"Sapienza" University of Rome
Faculty of Information Engineering,
Informatics and Statistics
Department of Computer Science

Graph Theory

Lecture notes integrated with the book TODO

Author
Alessio Bandiera

March 20, 2025

Contents

Information and Contacts

Personal notes and summaries collected as part of the *Graph Theory* course offered by the degree in Computer Science of the University of Rome "La Sapienza".

Further information and notes can be found at the following link:
https://github.com/aflaag-notes. Anyone can feel free to report inaccuracies, improvements or requests through the Issue system provided by GitHub itself or by contacting the author privately:

- Email: **alessio.bandiera02@gmail.com**
- LinkedIn: **Alessio Bandiera**

The notes are constantly being updated, so please check if the changes have already been made in the most recent version.

Suggested prerequisites:

- Progettazione degli Algoritmi

Licence:

These documents are distributed under the **GNU Free Documentation License**, a form of copyleft intended for use on a manual, textbook or other documents. Material licensed under the current version of the license can be used for any purpose, as long as the use meets certain conditions:

- All previous authors of the work must be **attributed**.
- All changes to the work must be **logged**.
- All derivative works must be **licensed under the same license**.
- The full text of the license, unmodified invariant sections as defined by the author if any, and any other added warranty disclaimers (such as a general disclaimer alerting readers that the document may not be accurate for example) and copyright notices from previous versions must be maintained.
- Technical measures such as DRM may not be used to control or obstruct distribution or editing of the document.

1

Basics of Graph Theory

In the 18th century, in the city of Königsberg (Prussia), a puzzle captured the imagination of the townspeople. Königsberg, nestled along the winding *Pregel River*, was divided into four land masses — two parts of the mainland and two islands, Kneiphof and Lomse. Connecting these regions were **seven bridges**, crisscrossing the river back and forth.

Over time, a curious question arose among the people of Königsberg: was it possible to take a *walk* through the city, crossing each of the seven bridges **exactly once**, without retracing any steps, and ending the walk in the same place where it started? This is known as the Seven Bridges of Königsberg problem.

It seemed simple enough, yet no one had managed to do it. The challenge became a favorite pastime, debated in marketplaces and whispered about in taverns. Some claimed it was possible with the right path, while others remained skeptical.

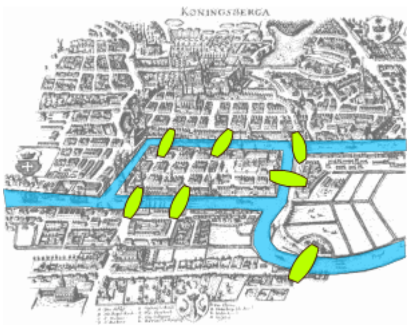


Figure 1.1: The map of Königsberg in Euler's time, showing the actual layout of the seven bridges. [2]

Word of this peculiar problem reached the brilliant Swiss mathematician, Leonhard Euler, a man whose mind was always drawn to patterns and logic. Intrigued, Euler set out to solve the riddle — not by drawing endless maps or walking the streets himself, but by **abstracting** the problem into something entirely new.

Euler realized that the specific layout of the city was *irrelevant*. What truly mattered was the way the landmasses were connected by the bridges. He represented each landmass as a **dot** and each bridge as a **line** between them. In doing so, he stripped away unnecessary details and created a simple, elegant combinatorial structure that we know refer to as **graph**.

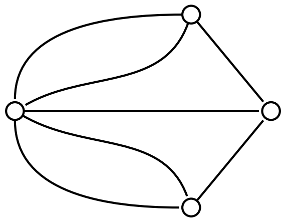


Figure 1.2: The graph drawn by Euler which models the *Seven Bridges of Königsberg* problem.

Through his analysis, Euler discovered a fundamental rule: for a walk to cross each bridge exactly once and return to the starting point, every landmass had to be connected by an **even** number of bridges. In Königsberg's case, however, each landmass had an odd number of bridges, making the task impossible.

Euler's proof, published in 1736, was groundbreaking — not just because he solved the Königsberg puzzle, but because he laid the foundation for an entirely new branch of mathematics: graph theory. His ideas would go on to shape the study of networks, from modern transportation systems to social media connections and even the vast web of the internet itself. And so, from a simple question about bridges in a small Prussian city, a whole new field of mathematics was born—one that continues to shape the world centuries later.

This chapter will discuss the basics of the field of **graph theory**, and will lay the foundation for later chapters.

1.1 Introduction

Definition 1.1: Graph

A **graph** is a pair $G = (V, E)$, where V is the — finite — set of **vertices** of the graph, and E is the set of **edges**.

For now, will assume to be working with **simple** and **undirected** graphs, i.e. graphs in which the set of edges is defined as follows

$$E \subseteq [V]^2 = \{\{x, y\} \mid x, y \in V \land x \neq v\}$$

where the notation $\{x, y\}$ will be used to indicate an edge between two nodes $x, y \in V$, and will be replaced with $xy = yx$ directly — the *set* notation for edges is used to highlight that edges have no direction.

We will indicate with n and m the cardinality of $|V|$ and $|E|$, respectively. Moreover, we will indicate with $V(G)$ and $E(G)$ the set of the vertices and edges of G respectively when there is ambiguity.

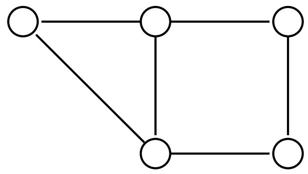


Figure 1.3: A simple graph.

Note that, in this definition, we are assuming that each edge has exactly 2 *distinct* endpoints — i.e. the graphs do not admit **loops** — and there cannot exist two edges with the same endpoints. In fact, if we drop these assumption we obtain what is called a **multigraph**.

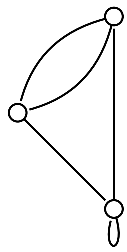


Figure 1.4: A multigraph.

Definition 1.2: Subgraph

Given a graph $G = (V, E)$, a **subgraph** $G' = (V', E')$ of G is a graph such that $V' \subseteq V$ and $E' \subseteq E$.

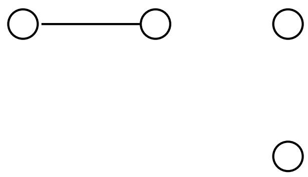


Figure 1.5: This is a subgraph of the graph shown in Figure 1.3.

Definition 1.3: Induced subgraph

Given a graph $G = (V, E)$, a subgraph $G' = (V', E')$ of G is **induced** if every edge of G with both ends in V is an edge of V'.

This definition is *stricter* than the previous one: in fact, the last graph is *not* an example of an induced subgraph, but the following is:

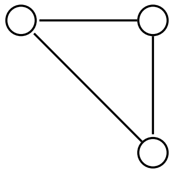
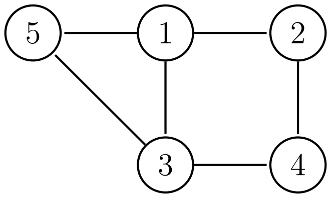


Figure 1.6: This is an *induced* subgraph of the graph shown in Figure 1.3.

