Game Theory

3. Nash Equilibrium Computation Algorithms

3.1. Finite Zero-Sum Games

Albert-Ludwigs-Universität Freiburg

Bernhard Nebel and Robert Mattmüller

Summer semester 2020

Motivation

- We know: In finite strategic games, mixed-strategy Nash equilibria are guaranteed to exist.
- We don't know: How to systematically find them?
- Challenge: There are infinitely many mixed strategy profiles to consider. How to do this in finite time?

This section:

- Computation of mixed-strategy Nash equilibria for finite zero-sum games.

Next section:

- Computation of mixed-strategy Nash equilibria for general finite two player games.

Mixed-Strategy Nash Equilibria in Finite Zero-Sum Games

We start with finite zero-sum games for two reasons:

- They are easier to solve than general finite two-player games.
- Understanding how to solve finite zero-sum games facilitates understanding how to solve general finite two-player games.

Mixed-Strategy Nash Equilibria in Finite Zero-Sum Games

In the following, we will exploit the zero-sum property of a game G when searching for mixed-strategy Nash equilibria. For that, we need the following result.

Proposition

Let G be a finite zero-sum game. Then the mixed extension of G is also a zero-sum game.

Proof.

Homework. $\square$

Mixed-Strategy Nash Equilibria in Finite Zero-Sum Games

Let G be a finite zero-sum game with mixed extension G'.

Then we know the following:

1. Previous proposition implies: G' is also a zero-sum game.
2. Nash's theorem implies: G' has a Nash equilibrium.
3. Maximinimizer theorem + 1 + 2 implies: Nash equilibria and pairs of maximinimizers in G' are the same.

Mixed-Strategy Nash Equilibria in Finite Zero-Sum Games

Consequence:
When looking for mixed-strategy Nash equilibria in G, it is sufficient to look for pairs of maximinimizers in G'.

Method: Linear Programming

Linear Program Encoding

Approach:

- Let $G = \langle N, (A_i)_{i \in N}, (u_i)_{i \in N} \rangle$ be a finite zero-sum game:
  - $N = \{1, 2\}$.
  - A_1 and A_2 are finite.
  - $U_1(\alpha, \beta) = -U_2(\alpha, \beta)$ for all $\alpha \in \Delta(A_1), \beta \in \Delta(A_2)$.
- Player 1 looks for a maximinimizer mixed strategy α.
- For each possible α of player 1:
  - Determine expected utility against best response of pl. 2. (Only need to consider finitely many pure candidates for best responses because of Support Lemma).
  - Maximize expected utility over all possible α.

Linear Program Encoding

- Result: maximinimizer α for player 1 in G'
  (= Nash equilibrium strategy for player 1)
- Analogously: obtain maximinimizer β for player 2 in G'
  (= Nash equilibrium strategy for player 2)
- With maximinimizer theorem: we can combine α and β into a Nash equilibrium.

Linear Program Encoding

"For each possible α of player 1, determine expected utility against best response of player 2, and maximize."

translates to the following LP:

$$
\begin{aligned}
&\text{Maximize } u && \text{subject to} \\
&\alpha(a) \geq 0 && \text{for all } a \in A_1 \\
&\sum_{a \in A_1} \alpha(a) = 1 && \\
&\underbrace{\sum_{a \in A_1} \alpha(a) \cdot u_1(a,b)}_{=U_1(\alpha,b)} \geq u && \text{for all } b \in A_2
\end{aligned}
$$

Note: Each $\alpha(a)$ is a single LP variable, and so is u.
The values $u_1(a,b)$ are constant coefficients.

Linear Program Encoding

Example (Matching pennies)

	H	T
H	$1, -1$	$-1, \ 1$
T	$-1, \ 1$	$1, -1$

Linear program for player 1:

Maximize u subject to the constraints

$$\alpha(H) \geq 0,\ \alpha(T) \geq 0,\ \alpha(H) + \alpha(T) = 1,$$
$$\alpha(H) \cdot u_1(H,H) + \alpha(T) \cdot u_1(T,H) = \alpha(H) - \alpha(T) \geq u,$$
$$\alpha(H) \cdot u_1(H,T) + \alpha(T) \cdot u_1(T,T) = -\alpha(H) + \alpha(T) \geq u.$$

Solution: $\alpha(H) = \alpha(T) = 1/2$, $u = 0$.

