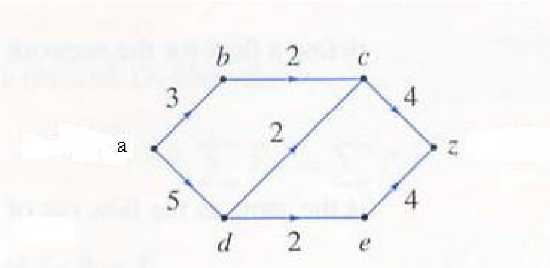
Chapter 3 Network Theory

3-1 Network and Flow

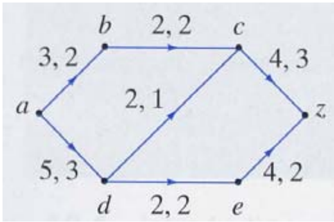


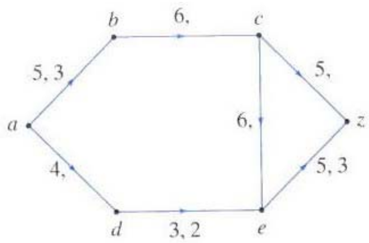
Network: A simple, weighted, directed graph satisfies: (a) The source has no incoming edges. (b) The sink has no outgoing edges, (c) The weight C_{ij} of the directed edge (i,j) is a nonnegative number. C_{ij} is called the capacity of (i,j).

Flow: A flow F_{ij} of the directed edge (i,j) is a nonnegative number and satisfies:

(a) $F_{ij} \leq C_{ij}$. (b) $\sum_i F_{ij} = \sum_i F_{ji}$ for each vertex j, neither the source and the sink..

Eg. A network with edges label by capacity (left) and flow (right).

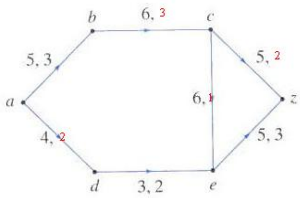


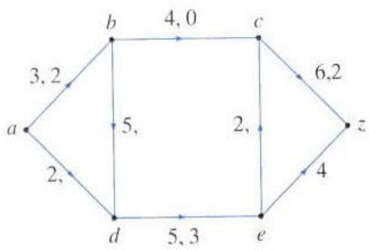


Eg. Fill in the missing edge flows of the left network.

(Sol.) F_{bc}=F_{ab}=3, F_{ad}=F_{de}=2

F_{ce}=F_{ez} -F_{de}=3-2=1, F_{cz}= F_{bc}-F_{ce}=3-1=2

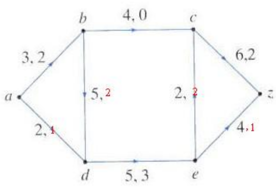




Eg. Fill in the missing edge flows of the left network.

(Sol.) F_{bd}=F_{ab}–F_{bc}=2, F_{ec}=F_{cz}-F_{bc}=2, F_{ad}= F_{de}-F_{bd}=1

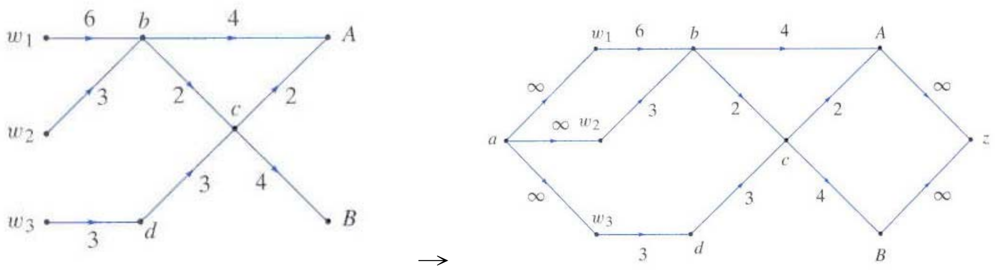
F_{ez}= F_{de}-F_{ec}=1



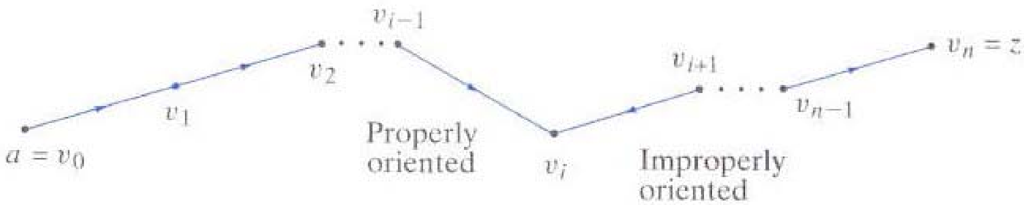
Theorem **The flow out of the source equals the flow into the sink. That is,** $\sum_i F_{ai} = \sum_i F_{iz}$.