Note that every induced subgraph of a graph is **unique** by definition, and we indicate each induced subgraph as follows: suppose that the graph in Figure 1.3 had the following *labeling* on the vertices



then, the induced subgraph in Figure 1.6 would have been referred to as $G[\{1, 3, 5\}]$.

Intuitively, two vertices $x, y \in V$ are said to be **adjacent**, if there is an edge $xy \in E$, and we write $x \sim y$. If there is no such edge, we write $x \nsim y$ for non-adjacency. The **neighborhood** of a vertex $x \in V$ is the set of vertices that are adjacent to x, and it will be indicated as follows

$$\mathcal{N}(x) := \{y \in V \mid x \sim y\}$$

Similarly, the neighborhood of a set of vertices will be defined as follows

$$\forall S \subseteq V \quad \mathcal{N}(S) := \bigcup_{v \in S} \mathcal{N}(v)$$

The **degree** of a vertex $x \in V$, denoted with $\deg(x)$, is exactly $|\mathcal{N}(x)|$. We will use the following notation for the **minimum** and **maximum** degree of a graph, respectively

$$\delta := \min_{x \in V} \deg(x) \qquad \Delta := \max_{x \in V} \deg(x)$$

Lemma 1.1: Handshaking lemma

Given a graph $G = (V, E)$, it holds that

$$\sum_{x \in V} \deg(x) = 2\,|E|$$

Proof. Trivially, the sum of the degrees counts every edge in E exactly twice, once for each of the 2 endpoints. □

Definition 1.4: k-regular graph

A graph G is said to be **k-regular** if every vertex of G has degree k.

Note that in a k-regular graph it holds that

$$\sum_{x \in V} \deg(x) = k \cdot n$$

Proposition 1.1

There are no k-regular graphs with k odd and an odd number of vertices.

Proof. By way of contradiction, suppose that there exists a k-regular graph $G = (V, E)$ such that both k and n are odd; however, by the handshaking lemma we would get that

$$2\,|E| = \sum_{x \in V} \deg(x) = k \cdot n$$

but the product of two odd numbers, namely k and n, is still an odd number, while $2\,|E|$ must be even $\lightning$. $\square$

1.2 Important structures

1.2.1 Paths, walks and cycles

Definition 1.5: Path

A **path** is a *graph* with vertex set $x_0, \ldots, x_n$ and edge set $e_1, \ldots, e_n$ such that $e_i = x_{i-1}x_i$.

The **length** of a path is the number of edges between x_0 and x_n, i.e. $|\{e_1, \ldots, e_n\}|$, namely n in this case. A path of length 1 is called *trivial* path.

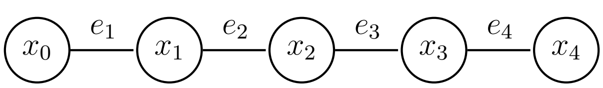


Figure 1.7: A path graph of length 4 that links x_0 and x_4.

Through *paths* we can provide the definition of **distance** between two nodes of a graph.

Definition 1.6: Distance

Given a graph $G = (V, E)$, and two vertices $x, y \in V$, the **distance** between x and y in G, denoted with $\text{dist}_G(x, y)$, is defined as the length of the *shortest* path between x and y in G.

If there is no ambiguity, we will simply write $\text{dist}(x, y)$ instead of $\text{dist}_G(x, y)$. Finally, given a path P and two vertices $u, v \in V(P)$, we will denote with $u\ P\ v$ the *subpath* of P between u and v. Now, consider the following definition.

Definition 1.7: Walk

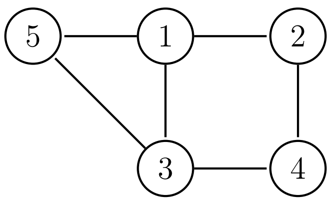
Given a graph $G = (V, E)$, a **walk** is a *sequence* of vertices and edges

$$x_0\ e_1\ x_1\ \dots\ x_{k-1}\ e_k\ x_k$$

where $x_0, \dots, x_k \in V$, $e_1, \dots, e_k \in E$ and $e_i = x_{i-1}x_i$.

The **length** of a walk is the number of edges between x_0 and x_k, i.e. $|\{e_1, \dots, e_k\}|$, namely k in this case. If $x_0 = x_k$ we say that the walk is **closed**.

If there is a path – or a walk — between two vertices $x, y \in V$, we say that the path — or the walk — **links** x and y, and we write this as $x \to y$. Any vertex that is different from x and y is called *internal node*. For instance, given the previous graph labeled as follows



an example of a walk over this graph is given by the following sequence

$$1\ \{1, 2\}\ 2\ \{2, 4\}\ 4\ \{4, 3\}\ 3\ \{3, 1\}\ 1\ \{1, 5\}\ 5$$

that *links* 1 and 5, i.e. the walk is of the form $1 \to 5$.

Note that there is a subtle difference between the definitions of **path** and **walk**: the definition of a path implies that this is always a *graph* on its own, while a walk is defined as a *sequence*. Nonetheless, we will treat *paths* as if they where *sequences* as well. This assumption holds for the following structures that will be discussed as well.

However, by definition of path, not every alternating sequence of vertices and edges is a valid path, in fact:

- in a *walk* it is possible to repeat both vertices and edges
- in a *path* there can be no repetition of vertices nor edges (note that *edge* repetition implies *vertex* repetition)

For instance, the previous example of *walk* is not a valid *path*, because the vertex 1 is repeated.

Theorem 1.1: Paths and walks

Given a graph $G = (V, E)$ and two vertices $x, y \in V$, in G there is a path $x \to y$ if and only if there is a walk $x \to y$.

Proof. By definition, every path is a walk, thus the direct implication is trivially true. To prove the converse implication, consider two vertices x and y for which there is at least one walk $x \to y$ in G. Now, out of all the possible walks $x \to y$ in G, consider the *shortest* one, i.e. the one with the least amount of edges, and let it be the following sequence

$$x \ e_1 \ x_1 \ \ldots \ x_{k-1} \ e_k \ y$$

By way of contradiction, assume that this walk is not a path. Therefore, there must be either one vertex or one edge repeated, but since edge repetition always implies vertex repetition, we just need to take this case into account. Assume that there are two indices $i, j \in [k-1]$ such that $i \neq j$ and $x_i = x_j$; however this implies that

$$x \ e_1 \ \ldots \ x_{i-1} \ e_i \ x_i \ e_{j+1} \ x_{j+1} \ \ldots \ x_{k-1} \ e_k \ y$$

is still a walk $x \to y$ of strictly shorter length, but we chose the original sequence to be the *shortest* possible walk $x \to y$ $\lightning$. □

Proposition 1.2

The longest path in any graph has a length of at least δ.

Proof. Consider a graph $G = (V, E)$, and let P be a longest path in G, labeled as follows

$$x_0 \ e_1 \ x_1 \ \ldots \ x_{k-1} \ e_k \ x_k$$

and assume that its length is k. Since P is a longest path in G, x_k cannot have neighbors outside P itself, otherwise P would not have been the longest path of G — it could have been extended by one of x_k's neighbors. This implies that

$$\mathcal{N}(x_k) \subseteq \{x_0, \ldots, x_{k-1}\}$$

and since $\delta \leq \deg(x_k) := |\mathcal{N}(x_k)|$ by definition of δ, this implies that

$$\delta \leq |\{x_0, \ldots, x_{k-1}\}| = k$$

□

Definition 1.8: Cycle

A **cycle** is a *graph* with vertex set $x_1, \ldots, x_n$ and edge set $x_1x_2, x_2x_3, \ldots, x_{n-1}x_n, x_nx_1$.