Linear Program Encoding

Theorem

A mixed strategy α is a maximinimizer with payoff u if and only if it is a solution to the LP encoding over α and u.

Proof.

By construction. □

Similarly with β and v for the opposite player.

Linear Program Encoding

Resulting LPs can be solved using off-the-shelf LP solver, e. g.:

- lp_solve
- CLP
- GLPK
- CPLEX
- gurobi

Linear Program Encoding

- Remark: There is an alternative encoding based on the observation that in zero-sum games that have a Nash equilibrium, maximinimization and minimaximization yield the same result.
- Idea: Formulate linear program with inequalities

$$U_1(a,\beta) \leq u \qquad \text{for all } a \in A_1$$

and minimize u. Analogously for β.

Summary

Summary:

- Computing mixed-strategy Nash equilibria in finite zero-sum games can be reduced to solving certain linear programs.
- Some theory is required to justify the reduction: Nash's theorem, maximinimizer theorem, support lemma.
- Resulting LPs are of linear size.
  $\rightsquigarrow$ polynomial-time Nash equilibrium computation

Software:

- Gambit (`http://www.gambit-project.org`) can be used to compute Nash equilibria.
- It also has LP solving built-in as one of the solution methods.

Game Theory

3. Nash Equilibrium Computation Algorithms

3.2. General Finite Two-Player Games

Albert-Ludwigs-Universität Freiburg

UNI FREIBURG

Bernhard Nebel and Robert Mattmüller

Summer semester 2020

Motivation

- We know: In finite strategic games, mixed-strategy Nash equilibria are guaranteed to exist.
- We don't know: How to systematically find them?
- Challenge: There are infinitely many mixed strategy profiles to consider. How to do this in finite time?

Previous section:

- Computation of mixed-strategy Nash equilibria for finite zero-sum games.

This section:

- Computation of mixed-strategy Nash equilibria for general finite two player games.

General Finite Two-Player Games

- For general finite two-player games, the LP approach does not work.
- Instead, use instances of the linear complementarity problem (LCP):
  - Linear (in-)equalities as with LPs.
  - Additional constraints of the form $x_i \cdot y_i = 0$ (or equivalently $x_i = 0 \vee y_i = 0$) for variables $X = \{x_1, \dots, x_k\}$ and $Y = \{y_1, \dots, y_k\}$, and $i \in \{1, \dots, k\}$.
  - no objective function.
- With LCPs, we can compute Nash equilibria for arbitrary finite two-player games.

General Finite Two-Player Games

Let A_1 and A_2 be finite and let (α, β) be a Nash equilibrium with payoff profile (u, v). Then consider this LCP encoding:

$$
\begin{aligned}
u - U_1(a, \beta) &\geq 0 && \text{for all } a \in A_1 && (1)\\
v - U_2(\alpha, b) &\geq 0 && \text{for all } b \in A_2 && (2)\\
\alpha(a) \cdot (u - U_1(a, \beta)) &= 0 && \text{for all } a \in A_1 && (3)\\
\beta(b) \cdot (v - U_2(\alpha, b)) &= 0 && \text{for all } b \in A_2 && (4)\\
\alpha(a) &\geq 0 && \text{for all } a \in A_1 && (5)\\
\sum_{a \in A_1} \alpha(a) &= 1 && && (6)\\
\beta(b) &\geq 0 && \text{for all } b \in A_2 && (7)\\
\sum_{b \in A_2} \beta(b) &= 1 && && (8)
\end{aligned}
$$

UNI FREIBURG

Remarks about the encoding:

- In (3) and (4): for instance,

$$\alpha(a) \cdot (u - U_1(a, \beta)) = 0$$

  if and only if

$$\alpha(a) = 0 \qquad \text{or} \qquad u - U_1(a, \beta) = 0.$$

  This holds in every Nash equilibrium, because:
  - if $a \notin supp(\alpha)$, then $\alpha(a) = 0$, and
  - if $a \in supp(\alpha)$, then $a \in B_1(\beta)$, thus $U_1(a, \beta) = u$.
- With additional variables, the above LCP formulation can be transformed into LCP normal form.

UNI FREIBURG

Theorem

A mixed strategy profile (α, β) with payoff profile (u, v) is a Nash equilibrium if and only if it is a solution to the LCP encoding over (α, β) and (u, v).

Proof.