Value of the flow: The value of $\sum_i F_{ai} = \sum_i F_{iz}$.

Supersource and supersink: To be added in the original network without the source and the sink.



Properly oriented path and improperly oriented path:

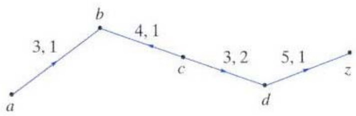


Path P

Theorem **Let P be a path from a to z in a network G. Let $\triangle$=min(C_{ij} - F_{ij} for properly oriented edges (i,j), F_{ij} for improperly oriented edges (i,j)). Define**

$$F_{ij}^* = \begin{cases} F_{ij}, & \text{if } (i,j) \text{ is not in } P \\ F_{ij} + \Delta, & \text{if } (i,j) \text{ is properly oriented in } P \\ F_{ij} - \Delta, & \text{if } (i,j) \text{ is improperly oriented in } P \end{cases}$$

, and then $F^* > F$.

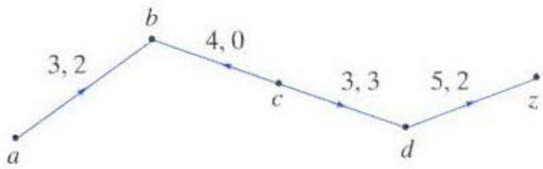


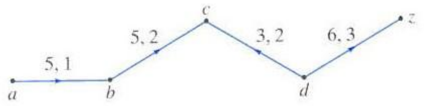
Eg. Increase the flow of each edge in the left path.

(Sol.) $\triangle$=min(3-1,1,3-2,5-1)=1

New flows: 1+1=2, 1-1=0, 2+1=3, 1+1=2

We have the new flows in the path:

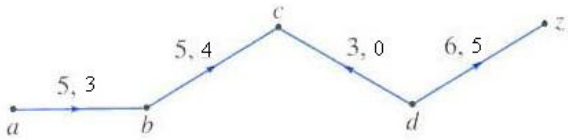




Eg. Increase the flow of each edge in the left path.

(Sol.) $\triangle$=min(5-1,5-2,2,6-3)=2

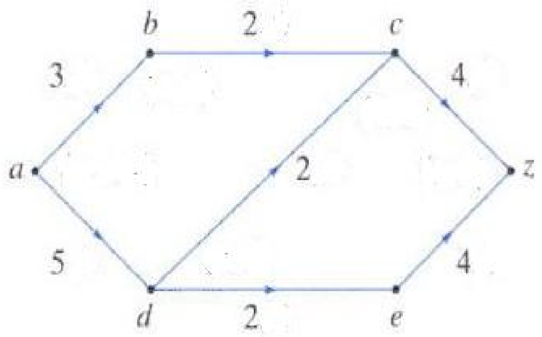
New flows: 1+2=3, 2+2=4, 2-2=0, 3+2=5. We have the new flows in the path:



Maximal flow algorithm

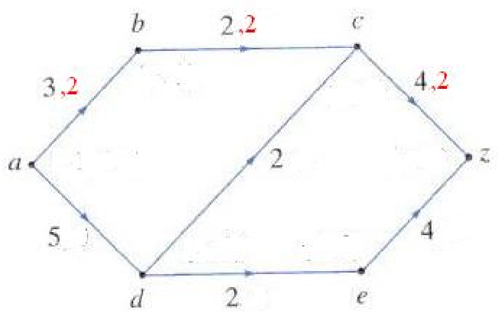
```
Input:  A network with source a, sink z, capacity C,
        vertices a = v0, ..., vn = z, and n
Output: A maximal flow F

max_flow(a, z, C, v, n) {
  // v's label is (predecessor(v), val(v))
  // start with zero flow
  for each edge (i, j)
    Fij = 0
  while (true) {
    // remove all labels
    for i = 0 to n {
      predecessor(vi) = null
      val(vi) = null
    }
    // label a
    predecessor(a) = —
    val(a) = ∞
    // U is the set of unexamined, labeled vertices
    U = {a}
    // continue until z is labeled
    while (val(z) == null) {
      if (U == ∅) // flow is maximal
        return F
      choose v in U
      U = U − {v}
      Δ = val(v)
      for each edge (v, w) with val(w) == null
        if (Fvw < Cvw) {
          predecessor(w) = v
          val(w) = min{Δ, Cvw − Fvw}
          U = U ∪ {w}
        }
      for each edge (w, v) with val(w) == null
        if (Fwv > 0) {
          predecessor(w) = v
          val(w) = min{Δ, Fwv}
          U = U ∪ {w}
        }
    } // end while (val(z) == null) loop
    // find path P from a to z on which to revise flow
    w0 = z
    k = 0
    while (wk ¬= a) {
      wk+1 = predecessor(wk)
      k = k + 1
    }
    P = (wk+1, wk, ..., w1, w0)
    Δ = val(z)
    for i = 1 to k + 1 {
      e = (wi, wi−1)
      if (e is properly oriented in P)
        Fe = Fe + Δ
      else
        Fe = Fe − Δ
    }
  } // end while (true) loop
}
```

