Graph Theory
MATH20150

Contents

Chapter 1

Basic notions

Definition 1.1. *A graph $G = (V, E)$ is an ordered pair of finite sets, where*

- *V is non-empty;*
- *E is a set of finite subsets of V, each of which contains 2 elements. E may be empty.*

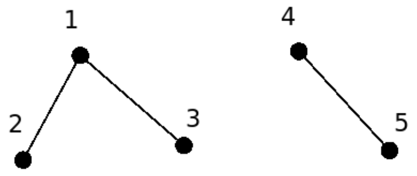
Definition 1.2. *An element of V is called a vertex.*
An element $\{p, q\}$ of E is called an edge, and we say that it joins the vertices p and q.

From the definition, an edge cannot join a vertex to itself (because it would then be of the form $\{p, p\} = \{p\}$ which is a set with only one element).

Remark 1.3. *What we call a graph is sometimes called a simple graph in texts (so: check the terminology when you look at a book).*

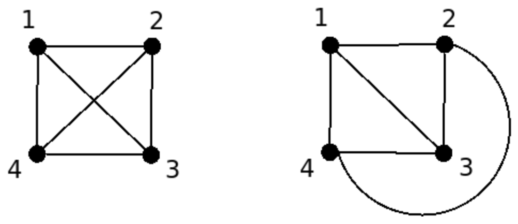
A graph can always be represented as a picture: Represent each vertex in V by a point, and draw a line between any two vertices joined by an edge (the lines may intersect).

Example 1.4. *1. $G = (V, E)$ with $V = \{1, 2, 3, 4, 5\}$, $E = \{\{1, 2\}, \{1, 3\}, \{4, 5\}\}$.*



2. $G = (\{a\}, \emptyset)$.

3.



These two pictures represent the same graph:

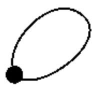
$$G = (\{1,2,3,4\}, \{\{1,2\},\{1,3\},\{1,4\},\{2,3\},\{2,4\},\{3,4\}\}.$$

The fact that the edges $\{1,3\}$ *and* $\{2,4\}$ *intersect in the first picture has no meaning. It is not always possible to avoid such intersections, see the section on planar graphs.*

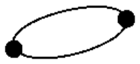
Notation 1.5. *We will usually simply denote the edge* $\{p,q\}$ *by* pq*.*

The following types of edges are excluded by definition:

1. A self-loop: An edge from a vertex to itself.



2. Parallel edges:

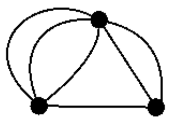


(Since E is a set, the edge $\{u,v\}$ is an element of E, and elements of a set are not repeated.)

Definition 1.6. *A multigraph is a graph in which the existence of several edges joining 2 vertices (parallel edges) is permitted.*

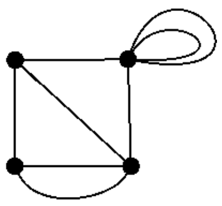
(In this case we need to replace E *by something allowing repetition, for instance* E *could be a tuple that contain subsets of* V *of 2 elements.)*

Example 1.7.



Definition 1.8. *A pseudograph is a multigraph in which self-loops are permitted (as well as parallel self-loops).*

Example 1.9.



Vy

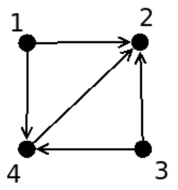
It is also sometimes useful to give a direction to the edges:

Definition 1.10. *A directed graph (or digraph) is an ordered pair of finite sets (V, E) where*

- *V is non-empty;*
- *E is a set of ordered pairs of distinct elements of V (called arcs).*

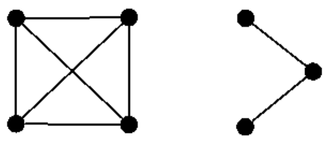
Example 1.11.

$$(\{1,2,3,4\}, \{(1,4),(1,2),(3,2),(3,4),(4,2)\})$$



Definition 1.12. *Let $G = (V, E)$ be a graph with $V = \{v_1, \ldots, v_n\}$ and $E = \{e_1, \ldots, e_m\}$. The order of G is equal to $|V| = n$ (i.e., is the number of vertices) and the size of G is $|E| = m$ (i.e., the number of edges).*

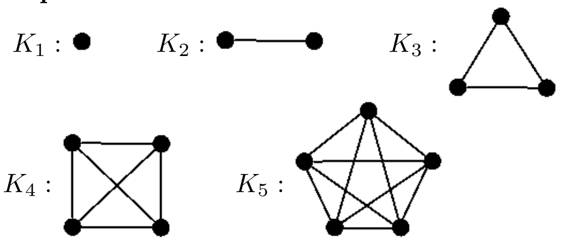
Example 1.13. *The graph*



has order 7 and size 8.

Definition 1.14. *The complete graph of order n, denoted K_n, is the graph with n vertices such that every pair of distinct vertices is joined by an edge.*

Example 1.15.



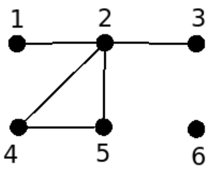
Proposition 1.16. *The size of K_n is $\frac{n(n-1)}{2}$.*

Proof. An edge in K_n is given by any 2 different vertices. The number of ways of picking 2 different vertices among n is $\begin{pmatrix} n \\ 2 \end{pmatrix} = \frac{n(n-1)}{2}$. □

Definition 1.17. *1. If p, q are two vertices of a graph and $e = \{p, q\}$ is an edge of the graph, we say that p is adjacent to q (and of course q is adjacent to p), and that e is incident with p and with q.*

2. *If v is a vertex of a graph G, the degree of v, denoted $d(v)$, is the number of edges incident with v.*

Example 1.18. *The graph*



has $d(1) = 1$, $d(2) = 4$, $d(3) = 1$, $d(4) = 2$, $d(5) = 2$, $d(6) = 0$.

Proposition 1.19. *If G is a graph of order n and v is a vertex of G, then $0 \leq d(v) \leq n-1$.*

Proof. Obvious. $\square$

Definition 1.20. *Let G be a graph of order n and let $d_1, \ldots, d_n$ be the degrees of the vertices of G, in non-decreasing order ($d_1 \leq d_2 \leq \cdots \leq d_n$). The sequence $d_1, \ldots, d_n$ is called the degree sequence of G.*

Proposition 1.21 (Degree sum formula, a.k.a. the handshaking lemma)**.** *Let $d_1, \ldots, d_n$ be the degree sequence of a graph $G = (V, E)$ or order n. Then*

$$\sum_{i=1}^{n} d_i = 2|E|.$$

Proof. If rs is an edge in G, it contributes 1 to $d(r)$ and 1 to $d(s)$. Thus each edge contributes 2 to the sum $\sum_{i=1}^{n} d_i$, so $\sum_{i=1}^{n} d_i = 2|E|$. $\square$

Corollary 1.22. *In any graph, the number of vertices of odd degree is even.*

Proof. Let $v_1, \ldots, v_k$ be the vertices of odd degree and $u_1, \ldots, u_\ell$ be the vertices of even degree. Then

$$d(v_1) + \cdots + d(v_k) + \underbrace{d(u_1) + \cdots + d(u_\ell)}_{even} = \underbrace{2|E|}_{even}.$$

Therefore $d(v_1) + \cdots + d(v_k)$ is even. Since it is a sum of odd numbers, we must have that k is even. $\square$

Example 1.23. *There is no graph with degree sequence* $1, 1, 1, 2, 4, 5, 6, 7, 7, 7$.

Corollary 1.24. *If G is a graph of order $n > 1$ then G has at least two vertices of the same degree.*

Proof. Suppose not. Then the degree sequence of G is such that

$$0 \leq d_1 < d_2 < \cdots < d_n \leq n-1.$$

The only possibility is $d_1 = 0$, $d_2 = 1$, $\ldots$, $d_n = n-1$. Since $d_1 = 0$ there is a vertex adjacent to no other vertex. Since $d_n = n-1$, there is a vertex adjacent to every other vertex. Contradiction. $\square$

Definition 1.25. *Let $G = (V, E)$ be a graph.*

1. *A graph $G' = (V', E')$ is called a subgraph of G if $V' \subseteq V$ and $E' \subseteq E$. We write $G' \subseteq G$.*

2. *If G' is a subgraph of G, we denote by $G \setminus G'$ (G without G') the graph obtained from G by removing all the vertices from G' and all the edges adjacent to at least one vertex of G'.*

Definition 1.26. *Let G be a graph. A walk in G is an alternating sequence of vertices and edges in G:*

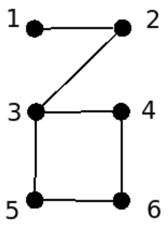
$$W = v_0, e_1, v_1, e_2, \dots, v_{\ell-1}, e_\ell, v_\ell$$

where e_i is the edge $v_{i-1}v_i$ for $i = 1, \dots, \ell$.

The length of the walk W is ℓ, the number of edges in W. We write for short $W = v_0 v_1 \cdots v_\ell$ (since there is only one edge between v_{i-1} and v_i).

A closed walk is a walk with $v_0 = v_\ell$.

Example 1.27.



1 2 3 4 3 5 6 is a walk in G.

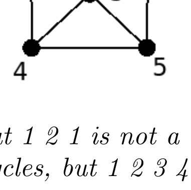
Definition 1.28. 1. *A trail is a walk in which no edge is repeated (but a vertex could be repeated).*

2. *A path is a walk with no repeated vertices. A path is always a trail since to repeat an edge you must repeat at least one vertex.*

3. *A circuit is a trail $v_0 v_1 \dots v_\ell$ where $v_0 = v_\ell$ (and $\ell \geq 3$ because it is a trail).*

4. *A cycle is a walk $v_0 v_1 \dots v_\ell$ with $l \geq 3$, $v_0 = v_\ell$ and $v_0, \dots, v_{\ell-1}$ are all different (so that no edge and no vertex is repeated; a cycle is a path that is also a circuit).*

Example 1.29. *In the graph*



1 2 3 4 5 3 1 is a circuit, but 1 2 1 is not a circuit;
1 2 5 4 1 and 1 2 3 1 are cycles, but 1 2 3 4 5 3 1 is not a cycle.

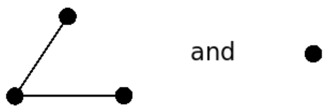
Proposition 1.30. *If K is a circuit, then K contains a cycle.*

Proof. Let L be a subgraph of K that is a circuit with the smallest possible number of edges (it could be K itself): $L = v_0 v_1 \dots v_n v_0$. We show that $v_0, \dots, v_\ell$ are all different (i.e., the L is a cycle): If it were not the case, say $v_r = v_s$ for some $0 \leq r < s \leq n$ then $v_r v_{r+1} \dots v_s$ would be a circuit in K with less edges than L, contradiction. □

Definition 1.31. *Let x and y be vertices in a graph G, with $x \neq y$. We say that x and y are connected if and only if there exists a path (equivalently, a walk, cf. Proposition 1.33) in G of the form $xv_1 \cdots v_r y$ (connecting x and y).*

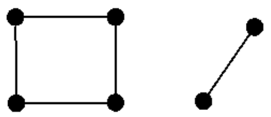
A graph G is called connected if every pair of vertices x, y in G with $x \neq y$ is connected, or if G has only one vertex.

Example 1.32.



are connected graphs.

The graph



is not connected.

Proposition 1.33. *Let x, y be distinct vertices in a graph G. If there exists a walk in G from x to y, then there exists a path in G from x to y.*

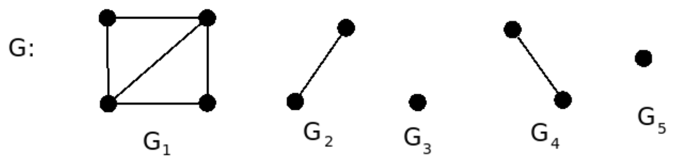
Proof. Take W a walk from x to y of minimal length (i.e., with minimal number of edges): $W = (x = v_0)v_1 v_2 \cdots v_n (y = v_{n+1})$.

We check that W is a path: If not then we have $v_r = v_s$ for some $0 \leq r < s \leq n+1$. If we remove the section $v_s v_{s+1} \cdots v_{r-1}$ from W we get $v_0 \cdots v_{s-1}(v_r = v_s)v_{r+1} \cdots v_{n+1}$, which is also a walk from x to y, but is shorter than W, a contradiction. □

Proposition 1.34. *Let $G = (V, E)$ be a graph. There exist $k \in \mathbb{N}$ and subgraphs $G_1, \dots, G_k$ of G such that*

1. *G is the union of $G_1, \dots, G_k$, i.e., if $G_i = (V_i, E_i)$ then $V = V_1 \cup \cdots \cup V_k$ and $E = E_1 \cup \cdots \cup E_k$.*
2. *Each G_i is connected.*
3. *For every $i \neq j$, G_i and G_j have no vertex in common.*

Example 1.35.



Proof. By induction on n, the order of G:

- $n = 1$. Then G has only one vertex and is thus connected.
- If $n > 1$ (we assume that the result has been checked for graphs of order $< n$).

  Let v be a vertex of G and let

  $$V_1 = \{x \mid x \text{ vertex of } G,\ x \text{ connected to } v\} \cup \{v\},$$

  $$E_1 = \text{ the set of all edges } xy \text{ of } G \text{ with } x, y \in V_1.$$

  Let $G_1 = (V_1, E_1)$. G_1 is a graph and is connected: If $x, y \in V_1$, then there are paths from x to y (and thus at least a walk from x to y).

  If $G_1 = G$ the proof is finished

  If $G_1 \neq G$, let G' be the graph obtained from G by removing every edge and every vertex from G_1 (i.e. $G' = G \backslash G_1$ in the terminology of definition 1.38).

  Then $G = G_1 \cup G'$, G_1 and G' have no common vertex, and G' has order less than n. By induction we have $G' = G_2 \cup \cdots G_n$ where the G_i are connected and have no vertices in common. Therefore $G = G_1 \cup G_2 \cup \cdots \cup G_n$ where the G_i are connected and have no vertices in common.

□

The graph G_i in the proof is a maximal connected subgraph of G, i.e., a connected subgraph that is not contained in a larger connected subgraph.

Definition 1.36. *A maximal connected subgraph of G is called a component of G.*

So each graph is the disjoint union of its components (see the previous example).

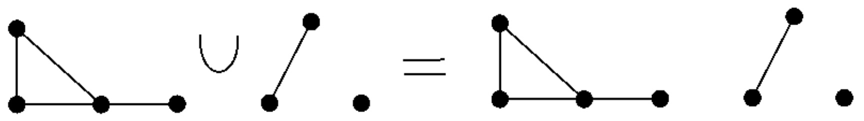
Definition 1.37. *We denote by $\omega(G)$ the number of components of G.*

Definition 1.38. *Some operations with graphs:*

1. *The union: If $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ then*

   $$G_1 \cup G_2 = (V_1 \cup V_2, E_1 \cup E_2).$$

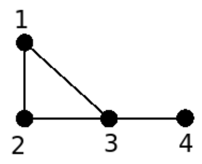
   For instance:

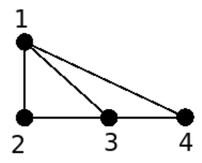
   Adding edges: If E_2 is a set of new edges in G_1, we define

   $$G_1 \cup E_2 = (V_1, E_1 \cup E_2).$$

For instance, if G_1 is



and $E_2 = \{14\}$, then $G \cup E_2$ is

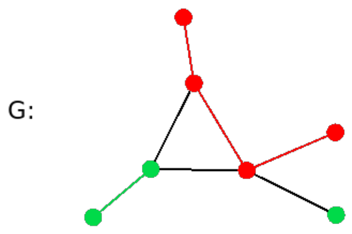


2. *The difference: Let $H = (V_H, E_H)$ be a subgraph of G. Then*

$$G \setminus H = (V', E'), \text{ where}$$

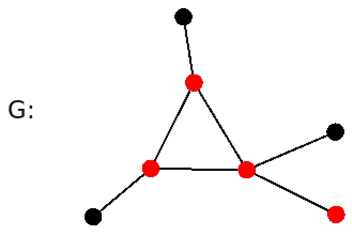
$V' = V \setminus V_H$ and $E' = E$ from which we remove all the edges that have at least one end in V_H.