The **length** of a cycle is the number of edges between x_1 and x_n, namely n in this case. A cycle of length n is denoted as C_n.

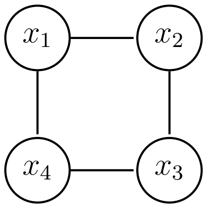


Figure 1.8: A cycle graph of length 4.

A graph that does not admit cycle subgraphs — or *cycles*, for short — is said to be **acyclic**.

Proposition 1.3

Every graph with $\delta \geq 2$ has a cycle of length at least $\delta + 1$.

Proof. Consider the proof of Proposition 1.2; by applying the same reasoning, we know that x_k cannot have neighbors outside P itself. However, since $\delta \geq 2$, and $x_k \sim x_{k-1}$, there must be at least one vertex in x_k's neighborhood that lies in P. Therefore, let x_i be the first vertex of P — w.r.t. our labeling of P — that is adjacent to x_k; hence, we have

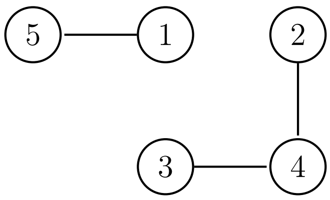
$$\mathcal{N}(x_k) \subseteq \{x_i, \ldots, x_{k-1}\} \implies \delta \leq |\{x_i, \ldots, x_{k-1}\}|$$

which implies that $x_i, \ldots, x_{k-1}, x_k, x_i$ is a cycle of length at least $\delta + 1$. □

Definition 1.9: Connected graph

An undirected graph $G = (V, E)$ is said to be **connected** if and only if for each vertex pair $x, y \in V$ there is a path $x \to y$.

All the graphs that we presented so far are *connected*, thus the following figure provides an example of an **disconnected** graph.



Definition 1.10: Component

Given a graph G, a **component** of G is a maximal connected subgraph of G.

For instance, the graph of the previous example is made up of 2 components, namely the following two subgraphs

$$C_1 = (\{1, 5\}, \{\{1, 5\}\})$$

$$C_2 = (\{2,3,4\}, \{\{2,4\}, \{4,3\}\})$$

Proposition 1.4

If G is a connected graph, and C is a cycle in G, then for any edge $e \in C$ it holds that $G - \{e\}$ is still connected.

Proof. Consider a graph $G = (V, E)$ that has a cycle C, and any two vertices $x, y \in V$; in particular, since G is connected, there must be a path $x \to y$ in G, and let this path be

$$P = x\ e_1\ x_1\ \ldots\ x_{k-1}\ e_k\ y$$

Consider an edge $e \in C$; if P does not traverse e, trivially $G - \{e\}$ will still contain P.

Now let

$$C = z_1\ f_2\ z_2\ \ldots\ z_{l-1}\ f_l\ z_l\ f_{l+1}\ z_1$$

and w.l.o.g. assume that $e = f_2 = z_1 z_2 = x_i x_{i+1} = e_{i+1}$ for some $i \in [k-1]$. Thus we can construct the following walk

$$x\ e_1\ x_1\ \ldots\ x_i\ f_{l+1}\ z_l\ \ldots\ f_3\ z_2\ e_{i+2}\ x_{i+2}\ \ldots\ x_{k-1}\ e_k\ y$$

from x to y, and by Theorem 1.1 we have that there is a path from $x \to y$, which proves that $G - \{e\}$ is still connected. □

1.2.2 Trees

Definition 1.11: Tree

A **tree** is a connected acyclic graph. Usually, but not necessarily, there is a fixed vertex called **root**, and any vertex that has degree 1 in the tree is called **leaf**.

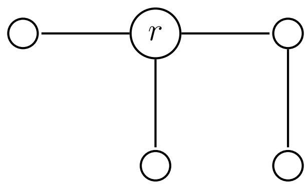


Figure 1.9: A tree with tree leaves, rooted in r.

A **forest** is an disconnected graph in which each component is a *tree*, as in the following example

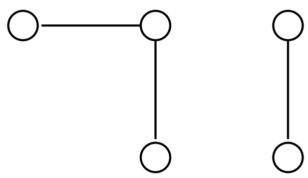


Figure 1.10: A forest.

Given a tree T rooted in some node $r \in V(T)$, and two vertices $x, y \in V(T)$, consider the paths P_x of the form $x \to r$ and P_y of the form $y \to r$, respectively. The first vertex of P_y that is encountered by tracing P_x from x to r is called **lowest common ancestor** (LCA) of x and y.

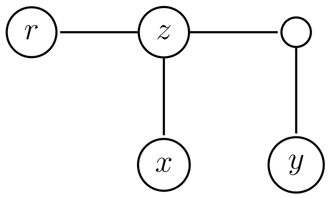


Figure 1.11: For instance, in this tree — rooted in r — the LCA of x and y is the vertex labeled with z.

Note that the LCA between any two vertices of a tree is *always defined*, since in the "worst case" it is the root r itself.

Theorem 1.2: Alternative definitions of tree

Given a graph $T = (V, E)$, the following statements are equivalent:

1. T is a tree
2. every vertex pair of T is connected by a unique path
3. T is *minimally connected*, i.e. T is connected and $\forall e \in E$ it holds that $T - \{e\}$ is disconnected
4. T is *maximally acyclic*, i.e. T is acyclic and $\forall x, y \in V$ such that $x \nsim y$, it holds that $T \cup \{xy\}$ has a cycle

Proof. We will prove the statements cyclically.

- $1 \implies 2$. By contrapositive, assume that in T there exist two vertices $x, y \in V$ for which there are two distinct paths P and Q of the form $x \to y$. If P and Q are edge-disjoint, then $P \cup Q$ is a cycle, which implies that T is not a tree by definition.

  Otherwise, assume that P and Q are not edge-disjoint. If we start say in x, and we follow Q edge by edge since P and Q are distinct, at some point we will encounter an edge $\{u, v\}$ such that $u \in P \cap Q$ and $v \in Q - P$ — possibly, $u = x$ itself. Moreover, since both paths lead to y, if we keep following Q we will encounter a vertex $z \in P \cap Q$ — possibly, $z = y$ itself — from which the two paths will coincide. Let Q' be the subpath of P starting with u and ending in z; then, $Q' \cup (Q - P)$ is a cycle in T, which implies that T is not a tree by definition.

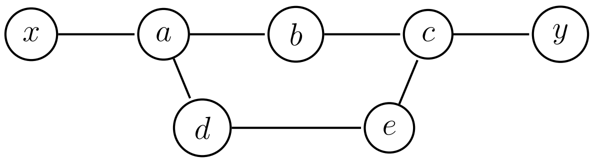


Figure 1.12: For instance, applying the argument of the proof in this graph we would get that $P = \{x, a, b, c, y\}$, $Q = \{x, a, d, e, c, y\}$, $Q - P = \{d, e\}$, $u = a$, $z = c$ and $Q' = \{a, b, c\}$, in fact $Q' \cup (Q - P) = \{a, b, c, e, d\}$ which is a cycle.

