Theorem

An optimal vector x^* to an LP can be computed in polynomial time with respect to the LP encoding size.

- Any problem that we can model as a polynomial-size LP can be solved in polynomial time. (black box)
- If our model has no feasible solution, the solver will inform us about this also in polynomial time.
- For optimization problems that arise from NP-hard problems, it is unlikely that there is a (polynomial-size) LP formulation.

Fundamental Result: Solving LPs is in P

Theorem

An optimal vector x^* to an LP can be computed in polynomial time with respect to the LP encoding size.

- Any problem that we can model as a polynomial-size LP can be solved in polynomial time. (black box)
- If our model has no feasible solution, the solver will inform us about this also in polynomial time.
- For optimization problems that arise from NP-hard problems, it is unlikely that there is a (polynomial-size) LP formulation.
  - still useful, as we will see

Integer Linear Programs

Many NP-hard problems are still linear optimization problems, but we require integral solution $x \in \mathbb{Z}$.

Integer Linear Programs

Many NP-hard problems are still linear optimization problems, but we require integral solution $x \in \mathbb{Z}$.

Integer Linear Program (ILP)

$$\begin{aligned} \min c^T x \\ \text{s.t. } Ax \geq b \\ x \in \mathbb{N} \end{aligned}$$

Integer Linear Programs

Many NP-hard problems are still linear optimization problems, but we require integral solution $x \in \mathbb{Z}$.

Integer Linear Program (ILP)

$$\begin{aligned} \min c^T x& \\ \text{s.t. } Ax &\geq b \\ x &\in \mathbb{N} \end{aligned}$$

- All problems in the course can be modeled as (I)LP.

Integer Linear Programs

Many NP-hard problems are still linear optimization problems, but we require integral solution $x \in \mathbb{Z}$.

Integer Linear Program (ILP)

$$\begin{aligned} \min c^T x \\ \text{s.t. } Ax \geq b \\ x \in \mathbb{N} \end{aligned}$$

- All problems in the course can be modeled as (I)LP.
- There is not just one model. Modeling is an art!

Integer Linear Programs

Many NP-hard problems are still linear optimization problems, but we require integral solution $x \in \mathbb{Z}$.

Integer Linear Program (ILP)

$$\begin{aligned} \min c^T x & \\ \text{s.t. } Ax &\geq b \\ x &\in \mathbb{N} \end{aligned}$$

- All problems in the course can be modeled as (I)LP.
- There is not just one model. Modeling is an art!
- The integrality constraint makes a huge difference.

Integer Linear Programs

Many NP-hard problems are still linear optimization problems, but we require integral solution $x \in \mathbb{Z}$.

Integer Linear Program (ILP)

$$\begin{aligned} \min c^T x& \\ \text{s.t. } Ax &\geq b \\ x &\in \mathbb{N} \end{aligned}$$

- All problems in the course can be modeled as (I)LP.
- There is not just one model. Modeling is an art!
- The integrality constraint makes a huge difference.

Theorem

It is NP-complete to decide whether a given integer linear program has a feasible solution.

Roadmap for today

1. Some modeling examples
2. Some background, geometric interpretation and important results
3. Usefulness of LP solutions as a benchmark

   → **LP relaxations**

Modeling Examples

Maximum Matching

Problem: Maximum Matching

Given a graph $G = (V, E)$, find a matching of maximum cardinality. (A matching is a set of edges $M \subseteq E$ such that no two edges in M are incident to the same node.)

Maximum Matching

Problem: Maximum Matching

Given a graph $G = (V, E)$, find a matching of maximum cardinality. (A matching is a set of edges $M \subseteq E$ such that no two edges in M are incident to the same node.)

Solution variable x_e indicating whether an edge $e \in E$ is part of the matching M.

Maximum Matching

Problem: Maximum Matching

Given a graph $G = (V, E)$, find a matching of maximum cardinality. (A matching is a set of edges $M \subseteq E$ such that no two edges in M are incident to the same node.)

Solution variable x_e indicating whether an edge $e \in E$ is part of the matching M. $\longrightarrow x_e \in \{0, 1\}$!

Maximum Matching

Problem: Maximum Matching

Given a graph $G = (V, E)$, find a matching of maximum cardinality. (A matching is a set of edges $M \subseteq E$ such that no two edges in M are incident to the same node.)

Solution variable x_e indicating whether an edge $e \in E$ is part of the matching M. $\longrightarrow$ $x_e \in \{0, 1\}$!

ILP formulation

Maximum Matching

Problem: Maximum Matching

Given a graph $G = (V, E)$, find a matching of maximum cardinality. (A matching is a set of edges $M \subseteq E$ such that no two edges in M are incident to the same node.)

Solution variable x_e indicating whether an edge $e \in E$ is part of the matching M. $\longrightarrow x_e \in \{0, 1\}$!

ILP formulation

$$\begin{aligned} \max \sum_{e \in E} x_e & & \\ \text{s.t.} \sum_{e \in \delta(v)} x_e \le 1, & \qquad \text{for all } v \in V \\ x_e \in \{0, 1\}, & \qquad \text{for all } e \in E. \end{aligned}$$

Minimum-Cost Flow

Problem: Minimum-Cost Flow

Given a network $\mathcal{N} = (V, E, c, b, p)$ consisting of a directed graph (V, E) with edge capacities $c_e \in \mathbb{R}_{\geq 0}$, costs $p_e \in \mathbb{R}_{\geq 0}$, and a supply/demand function $b : V \to \mathbb{R}$, find a feasible flow that satisfies all supply to demand and minimizes the total cost.

Minimum-Cost Flow

Problem: Minimum-Cost Flow

Given a network $\mathcal{N} = (V, E, c, b, p)$ consisting of a directed graph (V, E) with edge capacities $c_e \in \mathbb{R}_{\geq 0}$, costs $p_e \in \mathbb{R}_{\geq 0}$, and a supply/demand function $b : V \to \mathbb{R}$, find a feasible flow that satisfies all supply to demand and minimizes the total cost.

Flow variable x_e indicating how much flow is sent along arc $e \in E$.

Minimum-Cost Flow

Problem: Minimum-Cost Flow

Given a network $\mathcal{N} = (V, E, c, b, p)$ consisting of a directed graph (V, E) with edge capacities $c_e \in \mathbb{R}_{\geq 0}$, costs $p_e \in \mathbb{R}_{\geq 0}$, and a supply/demand function $b : V \to \mathbb{R}$, find a feasible flow that satisfies all supply to demand and minimizes the total cost.

Flow variable x_e indicating how much flow is sent along arc $e \in E$.

LP formulation

Universität
Bremen

Minimum-Cost Flow

Problem: Minimum-Cost Flow

Given a network $\mathcal{N} = (V, E, c, b, p)$ consisting of a directed graph (V, E) with edge capacities $c_e \in \mathbb{R}_{\geq 0}$, costs $p_e \in \mathbb{R}_{\geq 0}$, and a supply/demand function $b : V \to \mathbb{R}$, find a feasible flow that satisfies all supply to demand and minimizes the total cost.

Flow variable x_e indicating how much flow is sent along arc $e \in E$.

LP formulation

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e x_e & & \\ \text{s.t.} \quad & \sum_{e \in \delta^+(v)} x_e - \sum_{e \in \delta^-(v)} x_e = b(v), & & \text{for all } v \in V \\ & x_e \leq c_e, & & \text{for all } e \in E \\ & 0 \leq x_e, & & \text{for all } e \in E \end{aligned}$$

Universität Bremen

The Knapsack Problem

Problem: (Unbounded) Knapsack Problem

There is given a capacity K and an unlimited supply of items of n different types. An item of type i has weight w_i and profit v_i.

Task: Select a subset of items of maximum total profit such that the total weight does not exceed the capacity bound K.

The Knapsack Problem

Problem: (Unbounded) Knapsack Problem

There is given a capacity K and an unlimited supply of items of n different types. An item of type i has weight w_i and profit v_i.

Task: Select a subset of items of maximum total profit such that the total weight does not exceed the capacity bound K.