For instance, if H is the red subgraph, then $G \setminus H$ is the green subgraph:

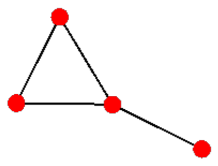


3. *Induced subgraph (restriction): Let $G = (V, E)$, and let $V' \subseteq V$ be a subset of vertices of G. Then $G[V'] = (V', E')$ where E' is the set of all edges of E that have both ends in V'.*

For instance, if V' is the red set of vertices:

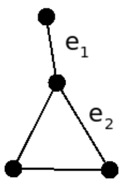


then $G[V']$ is

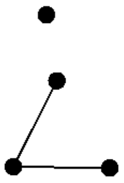


4. *Removing edges: Let $G = (V, E)$ and let E' be a subset of E. Then $G \setminus E' = (V, E \setminus E')$.*

   For instance, if G is



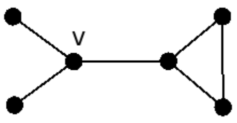
   then $G \setminus \{e_1, e_2\}$ is



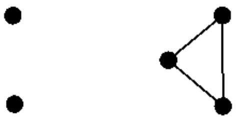
Definition 1.39. *Let $G = (V, E)$ be a graph.*

1. *A vertex $v \in V$ is called a cut-vertex if $\omega(G \setminus \{v\}) > \omega(G)$.*

   For instance, if G is

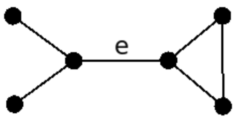


   then $\omega(G) = 1$, but $\omega(G \setminus \{v\}) = 3$:

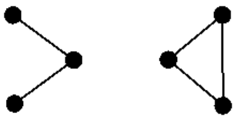


2. *An edge $e \in E$ is called a bridge (also cut-edge) if $\omega(G \setminus \{e\}) > \omega(G)$.*

   For instance, if G is



   then $\omega(G) = 1$, but $\omega(G \setminus \{e\}) = 2$:

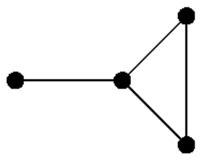


Chapter 2

Graphic sequences, adjacency matrix

Definition 2.1. *A sequence of integers $(d_1, \ldots, d_n)$ is called graphic if it is the degree sequence of a graph.*

Example 2.2. *1.* $(1, 2, 2, 3)$ *is graphic:*



2. $(1, 1, 1, 2, 2, 3, 4, 5)$ *is not graphic (there would be an odd number of vertices of odd degree, see corollary 1.22).*

3. $(2, 3, 3, 4, 5, 6, 7)$ *is not graphic (the graph would have 7 vertices and one of them would have degree* 7*, which is impossible).*

Problem: Given a sequence $(d_1, \ldots, d_n)$ of elements of $\mathbb{N}$, determine whether or not it is graphic. Clearly, if we have $d_i \geq n$ for some i, then the sequence is not graphic (having a vertex with degree equal at least to the number of vertices would require self-loops or parallel edges). The other case is answered by this result:

Theorem 2.3 (Havel (1955)- Hakimi (1962))**.** *Let $d = (d_1, \ldots, d_n)$ be a sequence of elements of $\mathbb{N} \cup \{0\}$ such that $d_1 \leq d_2 \leq \cdots \leq d_n < n$. Construct*

$$d' = (\underbrace{d_1,\, d_2,\, \ldots}_{\substack{n-d_n-1 \text{ elements} \\ \text{can be empty} \\ (\text{if } d_n = n-1)}}, \underbrace{d_{n-d_n} - 1, \ldots, d_{n-1} - 1}_{d_n \text{ elements}})$$

Then d is graphic if and only if d' is graphic (after being re-ordered to be in non-decreasing order, if necessary).

Example 2.4. *1.*

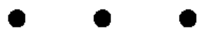
$$(0,1,1,1,2) \overset{apply\ Thm\ .2.3}{\longrightarrow} (0,1,0,0) \overset{make\ it\ non\text{-}decreasing}{\longrightarrow} (0,0,0,1)$$

$$\overset{apply\ Thm.\ 2.3}{\longrightarrow} (0,0,-1)\ \textit{not graphic}$$

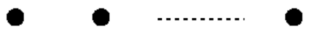
2.

$$(1,1,2,2,2,4) \overset{apply\ Thm.\ 2.3}{\longrightarrow} (1,0,1,1,1) \overset{make\ it\ non\text{-}decreasing}{\longrightarrow} (0,1,1,1,1)$$

$$\overset{apply\ Thm.\ 2.3}{\longrightarrow} (0,1,1,0) \overset{make\ it\ non\text{-}decreasing}{\longrightarrow} (0,0,1,1) \longrightarrow (0,0,0)\ \textit{graphic:}$$



Remark 2.5. *The sequence $(0,0,\ldots,0)$ is always graphic:*



(a graph with no edges).

Consequently: Applying the transformation from theorem 2.3. we can always reduce to either

- *A sequence of zeros (which is graphic, so the starting sequence is graphic)*

or

- *A sequence containig negative numbers (which is not graphic, so the starting sequence is not graphic).*

Proof. (of theorem 2.3)

- We first show that d' graphic implies d graphic:

  By hypothesis d' is the degree sequence of some graph $G' = (V', E')$, and

  $$d' = (d_1, d_2, \ldots, d_{n-d_n} - 1, \ldots, d_{n-1} - 1).$$

  If we denote the coefficients of d' above by $d'_1, \ldots, d'_{n-1}$, we have $d'_i = d(v_i)$ where $V' = \{v_1, \ldots, v_{n-1}\}$.

  To get $G = (V, E)$ out of G' we add one vertex: $V = V' \cup \{v_n\}$ and we add edges from v_n to $v_{n-d_n}, \ldots, v_{n-1}$.:

  $$E = E' \cup \{v_n v_{n-d_n}, \ldots, v_n v_{n-1}\}.$$

  Then the degree sequence of G is d.

- We now show that d graphic implies d' graphic:

  We write

  $$d = (d_1, \ldots, d_{n-d_n}, \ldots, d_{n-1}, d_n),$$

  where $d_i = d(v_i)$ and $V = \{v_1, \ldots, v_n\}$. We consider two cases

  Case 1: If v_n is adjacent to $v_{n-d_n}, \ldots, v_{n-1}$. Then we remove v_n and all the edges $v_n v_{n-d_n}, \ldots, v_n v_{n-1}$. The resulting graph has degree sequence d'.

Case 2: If v_n is not adjacent to all of $v_{n-d_n}, \dots, v_{n-1}$.

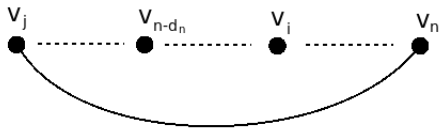
Consider the following Fact

<u>Fact</u>: We can modify the edges in the graph G to obtain a graph with the same degree sequence d, but where the amount of neighbours of v_n among $v_{n-d_n}, \dots, v_{n-1}$ is increase by 1

Applying this Fact several times, we will be back to case 1, from which we can conclude. So we only have to justify why this fact is true:

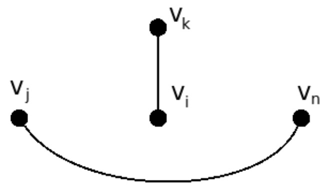
Let $i \in \{n - d_n, \dots, n-1\}$ be the least integer such that v_n is not a neighbour of v_i.

Since $d(v_n) = d_n$, v_n is adjacent to some v_j with $j < n - d_n$:

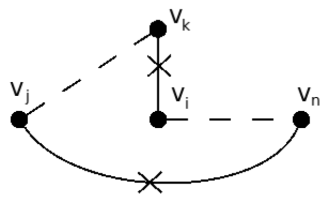


If $d(v_j) = d(v_i)$, we simply relabel the vertices and swap v_i and v_j (the degree sequence is unchanged).

If $d(v_j) < d(v_i)$: There is a vertex v_k adjacent to v_i but not to v_j (and $v_k \neq v_n$ because v_n is adjacent to v_j):

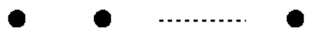


In this case we remove the edges v_jv_n and v_iv_k, then add the edges v_jv_k and v_iv_n:

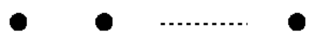


Doing this preserves the degree of all vertices, so the degree sequence is unchanged, but now v_n has one more neighbour in $v_{n-d_n}, \dots, v_{n-1}$ (namely v_i). □

The proof of theorem 2.3 is an algorithm. We saw in the previous remark that is a sequence is graphic, repeated applications of theorem 2.3 will give the sequence $(0, \dots, 0)$, corresponding to the graph



Using the part "d' graphic $\Rightarrow$ d graphic" of the proof of theorem 2.3 (the easy part), and starting from the graph



we can construct a graph with degree sequence d.

Example 2.6. *1.*

(1,1,2,2,2,4)

$\downarrow$

(1,0,1,1,1)

$\downarrow$

(0,1,1,1,1)

$\downarrow$

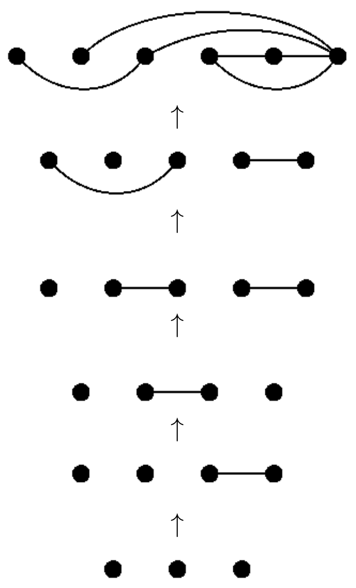
(0,1,1,0)

$\downarrow$

(0,0,1,1)

$\downarrow$

(0,0,0) $\longrightarrow$

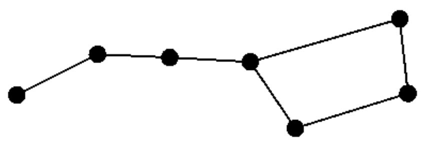


$n = 6,\ d_n = 4$

$n = 5,\ d_n = 1$

$n = 4,\ d_n = 1$

2. Ursa major:



Its degree sequence is $(1, 2, 2, 2, 2, 2, 3)$. *We know it is graphic, but we still apply the above algorithm:*

(1,2,2,2,2,2,3)

$\downarrow$ $\quad$ $n = 7,\ d_n = 3$

(1,2,2,1,1,1)

$\downarrow$

(1,1,1,1,2,2)

$\downarrow$ $\quad$ $n = 6,\ d_n = 2$

(1,1,1,0,1)

$\downarrow$

(0,1,1,1,1)

$\downarrow$ $\quad$ $n = 5,\ d_n = 1$

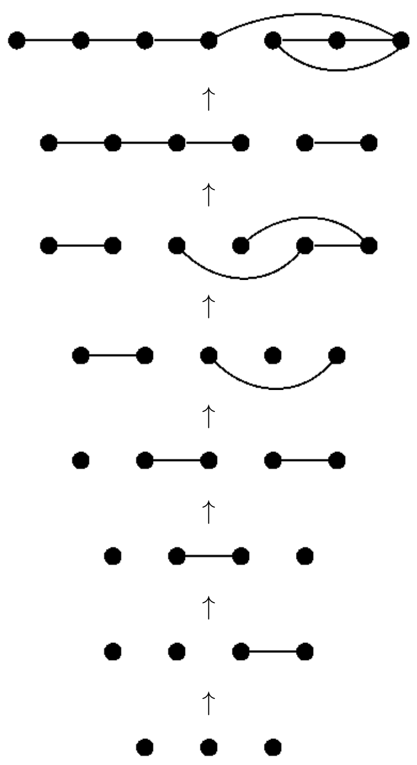
(0,1,1,0)

$\downarrow$

(0,0,1,1)

$\downarrow$ $\quad$ $n = 4,\ d_n = 1$

(0,0,0) $\longrightarrow$



The result we obtained here does not look like Ursa Major. Conclusion: There can be different graphs with the same degree sequence.

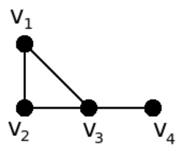
Definition 2.7. *Let $G = (V, E)$ be any graph (graph, multigraph, pseudograph, directed graph). Consider any labelling $V = \{v_1, \ldots, v_n\}$ of the vertices of G. The adjacency matrix of G (with respect to this labelling) is*

$$A = (a_{ij})_{i,j=1,\ldots,n},$$

where

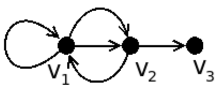
$$a_{ij} = \text{the number of edges in } G \text{ starting at } v_i \text{ and ending at } v_j.$$

Example 2.8. *1.*



$$A = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

2.



$$A = \begin{pmatrix} 1 & 2 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}.$$

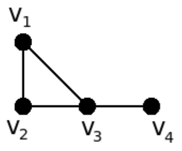
Some properties of A (depending on the type of graph):

- "normal" graph: $a_{ij} \in \{0,1\}$.
- no self-loops: $a_{ii} = 0$.
- undirected: $a_{ij} = a_{ji}$, i.e., $A = A^t$.

Proposition 2.9. *Let* $A^k = \underbrace{A \cdot A \cdot \dots \cdot A}_{k\ times} = (a_{ij}^{(k)})_{i,j=1,\dots,n}$. *Then*

$$a_{ij}^{(k)} = \text{ the number of walks of length } k \text{ from } v_i \text{ to } v_j.$$

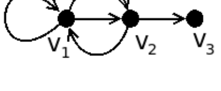
Example 2.10. *1.*



$$A^2 = \begin{pmatrix} 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 \\ 1 & 1 & 3 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}.$$

There are 2 *walks of length* 2 *from* v_1 *to* v_1, *and one walk of length* 2 *from* v_2 *to* v_3.

2.



$$A^2 = \begin{pmatrix} 3 & 2 & 2 \\ 1 & 2 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

There are 3 walks of length 2 from v_1 *to* v_1, *there are 2 walks of length 2 from* v_1 *to* v_3.

Proof. (of theorem 2.9) By induction on k.

- $k = 1$. It is clear by definition of A:

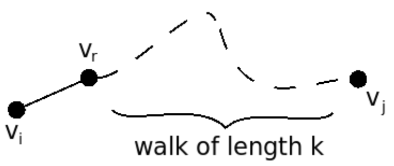
$$\begin{aligned} a_{ij}^{(1)} = a_{ij} = &\text{ the number of edges from } v_i \text{ to } v_j \\ = &\text{ the number of walks of length 1 from } v_i \text{ to } v_j \end{aligned}$$

- We assume that the statement is true for k and prove it for $k+1$. We have $A^{k+1} = A \cdot A^k$, so

$$a_{ij}^{(k+1)} = \sum_{r=1}^{n} a_{ir}^{(1)} a_{rj}^{(k)}.$$

  Every walk of length $k+1$ from v_i to v_j is of the form:

  - an edge from v_i to some v_r, followed by
  - a walk of length k from v_r to v_j.



