# Weighted vertex cover

$$Min \sum_{i=1}^{n} w_i x_i$$

$$x_i + x_j \geq 1 \qquad \forall [v_i, v_j] \in E$$

$$x_i \in \{0,1\} \qquad \forall v_i \in V$$

$$\Downarrow$$

$$z^* LR$$

# App-MIN-WVC (G,W)

- c=∅
  $\bar{x}$ (Opt LR)
  for i=1 to n do
  
  if $\bar{x}$>= ½ then c=c∪{$u_i$}
  
  return c

$$z^* \leq w(c^*)$$

$$z^* = \sum_{i=1}^{n} w_i \bar{x}_i \geq \sum_{i \mid \bar{x}_i \geq \frac{1}{2}} w_i \bar{x}_i \geq \sum_{i \mid \bar{x}_i \geq \frac{1}{2}} w_i \frac{1}{2} = \frac{1}{2} w(c)$$

$$w(c) \leq 2z^* \leq 2w(c^*) \Rightarrow \frac{w(c)}{w(c^*)} \leq 2$$

# UnWeighted Vertex Cover

$$c \quad c^{*} \quad \Rightarrow \frac{c}{c^{*}} \leq 2$$

$$c^{*} \geq \frac{1}{2}c \Rightarrow \frac{c}{c^{*}} \leq \frac{c}{\frac{1}{2}c} = 2 \Rightarrow$$

$$\frac{c}{c^{*}} \leq 2$$

# Vertex cover/polynomial variants

- Trees/Forests
- Interval graphs
- Chordal graphs
- Bipartite graphs

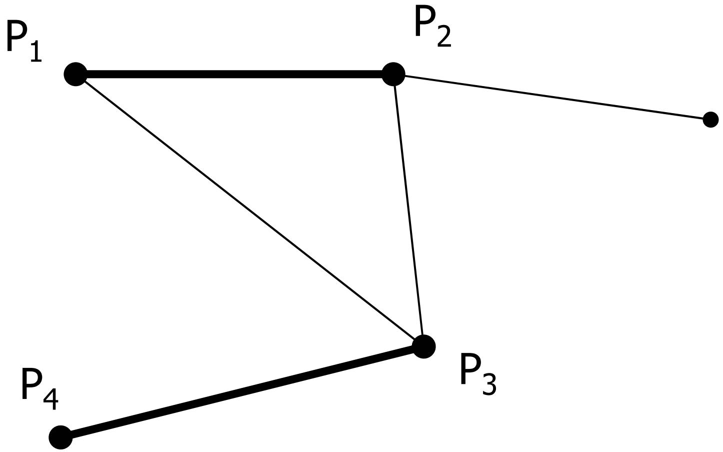
# Minimum Bin Packing

- Finite set of objects
- t(n) size of object $u \in U$ (integer)
- B capacity of a bin
- U1, U2, …,Um with

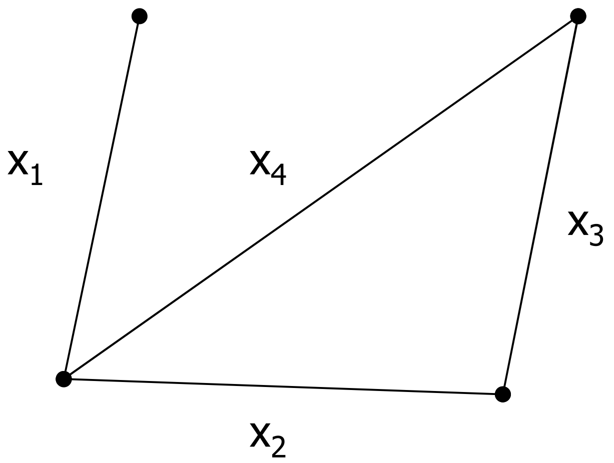
$$\sum_{u \in U_i} t(u) \leq B, \qquad 1 \leq i \leq m$$

# Bin Packing (polynomial case)

- $P_1, P_2, \ldots, P_n$ objects and infinite number of bins of capacity B
- $P_i > B/3 \Rightarrow 2$ από $P_i$ at most
- max couples



