ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Special Topics on Algorithms

Introduction to Linear and Integer Programming

Vangelis Markakis   Ioannis Milis
George Zois

Introduction to Linear Programming

Linear Programming

- Nothing to do with programming!
- A particular way of formulating certain optimization problems with linear constraints and a linear objective function
- One of the most useful tools in Algorithms and Operations Research
- Extremely useful also in the design of approximation algorithms

Linear Programming

Applications of Linear Programming: Too many to enumerate!

- Operations Research
- Theory of Algorithms and Combinatorial Optimization
- Game theory and Microeconomics
- Medicine
- And many more...

Linear Programming Examples

Example 1:

- A farmer possesses a land of 10 km^2
- He wants to plant the land with wheat, or barley or a combination of them
- The farmer has a limited amount of fertilizer, say 16 kgs
- And a limited amount of pesticide, say 18 kgs
- Each square km of wheat requires 1kg of fertilizer and 2 kgs of pesticide
- Each square km of barley requires 2kg of fertilizer and 1.2 kgs of pesticide
- Revenue to the farmer: 3 (thousand $) from each square km of wheat and 4 (thousand $) from each square km of barley
- Find out what the farmer should do (i.e., how many square km of barley and how many of wheat he should plant) to maximize his revenue.

Linear Programming Examples

Formulation as a linear program:

First step: We need to define the decision variables of our problem

- x_1 = number of square km for wheat
- x_2 = number of square km for barley
- Often multiple ways for doing this step
- Objective function: maximize $3x_1 + 4x_2$
- Observe that: objective function is linear

Linear Programming Examples

Formulation as a linear program:

Second step: formulation of constraints on the variables x_1, x_2

- Area constraint: $x_1 + x_2 \leq 10$
- Constraint for fertilizer: $x_1 + 2x_2 \leq 16$
- Constraint for pesticide: $2x_1 + 1.2x_2 \leq 18$
- Nonnegativity constraints: $x_1 \geq 0$, $x_2 \geq 0$ (cannot plant an area with negative surface)
- Observe: all constraints are also linear

Linear Programming Examples

Usual writing style:

$$\max\ 3x_1 + 4x_2$$

Objective function

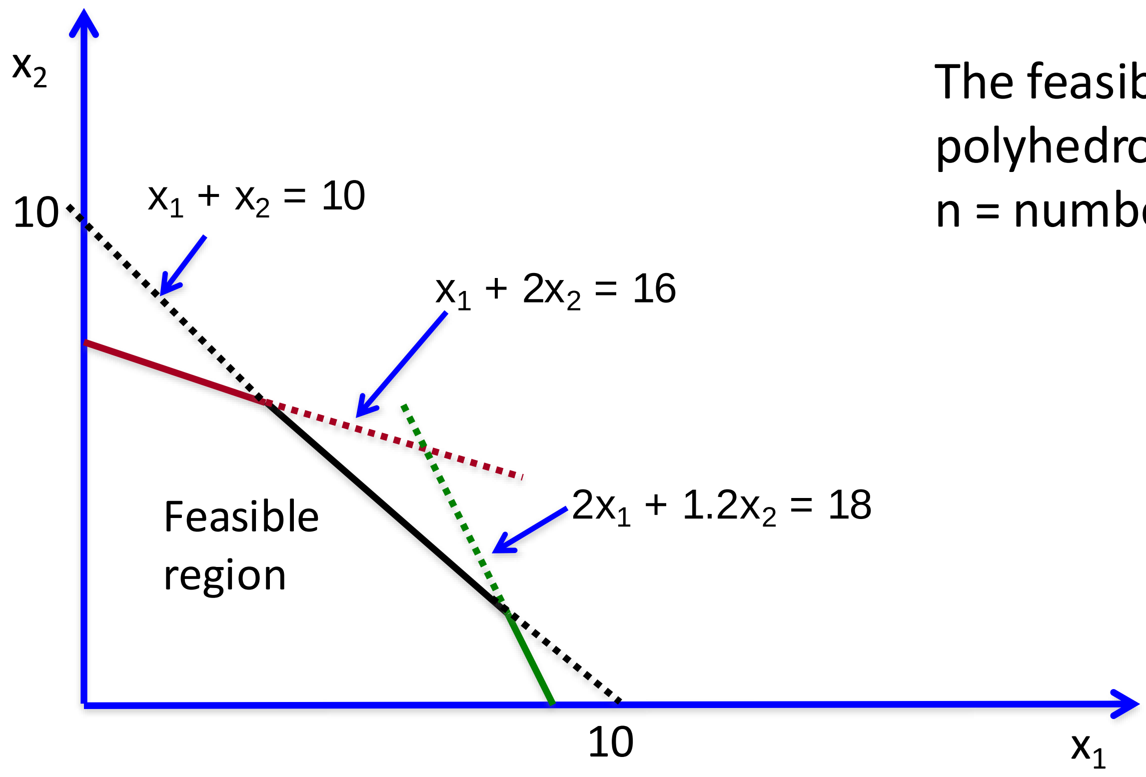
$$
\begin{aligned}
s.t.\quad & x_1 + x_2 \text{ £} 10 \\
& x_1 + 2x_2 \text{ £} 16 \\
& 2x_1 + 1.2x_2 \text{ £} 18 \\
& x_1, x_2 \ {}^{3}\ 0
\end{aligned}
$$

constraints

- It can be either a minimization or a maximization problem
- Feasible space (or region): the set of all pairs (x_1, x_2) that satisfy the constraints
- In the example: the feasible region is a subset of R^2

Linear Programming Examples

Geometrically:



The feasible region is a polyhedron in R^2, where n = number of variables

Linear Programming Examples

Example 2:

- A manufacturing company selling glass and aluminum products is trying to invest in launching 2 new products
- The company has 3 plants
  - Plant 1: for processing aluminum
  - Plant 2: for processing glass
  - Plant 3: for assembling and finalizing products
- Product 1 requires processing in Plant 1 and Plant 3
- Product 2 requires processing in Plant 2 and Plant 3
- Since the company processes other products as well, there are constraints on the amount of time available in each plant.