  The number $a_{ir}^{(1)} a_{rj}^{(k)}$ is the number of walks of length $k+1$ from v_i to v_j that have $v_i v_r$ as first step (indeed: $a_{ir}^{(1)}$ is the number of different edges from v_i to v_r, and $a_{rj}^{(k)}$ is the number of different walks of length k from v_r to v_j).

  Therefore $\sum_{r=1}^{n} a_{ir}^{(1)} a_{rj}^{(k)}$ is the number of all possible walks of length $k+1$ from v_i to v_j.

□

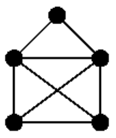
Chapter 3

Eulerian graphs

Definition 3.1. *Let G be a connected multigraph.*

1. *A walk in G is called an Euler trail if it traverses every edge of G exactly once.*
2. *An Euler circuit in G is a closed Euler trail (i.e., and Euler trail with the same first and last vertex).*
3. *G is Eulerian if it admits an Euler circuit.*

Example 3.2.



Has an Euler trail (exercise). It has even more than just one.

Theorem 3.3 (Euler 1736)**.** *A connected multigraph is Eulerian if and only if every vertex has even degree.*

Proof. Let G be the graph.

"$\Rightarrow$" We assume that G is Eulerian. Let C be an Euler circuit in G. In traversing C, every time we "enter" a vertex by an edge, we "leave" it by another edge, so this contributes 2 to the degree of the vertex (it is alos true for the first and last edges: they contribute 2 to the degree of the starting vertex)

Since all the edges are in C and G is connected, every vertex appears in C, and every edge. So each vertex has even degree.

"$\Leftarrow$" We assume that every vertex has even degree. Let $W = v_0e_0v_1 \dots e_{k-1}v_k$ be a longest walk in G with no repeated edge (but vertices can be repeated). We will show that W is an Euler circuit. We first consider v_k more closely:

1. Every edge adjacent to v_k appears in W: Because otherwise we could use an edge v_kv_{k+1} not in W to a walk longer than W, a contradiction.
2. $v_k = v_0$ (so W is a closed walk):

$$v_0 \underbrace{e_0v_1e_1v_2 \dots v_{k-1}e_{k-1}}_{\text{we look at this part of } W} v_k$$

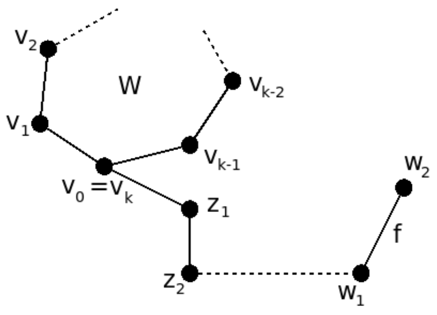
Since every edge adjacent to v_k appears in W exactly once, to determine $d(v_k)$ it suffices to count the adges adjacent to v_k as they appear along W.

- For each $i \in \{1, \dots, k-1\}$ such that $v_i = v_k$ (the "inner" vertices of W equal to v_k), the number $d(v_k)$ increases by 2 (one for the "arriving" vertex, one for the "leaving" vertex).
- The final part of W: e_{k-1} increases $d(v_k)$ by one.

This will give an odd number (which is impossible by assumption), unless $v_0 e_0$ contributes to $v(v_k)$, i.e., $v_0 = v_k$.

Suppose now that W is not an Euler circuit. Then:
Fact: G has an edge e outside of W but incident with some vertex v_i of W (so $e = uv_i$).
Proof of the fact: Since W is not en Euler circuit, there is an edge $f = w_1 w_2$ in G that is not in W. Since G is connected, there is a path C from v_k to $w1$:

$$C = v_k f_1 z_1 f_2 z_2 \dots z_\ell f_{\ell+1} w_1.$$



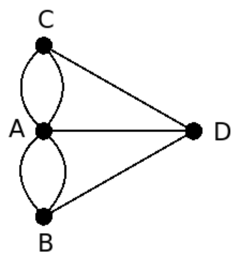
By the observation above we know that f_1 is in W. We take for e the first edge in C that is not in W, if it exists. If it does not exist we take $e = f$. End of the proof of the fact.

Then the walk

$$uev_i e_i \dots e_{k-1} v_k e_0 v_1 \dots e_{i-1} v_i$$

has no repeated edges and is longer than W, a contradiction. □

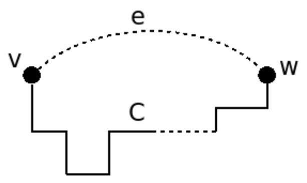
Answer to the Königsberg bridge problem:
The problem is represented by the graph



with $d(A) = 5$, $d(B) = 3$, $d(C) = 3$, $d(D) = 3$, so there is no Euler circuit.

Corollary 3.4. *A connected multigraph G has an Euler trail that is not an Euler circuit if and only if it has exactly two vertices of odd degree.*

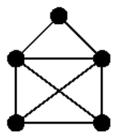
Proof. "$\Rightarrow$" Let C be an Euler trail in G, starting at v and ending at w. We add a new edge e joining v and w, and leT $G' = G \cup \{e\}$.



We then have an Euler circuit in G' nad, by theorem 3.3, all vertices in G' have even degree. Therefore all vertices in G have even degree, except v and w, which have odd degree.

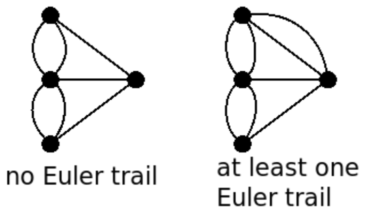
"$\Leftarrow$" Let v and w be the tw overtices of G with odd degree. We add a new edge e between v and w and get the graph $G' = G \cup \{e\}$. In G' all vertices have even degree, so by theorem 3.3 there is an Euler cricuit C in G'. Removing e from C gives an Euler trail in G. ☐

Example 3.5. *1.*



has an Euler trail.

2. Adding one bridge in Königsberg will make an Euler trail possible:



Chapter 4

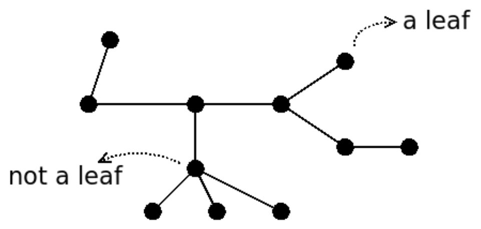
Trees

Definition 4.1. *1. A connected graph with no cycles in called a tree (terminology: "with no cycles" = acyclic).*

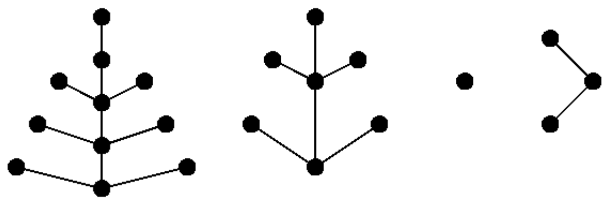
2. A graph with no cycles is called a forest (because it is the union of its components, which are connected graphs with no cycles, i.e., trees).

3. The vertices of degree 1 in a tree are called leaves.

Example 4.2. *A tree:*

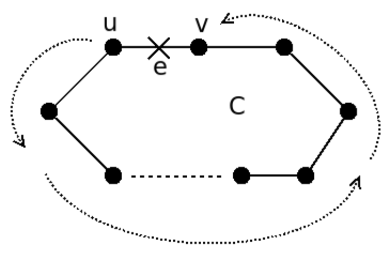


A forest:



Lemma 4.3. *Let G be a connected graph which contains a cycle C. Let e be an edge in C. Then $G \setminus \{e\}$ is connected.*

Proof. Sketch of the proof:



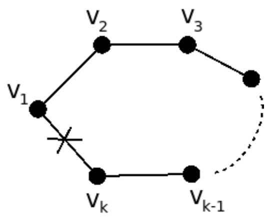
In any path, going from u to v via $e = uv$ can be replaced by going around the "other side" of the cycle. □

Theorem 4.4. *Let G be a graph of order n. Then the following are equivalent.*

1. *G is a tree.*
2. *G is connected and, whenever e is an edge in G, $G \setminus \{e\}$ is not connected.*
3. *G contains no cycle and, whenever x and y are non-adjacent vertices of G, $G \cup \{xy\}$ contains exactly one cycle.*
4. *G is connected and has $n - 1$ edges.*
5. *G contains no cycle and has $n - 1$ edges.*
6. *Whenever x and y are vertices in G, there is exactly one path from x to y in G.*

Proof. 1. $\Rightarrow$ 2. Let $e = xy$ be an edge of G wiht x, y vertices. Suppose that $G \setminus \{e\}$ is connected, and let $xv_1 \dots v_k y$ be a path in G from x to y. Then $xv_1 \dots v_k yx$ is a cycle in G, contradiction.

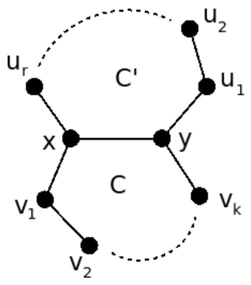
2. $\Rightarrow$ 1. We have to show that G contains no cycle. Suppose that $v_1 v_2 \dots v_k v_1$ is a cycle in G.



Then $G \setminus \{v_1 v_k\}$ is connected (by lemma 4.3), contradiction.

2. $\Rightarrow$ 3. We assume 2. (and thus 1., since 1. $\Leftrightarrow$ 2.) Let x, y be non-adjacent vertices in G. Since G is connected we have a path $xv_1 \dots v_k y$ from x to y in G. Thus $xv_1 \dots v_k yx$ is a cycle in $G \cup \{xy\}$. We have to show that it is the only cycle:

Suppose we have another cyle C' in $G \cup \{xy\}$. Since G contains no cycle, C' must contain the edge xy. We write C' starting from y: $yu_1 \dots u_r xy$, so

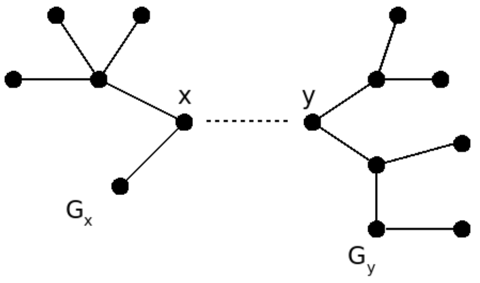


Then $D = xv_1 \dots v_k y u_1 \dots u_r x$ is a path from x to x in G with at least 2 different edges (since $C \neq C'$). Then D contains a cycle (same idea as the proof of proposition 1.30, exercise), contradicting hypothesis 1.

3. $\Rightarrow$ 1. We know that G contains no cycleco we only need to check that G is connected. Let x, y be two different vertices in G. If x and y are adjacent, then there is a path from x to y. If x and y are not adjacent, by hypothesis there is a cycle in $G \cup \{xy\}$. The "other side" of the cycle gives a path in G from x to y.

1. $\Rightarrow$ 4. We proceed by induction on n the number of vertices of G.

- $n = 1$. Then G has no edges (only one vertex).
- $n > 1$ and assume that the result holds for graphs with less than n vertices. Let xy be an edge in G



The graph $G \setminus \{xy\}$ is not connected (by property 2).

Let G_x be the component of $G \setminus \{xy\}$ containing x, and G_y the component of $G \setminus \{xy\}$ containing y. Since G contains no cycle and is connected, we can check (exercise) that G_x and G_y have no vertices in common and are the only components of $G \setminus \{xy\}$.

So $G = G_x \cup G_y \cup \{xy\}$ and both G_x and G_y are connected without cycles (i.e., are trees). Let n_x be the order of G_x and n_y be the order of G_y. We have $n = n_x + n_y$ and thus $n_x, n_y < n$, so by induction hypothesis, the number of edges in G_x is $n_x - 1$ and the number of edges in G_y is $n_y - 1$. So the number of edges in G is

$$(n_x - 1) + (n_y - 1) + \underbrace{1}_{\text{for } xy} = n_x + n_y + 1 = n - 1.$$

4. $\Rightarrow$ 1. We will do it after Proposition 4.9 (and we will not use it before).

1. $\Rightarrow$ 5 Obvious with 4.

5. $\Rightarrow$ 1. We have to show that G is connected. Suppose not and let $G_1, \ldots, G_k$ be the components of G ($k \geq 2$), of respective orders $n_1, \ldots, n_k$. Each G_i is a tree (connected graph with no cycle), so has $n_i - 1$ edges. Therefore G has $(n_1 - 1) + \cdots + (n_k - 1) = n - k$ edges, contradiction to the hypothesis. (since $k \geq 2$).

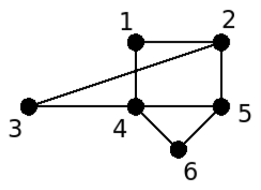
1. $\Rightarrow$ 6. Outline: Since G is connected, there is at least one path between any two vertices. If there were 2 distinct paths there would be a circuit and then a cycle, contradiction.

6. $\Rightarrow$ 1. The graph G is connected. It also has no cycles, since otherwise there would be two different paths between 2 vertices, a contradiction. $\square$

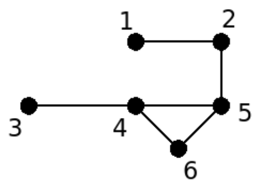
Definition 4.5. *Let $G = (V, E)$ be a graph, and let $H = (V', E')$ be a subgraph of G. If $V = V'$ we say that H is a spanning subgraph of G.*

If T is a spanning subgraph of G and is also a tree, we call T a spanning tree of G (it can only exist if G is connected)

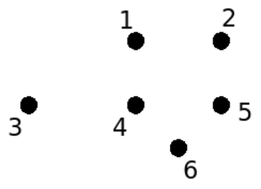
Example 4.6. *Let G be*



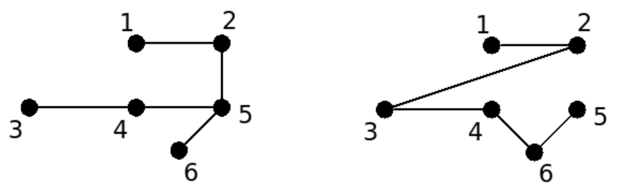
then



is a spanning subgraph, and so is



These are examples of two spanning trees of G:



Trees appear naturally to solve the following kind of problem:

Water is to be supplied by pipelines to n villages, such that the pipelines only meet at villages. Waht is the lest length of pipeline necessary to connect all villages?

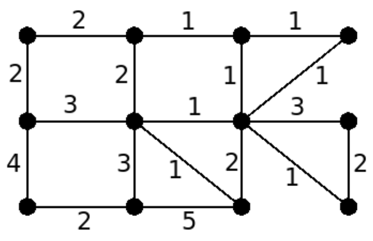
To solve this we need

Definition 4.7. *A weighted graph is a graph such that every edge has a positive number, its weight, attached to it.*

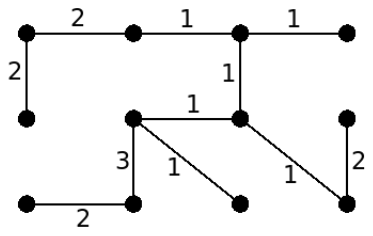
The weight of a graph is the sum of the weights of all its edges.

In the case of the problem above, the weight of the edge betwwe 2 villages is the length of pipeline necessary to connec them. The problem then amounts to finding a connected subgraph of miminum weight.