- $2 \implies 3$. Consider an edge $xy \in E$; this edge itself is a path $x \to y$, and if we assume statement 2 this implies that it is the *only* path from x to y. This implies that $T - \{xy\}$ cannot contain a path from x to y, therefore $T - \{xy\}$ is disconnected.
- $3 \implies 4$. Since statement 3 implies that T is connected, by Proposition 1.4 we have that T is acyclic. Now, pick $x, y \in V$ such that $x \nsim y$; by connectivity of T there must be a path $x \to y$ in T, and let this path be P. Lastly, since $x \nsim y$, we have that $P \cup \{xy\}$ is a cycle in T.
- $4 \implies 1$ By contrapositive, we want to prove that if T is not a tree, then T is not maximally acyclic. Note that if T is not a tree, we have two options:
  - if T is connected but contains a cycle, then T is clearly not maximally acyclic
  - if T is acyclic but disconnected, then by definition there must be two vertices $x, y \in V$ such that there is no path $x \to y$, which implies that $T \cup \{xy\}$ still does not contain any cycle

□

Lemma 1.2

Every tree with at least 2 vertices has a leaf.

Proof. By way of contradiction, assume T is a tree with at least 2 vertices that does not contain any leaves; then $\delta \geq 2$ in T, which implies that T contains a cycle of length at least $\delta + 1$ by Proposition 1.3. □

Lemma 1.3

Given a tree T, and a leaf v of T, it holds that $T - \{v\}$ is still a tree.

Proof. Since T is acyclic by definition, $T - \{v\}$ is still acyclic, we just need to prove that $T - \{v\}$ is still connected. By way of contradiction, assume that in $T - \{v\}$ there exist two vertices x and y such that there is no path between them. However, since T is connected, there is a path P of the form $x \to y$ in T.

Note that, if by removing v from T we disconnect x and y, it must be that v lies in P. Moreover, since v is in T but not in $T - \{v\}$, while both x and y are also in $T - \{v\}$, it must be that v is an *internal* node of P, i.e. $v \neq x, y$, which implies that $\deg(v) \geq 2$ by definition of path, contradicting the hypothesis for which v was a leaf $\lightning$. □

Proposition 1.5

If T is a tree, then $m = n - 1$.

Proof. We will prove the statement by induction on n

Base case. When $n = 1$, there are no edges in the tree, and $0 = m = 1 - 1$.

Inductive hypothesis. Assume that for a tree that has $n = k - 1$ nodes the statement holds.

Inductive step. We will prove the statement for a tree T that has $n = k$ nodes. Note that, since $n = 1$ is the base case, we can assume that $n = k \geq 2$, hence by Lemma 1.2 T contains at least one leaf. Let this leaf be v; then, by Lemma 1.3 it holds that $T - \{v\}$ is still a tree, and clearly $T - \{v\}$ has $k - 1$ nodes, which implies that we can apply the inductive hypothesis on $T - \{v\}$, i.e.

$$|E(T - \{v\})| = |V(T - \{v\})| - 1 = k - 1 - 1 = k - 2$$

However, note that v is a leaf, concluding that

$$m = |E(T)| = |E(T - \{v\})| + 1 = k - 2 + 1 = k - 1 = n - 1$$

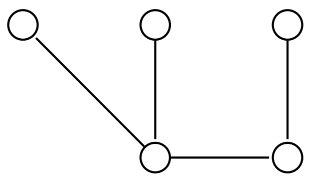
□

Definition 1.12: Spanning tree

Given a graph $G = (V, E)$, a **spanning tree** T of G is a subgraph of G such that

- T is a tree
- $V(T) = V(G)$, i.e. T *spans* every vertex of G

For instance, given the graph in Figure 1.3, a possible spanning tree is the following:



Lemma 1.4

Any connected graph has a spanning tree.

Proof. Consider a connected graph G, and keep removing edges from $E(G)$ — and their relative endpoints from $V(G)$ — one by one, as long as G is still connected. If no other edge can be removed from G without violating connectivity, we will end up with a graph that must be a tree by statement 3 of Theorem 1.2. □

Thanks to this last proposition, we can actually prove a stronger version of the Proposition 1.5, which is the following.

Theorem 1.3

T is a tree if and only if T is connected and $m = n - 1$.

Proof. The direct implication is proved in Proposition 1.5, so we just need to prove the converse implication. Consider a connected graph T such that $m = n - 1$; by Lemma 1.4 T must have a spanning tree T', and by Proposition 1.5 itself it holds that $|E(T')| = |V(T')| - 1$. However, T' is a spanning tree of T, therefore

$$V(T) = V(T') \implies |V(T)| = |V(T')| \implies |E(T')| = |V(T)| - 1 = |E(T)|$$

which implies $E(T) = E(T')$ because T' is a subgraph of T, therefore $T = T'$, concluding that T must be a tree since T' is a tree. □

1.2.3 Bipartite graphs

Definition 1.13: Bipartite graph

A graph $G = (V, E)$ is said to be **bipartite** if there exists a set $X \subseteq V$ such that every edge of G has exactly one endpoint in X and one in $V - X$. If such a set X exists, we say that $(X, V - x)$ is a **bipartition** of G.

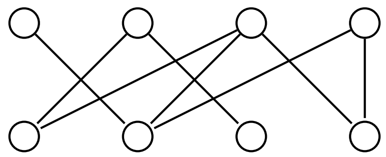


Figure 1.13: An example of a *bipartite graph*. In particular, if we call the uppermost set of nodes A and the lowermost one B, then (A, B) is a bipartition of the graph.

There are various types of graphs that can be bipartitioned. For example, any **tree** T can be bipartitioned through a bipartition $(X, V(T) - X)$ by considering the following set of vertices

$$X := \{v \in V(T) \mid \mathrm{dist}(r, v) \text{ even}\}$$

where r is T's root.

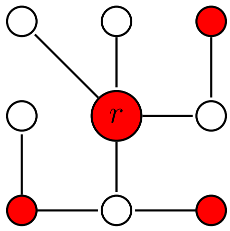


Figure 1.14: The set of red vertices X defines a bipartition $(X, V(T) - X)$ of this tree T.

However, *not every type* of graph can be bipartitioned. For instance, consider the following type.

Definition 1.14: Clique

A **clique** is a *graph* in which each vertex is adjacent to any other vertex of the graph. A clique that has n vertices is denoted as K_n.

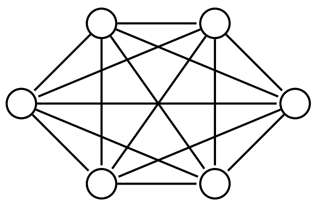


Figure 1.15: The clique K_6.

It is easy to see that no clique K_n can be bipartitioned, since there is an edge between any pair of vertices of the graph. However, this is not the only type of graph that cannot be bipartitioned.

Lemma 1.5

If G is a bipartite graph, and H is a subgraph of G, then H must be bipartite.