Linear Programming Examples

Plant	Time needed per batch (hours)		Total available time per week (hours)
	Product		
	1	2	
1	1	0	4
2	0	2	12
3	3	2	18
Profit per batch	3000	5000	

- Goal: Decide how many batches of Product 1 and Product 2 to produce so as
  - Not to exceed the available time capacity in each plant
  - Maximize total revenue from the batches produced

Linear Programming Examples

Formulation as a linear program:

First step: determine the decision variables of our problem

- x_1 = number of batches of product 1, produced per week
- x_2 = number of batches of product 2, produced per week

Second step: formulation of constraints on the variables x_1, x_2

- Time constraints for Plant 1: $x_1 \leq 4$
- Time constraints for Plant 2: $2x_2 \leq 12$
- Time constraints for Plant 3: $3x_1 + 2x_2 \leq 18$
- Nonnegativity constraints: $x_1 \geq 0$, $x_2 \geq 0$ (number of batches produced cannot be negative)

Objective function: maximize $3x_1 + 5x_2$

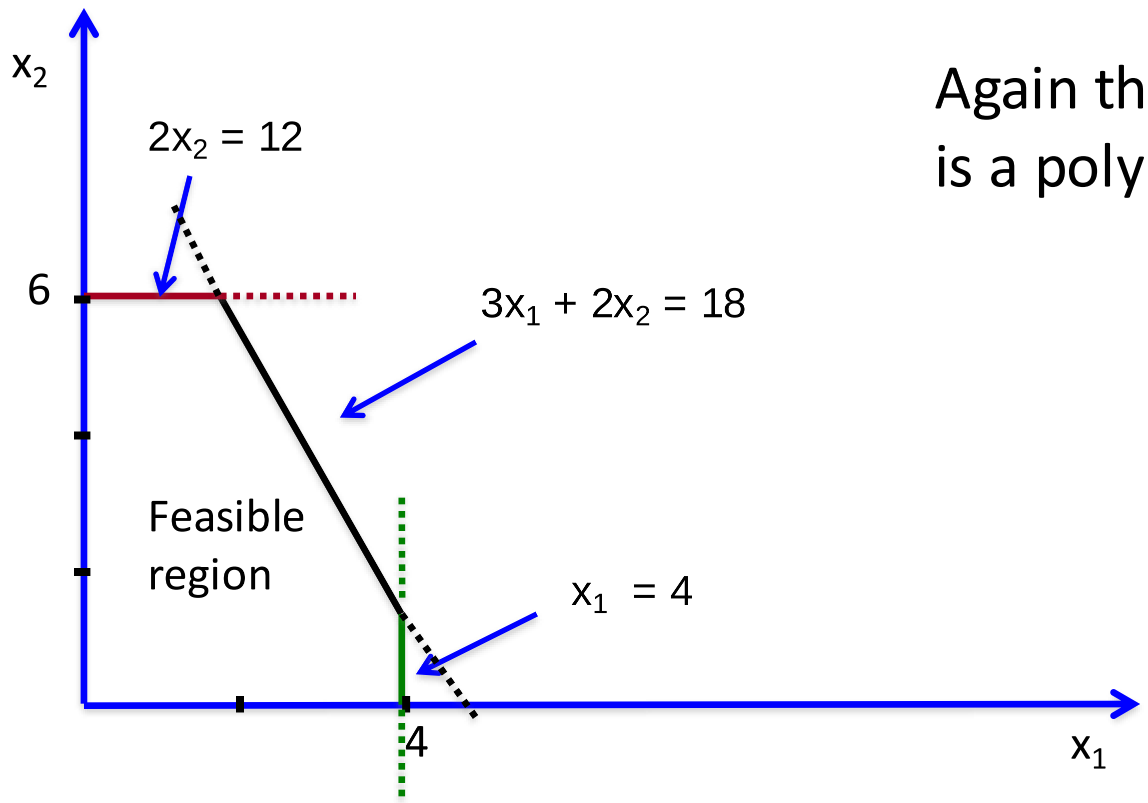
Linear Programming Examples

Hence:

$$
\begin{array}{ll}
\max \; 3x_1 + 5x_2 & \longleftarrow \text{Objective function} \\
s.t. \quad \left.\begin{array}{l}
x_1 \le 4 \\
2x_2 \le 12 \\
3x_1 + 2x_2 \le 18 \\
x_1, x_2 \ge 0
\end{array}\right\} & \text{constraints}
\end{array}
$$

Linear Programming Examples

Geometrically:



Again the feasible region is a polyhedron in R^2

Linear Programming Examples

A more succinct notation (canonical form)

We can represent Example 2 as:

$$
\begin{aligned}
\text{max.} \quad & c^T x \\
\text{s.t.} \quad & \\
& Ax \le b \\
& x \ge 0
\end{aligned}
$$

$$where\ x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, c = \begin{pmatrix} 3 \\ 5 \end{pmatrix}, b = \begin{pmatrix} 4 \\ 12 \\ 18 \end{pmatrix}, A = \begin{pmatrix} 1 & 0 \\ 0 & 2 \\ 3 & 2 \end{pmatrix}$$

Notation: $x \ge 0$ for a vector x means that the inequality should hold component-wise (for every coordinate)

General Form of Linear Programs

Given:

- $c_1, c_2, \ldots, c_n$
- $b_1, b_2, \ldots, b_m$
- The constraint matrix $A = (a_{ij})$ with $1 \le i \le m$, $1 \le j \le n$,