# Edge cover (exercise)



# M-Processor Scheduling

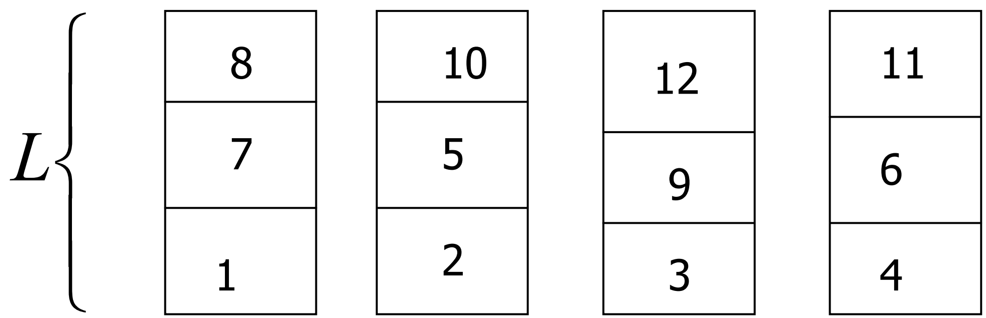
- m identical Processors
  - $P_1, \ldots, P_m$
- n independent Tasks (n>m)
- execution times: $t_1, t_2, \ldots, t_n$
- Min finished time (without preemption)
- NP-complete even if m=2

# Approximation Algorithm
# List Scheduling

- For i=1 to n

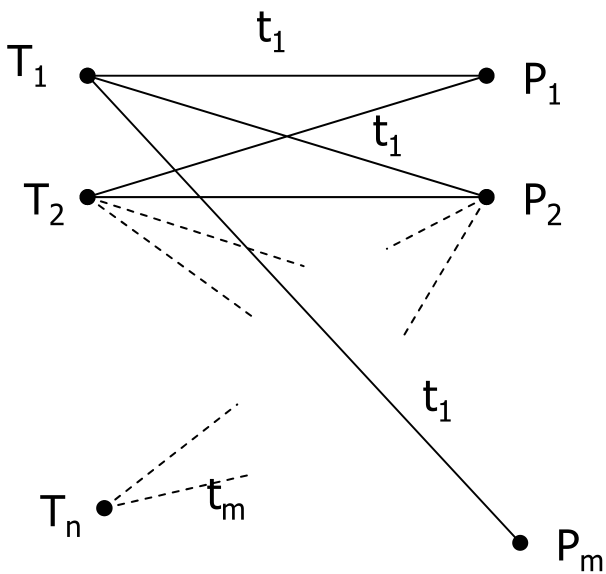
  place task i on the processor with the least charge

  - Ex. n=12, m=4




$$\frac{A(I)}{OPT(I)} \leq 2 - \frac{1}{m}$$

# Identical Machines




$$Min \quad C_{\max}$$

$$\sum_{i=1}^{n} t_i x_{ip} \le C_{\max}, \qquad p = 1, \ldots, m$$

$$\sum_{p=1}^{m} x_{ip} = 1, \qquad i = 1, \ldots, n$$

$$x_{ip} = \begin{cases} 0 \\ 1 \quad if\_i\_on\_p \end{cases}$$

$$1 \le i \le n$$

$$1 \le p \le m$$

# List scheduling

- I instance
- $P_1$ the most charged (by A)
- $P_1 \rightarrow L$
- J the last placed on $P_1$ (here j=8)
- $\forall P_k$ charge $\geq L-t_j$
- charge of $P_1 = L-t_j$ before placement of j