Proof. Given a bipartite graph $G = (V, E)$, assume that $(X, V - X)$ is a bipartition of G, and let H be a subgraph of G; then, it is easy to see that $(X \cap V(H), V(H) - X)$ is a bipartition for H. $\square$

Note that this lemma implies that G is bipartite if and only if every connected component of G is bipartite: in fact, the direct implication follows from this lemma, and the following figure provides an intuition for the converse implication.

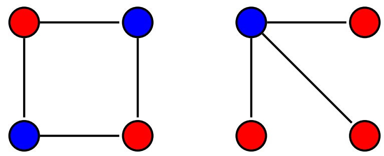


Figure 1.16: Consider the following disconnected graph G made of these two connected components, C_1 and C_2 respectively. Say that C_1 has a bipartition $(X_1, V(C_1) - X_1)$ where X_1 is the red set, and C_2 has a bipartition $(X_2, V(C_2) - X_2)$ where X_2 is the red set; thus, $(X_1 \cup X_2, V(C_1 \cup C_2) - X_1 - X_2)$ is clearly a bipartition of G. This process may be repeated for all the connected components of any disconnected graph.

Theorem 1.4: Bipartite graphs

G is bipartite if and only if G has no odd-length cycle.

Proof.

Direct implication. We will prove the contrapositive, i.e. if G has an odd-length cycle, then G cannot be bipartitioned. Consider a graph G with an odd-length cycle C_{2k+1} of vertices $x_1, \ldots, x_{2k+1}$; by way of contradiction, assume that G is bipartite through a bipartition $(X, V(G)) - X)$ for some $X \subseteq V(G)$. W.l.o.g. assume that $x_1 \in X$; then, since X defines a bipartition of G it must be that $x_2 \notin X$, and $x_3 \in X$ and so on and so forth. In particular, for any odd value of i we will have that $x_i \in X$, but this implies that both x_1 and x_{2k+1} must be inside X, which means that the edge $x_1 x_{2k+1}$ violates the bipartition induced by X $\lightning$.

Converse implication. Again, we will prove the contrapositive, i.e. if G is not bipartite it must contain an odd-length cycle. By the previous observation, G is not bipartite if and only if at least one connected component of G is not bipartite, and let this component be $\overline{G}$. Note that, since $\overline{G}$ is connected, it must contain a spannning tree T by Lemma 1.4. Moreover, as previously described, we can always define a bipartition on a tree, namely $(X, V(T) - X)$ where

$$X := \{v \in V(T) \mid \mathrm{dist}_T(r, v) \text{ even}\}$$

for some root node $r \in V(T)$.

Now, since T is a spanning tree of $\overline{G}$, which is now bipartite by hypothesis, there must be an edge $xy \in V(\overline{G})$ such that either $x, y \in X$ or $x, y \in V(T) - X$, i.e. the edge xy must have both endpoints in the same set of T's bipartition. Let z be the LCA between x and y in T, and P_x and P_y be the paths of the form $x \to r$ and $y \to r$, respectively. Note that, since xy has both endpoints in the same set, it must be that the lengths of P_x and P_y have the same parity by definition of X. Lastly, by statement 2 of Theorem 1.2 we have that $r\ P_x\ z = r\ P_y\ z$, which implies that the lengths of $z\ P_x\ x$ and $z\ P_y\ y$ must have the same parity. This concludes that

$$z\ P_x\ x \cup z\ P_y\ y \cup xy$$

is an odd-length cycle of G.

$\square$

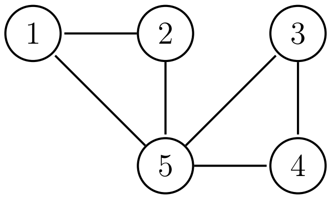
1.2.4 Eulerian tours

At the start of this chapter, we introduced the *Seven Bridges of Königsberg* problem, which led to the emergence of graph theory as a branch of combinatorics. Over time, as the field developed, this problem was formalized into the following definition.

Definition 1.15: Eulerian tour

An **Eulerian tour** over a graph G is a closed walk that traverses every edge of G exactly once.

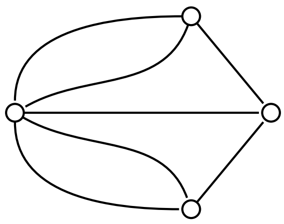
For instance, consider the following graph



After some trial an error, it is easy to find an Eulerian tour over this graph, for instance

$$1\ \{1,2\}\ 2\ \{2,5\}\ 5\ \{5,3\}\ 3\ \{3,4\}\ 4\ \{4,5\}\ 5\ \{5,1\}\ 1$$

Note that this is a valid Eulerian tour because there is no edge repetition, and vertex repetition is allowed by definition. On the counter side, the graph — or, more precisely, the *multigraph* — that the bridges of Königsberg define, which is the following



does not admit any Eulerian tour. But, given a graph, how can we determine with certainty whether it contains an Eulerian tour? The following theorem, proved by Euler himself in his original paper [3], answers this question. Note that this theorem holds both for graphs and multigraph.

Theorem 1.5: Eulerian tours

A graph G admits an Eulerian tour if and only G is connected and every vertex of G has even degree.

Proof.

Direct implication. Consider the contrapositive of the direct implication, and assume that G contains at least one odd-degree vertex v. Recall that an Eulerian tour is a closed walk that does not allow edge repetition, which implies that the *starting* point of the walk is actually not relevant. Therefore, we can assume w.l.o.g. that any possible Eulerian tour defined on G starts on v itself, but then to be *closed* it must end on v as well. Moreover, each time any Eulerian tour passes through v it must use 2 distinct edges, however v is an odd-degree vertex, therefore at the end of the Eulerian tour there is no way we can come back to v and close the walk.

Converse implication. Consider a graph G in which every vertex has even degree, and let W be the *longest* walk of G that does not repeat edges, and let it be labeled as follows

$$x_0 \; e_1 \; x_1 \; \dots \; x_{k-1} \; e_k \; x_k$$

Claim: $x_0 = x_k$, i.e. W is closed.

Proof of the Claim. By way of contradiction, assume that W $x_0 \neq x_k$. If this is the case, then $x_k = v$ for some vertex v that is not x_0. Note that W is a *walk*, therefore v may be repeated multiple times inside W, i.e.

$$x_0 \; e_1 \; x_1 \; \dots \; v \; \dots \; v \; \dots \; x_{k-1} \; e_k \; v$$

If l is the number of times v occurs in W *without counting* x_k, then clearly in W there are $2l + 1$ edges incident to v, namely e_k and 2 edges each other time v appears in W. However, since $2l + 1$ is odd and we assumed that G has no odd-degree vertices, there must be at least one edge $vu \in E(G)$ such that $u \notin V(W)$. This implies that

$$x_0 \; e_1 \; x_1 \; \dots \; x_{k-1} \; e_k \; v \; vu \; u$$

is a longer walk than W, and it does not repeat any edges because $uv \notin E(W)$, contradicting the definition of W $\lightning$. $\square$

This claim proves that W is closed, but we still need to prove that it traverses every edge to claim that it is indeed an Eulerian tour. By way of contradiction, assume that there exists at least one edge $e \notin E(W)$ not used by W. Consider an vertex x_i of W; by connectivity of G, there must be a path P beween x_i and each of the endpoints of e. Let $x_i u$ be the first edge of P, for some $u \notin E(W)$. This implies that

$$u \; ux_i \; x_i \; e_{i+1} \; x_{i+1} \; \dots \; x_{k-1} \; e_k \; x_k \; e_1 \; x_1 \; \dots \; x_{i-1} \; e_i \; x_i$$

is a longer walk than W that does not repeat any edge, since $ux_i \notin E(W)$, again contradicting the definition of W $\lightning$.

