5 Integer Linear Programming (ILP)

Definition: An Integer Linear Programming problem is an optimization problem of the form

$$\text{(ILP)} \quad \min \quad \underline{c}^T \underline{x}$$
$$A\underline{x} \geq \underline{b}$$
$$\underline{x} \geq \underline{0} \quad \text{with } \underline{x} \in \mathbb{Z}^n \quad (\underline{x} \text{ integer}).$$

If $x_j \in \{0, 1\}$ for all j, binary LP.

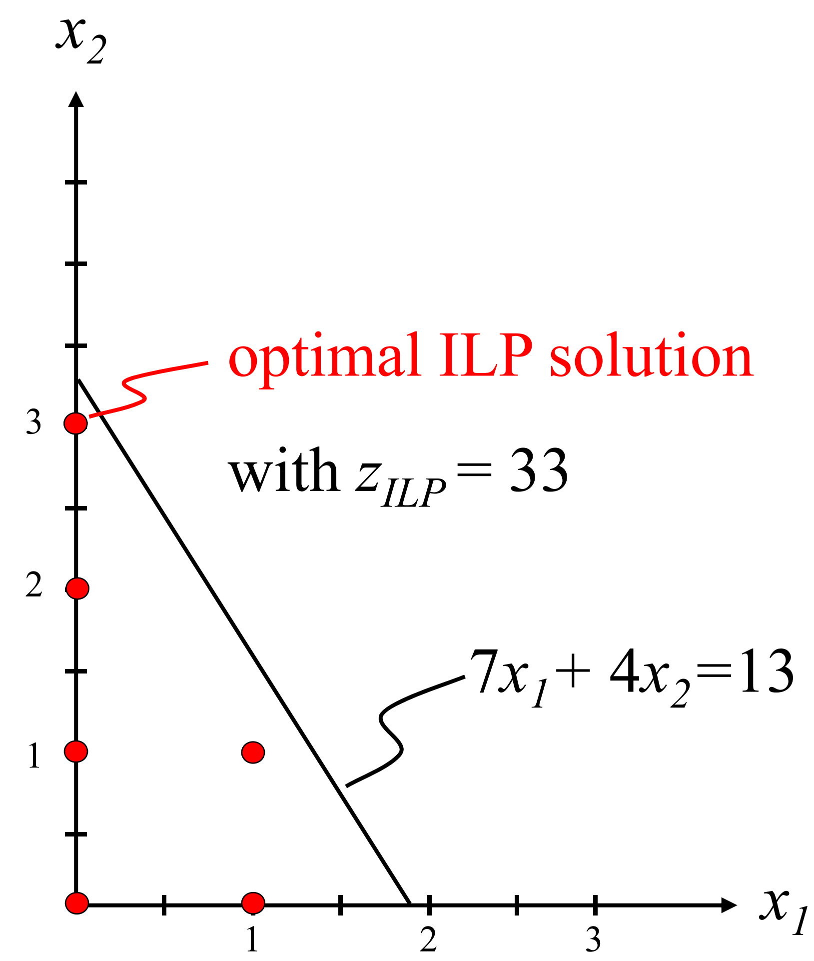
If not all x_j are integer, mixed integer LP.

Assumption: parameters A, $\underline{b}$ integer (without loss of generality).

Observation: The integrality condition $x_j \in \mathbb{Z}$ is non linear since it can be expressed as $\sin(\pi x_j) = 0$.

Example:

$$z_{ILP} = \max\ z = 21x_1 + 11x_2$$
$$s.t.\quad 7x_1 + 4x_2 \leq 13$$
$$x_1, x_2 \geq 0 \ \text{ integer}$$

