# Integer Linear Programming

Subhas C. Nandy
(nandysc@isical.ac.in)

Advanced Computing and Microelectronics Unit
Indian Statistical Institute
Kolkata 700108, India.

# Organization

# Linear Programming

A technique for optimizing a linear objective function, subject to a set of linear equality and linear inequality constraints.

Mathematically,

$$\begin{aligned}
&\text{maximize } c_1x_1 + c_2x_2 + \ldots + c_nx_n \\
&\text{Subject to:} \\
&\qquad a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1 \\
&\qquad a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leq b_2 \\
&\qquad \vdots \\
&\qquad \vdots \\
&\qquad a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \leq b_m \\
&\qquad x_i \geq 0 \text{ for all } i = 1, 2, \ldots, n.
\end{aligned}$$

# Linear Programming

**In matrix notation,**

maximize $C^T X$

Subject to:

$AX \leq B$

$X \geq 0$

where $C$ is a $n \times 1$ vector —- cost vector,

$A$ is a $m \times n$ matrix —- coefficient matrix,

$B$ is a $m \times 1$ vector —- requirement vector, and

$X$ is an $n \times 1$ vector of unknowns.

# History

- The linear programming method was first developed by Leonid Kantorovich in 1937.

# History

- The linear programming method was first developed by Leonid Kantorovich in 1937.
- He developed it during World War II as a way to plan expenditures and returns so as to reduce costs to the army and increase losses incurred by the enemy.

# History

- The linear programming method was first developed by Leonid Kantorovich in 1937.
- He developed it during World War II as a way to plan expenditures and returns so as to reduce costs to the army and increase losses incurred by the enemy.
- The method was kept secret until 1947 when George B. Dantzig published the simplex method and John von Neumann developed the theory of duality as a linear optimization solution.

# History

- The linear programming method was first developed by Leonid Kantorovich in 1937.
- He developed it during World War II as a way to plan expenditures and returns so as to reduce costs to the army and increase losses incurred by the enemy.
- The method was kept secret until 1947 when George B. Dantzig published the simplex method and John von Neumann developed the theory of duality as a linear optimization solution.
- Dantzig's original example was to find the best assignment of 70 people to 70 jobs subject to constraints.

# History

- The linear programming method was first developed by Leonid Kantorovich in 1937.
- He developed it during World War II as a way to plan expenditures and returns so as to reduce costs to the army and increase losses incurred by the enemy.
- The method was kept secret until 1947 when George B. Dantzig published the simplex method and John von Neumann developed the theory of duality as a linear optimization solution.
- Dantzig's original example was to find the best assignment of 70 people to 70 jobs subject to constraints.
- The computing power required to test all the permutations to select the best assignment is vast.

# History

- The linear programming method was first developed by Leonid Kantorovich in 1937.
- He developed it during World War II as a way to plan expenditures and returns so as to reduce costs to the army and increase losses incurred by the enemy.
- The method was kept secret until 1947 when George B. Dantzig published the simplex method and John von Neumann developed the theory of duality as a linear optimization solution.
- Dantzig's original example was to find the best assignment of 70 people to 70 jobs subject to constraints.
- The computing power required to test all the permutations to select the best assignment is vast.
- However, the theory behind linear programming drastically reduces the number of feasible solutions that must be checked for optimality.

# History

- Linear-programming problem was first shown to be solvable in polynomial time by Leonid Khachiyan in 1979.
- Major breakthrough - Narendra Karmarkar's method for solving LP (1984) using *interior point method*.

# An Example

- A farmer has a piece of farm land of area $\alpha$ square kilometers.
- Wanted to plant wheat and barley.
- The farmer has $\beta$ unit of fertilizer and $\gamma$ unit of pesticides in hand.
- $(f_1, p_1)$: The amount of fertilizer and pesticides needed for wheat per square kilometer.
- $(f_2, p_2)$: The amount of fertilizer and pesticides needed for barley per square kilometer.
- Profit of wheat and barley per square kilometer is $c_1$ and $c_2$

**The objective:**

To decide $x_1, x_2$, the area where wheat and barley needs to be planted.

# The Problem

Objective function:

$c_1 x_1 + c_2 x_2$

Constraints:

$f_1 x_1 + f_2 x_2 \leq \beta$

$p_1 x_1 + p_2 x_2 \leq \gamma$

$x_1 + x_2 \leq \alpha$, and $x_1, x_2 \geq 0$.

