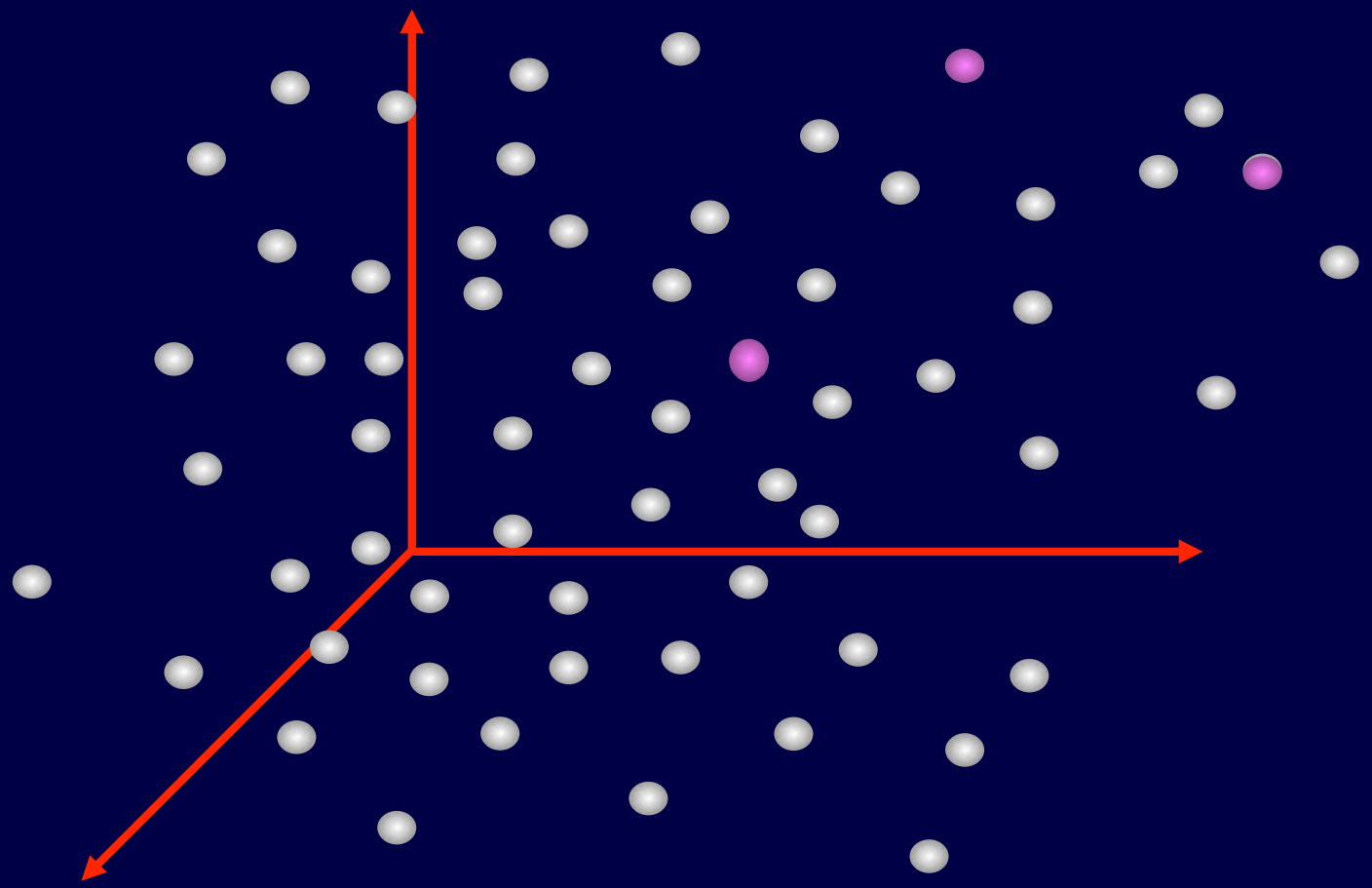
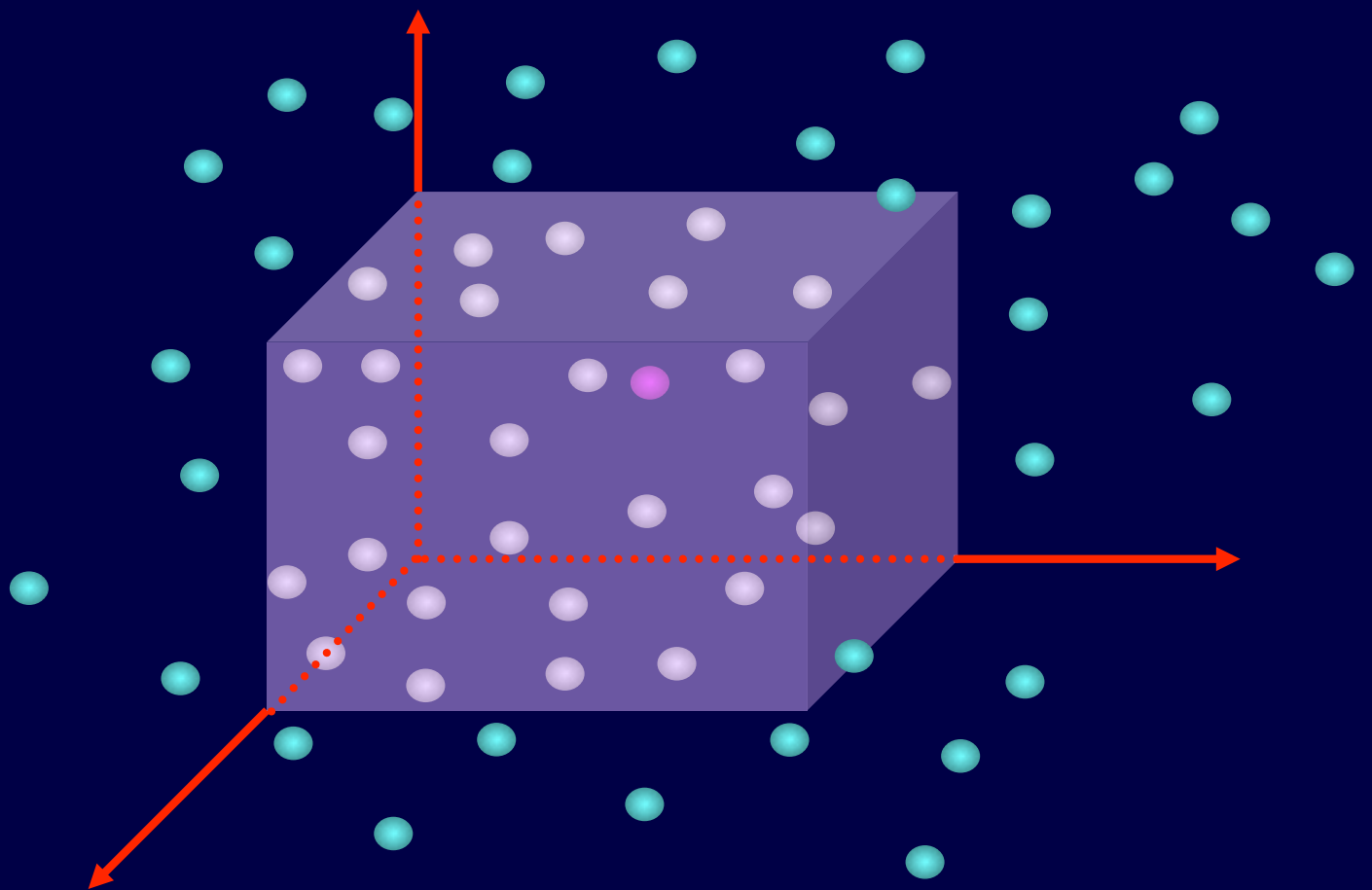
Optimality and Efficiency—
A Dual Objective Function for Optimization Algorithms

C. L. Liu
National Tsing Hua University
Hsinchu, Taiwan

Optimization Problems

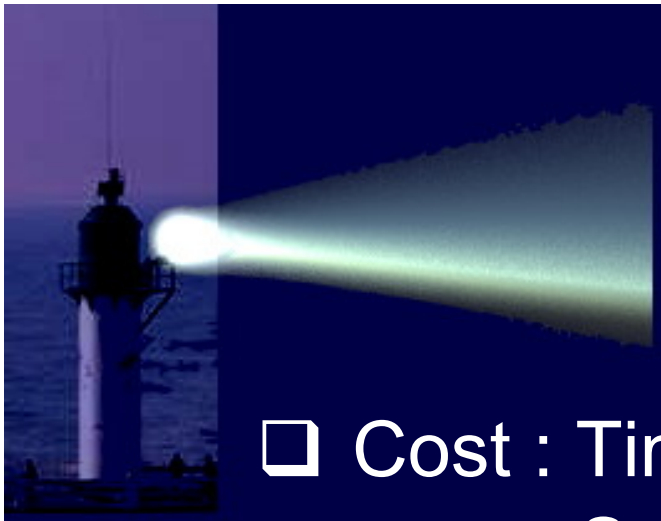


Constrained Optimization Problems



Solution of Optimization Problems

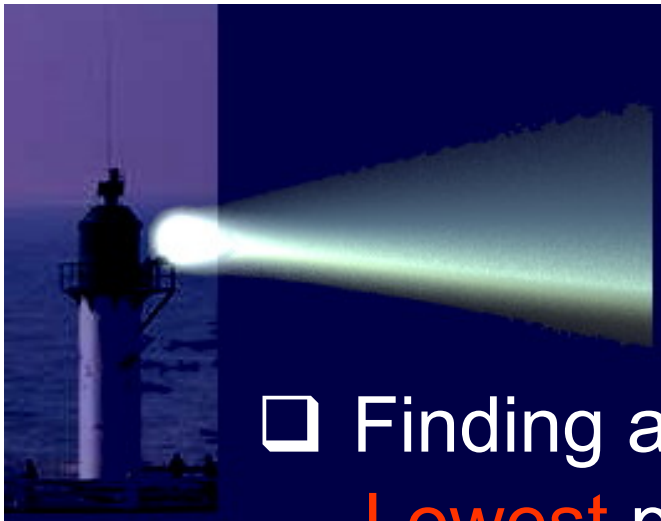
- Finding a Best possible solution at the Lowest possible cost



Solution of Optimization Problems

- Cost : Time
  Space
  Energy
  ⋮

Time is money



Solution of Optimization Problems

- Finding a Best possible solution at a Lowest possible cost

Examples :

1. Finding the largest of n numbers using $n - 1$ pairwise comparisons

2. Sorting n numbers in order using the Merge Sort Algorithm

Solution of Optimization Problems

- Finding a Best possible solution at a Low cost
  (Low : Polynomial in problem size)

  Examples :

  1. Sorting n numbers using the Bubble Sort Algorithm
  2. Solving the Linear Programming Problem using the Ellipsoid Algorithm or the Interior Point Algorithm
  3. Solving the Max Flow Problem using the Augmenting Path Algorithm

Solution of Optimization Problems

- Finding a Best possible solution at a High cost
  (High : Exponential in problem size)

Examples :