- Nash equilibria are solutions to the LCP: Obvious because of the support lemma.
- Solutions to the LCP are Nash equilibria: Let (α, β, u, v) be a solution to the LCP. Because of (5)–(8), α and β are mixed strategies.

General Finite Two-Player Games

Proof (ctd.)

- Solutions to the LCP are Nash equilibria (ctd.): Because of (1), u is at least the maximal payoff over all possible pure responses, and because of (3), u is exactly the maximal payoff.

  If $\alpha(a) > 0$, then, because of (3), the payoff for player 1 against β is u.

  The linearity of the expected utility implies that α is a best response to β.

  Analogously, we can show that β is a best response to α and hence (α, β) is a Nash equilibrium with payoff profile (u, v). □

Solution Algorithm for LCPs

Naïve algorithm:

Enumerate all $(2^n - 1) \cdot (2^m - 1)$ possible pairs of support sets.

For each such pair $(supp(\alpha), supp(\beta))$:

- Convert the LCP into an LP:
  - Linear (in-)equalities are preserved.
  - Constraints of the form $\alpha(a) \cdot (u - U_1(a, \beta)) = 0$ are replaced by a new linear equality:
    - $u - U_1(a, \beta) = 0$, if $a \in supp(\alpha)$, and
    - $\alpha(a) = 0$, otherwise,

    Analogously for $\beta(b) \cdot (v - U_2(\alpha, b)) = 0$.
  - Objective function: maximize constant zero function.
- Apply solution algorithm for LPs to the transformed program.

Solution Algorithm for LCPs

- Runtime of the naïve algorithm: $O(p(n+m) \cdot 2^{n+m})$, where p is some polynomial.
- Better in practice: Lemke-Howson algorithm.
- Complexity:
  - unknown whether $\textsc{LcpSolve} \in \mathbf{P}$.
  - $\textsc{LcpSolve} \in \mathbf{NP}$ is clear
    (naïve algorithm can be seen as a nondeterministic polynomial-time algorithm).

Summary

- Previous section: Computation of mixed-strategy Nash equilibria for finite zero-sum games using linear programs.
  $\rightsquigarrow$ polynomial-time computation
- This section: Computation of mixed-strategy Nash equilibria for general finite two player games using linear complementarity problem.
  $\rightsquigarrow$ computation in **NP**

Game Theory

3. Nash Equilibrium Computation Algorithms

Appendix. Linear Programming

Albert-Ludwigs-Universität Freiburg

UNI FREIBURG

Bernhard Nebel and Robert Mattmüller

Summer semester 2020

UNI FREIBURG

Appendix:

In this appendix, we briefly discuss linear programming.
(We need it to find Nash equilibria.)

Goal of linear programming:

Solving a system of linear inequalities over n real-valued variables while optimizing some linear objective function.

Linear Programming

UNI FREIBURG

Example

Production of two sorts of items with time requirements and profit per item. Objective: Maximize profit.

	Cutting	Assembly	Postproc.	Profit per item
(x) sort 1	25	60	68	30
(y) sort 2	75	60	34	40
per day	≤ 450	≤ 480	≤ 476	maximize!

Goal: Find numbers of pieces x of sort 1 and y of sort 2 to be produced per day such that the resource constraints are met and the objective function is maximized.

Example (ctd., formalization)

$$x \geq 0,\ y \geq 0 \tag{1}$$

$$25x + 75y \leq 450 \quad (\text{or } y \leq 6 - {}^{1}\!/_{3}\, x) \tag{2}$$

$$60x + 60y \leq 480 \quad (\text{or } y \leq 8 - x) \tag{3}$$

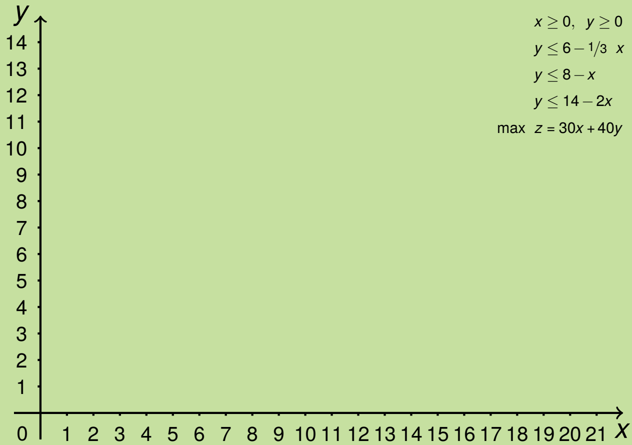
$$68x + 34y \leq 476 \quad (\text{or } y \leq 14 - 2x) \tag{4}$$

$$\text{maximize } z = 30x + 40y \tag{5}$$

- Inequalities (1)–(4): Admissible solutions
  (They form a convex set in $\mathbb{R}^2$.)
- Line (5): Objective function