# The Problem

Objective function:
$c_1x_1 + c_2x_2$

Constraints:
$f_1x_1 + f_2x_2 \leq \beta$
$p_1x_1 + p_2x_2 \leq \gamma$
$x_1 + x_2 \leq \alpha$, and $x_1, x_2 \geq 0$.

In matrix form:
Maximize

$$\begin{pmatrix} c_1 & c_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

Subject to

$$\begin{pmatrix} 1 & 1 \\ f_1 & f_2 \\ p_1 & p_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix}, \quad \text{and} \quad x_1, x_2 \geq 0.$$

# Linear Integer Programming

## The Problem

Objective Function: Maximize $8x_1 + 11x_2 + 6x_3 + 4x_4$

Subject to:

$$5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14$$
$$7x_1 + 2x_3 + x_4 \leq 10$$
$$x_i \in \mathbb{Z}^+ \text{ for all } i = 1, 2, 3, 4$$

# Linear Integer Programming

## Types of integer programming problems

**Pure Integer Programming Problem:** All variables are required to be integer.

**Mixed Integer Programming Problem:** Some variables are restricted to be integers; the others can take any value.

**Binary Integer Programming Problem:** All variables are restricted to be 0 or 1.

# Linear Integer Programming

## Linear Relaxation

Objective Function: Maximize $z = 8x_1 + 11x_2 + 6x_3 + 4x_4$
Subject to: $5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14$
$x_i \in \{0, 1\}$ for all $i = 1, 2, 3, 4$

Relax it to:

Objective Function: Maximize $z = 8x_1 + 11x_2 + 6x_3 + 4x_4$
Subject to: $5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14$
$x_i \in [0, 1]$ for all $i = 1, 2, 3, 4$

Non-optimality

Integrality relaxation: $x_1 = x_2 = 1$, $x_3 = 0.5$, $x_4 = 0$, and $z = 22$.

Rounding $x_3 = 0$: gives $z = 19$.

Rounding $x_3 = 1$: Infeasible.

Optimal integer solution: $x_1 = 0$, $x_2 = x_3 = x_4 = 1$, and $z = 21$.

# Converting finite valued integer variables to binary

Assume that $x_j$ can assume values in $\{p_1, p_2, \ldots, p_k\}$.

**To convert it to binary integer programming**

Introduce variables $y_j^1, y_j^2, \ldots y_j^k \in \{0, 1\}$, and
Substitute $x_j$ with: $p_1 y_j^1 + p_2 y_j^2 + \ldots + p_k y_j^k$ in objective function and all the constraints, and
Introduce a new constraint: $y_j^1 + y_j^2 + \ldots + y_j^p = 1$

# Converting finite valued integer variables to binary

## Example

Objective Function: $\text{Maximize } z = 8x_1 + 11x_2 + 6x_3 + 4x_4$

Subject to: $5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14$

$x_1 \in \{1, 2, 3\}$ and $x_j \in \{0, 1\}$ for all $i = 2, 3, 4$

## Converted Problem: substituting $x_1 = 1y_1^1 + 2y_1^2 + 3y_1^3$

Objective Function: $\text{Max } z = 8y_1^1 + 16y_1^2 + 24y_1^3 + 11x_2 + 6x_3 + 4x_4$

Subject to: $5y_1^1 + 10y_1^2 + 15y_1^3 + 7x_2 + 4x_3 + 3x_4 \leq 14$

$y_1^1 + y_1^2 + y_1^3 = 1$

$y_1^1, y_1^2, y_1^3 \in \{0, 1\}$ and $x_j \in \{0, 1\}$ for all $i = 2, 3, 4$

Solution: $y_1^1 = 0,\ y_1^2 = 1,\ y_1^3 = 0,\ x_2 = 0,\ x_3 = 1,\ x_4 = 0$, and $z = 22$.

Implying: $x_1 = 2,\ x_2 = 0,\ x_3 = 1,\ x_4 = 0$, and $z = 22$.

# Applications of ILP

**Vertex Cover Problem**

Given a graph $G = (V, E)$, find a subset of vertices $U \subseteq V$ such that each edge in $E$ is incident to at least one vertex in $U$.

ILP Formulation:

Variables: $\{x_1, x_2, \ldots x_n\}$ corresponding to each vertex in $V$.

Objective function: Min $\sum_{i=1}^{n} x_i$.