1. Solving the Linear Programming Problem using the Simplex Algorithm

2. Solving the Scheduling Problem using a Branch-and-Bound Algorithm

Solution of Optimization Problems

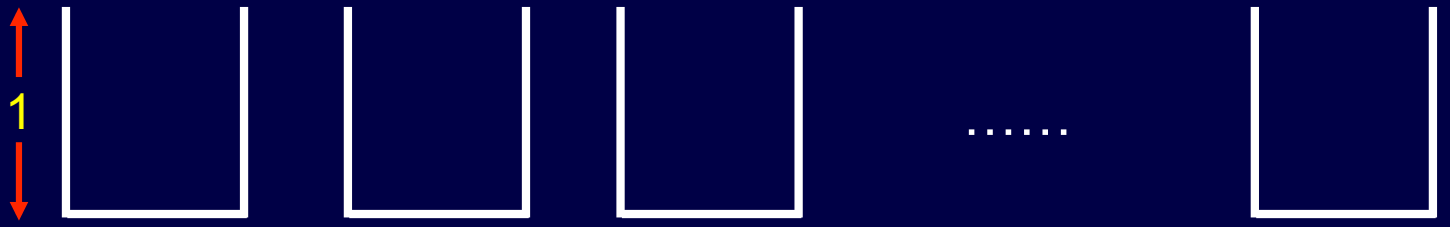
- Finding an Acceptable solution at a Low cost

Examples :

1. Solving the Bin Packing Problem using the Next Fit Algorithm
2. Solving the Traveling Salesman Problem using the Nearest Neighbor First Algorithm
3. Solving the Scheduling Problem using the Demand Scheduling Algorithm

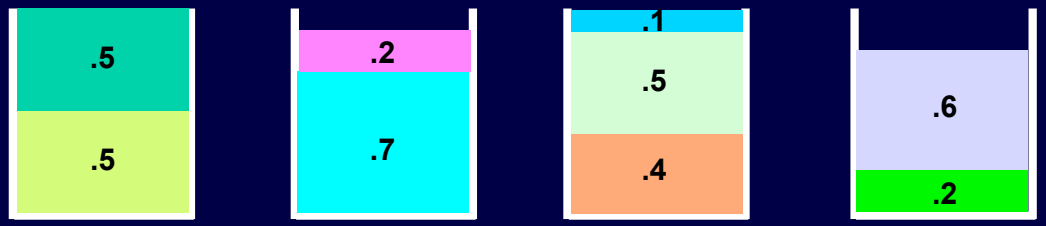
Approximation Algorithms
（Performance Guaranteed）

Bin Packing Problem



.5 .7 .5 .2 .4 .2 .5 .1 .6

Optimal Packing

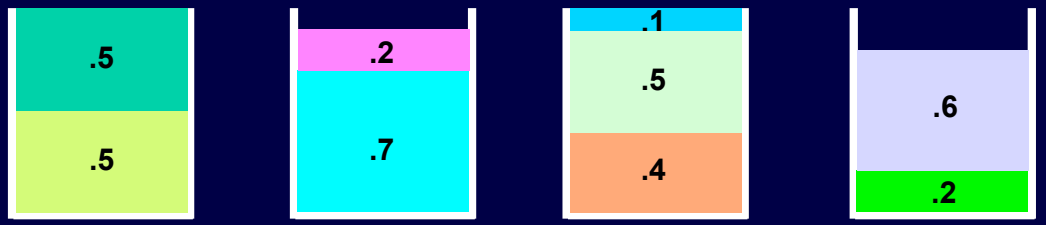


$N_0 = 4$

Approximation Algorithms
（Performance Guaranteed）

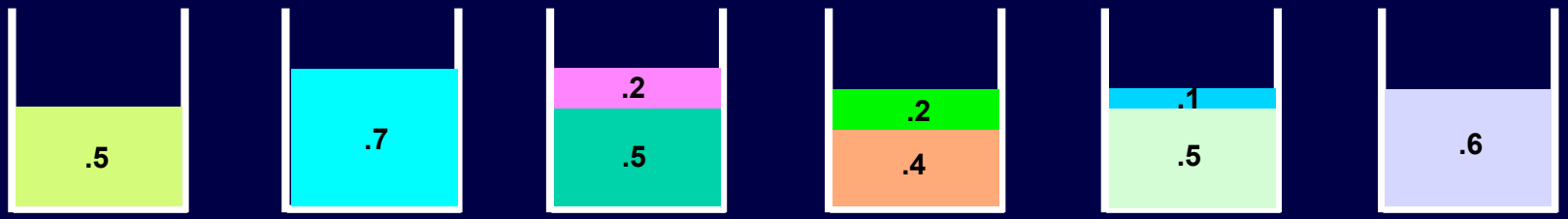
Bin Packing Problem

.5 .7 .5 .2 .4 .2 .5 .1 .6



$N_0 = 4$

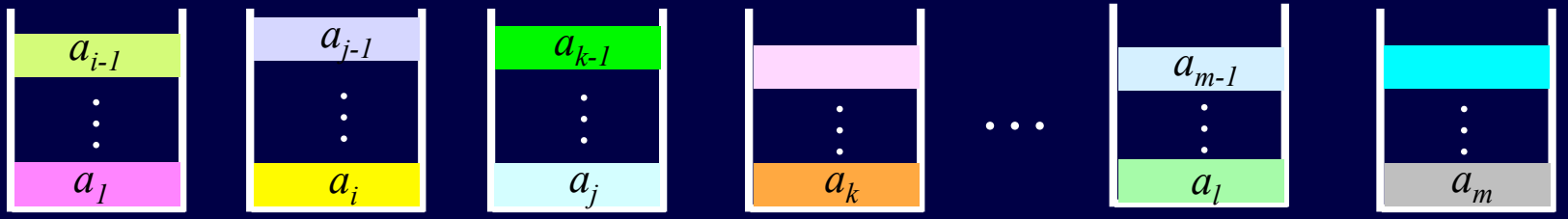
Next Fit Packing Algorithm



$N = 6$

$$\frac{N}{N_0} \leq 2$$

Next Fit Packing Algorithm



$a_1 + \ldots\ldots + a_i > 1$

$a_i + \ldots\ldots + a_j > 1$

$a_j + \ldots\ldots + a_k > 1$

$\vdots$

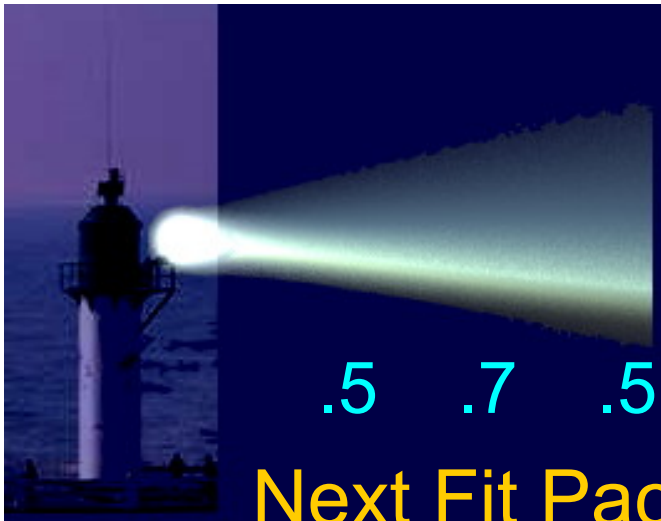
$a_l + \ldots\ldots + a_m > 1$

Let $a_1 + a_2 + \ldots\ldots = \Sigma$

$2\Sigma \geq N - 1$

$N_0 \geq \Sigma \geq \frac{N-1}{2} \geq \frac{N}{2}$

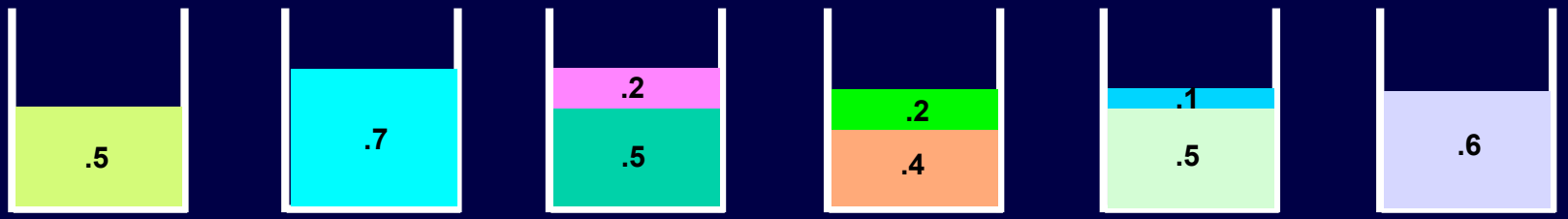
$\frac{N}{N_0} \leq 2$



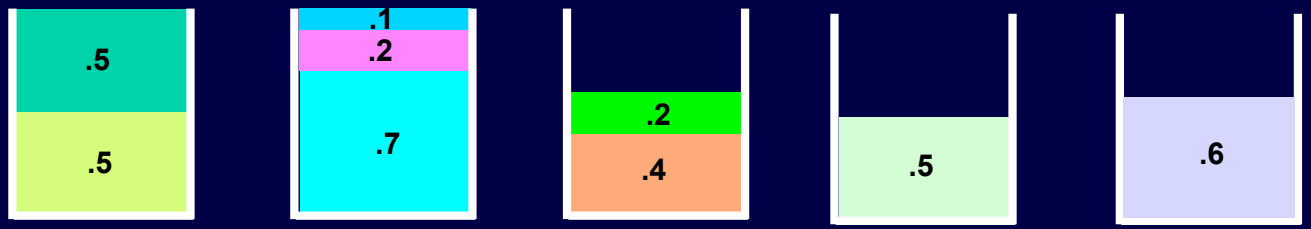
Approximation Algorithms
（Performance Guaranteed）

.5 .7 .5 .2 .4 .2 .5 .1 .6

Next Fit Packing Algorithm



First Fit Packing Algorithm



$N = 5$

$$\frac{N}{N_0} \leq 1.7$$

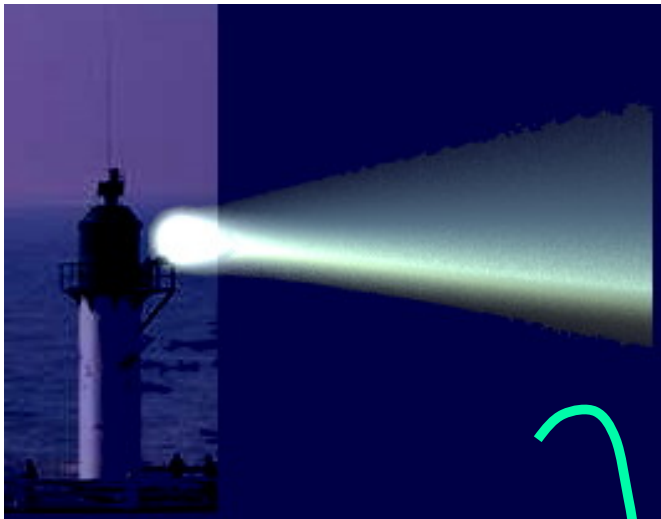
Solution of Optimization Problems

- Finding an Acceptable solution at a tolerably High cost

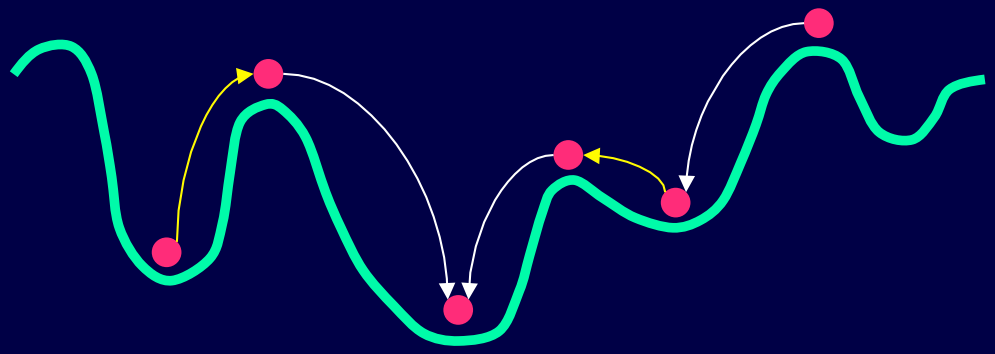
Examples :