# Approximation Algorithm List Scheduling

$$\left.\begin{aligned}\sum_{i=1}^{n} t_i &\geq m(L - t_j) + t_j \\ Opt(I) &\geq \frac{\sum_{i=1}^{n} t_i}{m}\end{aligned}\right\} \Rightarrow$$

$$Opt(I) \geq (L - t_j) + \frac{t_j}{m} = A(I) - (1 - \frac{1}{m})t_j$$

$$\Downarrow \qquad Opt(I) \geq t_j$$

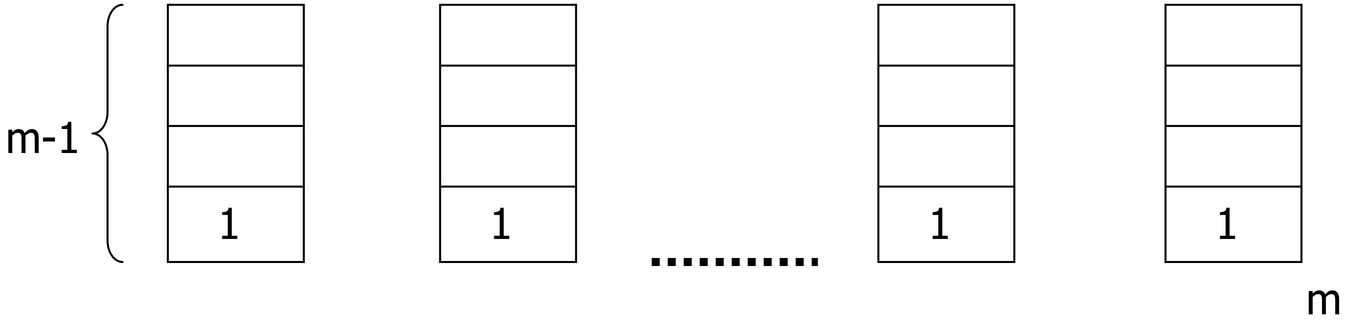
$$\frac{A(I)}{Opt(I)} \leq 2 - \frac{1}{m}$$

# The approximation is tight

- I instance with
  - $n=m(m-1)+1$
  - $t_1=t_2=\ldots=t_{n-1}=1$, $t_n=m$
  - $Opt(I)=m$,  $A(I)=2m-1$

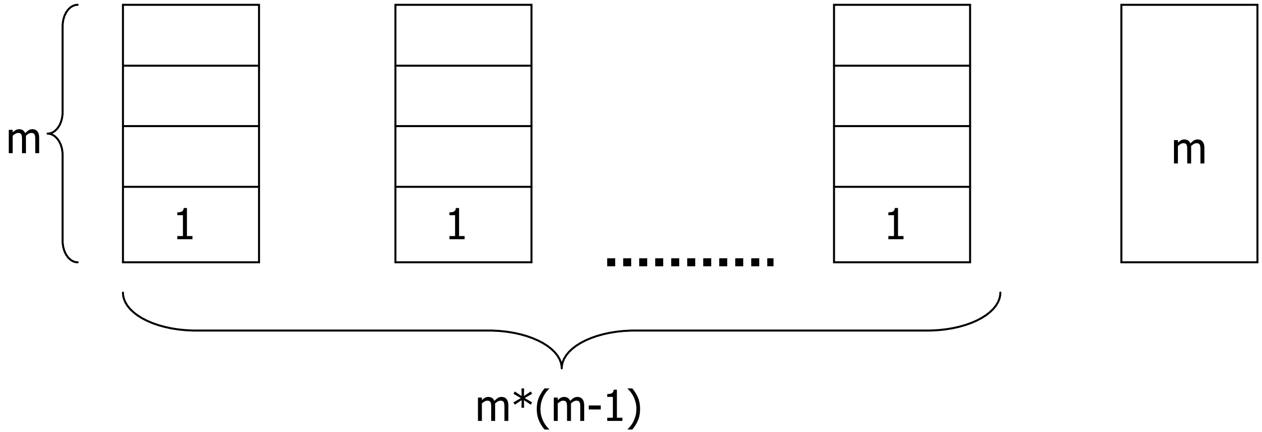
$$\frac{A(I)}{Opt(I)} = 2 - \frac{1}{m}$$

# The approximation is tight




- A(I)=2m-1
- Opt(I)=m

# The approximation is tight




$$\frac{A(I)}{Opt(I)} = \frac{2m-1}{m} = 2 - \frac{1}{m}$$

# Minimum m-Processor Scheduling

- T tasks: $e_1, e_2, \ldots, e_n$
- m fixed: $P_1, \ldots, P_m$ (processors)
- Task placement minimizing makespan (finished time without preemption)
- FPTAS
  - $\mathrm{FPTAS} \subseteq \mathrm{PTAS} \subseteq \mathrm{APX} \subseteq \mathrm{NPO}$