Constraints: For each member $e_{ij} = (v_i, v_j) \in E$, $x_i + x_j \geq 1$.

$x_i \in \{0, 1\}$ for all $i = 1, 2, \ldots, n$.

# Applications of ILP

## Set Cover Problem

Given a set of elements $U = \{1, 2, \ldots, M\}$, and a set $S$ of subsets $\{S_1, S_2, \ldots, S_k\}$ with the elements of $U$.
Choose the minimum number of subsets such that their union is $U$

Formulation:

Variables: $\{x_1, x_2, \ldots x_k\}$ corresponding to $\{S_1, S_2, \ldots, S_k\}$.

Objective function: Min $\sum_{i=1}^{k} x_i$.

Constraints: For each member $u_j \in U$, $\sum_{u_j \in S_i} x_i \geq 1$.

$x_i \in \{0, 1\}$ for all $i = 1, 2, \ldots, k$.

# Applications of ILP

## Maximum Clique Problem

A clique in a graph $G = (V, E)$ is a subset $C \subseteq V$ such that the subgraph of $G$ induced by those vertices is a complete graph.
Objective is to choose the largest clique in $G$

Formulation:

Variables: $\{x_1, x_2, \ldots x_n\}$ corresponding to every member of $V$.

Objective function: Max $\sum_{i=1}^{n} x_i$.

Constraints: $x_i + x_j \leq 1$ for all $(v_i, v_j) \notin E$ and $i < j$.

$x_i \in \{0, 1\}$ for all $i = 1, 2, \ldots, n$.

# Applications of ILP

## Maximum Independent Set Problem

An independent set in a graph $G = (V, E)$ is a subset $C \subseteq V$ such that there is no edge among any pair of vertices in $C$.
Objective is to choose the largest independent set in $G$

Formulation:

Variables: $\{x_1, x_2, \dots x_n\}$ corresponding to every member of $V$.
Objective function: Max $\sum_{i=1}^{n} x_i$.
Constraints: $x_i + x_j \leq 1$ for all $(v_i, v_j) \in E$ and $i < j$.
$x_i \in \{0, 1\}$ for all $i = 1, 2, \dots, n$.

# Computational Status of ILP

## Computational Status of ILP - NP-Hard

**Proof:**

### ILP $\in$ NP

Consider a 0-1 ILP, where each variable $x_1, x_2, \ldots, x_n$ can assume values 0 or 1. The number of constraints is $m$.

Example:

Objective Function: Maximize $x_1 - x_2 + x_3 + x_4$

Subject to: $x_1 + x_2 - x_3 + x_4 \geq 0$

$x_1 + x_3 - x_4 \geq 0$

$x_i \in \mathbb{Z}^+$ for all $i = 1, 2, 3, 4$

We can choose all possible $2^n$ assignments of $x_1, x_2, \ldots, x_n$ in non-deterministic manner.

Checking the feasibility of each assignment takes $O(nm)$ time, and Computing the value of the objective function for each feasible assignment takes $O(n)$ time.

# Computational Status of ILP

## ILP is NP-hard (Reduction from 3-SAT)

Given a 3-SAT expression with variables $x_1, x_2, \ldots, x_n$

Example: $(x_1 \vee \overline{x}_2 \vee \overline{x}_3) \wedge (x_2 \vee \overline{x}_4 \vee x_5)$

Create an ILP instance with variables $z_1, z_2, \ldots, z_n$, where

Each clause is converted as an inequation as follows:

$z_1 + (1 - z_2) + (1 - z_3) > 0$

$z_2 + (1 - z_4) + z_5 > 0$

$z_i \in \{0, 1\}$ for all $i = 1, 2, \ldots, 5$

Now, a feasible solution of this ILP indicates a satisfying truth assignent of the 3-SAT.

**Conclusion:** ILP is NP-hard since 3-SAT is NP-complete.

# An important property of ILP

## Important Result

- An $m \times n$ matrix is said to be unimodular if for every submatrix of size $m \times m$ the value of its determinant is 1, 0, or $-1$.
- A matrix is said to be *totally unimodular* (TUM) if its all possible square submatrices are unimodular.
- If the coefficient matrix of an integer linear program is a TUM, then it is polynomially solvable.

# Polynomial Solvability of Unimodular ILP

## Result

If the constraint matrix $A$ in LP is totally unimodular and $b$ is integer valued, then every vertex of the feasible region is integer valued.