We want to:

$$
\begin{aligned}
&\text{maximize} \quad Z = c_1x_1 + c_2x_2 + \ldots + c_nx_n \\
&\text{subject to:} \\
&\qquad a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \le b_1 \\
&\qquad a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \le b_2 \\
&\qquad \vdots \\
&\qquad a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \le b_m \\
&\qquad x_1 \ge 0, x_2 \ge 0, \ldots, x_n \ge 0
\end{aligned}
$$

General Form of Linear Programs

More concisely:

$$
\begin{aligned}
\text{max:}\ & Z = c^T \cdot x \\
\text{s. t.:}\ & \\
& A \cdot x \leq b \\
& x \geq 0
\end{aligned}
$$

Where:

- c and x are n-dimensional vectors
- b is an m-dimensional vector
- n decision variables
- m inequality constraints
- n nonnegativity constraints

Linear Programming

Other forms of LPs we could encounter:

1. Minimization problem instead of maximization
2. >= inequalities in the constraints
3. Equality constraints
4. Absence of nonnegativity constraints

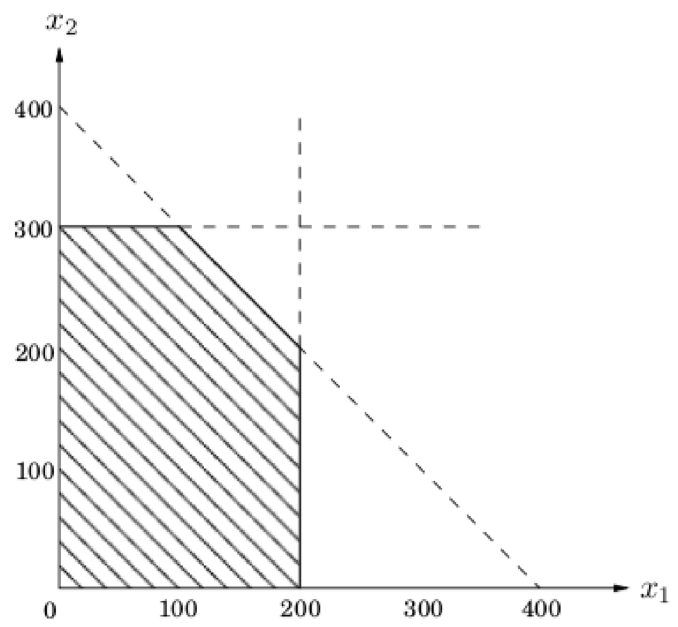
Claim: All these are equivalent forms, and can be reduced to one another

- If we have a minimization problem: revert the signs in the coefficients of the objective function and maximize the new function.
- >= constraints: again revert signs to bring them to <= constraints
- Equality constraints: replace them by 2 constraints (one with >=, and one with <=)

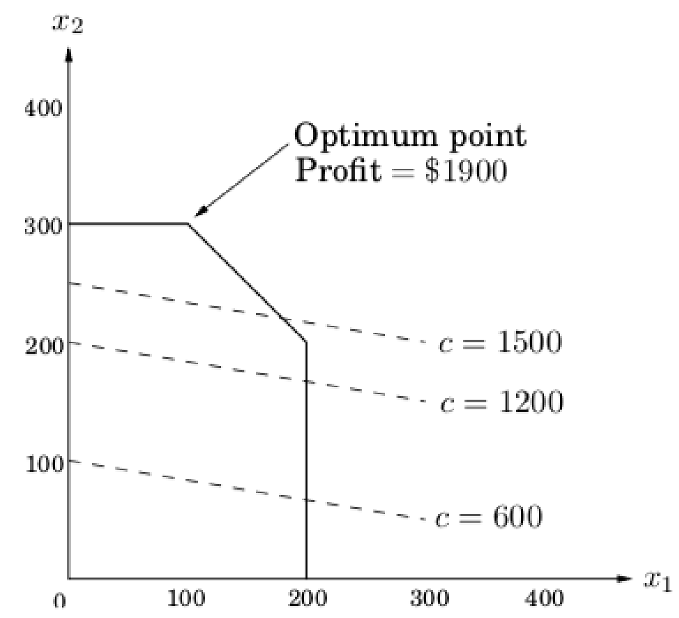
Geometry of Linear Programming

Objective function $\max x_1 + 6x_2$

Constraints
$$x_1 \leq 200$$
$$x_2 \leq 300$$
$$x_1 + x_2 \leq 400$$
$$x_1, x_2 \geq 0$$