ILP formulation:

$$\begin{aligned} \max \quad & v_1 \cdot x_1 + v_2 \cdot x_2 + \ldots + v_n \cdot x_n \\ s.t. \quad & w_1 \cdot x_1 + w_2 \cdot x_2 + \ldots + w_n \cdot x_n \leq K \\ & x_i \geq 0 \qquad i \in \{1, \ldots, n\} \\ & x_i \in \mathbb{Z} \qquad i \in \{1, \ldots, n\} \end{aligned}$$

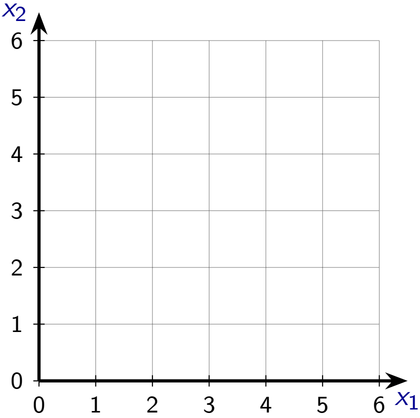
Linear Programming

A Geometrical View

The Real, *n*-dimensional Space

$\mathbb{R}^n := \{x = (x_1, \ldots, x_n) : x_1, \ldots, x_n \in \mathbb{R}\}$

Elements $x \in \mathbb{R}^n$ can be seen as

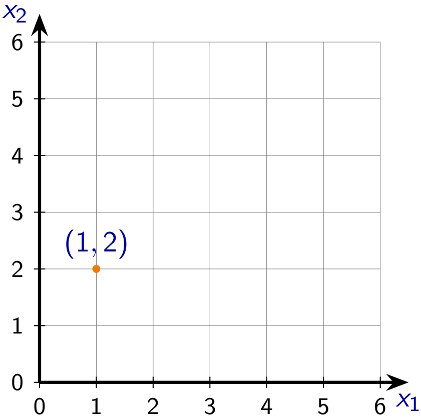


The Real, *n*-dimensional Space

$\mathbb{R}^n := \{x = (x_1, \ldots, x_n) : x_1, \ldots, x_n \in \mathbb{R}\}$

Elements $x \in \mathbb{R}^n$ can be seen as

- Points

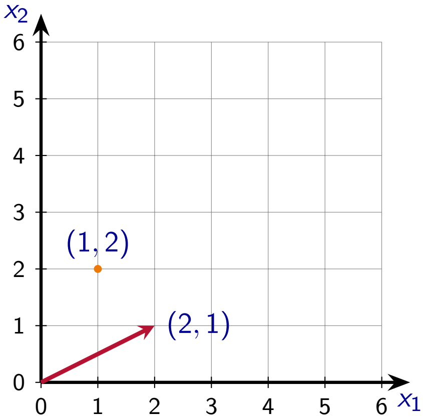


The Real, *n*-dimensional Space

$\mathbb{R}^n := \{x = (x_1, \ldots, x_n) : x_1, \ldots, x_n \in \mathbb{R}\}$

Elements $x \in \mathbb{R}^n$ can be seen as

- Points
- Vectors



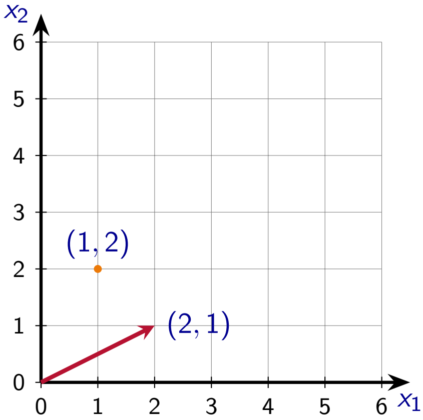
The Real, *n*-dimensional Space

$\mathbb{R}^n := \{x = (x_1, \ldots, x_n) : x_1, \ldots, x_n \in \mathbb{R}\}$

Elements $x \in \mathbb{R}^n$ can be seen as

- Points
- Vectors

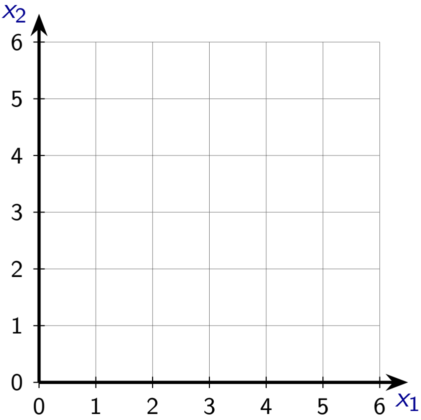
An n-tuple may represent the net profit of n different goods, or their inventory level, or the cost of production, etc.



Linear Equations

Let $a \in \mathbb{R}^n$ be a profit vector.

If you want the profit to be exactly b, how much should you produce?



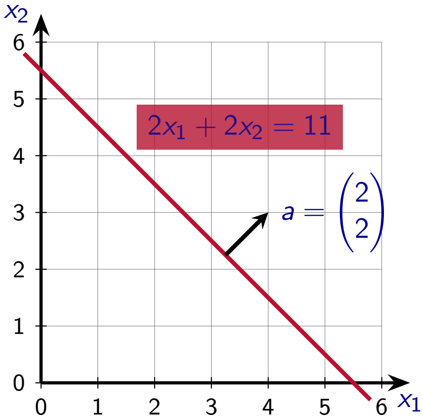
Linear Equations

Let $a \in \mathbb{R}^n$ be a profit vector.

If you want the profit to be exactly b, how much should you produce?

The answer is given as the set $\{x \in \mathbb{R}^n : a_1 x_1 + \ldots + a_n x_n = b\}$.

This is a hyperplane in $\mathbb{R}^n$.



Linear Equations

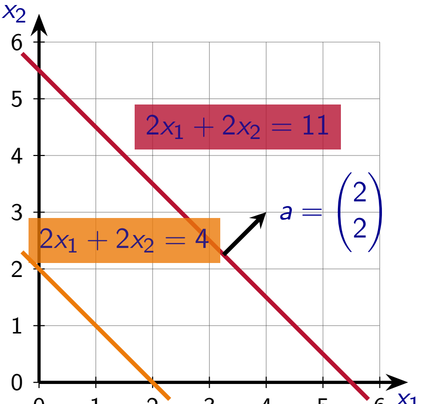
Let $a \in \mathbb{R}^n$ be a profit vector.

If you want the profit to be exactly b, how much should you produce?

The answer is given as the set $\{x \in \mathbb{R}^n : a_1x_1 + \ldots + a_nx_n = b\}$.

This is a hyperplane in $\mathbb{R}^n$.

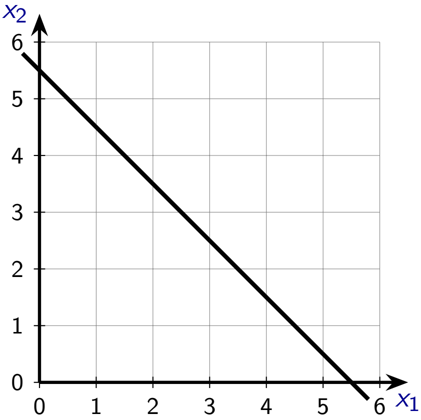
Changing the right hand side b corresponds to "moving" the hyperplane along the vector a.



Linear Inequalities

Let $a \in \mathbb{R}^n$ be a profit vector.

If you want to earn at least (at most) b, how much should you produce?



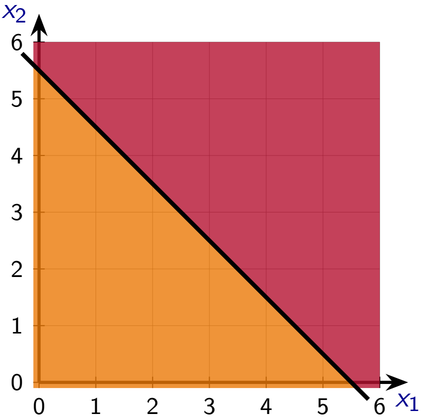
Linear Inequalities

Let $a \in \mathbb{R}^n$ be a profit vector.

If you want to earn at least (at most) b, how much should you produce?

The answer is given as the set $\{x \in \mathbb{R}^n : a_1x_1 + \ldots + a_nx_n \geq (\leq) b\}$.

This is a halfspace in $\mathbb{R}^n$.