## Proof

Let $A_{m \times n}$ be unimodular, and $m < n$.
Let $B_{m \times m}$ be a basis, and $x_B = B^{-1}b$ be a basic solution.
By Cramer's rule, $B^{-1} = \frac{C^T}{det(B)}$, where $C = ((c_{ij}))$ is the cofactor matrix of $B$.

$$c_{ij} = (-1)^{i+j} det \begin{pmatrix} b_{1,1} & \dots & b_{1,j-1} & X & b_{1,j+1} & \dots & b_{1,m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ b_{i-1,1} & \dots & b_{i-1,j-1} & X & b_{i-1,j+1} & \dots & b_{i-1,m} \\ X & X & X & X & X & X & X \\ b_{i+1,1} & \dots & b_{i+1,j-1} & X & b_{i+1,j+1} & \dots & b_{i+1,m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ b_{m,1} & \dots & b_{m,j-1} & X & b_{m,j+1} & \dots & b_{m,m} \end{pmatrix}$$

Thus,

- Each element of the matrix $B^{-1}$ is integer.
- Now, if the elements of the requirement vector $b$ are integer, the basic feasible solution corresponding to the basis $B$ is integer valued.
- The same is true for all possible basis of the matrix $A$.

Conclusion: Thus, if the coefficient matrix of an ILP is a TUM, then it can be solved in polynomial time.

# Total unimodularity and bipartite graphs

**Result**

A graph is bipartite if and only if its incidence matrix is totally unimodular.

**Proof:**

If part: Let $A$ be a TUM. Assume that $G$ is not bipartite. Then $G$ contains an odd cycle. The submatrix of $A$ which corresponds to the odd cycle has determinant 2. This contradicts the total unimodularity of the matrix $A$.

Only if part: Let $G$ be bipartite. Consider a $t \times t$ submatrix of $A$. Proof is by induction on $t$. ( $t = 1$ follows from the definition of the incidence matrix.)

# Konig Matching Theorem

## Konig Matching Theorem

Let $G$ be a bipartite graph. Cardinality of maximum matching = cardinality of minimum vertex cover $(\nu(G) = \tau(G))$.

**Proof:** We know that the matching number is given by
$\nu(G) = \max\{1^T x | Ax \leq 1, x \geq 0\}$, where
$A \longrightarrow$ vertex edge incidence matrix, and
$x$ corresponds to the edges in $G$.

Its dual problem: $d^* = \min\{1^T y | y \geq 0, A^T y \geq 1\}$,
where $y$ corresponds to the vertices in $G$.

By LP duality, $d^* = \nu(G)$.

- Let $y$ be the incidence vector of the minimum vertex cover. Then $y$ is feasible in the dual problem with value of the objective function $\tau(G)$.
- Suppose $y$ is a non-optimal solution of the dual problem.
- Then there is a (0-1) optimum solution $y^*$ with $d^* < \tau(G)$.
- But, this (0-1) solution $y^*$ is also a feasible solution of the vertex cover problem.
- Indication $y$ is not optimum for vertex cover problem $\Longrightarrow$ Contradiction.

**Conclusion:**

Vertex cover problem is polynomially solvable for bipartite graph.

# Incidence matrix of a directed graph

Let $G = (V, E)$ be a directed graph

Define a $|V| \times |E|$ incidence matrix $A = ((a_{i,j}))$ as follows:

$$\begin{aligned} a_{v,e} &= 1 \text{ if } e \text{ leaves } v \\ &= -1 \text{ if } e \text{ enters } v \\ &= 0 \text{ otherwise} \end{aligned} \tag{1}$$

**Note:** Every column of $A$ has exactly one 1 and one $-1$, the other elements are all zeroes.

# Total unimodularity and directed graphs

**Result**

The incidence matrix $A$ of a directed graph $G$ is totally unimodular.

**Proof:** Let $B$ be a $t \times t$ square submatrix of $A$. Proof by induction on $t$.

Case 0: $t = 1$. This case is trivial.

Case 1: $B$ has a zero column $\Longrightarrow det(B) = 0$.

Case 2: $B$ has a column with exactly one 1 (or -1). Calculate $det(B)$ using this column and use the induction assumption.
$det(B) = 1$ or $-1$ or $0$

Case 3: Every column of $B$ has one 1 and one 1. The row vectors of $B$ add up to the zero vector $\Longrightarrow det(B) = 0$.