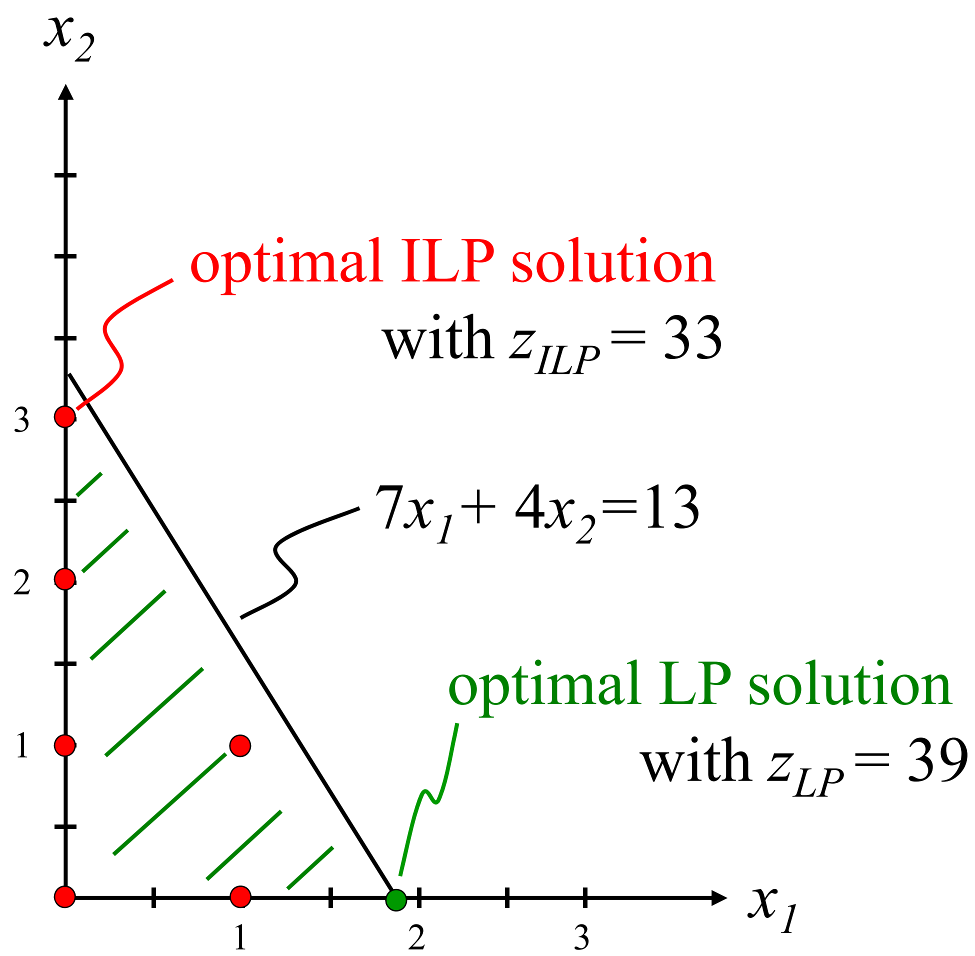


ILP feasible region ≡ lattice (with a finite or infinite number of points).

Example cont.:

By deleting the integrality constraints, we obtain the following Linear Program :

$$z_{LP} = \max\ z = 21x_1 + 11x_2$$
$$s.t. \quad 7x_1 + 4x_2 \leq 13$$
$$x_1, x_2 \geq 0$$



ILP feasible region ≡ lattice (with a finite or infinite number of points)

Definition: Let

$$
\text{(ILP)} \qquad \begin{aligned} z_{ILP} := \max\ & \underline{c}^T \underline{x} \\ A\underline{x} &\le \underline{b} \\ \underline{x} &\ge \underline{0} \quad \text{integer} \end{aligned} \Bigg\} \; X_{ILP}
$$

The problem

$$
\text{(LP)} \qquad \begin{aligned} z_{LP} := \max\ & \underline{c}^T \underline{x} \\ A\underline{x} &\le \underline{b} \\ \underline{x} &\ge \underline{0} \end{aligned} \Bigg\} \; X_{LP} \supseteq X_{ILP}
$$

is the *linear (continuous) relaxation* of (ILP).

Property: For any ILP with max, we have $z_{ILP} \le z_{LP}$, i.e., z_{LP} is an upper bound on the optimal value of (ILP).

Observation: For any ILP with min, we have $z_{ILP} \ge z_{LP}$, i.e., z_{LP} is a lower bound on the optimal value of (ILP).

First idea: relax the integrality constraints of (ILP) and round up/down the optimal solution of the linear relaxation (LP).