Example: Pizza and Lasagna

Modelling $x_1 =$ Number of pizzas

$x_2 =$ Number of lasagnas

$$\begin{aligned} \max\quad & 8x_1 + 7x_2 & & \text{(Profit)} \\ s.t.\quad & 2x_1 + 3x_2 \leq 18 & & \text{(Tomatoes)} \\ & 4x_1 + 3x_2 \leq 24 & & \text{(Cheese)} \\ & x_1, x_2 \geq 0 & & \end{aligned}$$

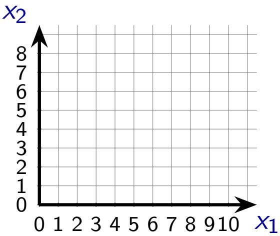
Example: Pizza and Lasagna

Modelling x_1 = Number of pizzas

x_2 = Number of lasagnas

$$\begin{aligned} \max \quad & 8x_1 + 7x_2 & & \text{(Profit)} \\ s.t. \quad & 2x_1 + 3x_2 \leq 18 & & \text{(Tomatoes)} \\ & 4x_1 + 3x_2 \leq 24 & & \text{(Cheese)} \\ & x_1, x_2 \geq 0 & & \end{aligned}$$

Graphical Solution



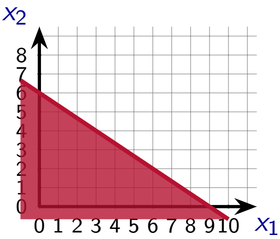
Example: Pizza and Lasagna

Modelling $x_1 =$ Number of pizzas

$x_2 =$ Number of lasagnas

$$\begin{aligned} \max \quad & 8x_1 + 7x_2 & & \text{(Profit)} \\ s.t. \quad & 2x_1 + 3x_2 \leq 18 & & \text{(Tomatoes)} \\ & 4x_1 + 3x_2 \leq 24 & & \text{(Cheese)} \\ & x_1, x_2 \geq 0 & & \end{aligned}$$

Graphical Solution



Which points satisfy $2x_1 + 3x_2 \leq 18$?

- Points on $2x_1 + 3x_2 = 18$.
- As well as points below

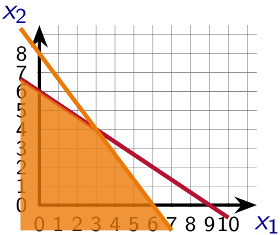
Example: Pizza and Lasagna

Modelling $x_1 =$ Number of pizzas

$x_2 =$ Number of lasagnas

$$
\begin{array}{rlr}
\max & 8x_1 + 7x_2 & \text{(Profit)} \\
s.t. & 2x_1 + 3x_2 \leq 18 & \text{(Tomatoes)} \\
 & 4x_1 + 3x_2 \leq 24 & \text{(Cheese)} \\
 & x_1, x_2 \geq 0 &
\end{array}
$$

Graphical Solution



Which points satisfy $4x_1 + 3x_2 \leq 24$?

- Points on $4x_1 + 3x_2 = 24$.
- As well as points below

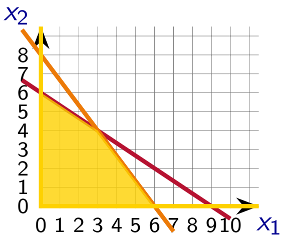
Example: Pizza and Lasagna

Modelling $x_1 =$ Number of pizzas

$x_2 =$ Number of lasagnas

$$\begin{aligned} \max \quad & 8x_1 + 7x_2 && \text{(Profit)} \\ s.t. \quad & 2x_1 + 3x_2 \leq 18 && \text{(Tomatoes)} \\ & 4x_1 + 3x_2 \leq 24 && \text{(Cheese)} \\ & x_1, x_2 \geq 0 \end{aligned}$$

Graphical Solution



Which points satisfy $x_1, x_2 \geq 0$?

- Points on $x_1 = 0$ or $x_2 = 0$.
- As well as points below

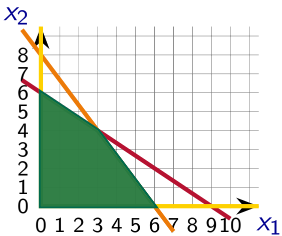
Example: Pizza and Lasagna

Modelling x_1 = Number of pizzas

x_2 = Number of lasagnas

$$\begin{aligned} \max\quad & 8x_1 + 7x_2 & & \text{(Profit)} \\ s.t.\quad & 2x_1 + 3x_2 \leq 18 & & \text{(Tomatoes)} \\ & 4x_1 + 3x_2 \leq 24 & & \text{(Cheese)} \\ & x_1, x_2 \geq 0 & & \end{aligned}$$

Graphical Solution



Which points are optimal?

- Points on $8x_1 + 7x_2 = z$ have objective value z
- Start with $z = 0$.

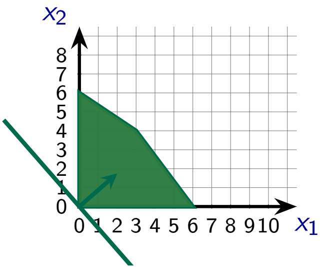
Example: Pizza and Lasagna

Modelling x_1 = Number of pizzas

x_2 = Number of lasagnas

$$
\begin{aligned}
\max\quad & 8x_1 + 7x_2 & & \text{(Profit)}\\
s.t.\quad & 2x_1 + 3x_2 \leq 18 & & \text{(Tomatoes)}\\
& 4x_1 + 3x_2 \leq 24 & & \text{(Cheese)}\\
& x_1, x_2 \geq 0 & &
\end{aligned}
$$

Graphical Solution



Which points are optimal?

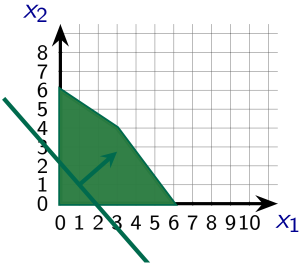
- Points on $8x_1 + 7x_2 = z$ have objective value z
- Start with $z = 0$.
- Shift until last feasible point.

Example: Pizza and Lasagna

Modelling $x_1 =$ Number of pizzas
$x_2 =$ Number of lasagnas

$$\begin{aligned} \max\quad & 8x_1 + 7x_2 & & \text{(Profit)} \\ s.t.\quad & 2x_1 + 3x_2 \leq 18 & & \text{(Tomatoes)} \\ & 4x_1 + 3x_2 \leq 24 & & \text{(Cheese)} \\ & x_1, x_2 \geq 0 & & \end{aligned}$$

Graphical Solution



Which points are optimal?

- Points on $8x_1 + 7x_2 = z$ have objective value z
- Start with $z = 0$.
- Shift until last feasible point.

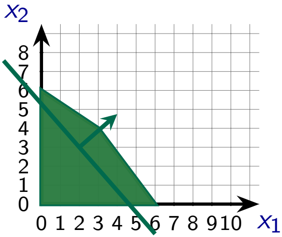
Example: Pizza and Lasagna

Modelling x_1 = Number of pizzas

x_2 = Number of lasagnas

$$\begin{aligned} \max \quad & 8x_1 + 7x_2 & & \text{(Profit)} \\ s.t. \quad & 2x_1 + 3x_2 \leq 18 & & \text{(Tomatoes)} \\ & 4x_1 + 3x_2 \leq 24 & & \text{(Cheese)} \\ & x_1, x_2 \geq 0 & & \end{aligned}$$

Graphical Solution



Which points are optimal?

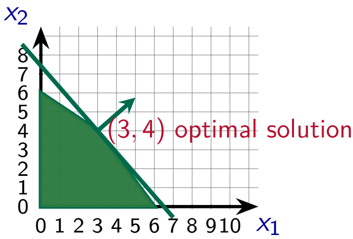
- Points on $8x_1 + 7x_2 = z$ have objective value z
- Start with $z = 0$.
- Shift until last feasible point.

Example: Pizza and Lasagna

Modelling $x_1 =$ Number of pizzas
$x_2 =$ Number of lasagnas

$$\begin{aligned} \max \quad & 8x_1 + 7x_2 & & \text{(Profit)} \\ s.t. \quad & 2x_1 + 3x_2 \leq 18 & & \text{(Tomatoes)} \\ & 4x_1 + 3x_2 \leq 24 & & \text{(Cheese)} \\ & x_1, x_2 \geq 0 & & \end{aligned}$$

Graphical Solution



Which points are optimal?

- Points on $8x_1 + 7x_2 = z$ have objective value z
- Start with $z = 0$.
- Shift until last feasible point.

Recipe for a graphical solution in $\mathbb{R}^2$

1. Determine set X of all feasible solutions.

Recipe for a graphical solution in $\mathbb{R}^2$

1. Determine set X of all feasible solutions.
   - For each constraint $a_1x_1 + a_2x_2 \leq b$ draw $a_1x_1 + a_2x_2 = b$

Recipe for a graphical solution in $\mathbb{R}^2$

1. Determine set X of all feasible solutions.
   - For each constraint $a_1x_1 + a_2x_2 \leq b$ draw $a_1x_1 + a_2x_2 = b$
   - and determine feasible half space (e.g. by checking a specific point).