Example 4.8. *Let G be the (weighted) graph*



A minimum weight connected spanning subgraph is



It has weight 17 *and it is also a tree. There is in general more than one minimum spanning graph (here for instance, you can try to find others).*

Proposition 4.9. *Let G be a connected weighted graph, and let M be a least weight connected spanning subgraph in G. Then M is a tree.*

Proof. M is connected by definition, so we just want to show that M does not contain any cycle. If M contains a cycle C and e is an edge in C, then $M \setminus \{e\}$ is a connected subgraph of G of weigth less than the weight of M, a contradiction. $\square$

Proof of 4. $\Rightarrow$ 1. from Theorem 4.4. Let T be a spanning tree of G (take any minimum weight spanning subgraph of G). Then T has $n-1$ edges by 1. $\Rightarrow$ 4. Since G has also $n-1$ edges, we must have $G = T$, so G is a tree. $\square$

To find a least weight spanning tree in a connected weighted graph, the use the following:

Kruskal's algorithm (1956)

Let $G = (V, E)$ be a weighted connected graph of order n. The algorithm proceeds as follows.

Step 1 Choose an edge e_1 of G such that $w(e_1)$ is as small as possible.

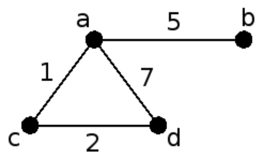
Step 2 If the edges $e_1, \ldots, e_k$ have already been chosen, choose $e_{k+1} \in E \setminus \{e_1, \ldots, e_k\}$ such that

(i) $\{e_1, \ldots, e_k, e_{k+1}\}$ does not contain a cycle (i.e., e_{k+1} does not form a cycle withe the other edges from $\{e_1, \ldots, e_k\}$.

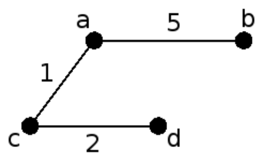
(ii) $w(e_{k+1})$ is as small as possible subject to (i).

Step 3 Repeat Step 2 until it is impossible to do so.

Example 4.10. *1. We apply the algorithm to the graph*

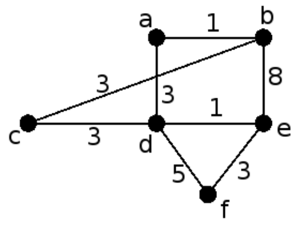


Step 1: ac, Step 2: ac, cd, Step 2: ac, cd, ab, we stop. The result is

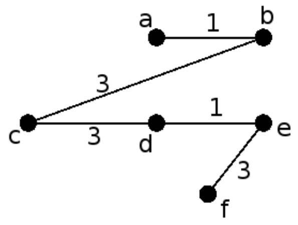


and has weight 8.

2. We apply the algorithm to the graph



Step 1: ab, Step 2: ab, de, Step 2: ab, de, ef, Step 2: ab, de, ef, cb, Step 2: ab, de, ef,, cb, cd we stop. The result is



and has weight 15.

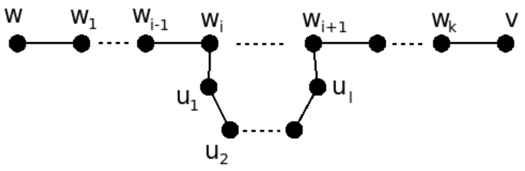
Theorem 4.11. *This algorithm produces a minimal weight spanning tree.*

Lemma 4.12. *Let $G = (V, E)$ be a connected graph and let $H = (V', E')$ be an acyclic subgraph of G with as many edges as possible. Then H is a spanning tree of G.*

Proof. We first show that H is a spanning subgraph of G, i.e., that $V' = V$. Suppose there is $v \in V$ such that $v \notin V'$. Let $w \in H$. Since G is connected there is apath $P = \underbrace{w_0}_{=w} w_1 \dots w_k \underbrace{w_{k+1}}_{=v}$ in G from w to v.

We take such a path with the minimum number of edges outside E'. Since $v \notin V'$, the edge $w_k v$ is not in E'.

Let $w_i w_{i+1}$ be the first edge not in H. So $w_{i-1}w_i \in E'$ and thus $w_i \in V'$. If $w_{i+1} \notin H$ we can add the edge $w_i w_{i+1}$ to H (it will not create a cycle), contradicting the definition of H. So $w_{i+1} \in H$, and we cannot add $w_i w_{i+1}$ to H, i.e., $H \cup \{w_i w_{i+1}\}$ contains a cycle $C = w_{i+1} w_i u_1 \dots u_\ell w_{i+1}$



Therefore $ww_1 \dots w_i u_1 \dots u_\ell w_{i+1} \dots w_k v$ is a path in G from w to v with less edges not in E' than P, contradiction.

We now show that H is a tree. We know that H is acyclic, so we only have to show that H is connected. Let $H_1, \dots, H_t$ be the components of H (they are trees), and assume $t \geq 2$.

Since G is connected and H contains all vertices of G, there is an edge $e \in E \setminus E'$ such that e connects H_1 and H_i for some $i > 1$ (take a vertex x in H_1, y a vertex in H_2 and a path P in G from x to y. Take for e the first edge not in H_1).

Then we can add e to H without creating a cycle (if it created a cycle, the "other side" of the cycle could connect H_1 and H_i in H, impossuble), contradicting the definition of H. □

Proof of Theorem 4.11. Let H be the graph produced by the algorithm. By lemma 4.12 it is a spanning tree.

Let $e_1, \dots, e_{n-1}$ be the edges of H (where n is the number of vertices of G, i.e., of H), in the order in which they are added by the algorithm.

If T is a different spanning tree of G, there is an edge of H that is not in T: Otherwise T would have more edges that H, which is impossible (they are both trees with n vertices).

We denote by $f(T)$ the smallest i such that e_i is not in T.

Assume now that H is not a minimal weight spanning tree and let T be a minimal weight spanning tree with $f(T)$ as large as possible. Let $k = f(T)$, i.e., $e_1, \dots, e_{k-1}$ are in H and T, but e_k is not in T. By theorem 4.4, $T \cup \{e_k\}$ contains a unique cycle C. Let e'_k be an edge in C that is in T but not in H (there is such an edge: if every edge of T that is in C (=all edges of C, except e_k) were in H, we would have C in H (since $e_k \in H$), contradiction).

Since C is a cylcle, $T' = (T \cup \{e_k\}) \setminus \{e'_k\}$ is connected and has $n-1$ edges. So by theorem 4.4 T' is another spanning tree of G. Clearly:

$$w(T') = w(T) + w(e_k) - w(e'_k).$$

In Kruskal's algorithm, e_k was chosen with the smallest weigth such that $\{e_1, \dots, e_k\}$ is acyclic. Since $\{e_1, \dots, e_{k-1}, e'_k\}$ is a subgraph of T, it is also acyclic, so we must have $w(e'_k) \geq w(e_k)$. Therefore $w(T') \leq w(T)$ and T' is a minimal weight spanning tree. However:

$$\underbrace{f(T') > k}_{\text{since } e_1, \dots, e_k \text{ are in } T'} \qquad = f(T),$$

contradicting the choice of T. □

Chapter 5

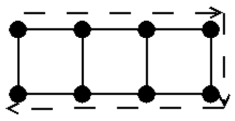
Hamiltonian graphs

Definition 5.1. *A Hamiltonian path in G is a path that contain every vertex of G exactly once.*

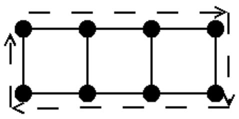
A Hamiltonian cycle is a closed Hamiltonian path.

A graph which contains a Hamiltonian cycle is called a Hamiltonian graph.

Example 5.2. *1. A Hamiltonian path:*

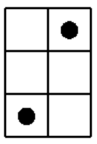


2. A Hamiltonian cycle:

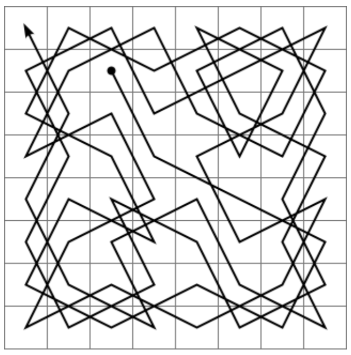


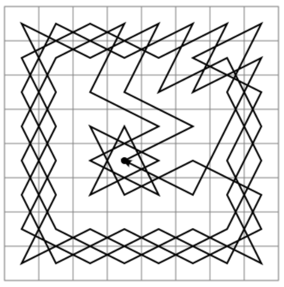
3. On a chessboard, can a knight visit every square exactly once and finish at its starting point?

Recall that a knight move is:



The answer is yes:



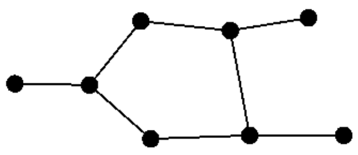


In terms of graph theory: Let $G = (V, E)$ where V is the set of all sqaures of the chessboard, and there is an edge between two vertices if we can go from one to the other by a knight move.

A solution to this problem is a Hamiltonian cycle. Observe that there are several different solutions to this problem.

4. *It is related to the traveling salesman problem: A salesman wants to visit some cities exactly once and go back to his starting point. What is the path with minimum distance? (mathematically formulated by Hamilton)*

   On a weighted graph, it means finding a Hamilton cycle of minumum weight.

5. 

   This graph is not Hamiltonian: It has vertices of degree 1 and a vertex of degree 1 cannot be in a cycle. So for a graph to be Hamiltonian we need each vertex to have degree at least 2. But it is not enough (we will see graphs like this that are not Hamiltonian).

6. *A circuit is a Hamiltonian graph (obviously).*

7. *The complete graph K_n for $n \geq 3$ is Hamiltonian:*

- *Enumerate the vertices from* 1 *to n (in any way).*
- *Start at the first one, go to the second, then the third, ..., then the nth one, then go back to the first. It can be done since any two different vertice are always connected.*

Theorem 5.3. *Let $G = (V, E)$ be a Hamiltonian graph. Then for any subset S of V with $S \neq \emptyset$ and $S \neq V$, we have*

$$\omega(G \setminus S) \leq |S|.$$

(Recall that $\omega(G \setminus S)$ denotes the number of components of $G \setminus S$.)

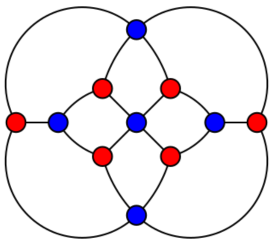
Proof. Let $n = |V|$ and let $G_1, \dots, G_k$ be the components of $G \setminus S$. Let $C = v_1 v_2 \dots v_n v_1$ be any Hamiltonian cycle in G, labelled such that $v_1 \in G_1$.

Let $i \in \{1, \dots, k\}$ and consider the component G_i. Since every vertex of G appears in C, and at least one element of C is not in G_i (for instance an element of S), we can define u_i to be the last element of C that belongs to G_i. Let s_i be the vertex after u_i in C.

Then $s_i \notin G_i$, which implies $s_i \in S$ (otherwise $s_i \in G \setminus S$, the edge $u_i s_i$ is in $G \setminus S$, so s_i is connected to u_i and thus belongs to G_i, a contradiction).

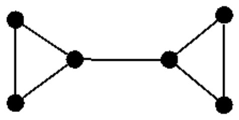
So each component G_i of $G \setminus S$ defines an element $s_i \in S$, and $s_i \neq s_j$ if $i \neq j$ (if $s_i = s_j$ for $i \neq j$ then $u_i = u_j \in G_i \cap G_j$, impossible). The map $\{1, \dots, k\} \to S$, $i \mapsto s_i$ is thus injective, proving that $k \leq |S|$. □

Example 5.4. *The Herschel graph:*



is not Hamiltonian: If we take for S the set of blue vertices, then $G \setminus S$ is so $\omega(G \setminus S) = 6 > |S| = 5$.

Exercise 5.5. *Show that this graph is not Hamiltonian:*



Theorem 5.6 (Bondy, Chvátal, 1972)**.** *Let G be a graph of order n and let u, v be any non-adjacent vertices of G such that $d(u) + d(v) \geq n$. Then G is Hamiltonian if and only if $G \cup \{uv\}$ is Hamiltonian.*

Proof. "⇒" Let C be a Hamilton cycle in G. Then C is also a Hamilton cycle in $G \cup \{uv\}$.

"⇐" Suppose that G is not Hamiltonian. Then any Hamilton cycle in $G \cup \{uv\}$ must include the edge uv (otherwise it would be a Hamilton cycle in G).

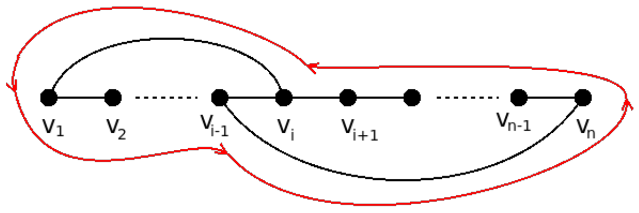
Removing the edge uv, we get a path from u to v in G that contains every vertex exactly once (a Hamiltonian path). Let

$$\underbrace{v_1}_{=u} v_2 \ldots v_{n-1} \underbrace{v_n}_{=v}$$

be this path.

If v_1 is adjacent to v_i, then v_n cannot be adjacent to v_{i-1}, because otherwise

$$v_1 v_2 \ldots v_{i-1} v_n v_{n-1} \ldots v_{i+1} v_i v_1$$



would be a Hamilton cycle in G, a contradiction. So whenever v_1 is adjacent to v_i, v_n is not adjacent to v_{i-1}.

Therefore, if $k = d(v_1)$ there are k vertices to which v_1 is adjacent and k vertices to which v_n is not adjacent. So $d(v_n) \leq n - 1 - k$. But then

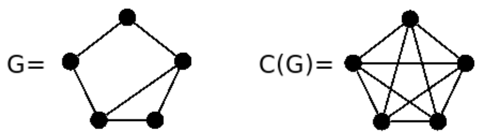
$$d(u) + d(v) = d(v_1) + d(v_n) \leq k + n - 1 - k = n - 1 < n,$$

contradicting the hypothesis. □

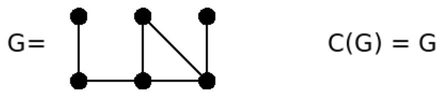
Conclusion: Whenever we find two vertices u, v such that $d(u)+d(v) \geq n$, we can add the edge uv and the answer to the question "is the graph Hamiltonian?" stays the same. We can keep doing this until we cannot add any new such edge:

Definition 5.7. *Let G be a graph of order n. The closure of G is the graph $c(G)$ obtained by successively joining pairs of non-adjacent vertices u, v such that $d(u) + d(v) \geq n$ (degree computed in the latest graph obtained), until it is no longer possible to do so.*

Example 5.8. *1.*



2.



We must show that $c(G)$ does not depend on the order in which the edges are added to G (otherwise we could have several possible $c(G)$ for a given graph G).

Lemma 5.9. *$c(G)$ does not depend on the order in which we add the vertices.*

Proof. Suppose G_1, G_2 are obtained from G as described in Definition 5.7. We show that $G_1 = G_2$. They have the same vertices, so we have to show that they have the same edges.

Let $e_1, \dots, e_r$ and $f_1, \dots, f_s$ be the sequences of edges added to G to obtain G_1, respectively G_2. We show that all e_i are edges of G_2 and all f_i are edges of G_1:

Suppose that some edge in $e_1, \dots, e_r$ is not in G_2 and let $e_{k_1} = uv$ be the first such edge. Let $H = G \cup \{e_1, \dots, e_k\}$. Since $e_{k+1} = uv$ is added next, we have $d_H(u) + d_H(v) \geq n$ (where d_H is the degree computed in H). But H is a subgraph of G_2 since $e_1, \dots, e_k$ appear in G_2, so $d_{G_2}(u) + d_{G_2}(v) \geq n$, and we will thus join u and v at some point in the construction of G_2, contradiction.