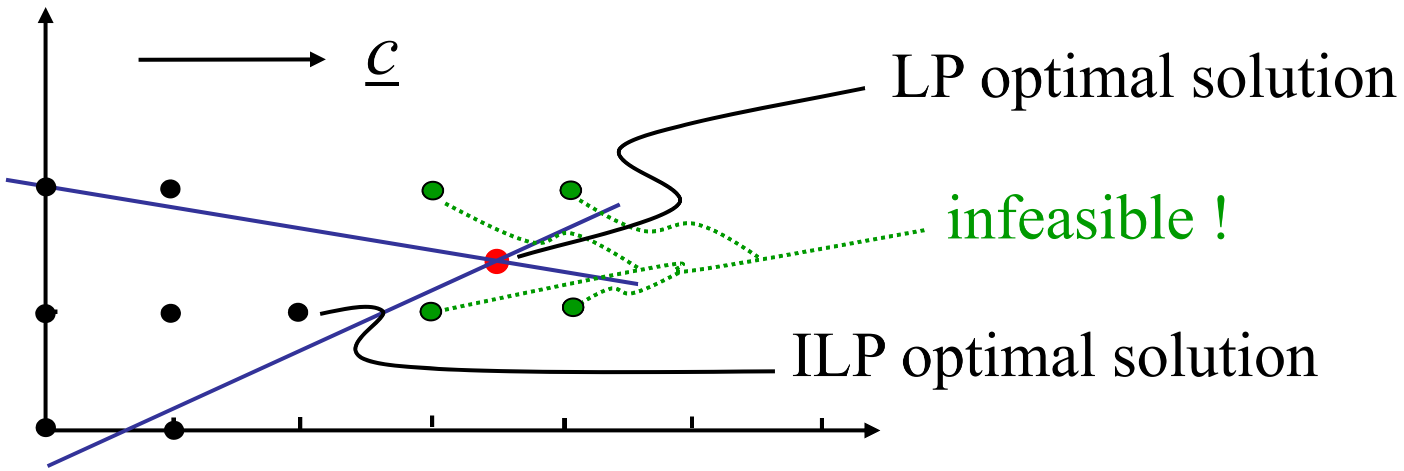
If an optimal solution of (LP) is integer, then it is also an optimal solution of (ILP).

But often the rounded optimal solutions of (LP) are:

- infeasible for (ILP),

- useless -- very different from an optimal solution of (ILP).

- Infeasible rounded solutions



- Useless rounded solutions

When the integer variables take small values at optimality.

E.g., binary assignment variables (job to machine) or activation variables (plants),...

- Useful rounded solutions

When the integer variables take large values at optimality.

E.g., number of pieces to produce,…

Observation: It also depends on the unit costs (coefficients of the objective function).

Example 1: Knapsack problem

n	objects $j = 1, \ldots, n$
p_j	profit (value) of object j
v_j	volume (weight) of object j
b	maximum knapsack capacity.

Determine a subset of objects that maximizes the total profit, while respecting the knapsack capacity.

Variables: $x_j = \begin{cases} 1 & j\text{-th object is selected} \\ 0 & \text{otherwise} \end{cases}$

$$\max \sum_{j=1}^{n} p_j x_j$$

$$\sum_{j=1}^{n} v_j x_j \leq b$$

$$x_j \in \{0,1\} \quad \forall\, j$$

Wide range of direct and indirect applications:

- loading (containers, vehicles, CDs,…),
- investments (p_j = expected return, v_j = amount to invest, available capital),
- as a suproblem…

Example 2: Assignment problem

m	machines	$i = 1, \ldots, m$
n	jobs	$j = 1, \ldots, n$
c_{ij}	cost of assigning job j to machine i	

Assumption: $n > m$

Determine an assignment of the jobs to the machines so as to minimize the total cost, while assigning at least one job per machine and at most one machine for each job.

Variables: $x_{ij} = \begin{cases} 1 & \text{machine } i \text{ execute job } j \\ 0 & \text{otherwise} \end{cases}$

$$\min \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}$$

$$\sum_{i=1}^{m} x_{ij} \leq 1 \quad \forall\, j = 1, \ldots, n$$ (at most one machine for each job)

$$\sum_{j=1}^{n} x_{ij} \geq 1 \quad \forall\, i = 1, \ldots, m$$ (at least one job for each machine)

$$x_{ij} \in \{0,1\} \quad \forall\, i, j$$

Example 3: Transportation problem (single product)

m	production plants	$i = 1,\dots, m$
n	clients	$j = 1,\dots, n$
c_{ij}	transportation cost of one unit of product from plant i to client j	
p_i	production capacity of plant i	
d_j	demand of client j	
q_{ij}	maximum amount to be transported from plant i to client j	

Determine a transportation plan that minimizes total costs while satisfying plant capacitiy and client demands.