Recipe for a graphical solution in $\mathbb{R}^2$

1. Determine set X of all feasible solutions.
   - For each constraint $a_1x_1 + a_2x_2 \leq b$ draw $a_1x_1 + a_2x_2 = b$
   - and determine feasible half space (e.g. by checking a specific point).
   - X is intersection of all halfspaces.

Recipe for a graphical solution in $\mathbb{R}^2$

1. Determine set X of all feasible solutions.
   - For each constraint $a_1x_1 + a_2x_2 \leq b$ draw $a_1x_1 + a_2x_2 = b$
   - and determine feasible half space (e.g. by checking a specific point).
   - X is intersection of all halfspaces.

2. Draw $c_1x + c_2x_2 = z$, e.g. for $z = 0$, to determine all points with objective value z.

Recipe for a graphical solution in $\mathbb{R}^2$

1. Determine set X of all feasible solutions.
   - For each constraint $a_1x_1 + a_2x_2 \leq b$ draw $a_1x_1 + a_2x_2 = b$
   - and determine feasible half space (e.g. by checking a specific point).
   - X is intersection of all halfspaces.
2. Draw $c_1x + c_2x_2 = z$, e.g. for $z = 0$, to determine all points with objective value z.
3. Shift objective function up to the last (first) feasible point.

Recipe for a graphical solution in $\mathbb{R}^2$

1. Determine set X of all feasible solutions.
   - For each constraint $a_1x_1 + a_2x_2 \leq b$ draw $a_1x_1 + a_2x_2 = b$
   - and determine feasible half space (e.g. by checking a specific point).
   - X is intersection of all halfspaces.

2. Draw $c_1x + c_2x_2 = z$, e.g. for $z = 0$, to determine all points with objective value z.

3. Shift objective function up to the last (first) feasible point.

4. Read optimal solution $x^* = (x_1^*, x_2^*)$.

Recipe for a graphical solution in $\mathbb{R}^2$

1. Determine set X of all feasible solutions.
   - For each constraint $a_1x_1 + a_2x_2 \leq b$ draw $a_1x_1 + a_2x_2 = b$
   - and determine feasible half space (e.g. by checking a specific point).
   - X is intersection of all halfspaces.

2. Draw $c_1x + c_2x_2 = z$, e.g. for $z = 0$, to determine all points with objective value z.

3. Shift objective function up to the last (first) feasible point.

4. Read optimal solution $x^* = (x_1^*, x_2^*)$.

5. Compute optimal objective function value $z^* = z(x_1^*, x_2^*)$.

Example: Pizza and Lasagne

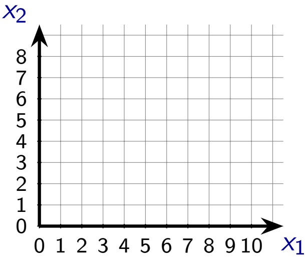
LP Model

x_1 = number of produced pizzas

x_2 = number of produced lasagne

$$
\begin{array}{llll}
\max & 8x_1 + 7x_2 & & \text{(profit)} \\
s.t. & 2x_1 + 3x_2 & \le 18 & \text{(tomato)} \\
 & 4x_1 + 3x_2 & \le 24 & \text{(cheese)} \\
 & x_1, x_2 \ge 0 & &
\end{array}
$$

Graphical representation and solution



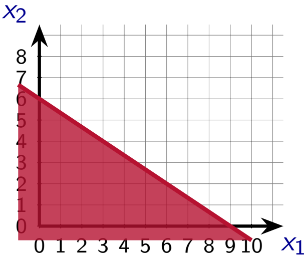
Example: Pizza and Lasagne

LP Model

$x_1 =$ number of produced pizzas
$x_2 =$ number of produced lasagne

$$\begin{array}{rlr} \max & 8x_1 + 7x_2 & \text{(profit)} \\ s.t. & 2x_1 + 3x_2 \leq 18 & \text{(tomato)} \\ & 4x_1 + 3x_2 \leq 24 & \text{(cheese)} \\ & x_1, x_2 \geq 0 & \end{array}$$

Graphical representation and solution



Example: Pizza and Lasagne

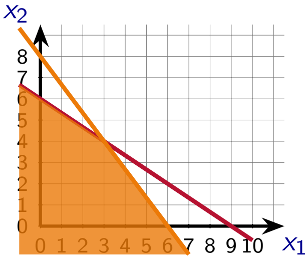
LP Model

x_1 = number of produced pizzas

x_2 = number of produced lasagne

$$\begin{aligned} \max \quad & 8x_1 + 7x_2 & & \text{(profit)} \\ s.t. \quad & 2x_1 + 3x_2 \leq 18 & & \text{(tomato)} \\ & 4x_1 + 3x_2 \leq 24 & & \text{(cheese)} \\ & x_1, x_2 \geq 0 & & \end{aligned}$$

Graphical representation and solution



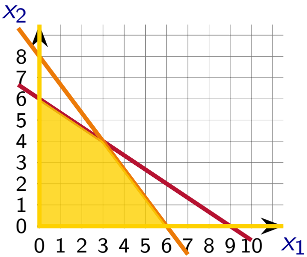
Example: Pizza and Lasagne

LP Model $x_1 =$ number of produced pizzas

$x_2 =$ number of produced lasagne

$$\begin{array}{rlr} \max & 8x_1 + 7x_2 & \text{(profit)} \\ s.t. & 2x_1 + 3x_2 \leq 18 & \text{(tomato)} \\ & 4x_1 + 3x_2 \leq 24 & \text{(cheese)} \\ & x_1, x_2 \geq 0 & \end{array}$$

Graphical representation and solution



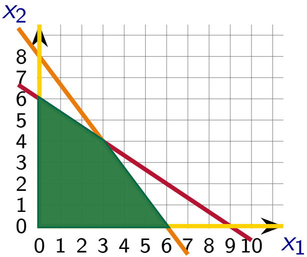
Example: Pizza and Lasagne

LP Model

x_1 = number of produced pizzas
x_2 = number of produced lasagne

$$\begin{array}{rlr} \max & 8x_1 + 7x_2 & \text{(profit)} \\ s.t. & 2x_1 + 3x_2 \leq 18 & \text{(tomato)} \\ & 4x_1 + 3x_2 \leq 24 & \text{(cheese)} \\ & x_1, x_2 \geq 0 & \end{array}$$

Graphical representation and solution



Example: Pizza and Lasagne

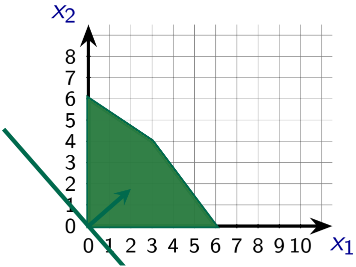
LP Model

x_1 = number of produced pizzas

x_2 = number of produced lasagne

$$\begin{aligned} \max \quad & 8x_1 + 7x_2 & & \text{(profit)} \\ s.t. \quad & 2x_1 + 3x_2 \leq 18 & & \text{(tomato)} \\ & 4x_1 + 3x_2 \leq 24 & & \text{(cheese)} \\ & x_1, x_2 \geq 0 & & \end{aligned}$$

Graphical representation and solution



Example: Pizza and Lasagne

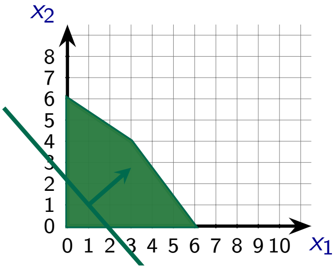
LP Model

x_1 = number of produced pizzas

x_2 = number of produced lasagne

$$\begin{array}{llr} \max & 8x_1 + 7x_2 & \text{(profit)} \\ s.t. & 2x_1 + 3x_2 \leq 18 & \text{(tomato)} \\ & 4x_1 + 3x_2 \leq 24 & \text{(cheese)} \\ & x_1, x_2 \geq 0 & \end{array}$$

Graphical representation and solution

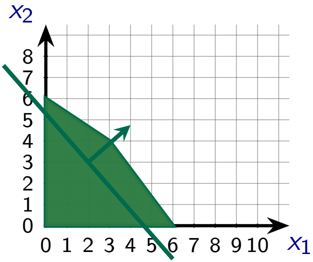


Example: Pizza and Lasagne

LP Model $x_1 =$ number of produced pizzas
$x_2 =$ number of produced lasagne

$$\begin{array}{rrcll} \max & 8x_1 + 7x_2 & & & \text{(profit)} \\ s.t. & 2x_1 + 3x_2 & \leq & 18 & \text{(tomato)} \\ & 4x_1 + 3x_2 & \leq & 24 & \text{(cheese)} \\ & x_1, x_2 & \geq & 0 & \end{array}$$