UNI FREIBURG

Example (ctd., visualization)



$x \geq 0, \quad y \geq 0$

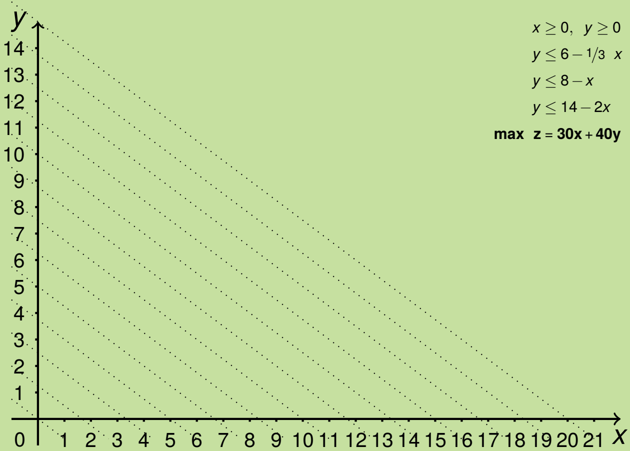
$y \leq 6 - 1/3 \ x$

$y \leq 8 - x$

$y \leq 14 - 2x$

max $\ z = 30x + 40y$

Example (ctd., visualization)



$x \geq 0, \quad y \geq 0$

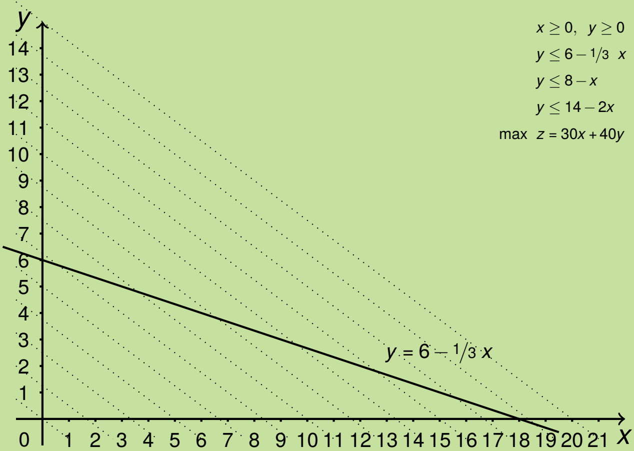
$y \leq 6 - 1/3\ x$

$y \leq 8 - x$

$y \leq 14 - 2x$

max $\mathbf{z = 30x + 40y}$

Example (ctd., visualization)



$x \geq 0, \quad y \geq 0$

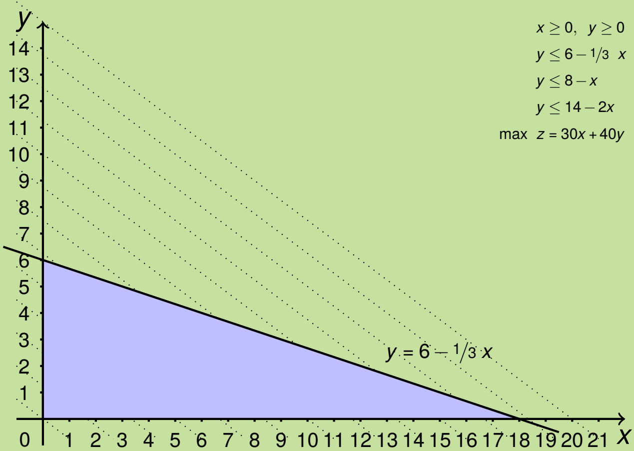
$y \leq 6 - 1/3\ x$

$y \leq 8 - x$

$y \leq 14 - 2x$

max $z = 30x + 40y$

Example (ctd., visualization)



