Missing Roads

Problem

- Given: undirected graph G and an integer k
- Find: set of k cities $\{v_1, v_2, \dots, v_k\}$ and edges $\{e_1, e_2, \dots, e_k\}$ such that
  a) each city $\boldsymbol{v_i}$ is **adjacent** to the edge $\boldsymbol{e_i}$
  b) the sum

$$\sum_{i=1}^{k} \boldsymbol{cost(e_i)}$$

is the smallest possible

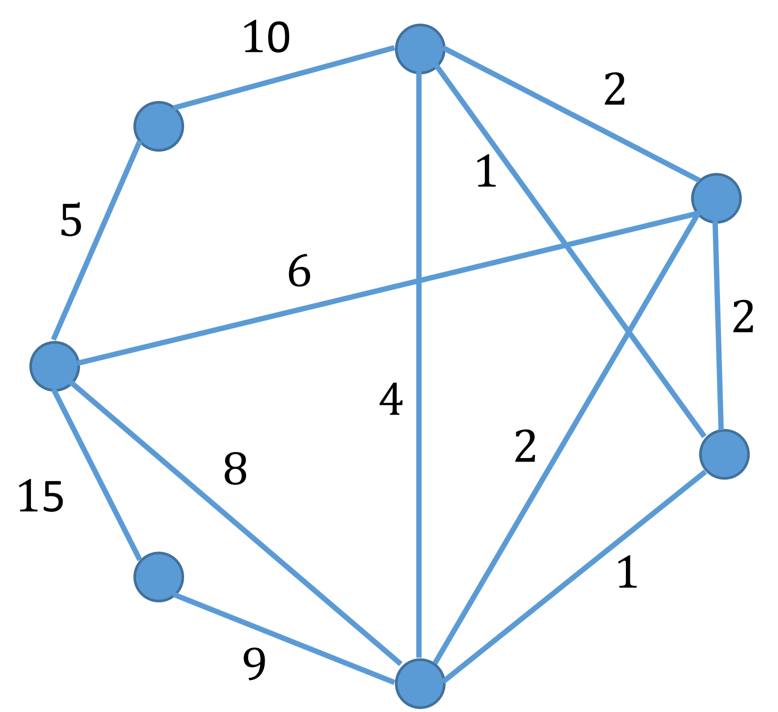
Problem

- Given: undirected graph G and an integer k
- Find: set of k cities $\{v_1, v_2, \dots, v_k\}$ and edges $\{e_1, e_2, \dots, e_k\}$ such that
  a) each city $\boldsymbol{v_i}$ is **adjacent** to the edge $\boldsymbol{e_i}$
  b) the sum

$$\sum_{i=1}^{k} \boldsymbol{cost(e_i)}$$

is the smallest possible

$k = 5$



Problem

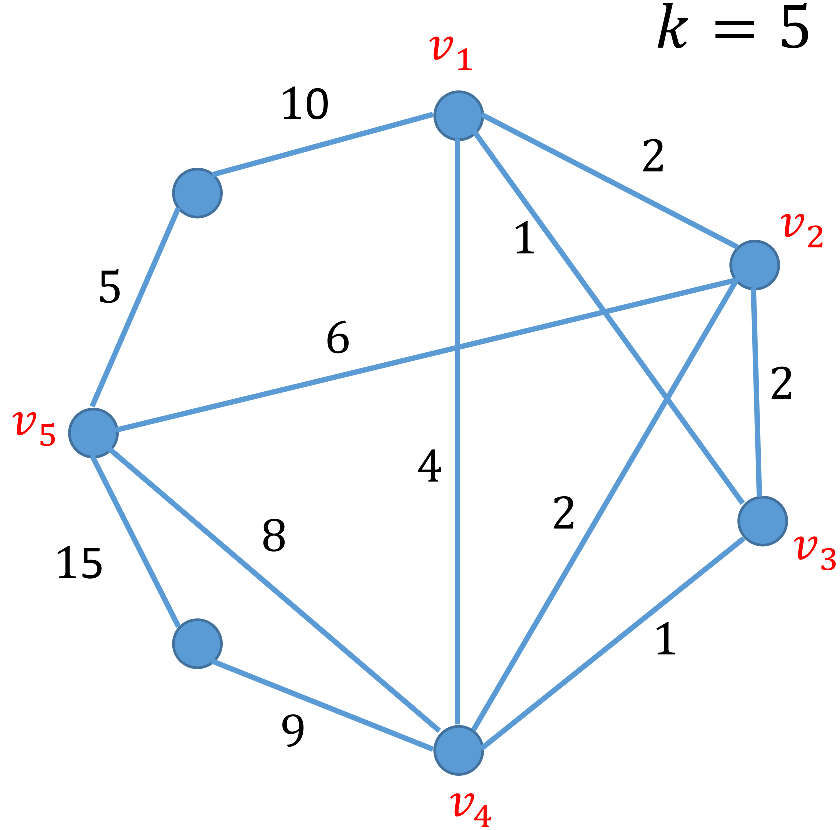
- Given: undirected graph G and an integer k