<u>Assumption</u>: $\sum_{i=1}^{m} p_i \geq \sum_{j=1}^{n} d_j$

Variables: x_{ij} = amount transported from plant i to client j

$$\min \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}$$

$$\sum_{j=1}^{n} x_{ij} \leq p_i \qquad \forall\, i = 1, \ldots, m \qquad \text{(plant capacity)}$$

$$\sum_{i=1}^{m} x_{ij} \geq d_j \qquad \forall\, j = 1, \ldots, n \qquad \text{(client demand)}$$

$$0 \leq x_{ij} \leq q_{ij} \qquad \forall\, i, j \qquad \text{(transportation capacity)}$$

Property of transportation and assignment problems:

Optimal solution of the linear relaxation ≡ optimal solution of the ILP !

Theorem: If in a Transporation problem the right hand side terms are integer, all the basic feasible solutions (vertices) of its linear relaxation are integer.

- In transportation problem, special $(mn+n+m)\times(mn)$ integer matrix A of the contraints: $a_{ij} \in \{-1, 0, 1\}$ with exactly three nonzero coefficients per column.
- Right hand side vector $\underline{b}$ has all integer components.

Optimal solution of the linear relaxation:

$$\underline{x}^* = \begin{pmatrix} B^{-1}\underline{b} \\ \underline{0} \end{pmatrix} \qquad B^{-1} = \frac{1}{\det(B)} \begin{pmatrix} \alpha_{11} & \dots & \alpha_{1,m} \\ \dots & \dots & \dots \\ \alpha_{m1} & \dots & \alpha_{mm} \end{pmatrix}^T$$

where $\alpha_{ij} = (-1)^{i+j} \, det(M_{ij})$ with M_{ij} square sub-matrix obtained from B by eliminating row i and column j.

B integer $\Rightarrow \alpha_{ij}$ integer

If $det(B) = \pm 1 \;\Rightarrow\; B^{-1}$ integer $\;\Rightarrow\; \underline{x}^*$ integer.

In fact it can be proved that A is *totally unimodular*, that is

$det(Q) = \{-1,0,1\}$ $\forall$ squared sub-matrix Q of A.

Example 4: Scheduling problem

m machines $k = 1, \ldots, m$

n jobs $j = 1, \ldots, n \backslash$

For each job j, deadline d_j

p_{jk} = processing time of job j on machine k (may be = 0)

Assumption: Each job must be processed once on each machine following the order of the machine indices 1, 2, …, m.

Determine an optimal sequence in which to process the jobs so as to minimize the total completion time while satisfying the deadlines.

Variables:

t_{jk} = time at which the processing of job j starts on machine k

t = upper bound on the completion time of all jobs

$$y_{ijk} = \begin{cases} 1 & \text{if job } i \text{ preceeds job } j \text{ on machine } k \\ 0 & \text{otherwise} \end{cases}$$

Parameter $M := \sum_{j=1}^{n} d_j$

$\min \quad t$

$t_{jm} + p_{jm} \leq t \qquad \forall_j$ (t is upper bound on overall completion time)

$t_{jm} + p_{jm} \leq d_j \qquad \forall j$ (satisfy deadlines)

$t_{ik} + p_{ik} \leq t_{jk} + M(1\text{-}y_{ijk}) \qquad \forall i,j,k \quad i < j$ (*)

$t_{jk} + p_{jk} \leq t_{ik} + M\, y_{ijk} \qquad \forall i,j,k \quad i < j$ (**)

$t_{jk} + p_{jk} \leq t_{j,k+1} \qquad \forall j, k = 1, \ldots, m\text{-}1$ (job processing order)

$t \geq 0,\ t_{jk} \geq 0 \qquad \forall j,k$

$y_{ijk} \in \{0,1\} \qquad \forall i,j,k$

$\Rightarrow$ mixed ILP

(*) and (**) make sure that 2 jobs are not simultaneously processed on the same machine

(*) active when $y_{ijk} = 1$ (i preceeds j on machine k) and ensures that i is completed before j starts (on k)

(**) active when $y_{ijk} = 0$ (j preceeds i on machine k) and ensures that j is completed before i starts (on k).

The ILP formulation can be easily extended to the case where each job j must be processed on m machines (or on a subset of them) according to a different order.

Most ILP problems are $\mathcal{NP}$-hard.

$\nexists$ efficient algorithms to solve them and the existence of a polynomial time algorithm for any one would imply $\mathcal{P} = \mathcal{NP}$!

extremely unlikely

Type of methods

- implicit enumeration
- cutting planes

} exact methods (global optimum)

- heuristic algorithms ("greedy", local search,…)

approximate methods (local optimum)

Implicit enumeration methods explore all feasible solutions explicitly or implicitly.

- “Branch and Bound” method
- Dynamic programming (see optimal paths in acyclic graphs)