# Multifit (Coffman, Garey, Johnson)

- Order tasks in decreasing execution time
- Compute lower and upper bounds $c_{min}$, $c_{max}$

# Multifit

- Repeat

  $c=(c_{min}+c_{max})/2$

  for i=1 to n (number of tasks)

      place task i on the first possible processor

  if (number of utilized processors)>m

      then $c_{min}=c$

      else $c_{max}=c$

  until $c_{min}=c_{max}$ or number of iterations expired

  end

# (With communications)

- $T=\{T_1, T_2, \ldots, T_n\}$
- $P=\{P_1, P_2, \ldots, P_m\}$
- $G=(V, E)$
- $c_{ij}$ (communication $T_i$ and $T_j$)
- $t_{ip}$ (execution task i on p)

# modeling

$$Min \quad C_{\max}$$

$$\sum_{i=1}^{n} t_{ip} x_{ip} + \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ip} (1 - x_{jp}) \leq C_{\max}, \quad p = 1, \ldots, m$$

$$\sum_{p=1}^{m} x_{ip} = 1, \quad i = 1, \ldots, n$$

$$x_{ip} \in \{0,1\}, \quad i = 1, \ldots, n, \quad p = 1, \ldots, m$$

# Δέντρο Steiner

- $G=(V,E,W)$, $M \subseteq V$
- Tree (steiner) $T=(V',E')$
  $M \subseteq T(V') \subseteq V$, $E' \subseteq E$, $W(E')$ minimum
- $|M|=2$ polynomial
- $|M|=|V|$ polynomial
- NP-hard

# DNH (Distance Network Heuristic)

- Construct $D_G(M)$ graph complete
  - $D_G(M)=(M, E, W)$
  - $W(e)=d([u,v])$ in G (distance)
- Find a Min Spanning Tree $T_1$ in $D_G(M)$
- Replace each e $\subseteq T_1$ by shortest path in G ($T_D$ the new graph)

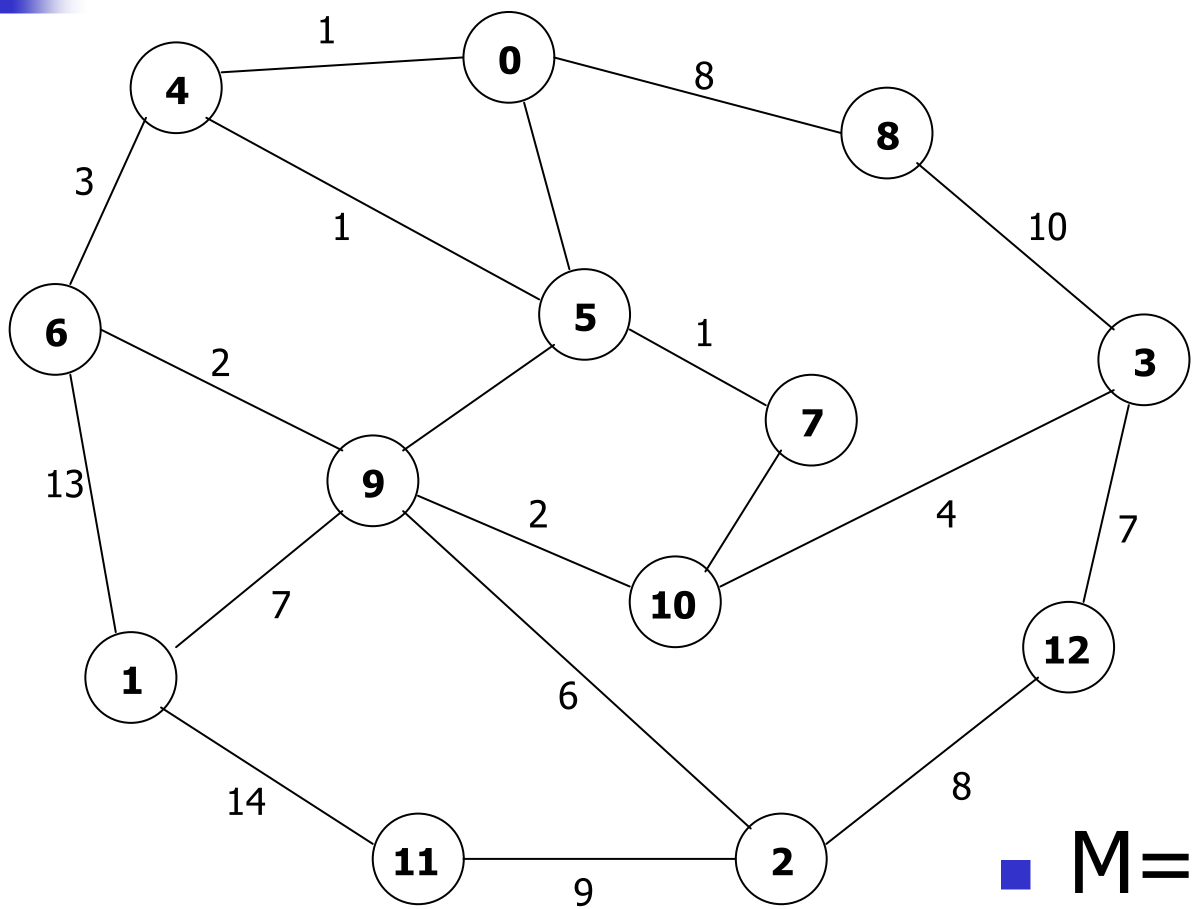
# DNH (Distance Network Heuristic)

