# Topic 5: Vertex Cover Problem: Use of LP

Minimize $w(S) = \sum_{i \in S} w_i$

where $S$ ($S \subset V$) is a vertex cover (i.e., each edge in $E$ has at least one end in $S$).

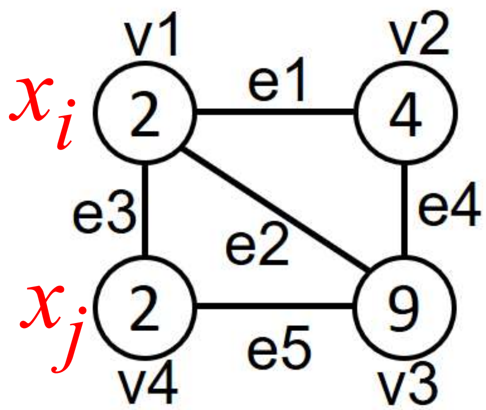
**Decision variable for each vertex $v_i$:** $x_i = \begin{cases} 0 & \text{if } i \notin S \\ 1 & \text{if } i \in S \end{cases}$

## VC-IP: Vertex Cover as an Integer Programming Problem

Minimize $w_{\mathrm{IP}}(\boldsymbol{x}) = \sum_{i \in V} w_i x_i$

subject to $x_i + x_j \geq 1$ for $(i, j) \in E$

$x_i \in \{0, 1\}$ for $i \in V$



# Matrix Form of VC-IP

Minimize $w_{\mathrm{IP}}(\boldsymbol{x}) = \boldsymbol{w}^t\ \boldsymbol{x}$

subject to $\mathbf{1} \geq \boldsymbol{x} \geq \mathbf{0}$, $A\boldsymbol{x} \geq \mathbf{1}$, and $\boldsymbol{x}$ is an integer vector.

$$\boldsymbol{x} = (x_1,\ x_2,\ \dots, x_{|V|})^t \qquad \mathbf{1} = (1,\ 1,\ \dots,\ 1)^t$$

$$\boldsymbol{w} = (w_1,\ w_2,\ \dots, w_{|V|})^t \qquad \mathbf{0} = (0,\ 0,\ \dots,\ 0)^t$$

Matrix $A$: Rows of $A$ correspond to edges in $E$

Columns of $A$ correspond to vertexes in $V$

$$A[e,\ i] = \begin{cases} 1 & \text{if vertex } v_i \text{ is an end of edge } e \\ 0 & \text{otherwise} \end{cases}$$

If $\boldsymbol{x}^*$ is the optimal solution of VC-IP, $S = \{v_i \in \mathrm{V}: x_i^* = 1\}$ is the optimal vertex cover $S^*$ with the minimum total weight $w(S^*)$.

## Matrix Form of VC-IP

Minimize $w_{\mathrm{IP}}(\boldsymbol{x}) = \boldsymbol{w}^t\ \boldsymbol{x}$

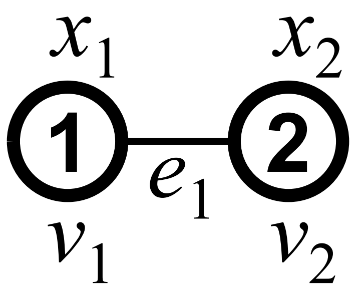
subject to $\mathbf{1} \geq \boldsymbol{x} \geq \mathbf{0}$, $A\boldsymbol{x} \geq \mathbf{1}$, and $\boldsymbol{x}$ is an integer vector.

$$\boldsymbol{x} = (x_1,\ x_2,\ \ldots, x_{|V|})^t \qquad \mathbf{1} = (1,\ 1,\ \ldots,\ 1)^t$$

$$\boldsymbol{w} = (w_1,\ w_2,\ \ldots, w_{|V|})^t \qquad \mathbf{0} = (0,\ 0,\ \ldots,\ 0)^t$$

Matrix $A$: Rows of $A$ correspond to edges in $E$
Columns of $A$ correspond to vertexes in $V$

$$A[e,\ i] = \begin{cases} 1 & \text{if vertex } v_i \text{ is an end of edge } e \\ 0 & \text{otherwise} \end{cases}$$




Minimize

$$(1\ \ 2)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

subject to

$$\begin{pmatrix} 1 \\ 1 \end{pmatrix} \geq \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \end{pmatrix},\quad (1\ \ 1)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \geq 1$$

## VC-LP: Linear Programming Relaxation of VC-IP

Minimize $w_{\mathrm{LP}}(\boldsymbol{x}) = \boldsymbol{w}^t\ \boldsymbol{x}$

subject to $\mathbf{1} \geq \boldsymbol{x} \geq \mathbf{0}$, $A\boldsymbol{x} \geq \mathbf{1}$ ~~, and $\boldsymbol{x}$ is an integer vector~~.