# Transportation Problem

## The Problem

Given

- a set of $m$ sources $s_1, s_2, \ldots, s_m$, and a set of destinations $d_1, d_2, \ldots, d_n$,
- the costs $c_{ij}$ of transporting from source $s_i$ to destination $d_j$,
- the supply $a_i$ at source $s_i$ for all $i = 1, 2, \ldots, m$,
- the requirement $b_j$ at destination $d_j$ for all $j = 1, 2, \ldots n$

The objective is to solve the LP

Minimize $\quad z = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} c_{ij}$

Subject to
$$\begin{aligned} &\textstyle\sum_{j=1}^{n} x_{ij} \le a_i \quad \forall \quad i = 1, 2, \ldots m \\ &\textstyle\sum_{i=1}^{m} x_{ij} \ge b_j \quad \forall \quad j = 1, 2, \ldots n \\ &x_{ij} \ge 0 \quad \forall \quad i = 1(1)m, \quad j = 1(1)n \end{aligned}$$

# Transportation Problem

## Balanced Transportation

Here, $\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$, and the constraints are eqalities.

## Example of a balanced transportation problem

Consider the following problem with 2 factories and 3 warehouses:

| | wirehouse 1 | wirehouse 2 | wirehouse 3 | supply |
|---|---|---|---|---|
| Factory 1 | $c_{11} = 2$ | $c_{12} = 5$ | $c_{13} = 7$ | 20 |
| Factory 2 | $c_{21} = 6$ | $c_{22} = 4$ | $c_{23} = 4$ | 10 |
| Demand | 7 | 10 | 13 | |

This is an LP, and can be solved in polynomial time.

# Assignment Problem

## Assignment Problem

- Here, the cost of doing a job $j$ by a machine $i$ is $c_{ij} \geq 0$.
  If $c_{ij} = \infty$, then machine $i$ can not be used for job job $j$.
- $m = n$
- $a_i = 1$ for all $i = 1, 2, \ldots, m$, and
- $b_j = 1$ for all $j = 1, 2, \ldots, n$.
- $x_{ij} = 0$ or $1$

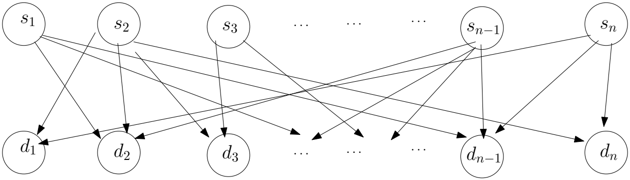
This is an ILP, where the constraints can be written as

$$
\begin{aligned}
\textstyle\sum_{j=1}^{n} x_{ij} &= 1 \quad \forall \quad i = 1, 2, \ldots n \\
-\textstyle\sum_{i=1}^{m} x_{ij} &= -1 \quad \forall \quad j = 1, 2, \ldots n \\
x_{ij} &= 0 \text{ or } 1 \quad \forall \quad i = 1(1)n, \quad j = 1(1)n
\end{aligned}
$$

and the objective function remains same as the transportation problem

Minimize $\qquad z = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} c_{ij}$

# Min-Cost Bipartite Matching




Since, each column of the coefficient matrix has an 1 and a -1, the matrix is unimodular.

Thus, the assignment problem can be solved in polynomial time.

# Problems on Interval Graph

An interval matrix has 0 1 entries and each row is of the form

$$(0, \ldots, 0, 1, \ldots\ldots, 1, 0, \ldots, 0)$$

**Result**

Each interval matrix A is totally unimodular.

**Proof:** Let $B$ be a $t \times t$ submatrix of $A$, and let.

$$N = \begin{pmatrix} 1 & -1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & -1 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \ldots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 1 & -1 \\ 0 & 0 & 0 & \ldots & 0 & 1 \end{pmatrix}$$

Note that, $det(N) = 1$. Now, $NB^T$ is a submatrix of the incidence matrix of some directed graph. Thus, $NB^T$ is totally unimodular, and hence $det(B) \in \{0, +1, -1\}$.

# Clique of Interval Graph

## Maximum Clique Problem

A clique in a graph $G = (V, E)$ is a subset $C \subseteq V$ such that the subgraph of $G$ induced by those vertices is a complete graph.
Objective is to choose the largest clique in $G$

Formulation:

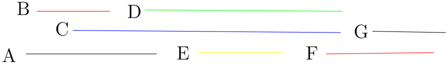
Variables: $\{x_1, x_2, \ldots x_n\}$ corresponding to every member of $V$.