1. Solving the Scheduling Problem using the Simulated Annealing Method

2. Solving the Graph Bi-partitioning Problem using a Genetic Algorithm



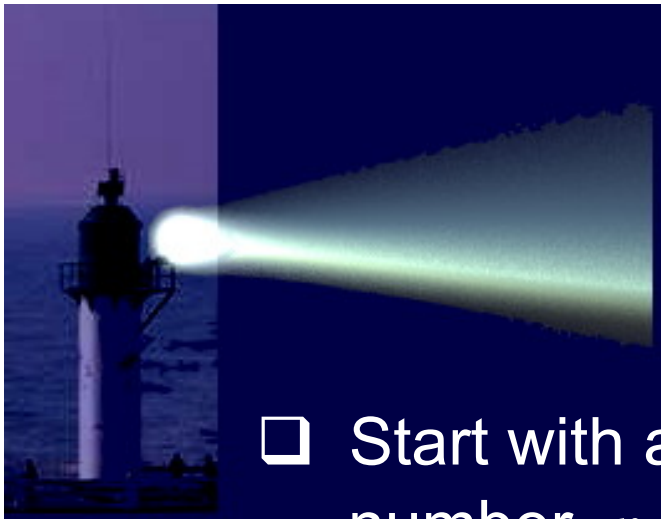
Simulated Annealing (Downhill - Uphill)



- Non-zero probability for up-hill moves
- Probability depends on the magnitude of up-hill move
- Probability depends on total search time

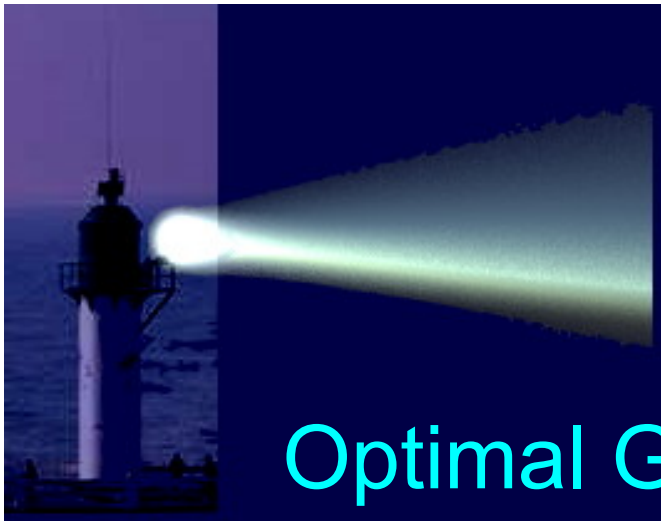
$$Prob\,(s \rightarrow s') = \begin{cases} 1 & if\ E' - E \le 0 \\ e^{-\frac{E'-E}{T}} & if\ E' - E > 0 \end{cases}$$

T : a decreasing function of time



Genetic Algorithm

- Start with an initial generation of solutions (a finite number, n, of randomly chosen solutions)
- Crossover Operator : Combine two current solutions to yield one or more offsprings
  Mutation Operator : Mutate a current solution to produce an offspring
- Select from the pool the next generation of n solutions according to a fitness criterion
- Stop according to some stopping criteria : number of generations, incremental improvement of the best solution



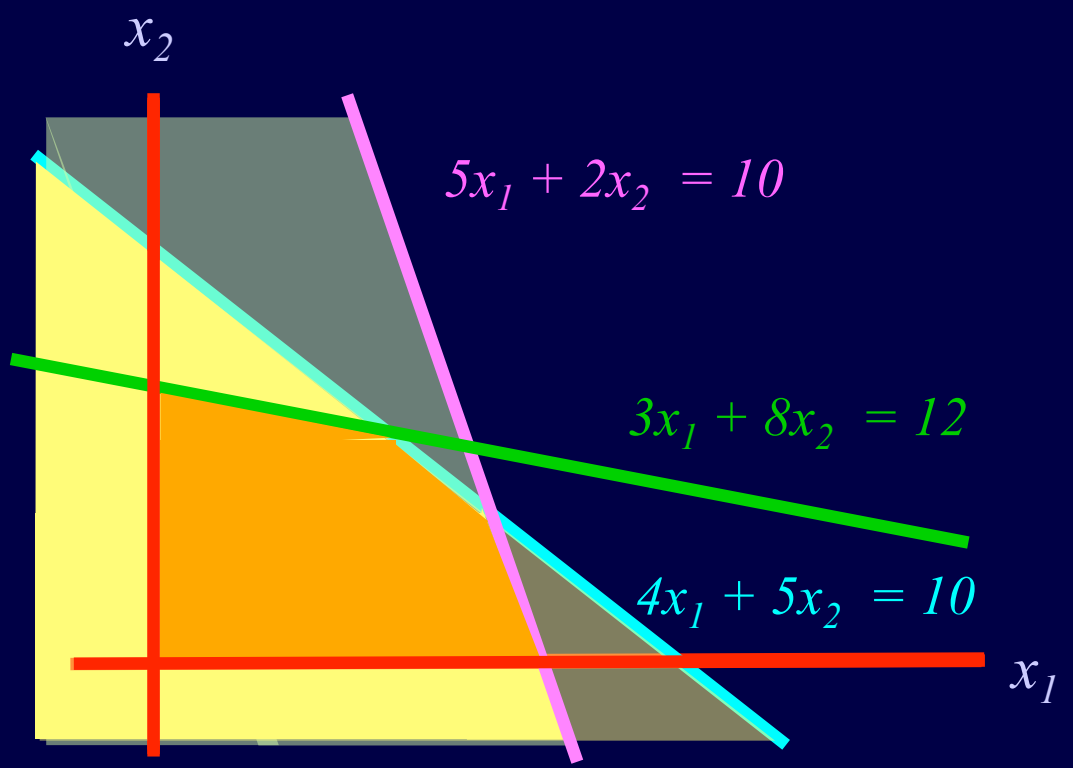
An Example

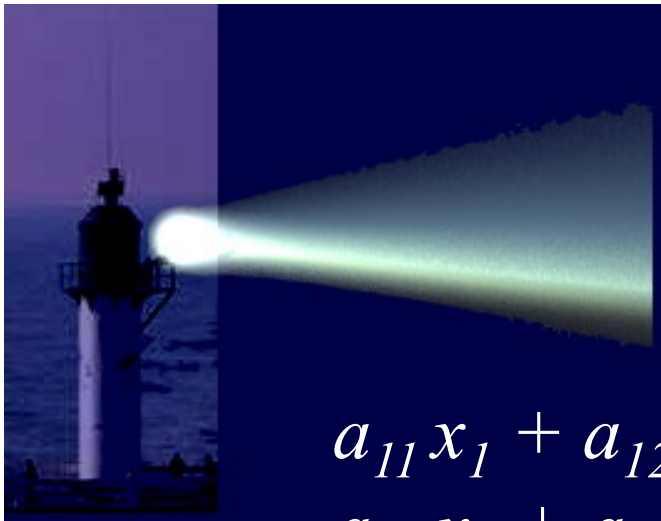
Optimal Graph Constraint Reduction

- An optimal solution
- An efficient algorithm
- A special case of a well-studied problem

Linear Programming

$$4x_1 + 5x_2 \leq 10$$
$$5x_1 + 2x_2 \leq 10$$
$$3x_1 + 8x_2 \leq 12$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$
$$\text{maximize} \quad x_1 + x_2$$





Linear Programming

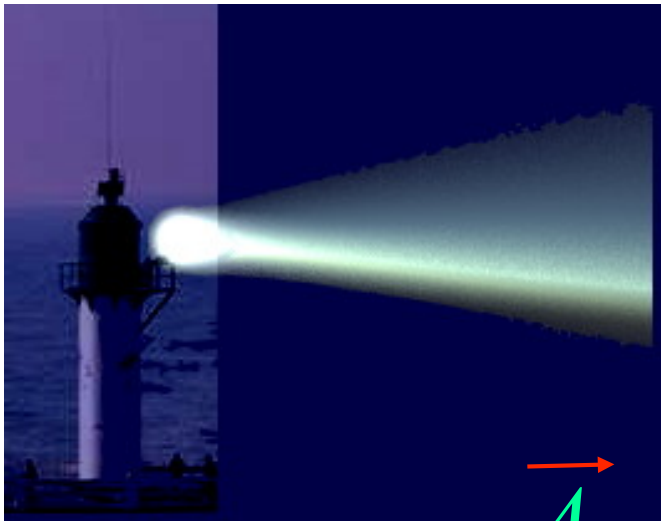
$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + \ldots\ldots + a_{1m}x_m &\le b_1 \\
a_{21}x_1 + a_{22}x_2 + \ldots\ldots + a_{2m}x_m &\le b_2 \\
&\vdots \\
a_{n1}x_1 + a_{n2}x_2 + \ldots\ldots + a_{nm}x_m &\le b_n
\end{aligned}
$$

$$
\textit{minimize} \quad c_1x_1 + c_2x_2 + \ldots\ldots + c_mx_m
$$

$$
\overrightarrow{A}_{n\times m} \quad \overrightarrow{X}_{m\times 1} \le \overrightarrow{b}_{n\times 1}
$$

Solution space
Convex polytope

$$
\textit{minimize} \quad \overrightarrow{c}_{1\times m} \quad \overrightarrow{X}_{m\times 1}
$$



Linear Programming

$$\vec{A}_{n\times m}\ \vec{X}_{m\times 1} \le \vec{b}_{n\times 1}$$

- A consistent system (non-empty solution space)

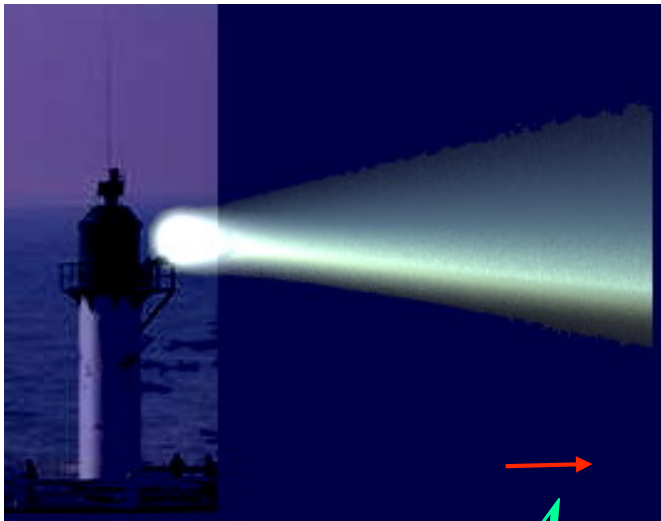
Linear Programming

$$\overrightarrow{A}_{n\times m}\ \overrightarrow{X}_{m\times 1} \le \overrightarrow{b}_{n\times 1}$$

- A redundant constraint

$$a_{i1}x_1 + a_{i2}x_2 + \ldots\ldots + a_{im}x_m \le b_i$$

$$\overrightarrow{A}_{(n-1)\times m}\ \overrightarrow{X}_{m\times 1} \le \overrightarrow{b}_{(n-1)\times 1}$$