- Find: set of k cities $\{v_1, v_2, \dots, v_k\}$ and edges $\{e_1, e_2, \dots, e_k\}$ such that
  a) each city v_i is **adjacent** to the edge e_i
  b) the sum

$$\sum_{i=1}^{k} cost(e_i)$$

is the smallest possible

$k = 5$

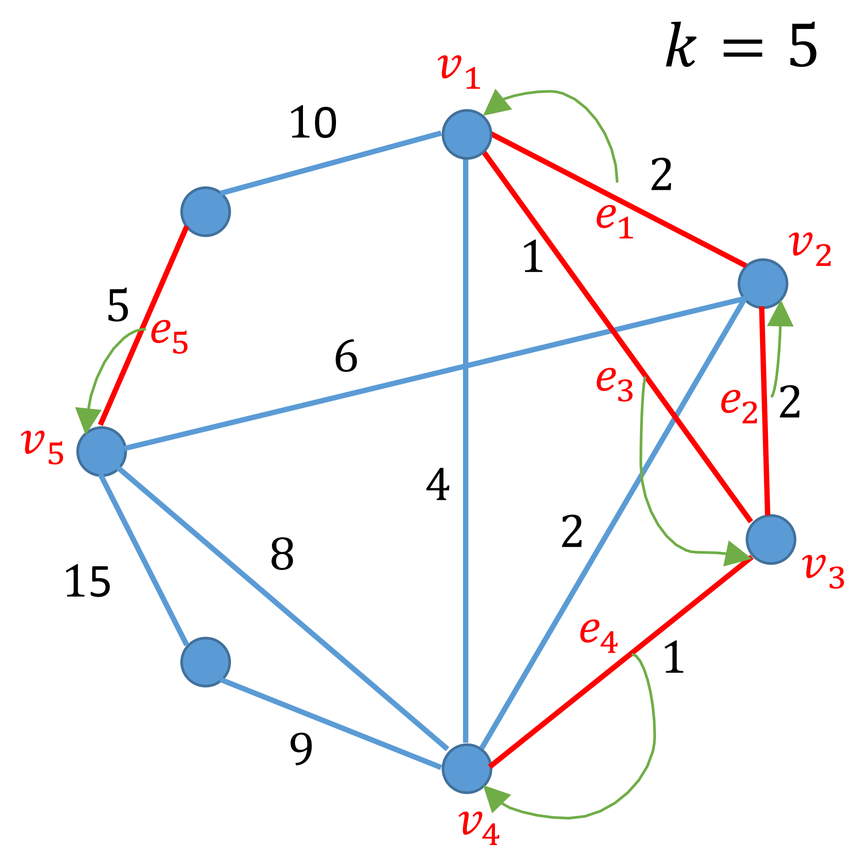


Problem

- Given: undirected graph G and an integer k
- Find: set of k cities $\{v_1, v_2, \dots, v_k\}$ and edges $\{e_1, e_2, \dots, e_k\}$ such that
  a) each city $\boldsymbol{v_i}$ is **adjacent** to the edge $\boldsymbol{e_i}$
  b) the sum

