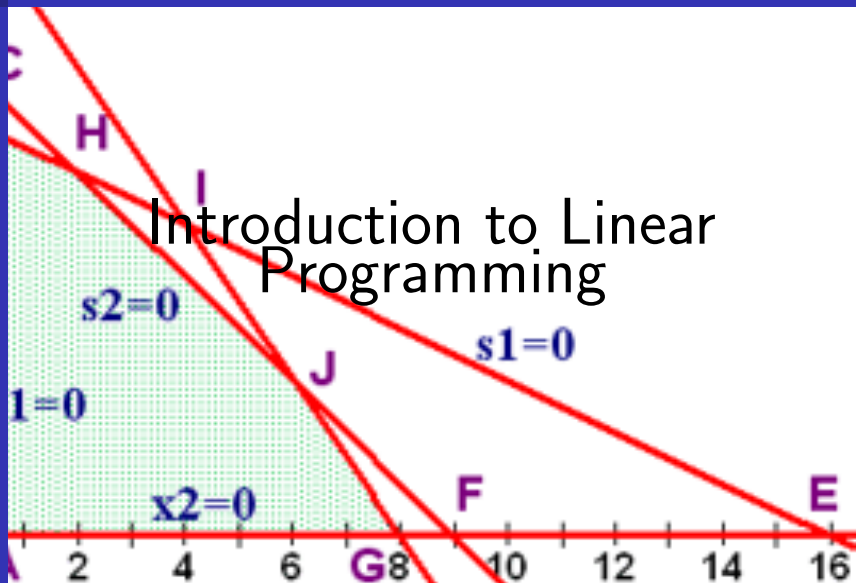


# Introduction to Linear Programming



# 1 Linear Programming

## 2 Duality

## 3 Integer LP

## 4 LP to devise approximation algorithms

# Linear Programming.

## Linear Programming

### Duality

### Integer LP

### LP to devise approximation algorithms

In a linear programming problem we are given a set of **variables**, an **objective function** a set of **linear constraints** and want to assign real values to the variables as to:

- satisfy the set of linear equations,
- maximize or minimize the objective function.

LP is of special interest because many combinatorial optimization problems can be reduced to LP: Max-Flow; Assignment problems; Matchings; Shortest paths; MinST; ...

## Example.

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

A company produces 2 products P1, and P2, and wishes to maximize the profits.

Each day, the company *can produce*  $x_1$  units of P1 and  $x_2$  units of P2.

The company *makes a profit* of 1 for each unit of P1; and a profit of 6 for each unit of P2.

Due to supply limitations and labor constraints we have the following additional constraints:  $x_1 \leq 200$ ,  $x_2 \leq 300$  and  $x_1 + x_2 \leq 400$ .

*What are the best levels of production?*

We format this problem as a **linear program**:

$$\begin{array}{ll}\text{Objective function:} & \max(x_1 + 6x_2) \\ \text{subject to the constraints:} & x_1 \leq 200 \\ & x_2 \leq 300 \\ & x_1 + x_2 \leq 400 \\ & x_1, x_2 \geq 0.\end{array}$$

Recall a linear equation in  $x_1$  and  $x_2$  defines a line in  $\mathbb{R}^2$ . A linear inequality define a half-space.

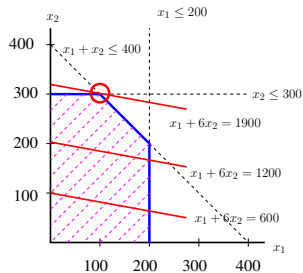
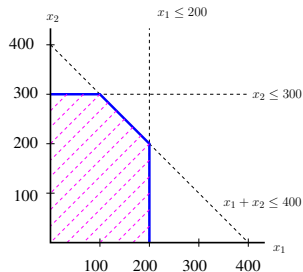
The **feasible region** of this LP are the  $(x_1, x_2)$  in the convex polygon defined by the linear constrains.