Graphical representation and solution



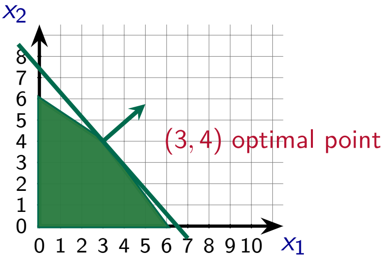
Example: Pizza and Lasagne

LP Model

x_1 = number of produced pizzas
x_2 = number of produced lasagne

$$\begin{array}{rlr} \max & 8x_1 + 7x_2 & \text{(profit)} \\ s.t. & 2x_1 + 3x_2 \leq 18 & \text{(tomato)} \\ & 4x_1 + 3x_2 \leq 24 & \text{(cheese)} \\ & x_1, x_2 \geq 0 & \end{array}$$

Graphical representation and solution



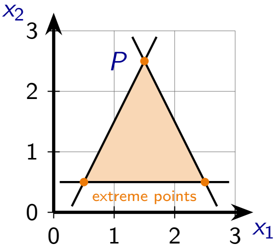
Extreme points

Definition

Let $M \neq \emptyset$ be a convex set. A point $x \in M$ is an extreme point of M if we cannot find two points $y, z \in M \setminus \{x\}$ and a scalar $\lambda \in (0,1)$ such that

$$x = \lambda y + (1-\lambda) z.$$

$$P = \left\{ x \in \mathbb{R}^2 : \begin{array}{rl} x_2 \geq & \frac{1}{2} \\ 2x_1 + x_2 \leq & \frac{11}{2} \\ -2x_1 + x_2 \leq & -\frac{1}{2} \end{array} \right\}$$



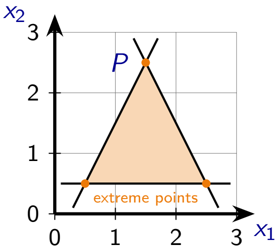
Extreme points

Definition

Let $M \neq \emptyset$ be a convex set. A point $x \in M$ is an extreme point of M if we cannot find two points $y, z \in M \setminus \{x\}$ and a scalar $\lambda \in (0, 1)$ such that

$$x = \lambda y + (1 - \lambda)z.$$

$$P = \left\{ x \in \mathbb{R}^2 : \begin{array}{rr} x_2 \geq & \frac{1}{2} \\ 2x_1 + x_2 \leq & \frac{11}{2} \\ -2x_1 + x_2 \leq & -\frac{1}{2} \end{array} \right\}$$



- If M has finitely many extreme points, they are called vertices.
- Convex sets with finitely many extreme points are polytopes (bounded) or polyhedra (unbounded).

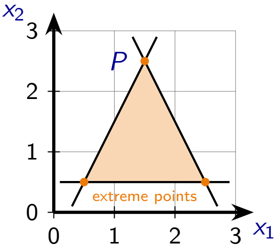
Extreme points

Definition

Let $M \neq \emptyset$ be a convex set. A point $x \in M$ is an extreme point of M if we cannot find two points $y, z \in M \setminus \{x\}$ and a scalar $\lambda \in (0,1)$ such that

$$x = \lambda y + (1-\lambda) z.$$

$$P = \left\{ x \in \mathbb{R}^2 : \begin{array}{rl} x_2 \geq & \frac{1}{2} \\ 2x_1 + x_2 \leq & \frac{11}{2} \\ -2x_1 + x_2 \leq & -\frac{1}{2} \end{array} \right\}$$



- If M has finitely many extreme points, they are called vertices.
- Convex sets with finitely many extreme points are polytopes (bounded) or polyhedra (unbounded).
- Examples of convex sets with infinitely many extreme points: circle or ball

Polytopes and Polyhedra

Definition (equivalent characterization)

- $P \subseteq \mathbb{R}^n$ is a polyhedron if P is the intersection of finitely many closed halfspaces.
- A bounded polyhedron is called polytope.

Polytopes and Polyhedra

Definition (equivalent characterization)

- $P \subseteq \mathbb{R}^n$ is a polyhedron if P is the intersection of finitely many closed halfspaces.
- A bounded polyhedron is called polytope.

Polyhedra and LPs

- The set $\{x \in \mathbb{R}^n \mid Ax \leq (\geq) b\}$ is called polyhedron.

Polytopes and Polyhedra

Definition (equivalent characterization)

- $P \subseteq \mathbb{R}^n$ is a polyhedron if P is the intersection of finitely many closed halfspaces.
- A bounded polyhedron is called polytope.

Polyhedra and LPs

- The set $\{x \in \mathbb{R}^n \,|\, Ax \leq (\geq) b\}$ is called polyhedron.
- The set of feasible solutions of an LP is a polyhedron.

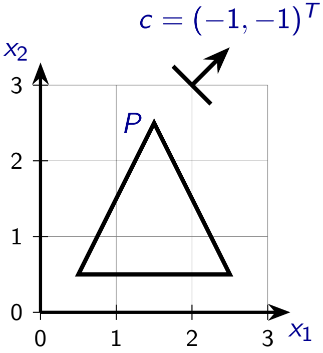
Optimal Solutions

Let $P = P(A, b) = \{x \in \mathbb{R}^n \mid Ax \leq b\} \neq \emptyset$ a polyhedron.

Theorem

If the optimization problem $\min\{c^T x \mid x \in P\}$ has an optimal solution, then at least one optimal solution is attained at a vertex of P.

$$\begin{array}{lrcrcr}
\max & -x_1 & - & x_2 & & \\
\text{s.t.} & & & x_2 & \geq & \frac{1}{2} \\
 & 2x_1 & + & x_2 & \leq & \frac{11}{2} \\
 & -2x_1 & + & x_2 & \leq & -\frac{1}{2}
\end{array}$$



Optimal Solutions

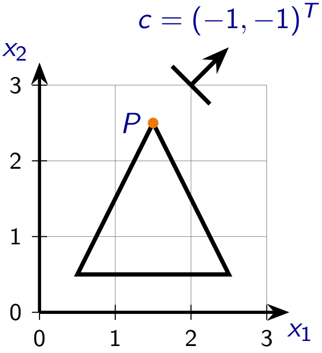
Let $P = P(A, b) = \{x \in \mathbb{R}^n \mid Ax \leq b\} \neq \emptyset$ a polyhedron.

Theorem

If the optimization problem $\min\{c^T x \mid x \in P\}$ has an optimal solution, then at least one optimal solution is attained at a vertex of P.

$$\begin{array}{lrcrcr} \max & -x_1 & - & x_2 & & \\ \text{s.t.} & & & x_2 & \geq & \frac{1}{2} \\ & 2x_1 & + & x_2 & \leq & \frac{11}{2} \\ & -2x_1 & + & x_2 & \leq & -\frac{1}{2} \end{array}$$

optimal solution



Idea for an Algorithm

Idea: Enumerate all vertices of $P(A, b)$ and choose the best.

Idea for an Algorithm

Idea: Enumerate all vertices of $P(A, b)$ and choose the best.

- Polyhedron P has finitely many extreme points by definition.

Idea for an Algorithm

Idea: Enumerate all vertices of $P(A, b)$ and choose the best.

- Polyhedron P has finitely many extreme points by definition.
- P is defined by m inequalities. For each vertex, n linearly independent inequalities have to be active.

$$\Rightarrow \text{ Number of vertices } \leq \binom{m}{n} \qquad \text{(Huge!)}$$

Idea for an Algorithm

Idea: Enumerate all vertices of $P(A, b)$ and choose the best.

- Polyhedron P has finitely many extreme points by definition.
- P is defined by m inequalities. For each vertex, n linearly independent inequalities have to be active.

$$\Rightarrow \text{ Number of vertices } \leq \binom{m}{n} \qquad\qquad \text{(Huge!)}$$

Example: Unit cube $P = \{x \in \mathbb{R}^n \,|\, 0 \leq x_i \leq 1, i = 1, \ldots, n\}$

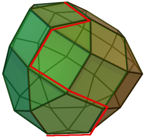
Number of inequalities: $m = 2n$
Number of vertices: 2^n

Enumeration is no option! Smarter algorithms needed.