$\square$

Note that this theorem proves that it is not possible to describe an Eulerian tour over the bridges and landmasses of Königsberg, because every vertex of the multigraph has odd degree.

1.2.5 Hamiltonian cycles

Eulerian tours are closed walks that traverse every edge of the graph exactly once. But what if we are interested in traversing each *vertex* exactly once instead?

Definition 1.16: Hamiltonian paths and cycles

A **Hamiltonian path** over a graph G is a subgraph P_n of G. A **Hamiltonian cycle** over a graph G is a subgraph C_n of G.

Hamiltonian paths and cycles are named after W. R. Hamilton. Note that the notation P_n (or C_n) implies that the length of the path (or cycle) is n, hence this definition matches our requirements.

As for the case of Eulerian tours, when some conditions are met, Hamiltonian cycles are guaranteed to exist, as discussed in the following theorem.

Theorem 1.6: Hamiltonian cycles

A graph G such that $\delta \geq \frac{n}{2}$ contains a Hamiltonian cycle.

Proof. First, we will prove that the condition of the statement implies that G is connected.

Claim: G is connected.

Proof of the Claim. By way of contradiction, suppose that G is not connected; therefore G has at least two connected components. Let H be the smallest connected component of G; then, clearly $|V(H)| \leq \frac{n}{2}$. However, note that $|\mathcal{N}(x)| \geq \delta \geq \frac{n}{2}$ and since $\{x\} \cup \mathcal{N}(x) \subseteq V(H)$, we get that $V(H)$ must have at least $\frac{n}{2} + 1$ nodes $\lightning$. □

Let P be the longest path of G, and let $x_0, \ldots, x_k$ be its vertices.

Claim: There exists an index ℓ such that $x_0 \ e_1 \ x_1 \ \ldots \ x_k \ e_\ell \ x_\ell \ e_{\ell+1} \ x_{\ell+1} \ e_0 \ x_0$ is a cycle.

Proof of the Claim. By the same argument used in the proof of Proposition 1.2, we know that

$$\mathcal{N}(x_0), \mathcal{N}(x_k) \subseteq \{x_0, \ldots, x_k\}$$

Let I_0 and I_k be the following two sets

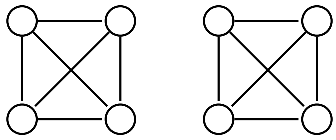
$$I_0 := \{i \mid i \in [1, k], x_i \in \mathcal{N}(x_0)\} \implies |I_0| = |\mathcal{N}(x_0)|$$

$$I_k := \{i \mid i \in [1, k], x_{i-1} \in \mathcal{N}(x_k)\} \implies |I_k| = |\mathcal{N}(x_k)|$$

Since $\delta \geq \frac{n}{2}$, we have that $|I_0|, |I_k| \geq \frac{n}{2}$. However, note that $k \leq n - 1$ — since we started counting at 0 — hence by the pigeonhole principle there must be at least one index $\ell \in I_0 \cap I_k$, meaning that $x_0 \sim x_\ell$ and $x_k \sim x_{\ell-1}$, defining a cycle as described in the statement of the claim. □

This means that we found a cycle C in the graph that uses all the vertices of P, namely $x_0, \ldots, x_k$. By way of contradiction, assume that $k < n - 1$, i.e. C has less than n nodes, meaning that C is a non-Hamiltonian cycle. In particular, if $k < n - 1$, we have that $|V(G)| - |V(C)| \neq \varnothing$, thus let $y \in V(G) - V(C)$. By the previous claim, we know that G is connected, there must be an edge $xy \in E(G)$ such that $x \in V(C)$. However, this would imply that $P \cup \{xy\}$ is a path of longer path than P $\lightning$. $\square$

Note that the statement of this theorem cannot be improved, even by 1; for instance consider the following graph



composed of two disconnected K_4. Here, we have that

$$\delta = 4 - 1 = 3 \geq \frac{2 \cdot 4}{2} - 1 = 4 - 1 = 3 = \frac{n}{2} - 1$$

However, $\delta \geq \frac{n}{2} - 1$ is not sufficient to guarantee connectivity.

1.3 Exercises

Problem 1.1

Let $G = (V, E)$ be a graph of n vertices, where $n \geq 2$. Show that there must be two vertices $x, y \in V$ such that $\deg(x) = \deg(y)$.

Solution. By definition, the range of the possible degrees for any node of G is $[0, n - 1]$. By way of contradiction, assume that for any two vertices $x, y \in V$ it holds that $\deg(x) \neq \deg(y)$; hence, since the graph has n nodes, it must be that each node is assigned a different degree, and that we use all the possible degrees in $[0, n - 1]$. In particular, this implies that there are two vertices $u, v \in V$ such that $\deg(u) = 0$ and $\deg(v) = n - 1$, but this is a contradiction because if the degree of v is $n - 1$, it must be adjacent to all the other nodes of V, including u, and $\deg(u) = 0$ $\lightning$.

2

Matchings

Definition 2.1: Matching

Given a graph $G = (V, E)$, a **matching** of G is a set of edges $M \subseteq E$ such that

$$\forall e, e' \in M \quad e \cap e' = \varnothing$$

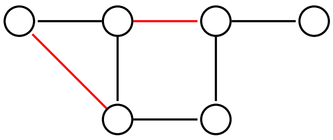


Figure 2.1: A *matching* of the previous graph.

As shown in figure, a matching is nothing more than a set of edges that must not share endpoints with each other — for this reason, in literature it is often referred to as **independent edge set**.

Given a matching M in a graph G, we say that a vertex $v \in V(G)$ is **free** w.r.t. M if there are no edges $e \in M$ such that $e \cap v \neq \varnothing$. An edge $xy \in E(G)$ is said to be *disjoint* from M if both x and y are free w.r.t. M.

In graph theory we are often interested in the matching that has the largest possible cardinality of a graph. For this purpose, we often distinguish the two following concepts, namely *maximal* and *maximum* matching.

Definition 2.2: Maximal matching

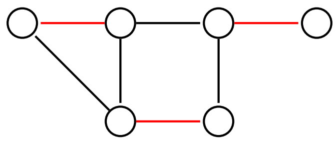
A **maximal matching** is a matching that cannot be extended any further.

For instance, the matching shown in Figure 2.1 is actually a **maximal matching**, because no other edge in E can be added to che current set of edges M of the matching without breaking the matching condition.

Definition 2.3: Maximum matching

A **maximum matching** is a matching that has the largest cardinality.