$$\sum_{i=1}^{k} \boldsymbol{cost(e_i)}$$

is the smallest possible

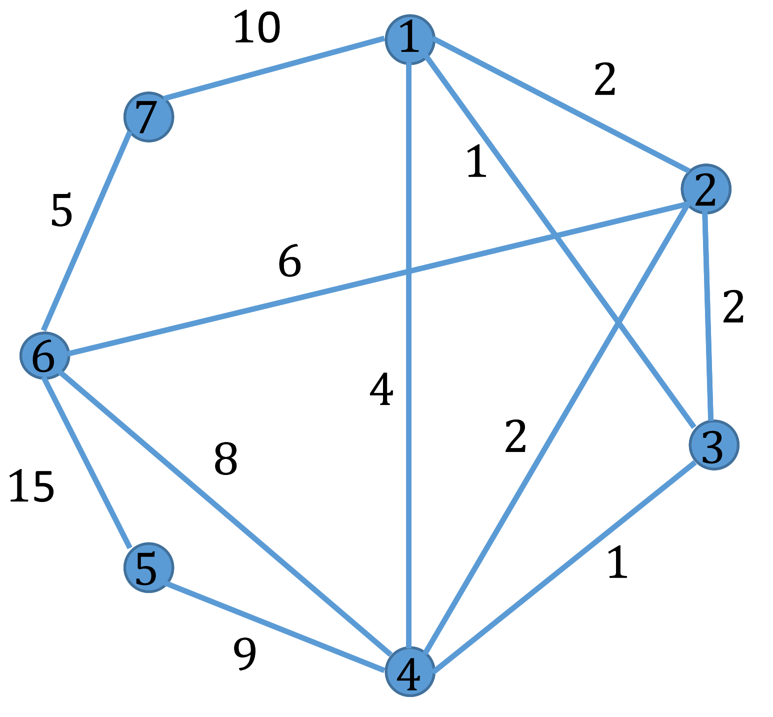


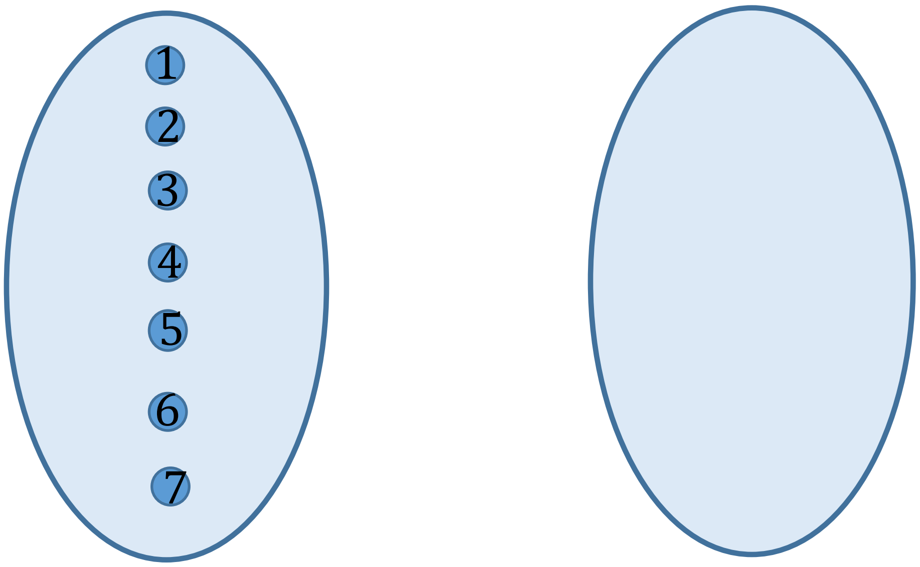
Solution

- Create auxiliary bipartite graph B:
  - one side – vertices of G
  - the other side – edges of G (now as vertices in B)
  - edge between vertex v and an edge e if they are adjacent in G

Solution

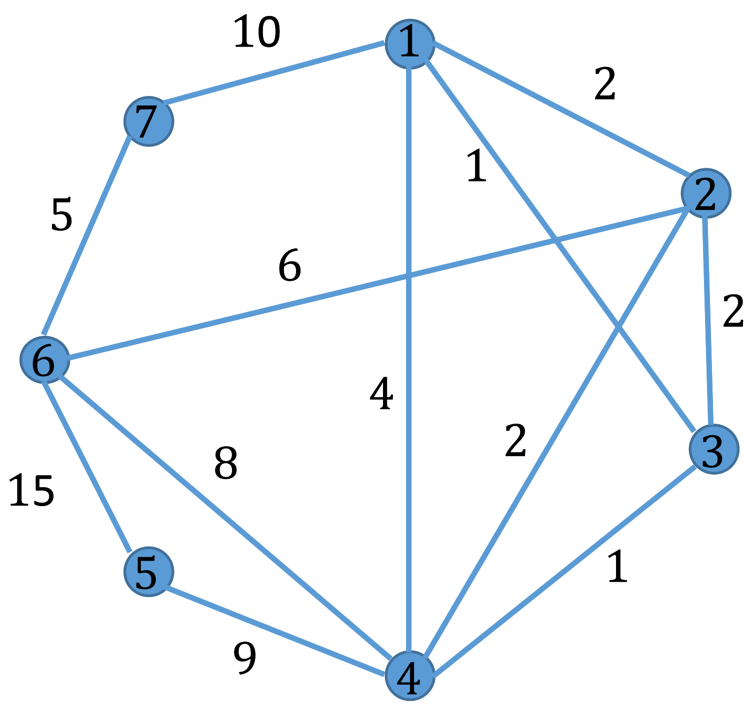
- Create auxiliary bipartite graph B:
  - one side – vertices of G
  - the other side – edges of G (now as vertices in B)
  - edge between vertex v and an edge e if they are adjacent in G