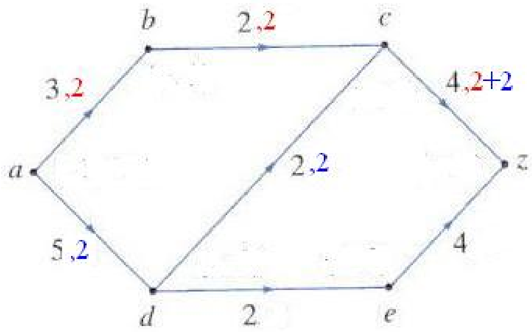


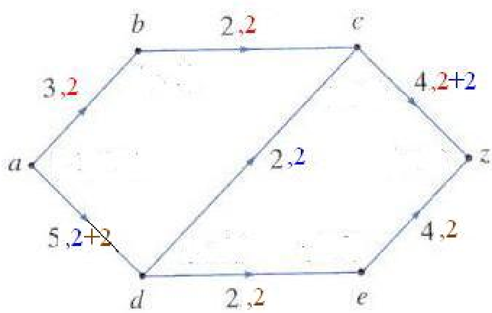
Eg. Find the maximum flow of the left path. The capacity C_{ij} of each edge (i,j) has been labeled on the network.

(Sol.)

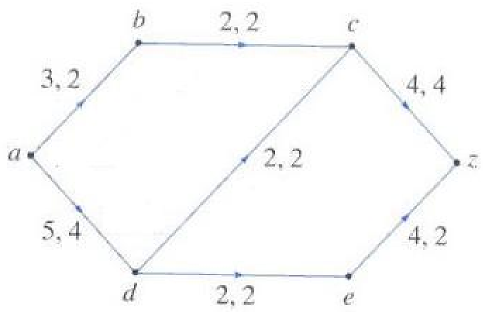


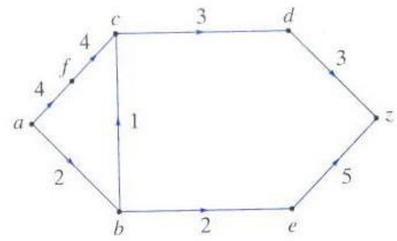
→





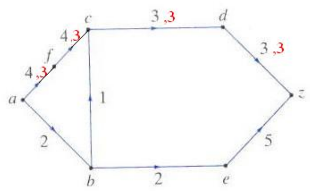
→



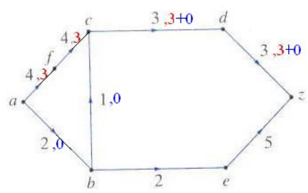


Eg. Find the maximum flow of the left path. The capacity C_{ij} of each edge (i,j) has been labeled on the network.

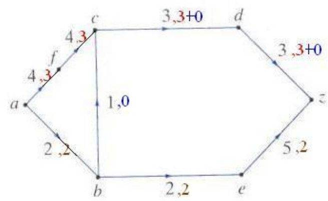
(Sol.)



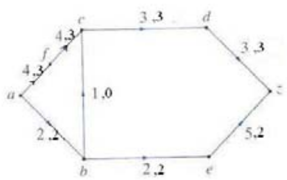
→

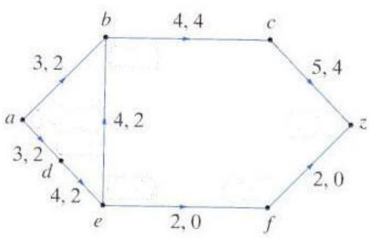


→



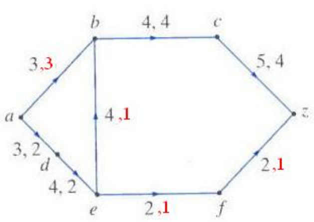
→



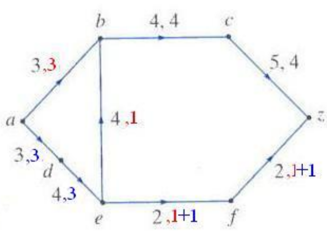


Eg. Find the maximum flow of the left path. The capacity C_{ij} of each edge (i,j) has been labeled on the network.