## VC-LP: Linear Programming Relaxation of VC-IP

Minimize $w_{\text{LP}}(\boldsymbol{x}) = \boldsymbol{w}^t \boldsymbol{x}$

subject to $\mathbf{1} \geq \boldsymbol{x} \geq \mathbf{0}, \quad A\boldsymbol{x} \geq \mathbf{1}$.

Different optimal solutions

**Optimal value of VC-LP ≤ Optimal value of VC-IP**

$$w_{\text{LP}}(\boldsymbol{x}^*_{\text{LP}}) \leq w_{\text{IP}}(\boldsymbol{x}^*_{\text{IP}})$$

This can be an invalid solution as a vertex cover (e.g., $x_1 = 0.5$, $x_2 = 0.5$, $x_3 = 1$).

## LP: Linear Programming

Most frequently used optimization method
(a number of software packages are available)

The following relation holds among the LP solution $\boldsymbol{x}^*_{\text{LP}}$, the optimal solution $S^*$, and any greedy solution $S$:

$$w_{\text{LP}}(\boldsymbol{x}^*_{\text{LP}}) \leq w_{\text{IP}}(\boldsymbol{x}^*_{\text{IP}}) = w(S^*) \leq w(S)$$

## LP Problem Example

A chemical company is producing two products A and B (e.g., special liquid). The profit from product A is \$2000/ton, and the profit from product B is \$3000/ton. To generate 1 kg of A, they need 1 kg of material X and 3 kg of material Y. To generate 1 kg of B, they need 6 kg of material X and 1 kg of material Y. These two materials are limited. 28.8 ton of material X and 9.9 ton of material Y are available every day. The problem is to determine the amount of each product to be produced every day in order to maximize the total profit.

## IP Problem Example.

Assume the profit per chair is \$20, whereas the profit per table is \$30. To build a chair, a single unit of wood is required and three man-hours of labor. To build a table, six units of wood are required and one man-hour of labor. The production process has some restrictions: all the machines can only process 288 units of wood per day and there are only 99 man-hours of available labor each day. The question is, how many chairs and tables should the company build to maximize its profit?

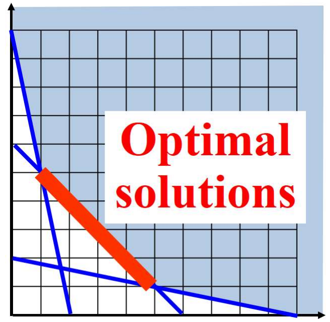
# Exercise 10-1 (Use of LP):

(1) Find at least two LP (Linear Programming) software packages.

(2) Generate a simple LP problem example with a single optimal solution.

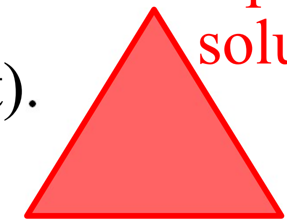
(3) Solve the generated example using your LP software packages. This is to confirm that you are correctly using the packages (i.e., to check whether the optimal solution is obtained by each of your packages).

(4) Generate a simple LP problem example with an optimal solution set on a line (i.e., a one-dimensional solution set).




(5) Solve the generated example using your LP software packages. This is to examine which solution is obtained by each of your packages.

(6) Generate a simple LP problem example with an optimal solution set on a plane (i.e., a two-dimensional solution set).




(7) Solve the generated example using your LP software packages. This is to examine which solution is obtained by each of your packages.

Your presentation will be mainly about your LP software packages, your LP problem examples, and your experimental results. Use LP packages. **Do not use IP (Integer Programming) packages.**

**Exercise 10-2 (Examine the performance of LP)**:

Our TA will give you three LP problem instances (small-scale, medium-scale, and large-scale instances). Solve the given problem instances using your LP packages. Then compare your packages using the following criteria:

(1) Scalability (i.e., whether your LP packages are applicable to all the given problem instances).

(2) Computation time of each package on each problem instance.

# Hints:

1. If you find different LP packages based on different LP algorithms in the same platform (with the same computer language), you will be able to compare them from the following viewpoints:

- the obtained solution by each algorithm, which can be different,
- the computation time of each algorithm, which can be totally different,
- the applicability (scalability) of each algorithm to large-scale LP problems, which may be similar.

You will be able to do the same if you find an LP package with different LP algorithms.

2. If you find different LP packages based on the same LP algorithm in different platforms (with different computer languages), you will be able to compare them from the following viewpoints:

- the obtained solution by each package, which should be the same,
- the computation time of each package, which can be totally different,
- the applicability (scalability) of each package to large-scale LP problems, which can be totally different.