Linear Programming

$$\vec{A}_{n\times m}\ \vec{X}_{m\times 1} \le \vec{b}_{n\times 1}$$

- Remove a largest possible subset of redundant constraints

$$\vec{A}_{n'\times m}\ \vec{X}_{m\times 1} \le \vec{b}_{n'\times 1}$$

Linear Programming

$$\vec{A}_{n\times m}\ \vec{X}_{m\times 1} \le \vec{b}_{n\times 1}$$

- A smallest possible set of constraints

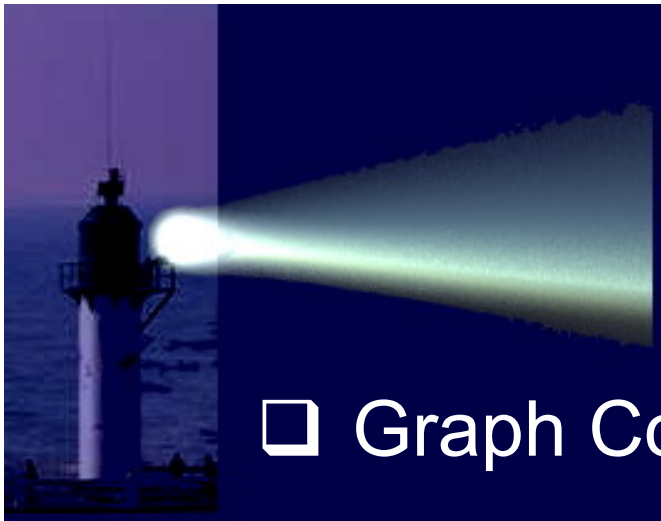
$$\vec{B}_{n''\times m}\ \vec{X}_{m\times 1} \le \vec{d}_{n''\times 1}$$

Graph Constraints

$$x_i - x_j \geq c$$

A System of Graph Constraints

- Consistency
- A redundant constraint
- Removal of a largest possible subset of constraints
- A smallest possible subset of graph constraints



Graph Constraints and Constraint Graph

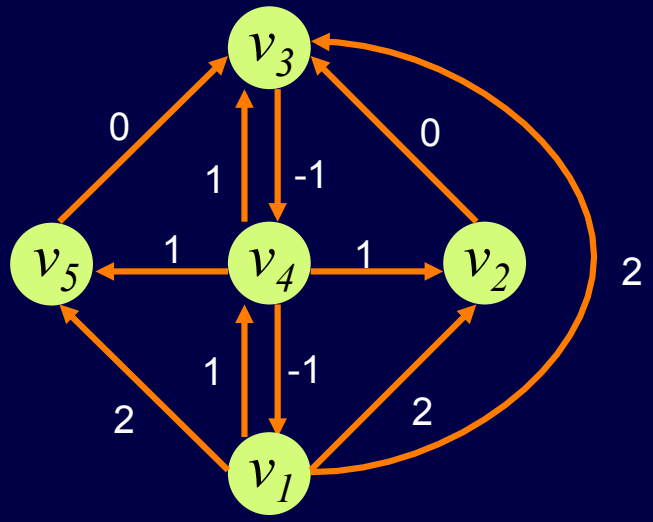
- Graph Constraint

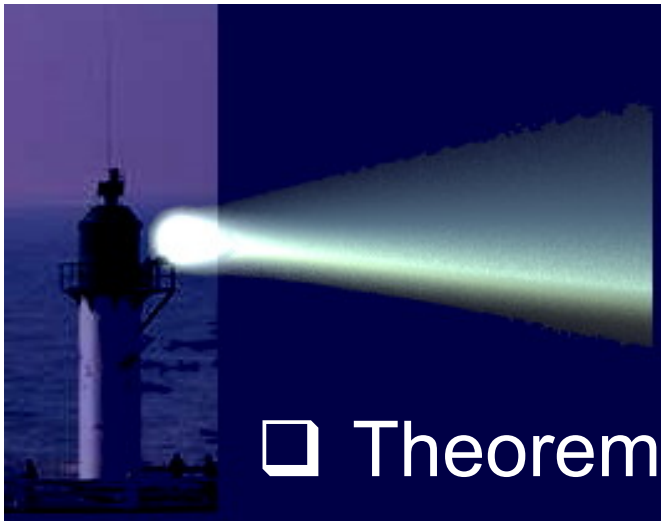
$$x_i - x_j \geq c$$

$$v_j \xrightarrow{c} v_i$$

- Constraint graph of a system of graph constraints

$$x_2 - x_1 \geq 2$$
$$x_3 - x_1 \geq 2$$
$$x_4 - x_1 \geq 1$$
$$x_5 - x_1 \geq 2$$
$$x_3 - x_2 \geq 0$$
$$x_4 - x_3 \geq -1$$
$$x_1 - x_4 \geq -1$$
$$x_2 - x_4 \geq 1$$
$$x_3 - x_4 \geq 1$$
$$x_5 - x_4 \geq 1$$
$$x_3 - x_5 \geq 0$$

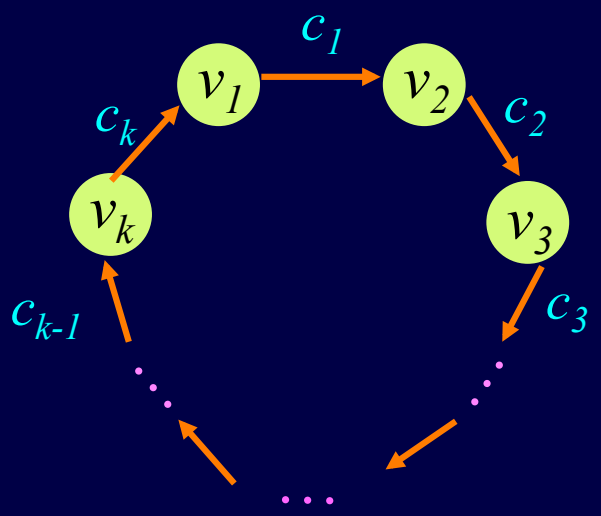




Consistency

- Theorem

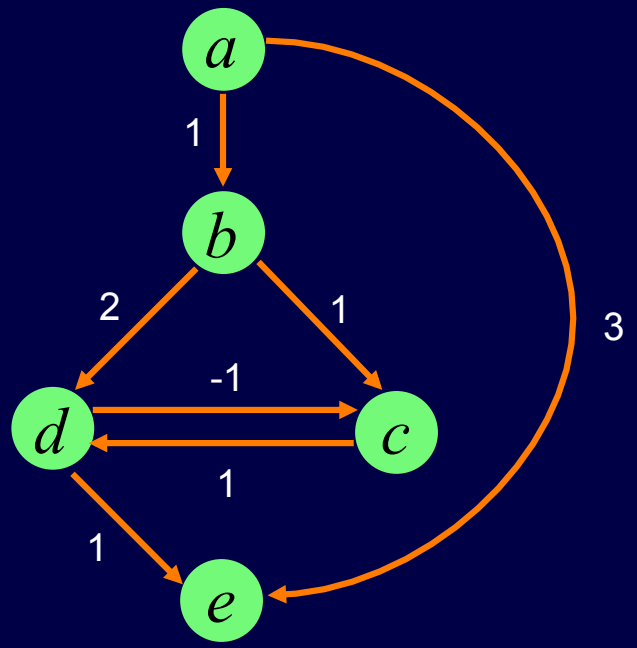
  A system of graph constraints is consistent if and only if its constraint graph has no directed positive cycle



$$c_1 + c_2 + c_3 + \ldots\ldots + c_{k-1} + c_k > 0$$

Redundancy

- An edge (u, v) is redundant if there is a (directed) path from u to v, P, such that $w(u,v) \leq w(P)$ and (u,v) is not on P



(a, e) is redundant

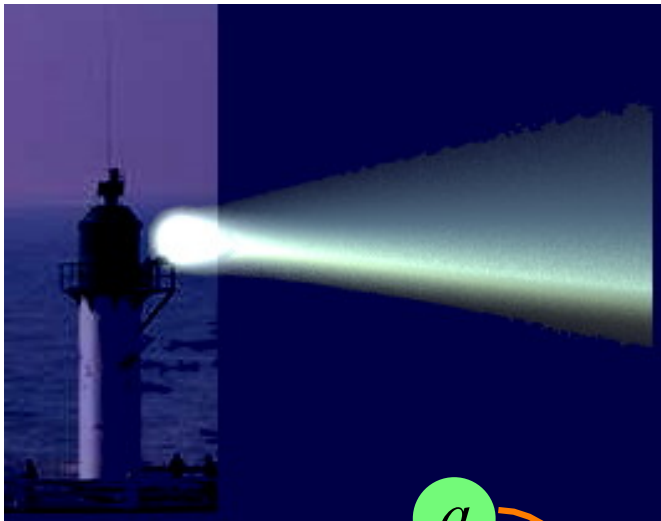
$w(a \rightarrow b \rightarrow d \rightarrow e) = 4$

(b, c) is redundant

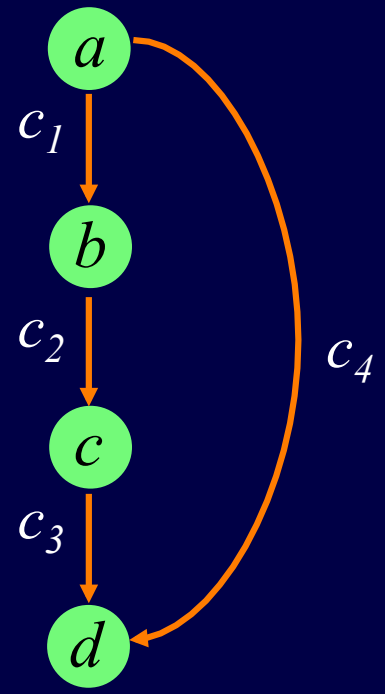
$w(b \rightarrow d \rightarrow c) = 2$

(b, d) is redundant

$w(b \rightarrow c \rightarrow d) = 2$



Redundancy



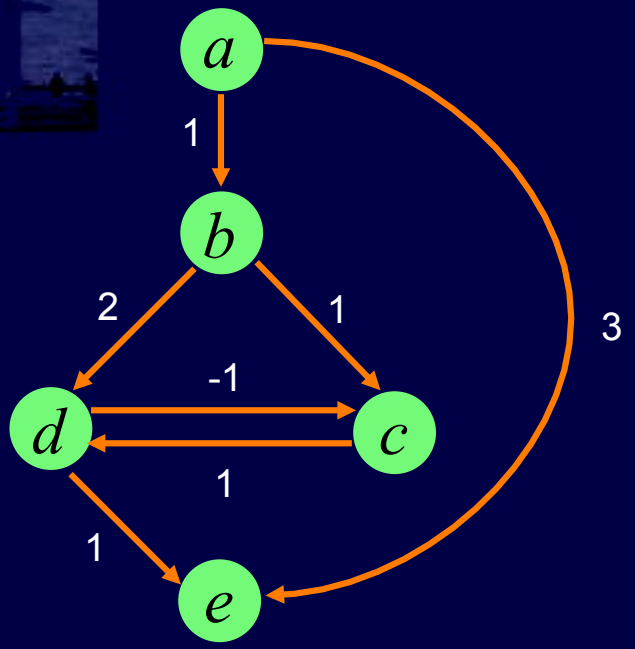
$$b - a \geq c_1$$
$$c - b \geq c_2$$
$$d - c \geq c_3$$