Solving LPs (First Algorithm)

Simplex method (George Dantzig, 1947)

- First general method for solving LPs
- Starts from some vertex and iteratively moves to an adjacent vertex with better objective function value.
- In general, it does not run in polynomial time.
- However, works very well and fast in practice.



Solving LPs in Polynomial Time

Ellipsoid method (Leonid Khachiyan, 1979)

- First algorithm that solves LPs in polynomial time.
- Builds in earlier work by Shor, Yudin, and Nemirovskii
- Intuitive idea:
  - Binary search for optimal objective value z.
  - Add constraint for target value $c^T x \leq z$.
  - Resulting problem: decide whether a given polytope P is non-empty.
  - Generates sequence of ellipsoids of decreasing volume containing P.
  - For each ellipsoid: verify whether center point is in P (done) or find a separating hyperplane. (separation problem)
- Considered as computationally impractical
- Its consequences in linear optimization are enormous!

Solving LPs in Polynomial Time

Ellipsoid method (Leonid Khachiyan, 1979)

- First algorithm that solves LPs in polynomial time.
- Builds in earlier work by Shor, Yudin, and Nemirovskii
- Intuitive idea:
  - Binary search for optimal objective value z.
  - Add constraint for target value $c^T x \leq z$.
  - Resulting problem: decide whether a given polytope P is non-empty.
  - Generates sequence of ellipsoids of decreasing volume containing P.
  - For each ellipsoid: verify whether center point is in P (done) or find a separating hyperplane. (separation problem)
- Considered as computationally impractical
- Its consequences in linear optimization are enormous!

Interior point methods

- By now several poly-time interior point methods are known.

Equivalence of Separation and Optimization

Separation problem: Given a point x and a polyhedron P, decide whether $x \in P$. If $c \notin P$ then give a certificate, i.e., a separating hyperplane (an inequality satisfied by all points in P but not by x).

Equivalence of Separation and Optimization

Separation problem: Given a point x and a polyhedron P, decide whether $x \in P$. If $c \notin P$ then give a certificate, i.e., a separating hyperplane (an inequality satisfied by all points in P but not by x).

Careful:

- Easy, since we can simply check all constraints.

Equivalence of Separation and Optimization

Separation problem: Given a point x and a polyhedron P, decide whether $x \in P$. If $c \notin P$ then give a certificate, i.e., a separating hyperplane (an inequality satisfied by all points in P but not by x).

Careful:

- Easy, since we can simply check all constraints.
- Running time is polynomial in LP size.

Equivalence of Separation and Optimization

Separation problem: Given a point x and a polyhedron P, decide whether $x \in P$. If $c \notin P$ then give a certificate, i.e., a separating hyperplane (an inequality satisfied by all points in P but not by x).

Careful:

- Easy, since we can simply check all constraints.
- Running time is polynomial in LP size.
- What if the number of constraints is exponential?

Equivalence of Separation and Optimization

Separation problem: Given a point x and a polyhedron P, decide whether $x \in P$. If $c \notin P$ then give a certificate, i.e., a separating hyperplane (an inequality satisfied by all points in P but not by x).

Careful:

- Easy, since we can simply check all constraints.
- Running time is polynomial in LP size.
- What if the number of constraints is exponential?

Equivalence of Separation and Optimization

Separation problem: Given a point x and a polyhedron P, decide whether $x \in P$. If $c \notin P$ then give a certificate, i.e., a separating hyperplane (an inequality satisfied by all points in P but not by x).

Careful:

- Easy, since we can simply check all constraints.
- Running time is polynomial in LP size.
- What if the number of constraints is exponential?

Theorem (informally)

Solving the separation problem in polynomial time is equivalent to solving the optimization problem in polynomial time.

Very powerful theorem! (By the Ellipsoid method.)

Example: Minimum Cut

Given $G = (V, E)$ and $s, t \in V$, let $\mathcal{P}$ be the collection of all s-t-paths in G.

Example: Minimum Cut

Given $G = (V, E)$ and $s, t \in V$, let $\mathcal{P}$ be the collection of all s-t-paths in G. The problem of finding a minimum s-t-cut in G is

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e \cdot x_e & \\ s.t. \quad & \sum_{e \in P} x_e \geq 1, & \forall P \in \mathcal{P} \\ & 0 \leq x_e \leq 1, & \forall e \in E \end{aligned}$$

Example: Minimum Cut

Given $G = (V, E)$ and $s, t \in V$, let $\mathcal{P}$ be the collection of all s-t-paths in G. The problem of finding a minimum s-t-cut in G is

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e \cdot x_e & \\ s.t. \quad & \sum_{e \in P} x_e \geq 1, & \forall P \in \mathcal{P} \\ & 0 \leq x_e \leq 1, & \forall e \in E \end{aligned}$$

- Solving this LP gives a minimum s-t-cut. Not obvious, since it is not clear that extreme points are integral. (totally unimodularity)

Universität Bremen

Example: Minimum Cut

Given $G = (V, E)$ and $s, t \in V$, let $\mathcal{P}$ be the collection of all s-t-paths in G. The problem of finding a minimum s-t-cut in G is

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e \cdot x_e & \\ s.t. \quad & \sum_{e \in P} x_e \geq 1, & \forall P \in \mathcal{P} \\ & 0 \leq x_e \leq 1, & \forall e \in E \end{aligned}$$

- Solving this LP gives a minimum s-t-cut. Not obvious, since it is not clear that extreme points are integral. (totally unimodularity)
- Can we solve the LP in polynomial time?

Example: Minimum Cut

Given $G = (V, E)$ and $s, t \in V$, let $\mathcal{P}$ be the collection of all s-t-paths in G. The problem of finding a minimum s-t-cut in G is

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e \cdot x_e & \\ s.t. \quad & \sum_{e \in P} x_e \geq 1, & \forall P \in \mathcal{P} \\ & 0 \leq x_e \leq 1, & \forall e \in E \end{aligned}$$

- Solving this LP gives a minimum s-t-cut. Not obvious, since it is not clear that extreme points are integral. (totally unimodularity)
- Can we solve the LP in polynomial time? Exponential number of inequalities! Running time polyn. in # variables and constraints, not in input!

Example: Minimum Cut

Given $G = (V, E)$ and $s, t \in V$, let $\mathcal{P}$ be the collection of all s-t-paths in G. The problem of finding a minimum s-t-cut in G is

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e \cdot x_e & \\ s.t. \quad & \sum_{e \in P} x_e \geq 1, & \forall P \in \mathcal{P} \\ & 0 \leq x_e \leq 1, & \forall e \in E \end{aligned}$$

- Solving this LP gives a minimum s-t-cut. Not obvious, since it is not clear that extreme points are integral. (totally unimodularity)
- Can we solve the LP in polynomial time? Exponential number of inequalities! Running time polyn. in # variables and constraints, not in input!
- Separation problem: For $\bar{x} \in \mathbb{R}^{|E|}$, check if $\sum_{e \in P} \bar{x}_e \geq 1, \forall P \in \mathcal{P}$?

Example: Minimum Cut

Given $G = (V, E)$ and $s, t \in V$, let $\mathcal{P}$ be the collection of all s-t-paths in G. The problem of finding a minimum s-t-cut in G is

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e \cdot x_e & \\ s.t. \quad & \sum_{e \in P} x_e \geq 1, & \forall P \in \mathcal{P} \\ & 0 \leq x_e \leq 1, & \forall e \in E \end{aligned}$$

- Solving this LP gives a minimum s-t-cut. Not obvious, since it is not clear that extreme points are integral. (totally unimodularity)
- Can we solve the LP in polynomial time? Exponential number of inequalities! Running time polyn. in # variables and constraints, not in input!
- Separation problem: For $\bar{x} \in \mathbb{R}^{|E|}$, check if $\sum_{e \in P} \bar{x}_e \geq 1, \forall P \in \mathcal{P}$?
- This is: Find min-weight s-t-path in G with weights $\bar{x}$. (poly-time)

Intermediate Summary: LPs are easy, in general

LPs can be solved in polynomial time!

- Interior Point method
- Ellipsoid method

Intermediate Summary: LPs are easy, in general

LPs can be solved in polynomial time!

- Interior Point method
- Ellipsoid method

Caution: Not known if Simplex runs in polynomial time
But it works well in practice, implemented in every solver

Intermediate Summary: LPs are easy, in general

LPs can be solved in polynomial time!

- Interior Point method
- Ellipsoid method

Caution: Not known if Simplex runs in polynomial time
But it works well in practice, implemented in every solver

Optimization $=$ Separation

Intermediate Summary: LPs are easy, in general

LPs can be solved in polynomial time!