Therefore $e_1, \dots, e_r$ are all in G_2. Similarly $f_1, \dots, f_s$ are all in G_1. □

Corollary 5.10. *G is Hamiltonian if and only if $c(G)$ is Hamiltonian.*

(This is clear by Theorem 5.6.)

Corollary 5.11. *Let G be a graph of order $n \geq 3$ such that $c(G) = K_n$. Then G is Hamiltonian.*

Corollary 5.12 (Ore's theorem, 1963)**.** *Let G be a graph of order $n \geq 3$ such that the sum of the degrees of every pair of non-adjacent vertices is at least n. Then G is Hamiltonian.*

Proof. In this case we have $c(G) = K_n$. □

Corollary 5.13 (Dirac's theorem, 1952)**.** *Let G be a graph of order $n \geq 3$. If the degree of every vertex is at least $n/2$ then G is Hamiltonian.*

This leads to a more general sufficient condition for a graph to be Hamiltonian:

Theorem 5.14 (Chvátal, 1972)**.** *Let $G = (V, E)$ be a connected graph of order $n \geq 3$ and degree sequence $d_1 \leq d_2 \leq \cdots \leq d_n$ such that for every k with $1 \leq k < n/2$ at least one of the follwing is true:*

$$(i)\ d_k > k \qquad\qquad (ii)\ d_{n-k} \geq n - k.$$

Then G is Hamiltonian.

Proof. We show that $c(K) = K_n$: Suppose that $c(G) \neq K_n$. if w is a vertex in G, we denote by $d'(w)$ the degree of w in $c(G)$. Let $d'_1 \leq d'_2 \leq \cdots \leq d'_n$ be the degree sequence of $c(G)$. Note that $d(w) \leq d'(w)$ and $d_i \leq d'_i$.

Let $u, v \in V$ be non-adjacent in $c(G)$, and choose them such that

$$d'(u) + d'(v) \text{ is as large as possible.} \tag{5.1}$$

Choose the labels u, v such that $d'(u) \leq d'(v)$. Note that

$$d'(u) + d'(v) < n, \tag{5.2}$$

otherwise u and v would be connected in $c(G)$. Thus $2d'(u) \leq d'(u) + d'(v) < n$ and $d'(u) < n/2$.

Let $k = d'(u)$. Since G is connected, $k \geq 1$, so

$$1 \leq k < n/2. \tag{5.3}$$

We show that k does not satify conditions (i) and (ii), thus reaching a contradiction:

For any vertex x, let $N(x)$ be the set of vertices different from x that are not adjacent to x in $c(G)$. Then

$$|N(x)| = n - d'(x) - 1. \tag{5.4}$$

Let $w \in N(v)$. Then $d'(w)\ led'(u)$ (otherwise $d'(w) + d'(v) > d'(u) + d'(v)$, contradicting (5.1)). Similarly, if $x \in N(u)$ we have $d'(x) \leq d'(v)$, so for every $x \in N(u) \cup \{u\}$ we have

$$d'(x) \leq d'(v) \tag{5.5}$$

(recall that $d'(u) \leq d'(v)$).

By (5.2) we have $d'(u) < n - d'(v)$, so $d'(u) \leq n - d'(v) - 1 = |N(v)|$ i.e., $k \leq |N(v)|$, we have at least k vertices in $N(v)$.

Consider now a vertex $w \in N(v)$. We know that $d'(w) \leq d'(u) = k$, so we have at least k vertices in $c(G)$ with degree at most k, and thus in the degree sequence of $c(G)$ at least $d'_1, \dots, d'_k$ are at most k. In particular

$$d_k \leq d'_k \leq k. \tag{5.6}$$

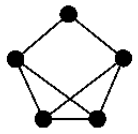
Using (5.3) with (5.6) we get $d_k < n/2$, so condition (i) is not true.

With (5.2) we obtain $d'(v) < n - d'(u) = n - k$, and by (5.5) $d'(x) < n - k$ for every $x \in N(u) \cup \{u\}$.

$$\begin{aligned} |N(u) \cup \{u\}| &= |N(u)| + 1 \\ &= (n - 1 - d'(u)) + 1 \quad \text{by (5.4)} \\ &= n - d'(u) = n - k. \end{aligned}$$

Therefore we have at least $n - k$ vertices of degree less than $n - k$ in $c(G)$, so at least $d'_1, d'_2, \dots, d'_{n-k}$ are less that $n - k$. In particular $d_{n-k} \leq d'_{n-k} < n - k$, so condition (ii) is also not satisfied. □

Example 5.15. *1.*



$n = 5$, degree sequence $2, 3, 3, 3, 3$. Since $n = 2$ we need to consider $k = 1, 2$.

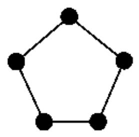
$$d_1 = 2 > 1, \qquad \textit{and } d_2 = 3 \geq 2,$$

so this graph is Hamiltonian by Chvátal's theorem.

In this case we could also use Ore's theorem or simply check that $c(G) = K_5$.

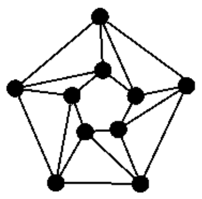
See exercise sheets for a graph where Chvátal's theorem works, but not Ore's.

2. It is possible for a graph to be Hamiltonian without satisfying Chvatal's condition:

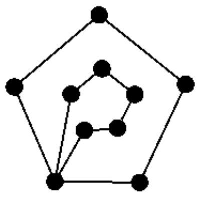


G is clearly Hamiltonian, with degree sequence $2, 2, 2, 2, 2$ and $n = 5$. So $1 \leq k < n/2$ means $k = 1, 2$. We have $d_1 = 2 > 1$, but $d_2 = 2 \not> 2$ and $d_{5-2} = d_3 = 2 \not\geq 3$.

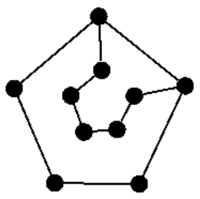
We have seen two notions of "traversability": Edgewise (Eulerian graphs) and vertexwise (Hamiltonian graphs). It is easy to check if a graph is Eulerian, harder to check if it is Hamiltonian. And there is no obvious link between these two notions:



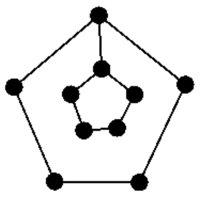
Eulerian and Hamiltonian



Eulerian, not Hamiltonian



Hamiltonian, not Eulerian

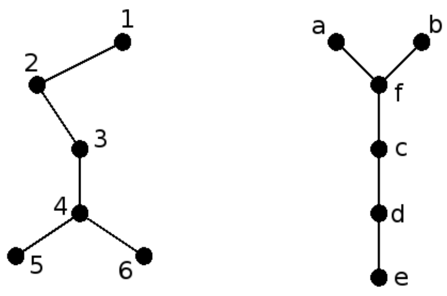


Neither

Chapter 6

Graph isomorphisms

We already used this concept without defining it formally. The graphs



are "the same".

Definition 6.1. *Let $G_q = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be graphs. We say that G_1 is isomorphic to G_2 if an d only if there is a bijection $f : V_1 \to V_2$ such that, for every $u, v \in V_1$:*

$$uv \in E_1 \Leftrightarrow f(u)f(v) \in E_2.$$

We write $G_1 \cong G_2$ and say that f is an isomorphism from G_1 to G_2 (or between G_1 and G_2).

In the example above, the map f could be:

$$5 \mapsto a,\ 6 \mapsto b,\ 4 \mapsto f,\ 3 \mapsto c,\ 2 \mapsto d,\ 1 \mapsto e$$

(there may be more than one such map f).

Remark 6.2. *In case of a multigraph (where there can be parallel edges) an isomorphism f from G_1 to G_2 is given by two bijections: $f_v : V_1 \to V_2$ and $f_e : E_1 \to E_2$ such that, for every $u, v \in V_1$ and $e \in E_1$:*

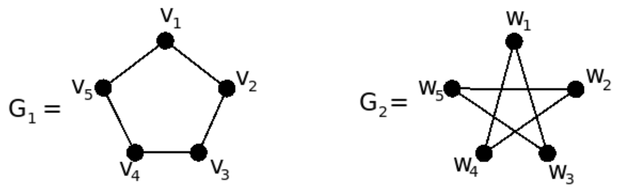
$$e \text{ is an edge between } u \text{ and } v \Leftrightarrow f_e(e) \text{ is an edge between } f_v(u) \text{ and } f_v(v).$$

Remark 6.3. *1. Isomorphic graphs have:*
-The same order;
-The same size;
-The same degree sequence (it follows from the definition that $d(u) = d(f(u))$);
-etc.

2. *In fact, if $G_1 \cong G_2$, every graph-theoretic property of G_1 can be "translated" into the corresponding property of G_2, using f.*

3. *If f is an isomorphism from G_1 to G_2, then f^{-1} is an isomorphism from G_2 to G_1.*

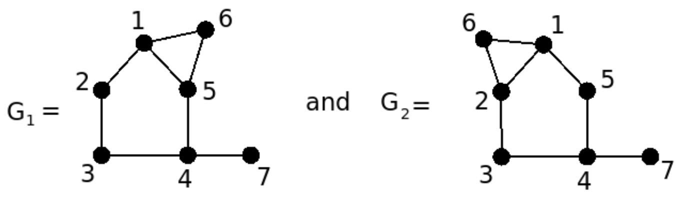
Example 6.4. *1.*



Here $G_1 \cong G_2$. A possible isomorphim is

$$f(v_1) = w_1,\ f(v_2) = w_3,\ f(v_3) = w_5,\ f(v_4) = w_2,\ f(v_5) = w_4.$$

2. The graphs



are not isomorphic: Suppose for the sake of contradiction that they are, and let f be the isomorphism. Since $d(u) = d(f(u))$ we must have $d(f(1)) = d(1) = 3$, $d(f(2)) = d(2) = 2$, $d(f(3)) = 2$, $d(f(4)) = 3$, $d(f(5)) = 3$, $d(f(6)) = 2$, $d(f(7)) = 1$. Since 7 is the only vertex of degree 1 in G_2, we must have $f(7) = 7$. 47 is an edge in G_1, so $f(4)f(7) = f(4)7$ is an edge in G_2. So $f(4) = 4$.

5 is adjacent to 4, so $f(5)$ is adjacent to $f(4) = 4$, so $f(5) = 5$ or $f(5) = 3$. But $d_{G_1}(5) = 3$ and $d_{G_2}(5) = 2$, $d_{G_2}(3) = 2$, so we cannot have $f(5) = 5$ or $f(5) = 3$. So G_1 and G_2 are not isomorphic.

3. *In general, the question of checking if two graphs are isomorphic is very difficult.*

Chapter 7

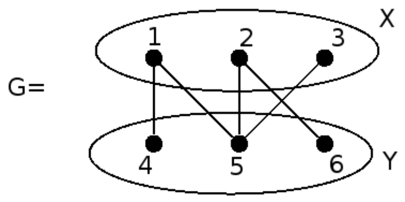
Bipartite graphs

Definition 7.1. *A graph $G = (V, E)$ is bipartite if the following hold*

1. *$V = X \cup Y$ for some sets X, Y with $X \cap Y = \emptyset$;*
2. *For every edge uv in E, one of u, v is in X and the other in Y.*

The pair (X, Y) is called a bipartition of G.

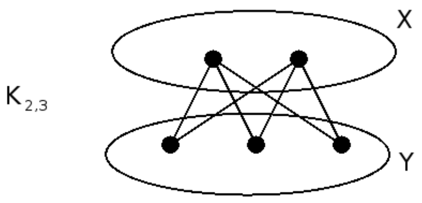
Example 7.2.



is bipartite with $X = \{1, 2, 3\}$ and $Y = \{4, 5, 6\}$.

Definition 7.3. *If X and Y are two disjoint sets such that X has r elements and Y has s elements, the complete bipartite graph $K_{r,s}$ is the graph with set of vertices $X \cup Y$ and set of edges $\{uv \mid u \in X,\ v \in Y\}$.*

Example 7.4.



Definition 7.5. *A cycle C in a graph is called an odd cycle if C has an odd number of edges.*

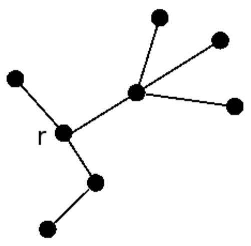
Theorem 7.6. *A graph is bipartite if and only if it contains no odd cycles.*

To obtain a simple proof of this theorem, we require a little bit more knowledge about trees.

Recall that if T is a tree and x, y are vertices in T, there is a unique path in T from x to y. We denote this path by xTy.

It is sometimes convenient to single out a vertex in a tree T. Such a vertex is then called the root of T (it can be any vertex). A tree T with a fixed root r is called a rooted tree.

Example 7.7.



(We can take any vertex as a root.)

Proof of Theorem 7.6. "$\Rightarrow$" If $C = u_1 \ldots u_r$ (with $u_r = u_1$) is an odd cycle, then u_{r-1} and u_1 must be both in X or both in Y, impossible.

"$\Leftarrow$" A graph is bipartite if and only if its components are bipartite, so we can assume that G is connected. The result is clear if G has only 2 vertices, so we will assume that G has at least 3 vertices.

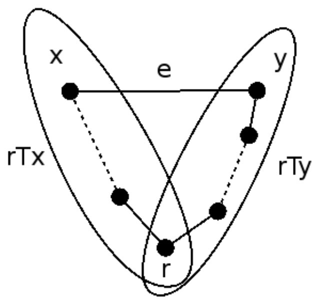
Let T be a spanning tree of G, and pick a root r of T. For $v \in V$ the path rTv has odd or even length. We define

$$X = \{v \in V \mid rTv \text{ has odd length}\},$$

$$Y = \{v \in V \mid rTv \text{ has even length}\}.$$

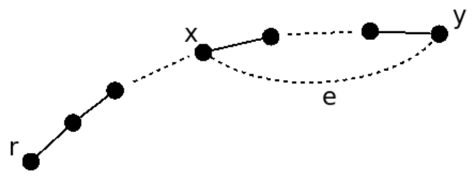
We check that (X, Y) is a bipartition of G: Let $e = xy$ be an edge in G. We consider two cases

1) $e \in T$: Observe that $x \in rTy$ or $y \in rTx$ (otherwise



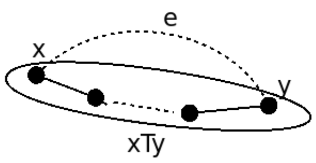
and we obtain a cycle in T, contradiction).

We assume $x \in rTy$ (the other case is similar).



Since $e = xy$ is an edge in T, we must have $rTy = rTxy$ (otherwise we get a cycle in T). So one of x, y is in X and the other in Y.

2) $e \notin T$. Then $xTy \cup \{e\}$ is a cycle:



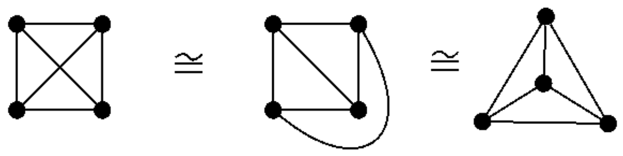
The edges in xTy are all in T so, by case 1, the vertices in xTy alternate between X and Y. Since $xTy \cup \{e\}$ is even by hypothesis, we get that xTy has odd length, and thus that x and y cannot be both in X or both in Y. $\square$

Chapter 8

Planar graphs

Definition 8.1. *A graph is called planar if it can be drawn in the plane so that edges intersect only at vertices to which they are incident.*

Example 8.2. *Different representations of K_4:*

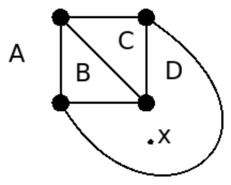


Definition 8.3 (Continued)**.** *Such a representatiopn of G without edge crossings is called a planar representation (or planar embedding) of G.*

Definition 8.4. *A planar representation of G divides the planed into regions called faces:*

Let x be a point in the plane. The face of G (more precisely: of this planar representation of G) containing x is the set of points of the plane that can be reached from x by a curve that does not contain a vertex or a point from an edge.