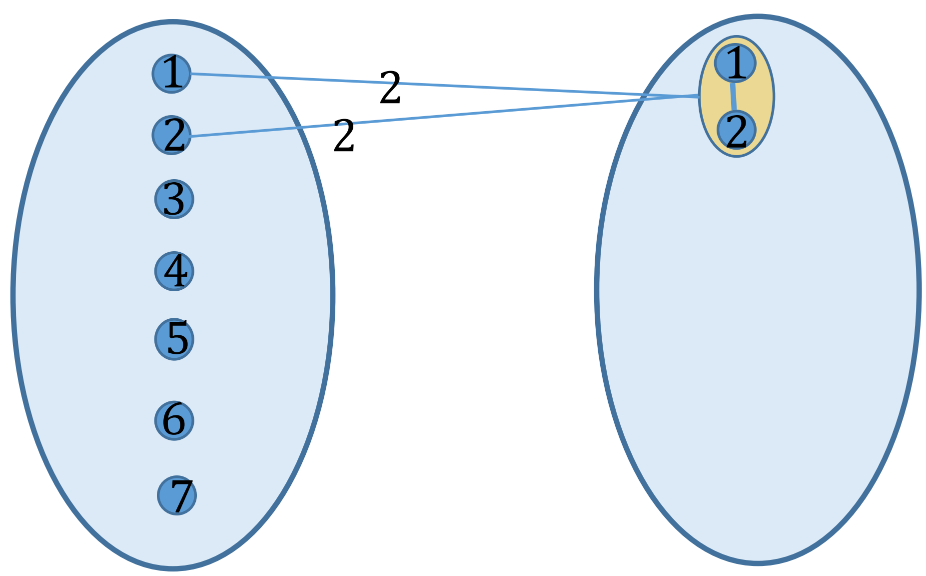




Solution

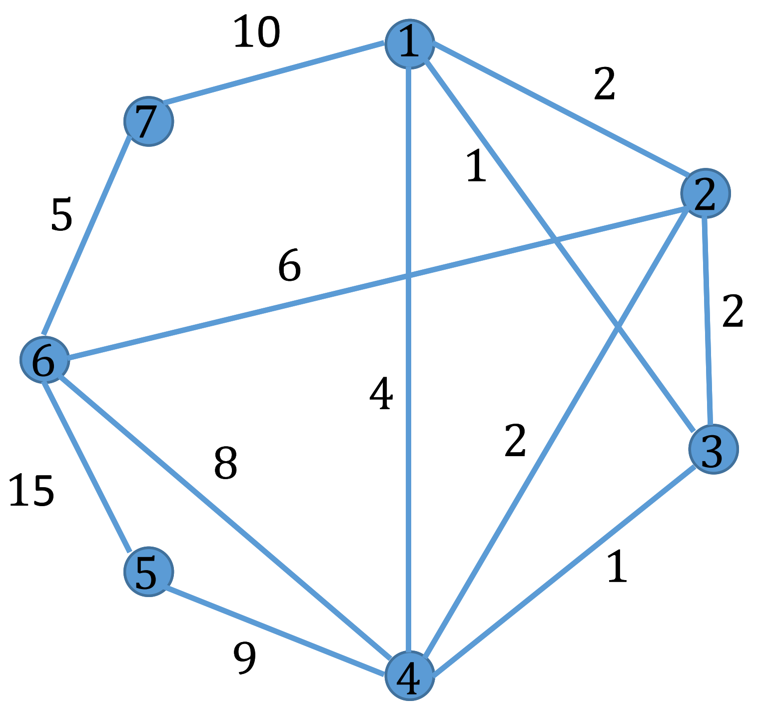
- Create auxiliary bipartite graph B:
  - one side – vertices of G
  - the other side – edges of G (now as vertices in B)
  - edge between vertex v and an edge e if they are adjacent in G

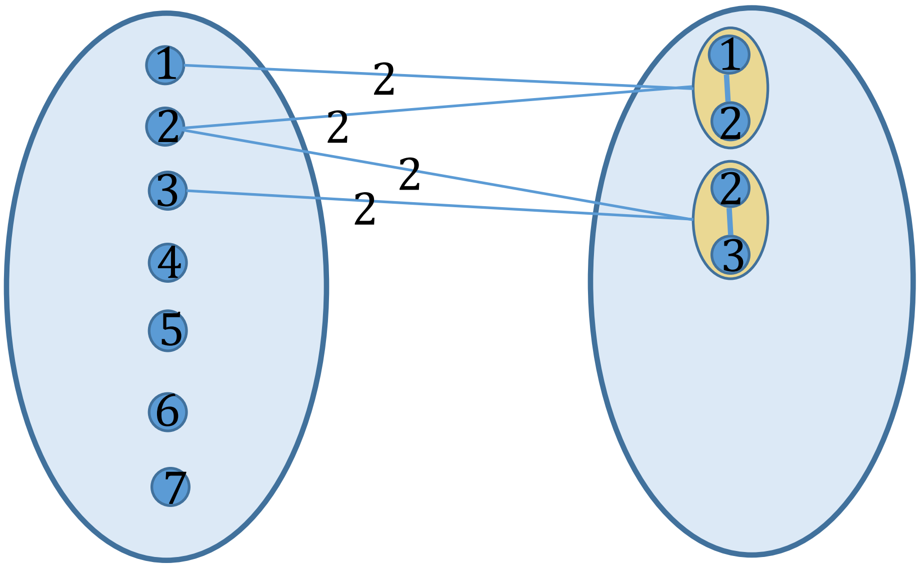




Solution

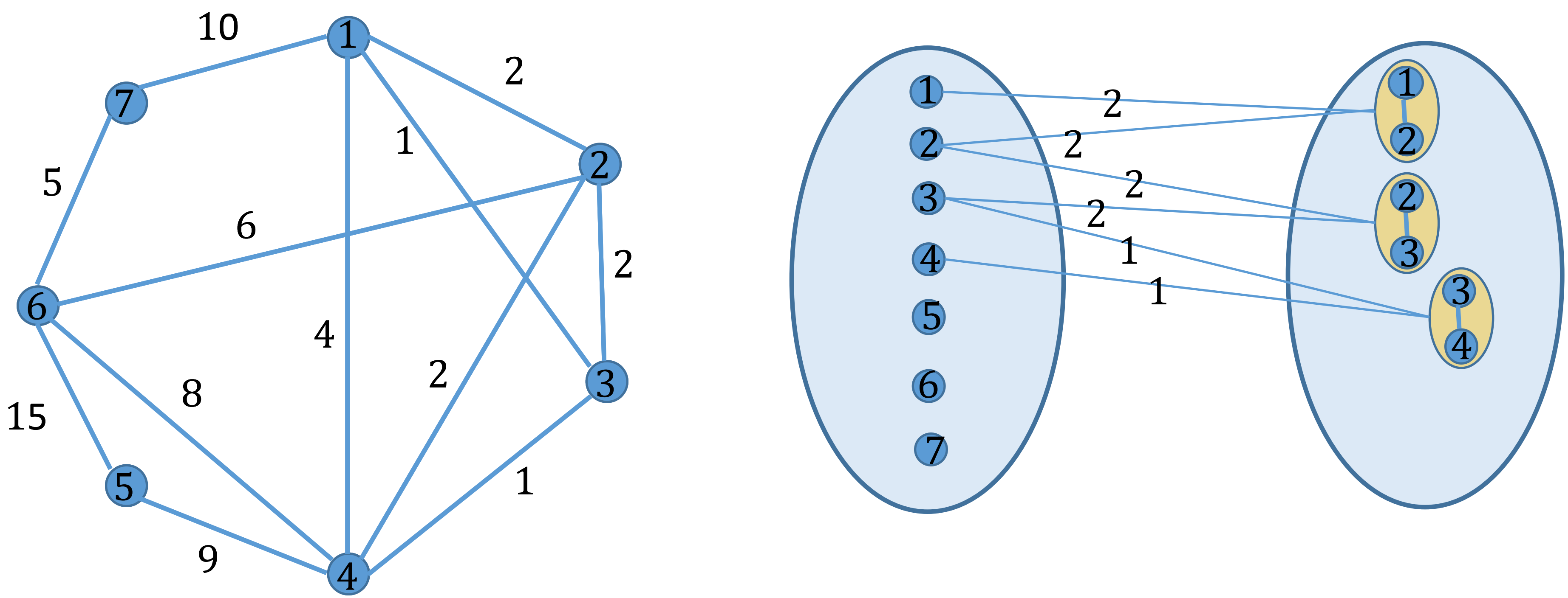
- Create auxiliary bipartite graph B:
  - one side – vertices of G
  - the other side – edges of G (now as vertices in B)
  - edge between vertex v and an edge e if they are adjacent in G