$x \geq 0, \quad y \geq 0$

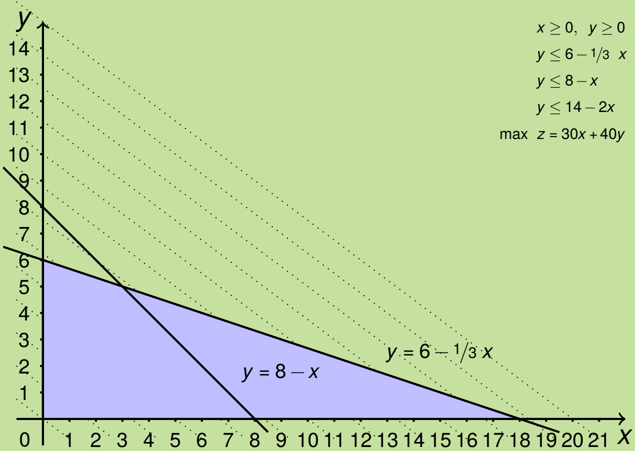
$y \leq 6 - 1/3\ x$

$y \leq 8 - x$

$y \leq 14 - 2x$

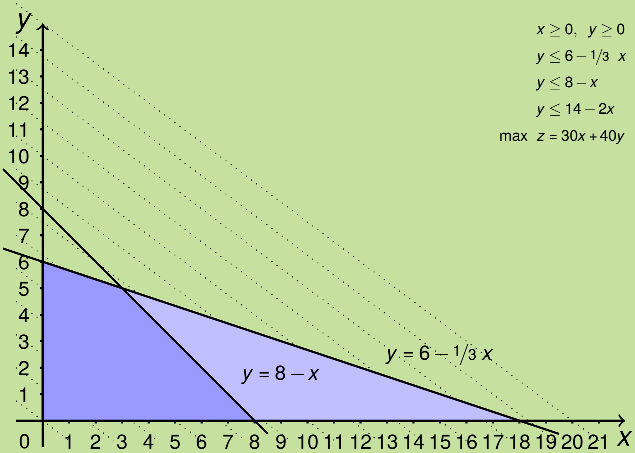
max $z = 30x + 40y$

Example (ctd., visualization)



$$
\begin{aligned}
& x \geq 0, \quad y \geq 0 \\
& y \leq 6 - 1/3 \ x \\
& y \leq 8 - x \\
& y \leq 14 - 2x \\
\text{max} \quad & z = 30x + 40y
\end{aligned}
$$

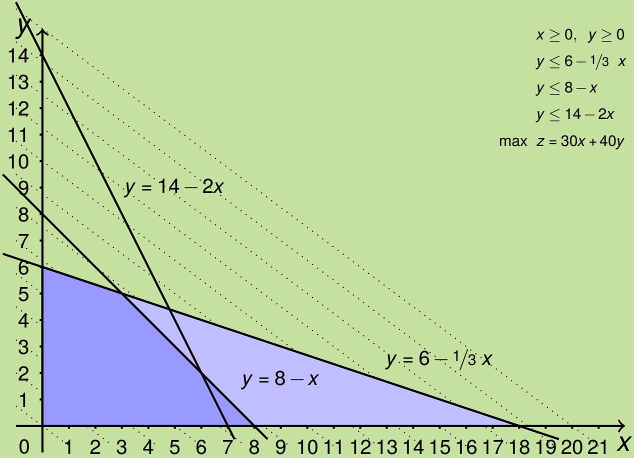
Example (ctd., visualization)



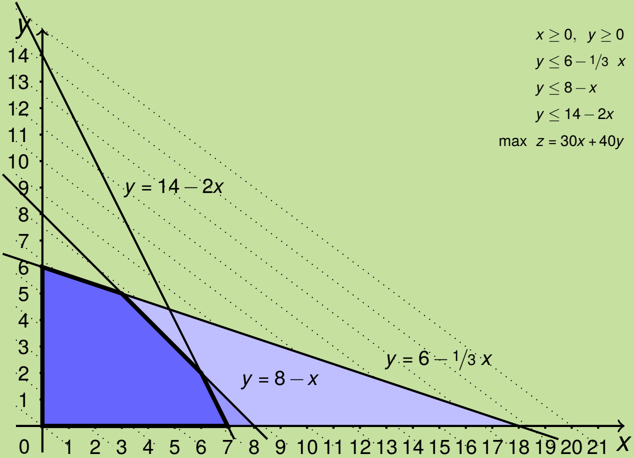
$$x \geq 0, \quad y \geq 0$$
$$y \leq 6 - 1/3\ x$$
$$y \leq 8 - x$$
$$y \leq 14 - 2x$$
$$\max \quad z = 30x + 40y$$