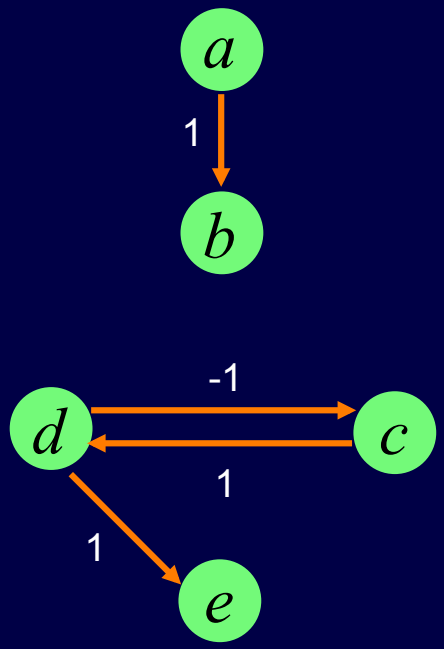
$$d - a \geq c_1 + c_2 + c_3$$

which implies $d - a \geq c_4$

if $c_1 + c_2 + c_3 \geq c_4$

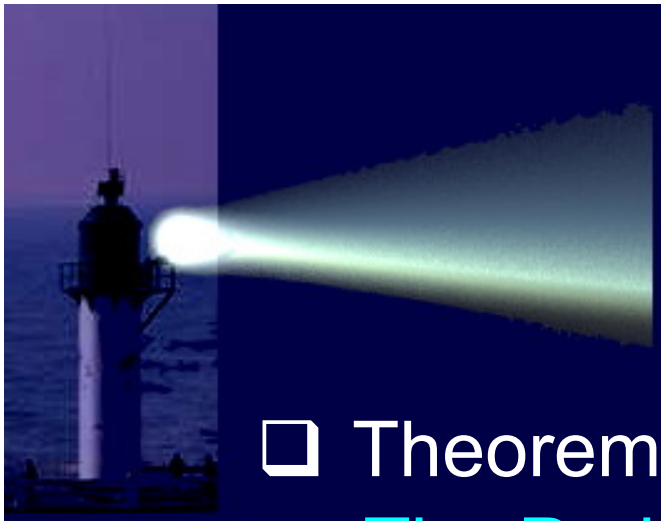
Redundancy





(a, e) is redundant
(b, c) is redundant
(b, d) is redundant

- *Remove one redundant edge at a time*
- *Resultant graph is order dependent*



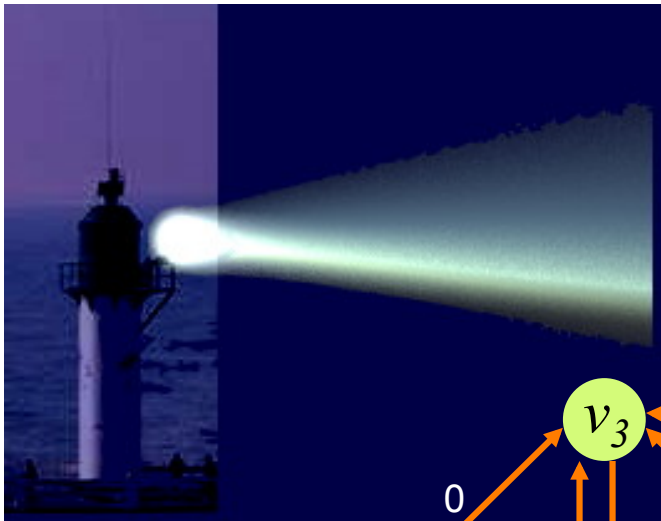
Removal of Redundant Constraints

- Theorem :

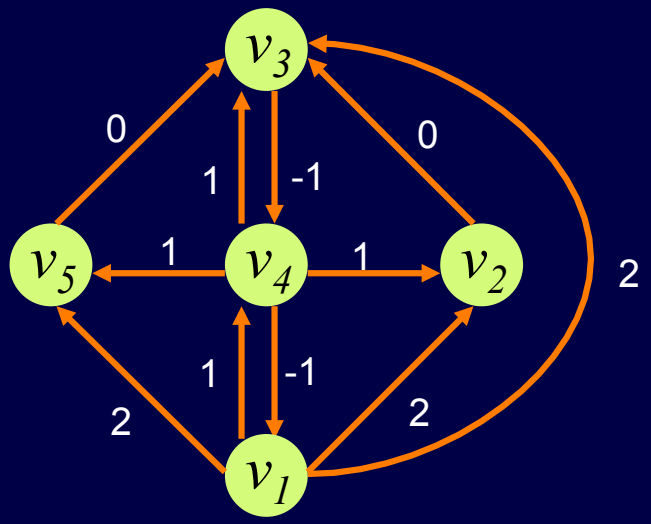
  The Problem of removing a largest possible subset of redundant constraints is NP-complete

Proof :

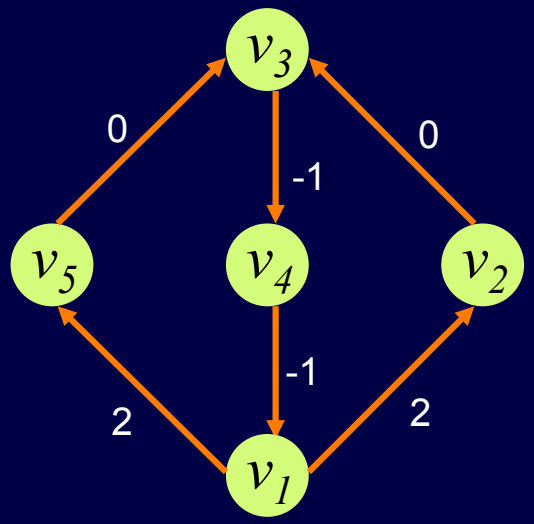
Reduction from the directed Hamiltonian Cycle Problem

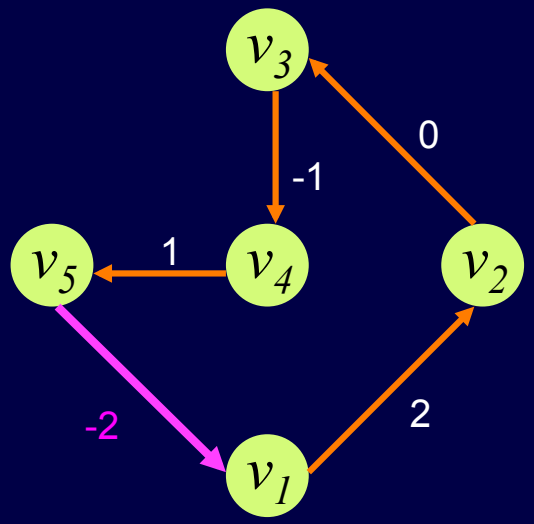


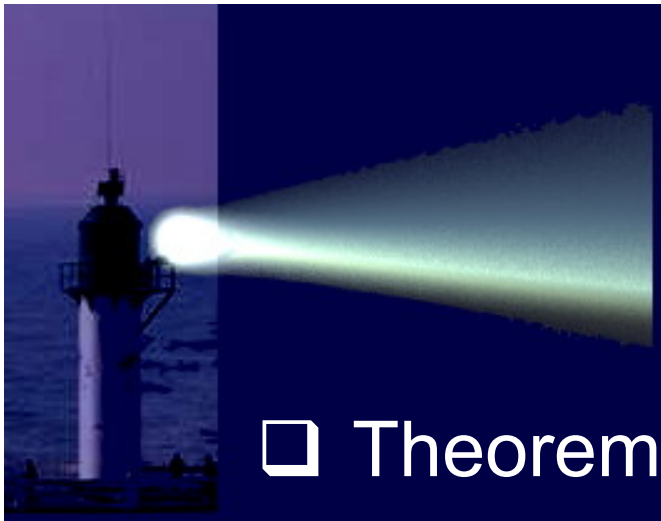
Optimal Reduction



$$x_2 - x_1 \geq 2$$
$$x_3 - x_2 \geq 0$$
$$x_4 - x_3 \geq -1$$
$$x_5 - x_4 \geq 1$$
$$x_1 - x_5 \geq -2$$





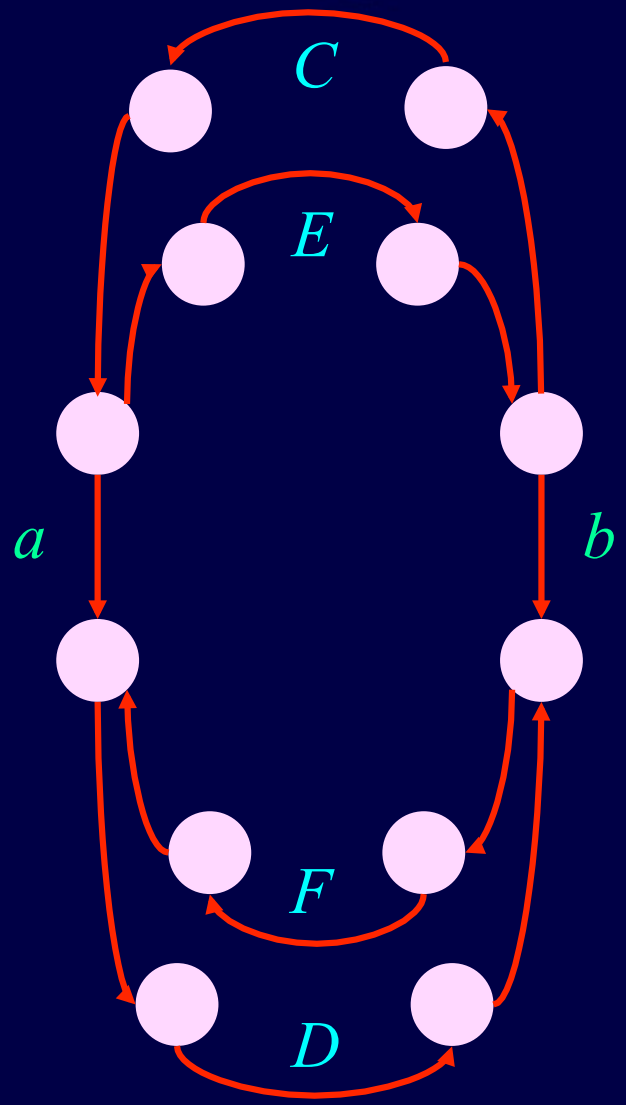


Zero Cycles

- Theorem

  If G contains no zero-cycle, $G - R_G$ is the unique optimal reduction of G where R_G is the set of all redundant edges of G

Zero Cycles



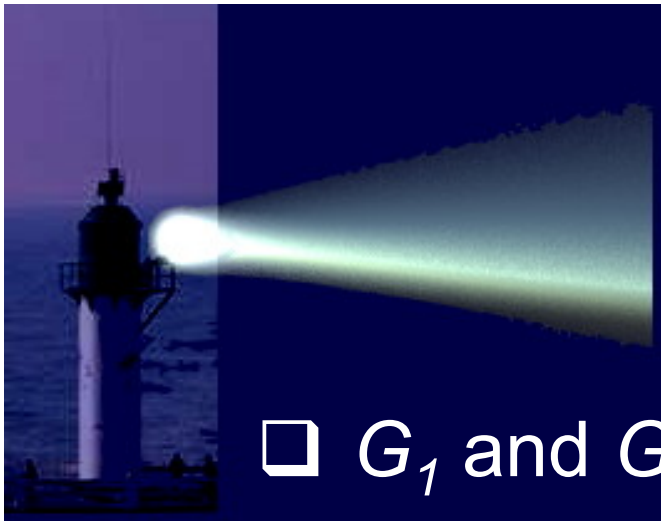
$$a \leq E + b + F$$

$$b \leq C + a + D$$

$$(C + E) + (D + F) \geq 0$$

$$C + E = 0$$

$$D + F = 0$$



Optimal Reduction

- G_1 and G_2 are constraint graphs with the same vertex set
- $W_{G_1}^*(u, v)$ is the path length of the longest path from u to v in G_1
  $W_{G_2}^*(u, v)$ is the path length of the longest path from u to v in G_2
- Theorem :
  G_1 and G_2 define the same solution space if and only if $W_{G_1}^*(u, v) = W_{G_2}^*(u, v)$ for all u and v