- Find a Min Spanning Tree $T_2$ in subgraph of G induced by $T_D$
- Construct $T_{DNH}$ from $T_2$ vertices $v \in V$, $v \notin M$ with $d(v)=1$
- Complexity: $O(|M|\ |V|^2)$
  $O(|M|\ (|E|+|V|\log|V|))$ heap Fibonacci

# DNH (an approximate algorithm)

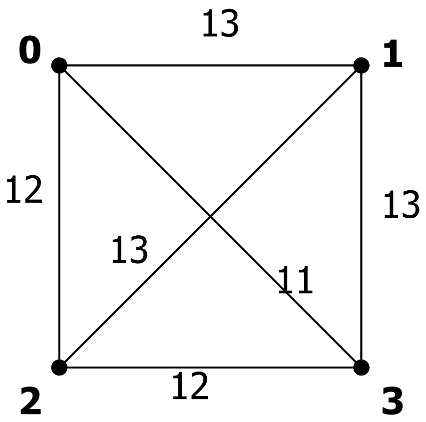
- $\rho$=2 (approximation)
- in practice 5%

# Example

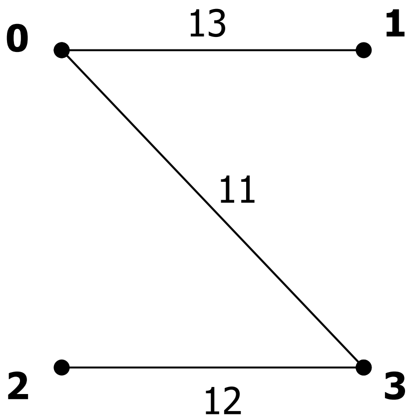



- M={0, 1, 2, 3}
- (9,5)8, (7,10)4

# Example

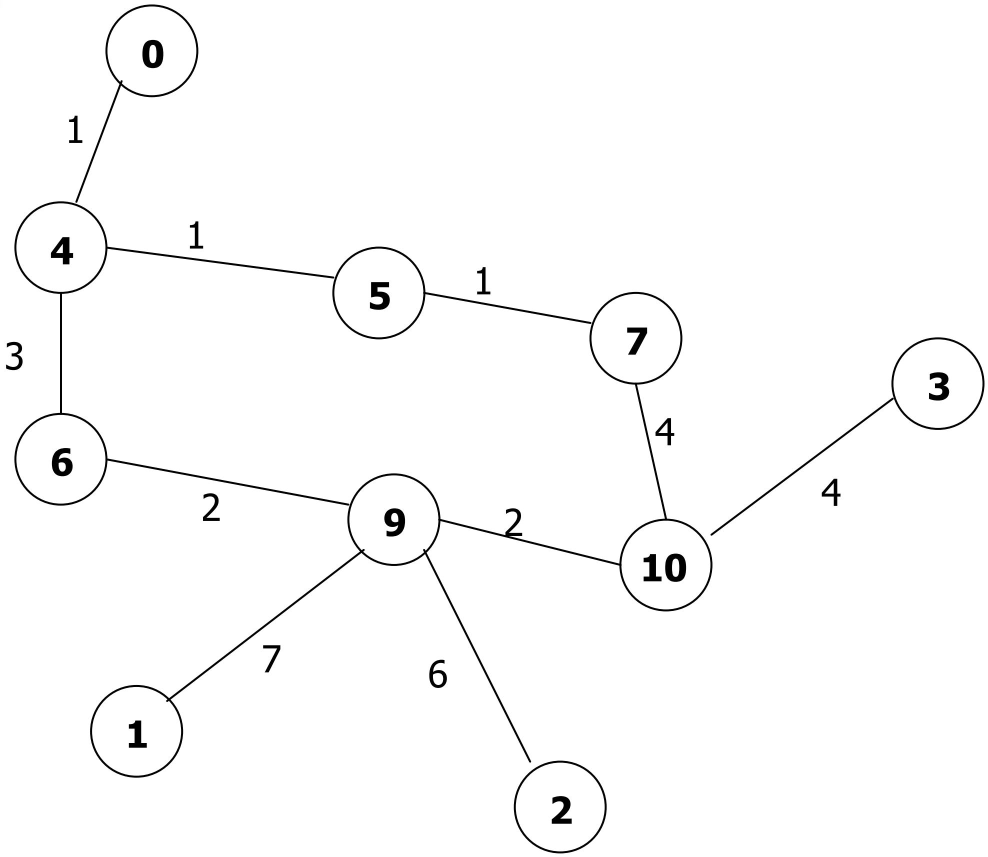



$D_G(M)$




$T_1$: spanning Tree (MIN)