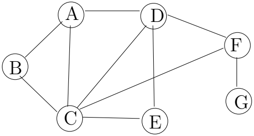
Objective function: Max $\sum_{i=1}^{n} x_i$.

Constraints: $x_i + x_j \leq 1$ for all $(v_i, v_j) \notin E$ and $i < j$.

$x_i \in \{0, 1\}$ for all $i = 1, 2, \ldots, n$.

# Clique of Interval Graph



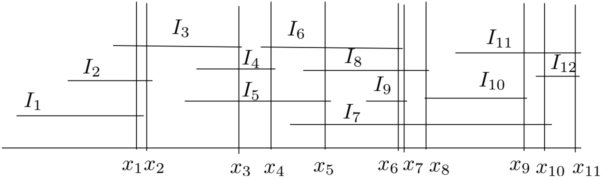

The corresponding matrix

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ & 1 & 0 & 0 & 0 & 0 \\ & & 1 & 1 & 1 & 0 \\ & & & 1 & 1 & 0 \\ & & & & 0 & 0 \\ & & & & & 1 \end{bmatrix}$$

**Observation**

As the consecutive 1 property is satisfied by the coefficient matrix, the clique problem for interval graph can be solved in polynomial time.

# Clique Cover Problem for Interval Graph




The corresponding matrix

| | $I_1$ | $I_2$ | $I_3$ | $I_4$ | $I_5$ | $I_6$ | $I_7$ | $I_8$ | $I_9$ | $I_{10}$ | $I11$ | $I_{12}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $x_1$ | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $x_2$ | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $x_3$ | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $x_4$ | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $x_5$ | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| $x_6$ | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| $x_7$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| $x_8$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| $x_9$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| $x_{10}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| $x_{11}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

# Clique Cover Problem for Interval Graph

**Minimum Clique Cover**

Minimize $\sum_{i=1}^{n} x_i$ subject to $\sum_{i \in I_i} x_i \geq 1$ for all intervals $I_i$.

**Time complexity**

Thus, the minimum clique cover corresponds to minimum number of vertical line that stab all the segments.

As the coefficient matrix satisfies consecutive 1 property, and can be solved in polynomial time.

# Solving ILP - Cutting plane method

**ILP Problem**

Minimize $c'x$

Subject to $Ax = b$

$x \geq 0$, $x$ integer.

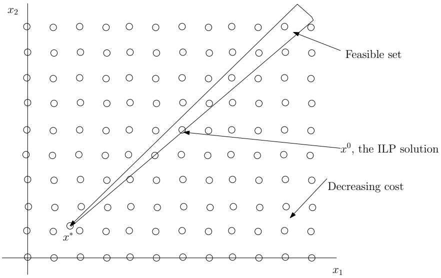
**Corresponding LP Problem**

Minimize $c'x$

Subject to $Ax = b$

$x \geq 0$.

- $x^*$ is a basic feasible solution of the LP.
- The greedy strategy is that choose the nearest integer of $x^*$ as the solution of ILP.
- But, such a solution may not be feasible.



# Solving ILP - Cutting Plane method

**Theme of the Algorithm**

Step 1: Solve the LP relaxation of the ILP problem.

Step 2: If the solution is not an integer then add a new constraint, *called a cutting plane*, that does not exclude any feasible integer solution of the LP problem.

Step 3: Go to Step 1

# Getting a Cutting Plane - Gomory's technique

Recall the simplex tableau

| | | | $c_1$ | $c_2$ | $\dots$ | $\dots$ | $c_j$ | $\dots$ | $\dots$ | $c_n$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $x_B$ | $c_B$ | $y_0$ | $y_1$ | $y_2$ | $\dots$ | $\dots$ | $y_j$ | $\dots$ | $\dots$ | $y_n$ |
| $x_{B_1}$ | $c_{B_1}$ | $y_{10}$ | $y_{11}$ | $y_{12}$ | $\dots$ | $\dots$ | $y_{1j}$ | $\dots$ | $\dots$ | $y_{1n}$ |
| $x_{B_2}$ | $c_{B_2}$ | $y_{20}$ | $y_{21}$ | $y_{22}$ | $\dots$ | $\dots$ | $y_{2j}$ | $\dots$ | $\dots$ | $y_{2n}$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $x_{B_i}$ | $c_{B_i}$ | $y_{i0}$ | $y_{i1}$ | $y_{i2}$ | $\dots$ | $\dots$ | $y_{ij}$ | $\dots$ | $\dots$ | $y_{in}$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $x_{B_m}$ | $c_{B_m}$ | $y_{m0}$ | $y_{m1}$ | $y_{m2}$ | $\dots$ | $\dots$ | $y_{mj}$ | $\dots$ | $\dots$ | $y_{mn}$ |