Optimal Reduction

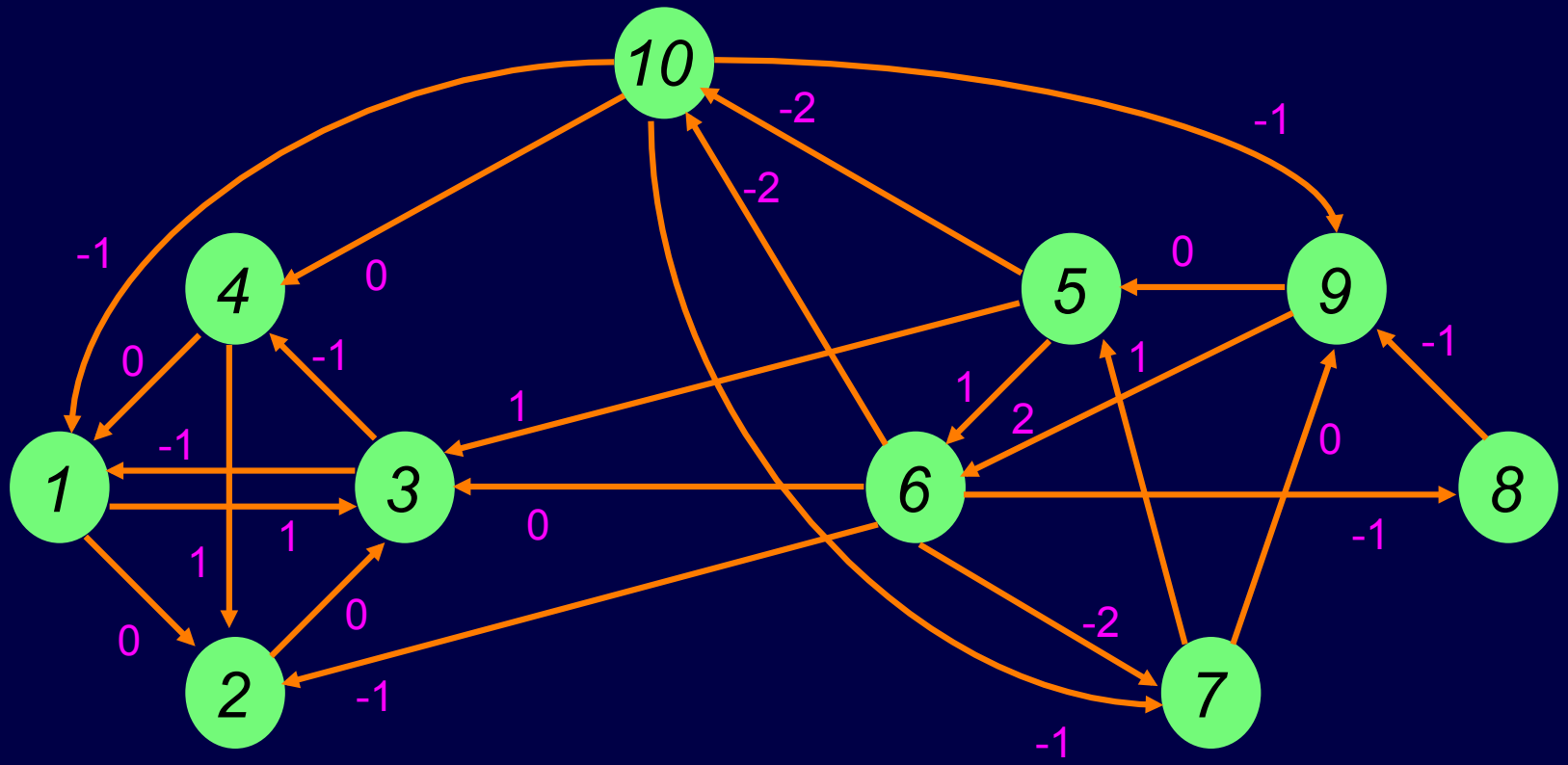
Problem :

For a given G, find G_0 such that

(i) $W_G^*(u, v) = W_{G_0}^*(u, v)$ for all u and v

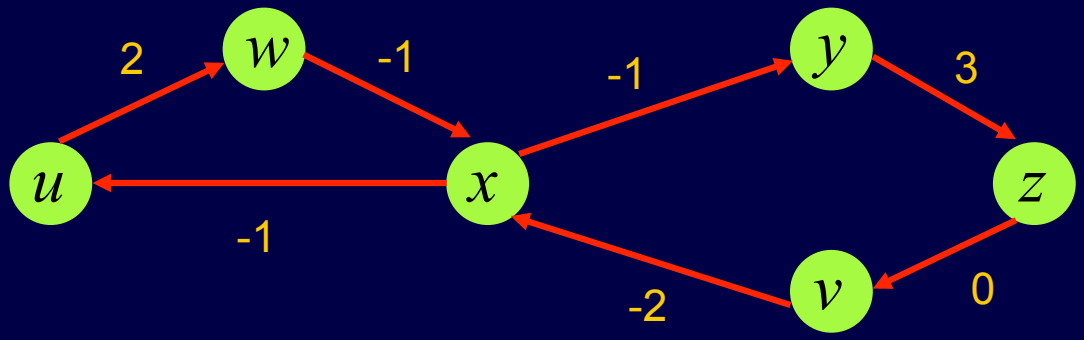
(ii) G_0 has the minimum number of edges

Constraint Graph



Dependency Relation

- u and v are dependent if
  - (i) $u = v$, or
  - (ii) there is a zero closed walk that contains u and v

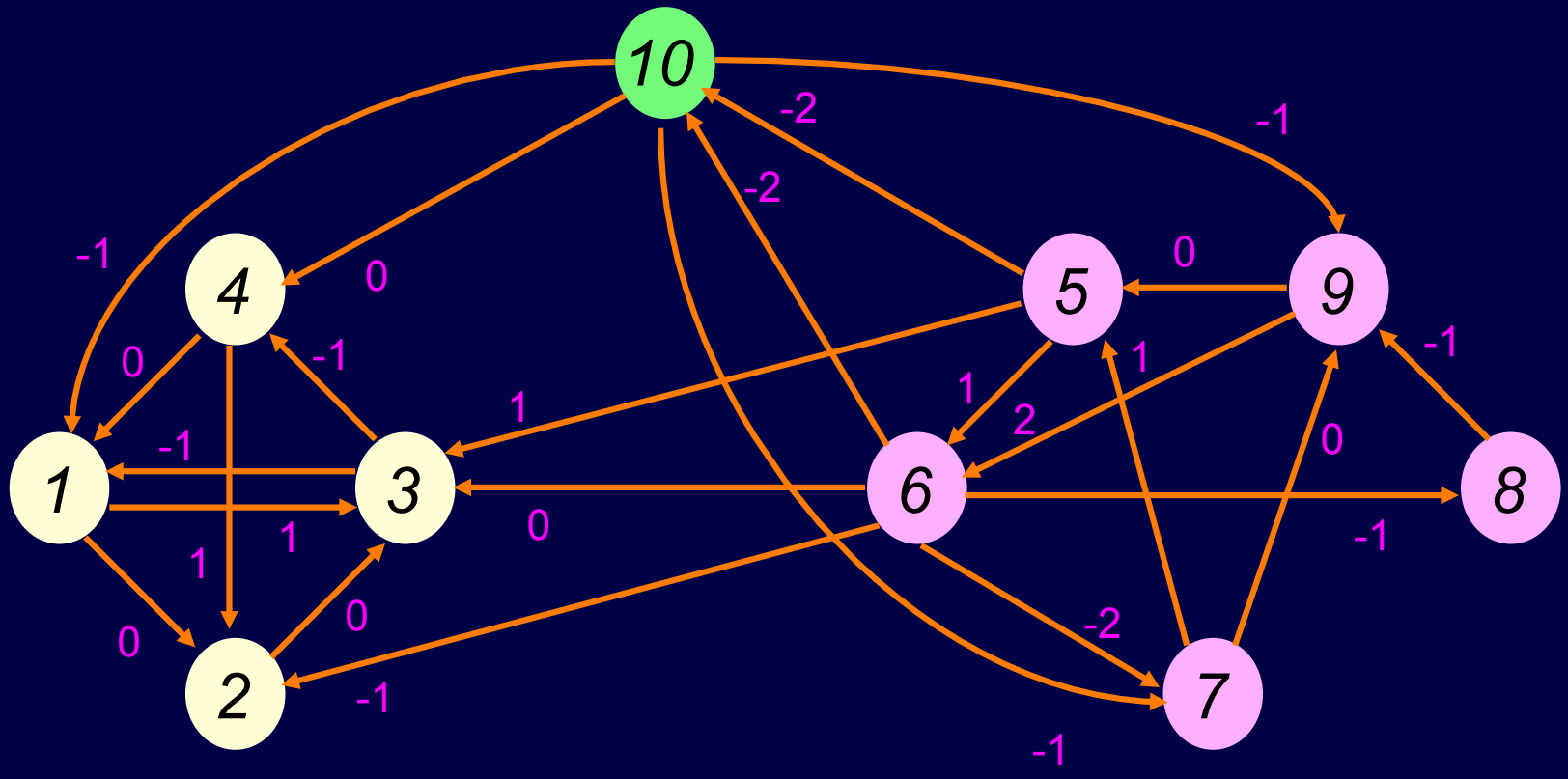


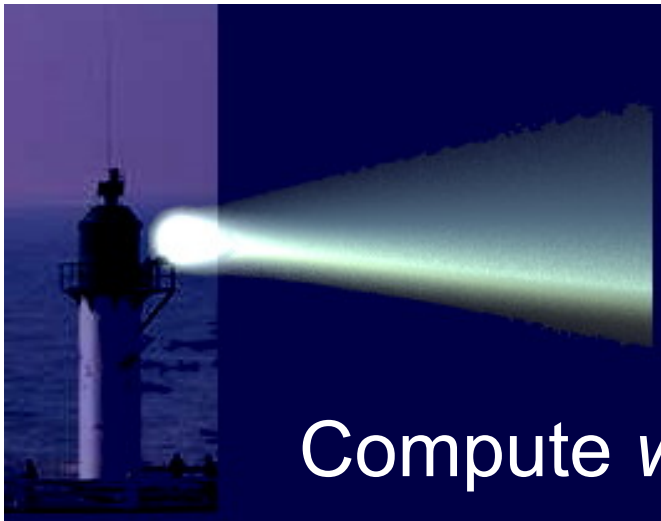
- u and v are dependent if and only if

  $w_G^*(u, v) + w_G^*(v, u) = 0$

Dependency Relation

The Dependency Relation is an equivalence relation which induces a partition on the set of vertices, π_G





Step 1

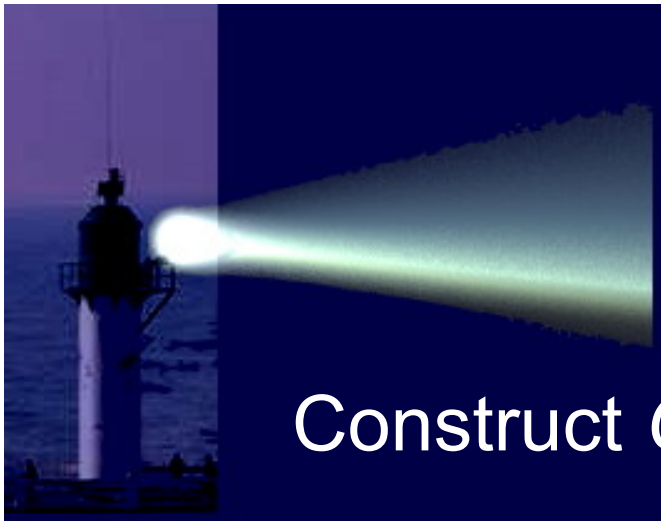
Compute $w_G^*(u, v)$ and π_G

v / u	1	2	3	4	5	6	7	8	9	10
1	0	1	1	0						
2	-1	0	0	-1						
3	-1	0	0	-1						
4	0	1	1	0						
5	0	1	1	0	0	1	-1	0	-1	-1
6	-1	0	0	-1	-1	0	-2	-1	-2	-2
7	1	2	2	1	1	2	0	1	0	0
8	0	1	1	0	0	1	-1	0	-1	-1
9	1	2	2	1	1	2	0	1	0	0
10	0	1	1	0	0	1	-1	0	-1	0