Example 8.5.



This planar representation of K_4 divides the plane into 4 faces: A, B, C and D. The face containing x is D.

The faces depend on the particular plane representation, not only on the graph.

Each plane representation partitions the plane into finitely many faces. Exactly one is infinite in size, we call it the exterior face. The other faces are called interior faces.

Do two different plane representations give the same number of faces? (Yes, Euler, see theorem 8.13.

We denote by $F(G)$ the set of faces of a given planar representation of G (we will make the slight abuse of notation of still calling it G).

Definition 8.6. *Let G be a graph. An edge e of G is called a cut-edge if $\omega(G \setminus \{e\}) > \omega(G)$.*

(Recall that $\omega(G)$ is the number of components of G.)

Proposition 8.7. *An edge e of G is a cut edge if and only if e is not contained in any cycles of G.*

Sketch of the proof: Let K be the component of G containing $e = uv$.

"$\Rightarrow$" Then $\omega(K \setminus \{e\}) > 1$. If e is in a cycle in K, then $K \setminus \{e\}$ is still connected, contradiction.

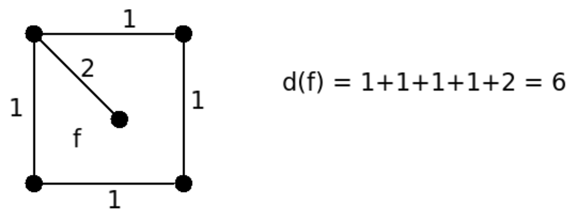
"$\Leftarrow$" If e is not a cut-edge $\omega(K \setminus \{e\}) = 1$, so $K \setminus \{e\}$ is connected. Let $uw_1 \dots w_k v$ be a path in $K \setminus \{e\}$. Then $uw_1 \dots w_k vu$ is a cycle in K, contradiction. $\square$

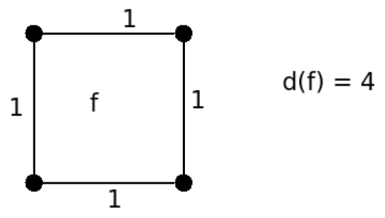
We record the following reformulation of Theorem 4.4 1. $\Leftrightarrow$ 2.

Proposition 8.8. *A connected graph is a tree if and only if every edge is a cut edge.*

Definition 8.9. *Let G be a plane representation of a planar graph. An edge e is called incident with a face f if e is part of the boundary of f.*

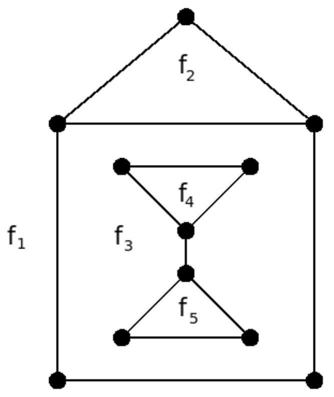
Definition 8.10. *The degree $d(f)$ of a face f is the number of edges incident with f, where each edge entirely inside f (called a pendant edge) is counted twice.*





Why counted twice? Because we count the sides of the edge that are in contact with f (see the proof of Proposition 8.12), in case of a pendant edge, both sides of the edge in contact with f.

Example 8.11.



$$d(f_1) = 5,\ d(f_2) = 3,\ d(f_3) = 12,\ d(f_4) = 3,\ d(f_5) = 3.$$

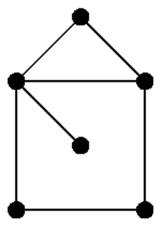
Proposition 8.12. *Let F be the set of faces of a planar graph $G = (V, E)$. Then*

$$\sum_{f \in F} d(f) = 2|E|.$$

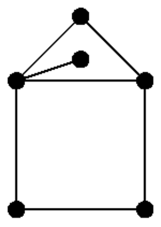
Proof. If an edge is between 2 faces, it is counted twice on the left-hand side (once for each face).

If a face is entirely inside a face f, it is counted twice in $d(f)$ by definition. □

Observe that we can have different planar representations of the same graph, with different degree sequences of the face:



degree sequence of the faces: $5, 3, 6$,

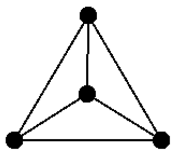


degree sequence of the faces: $5, 5, 4$.

Theorem 8.13 (Euler's formula)**.** *Let $G = (V, E)$ be a connected planar graph, and consider a planar representation of G with set of faces F. Then*

$$|V| - |E| + |F| = 2.$$

Example 8.14.



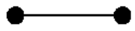
$|V| - |E| + |F| = 4 - 6 + 4 = 2.$

Proof of Theorem 8.13. By induction on $|E|$.

- If $|E| = 0$, then G is

  and $|V| - |E| + |F| = 1 + 1 = 2.$

- If $|E| = 1$, then G is

  and $|V| - |E| + |F| = 2 - 1 + 1 = 2.$

- If $|E| > 1$. We consider two cases.

  Case 1: If G is a tree. Then $|E| = |V| - 1$ and $|F| = 1$ since a tree has only one face (we admit this: a tree has only the exterior face since to have an interior face it would need to have a cycle). So $|V| - |E| + |F| = 2$.

  Case 2: If G is not a tree. Then (since G is connected) G contains a cycle C. Let e be an edge in C. Since e is not a cut edge, $G \setminus \{e\}$ is connected, and has of course $|E| - 1$ edges.

  Since e is on a cycle, it separates 2 faces (we admit this), so removing e merges these two faces, and thus $G \setminus \{e\}$ has $|F| - 1$ faces.

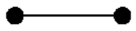
  By induction on $|E|$, the result is true for the graph $G \setminus \{e\}$ i.e., $|V| - (|E| - 1) + (|F| + 1) = 2$ i.e., $|V| - |E| + |F| = 2$.

□

Corollary 8.15. *Let $G = (V, E)$ be a planar graph with $|V| \geq 3$. Then $|E| \leq 3|V| - 6$.*
(In other words: We cannot have "too many" edges compared to the number of vertices.)

Proof. We consider two cases.
Case 1: G is connected. The result is true if $|E| < 3$ (since $3 \leq 3 \cdot 3 - 6$), so we assume $|E| \geq 3$.

Let F be the set of faces of a planar embedding of G. For any $f \in F$, $d(f) \geq 3$ (if f is an interior face it has at least 3 edges in its boundary, since parallel edges are not allowed; if f is the exterior face and $d(f) = 2$ then G is



contradicting that $|E| \geq 3$).

So $\sum_{f \in F} d(f) \geq 3 \cdot |F|$. But we know that $\sum_{f \in F} = 2|E|$, so $2|E| \geq 3|F|$ i.e., $|F| \leq \frac{2}{3}|E|$. Using now Euler's formula, we obtain

$$2 = |V| - |E| + |F| \leq |V| - |E| + \frac{2}{3}|E| = |V| - \frac{1}{3}|E|.$$

Therefore $|E| \leq 3|V| - 6$.

Case 2: If G is not connected. Write $G = G_1 \cup \cdots \cup G_k$, where the G_i are the components of G. Connect the components by adding $k-1$ new edges while still keeping the graph planar (for instance: arrange the components around a circle, and connect G_1 to G_2, G_2 to G_3, etc). The resulting graph is connected and has $|E| + (k-1)$ edges. By the previous case $|E| + k - 1 \leq 3|V| - 6$ and the result follows. □

Corollary 8.16. *Let $G = (V, E)$ be a planar graph. Then some vertex of G has degree at most* 5.

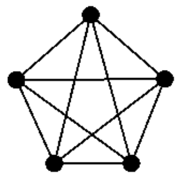
Proof. Let $n = |V|$. If $n = 1$ or $n = 2$ the result is clear, so assume $n \geq 3$. Let d be the smallest degree of any vertex in G. We have $E \leq 3n - 6$ and

$$nd \leq \sum_{v \in V} d(V) = 2|E| \leq 2(3n-6),$$

i.e., $nd \leq 6n - 12$ so $d \leq 6 - \frac{12}{n}$ and thus $d \leq 5$. □

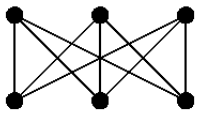
Corollary 8.17. *The graph K_5 is not planar.*

Recall that K_5 is



Lemma 8.18. *The graph $K_{3,3}$ is not planar.*

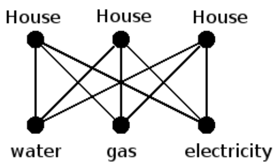
Recall that $K_{3,3}$ is



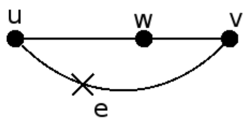
Proof. Suppose it is and consider a planar representation of $K_{3,3}$. Since $K_{3,3}$ is bipartite, it contains no odd cycles, so there are no cycles of length less than 4 (since a cycle of length less than 4 has length 3).

So $d(f) \geq 4$ for every face, and $2|E| = \sum_{f \in F} d(f) \geq 4 \cdot |F|$. Therefore $|E| \geq 2|F|$. Since $K_{3,3}$ has 9 edges we get $9 \geq 2 \cdot |F|$ i.e., $|F| \leq 4$. But $|V| - |E| + |F| = 2$ i.e., $6 - 9 + |F| = 2$, so $|F| = 5$, contradiction. □

Example 8.19. *We want to provide gas, water and electricity to 3 houses. It cannot be done without crossing some of the lines, because it corresponds to the graph $K_{3,3}$:*

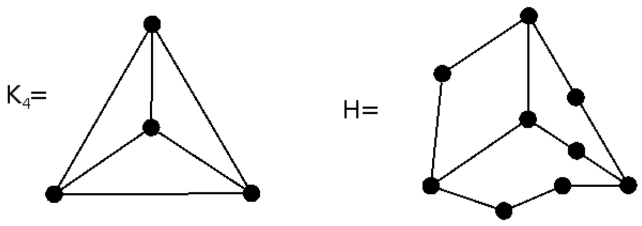


Definition 8.20. *Let $G = (V, E)$ be a graph, and let $e = uv$ be an edge in G. We say that we subdivide the edge e when we delete it and replace it by a path uwv, where w is a new vertex (i.e., $w \notin V$):*



We say that e has been subdivided. A subdivision of G is a graph that can be obtained from G by a sequence of subdivisions.

Example 8.21.



H is a subdivision of K_4.

The next two lemmas are obvious.

Lemma 8.22. *Let G be a graph. G is planar if and only if every subdivision of G is planar if and only if one of the subdivisions of G is planar.*

Lemma 8.23. *Let G be a graph. G is planar if and only if every subgraph of G is planar.*

(Recall that G is a subgraph of G.)

Definition 8.24. *Any subdivision of K_5 or of $K_{3,3}$ is called a Kuratowski graph. A K-subgraph is a subgraph that is a Kuratowski graph.*

By Lemma 8.22, a Kuratowski graph is non-planar, and by Lemma 8.23, a graph with a K-subgraph is non-planar. The converse is also true:

Theorem 8.25 (Kuratowski, 1930)**.** *A graph is planar if and only if it does not contain a K-subgraph.*

No proof (too long).

Chapter 9

Networks

Stuttgart 1968: A new road is built. Consequence: The traffic gets worse until the road is closed.
New York 1990: 42nd street is closed. Consequence: The traffic improves.

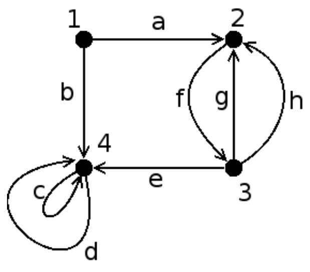
(Braess paradox)

We are not quite going to look into this, because it is also caused by drivers trying to maximise apparent immediate gain rather than thinking about longer-term consequences, but it illustrates why looking at how "things" can move in a network is interesting.

Definition 9.1. *A directed pseudograph (DPG) is a triple (V, E, ψ), where*

- *V is a non-empty set of points, called vertices.*
- *E is a set, disjoint from V, whose elements are called arcs.*
- *ψ is a map, called the incidence function, that associates to each arc in E an ordered pair of vertices of V (the vertices may be equal).*

Example 9.2.



$V = \{1, 2, 3, 4\}$, $E = \{a, b, c, d, e, f, g, h\}$

$\psi(a) = (1, 2)$, $\psi(b) = (1, 4)$, $\psi(c) = (4, 4)$, $\psi(d) = (4, 4)$, $\psi(e) = (3, 4)$, $\psi(f) = (2, 3)$, $\psi(g) = (3, 2)$, $\psi(h) = (3, 2)$.

We cannot simply represent an arc by, for instance, $(3, 2)$, because there can be several such arcs. So we give them all a different name and use the incidence function to tell us from where to where they go.

Definition 9.3. *Let $D = (V, E, \psi)$ be a DPG.*

1. *A directed walk W in D is a sequence $W = (v_0, a_1, v_1, a_2, \ldots, a_k, v_k)$ with $v_i \in V$, $a_i \in E$, and such that $\psi(a_i) = (v_{i-1}, v_i)$ for $i = 1, \ldots, k$.*

   We define similarly directed paths, cycles, etc.

2. *A walk in D is a sequence $W = (v_0, a_1, v_1, a_2, \ldots, a_k, v_k)$ with $v_i \in V$, $a_i \in E$, and such thatfor $i = 1, \ldots, k$, $\psi(a_i) = (v_{i-1}, v_i)$ or $\psi(a_i) = (v_i, v_{i-1})$ (in this later case we say that a_i is a reverse arc of W).*

   We define similarly paths, cycles, etc.

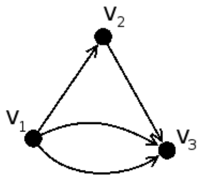
Example 9.4. *Using the DPG from the previous example, $(1, b, 4, c, 4, d, 4, c, 4)$ is a directed path, and $(1, b, 4, c, 4, e, 3)$ is a path which is not a directed path (e is a reverse arc).*

Definition 9.5. *Let $D = (V, E, \psi)$ be a DPG. For $v \in V$ we define*

$$\begin{aligned} d_+(v) &= \text{ the in-degree of } v \\ &= \text{ the number of arcs that end at } v \end{aligned}$$

$$\begin{aligned} d_-(v) &= \text{ the out-degree of } v \\ &= \text{ the number of arcs starting at } v \end{aligned}$$

Example 9.6.



$d_+(v_1) = 0$, $d_-(v_1) = 3$, $d_+(v_2) = 1$, $d_-(v_2) = 1$, $d_+(v_3) = 3$, $d_-(v_3) = 0$.

Proposition 9.7. *Let $D = (E, V, \psi)$ be a DPG. Then*

$$\sum_{v \in V} d_+(v) = \sum_{v \in V} d_-(v) = |E|.$$

Proof. Exercise, see Proposition 1.21. $\square$

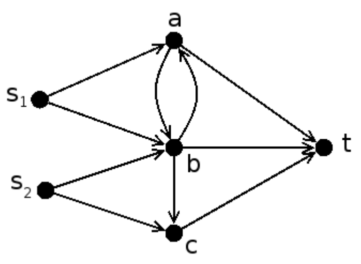
Definition 9.8. *A network is a DPG (V, E, ψ) together with two non-empty subsets of V:*

- *S, called the source, such that $d_+(v) = 0$ for every $v \in S$;*
- *T, called the sink, such that $d_-(v) = 0$ for every $v \in T$.*

The remaining vertices are called intermediate vertices.