Consider a typical equation (for some $i \in [0, 1, \dots, m]$) of the tableau

$$x_{B(i)} + \sum_{j \notin B} y_{ij} x_j = y_{i0} \dots\dots\dots (*)$$

# Getting a Cutting Plane - Gomory's technique

Consider a typical equation (for some $i \in [0, 1, \ldots, m]$) of the tableau

$$x_{B(i)} + \sum_{j \notin B} y_{ij} x_j = y_{i0} \ldots\ldots\ldots (*)$$

We also have

$$\sum_{j \in B} \lfloor y_{ij} \rfloor x_j \leq \sum_{j \in B} y_{ij} x_j$$

Thus, we have

$$x_{B(i)} + \sum_{j \notin B} \lfloor y_{ij} \rfloor x_j \leq y_{i0}$$

Since the LHS is an integer, we can replace the constraint as

$$x_{B(i)} + \sum_{j \notin B} \lfloor y_{ij} \rfloor x_j \leq \lfloor y_{i0} \rfloor \ldots\ldots\ldots (*)$$

Subtracting (**) from (*), we have

# Getting a Cutting Plane - Gomory's technique

We have,

$$\sum_{j \notin B} (y_{ij} - \lfloor y_{ij} \rfloor) x_j \geq y_{i0} - \lfloor y_{i0} \rfloor$$

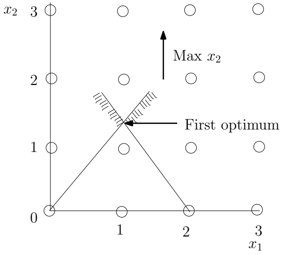
Taking $f_{ij} = y_{ij} - \lfloor y_{ij} \rfloor$ for all $i = 0, 1, \ldots, m$, we have

$$\sum_{j \notin B} f_{ij} x_j \geq f_{i0}$$

Adding slack variable

$$-\sum_{j \notin B} f_{ij} x_j + s = -f_{i0}$$

# Example




## ILP Problem

$$\begin{array}{ll} \text{Maximize} & x_2 \\ \text{Subject to} & 3x_1 + 2x_2 \leq 6 \\ & -3x_1 + 2x_2 \leq 0 \end{array}$$

$x_1, x_2 \geq 0$, $x$ integer.

## Initial Tableau

| $x_b$ | $c_b$ | $y_0$ | $y_1$ | $y_2$ | $y_3$ | $y_4$ |
|---|---|---|---|---|---|---|
| $c_j$ | - | - | 0 | 1 | 0 | 0 |
| $x_3$ | 0 | 6 | 3 | 2 | 1 | 0 |
| $x_4$ | 0 | 0 | -3 | 2 | 0 | 1 |
| $z_j - c_j$ | - | 0 | 0 | -1 | 0 | 0 |

## Final Tableau

| $x_b$ | $c_b$ | $y_0$ | $y_1$ | $y_2$ | $y_3$ | $y_4$ |
|---|---|---|---|---|---|---|
| $c_j$ | - | - | 0 | 1 | 0 | 0 |
| $x_1$ | 0 | 1 | 1 | 0 | $\frac{1}{6}$ | $-\frac{1}{6}$ |
| $x_2$ | 1 | $\frac{3}{2}$ | 0 | 1 | $\frac{1}{4}$ | $\frac{1}{4}$ |
| $z_j - c_j$ | - | $\frac{3}{2}$ | 0 | 0 | $\frac{1}{4}$ | $\frac{1}{4}$ |

- Solution of the LP: $x = (1, \frac{3}{2})$
- Cost: $x_2 = \frac{3}{2}$

# Example

The second row of the tableau says that

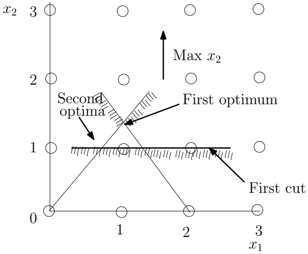
$$x_2 + \frac{1}{4}x_3 + \frac{1}{4}x_4 = \frac{3}{2}$$

Substituting the maximum integeral solution (i.e., $x_2 = 1$), we have

$$\frac{1}{4}x_3 + \frac{1}{4}x_4 \geq \frac{1}{2}$$

Implies a new constraint

. . . . . . . . . . $x_2 \leq 1$.