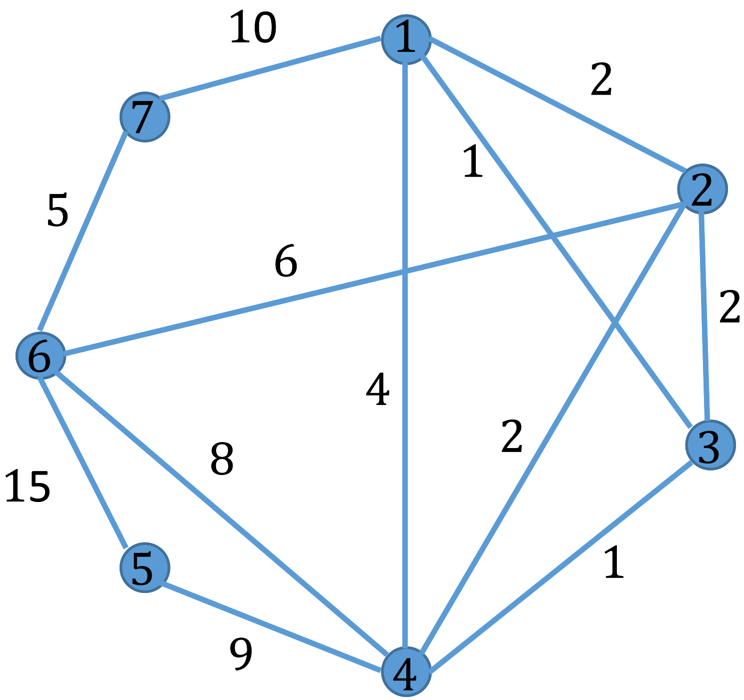
Solution

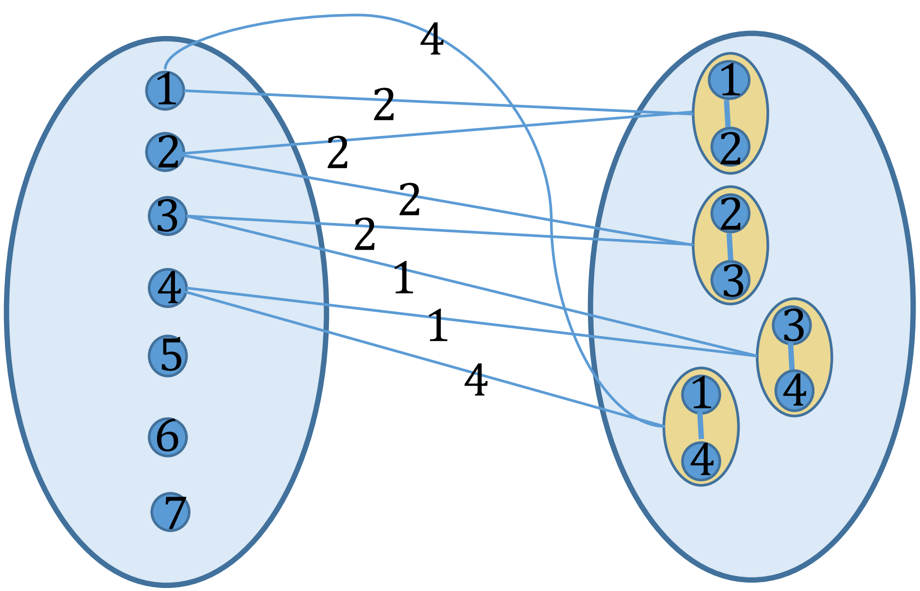
- Create auxiliary bipartite graph B:
    - one side – vertices of G
    - the other side – edges of G (now as vertices in B)
    - edge between vertex v and an edge e if they are adjacent in G