# Linear Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms



In a linear program *the optimum is achieved at a vertex of the feasible region.*

A LP is **infeasible** if

- The constraints are so tight that there are impossible to satisfy all of them. For ex.  $x \geq 2$  and  $x \leq 1$ ,
- The constraints are so loose that the feasible region is unbounded. For ex.  $\max(x_1 + x_2)$  with  $x_1, x_2 \geq 0$

## Higher dimensions.

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

The company produces products P1, P2 and P3, where each day it produces  $x_1$  units of P1,  $x_2$  units of P2 and  $x_3$  units of P3. and makes a profit of 1 for each unit of P1, a profit of 6 for each unit of P2 and a profit of 13 for each unit of P3. Due to supply limitations and labor constrains we have the following additional constrains:  $x_1 \leq 200$ ,  $x_2 \leq 300$ ,  $x_1 + x_2 + x_3 \leq 400$  and  $x_2 + 3x_3 \leq 600$ .

$$\max(x_1 + 6x_2 + 13x_3)$$

$$x_1 \leq 200$$

$$x_2 \leq 300$$

$$x_1 + x_2 + x_3 \leq 400$$

$$x_2 + 3x_3 \leq 600$$

$$x_1, x_2, x_3 \geq 0.$$

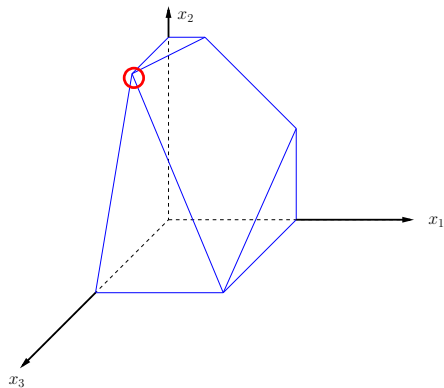


# Linear Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms



# Standard form of a Linear Program.

**INPUT:** Given real numbers  $(c_i)_{i=1}^n, (a_{ji})_{1 \leq j \leq m \& 1 \leq i \leq n} (b_i)_{i=1}^n$

**OUTPUT:** real values for variables  $(x_i)_{i=1}^n$

A **linear programming problem** is the problem of maximizing (minimizing) a linear function **the objective function**

$P = \sum_{i=1}^n c_i x_i$  subject to finite set of **linear constraints**

A LP is in **standard form** if the following are true:

- Non-negative constraints for all variables.
- All remaining constraints are expressed as = constraints.
- All  $b_i \geq 0$ .

$$\max \sum_{i=1}^n c_i x_i,$$

subject to:

$$\sum_{i=1}^n a_{ji} x_i = b_j, \quad 1 \leq j \leq m$$

$$x_i \geq 0, \quad 1 \leq i \leq n$$

# Equivalent formulations of LP.

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

A LP has many degrees of freedom:

- 1 It can be a maximization or a minimization problem.
- 2 Its constraints could be equalities or inequalities.
- 3 The variables are often restricted to be non-negative, but they also could be unrestricted.

Most of the "*real life*" constraints are given as inequalities.

The main reason to convert a LP into standard form is because the [simplex algorithm](#) starts with a LP in standard form.

But it could be useful the flexibility to be able to change the formulation of the original LP.

# Transformations among LP forms

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

- *To convert inequality  $\sum_{i=1}^n a_i x_i \leq b_i$  into equality:*  
introduce a **slack variable**  $s_i$  to get  $\sum_{i=1}^n a_i x_i + s_i = b_i$   
with  $s \geq 0$ .

The slack variable  $s_i$  measures the amount of “non-used resource.”

**Ex:**  $x_1 + x_2 + x_3 \leq 40 \Rightarrow x_1 + x_2 + x_3 + s_1 = 40$

So that  $s_1 = 40 - (x_1 + x_2 + x_3)$

- *To convert inequality  $\sum_{i=1}^n a_i x_i \geq -b$  into equality:*  
introduce a **surplus variable** and get  $\sum_{i=1}^n a_i x_i - s_i = b_i$   
with  $s \geq 0$ .

The surplus variable  $s_i$  measures the extra amount of used resource.

**Ex:**  $-x_1 + x_2 - x_3 \geq 4 \Rightarrow -x_1 + x_2 - x_3 - s_1 = 4$

# Transformations among LP forms (cont.)

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

- *To deal with an unrestricted variable  $x$  (i.e.  $x$  can be positive or negative):* introduce  $x^+, x^- \geq 0$ , and replace all occurrences of  $x$  by  $x^+ - x^-$ .

Ex:  $x$  unconstrained  $\Rightarrow x = x^+ - x^-$  with  $x^+ \geq 0$  and  $x^- \geq 0$ .

- *To turn max. problem into min. problem:* multiply the coefficients of the objective function by -1.

Ex:  $\max(10x_1 + 60x_2 + 140x_3) \Rightarrow$   
 $\min(-10x_1 - 60x_2 - 140x_3).$

Applying these transformations, we can rewrite any LP into standard form, in which variables are all non-negative, the constraints are equalities, and the objective function is to be minimized.