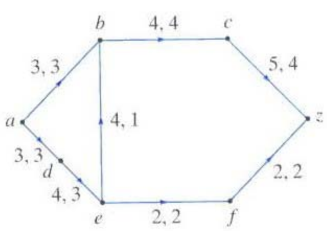
(Sol.)



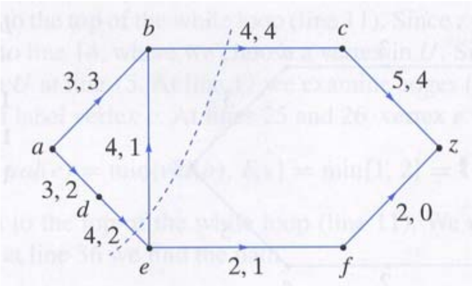
→



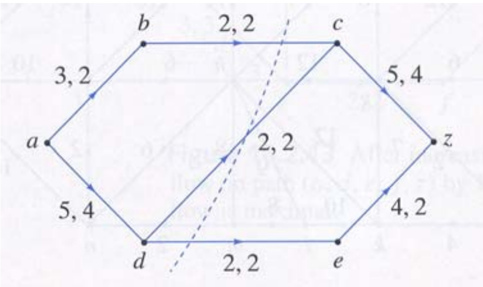
→



Cut $(P,\overline{P})$: A cut $(P,\overline{P})$ in G consists of a set P of vertices and the complement $\overline{P}$ of P, with $a \in P$ and $z \in \overline{P}$.

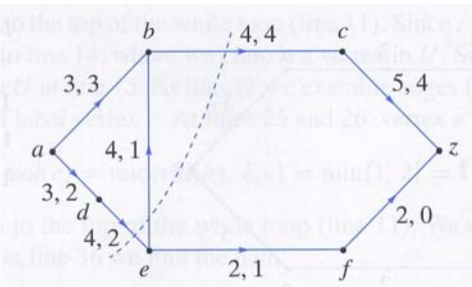


Eg. A cut ($P,\overline{P}$) in the left network, where P={a,b,d} and $\overline{P}$={c,e,f,z}.



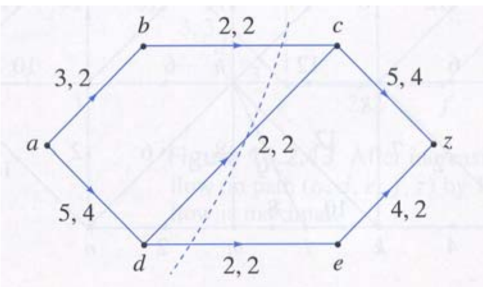
Eg. A cut ($P,\overline{P}$) in the left network, where P={a,b,d} and $\overline{P}$={c,e,z}.

Capacity of the cut ($P,\overline{P}$), $C(P,\overline{P})$: $C(P,\overline{P})=\sum_{i\in P}\sum_{j\in \overline{P}} C_{ij}$



Eg. Find the capacity of the cut ($P,\overline{P}$) in the left network.

(Sol.) $C_{bc}+C_{de}=4+4=8$



Eg. Find the capacity of the cut ($P,\overline{P}$) in the left network.

(Sol.) $C_{bc}+C_{dc}+C_{de}=2+2+2=6$

Theorem $\sum_{i\in P}\sum_{j\in \overline{P}} C_{ij} \ge \sum_{i} F_{ai}$.

Max flow and min cut Theorem If equality holds in $\sum_{i\in P}\sum_{j\in \overline{P}} C_{ij} \ge \sum_{i} F_{ai}$, then the flow is maximal and the cut is minimal. Moreover, equality holds in $\sum_{i\in P}\sum_{j\in \overline{P}} C_{ij} \ge \sum_{i} F_{ai}$ if and only if (a) $F_{ij}=C_{ij}$ for $i\in P$ and $j\in \overline{P}$ and (b) $F_{ij}=0$ for $i\notin P$ and $j\notin \overline{P}$.