Linear Programming

Example (ctd., visualization)



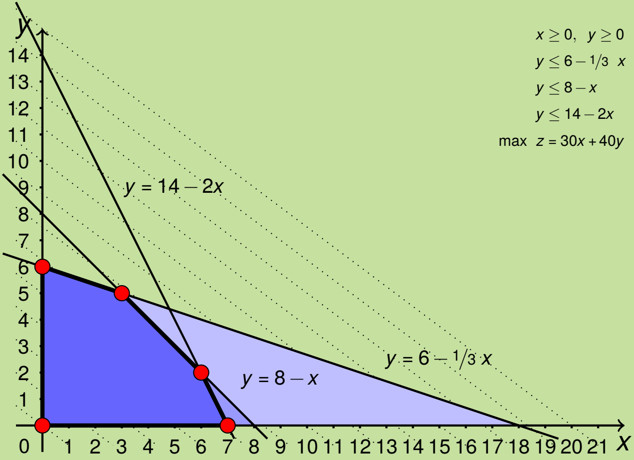
Example (ctd., visualization)



$x \geq 0, \quad y \geq 0$
$y \leq 6 - 1/3\ x$
$y \leq 8 - x$
$y \leq 14 - 2x$
max $z = 30x + 40y$

UNI FREIBURG

Example (ctd., visualization)



$x \geq 0, \quad y \geq 0$

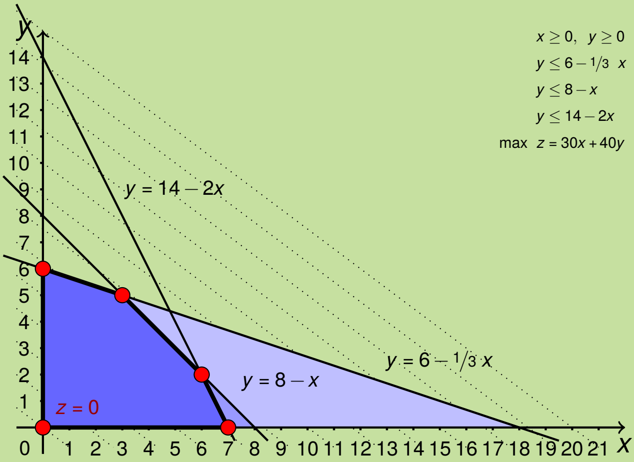
$y \leq 6 - 1/3 \; x$

$y \leq 8 - x$

$y \leq 14 - 2x$

max $z = 30x + 40y$

Example (ctd., visualization)



$x \geq 0, \quad y \geq 0$

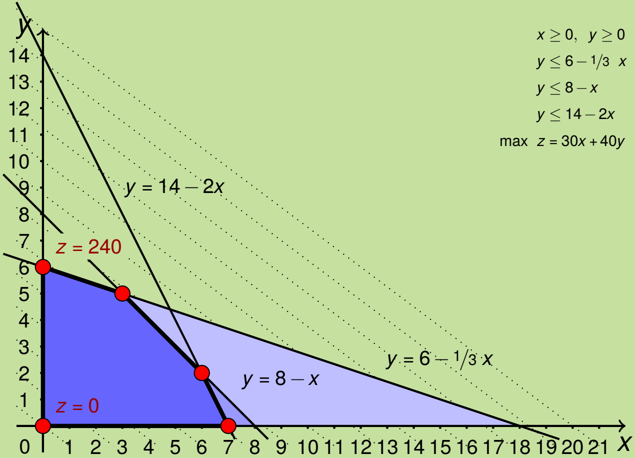
$y \leq 6 - 1/3\ x$

$y \leq 8 - x$

$y \leq 14 - 2x$

max $z = 30x + 40y$

Example (ctd., visualization)