# Example:

## Linear Programming

### Duality

### Integer LP

### LP to devise approximation algorithms

$$\max(10x_1 + 60x_2 + 140x_3)$$

$$x_1 \leq 20$$

$$x_2 \leq 30$$

$$x_1 + x_2 + x_3 \leq 40$$

$$x_2 + 3x_3 \leq 60$$

$$x_1, x_2, x_3 \geq 0.$$

$$\min(-10x_1 - 60x_2 - 140x_3)$$

$$x_1 + s_1 = 20$$

$$x_2 + s_2 = 30$$

$$x_1 + x_2 + x_3 + s_3 = 40$$

$$x_2 + 3x_3 + s_5 = 60$$

$$x_1, x_2, x_3, s_1, s_2, s_3, s_4, s_5 \geq 0.$$

# Algebraic representation of LP

Let  $c = (c_1, \dots, c_n)$ ,  $x = (x_1, \dots, x_n)$ ,  $b = (b_1, \dots, b_m)$  and let  $A$  be the  $m \times n$  matrix of the coefficients involved in the constraints.

A LP can be represented using matrix and vectors:

$$\begin{array}{ll} \max \sum_{i=1}^n c_i x_i & \max \sum_{i=1}^n c^T x \\ \text{subject to} & \Rightarrow \text{subject to} \\ \sum_{i=1}^n c_i x_i \leq b_j, \ 1 \leq j \leq m & Ax \leq b \\ x_i \geq 0, \ 1 \leq i \leq n & x \geq 0 \end{array}$$

Given a LP

$$\begin{array}{ll}\min & c^T x \\ \text{subject to} & \\ & Ax \geq b \\ & x \geq 0\end{array}$$

Any  $x$  that satisfies the constraints is a *feasible solution*.

A LP is *feasible* if there exists a feasible solution. Otherwise is said to be *infeasible*.

A feasible solution  $x^*$  is an *optimal solution* if

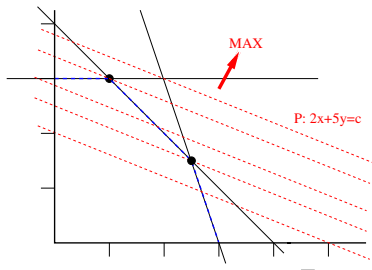
$$c^T x^* = \min\{c^T x \mid Ax \geq b, x \geq 0\}$$



# The Geometry of LP

Consider  $P$ :

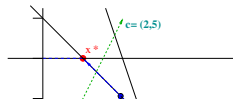
$$\begin{aligned} \max \quad & 2x + 5y \\ \text{s.t.} \quad & 3x + y \leq 9 \\ & y \leq 3 \\ & x + y \leq 4 \\ & x, y \geq 0 \end{aligned}$$



## Theorem

*If there exists an optimal solution to  $P$ ,  $x^*$ , then there exists one that is a vertex of the polytope.*

**Intuition of proof** If  $x$  is not a vertex, move in a non-decreasing



Linear  
Programming

Duality

Integer LP

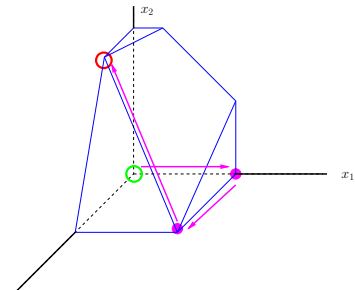
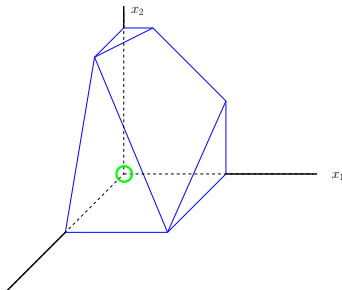
LP to devise  
approximation  
algorithms

# The Simplex algorithm

*LP can be solved efficiently:* George Dantzing (1947)



It uses **hill-climbing**: Start in a vertex of the feasible polytope and look for an adjacent vertex of better objective value. Until reaching a vertex that has no neighbor with better objective function.



Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

# Complexity of LP:

Input to LP: The number  $n$  of variables in the LP.