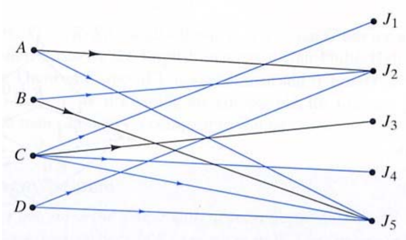
3-2 Matching

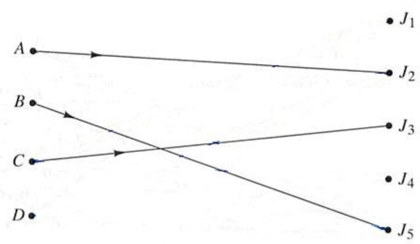
Matching: Let G be a directed, bipartite graph with disjoint vertex set V and W in which the edges are directed from vertices in V to vertices in W. A matching for G is a set of edges E with no vertices in common.

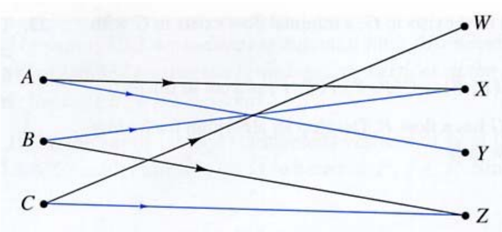
Maximal matching: A matching contains the maximum number of edges.

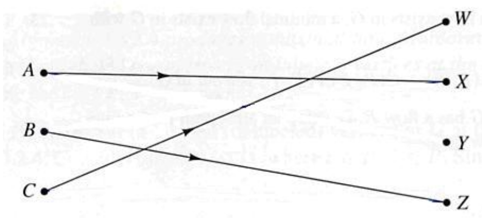
Complete matching: A matching having the property that if $v \in V$, then $(v,w) \in E$ for some $w \in W$.

Eg. Two examples of matching. The black lines show maximal matching in each graph.

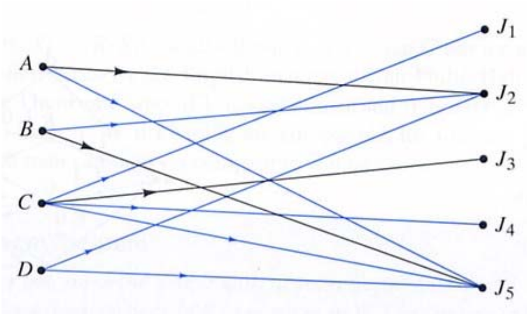


→Maximal matching: 



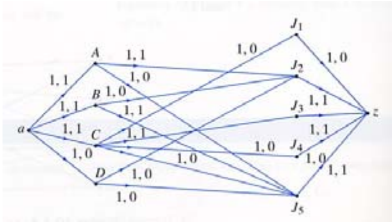
→Maximal matching: 

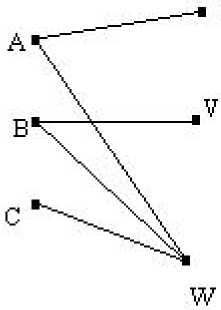
Matching network: Introducing a super source a and edges of capacity 1 from a to each of $v_i \in V$, a super sink z and edges of capacity 1 from each of $w_j \in W$ to z.



Eg. Transform the left matching for *G* into a matching network.

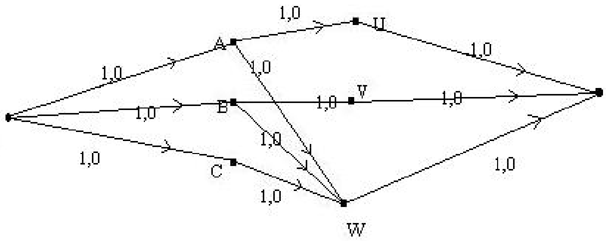
(Sol.)

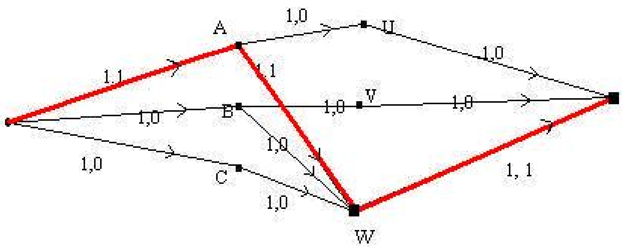




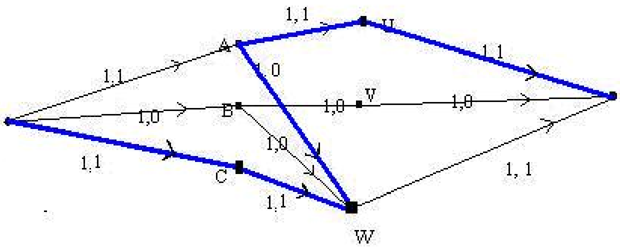
Eg. Find the maximal matching for the left graph.