(In networks, vertices are often called nodes.)

Example 9.9. *$N = (E, V, \psi, S, T)$ is a network:*

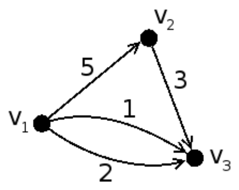


with $S = \{s_1, s_2\}$, $T = \{t\}$ and intermediate vertices a, b, c.

Intuitively: S gives the entrance points, and T the exit points of the network.

Definition 9.10. *Let $D = (V, E, \psi)$ be a DPG. A capacity on D is a function $c : E \to \mathbb{R}_+ = \{x \in \mathbb{R} \mid x > 0\}$.*

Example 9.11.



Intuitively, the capacity represents how much can travel along an arc.

Definition 9.12. *Let $N = (V, E, \psi, S, T)$ be a network and let c be a capacity on N. A flow on N is a function $f : E \to \mathbb{R}_+ \cup \{0\}$ such that*

1. *$0 \leq f(e) \leq c(e)$ for every $e \in E$.*
2. *For every intermediate vertex v, the in-flow:*

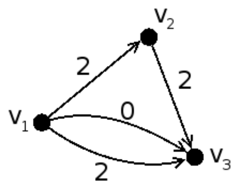
$$f_+(v) \overset{def}{=} \sum_{e \in E, \psi(e)=(u,v)} f(e)$$

is equal to the out-flow:

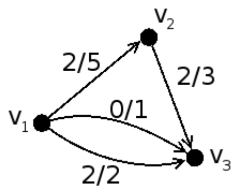
$$f_-(v) \overset{def}{=} \sum_{e \in E, \psi(e)=(v,u)} f(e)$$

(this is known as Kirchhof's rule).

Example 9.13. *With the capacity given by the previous example, this is a flow (where $S = \{v_1\}$ and $T = \{v_3\}$:*



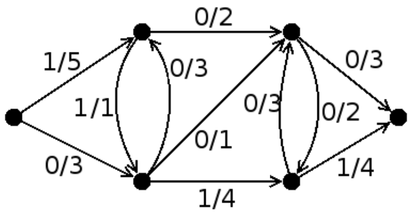
We often write



to indicate both the capacity and the flow at the same time on the drawing. (So, for instance, 2/5 is not a fraction. It just means that the arc has capacity 5 and that the flow on it is 2.)

The flow represents what is actually circulating in the network.

A more complicated example:



Question: For a given network and capacity, what is the "largest" possible flow, and how to find it?

Examples of applications: Electric circuits, pipelines, transportation systems, etc.

Lemma 9.14. *Let f be a flow on a network $N = (V, E, \psi, S, T)$ with capacity c. Then*

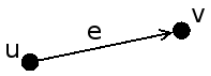
$$\underbrace{\sum_{s\in S} f_-(s)}_{\textit{out-flow from the source}} = \underbrace{\sum_{t\in T} f_+(t)}_{\textit{in-flow to the sink}} .$$

Proof. We have

$$\sum_{w\in V}(f_+(w) - f_-(w)) = \underbrace{\sum_{w\in I}(f_+(w) - f_-(w))}_{=0 \text{ by def of flow}} + \sum_{t\in T} f_+(t) - \sum_{s\in S} f_-(s).$$

So the result is true if we show that $\sum_{w\in V}(f_+(w) - f_-(w)) = 0$.

Consider an arc $e \in E$:



In the sum $\sum_{w\in V}(f_+(w) - f_-(w))$ the number $f(e)$ appears twice: Once as $f(e)$ in $f_+(v)-f_-(v)$, once as $-f(e)$ in $f_+(u)-f_-(u)$. Therefore $\sum_{w\in V}(f_+(w)-f_-(w)) = 0$. □

Notation 9.15. *We define, for $A, B \subseteq V$:*

- $f(A, B) = \sum_{e\in E, e \textit{ from } A \textit{ to } B} f(e)$.
- $f_-(A) = f(A, V \setminus A)$ *(what "leaves" A).*
- $f_+(A) = f(V \setminus A, A)$ *(what "enters" A).*

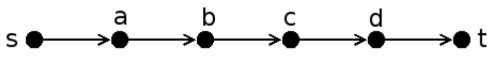
(The previous lemma says $f_-(S) = f_+(T)$.)

Lemma 9.16. *For any $A \subseteq V$:*

$$\sum_{v\in A} f_-(v) - f_+(v) = f_-(A) - f_+(A).$$

Proof. Exercise. □

Remark 9.17. *In general, $\sum_{v\in A} f_-(v) \neq f_-(A)$ and $\sum_{v\in A} f_+(v) \neq f_+(A)$. Try it for instance on*



for some sets A and some simples capacity and flow.

Definition 9.18. *The value of a flow f is*

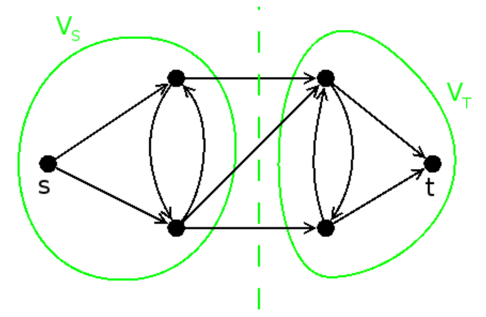
$$v(f) = f_-(S) = f_+(T).$$

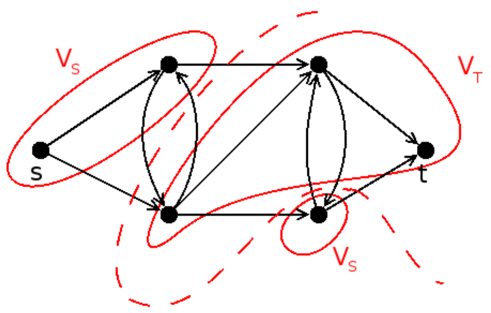
Can we construct a flow with maximal possible $v(f)$?

Definition 9.19. *A cut in a network $N = (V, E, \psi, S, T)$ is a pair (V_S, V_T) of subsets of V such that*

1. *$V = V_S \cup V_T$.*
2. *$V_S \cap V_T = \emptyset$ (so $V_S = V \setminus V_T$ and $V_T = V \setminus V_S$).*
3. *$S \subseteq V_S$.*
4. *$T \subseteq V_T$.*

Example 9.20.





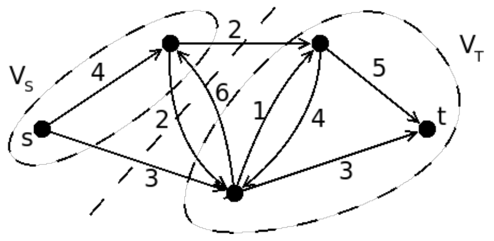
The green (V_S, V_T) and the red (V_S, V_T) are both cuts.

It is called a cut because it cuts N into two parts: one that contains S and the other that contains T.

Definition 9.21. *The capacity of a cut (V_S, V_T) is defined by*

$$c(V_S, V_T) = \sum_{e \in E, e \text{ from } V_S \text{ to } V_T} c(e).$$

Example 9.22.



Here $c(V_S, V_T) = 2 + 2 + 3 = 7$.

Lemma 9.23. *For any flow f and any cut (V_S, V_T) on N:*

$$v(f) = f_-(V_S) - f_+(V_S).$$

Proof. If $v \in V_S \setminus S (= I \cap V_S)$ we have $f_-(v) - f_+(v) = 0$ by definition of flow. So

$$\begin{aligned} f_-(V_S) - f_+(V_S) &= \sum_{v \in V_S} f_-(v) - f_+(v) \text{ by Lemma 9.16} \\ &= \sum_{v \in S} \big(f_-(v) - \underbrace{f_+(v)}_{=0 \text{ since } v \in S} \big) \\ &= f_-(S) \\ &= v(f) \end{aligned}$$

□

Notation 9.24. *We say that an arc $e \in E$ is*

- *f-zero if $f(e) = 0$;*
- *f-positive if $f(e) > 0$;*
- *f-saturated if $f(e) = c(e)$;*
- *f-unsaturated if $f(e) < c(e)$.*

Theorem 9.25. *For any flow f and any cut (V_S, V_T) on N:*

1. *$v(f) \leq c(V_S, V_T)$;*
2. *$v(f) = c(V_S, V_T)$ if and only if each arc from V_S to V_T is f-saturated and each arc from V_T to V_S is f-zero.*

Proof. 1. By Lemma 9.23 $v(f) = f_-(V_S) - f_+(V_S)$ where

$$\begin{aligned} f_-(V_S) &\overset{\text{def}}{=} f(V_S, V \setminus V_S) \\ &= f(V_S, V_T) \\ &= \sum_{e \in E, e \text{ from } V_S \text{ to } V_T} f(e) \\ &\leq \sum_{e \in E, e \text{ from } V_S \text{ to } V_T} c(e) \\ &= c(V_S, V_T). \end{aligned}$$

Since $f_+(V_S) \geq 0$ we obtain $v(f) \leq c(V_S, V_T)$.

2. To have $v(f) = c(V_S, V_T)$ we must have
   - $f_-(V_S) = c(V_S, V_T)$, so every arc from V_S to V_T must be f-saturated.
   - $f_+(V_S) = 0$, so every arc from V_T to V_S must be f-zero.

□

Definition 9.26. *1. A flow f on N is called a maximim flow if there is no flow f' on N such that $v(f') > v(f)$.*

2. A cut (V_S, V_T) on N is called a minimum cut if there is no cut (V'_S, V'_T) on N such that $c(V'_S, V'_T) < c(V_S, V_T)$.

As a consequence of theorem 9.25, if f^* is a maximum flow and (V^*_S, V^*_T) is a minimum cut, then

$$v(f^*) \leq c(V^*_S, V^*_T).$$

Corollary 9.27. *Let f be a flow and (V_S, V_T) be a cut such that $v(f) = c(V_S, V_T)$. Then f is a maximum flow and (V_S, V_T) is a minimum cut.*

Proof. Let f^* be a maximum flow and (V^*_S, V^*_T) be a minimum cut. Then

$$v(f) \leq v(f^*) \leq c(V^*_S, V^*_T) \leq c(V_S, V_T),$$

and $v(f) = c(V_S, V_T)$ by hypothesis. Therefore

$$v(f) = v(f^*) \text{ and } c(V_S, V_T) = c(V^*_S, V^*_T).$$

□

Let f be a flow in a network N with capacity c, and let P be a path in N. For an arc a in P we define:

$$i(a) = \begin{cases} c(a) - f(a) & \text{if } a \text{ is a forward arc in } P \\ f(a) & \text{if } a \text{ is a reverse arc in } P \end{cases}$$

and

$$i(P) = \min_{a \in P} i(a).$$

Intuitively, the idea is that, in case $i(P) > 0$, we might be able to add $i(P)$ to f on the arcs in P, and thus get a new flow with higher value (cf. Theorem 9.252.).

We say that P is f-saturated if $i(P) = 0$ and f-unsaturated if $i(P) > 0$. An f-incrementing path is an f-unsaturated path from some $x \in S$ to some $y \in T$ that only contains x and y as elements of $S \cup T$.

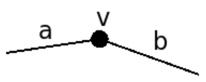
Lemma 9.28. *Let P be an f-incrementing path and define*

$$\tilde{f} : E \to \mathbb{R}_+ \cup \{0\}, \quad \tilde{f}(a) = \begin{cases} f(a) + i(P) & \textit{if } a \textit{ is a forward arc of } P \\ f(a) - i(P) & \textit{if } a \textit{ is a reverse arc of } P \\ f(a) & \textit{otherwise} \end{cases}$$

Then $\tilde{f}$ is a flow and $v(\tilde{f}) = v(f) + i(P)$. The flow $\tilde{f}$ is called the revised flow based on P.

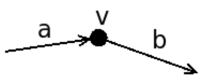
Proof. (1) The condition $0 \leq \tilde{f}(a) \leq c(a)$ follows from the definition of i.

(2) The condition $\sum_{a\in E,\ \psi(a)=(u,v)} \tilde{f}(a) = \sum_{a\in E,\ \psi(a)=(v,u)} \tilde{f}(a)$ (for $v \in I$):
If v is not in P, this is the same as for f.
If v is in P: Since $v \in I$ and P goes from S to T, P "traverses" v:



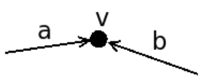
We consider several cases, depending on the "direction" of a, b:

- a and b are forward arcs of P:



  Then a adds $i(P)$ to the left hand side and b adds $i(P)$ to the right hand side. The equality is preseved.

- a is a forward are of P and b a reverse arc:



  Then a adds $i(P)$ to the left hand side and b adds $-i(P)$ to the left hand side. The equality is preserved.

- The other two cases are similar.

(3) $v(\tilde{f}) = v(f) + i(P)$:
Since P is an f-incrementing path, only the first arc of P is between S and I, so

$$v(\tilde{f}) = \sum_{x\in S, e\in E \text{ s.t. } \psi(e)=(x,u)} \tilde{f}(e) = \sum_{\cdots} f(e) + \underbrace{i(P)}_{\text{added by the first arc of } P} = v(f) + i(P).$$

$\square$

Theorem 9.29. *A flow f on N is a maximum flow if and only if N contains no f-incrementing path.*

Proof. "$\Rightarrow$" If f is a maximum flow then N contains no f-incrementing path by Lemma 9.28 (we would have $v(\tilde{f}) > v(f)$).

"$\Leftarrow$" Assume now that N contains no f-incrementing path. Define

$$V_S = S \cup \{\text{all the vertices } u \text{ such that there is an element } x \in S \\ \text{and an } f\text{-unsaturated path in } N \text{ from } x \text{ to } u\},$$

and

$$V_T = V \setminus V_S.$$

Since N has no f-incrementing path, no element of T is in V_S, so $T \subseteq V_T$ and (V_S, V_T) is a cut in N.

Fact: (i) Each arc from V_S to V_T is f-saturated.
(ii) Each arc from V_T to V_S is f-zero.
Proof of (i): Let a be an arc from $u \in V_S$ to $v \in V_T$. Since $u \in V_S$ there is an f-unsaturated path from some $x \in S$ to u. If a is f-unsaturated, then this path can be extended by a and would give an f-unsaturated path from x to v. So we would have $v \in V_S$, contradiction.
Proof of (ii): Similar, left as an exercise.

Using the Fact and Theorem 9.25, we obtain $v(f) = c(V_S, V_T)$, and by Corollary 9.27 f is a maximum flow (and (V_S, V_T) is a minimum cut). □

Remark 9.30. $(V_S, V \setminus V_S)$ *with* V_S *as in the proof is a minimal cut.*

This proof gives us the following: If we start with a maximum flow f, we get a minimal cut (V_S, V_T) such that $v(f) = c(V_S, V_T)$. Therefore

Theorem 9.31 (Max-flow, min-cut, Ford-Fulkerson 1956)**.** *In any network, the value of a maximum flow is equal to the capacity of a minimum cut.*

The Ford-Fulkerson algorithm.
It is easier to state using the following definition.