Simplex could be exponential on  $n$ : there exists specific input (the Klee-Minty cube) where the usual versions of the simplex algorithm may actually "cycle" in the path to the optimal. (see Ch.6 in Papadimitriou-Steiglitz, *Comb. Optimization: Algorithms and Complexity*)

In practice, the simplex algorithm is quite efficient and can find the global optimum (if certain precautions against cycling are taken).

It is known that simplex solves "typical" (random) problems in  $O(n^3)$  steps.

Simplex is the main choice to solve LP, among engineers.

But some software packages use interior-points algorithms, which guarantee poly-time termination.

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

1 Linear Programming

2 Duality

3 Integer LP

4 LP to devise approximation algorithms

# LP Duality

Most important concept in LP.

Allows proofs of optimality and approximation.

LP P (PRIMAL)

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

- Want to obtain upper bounds to  $OPT(P)$
- The best UB?

Take a linear combination of equations:  $y^T Ax$   
 $y^T Ax \geq y^T b$ , because of the restrictions, for any  $x$ ,  
we can select  $y$  so that  $y^T A = c^T$ ,  $c^T x = y^T Ax \geq y^T b$ ,  
With the hope that the highest value could be close to  
 $OPT(P)$ .

# The best lower bound for any $x$ ?

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

$$\begin{array}{ll}\max & b^T y \\ \text{s.t.} & A^T y = c \\ & y \geq 0\end{array}$$

But as we are maximizing this is equivalent to

$$\begin{array}{ll}\max & b^T y \\ \text{s.t.} & A^T y \leq c \quad \text{D (DUAL)} \\ & y \geq 0\end{array}$$

# Primal-Dual

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

P (PRIMAL)

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

$$\begin{array}{ll}\max & 2x_1 + 3x_2 \\ & 4x_1 + 8x_2 \leq 12 \\ & 2x_1 + x_2 \leq 3 \\ & 3x_1 + 2x_2 \leq 4 \\ & x, y \geq 0\end{array}$$

D (DUAL)

$$\begin{array}{ll}\max & b^T y \\ \text{s.t.} & A^T y \leq c \\ & y \geq 0\end{array}$$

$$\begin{array}{ll}\min & 12y_1 + 3y_2 + 4y_3 \\ & 4y_1 + 2y_2 + 3y_3 \geq 2 \\ & 8y_1 + y_2 + 2y_3 \geq 3 \\ & y_1, y_2, y_3 \geq 0,\end{array}$$

# Duality Theorem

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

The following theorem is proved in any course in OR, and it is a consequence of the fact that *every feasible solution to the dual  $D$  is a lower bound on the optimum value of the primal  $P$ , and vice versa.*

## Theorem (LP-duality theorem)

*Let  $x^* = (x_1^*, \dots, x_n^*)$  be a finite optimal solution for the primal  $P$ , and let  $y^* = (y_1^*, \dots, y_m^*)$  be the finite optimal for the dual  $D$ , then*

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



# Consequences of the P-D Theorem

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

If P and D are the primal and the dual of a LP, then one of the four following cases occurs:

- 1 Both P and D are infeasible.
- 2 P is unbounded and D is infeasible.
- 3 D is unbounded and P is infeasible.
- 4 Both are feasible and there exist optimal solutions  $x^*$  to P and  $y^*$  to D such that  $c^T x^* = b^T y^*$ .

# Linear programming formulation of max-flow.

$$\max f_{sa} + f_{sb} + f_{sc}$$

$$f_{sa} + f_{ba} = f_{ad}$$

$$f_{sc} + f_{dc} = f_{ce}$$

$$f_{sb} = f_{bd} + f_{ba}$$

$$f_{ad} + f_{bd} = f_{dc} + f_{de} + f_{dt}$$

$$f_{ce} + f_{de} = f_{et}$$

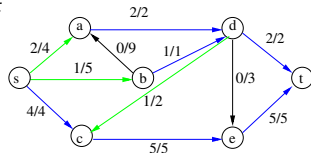
$$f_{sa} \leq 3; f_{sb} \leq 5; f_{dt} \leq 2; f_{sc} \leq 4$$

$$f_{ba} \leq 9; f_{ad} \leq 2; f_{bd} \geq 1; f_{et} \leq 5$$

$$f_{dc} \leq 1; f_{ce} \leq 5; f_{de} \geq 1$$

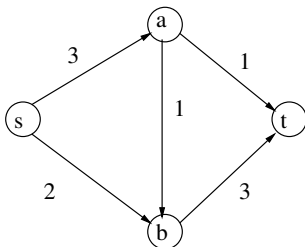
$$f_{sa}, f_{sb}, f_{sc}, f_{ba}, f_{ad}, f_{bd} \geq 0$$

$$f_{dc}, f_{ce}, f_{de}, f_{dt}, f_{et} \geq 0.$$



# Example: The Max-Flow problem

The max-flow min-cut theorem is a special case of duality.