Feasible region

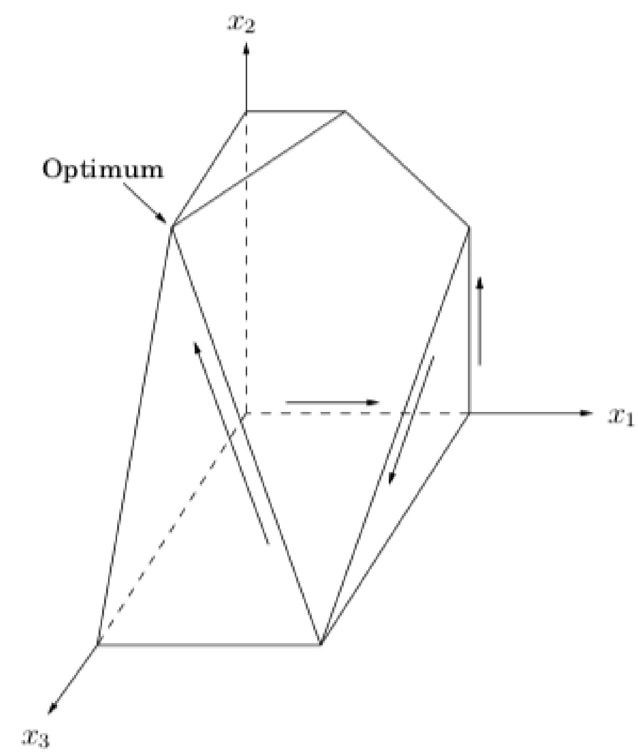


Profits

Geometry of Linear Programming

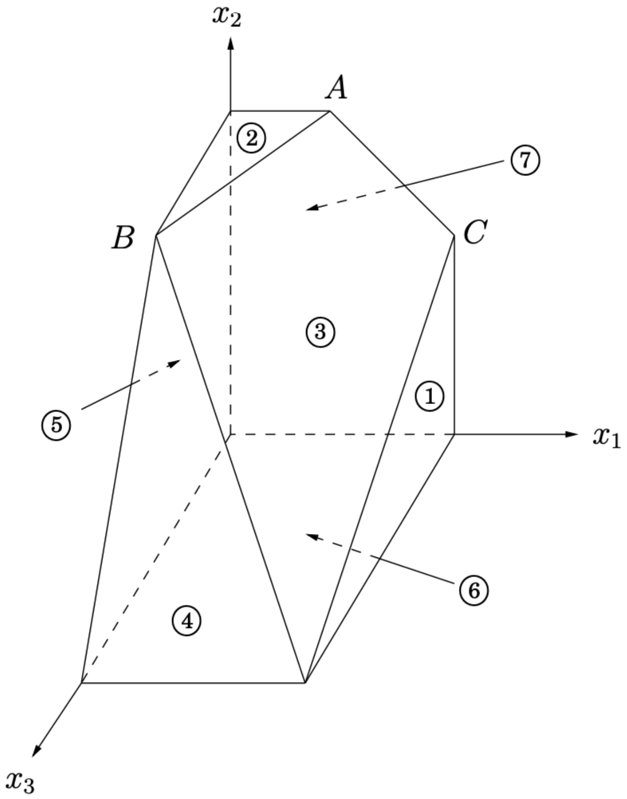
In 3 dimensions:

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



Each constraint corresponds to a face of the polyhedron

Geometry of Linear Programming



$$
\begin{aligned}
\max \quad & x_1 + 6x_2 + 13x_3 & \\
x_1 &\leq 200 & ① \\
x_2 &\leq 300 & ② \\
x_1 + x_2 + x_3 &\leq 400 & ③ \\
x_2 + 3x_3 &\leq 600 & ④ \\
x_1 &\geq 0 & ⑤ \\
x_2 &\geq 0 & ⑥ \\
x_3 &\geq 0 & ⑦
\end{aligned}
$$

Geometry of Linear Programming

- **Key property:** The optimum is achieved at a vertex of the feasible region
- The only exceptions are cases in which there is no optimum

1. The LP is **infeasible**

   too tight constraints; impossible to satisfy all of them

   e.g. $x_1 \leq 1$, $x_1 \geq 2$

2. The LP is **unbounded;**

   too loose constraints; the feasible region is unbounded

   e.g. arbitrarily high objective values

   max $x_1 + x_2$

   $x_1, x_2 \geq 0$

The Graphical Method

- Applicable for linear programs with 2 or 3 decision variables
- It helps us understand how to think about solving problems in higher dimensions

Solving Example 2:

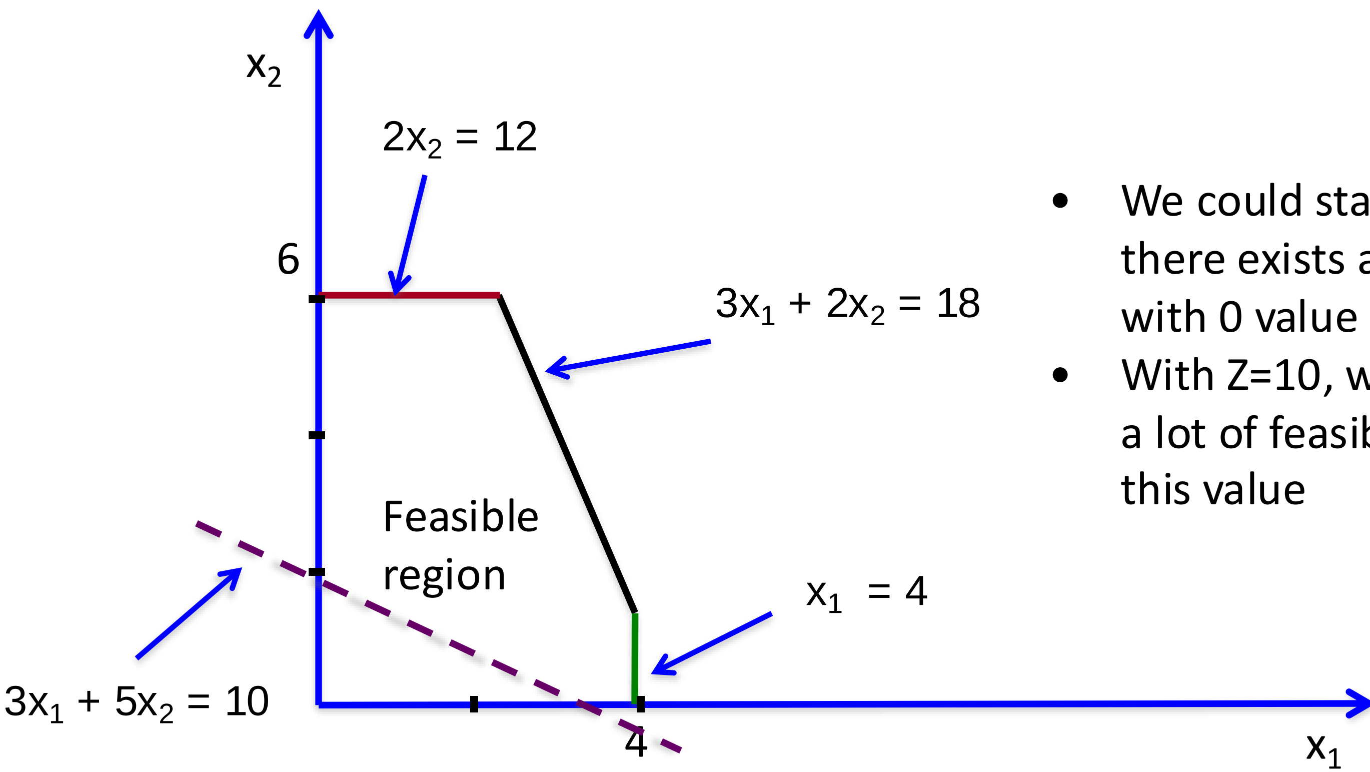
- Step 1: Draw the feasible region
- Step 2: "Guess" a value Z for the objective function and draw the line $3x_1 + 5x_2 = Z$
- If this line intersects the feasible region, it means we have at least one feasible solution with value Z
- Trial and error: Keep doing this, increasing Z till the line gets out of the feasible region