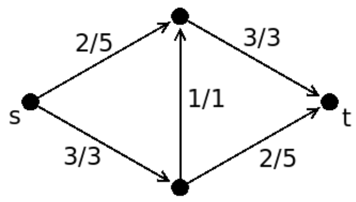
Definition 9.32. *Let N be a network and let f be a flow in N. The residual network of f is the network with:*

- *Same vertices, source and sink as N;*
- *Every arc $e \in E$ is given the new capacity*

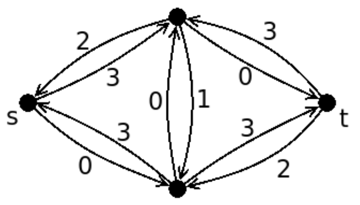
$$c_f(e) = c(e) - f(e);$$

- *For every arc in E from u to v we add an edge from v to u with capacity $f(e)$;*
- *Remove every arc e with $c_f(e) = 0$ (a capacity only takes positive values).*

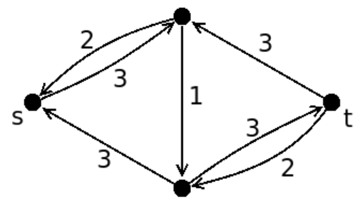
Example 9.33.



The residual network is



so



The algorithm:

1. Start with a flow f of value 0 (the zero flow).

2. Repear the following:

   (a) Build the residual network N_f of f.

   (b) Find a directed path P from some $s \in S$ to some $t \in T$ in N_f containing no other point from S or T. Stop if there is no such path (f is then a maximal flow).

   (c) Take $d = \min_{e \in P} c_f(e)$.

   (d) For every arc e of P, let e_N be the arc in the original network between the same vertices. We then add to the flow f at e:
   d if e and e_N go in the same direction,
   $-d$ is e and e_N go in opposite directions.

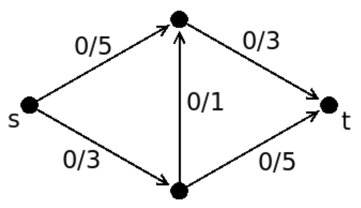
   (So we end up modifying the values of f along the path P.)

3. The final flow you obtained (before the residual network in which you could not find a path) is a maximum flow.

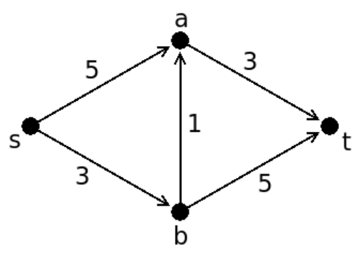
4. If at the end you wish to find a minimum cut, proceed as follows (cf. Remark 9.30): Working in the final residual network (the one where you could not find a path), put in the set V_S all the vertices of the source, and all the vertices that you can reach from a path starting in the source. Take the set of all remaining vertices for V_T (since you could not find a path from the source to the sink, V_T contains at least the sink).

   Then (V_S, V_T) is a minimum cut (again, see Remark 9.30).

We carry out the algorithm on the network above:
(1)



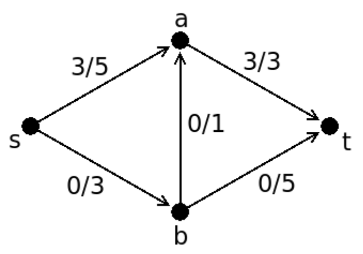
(2)(a)



(2)(b) Take $P = sat$ (I do not give a name to the edges in P since there are no parallel edges, so no choice).

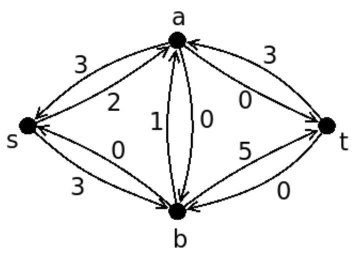
(2)(c) $d = 3$.

(2)(d)

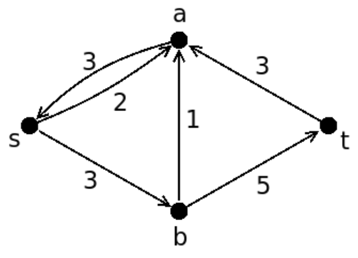


(new f)

(2)(a)



so

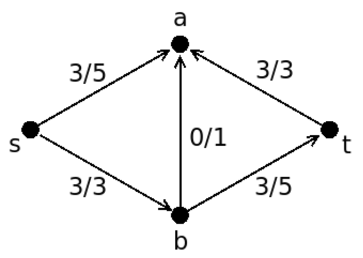


(=N_f)

(2)(b) Take for instance $P = sbt$.

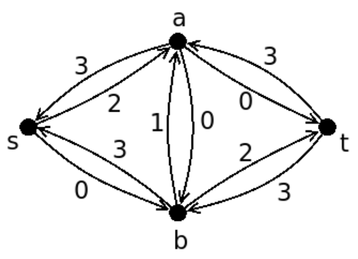
(2)(c) $d = 3$.

(2)(d)

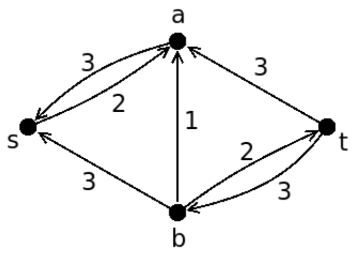


(new f)

(2)(a) N_f:



so



$(=N_f)$

There is no path P, so we stop and this f is a maximal flow, with value 6.

It is confirmed by a quick look at the original network: There is indeed a cut with capacity 6 (which must be a minimum cut, as we saw, and its capacity is equal to the value of a maximum flow).

Remark 9.34. *This algorithm will stop after a finite number of steps if the capacities of the network are rational numbers:*

It is easy to see if they are integers, since the algorithm gives an increase of the value of the flow by at least 1 at each step. It must stop in a finite number of steps since the value of maximum flow is bounded by the capacity of any cut. To get the same result for capacitites with rational values, just multiply everything by a large enough integer to go back to the integer case, then divide the flow at the end by the same large integer.

Surprisingly enough, the algorithm may not terminate if you allow the capacities to take values in the real numbers.

Chapter 10

Some applications of the Ford-Fulkerson result

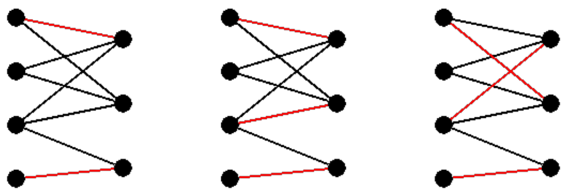
10.1 Matchings

Definition 10.1. *Let G be a graph.*

1. *We say that two edges of G are adjacent if they have a common vertex.*
2. *A matching in G is a set M of edges of G such that M does not contain adjacent edges.*
3. *A bipartite matching is a matching in a bipartite graph.*

Matchings are often used in bipartite graphs, in order to pair (match) objects of different type (see next section for an example).

Here are some matchings (in red) in the same bipartite graph:



It is often interesting to find matchings that are as large as possible. the first two items in the following definition present two possible interpretations of "as large as possible".

Definition 10.2. *Let G be a graph and let M be a matching in G.*

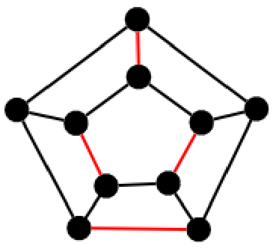
1. *We say that M is a maximal matching (in G) is there is no matching M' (in G) such that $M \subsetneq M'$.*
2. *We say that M is of maximum cardinality*[1] *if there is no matching M' (in G) such that $|M'| > |M|$.*

[1]such an M is commonly called a maximum matching, but this terminology conflicts with another common use of "maximum", and is too close to "maximal", so we will avoid it

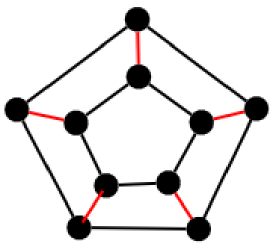
3. *A perfect matching M (in G) is a matching that contains every vertex of G, in the sence that for every vertex v in G, there is an edge e in M that is adjacent to v.*

Note that a matching of maximum cardinality is necessarily maximal, but that the converse does not hold (see example below). Also, a perfect matching is necessarily of maximum cardinality.

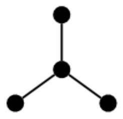
Example 10.3. *A maximal matching (that is not of maximum cardinality):*



A perfect matching:



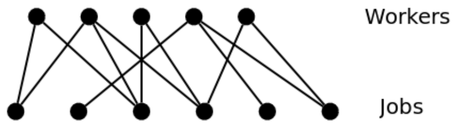
We can also observe that a graph with a perfect matching must have an even number of vertices, but having an even number of vertices is not sufficient to have a perfect matching. For instance the following graph does not have a perfect matching (obviously):



10.2 Allocation problems

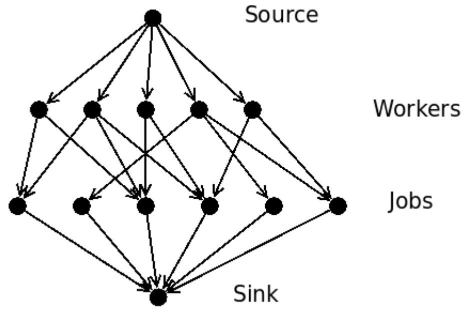
Consider the following problem: You have a group of workers, and a group of jobs to be done. Each worker can only do one job at a time, and is only qualified to do some of the jobs.

You want to give jobs to as many workers as possible. You build a bipartite graph G with bipartition (X, Y) as follows: X is the set of workers, Y is the set of jobs, and you put an edge between $x \in X$ and $y \in Y$ if worker x is qualified to do job y.



What you want is a matching in G of maximum cardinality. (You are "matching" as many workers as possible to a job, hence the terminology "matching".)

This can easily be solved using network flows: Turn all the edges into directed edges from the workers to the jobs. Add a source "before" the workers with arcs of capacity 1 between the source and each worker (it will ensure that no worker can be affected to more than 1 job). Then add a sink "after" the jobs, with arcs of capacity one between each job and the sink (it will ensure that each job is done by only 1 worker). Then look for a maximal flow.



(all arcs have capacity 1)

Remarkably, there a result very much resembling the max-flow / min-cut theorem: König's Theorem.

10.3 König's Theorem

Definition 10.4. *Let $G = (V, E)$ be a graph. A vertex cover of G is a set C of vertices such that every edge of G is adjacent to at least one vertex in C.*

Obviously V is a vertex cover of G, so finding large vertex covers is easy. We will be interested in vertex covers that are as small as possible.

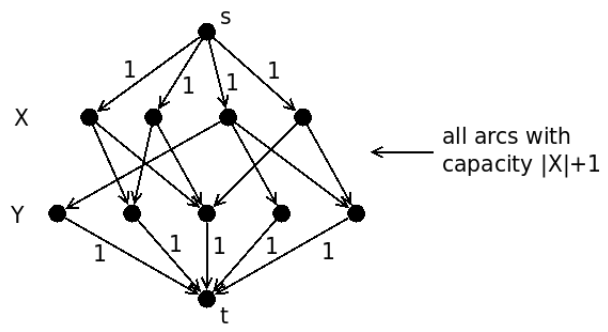
Definition 10.5. *Let G be a graph and let C be a vertex cover in G. We say that C is of minimum cardinality if there is not vertex cover C' in G such that $|C'| < |C|$.*

Definition 10.6. *Let G be a bipartite graph with bipartition (X, Y). The network $N(G)$ is built by taking the graph G, adding a source s and a sink t, and using the following arcs:*

(A1) Add an arc of capacity 1 from s to each vertex of X

(A2) Add an arc of capacity 1 from each vertex of Y to t.

(A3) Turn every edge xy in G with $x \in X$ and $y \in Y$ into an arc from x to y with capacity $|X| + 1$.



We will see in the next two lemmas that pairings in G correspond to flows in $N(G)$ and vertex covers in G correspond (essentially) to cuts in $N(G)$. These correspondences allow us to use the max-flow, min-cut theorem to get a result linking pairings of maximum cardinality and vertex covers of minimum cardinality.

Lemma 10.7. *Let G be a bipartite graph with bipartition (X, Y). Then each flow f in $N(G)$ gives a matching M in G with cardinality $v(f)$, and each matching in G gives a flow in $N(G)$ with value $|M|$.*

Therefore, a matching of maximum cardinality corresponds to a maximum flow f (and the cardinality of the matching is equal to the value of the flow).

Proof. You get the matching out of the flow by taking all the edges xy (with $x \in X$ and $y \in Y$) such that the arc xy has value 1. You get the flow out of the matching by giving value 1 to all the arcs xy that are in the matching. Check the details as an exercise. $\square$

Lemma 10.8. *Let G be a bipartite graph with bipartition (X, Y). Then each vertex cover C of G gives a cut of capacity $|C|$, and each cut (V_S, V_T) in $N(G)$ of capacity at most $|X|$ gives a vertex cover of cardinality $c(V_S, V_T)$,*

Proof. (1) Let C be a vertex cover of G, and define:

$$X_C = X \cap C \text{ and } Y_C = Y \cap C,$$

then

$$V_S = \{s\} \cup (X \setminus X_C) \cup Y_C \text{ and } V_T = \{t\} \cup X_C \cup (Y \setminus Y_C).$$

Clearly, (V_S, V_T) is a cut of $N(G)$: We have $s \in V_S$ and $t \in V_T$, and every vertex is in exactly one of V_S, V_T. We compute its capacity: Let uv be an arc with $u \in V_S$ and $v \in V_T$ (the type of arc that is used to compute the capacity). By construction of the arcs of $N(G)$ and by definition of V_S, V_T, we must have $u = s$ and $v \in X$ (so that $v \in X_C$, since $v \in V_T$), or $u \in Y$ (so that $u \in Y_C$, since $u \in V_S$) and $v = t$.

Altogether we have $|X_C| + |Y_C| = |C|$ such edges. They all have capacity 1, so the capacity of the cut is $|C|$.

(2) Let (V_T, V_S) be a cut in $N(G)$, with $c(V_S, V_T) \leq |X|$. Observe that:

1. By construction of $N(G)$ there is no arc xy with $x \in X$ and $y \in Y$ such that $x \in V_S$ and $y \in V_T$ (otherwise the capacity of the cut would be at least $|X| + 1$).

2. The arcs uv with $u \in V_S$ and $v \in V_T$ are therefore of the form $u = s$ and $v \in X \cap V_T$ (uv has capacity 1), or $u \in Y \cap V_S$ and $v = t$ (uv has capacity 1).

3. Therefore the capacity of (V_S, V_T) is $|X \cap V_T| + |Y \cap V_S|$.

Let $C = (X \cap V_T) \cup (Y \cap V_S)$. We first check that C is a vertex cover in G: Consider an edge xy in G with $x \in X$ and $y \in Y$, and assume that $x \notin C$ and $y \notin C$. Then $x \in V_S$ and $y \in V_T$, which is impossible as we saw in the first observation above. The cardinality of C is equal to the capacity of the cut (third observation above). □

Theorem 10.9 (König's Theorem)**.** *Let G be a bipartite graph with bipartition (X, Y). Then the number of edges in a matching of maximum cardinality is equal to the number of vertices in a vertex cover of minimum cardinality.*

Proof. We consider the network $N(G)$. Let M be a matching of maximum cardinality in G. It corresponds to a maximum flow f such that $v(f) = |M|$ by Lemma 10.7. Observe that $v(f) = f_-(s) \leq |X|$. Let (V_S, V_T) be the minimum cut corresponding to f (for instance by the Ford-Fulkerson algorithm). Then $c(V_S, V_T) = v(f) \leq |X|$, and by Lemma 10.8 there is an vertex cover C in G corresponding to (V_S, V_T) such that $|C| = c(V_S, V_T)$, and this vertex cover is of minimal cardinality.

Finally, $|C| = c(V_S, V_T) = v(f) = |M|$. □

Observe that the proof provides a way to build the vertex cover of minimum cardinality out of the matching of maximum cardinality (which we know how to get by using the Ford-Fulkerson algorithm on the network $N(G)$).

Remark 10.10. *A bipartite graph may not have a perfect matching. A full answer to this question is given by Hall's marriage theorem.*