$$\max f_{sa} + f_{sb}$$

$$f_{sa} \leq 3$$

$$f_{sb} \leq 2$$

$$f_{ab} \leq 1$$

$$f_{at} \leq 1$$

$$f_{bt} \leq 3$$

$$f_{sa} - f_{ab} - f_{at} = 0$$

$$f_{sb} + f_{ab} - f_{bt} = 0$$

$$f_{sa}, f_{sb}, f_{ab}, f_{at}, f_{bt} \geq 0.$$

The **dual** of the previous LP:

$$\min 3y_{sa} + 2y_{sb} + y_{ab} + y_{at} + y_{bt}$$

$$y_{sa} + u_a \geq 1$$

$$y_{sb} + u_b \geq 1$$

$$y_{ab} - u_a + u_b \geq 0$$

$$y_{at} - u_a \leq 1$$

$$y_{bt} - u_b \leq 3$$

$$y_{sa}, y_{sb}, y_{ab}, y_{at}, y_{bt}, u_a, u_b \geq 0.$$

This D - LP defines the **min-cut** problem where for  $x \in \{a, b\}$ ,  $u_x = 1$  iff vertex  $x \in S$ , and  $y_{xz} = 1$  iff  $(x, z) \in \text{cut}(S, T)$ .

*By the LP-duality Theorem, any optimal solution to max-flow  
= to any optimal solution to min-cut.*

# Min cost maximum flow

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

**Min Cost MaxFlow:** Given a flow network and a valuation of the cost of transporting a unit of flow along each edge. Find a maximum flow with minimum cost.

- Compute the value  $F$  of a maximum flow.
- Adapt the LP for MaxFlow to ensure that the flow value is  $F$  and incorporate the cost in the objective function.
- This approach provides a polynomial time algorithm for the problem.

Linear  
Programming

Duality

**Integer LP**

LP to devise  
approximation  
algorithms

1 Linear Programming

2 Duality

**3 Integer LP**

4 LP to devise approximation algorithms

# Integer Linear Programming (ILP)

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

Consider the **Min Vertex Cover problem**: Given an undirected  $G = (V, E)$  with  $|V| = n$  and  $|E| = m$ , want to find  $S \subseteq V$  with minimal cardinality s.t.. it covers all edges  $e \in E$ .

- This problem can be expressed as a linear program on  $\{0, 1\}$  variables, interpreting a solution as  
Let  $x \in \{0, 1\}^n$  be seen as a set  $S$ , in the usual way, for  $i \in V$ :

$$x_i = \begin{cases} 1 & \text{if } i \in S \\ 0 & \text{otherwise} \end{cases}$$

- Under this interpretation we the constraints  $\forall (i, j) \in E$   
 $x_i + x_j \geq 1$  are equivalent to say that  $S$  is a vertex cover.  
The constraints give  $Ax \geq 1$ .

# Integer Linear Programming

We can express the min VC problem as:

$$\begin{aligned} \min \quad & \sum_{i \in V} x_i \\ \text{subject to} \quad & x_i + x_j \geq 1, (i, j) \in E \\ & x_i \in \{0, 1\}, i \in V \end{aligned}$$

where we have a new constrain, **we require the solution to be 0,1**. This can be replaced by requiring the variables to be positive integers (as we are minimizing).

Asking for the best possible **integral** solution for a LP is known as the **Integer Linear Programming**:

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms



# Integer Linear Programming

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

The ILP problem is defined:

Given  $A \in \mathbb{Z}^{n \times m}$  together with  $b \in \mathbb{Z}^n$  and  $c \in \mathbb{Z}^m$ , find a  $x$  that max (min)  $c^T x$  subject to:

$$\begin{aligned} &\min c^T x \\ &\text{subject to} \\ &Ax \geq b \\ &x \in \mathbb{Z}^m, \end{aligned}$$

Big difference between LP and ILP:

Ellipsoidal/Interior point methods solve LP in polynomial time  
but ILP is NP-hard.

# Solvers for LP

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

Due to the importance of LP and ILP as models to solve optimization problem, there is a very active research going on to design new algorithms and heuristics to improve the running time for solving LP (algorithms) IPL (heuristics).