$x \geq 0, \quad y \geq 0$

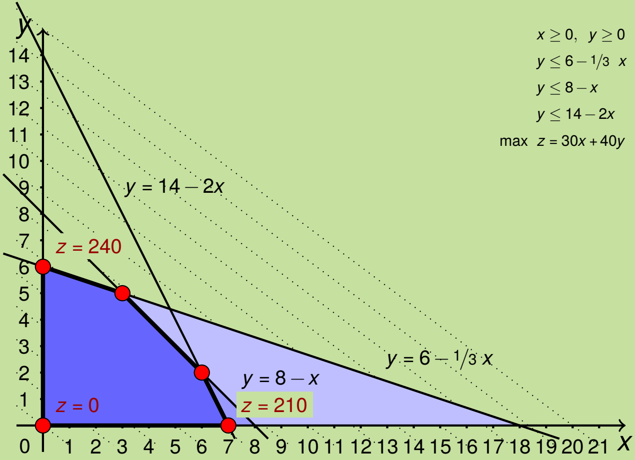
$y \leq 6 - 1/3\ x$

$y \leq 8 - x$

$y \leq 14 - 2x$

max $z = 30x + 40y$

Example (ctd., visualization)



$x \geq 0, \quad y \geq 0$

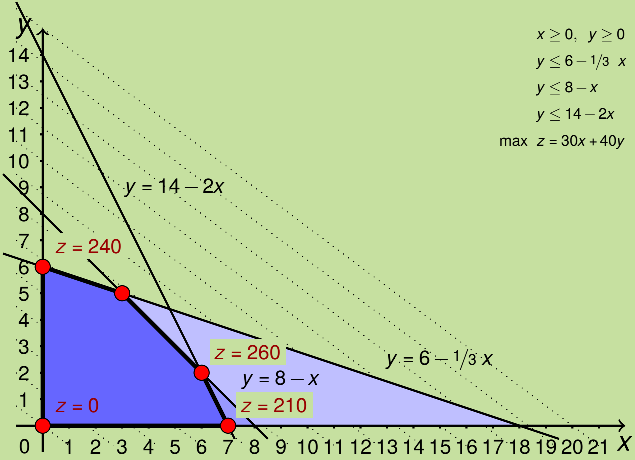
$y \leq 6 - 1/3\ x$

$y \leq 8 - x$

$y \leq 14 - 2x$

$\max \quad z = 30x + 40y$

Example (ctd., visualization)



$x \geq 0, \quad y \geq 0$

$y \leq 6 - 1/3\ x$

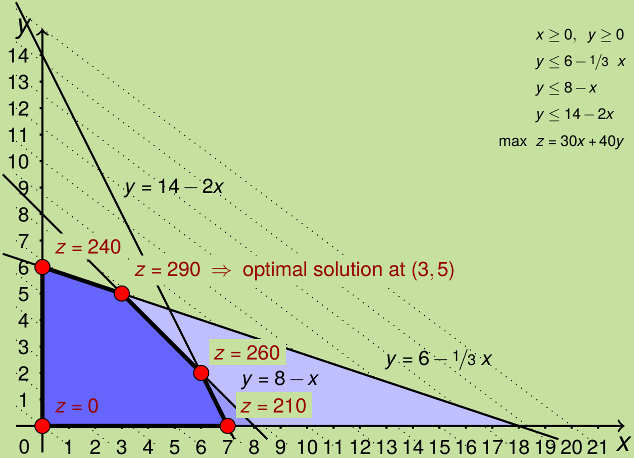
$y \leq 8 - x$

$y \leq 14 - 2x$

max $z = 30x + 40y$

UNI FREIBURG

Example (ctd., visualization)



$x \geq 0, \quad y \geq 0$

$y \leq 6 - 1/3\ x$

$y \leq 8 - x$

$y \leq 14 - 2x$

max $z = 30x + 40y$

Definition (Linear program)

A linear program (LP) in standard from consists of

- n real-valued variables x_i; n coefficients b_i;
- m constants c_j; $n \cdot m$ coefficients a_{ij};
- m constraints of the form

$$c_j \leq \sum_{i=1}^{n} a_{ij} x_i,$$

- and an objective function to be minimized ($x_i \geq 0$)

$$\sum_{i=1}^{n} b_i x_i.$$

Linear Programming

Solution of an LP:
assignment of values to the x_i satisfying the constraints and minimizing the objective function.

Remarks:

- Maximization instead of minimization: easy, just change the signs of all the b_i's, $i = 1, \ldots, n$.
- Equalities instead of inequalities: $x + y \leq c$ if and only if there is a $z \geq 0$ such that $x + y + z = c$ (z is called a slack variable).

Linear Programming

Solution algorithms:

- Usually, one uses the simplex algorithm (which is worst-case exponential!).
- There are also polynomial-time algorithms such as interior-point or ellipsoid algorithms.

Tools and libraries:

- lp_solve
- CLP
- GLPK
- CPLEX
- gurobi