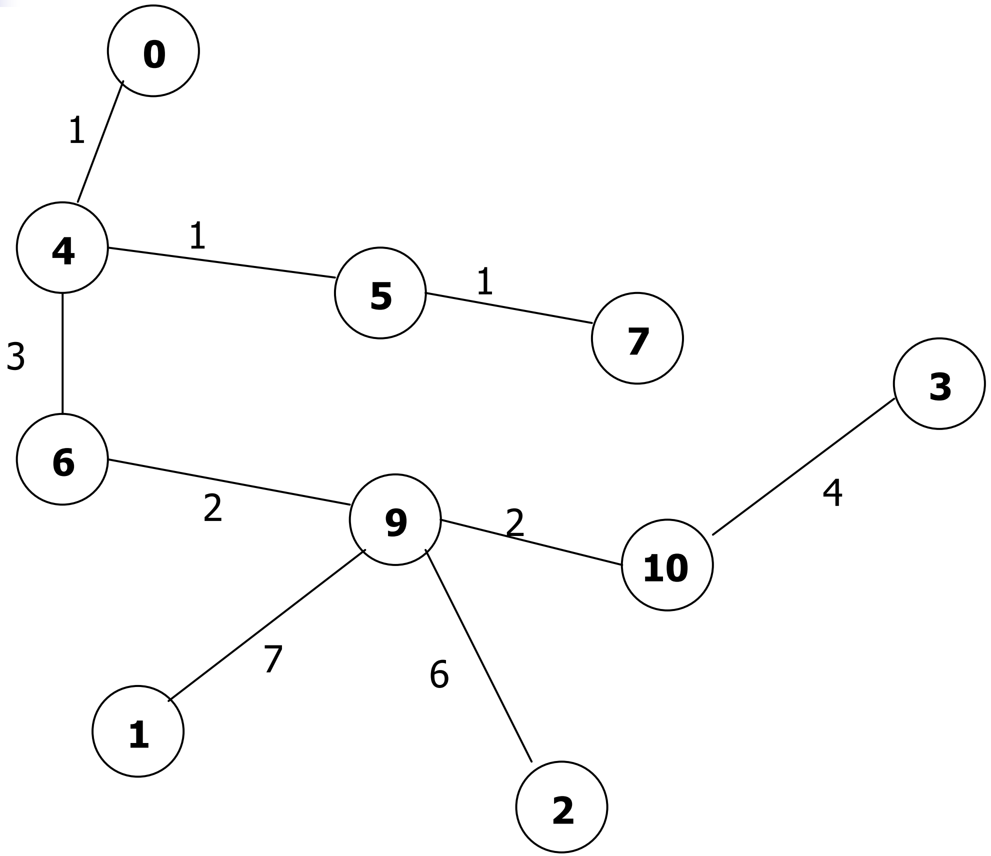
# Example




- 0 4 6 9 1
- 0 4 5 7 10 3
- 3 10 9 2

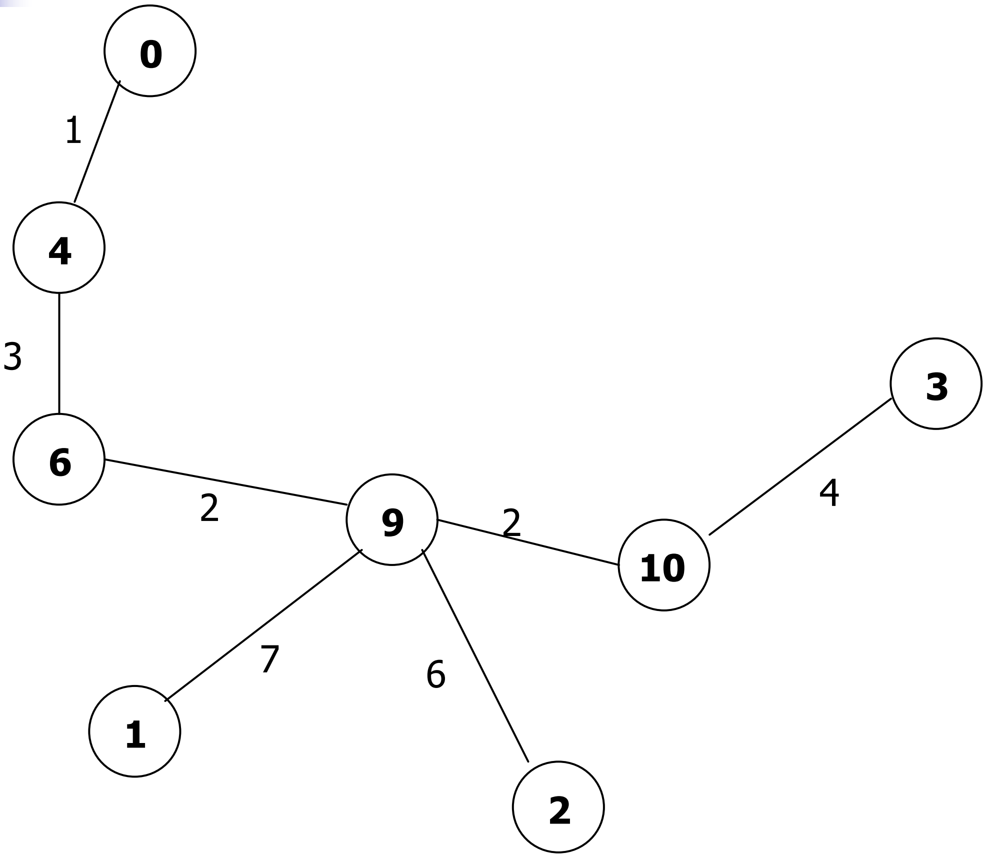
- graph $T_D$

# Example




- Min spanning tree $T_2$

# Final step




- d(7)=1
- d(5)=1

- Final tree: $T_{DNH}$