The Graphical Method

Solving Example 2:

•Step 1: Draw the feasible region

•Step 2: Trial and error: "Guess" a value Z for the objective function and draw the line $3x_1 + 5x_2 = Z$

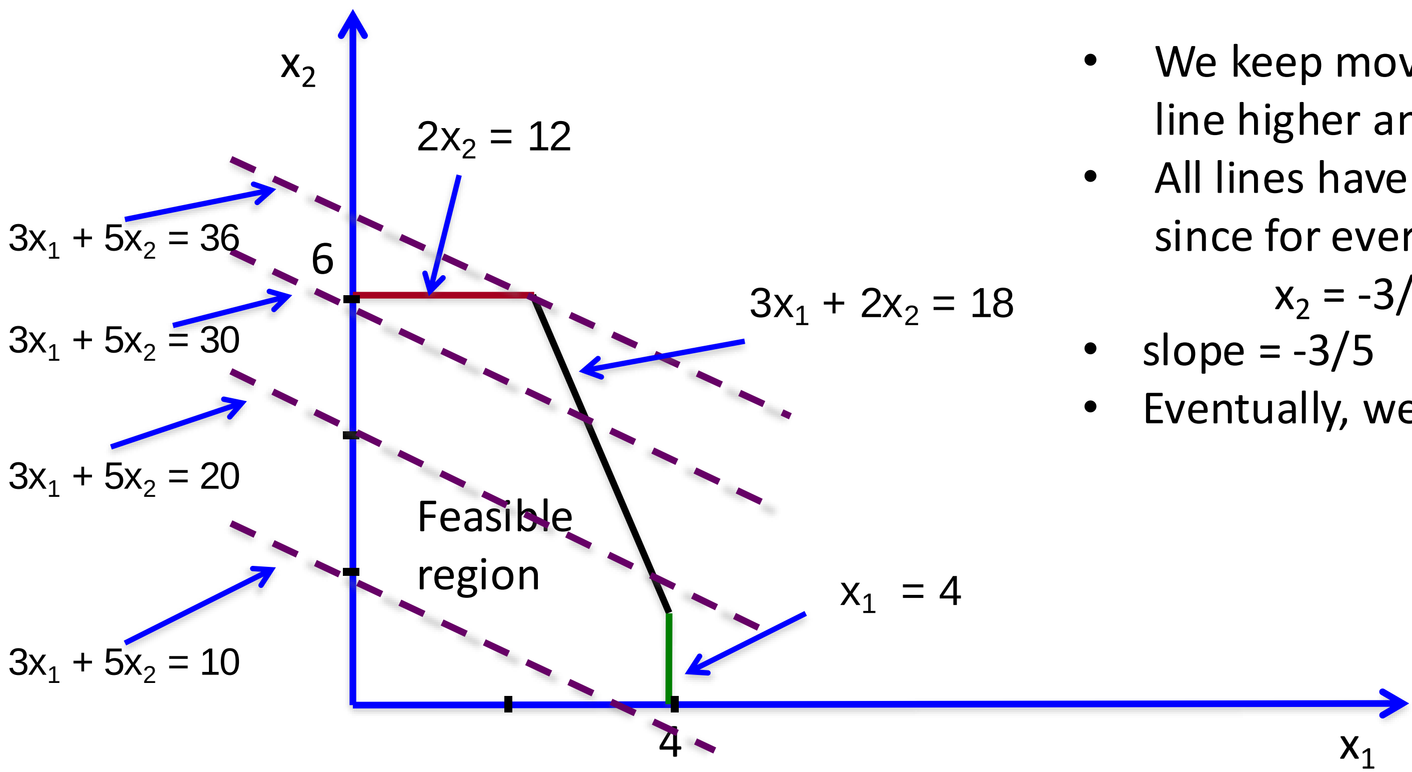


- We could start with Z=0 since there exists a feasible solution with 0 value
- With Z=10, we see there are still a lot of feasible solutions with this value

The Graphical Method

Solving Example 2:

- We can now keep examining higher values for Z, until we get out of the feasible region



- We keep moving the dashed line higher and higher
- All lines have the same slope, since for every Z:
  $x_2 = -3/5\ x_1 + 1/5\ Z$
- slope = -3/5
- Eventually, we stop at Z = 36