(Sol.) Matching network:

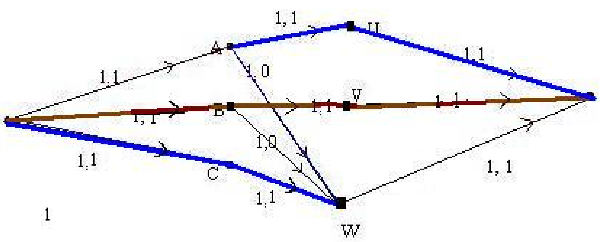




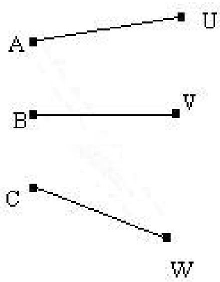
→



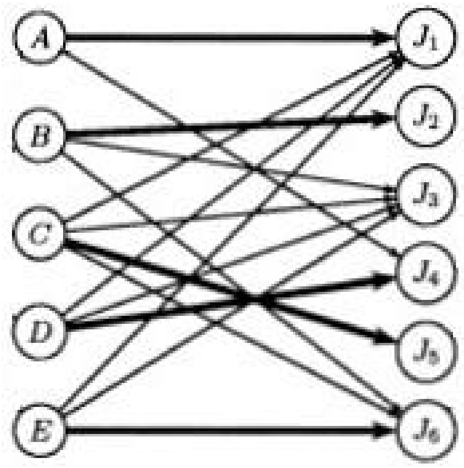
→



→

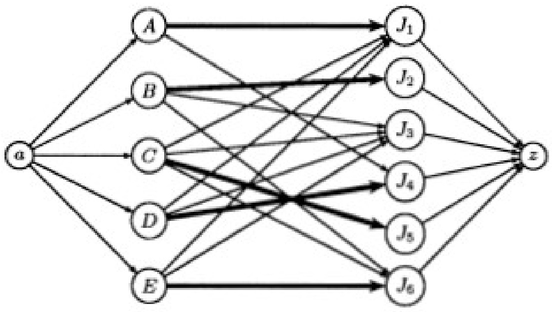


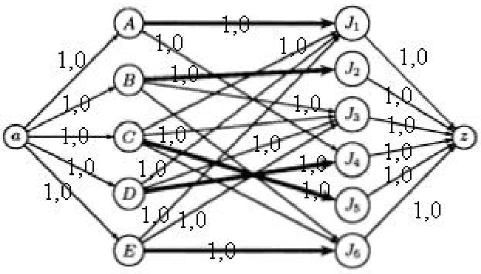
→

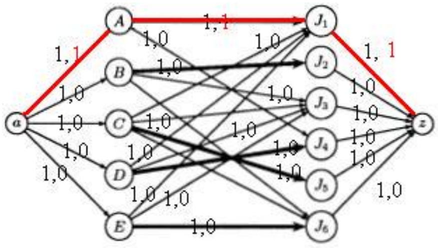


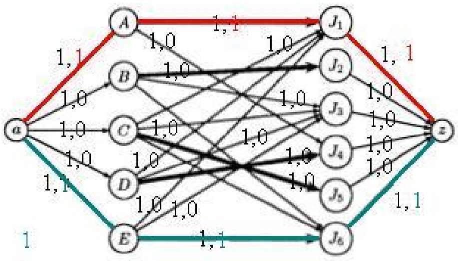
Eg. Find the maximal matching for the left graph.

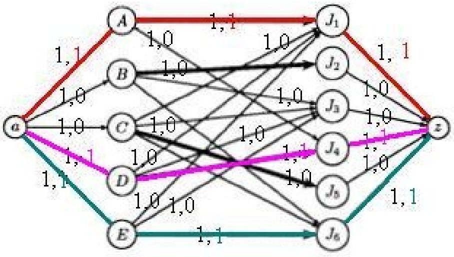
(Sol.) Matching network:

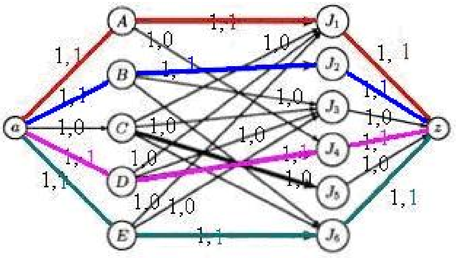


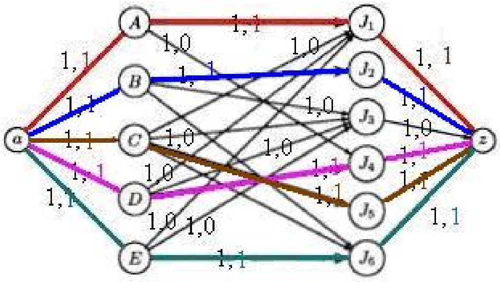
→ 

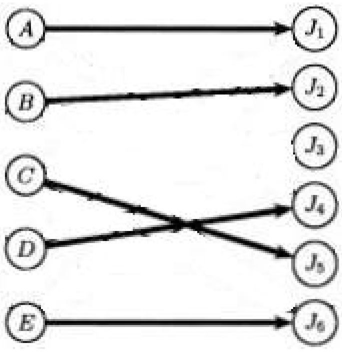
→ 

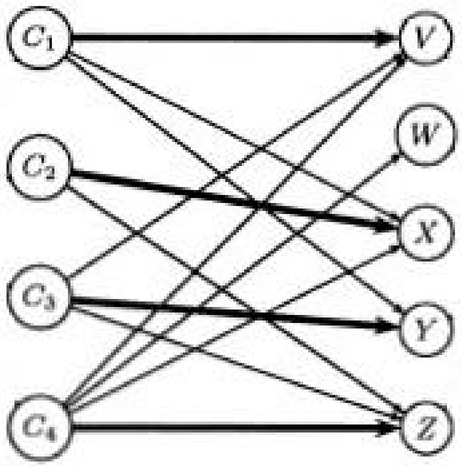
→ 

→ 

→ 

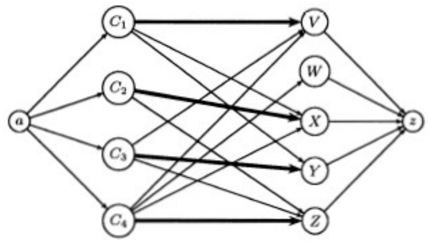
→ 

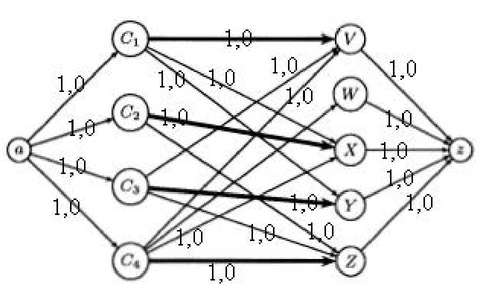
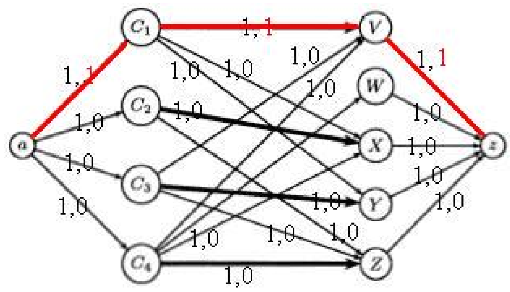
→ 

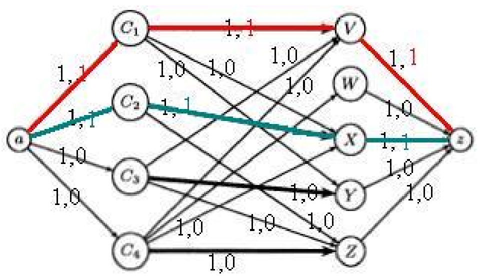
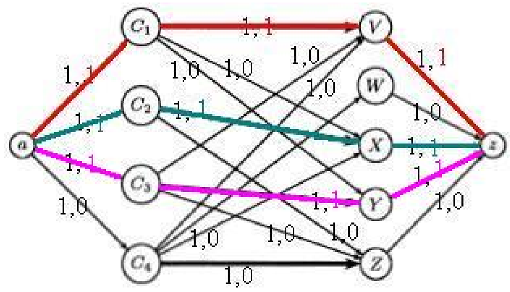


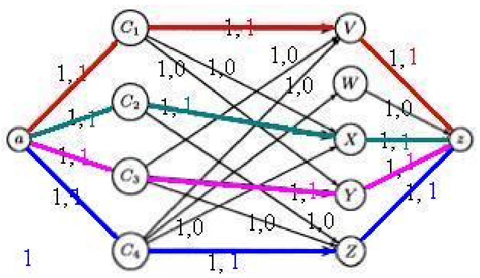
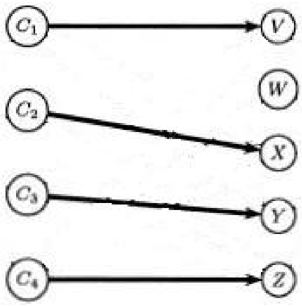
Eg. Find the maximal matching for the left graph.