Solution

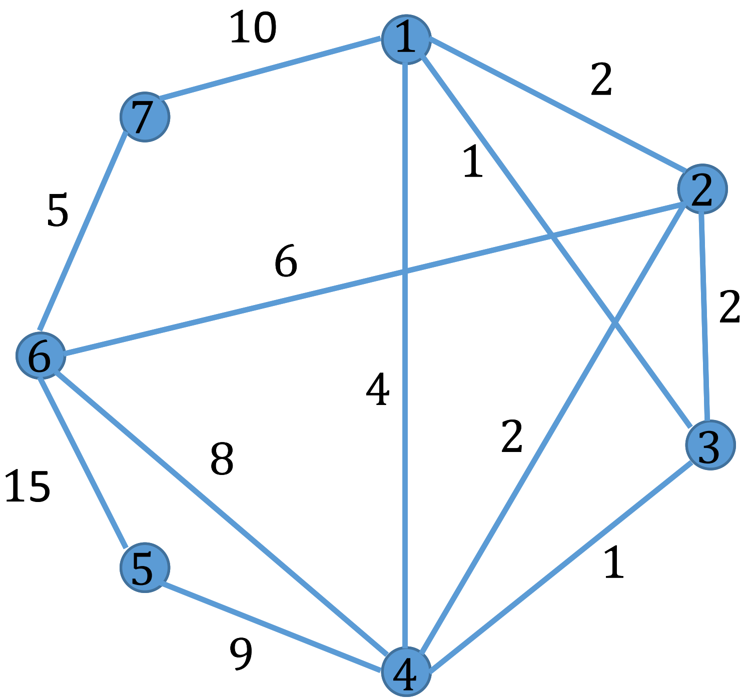
- Create auxiliary bipartite graph B:
  - one side – vertices of G
  - the other side – edges of G (now as vertices in B)
  - edge between vertex v and an edge e if they are adjacent in G

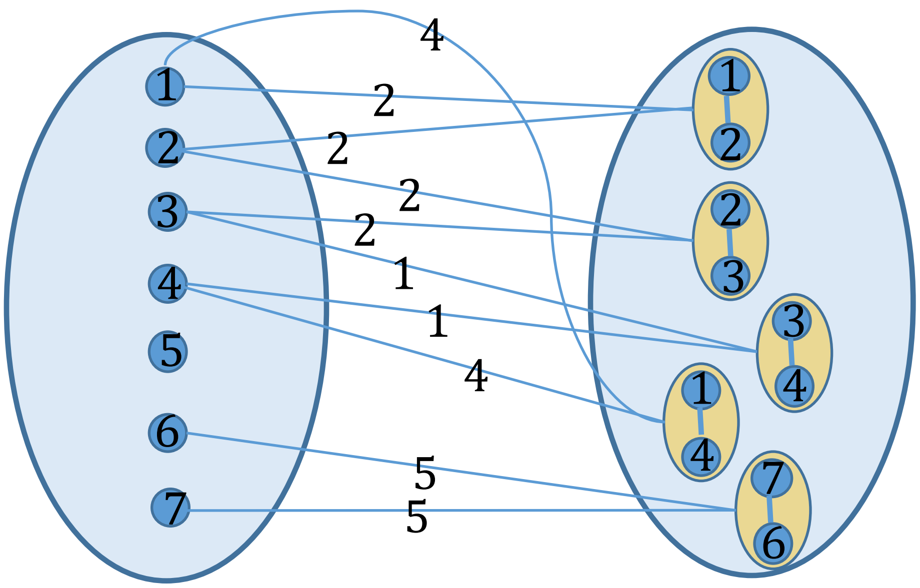




Solution

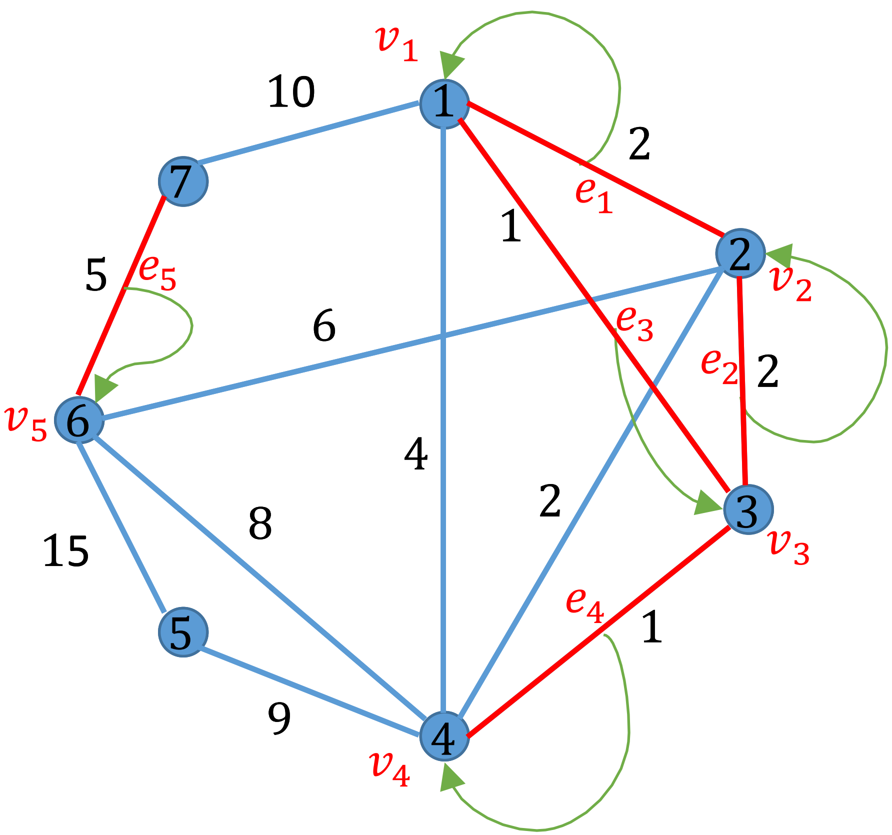
- Create auxiliary bipartite graph B:
  - one side – vertices of G
  - the other side – edges of G (now as vertices in B)
  - edge between vertex v and an edge e if they are adjacent in G