There are a myriad of solvers packages:

- **GLPK:** <https://www.gnu.org/software/glpk/>
- **LP-SOLVE:** <http://www3.cs.stonybrook.edu/>
- **CPLEX:**  
<http://ampl.com/products/solvers/solvers-we-sell/cplex/>
- **GUROBI Optimizer:**  
<http://www.gurobi.com/products/gurobi-optimizer>

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

1 Linear Programming

2 Duality

3 Integer LP

4 LP to devise approximation algorithms

# LP to approximate solving ILP?

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

- Starting from an ILP we can relax the integrality restrictions to get a LP (the relaxed version)
- We can solve the LP in polynomial time either
  - solve the LP and find a way to round the values in the optimal solution to construct a valid solution for the ILP
  - solve the LP using primal-dual to maintain a feasible solution for the ILP and a feasible solution of the dual, and improve it the as much as possible
- in the hope that the algorithm provides a good approximation rate.

# Vertex cover: LP relaxation

$$\begin{array}{ll}\text{IP} & \\ \min & \sum_{i=1}^n x_i \\ \text{s.t.} & x_i + x_j \geq 1 \quad \text{for all } (i, j) \in E \\ & x_i \in \{0, 1\} \quad \text{for all } i \in V\end{array}$$

$$\begin{array}{ll}\text{LP} & \\ \min & \sum_{i=1}^n x_i \\ \text{s.t.} & x_i + x_j \geq 1 \quad \text{for all } (i, j) \in E \\ & x_i \geq 0 \quad (x_i \in (0, 1)) \quad \text{for all } i \in V\end{array}$$

- Why can we drop  $x_i \leq 1$ ?
- Let **opt** be the size of an optimal solution of the VC instance.
- Let  $x^*$  be an optimal solution of the LP and  $s^* = \sum_{i=1}^n x_i^*$ .
- Is there any relationship between  $s^*$  and **opt**?  $s^* \leq \text{opt}$

# Vertex Cover: Rounding

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

- Relax+Round
  - Solve the relaxed LP. Let  $x^*$  be an optimal solution of the LP
  - For  $i \in V$ , if  $x_i^* \geq 1/2$ , set  $x_i = 1$  and otherwise set  $x_i = 0$ .
- $x$  is a feasible solution as every edge is covered.
- $\sum_{i \in V} x_i \leq 2 \sum_{i \in V} x_i^* \leq 2 \text{ opt}$
- Relax+Round is a 2-approximation for VC (in polynomial time).

# Vertex cover: Primal-Dual approximation

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

LP primal

$$\begin{array}{ll}\min & \sum_{i=1}^n x_i \\ \text{s.t.} & x_i + x_j \geq 1 \quad \text{for all } (i,j) \in E \\ & x_i \geq 0 \quad \text{for all } i \in V\end{array}$$

LP dual

$$\begin{array}{ll}\max & \sum_{e \in E} z_e \\ \text{s.t.} & \sum_{e=\{i,j\} \in E} z_e \leq 1 \quad \text{for all } i \in V \\ & z_e \geq 0 \quad \text{for all } e \in E\end{array}$$

- Start with the integer infeasible primal solution  $x = 0$ , and the dual feasible solution  $z = 0$ .
- Repeat while some constraint in primal  $(i,j)$  is unsatisfied:
  - Increase all (unfrozen) variables  $z_e$  incidents with  $i$  until its dual constraint becomes tight.
  - Set  $x_j = 1$ . Freeze all the increased variables  $z_e$  ( $i \in e$ ). If some dual constraint becomes tight set  $x_j = 1$ .

# Vertex cover: Primal-Dual approximation

Linear  
Programming

Duality

Integer LP

LP to devise  
approximation  
algorithms

- When the process stops, we have increased the variables  $z_e$  suitably.
- Some vertices  $i$  were chosen ( $x_i = 1$ )
- This set  $S$  of vertices is our output.
- Is  $S$  a vertex cover.  
Otherwise, we would have still an edge  $e$  with  $x_i + x_j < 1$  which is impossible
- Cost of the solution computed by alg?  
At the end of the algorithm, if  $z_e > 0$ ,  $z_e$  has been frozen. We do not know if for one (or both) endpoints the constraints got tight, but  $x_i + x_j \leq 2$ , for  $e = (i, j)$ . Therefore, we get a lower bound of 1 and an upper bound of 2. Which means that we have a 2-approximation algorithm.