Clearly, the previous example does not repreent a **maximum matching**, because the following set of edges



is still a valid matching for the graph, but has a larger cardinality than the previous set.

Given a matching M in a graph G, what conditions must be met to increase its cardinality? Trivially, if there exists an edge e disjoint from M, clearly $M \cup \{e\}$ is a larger matching than M — implying that M was not *maximal* in G. However, this is not the only situation in which the cardinality of M can be extended. In fact, consider the following definitions.

2.1 Augmenting paths

Definition 2.4: Alternating path

Given a graph G, and a matching M in G, an M**-alternating path** is a path of G that starts at a free node w.r.t. M, and is composed of edges that alternate between M and $E(G) - M$.

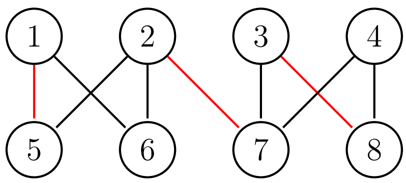


Figure 2.2: For instance, if M is the set of *red* edges — which forms a matching of the graph — then 6 $\{6, 2\}$ 2 $\{2, 7\}$ 7 $\{7, 3\}$ 3 $\{3, 8\}$ 8 is an M-alternating path.

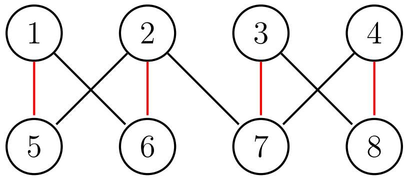
Definition 2.5: Augmenting path

Given a graph G, and a matching M in G, an M**-augmenting path** is an M-alternating path that ends at a free vertex w.r.t. M.

For example, if we consider the path

$$6 \ \{6, 2\} \ 2 \ \{2, 7\} \ 7 \ \{7, 3\} \ 3 \ \{3, 8\} \ 8 \ \{8, 4\} \ 4$$

this is actually an M-augmenting path of the previous graph. Augmenting paths are very useful because they can be used to *expand* the cardinality of an initial matching. In fact, in the previous graph we can actually define a *larger* matching by **swapping** the edges of this augmenting path, as shown below



2.1.1 Berge's theorem

This suggests that the presence of augmenting paths in a graph is a *sufficient* condition for a matching *not* to be maximum, but we can actually prove that it is also *necessary*, as stated in the following theorem, proved by Berge [1] in 1957.

Theorem 2.1: Berge's theorem

Given a graph G, M is a maximum matching of G if and only if there are no M-augmenting paths.

Proof.

Direct implication. By contrapositive, consider a graph G and a matching M such that there is an M-augmenting path P in G. Moreover, by way of contradiction assume that M is maximum; however $M \Delta E(P)$ is a larger matching than M — here, Δ is the symmetric difference, therefore the operation $M \Delta E(P)$ has the same effect of *swapping* the edges of P between the ones in M and in $E(P) - M$.

Converse implication. By contrapositive, consider a graph G and a matching M of G that is not maximum, i.e. there exists a matching M^* of G such that $|M| < |M^*|$. Consider the subgraph of G that has the vertices of $V(G)$ and the edges described by $M \Delta M^*$; the symmetric difference of these two sets will yield the set of edges that are either in M or in M^*, but not in $M \cap M^*$, therefore this subgraph is *not* a multigraph. Moreover, since M and M^* are both matchings, we have that

(1) the degrees of the vertices of this subgraph can be either 0, 1 or 2

(2) in each component of the subgraph the edges *must* alternate between M and M^*

By the observation (1), we have that each component of the subgraph can be either

- a isolated vertex

- a cycle
- a path

and by observation (2), we have that all the cycle components must have *even* length, which implies that they have the same number of edges of M and M^*. On the other hand, path components may have either even or odd length; in particular, even-length paths must have the same number of edges of M and M^* — as for cycle components — while odd-length paths have a different number of edges of M and M^*. However, since $|M| < |M^*|$, there must be at least one path component of this subgraph such that its edges of M are less than the edges of M^*, and this is clearly an M-augmenting path.

□

Given a graph G, and a matching M of G, what is the maximum possible value for $|M|$? To answer this question, we need to introduce the following combinatorial structure.

2.1.2 König's theorem

Definition 2.6: Vertex cover

Given a graph G, a **vertex cover** for G is a set of vertices $C \subseteq V(G)$ such that every edge in G is incident to at least one vertex in C. Using symbols

$$\forall (u, v) \in E(G) \quad u \in C \vee v \in C$$

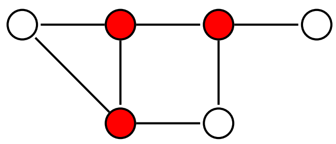


Figure 2.3: An example of a vertex cover.

As shown in figure, a vertex cover is simply a set of vertices that must *cover* all the edges of the graph. For vertex covers, we are interested in the *minimum* possible cardinality — the concepts of *minimal* and *minimum* are defined analogously.

As introduced before, through vertex covers we can bound the size of any matching of a graph.

Theorem 2.2: Matchings bound vertex covers

Given a graph G, a matching M, and a vertex cover S of G, it holds that $|M| \leq |S|$.

Proof. By definition, any vertex cover S of $G = (V, E)$ is also a vertex cover for $G^B = (V, B)$, for any set of edges $B \subseteq E$, and in particular this is true for $G^M = (V, M)$.

Now consider G^M, and a vertex cover C on it: by construction we have that $\Delta \leq 1$, therefore any vertex in C will cover at most 1 edge of M. This implies that if $|C| = k$, then C will cover at most k edges of G^M.

Lastly, since G^M has $|M|$ edges by definition, any vertex cover defined on G^M has to contain at least $|M|$ vertices. This implies that no vertex cover S of G smaller than $|M|$ can exist, because S will have to cover at least the edges in M. □

Corollary 2.1

Given a graph G, a maximum matching M^* and a minimum vertex cover S^*, it holds that $|M^*| \leq |S^*|$.

Moreover, if the graph is bipartite this theorem is actually *stronger*, as proved by Konig [6] in 1931.

Theorem 2.3: König's theorem

Given a bipartite graph G, a maximum matching M^* and a minimum vertex cover S^*, it holds that $|M^*| = |S^*|$.

Proof. Consider a graph G, a maximum matching M^* and a minimum vertex cover S^* of G; by the previous corollary, it follows that to prove the statement it suffices to show that there exists a vertex cover S such that $|S| = |M^*|$, because

$$|M^*| \leq |S^*| \leq |S| = |M^*| \implies |M^*| = |S^*|$$

Hence, we are going to construct the following vertex cover. Let G be bipartitioned through (A, B); then, for each edge $ab \in M^*$ such that $a \in A$ and $b \in B$, we place $b \in S$ if and only if there exists an M^*-alternating path that starts in A and ends at b, otherwise we place $a \in S$.

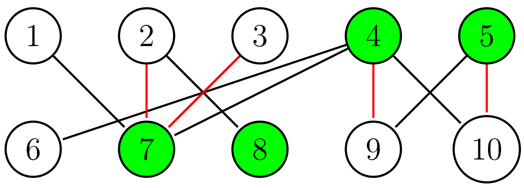


Figure 2.4: For instance, given this graph bipartitioned into (A, B) — where A is the uppermost row of vertices — and the *red* matching, we would construct the *green* vertex cover.