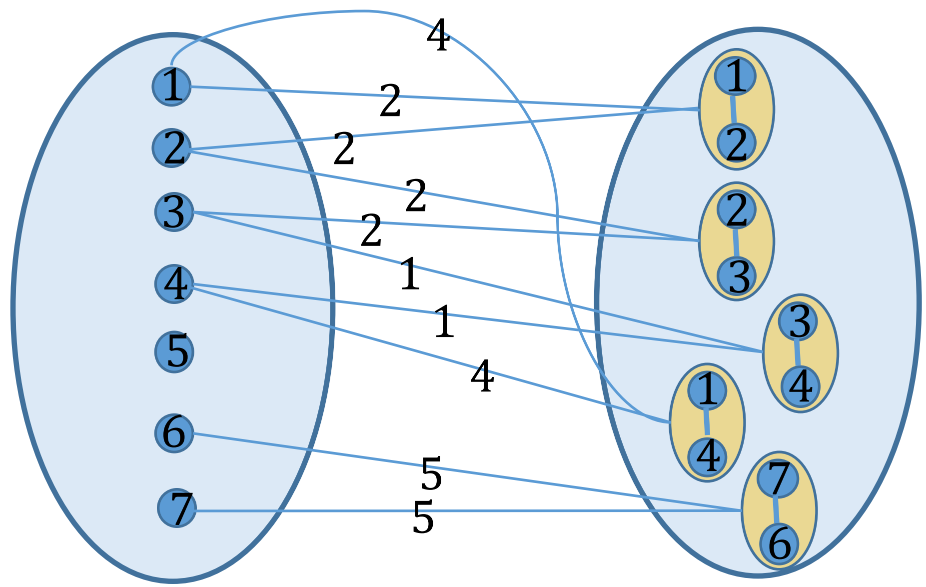




Solution

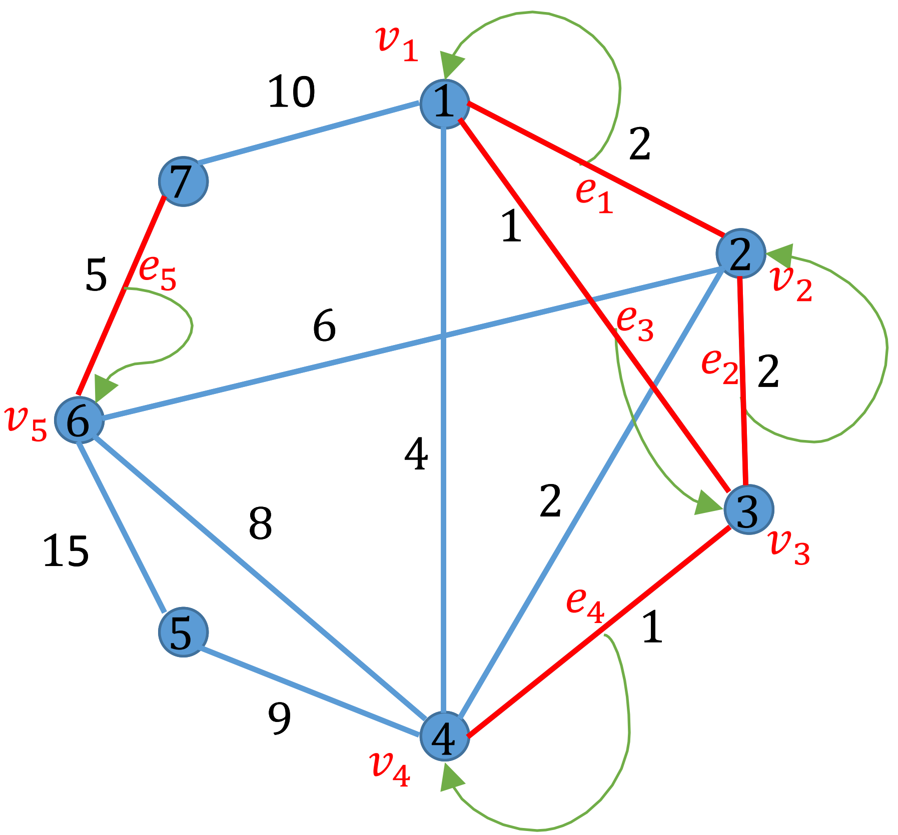
- Create auxiliary bipartite graph B:
  - one side – vertices of G
  - the other side – edges of G (now as vertices in B)
  - edge between vertex v and an edge e if they are adjacent in G