The Graphical Method

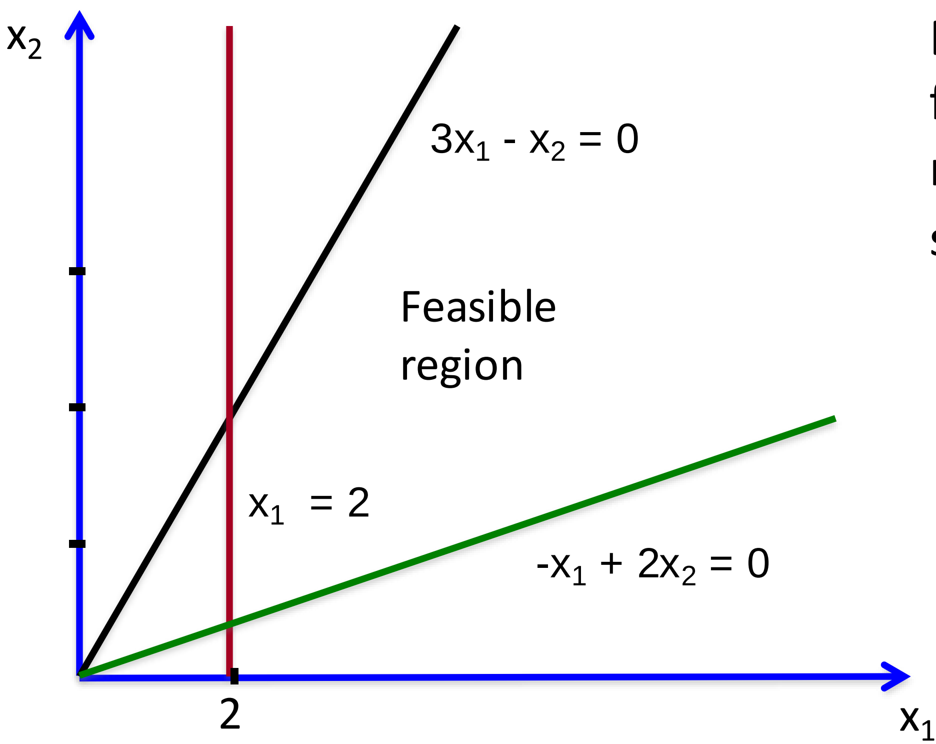
Observations:

- In 2 dimensions, the feasible region is a polygon
- We stop only when the dashed line intersects the feasible region in a corner point of the polygon
  - Or in degenerate cases, when the line coincides with one of the sides of the polygon
- How can we compute the values of x_1, x_2 when we stop?
  - A corner point is the intersection of 2 sides, hence they satisfy 2 constraints with equality
- In Example 2, we stop at Z=36
- The solution of
  - $2x_2 = 12$
  - $3x_1 + 2x_2 = 18$
- Hence, $x_1 = 2$, $x_2 = 6$

The Graphical Method

Can the graphical method keep going without ever terminating?

- YES, when the polyhedron is unbounded
- But if this happens, the optimal solution is $+\infty$



Example of an unbounded feasible region:

max $Z = 4x_1 + 2x_2$

s.t.

$$x_1 \geq 2$$
$$3x_1 - x_2 \geq 0$$
$$-x_1 + 2x_2 \geq 0$$
$$x_1, x_2 \geq 0$$

The Graphical Method

- Insights gained from the graphical method:
  – If an optimal solution exists, it is attained at a corner point of the polygon
- What about higher dimensions?
- Many real world problems have hundreds of variables
  – In higher dimensions, the feasible region is still a polyhedron
  – Again, it suffices to look at the corner points of the polyhedron
  – Till 3 dimensions, we can do this geometrically
  – When $n \geq 4$, we should do it algebraically

- Idea for higher dimensional problems: Try to examine corner points of the polyhedron till we reach the optimal one

The Graphical Method

- Q: What is a corner point in higher dimensions?
  - Definition: A feasible solution of a linear program with n variables is a corner point (or vertex) if it satisfies n linearly independent inequalities with exact equality
- Q: Could we enumerate all corner point solutions and pick the best one?
  - Not an efficient algorithm, polyhedra can have exponentially many corner points.
- BUT: We can try to think of a more clever way to search for the best corner point
  - Essentially what simplex does

The Simplex Method

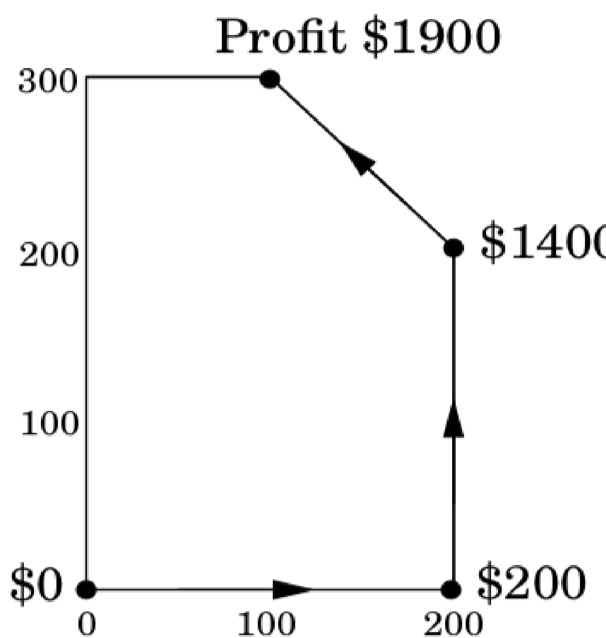
- Designed by Dantzig (1947)
  - One of the most important algorithms of the 20^{th} century
  - An algorithm that behaves extremely well in practice despite its exponential complexity in worst case
  - The design of the algorithm and the quest for better algorithms also contributed to building a rich theory around linear programming

The Simplex Method

- Starts at a vertex, say (0, 0)
- Repeatedly looks for an adjacent vertex of better objective value
- Halts upon reaching a vertex that has no better neighboring vertices and declares it as optimal