$V_1 = \{ 1, 2, 3, 4 \}$

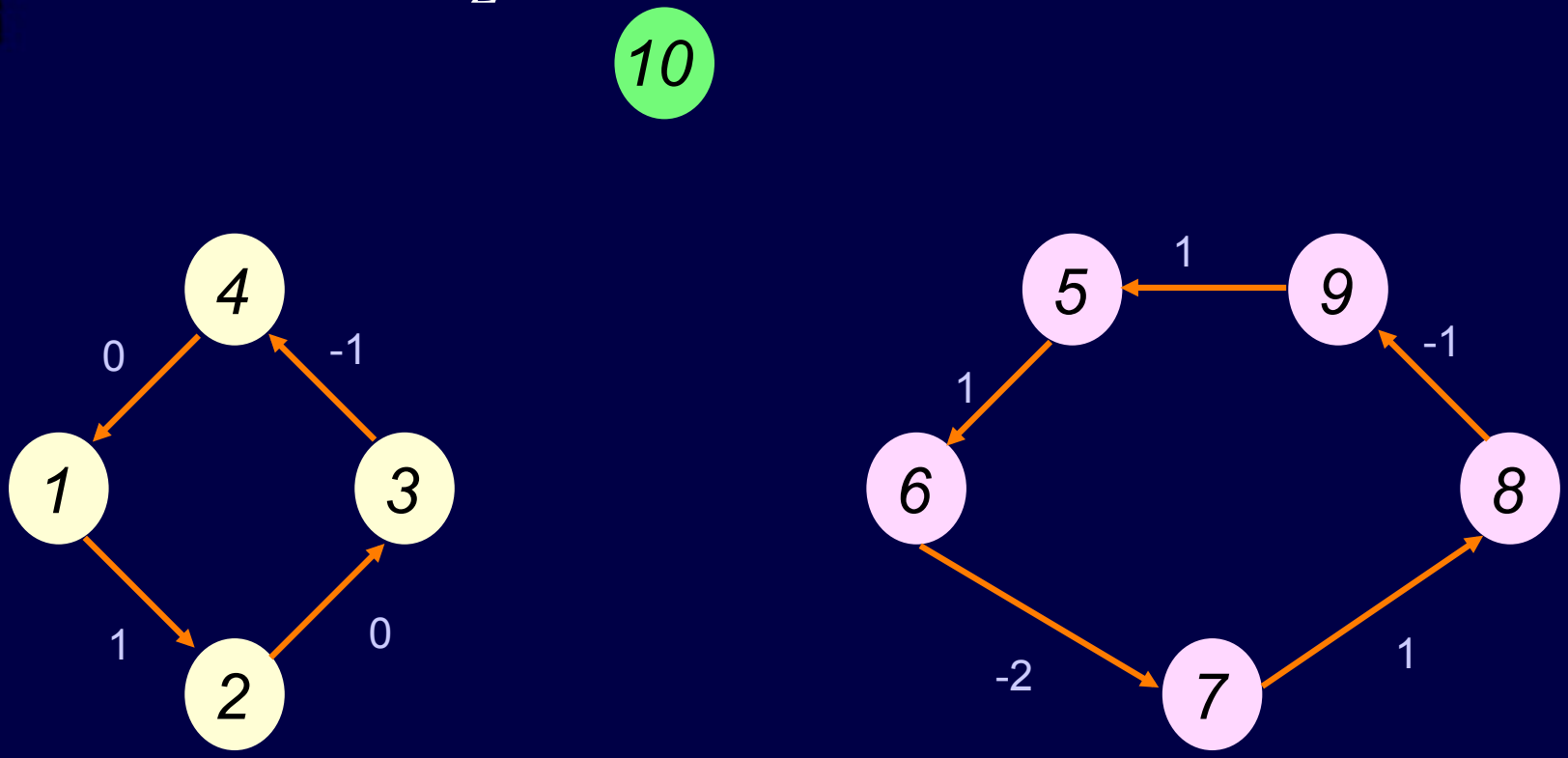
$V_2 = \{ 5, 6, 7, 8, 9 \}$

$V_3 = \{ 10 \}$



Step 2

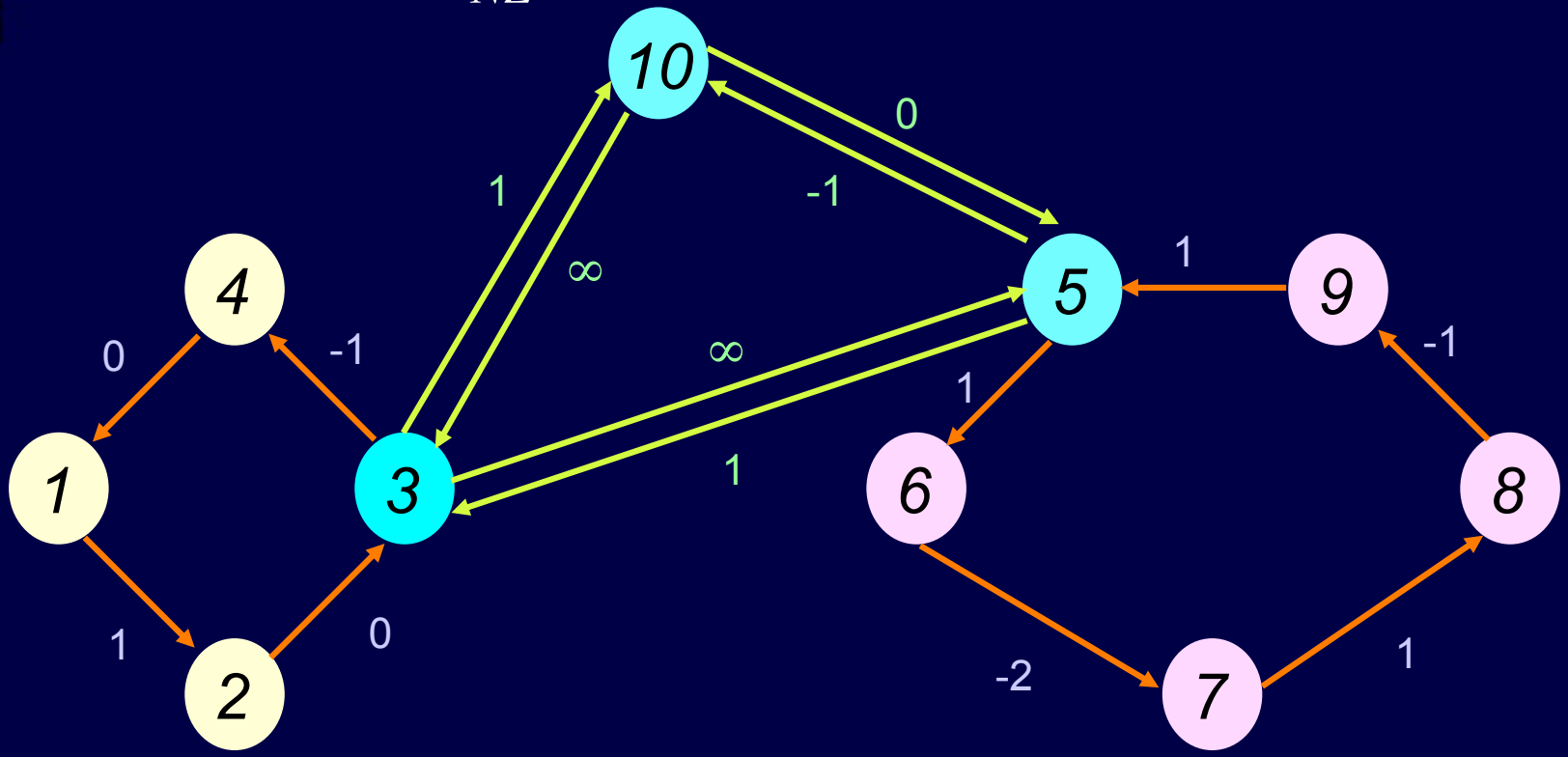
Construct G_Z



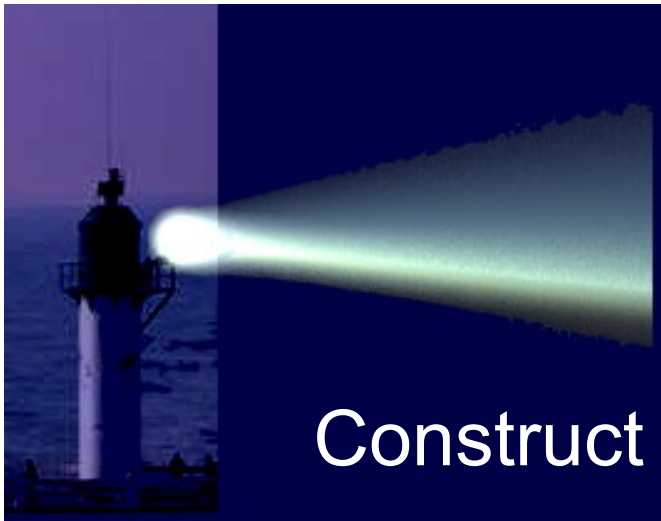
$V_1 = \{ 1, 2, 3, 4 \}$; $V_2 = \{ 5, 6, 7, 8, 9 \}$; $V_3 = \{ 10 \}$