(Sol.) Matching network:



→  → 

→  → 

→  → 

Theorem **Let G be a directed, bipartite graph with disjoint vertex set V and W in which the edges are directed from vertices in V to vertices in W.**

(a) A flow in the matching network gives a matching in G. The vertex $v \in V$ is matched with the vertex $w \in W$ if and only if the flow in edge $(v,w)=1$.

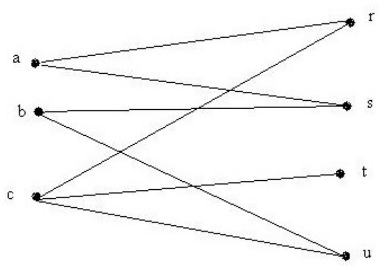
(b) A maximal flow corresponds to a maximal matching.

(c) A flow whose value is $|V|$ corresponds to a complete matching.

Hall's Marriage Theorem **Let G be a directed, bipartite graph with disadjoint vertex set V and W in which the edges are directed from vertices in V to vertices in W. There exists a complete matching in G if and only if $|S| \leq |R(S)|$ for all $S \subseteq V$, where $R(S)=\{w \in W \mid v \in S$ and (v,w) is an edge in $G\}$.**

Eg. There are 3 boys: a(周杰倫), b(劉德華), c(蘇友朋) and 4 girls: r(林志玲), s(侯佩岑), t(林嘉綺), u(白歆惠). If a likes r and s, b likes s and u, c likes r, t and u, can each boy marry a compatible girl?

(Sol.)

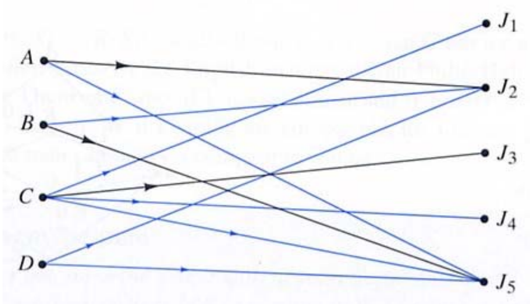


Choose $S_1=\{a,b,c\}$, we have $R(S_1)=\{r,s,t,u\}$ and $|S_1|=3<4=|R(S_1)|$

Choose $S_2=\{a,b\}$, we have $R(S_2)=\{r,s,u\}$ and $|S_2|=2<3=|R(S_2)|$

Choose $S_3=\{a,c\}$, we have $R(S_3)=\{r,s,t,u\}$ and $|S_3|=2<4=|R(S_3)|$. Choose $S_4=\{b,c\}$, we have $R(S_4)=\{r,s,t,u\}$ and $|S_4|=2<4=|R(S_4)|$, $\therefore$ Yes! Each boy can marry a compatible girl.

Eg. There are 4 members in female F4: A(Amy), B(Fanny), C(Tiffany), and D(Stacy), who choose J_1-J_5. Let $S=\{A,B,D\}$, we have $R(S)=\{J_2,J_5\}$ and $|S|=3>2=|R(S)|$, there is not a complete matching for the graph.

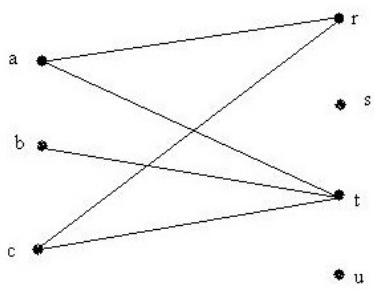








Eg. There are 3 boys: *a*(金城武), *b*(彭政閔), *c*(張家浩) and 4 girls: *r*(柯以柔), *s*(許純美), *t*(蔡淑臻), *u*(如花). If *a* likes *r* and *t*, *b* likes only *t*, *c* likes *r* and *t*, can each boy marry a compatible girl? If *s*(許純美) and *u*(如花) are replaced by 姚采穎 and 吳佩慈, how do you think about it?



(Sol.)

Choose $S=\{a,b,c\}$, we have $R(S)=\{r,t\}$ and $|S|=3>2=|R(S)|$, $\therefore$ No! Some boy can not marry a compatible girl. For example, if *a* married *r* and *b* marries *t*, *c* can not marry his compatible girl. Similarly, if *a* married *t* and *c* married *r*, *b* can not marry his compatible girl. In case *c* married *r* and *b* marries *t*, *a* can not marry his compatible girl.