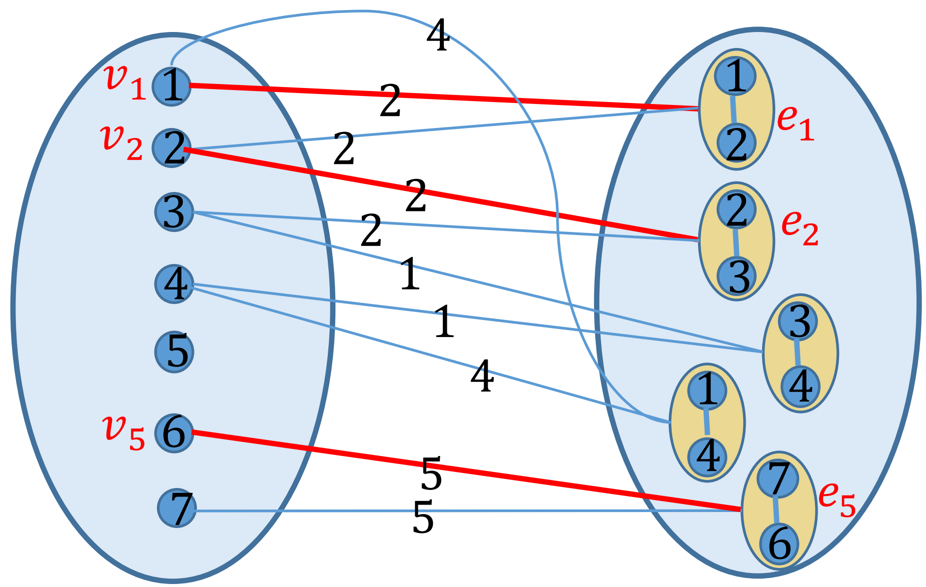




Solution

- Create auxiliary bipartite graph B:
  - one side – vertices of G
  - the other side – edges of G (now as vertices in B)
  - edge between vertex v and an edge e if they are adjacent in G





Solution

- Create auxiliary bipartite graph B:
  - one side – vertices of G
  - the other side – edges of G (now as vertices in B)
  - edge between vertex v and an edge e if they are adjacent in G

- Find a cheapest matching of size k
  1. If all costs are the same = check if the maximum matching is of size at least k

Solution

- Create auxiliary bipartite graph B:
  - one side – vertices of G
  - the other side – edges of G (now as vertices in B)
  - edge between vertex v and an edge e if they are adjacent in G

- Find a cheapest matching of size k
  1. If all costs are the same = check if the maximum matching is of size at least k
  2. Otherwise…

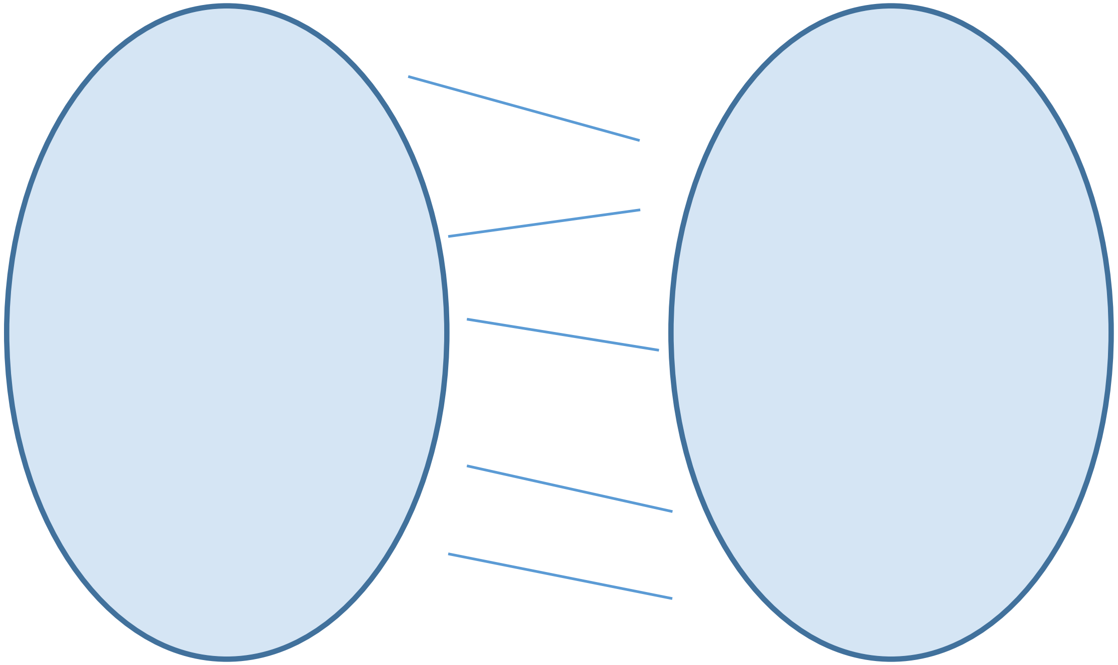
Solution

- Create auxiliary bipartite graph B:
  - one side – vertices of G
  - the other side – edges of G (now as vertices in B)
  - edge between vertex v and an edge e if they are adjacent in G

- Find a cheapest matching of size k
  1. If all costs are the same = check if the maximum matching is of size at least k
  2. Otherwise… reduce the problem to the **min-cost max-flow**

Solution

- Find a cheapest matching of size k
  1. If all costs are the same = check if the maximum matching is of size at least k
  2. Otherwise… reduce the problem to the **min-cost max-flow**



Solution

- Find a cheapest matching of size k
  1. If all costs are the same = check if the maximum matching is of size at least k
  2. Otherwise… reduce the problem to the **min-cost max-flow**