# Another Iteration

**Initial Tableau**

| $x_b$ | $c_b$ | $y_0$ | $y_1$ | $y_2$ | $y_3$ | $y_4$ |
|---|---|---|---|---|---|---|
| $c_j$ | - | - | 0 | 1 | 0 | 0 |
| $x_1$ | 0 | 1 | 1 | 0 | $\frac{1}{6}$ | $-\frac{1}{6}$ |
| $x_2$ | 1 | $\frac{3}{2}$ | 0 | 1 | $\frac{1}{4}$ | $\frac{1}{4}$ |
| $x_{s1}$ | 0 | $-\frac{1}{2}$ | 0 | 0 | $-\frac{1}{4}$ | $-\frac{1}{4}$ |
| $z_j - c_j$ | - | $\frac{3}{2}$ | 0 | 0 | $\frac{1}{4}$ | $\frac{1}{4}$ |

**Final Tableau**

| $x_b$ | $c_b$ | $y_0$ | $y_1$ | $y_2$ | $y_3$ | $y_4$ | $s_1$ |
|---|---|---|---|---|---|---|---|
| $c_j$ | - | - | 0 | 1 | 0 | 0 | 0 |
| $x_1$ | 0 | $\frac{2}{3}$ | 1 | 0 | 0 | $-\frac{1}{3}$ | $\frac{2}{3}$ |
| $x_2$ | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| $x_3$ | 0 | 2 | 0 | 0 | 1 | 1 | -4 |
| $z_j - c_j$ | - | 1 | 0 | 0 | 0 | 0 | 1 |

Obtained solution $x = (\frac{2}{3}, 1) \longrightarrow$ still not an integer solution.

The first row of the tableau says that $x_1 + (\frac{2}{3} - 1)x_4 + \frac{2}{3}x_{s1} = \frac{2}{3}$

Implying $x_1 - x_4 \leq \frac{2}{3}$

Since $x_1, x_4$ are both integers, we have $x_1 - x_4 \leq 0$

In other words, $\frac{2}{3}x_4 + \frac{2}{3}x_{s1} \geq \frac{2}{3}$

Combining Eq. 2, we have another constraint $x_1 \geq x_2$

# Explanation of last step

We have the followings:

1. $\frac{2}{3}x_4 + \frac{2}{3}x_{s1} \geq \frac{2}{3} \implies x_4 + x_{s1} \geq 1.$
2. $\frac{1}{4}x_3 + \frac{1}{4}x_4 \geq \frac{1}{2} \implies \frac{1}{4}x_3 + \frac{1}{4}x_4 + x_{s1} = \frac{1}{2}$
3. $3x_1 + 2x_2 + x_3 = 6$
4. $-3x_1 + 2x_2 + x_4 = 0$

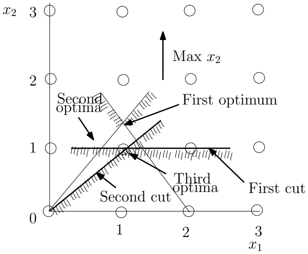
Substituting $x_{s1}$ from (2) in (1), we have $x_4 + (\frac{1}{2} + \frac{1}{4}x_3 + \frac{1}{4}x_4) \geq 1$

Implying, $5x_4 + x_3 \geq 6$

Putting $x_3$ and $x_4$ from (3) and (4) respectively, we have $x_1 \geq x_2$.

# Example

Combining Eq. 2, we have another constraint $x_1 \geq x_2$




Finally, we have the solution: $x_1 = 1$ and $x_2 = 1$.

# Questions !!!

**Questions !!!**

- The number of iterations is finite, but may be exponential in the number of variables and constraints.

# An information

**Result $\longrightarrow$ Eisenbrand and Rote, 2001**

A two variable integer programming defined by $m$ linear constraints where each coefficient is represented using $s$ bits can be solved in $O(m + s \log m)$ arithmetic , or $O(m + \log s \log m)M(s)$ bit operations, where $M(s)$ is the time needed for multiplying two integers of $s$ bits.