Step 3

Construct G_{NZ}

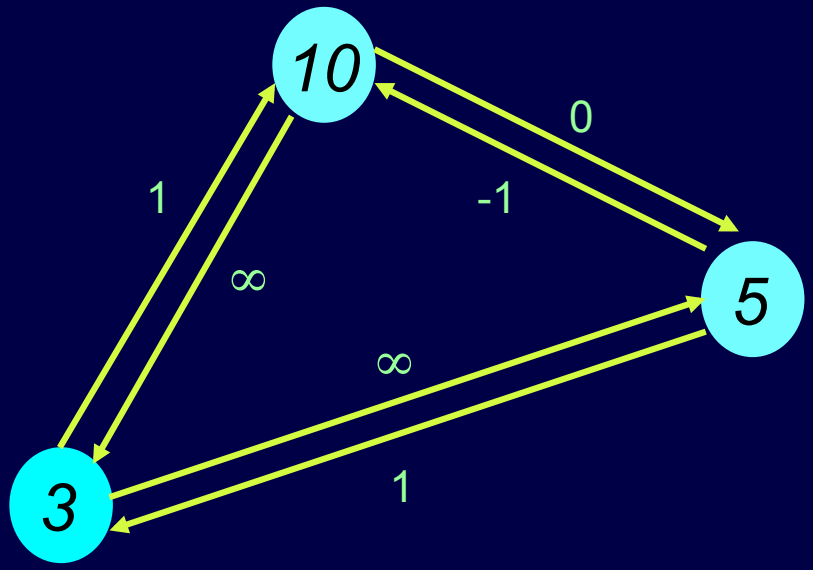


$V_1 = \{ 1, 2, 3, 4 \}$; $V_2 = \{ 5, 6, 7, 8, 9 \}$; $V_3 = \{ 10 \}$

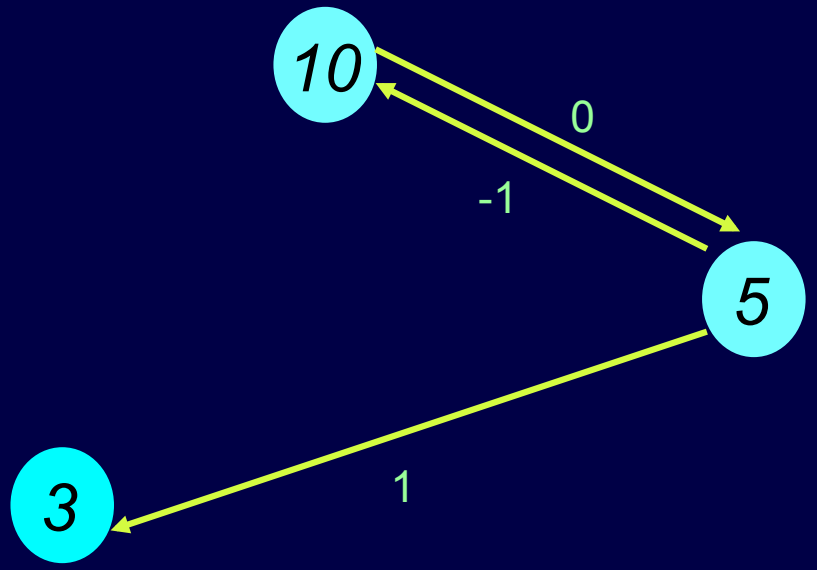


Step 4

Construct G_{NZ} and G'_{NZ}

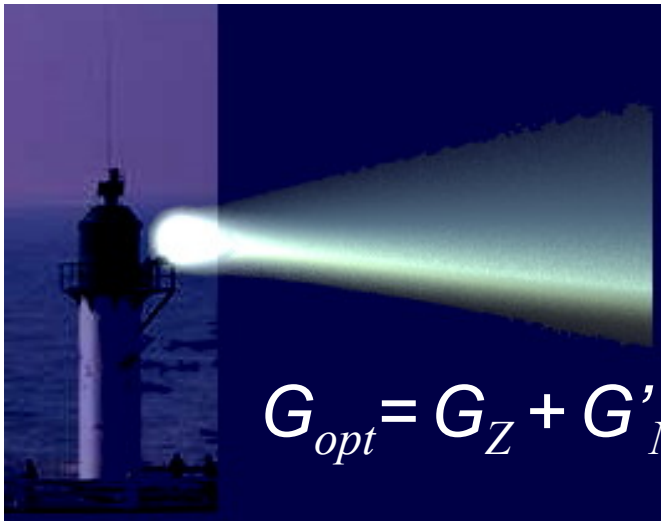


G_{NZ}



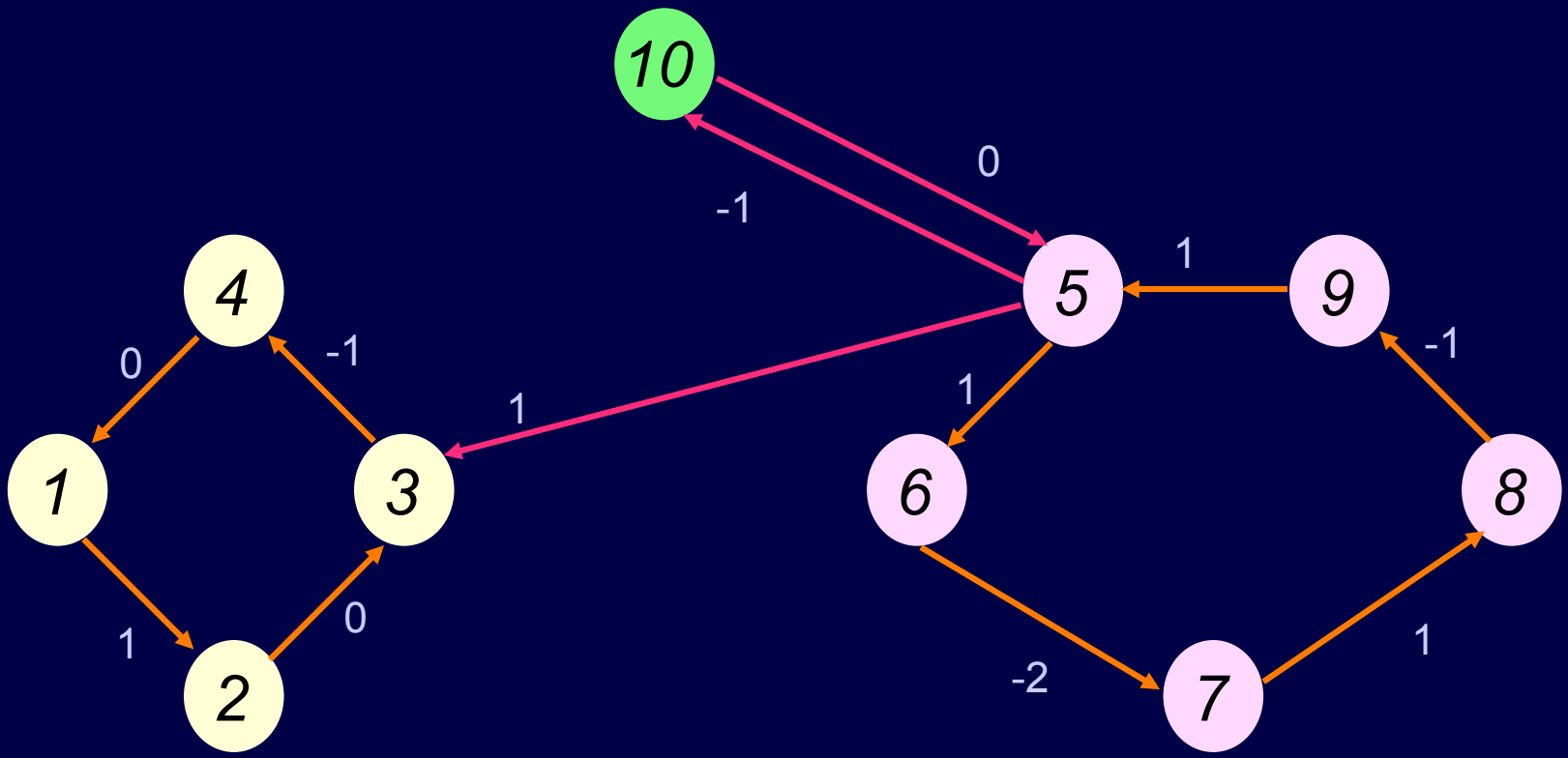
G'_{NZ}

$V_1 = \{1, 2, 3, 4\}$; $V_2 = \{5, 6, 7, 8, 9\}$; $V_3 = \{10\}$



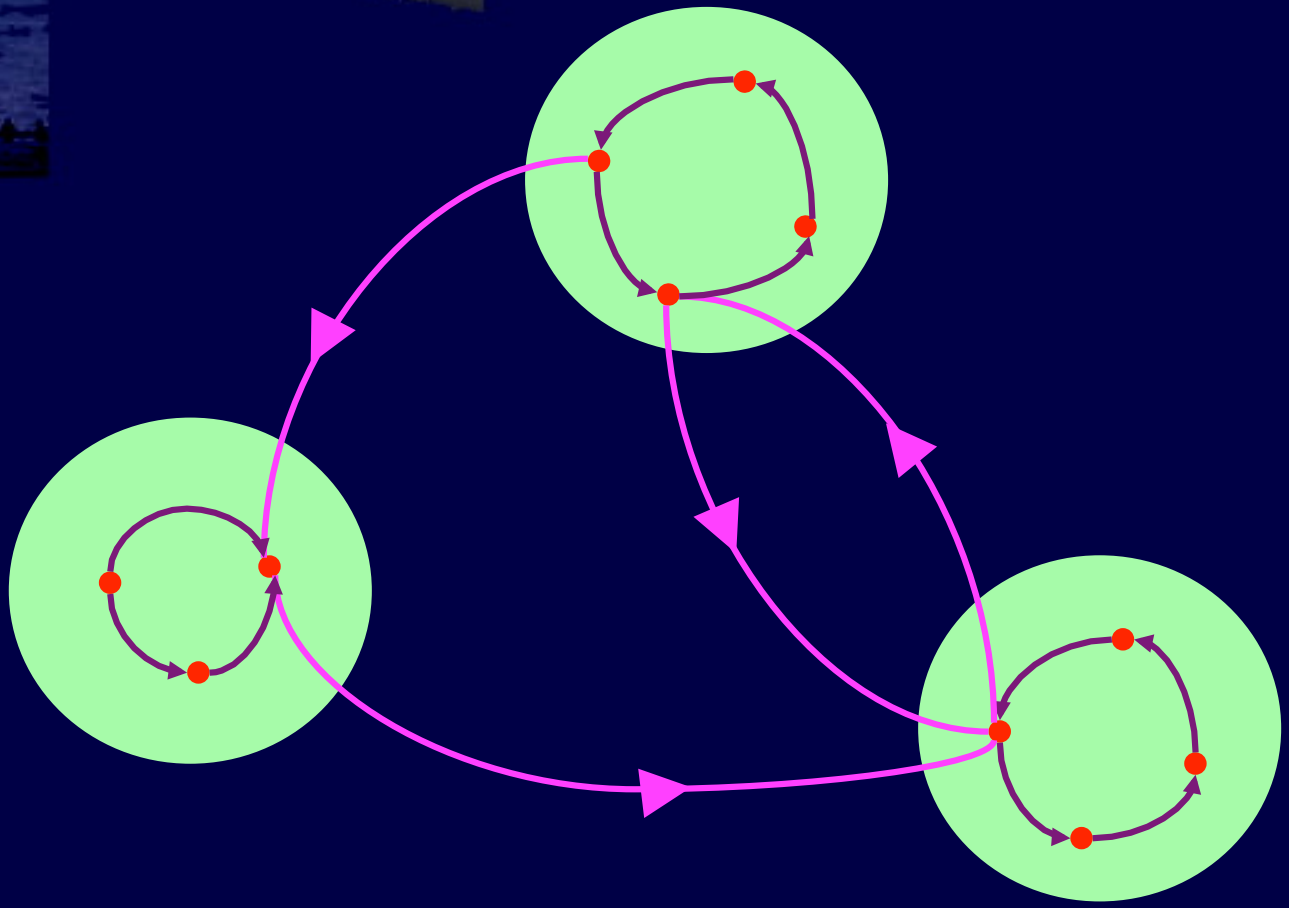
Step 5

$G_{opt} = G_Z + G'_{NZ}$: Optimal Reduction of G

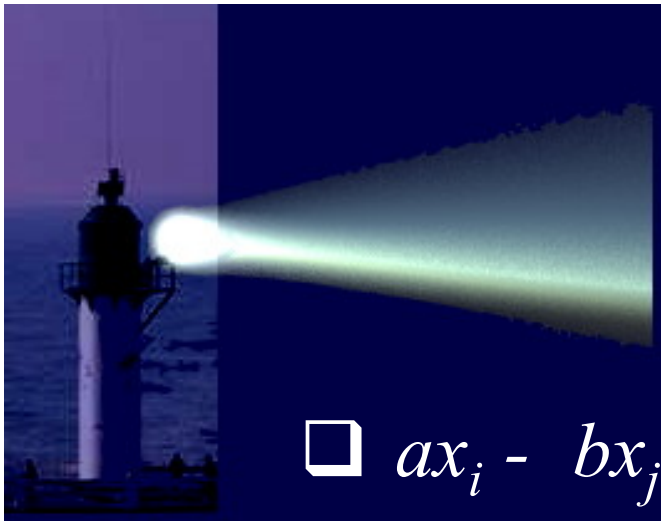


Time complexity : $O(n^2 \log n + mn)$

Why Minimum ?



Dependency Classes



Extensions

- $ax_i - bx_j \geq c$
- $x_i - x_j = x_k - x_l$
- $x_i - x_j - x_k \geq c$

Final Remark

optimality

slowly but surely

short and sweet

efficiency

too little, too late

quick and dirty