- Interior Point method
- Ellipsoid method

Caution: Not known if Simplex runs in polynomial time
But it works well in practice, implemented in every solver

Optimization = Separation

In general, ILPs cannot be solved in polynomial time, unless P=NP!

Integer Linear Programs

Integer Linear Programs (ILP)

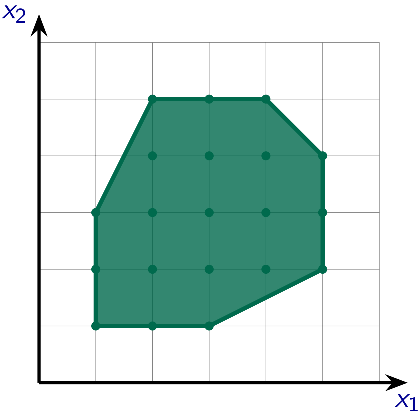
For many problems we find LP formulations only with integrality constraints. → Solving ILPs is NP-hard (in general).

Example:

$$\max \sum_{i=1}^{n} v_i \cdot x_i$$

$$s.t. \sum_{i=1}^{n} w_i \cdot x_i \leq K$$

$$x_i \in \mathbb{Z}_+ \qquad i \in \{1, \ldots, n\}$$



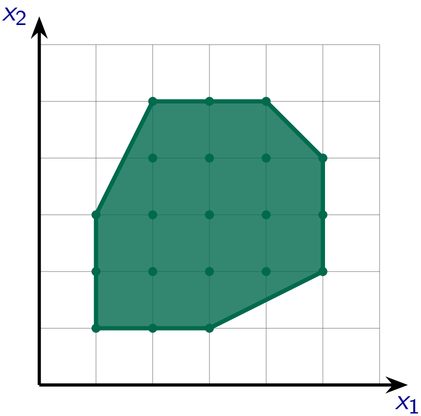
Can we use the machinery of efficiently solving LPs for solving ILPs?

Integer Linear Programs (ILP)

For many problems we find LP formulations only with integrality constraints. → Solving ILPs is NP-hard (in general).

Example: Knapsack problem

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} v_i \cdot x_i \\ s.t. \quad & \sum_{i=1}^{n} w_i \cdot x_i \leq K \\ & x_i \in \mathbb{Z}_+ \qquad i \in \{1, \dots, n\} \end{aligned}$$



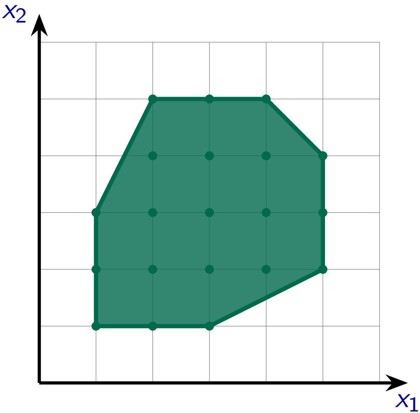
Can we use the machinery of efficiently solving LPs for solving ILPs?

Integer Linear Programs (ILP)

For many problems we find LP formulations only with integrality constraints. → Solving ILPs is NP-hard (in general).

Example: Knapsack problem

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} v_i \cdot x_i \\ s.t. \quad & \sum_{i=1}^{n} w_i \cdot x_i \leq K \\ & x_i \in \mathbb{Z}_+ \qquad i \in \{1, \ldots, n\} \end{aligned}$$



Can we use the machinery of efficiently solving LPs for solving ILPs?

$\longrightarrow$ LP Relaxation!

LP Relaxation

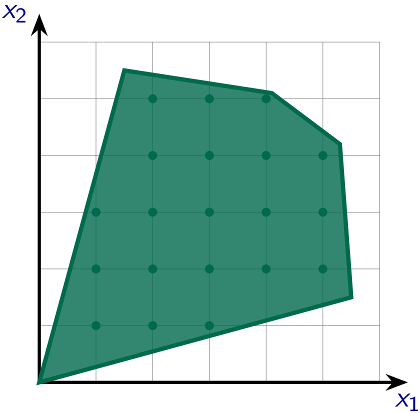
LP Relaxation

Given an ILP

$$\begin{array}{rrcl} \min & c^T x & & \\ \text{s.t.} & Ax & \geq & b \\ & x & \in & \mathbb{N} \end{array}$$

LP relaxation:
Replace $x_v \in \{0, 1\}$ by $x_v \geq 0$.

Observation. $z_{\text{LP}} \leq z_{\text{ILP}}$ (Every ILP solution is feasible for the LP.)



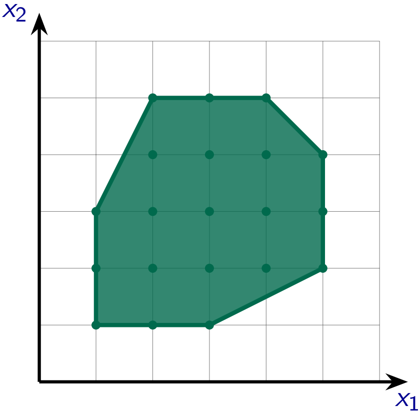
LP Relaxation

Given an ILP

$$\begin{array}{lrcl} \min & c^T x & & \\ \text{s.t.} & Ax & \geq & b \\ & x & \in & \mathbb{N} \end{array}$$

LP relaxation:
Replace $x_v \in \{0, 1\}$ by $x_v \geq 0$.

Observation. $z_{\text{LP}} \leq z_{\text{ILP}}$ (Every ILP solution is feasible for the LP.)



Ideal case: Polyhedron has integral vertices (= integral polyhedron)
$\Rightarrow$ LP relaxation has an integral optimal solution.

Universität Bremen

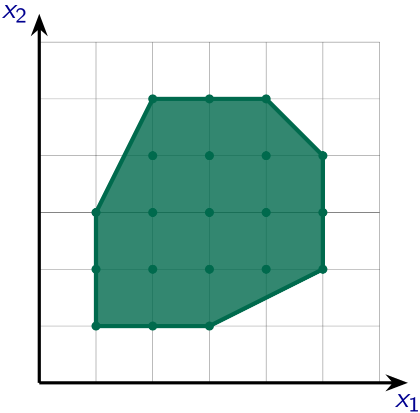
LP Relaxation

Given an ILP

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \geq b \\ & x \in \mathbb{N} \end{aligned}$$

LP relaxation:
Replace $x_v \in \{0,1\}$ by $x_v \geq 0$.

Observation. $z_{\text{LP}} \leq z_{\text{ILP}}$ (Every ILP solution is feasible for the LP.)



Ideal case: Polyhedron has integral vertices (= integral polyhedron)
⇒ LP relaxation has an integral optimal solution.

Question: When is a polyhedron integral?

ILPs with nice structure:

Totally Unimodular Matrices

When is a polyhedron integral?

Total unimodulare Matrizen

Definition

A matrix $A \in \mathbb{Z}^{m \times n}$ is totally unimodular (TU), if every quadratic submatrix of A has determinants $0, -1$ or $+1$.

A quadratic submatrix $B \in \mathbb{Z}^{k \times k}$ of A is obtained by deleting $m - k$ rows and $n - k$ columns in A.

Total unimodulare Matrizen

Definition

A matrix $A \in \mathbb{Z}^{m \times n}$ is totally unimodular (TU), if every quadratic submatrix of A has determinants $0, -1$ or $+1$.

A quadratic submatrix $B \in \mathbb{Z}^{k \times k}$ of A is obtained by deleting $m - k$ rows and $n - k$ columns in A.

Examples:

$$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 1 \\ 1 & 0 & -1 \end{bmatrix} \text{ is TU}$$

Total unimodulare Matrizen

Definition

A matrix $A \in \mathbb{Z}^{m \times n}$ is totally unimodular (TU), if every quadratic submatrix of A has determinants $0, -1$ or $+1$.

A quadratic submatrix $B \in \mathbb{Z}^{k \times k}$ of A is obtained by deleting $m - k$ rows and $n - k$ columns in A.

Examples:

$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 1 \\ 1 & 0 & -1 \end{bmatrix}$ is TU

$\begin{bmatrix} 1 & -1 & 1 \\ 1 & 0 & -1 \\ 1 & 0 & 1 \end{bmatrix}$ is not TU, since $\begin{vmatrix} 1 & 1 \\ 1 & -1 \end{vmatrix} = -2 \notin \{-1, 0, 1\}$

Totally Unimodular Matrices

Theorem (Hoffmann, Kruskal 1956)

A matrix $A \in \mathbb{Z}^{m \times n}$ is totally unimodular if and only if the polyhedron $P = \{x \mid Ax \leq b, x \geq 0\}$ has only integral vertices for any $b \in \mathbb{Z}^m$.