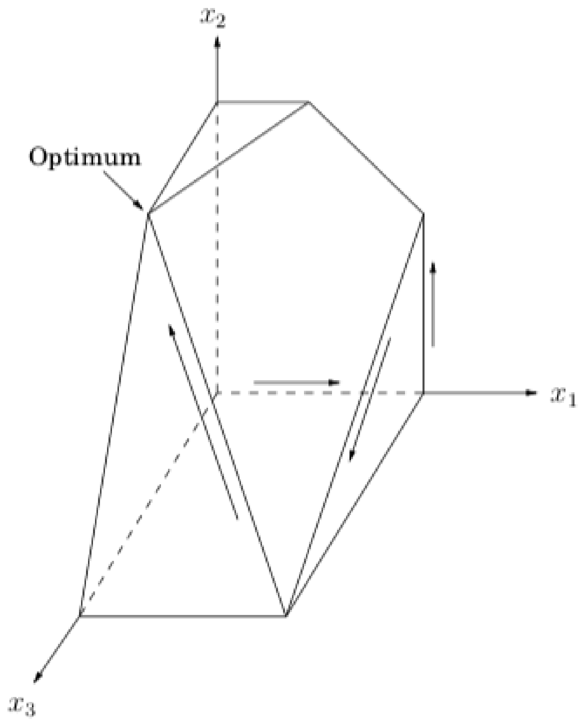
Does *hill-climbing* on the vertices of the polygon, from neighbor to neighbor so as to steadily increase profit along the way (Local Search)

$$
\begin{array}{ll}
\text{Objective function} & \max x_1 + 6x_2 \\
\text{Constraints} & x_1 \leq 200 \\
& x_2 \leq 300 \\
& x_1 + x_2 \leq 400 \\
& x_1, x_2 \geq 0
\end{array}
$$



The Simplex Method

In 3 dimensions:



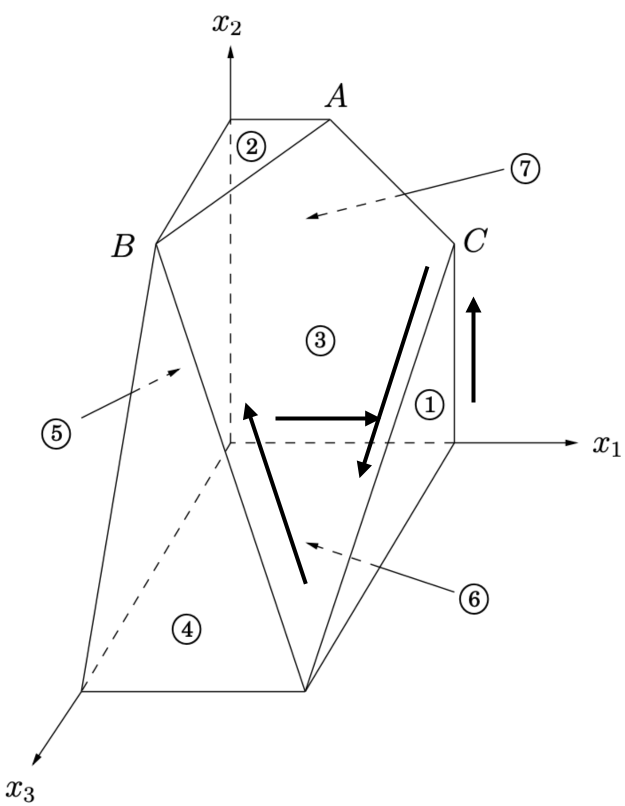
It would again move from vertex to vertex, along edges of the polyhedron, increasing profit steadily.

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

A possible trajectory

Vertices: (0,0,0) → (200,0,0) → (200,200,0) → (200,0,200) → (0,300,100)

The Simplex Method



$$
\begin{aligned}
\max \quad & x_1 + 6x_2 + 13x_3 & \\
& x_1 \le 200 & ① \\
& x_2 \le 300 & ② \\
& x_1 + x_2 + x_3 \le 400 & ③ \\
& x_2 + 3x_3 \le 600 & ④ \\
& x_1 \ge 0 & ⑤ \\
& x_2 \ge 0 & ⑥ \\
& x_3 \ge 0 & ⑦
\end{aligned}
$$

A possible trajectory

Vertices: (0,0,0) → (200,0,0) → (200,200,0) → (200,0,200) → (0,300,100)

The Simplex Method

Why are we interested in checking only neighboring corner points?

Optimality test for linear programs:

Consider an LP with at least one optimal solution. If a corner point solution has no adjacent corner point solutions that are better, according to the objective function, then it must be an optimal solution

- Hence, local optimality $\Rightarrow$ global optimality
- Very important property for linear programming
    - Also generalizes to continuous, convex functions

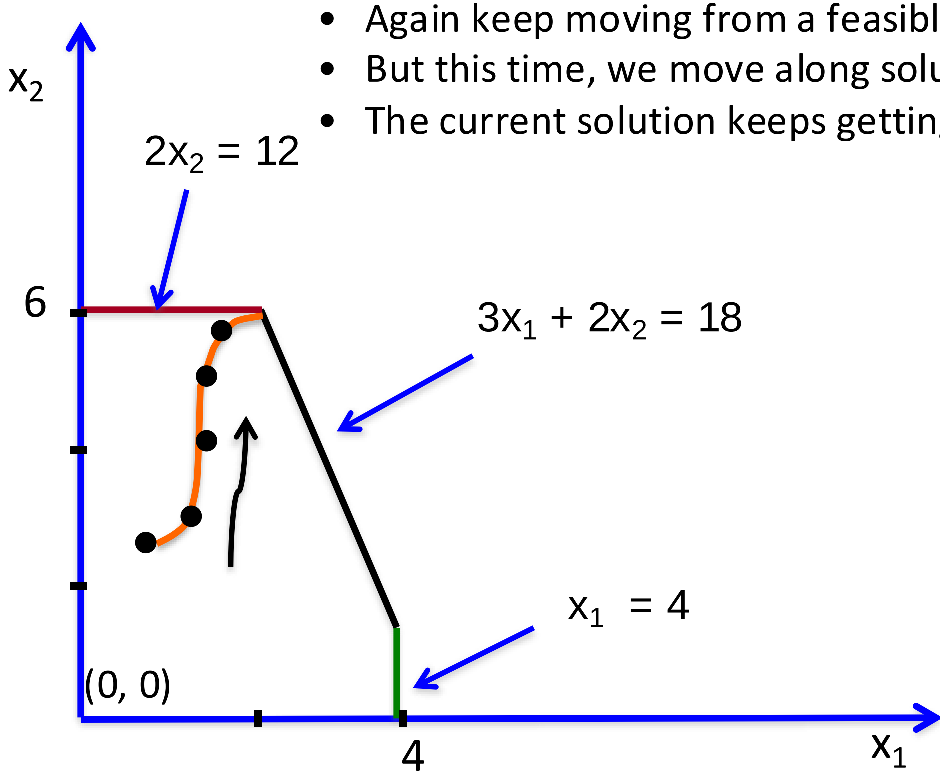
Complexity of Simplex

Extremely well-behaved in practice

- Empirically, number of iterations in simplex looks proportional to number of constraints
- Can we have a good theoretical upper bound on the number of iterations?
- NO! There are examples that need an exponential (2^n) number of iterations, discovered first by [Klee, Minty ’72]
- Despite that, it is still one of the preferred algorithms for solving linear programs!

Other Algorithms

- **The ellipsoid method:** The first polynomial time algorithm
  - By [Kachiyan '79], however not well behaved in practice
- **Interior point methods:** also polynomial time algorithms
  - First conceived by Karmarkar [1984]
  - Main ideas:
    - Again keep moving from a feasible solution to a better one
    - But this time, we move along solutions in the interior of the polytope
    - The current solution keeps getting closer and closer to a vertex of the polytope



Simplex vs Interior Point Algorithms

- **Comparisons**
  - In theory: interior point methods are polynomial time algorithms (for any n and m), simplex may need exponential time
  - In practice: average case complexity of simplex very low compared to worst case
  - One iteration of interior point methods needs much more computation time than in simplex to decide the next feasible solution
  - But: as the number of constraints increases, interior point methods do not need much more iterations
    - Interior point methods go through the internal part of the polytope
    - Adding more constraints reduces the feasible region, by adding more constraint boundaries

Method	Typical cost	Worst case cost
Simplex	$O(n^2m)$	Very bad
Ellipsoid	$O(n^8)$	$O(n^8)$

Everything you need to know about solving linear programs

Integer Programming

Integer Programming

What is an integer program?

- A way to model problems where some variables take integer values
- Also referred to as Integer Linear Program (ILP):
- Almost the same as Linear Programs
    - Linear objective function
    - Linear constraints

Applications:

- Comparable to applications of Linear Programming
- Operations Research
- Airline scheduling problems
- Medicine
- etc

Integer Programming Formulations

- It is not always clear how to model a problem as an integer program
- The tricky part is how to express the objective function using integer variables
- Usually: Assign a binary variable x_i to a candidate object that can be included in a solution
- Interpretation:

$$x_i = \begin{cases} 1, & \text{if item } i \text{ is in the solution} \\ 0, & \text{otherwise} \end{cases}$$

Integer Programming Formulations

Examples:

0-1 KNAPSACK:

I: A set of objects S = {1,…,n}, each with a positive integer weight w_i, and a value v_i, i=1,…,n, and a positive integer W

Q: find A $\subseteq$ S s.t. $\sum_{i \in A} w_i \leq W$ and $\sum_{i \in A} v_i$ is maximized

Equivalent IP formulation:

max $\Sigma_i v_i x_i$

s.t.

$\Sigma_i w_i x_i \leq W$

$x_i \in \{0,1\} \quad \forall\ i \in \{1,\ldots,n\}$

Integer Programming Formulations

Examples:

VERTEX COVER (VC):

I: A graph $G = (V,E)$

Q: Find $S \subseteq V$ s.t. $\forall$ $(u, v) \in E$ either $u \in S$ or $v \in S$ (or both) and $|S|$ is maximized

Equivalent IP formulation:

min $\Sigma_u x_u$

s.t.

$x_u + x_v \geq 1 \quad \forall (u, v) \in E$

$x_u \in \{0,1\} \quad \forall u \in V$

Integer Programming Formulations

Examples:

<u>**MAKESPAN**</u> **($P||C_{max}$)**

I: A set of objects S = {1,…,n}, each with a positive integer weight w_i, i = 1,…,n, and a positive integer M

Q: find a partition of S into $A_1, A_2, \ldots, A_M$ s.t. $\max_{1 \le j \le M} \{ \sum_{i \in A_j} w_i \}$ is minimized

Integer Programming Formulations

Examples:

MAKESPAN:

- Better to think of it as a job scheduling problem
- Items correspond to jobs that should be assigned to machines
- The weight corresponds to the processing time
- How do we model that a job j is assigned to machine i?

Equivalent IP formulation:

min t

s.t.

$\Sigma_j w_j x_{ij} \leq t$ $\forall i \in \{1,\ldots,m\}$ (The total processing time in each machine should be less or equal than the makespan

$\Sigma_i x_{ij} = 1$ $\forall j \in \{1,\ldots,n\}$ (every job must be assigned to exactly one machine)

$x_{ij} \in \{0,1\}$ $\forall j \in \{1,\ldots,n\}, i \in \{1,\ldots,m\}$

Complexity of Integer Programming

- Modeling a problem as an integer program does not provide any guarantee that we can solve it

Theorem: Integer Programming is NP-complete

- In fact many problems in discrete optimization are NP-complete
- Partly due to the discrete nature
- All such problems can be reduced to SAT and vice versa

Is this the end of the world?