Totally Unimodular Matrices

Theorem (Hoffmann, Kruskal 1956)

A matrix $A \in \mathbb{Z}^{m \times n}$ is totally unimodular if and only if the polyhedron $P = \{x \,|\, Ax \leq b, x \geq 0\}$ has only integral vertices for any $b \in \mathbb{Z}^m$.

Corollary

The problem $\min\{c^T x \,|\, Ax \leq b, x \geq 0\}$ with totally unimodular A and integral b has an integral optimal solution for any c.

Totally Unimodular Matrices

Theorem (Hoffmann, Kruskal 1956)

A matrix $A \in \mathbb{Z}^{m \times n}$ is totally unimodular if and only if the polyhedron $P = \{x \mid Ax \leq b, x \geq 0\}$ has only integral vertices for any $b \in \mathbb{Z}^m$.

Corollary

The problem $\min\{c^T x \mid Ax \leq b, x \geq 0\}$ with totally unimodular A and integral b has an integral optimal solution for any c.

Jackpot!

LP solver can find ILP solution

Characterizations

Lemma (Characterizations)

Let $A \in \{-1, 0, 1\}^{m \times n}$, then the following statements are equivalent:

1. A is totally unimodular.
2. A^T is totally unimodular.
3. There is no quadratic submatrix in A with determinant $+2$ or -2.

[Gomory]

Lemma [Poincaré, 1900]

Let $A \in \{-1, 0, 1\}^{m \times n}$ be a matrix with at most one $+1$ and at most one -1 in each column. Then a A is totally unimodular.

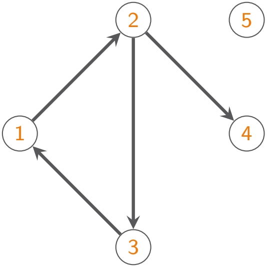
Graph Problems with TU Matrix?

Representation of Graphs

Incidence matrix

The incidence matrix $B \in \mathbb{N}^{m \times n}$ of an (un-)directed graph $G = (V, A)$ is defined as

$$b_{ij} := \begin{cases} 1, & \text{if edge } e_j = (v_i, u) \text{ exists} \\ -1, & \text{if edge } e_j = (u, v_i) \text{ exists (resp. 1 if undirected)} \\ 0, & \text{othw.} \end{cases}$$



$$B = \begin{array}{c} \\ \\ \\ \\ \\ \end{array}\begin{array}{cccc} (1,2) & (2,3) & (3,1) & (2,4) \\ \end{array}$$

$$B = \begin{pmatrix} 1 & 0 & -1 & 0 \\ -1 & 1 & 0 & 1 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Totally Unimodular Matrices for Graphs

Corollary

The incidence matrix of an undirected graph is totally unimodular if and only if the graph is bipartite.

Totally Unimodular Matrices for Graphs

Corollary

The incidence matrix of an undirected graph is totally unimodular if and only if the graph is bipartite.

Korollar

The incidence matrix of a directed graph is totally unimodular.

Totally Unimodular Matrices for Graphs

Corollary

The incidence matrix of an undirected graph is totally unimodular if and only if the graph is bipartite.

Korollar

The incidence matrix of a directed graph is totally unimodular.

Maximum matching in bipartite graphs:
Incidence matrix A is totally unimodular.

$$\begin{aligned} \max \quad & 1^T x \\ \text{s.t.} \quad Ax &\leq 1 \\ x &\geq 0 \end{aligned}$$

Totally Unimodular Matrices for Graphs

Corollary

The incidence matrix of an undirected graph is totally unimodular if and only if the graph is bipartite.

Korollar

The incidence matrix of a directed graph is totally unimodular.

Maximum matching in bipartite graphs:
Incidence matrix A is totally unimodular.

$$\begin{aligned} \max \quad & 1^T x \\ \text{s.t.} \quad & Ax \leq 1 \\ & x \geq 0 \end{aligned}$$

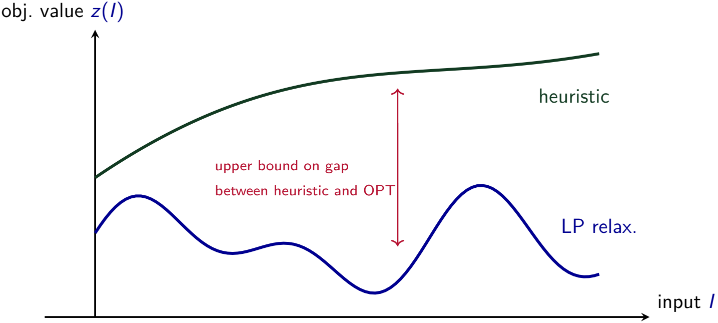
Thus there exists an integral optimal LP solution.

Often LP relaxation is not known to be integral – Still very useful!

The benefit of LP Relaxations

LP relaxation as a lower bound on the optimal solution!

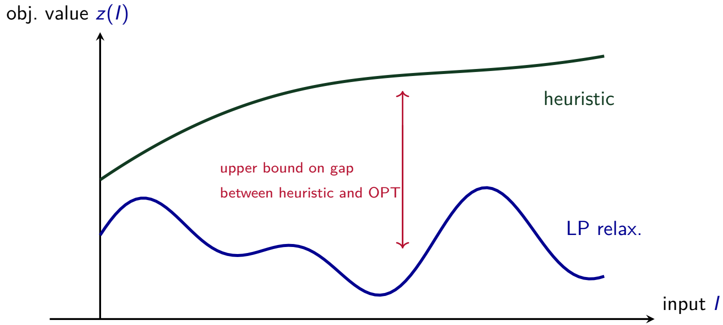
- Useful for evaluating the performance of heuristics



The benefit of LP Relaxations

LP relaxation as a lower bound on the optimal solution!

- Useful for evaluating the performance of heuristics



- Use an infeasible LP solution for constructing a (close) feasible integral solution.

Take Home Messages and Outlook

- Linear programs can be solved in polynomial time.

Take Home Messages and Outlook

- Linear programs can be solved in polynomial time.
- Optimization equivalent to Separation

Take Home Messages and Outlook

- Linear programs can be solved in polynomial time.
- Optimization equivalent to Separation
- Integer linear programs

Take Home Messages and Outlook

- Linear programs can be solved in polynomial time.
- Optimization equivalent to Separation
- Integer linear programs
  - Solving ILPs is NP-hard, in general.

Take Home Messages and Outlook

- Linear programs can be solved in polynomial time.
- Optimization equivalent to Separation
- Integer linear programs
  - Solving ILPs is NP-hard, in general.
  - Solution of LP relaxation is in general fractional and not feasible.

Take Home Messages and Outlook

- Linear programs can be solved in polynomial time.
- Optimization equivalent to Separation
- Integer linear programs
  - Solving ILPs is NP-hard, in general.
  - Solution of LP relaxation is in general fractional and not feasible.
- But the LP relaxation ca be very useful:

Take Home Messages and Outlook

- Linear programs can be solved in polynomial time.
- Optimization equivalent to Separation
- Integer linear programs
  - Solving ILPs is NP-hard, in general.
  - Solution of LP relaxation is in general fractional and not feasible.
- But the LP relaxation ca be very useful:
  - Some ILPs have a "nice" structure: totally unimodular matrices (there is an integral optimal solution to LP relaxation)

Take Home Messages and Outlook

- Linear programs can be solved in polynomial time.
- Optimization equivalent to Separation
- Integer linear programs
  - Solving ILPs is NP-hard, in general.
  - Solution of LP relaxation is in general fractional and not feasible.
- But the LP relaxation ca be very useful:
  - Some ILPs have a “nice” structure: totally unimodular matrices (there is an integral optimal solution to LP relaxation)
  - LP solution gives a bound on the optimal ILP value

Take Home Messages and Outlook

- Linear programs can be solved in polynomial time.
- Optimization equivalent to Separation
- Integer linear programs
  - Solving ILPs is NP-hard, in general.
  - Solution of LP relaxation is in general fractional and not feasible.
- But the LP relaxation ca be very useful:
  - Some ILPs have a “nice” structure: totally unimodular matrices (there is an integral optimal solution to LP relaxation)
  - LP solution gives a bound on the optimal ILP value
  - Round LP solution to “good” integral solution (not this course)

Linear Programming is a powerful machinery!