Note that, by definition, it holds that $|S| = |M^*|$.

Claim: S is a vertex cover for G.

Proof of the Claim. Consider an edge $ab \in E(G)$ such that $a \in A$ and $b \in B$; note that, by definition of S all the edges in M^* are already covered, hence we may assume that $ab \notin M^*$. We have two cases.

- a is free, i.e. $\nexists ab' \in M^*$ for $b' \in B$. Note that b cannot be free, otherwise $M^* \cup \{ab\}$ would still be a matching of G but greater than M^*. Hence, there must be an edge $a'b \in M^*$ for some $a' \in A$. Therefore, since a is free, the edge ab is a trivial M^*-alternating path ending at $b \in B$, meaning that $b \in S$ by definition, implying that ab is covered by S.
- a is matched, i.e. $\exists ab' \in M^*$ for $b' \in B$. Therefore, by definition of S, either a or b' lies in S. In particular, if $a \in S$, then ab is trivially covered by S, hence suppose that $b' \in S$.

  Observe that, by definition of S this implies that there must be an M^*-alternating path P that starts in A and ends at b'.

  - If $ab, ab' \notin E(P)$, then $P \cup \{b'a\} \cup ab$ is an M^*-augmenting path, which would contradict the fact that M^* is maximum by Theorem 2.1 $\lightning$
  - If $ab \notin E(P)$ but $ab' \in E(P)$, then P could not have been an M^*-alternating path starting at a vertex in A $\lightning$.
  - If $ab \in E(P)$ but $ab' \notin E(P)$, then P must have had the following form

    $$\ldots\ b''\ \{b'', a\}\ a\ \{a, b\}\ b\ \{b, a'\}\ a'\ \{a', b'\}\ b'$$

    where $a' \in A$, $b'' \in B$. However, since $ab' \in M^*$ and $ab \notin M^*$, and the edges of P must alternate w.r.t. M^*, ab'' must lie inside M^*, contradicting $ab' \in M^*$ by definition of matching $\lightning$.

  This implies that $ab', ab \in E(P)$, meaning that P encounters b "before" b'. However, this implies that $P - \{ab'\}$ is an M^*-alternating path that starts in A and ends at b, thus $b \in S$ by definition, concluding that ab is still covered by S.

$\square$

Hence, we have constructed a vertex cover S such that $|S| = |M^*|$, meaning that the statement holds because of the previous observation. $\square$

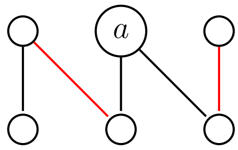
2.1.3 Finding maximum matchings

Consider a matching M of a graph G, and an M-augmenting path P; the idea of *swapping* the edges of P between M and $E(G) - M$ is very useful when G is **bipartite**. In fact, we can actually describe a procedure which is able to return a *maximum matching* of a bipartite graph, by swapping the edges of the augmenting paths present in G. However, for this algorithm to work, we first need a procedure capable of finding augmenting paths in bipartite graphs, which is defined down below:

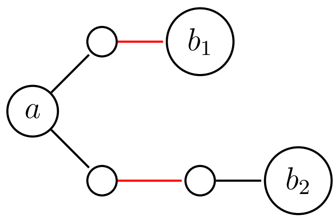
1. Assume that the considered graph G is bipartite through (A, B), and consider a matching M of G
2. Starting from a node $a \in A$ free w.r.t. M, compute a *modified* BFS such that the edges of its tree alternate between $E(G) - M$ and M

3. If the tree of the BFS contains a free leaf $b \in B$, then the path $v \to b$ is M-augmenting

For instance, given the following bipartite graph G, and a matching M of G — outlined in *red*



the *modified* BFS rooted in a would produce the following tree



and we observe that the path $a \to b_1$ is M-alternating, while the path $a \to b_2$ is M-augmenting. The next proposition guarantees that if there are M-augmenting paths that start in a, our *modified* BFS will find at least one of them.

Proposition 2.1

Given a bipartite graph G, bipartitioned into (A, B), and a matching M of G, if there exists an M-augmenting path in G that starts in a vertex $a \in A$ free w.r.t. M, then there exists an M-augmenting path in the tree T of the *modified* BFS.

Proof. Let P be an M-augmenting path that starts in a and minimizes the edges in $E(P) - E(T)$ and, by way of contradiction, assume that $E(P) - E(T) \neq \varnothing$, i.e. P is not completely contained in T. Therefore, let xy be the first edge in $E(P) - E(T)$ encountered while traversing P, starting at a, and w.l.o.g. assume that $x \in V(P)$.

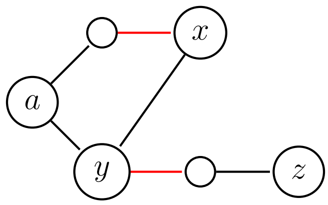
Claim: $x \in A$.

Proof of the Claim. By way of contradiction, assume that $x \in B$.

- Assume that x is not a leaf of T. Since the BFS starts at $a \in A$, and the edges of T alternate between M and $E(G) - M$, if $x \in B$ then the next edge xy' in T is an edge in M. Moreover — by the same reasoning — since P is M-augmenting, it must be that $xy \in M$, meaning that $xy, xy' \in M$ contradicting the definition of matching $\lightning$
- Now, assume that x is a leaf of T. By the same reasoning, xy must be in M because P is M-augmenting, but $xy \notin E(T)$ would imply that the BFS stopped before adding the edge xy to T $\lightning$.

□

For instance, given the following setting



a possible path for P would be $a \to x \to y \to z$. However, the path $a \to y \to z$ is still an M-augmenting path that starts at a but has one fewer edge not in T w.r.t. P, contradicting the definition of P $\lightning$.

Note that, in the general case we would consider the path $P' := a\ T\ y \cup y\ P$, however this is not guaranteed to be a path. The complete proof leverages the fact that G is bipartite in order to prove that P' is indeed a path, but it is very technical and outside the scope of these notes. □

Finally, now that we have a procedure which is guaranteed to find an augmenting path in a given bipartite graph, to return a maximum matching it suffices to run the following algorithm.

Algorithm 2.1: Maximum matching (bipartite graphs)

Given a bipartite graph G, the algorithm returns a maximum matching for G.

```
function MaximumMatchingBipGraphs(G)
    M := ∅
    do
        P := findAugmentingPath(G)          ▷ the previous procedure
        Swap the edges between M and E(G) − M in P
    while P ≠ None
    return M
end function
```

In particular, the output of this algorithm is guaranteed to be a *maximum* matching thanks to Theorem 2.1, since the algorithm terminates when there are no more augmenting paths left in the graph G.

2.2 Perfect matching

By definition, a matching is *not* forced to cover all the vertices of a graph. However, if this happens the matching is called **perfect matching**.

Definition 2.7: Perfect matching

Given a graph G, a **perfect matching** of G is a matching that covers all the vertices of G, i.e. M is a perfect matching if and only if

$$\forall v \in V(G) \quad \exists e \in M \quad v \cap e \neq \varnothing$$

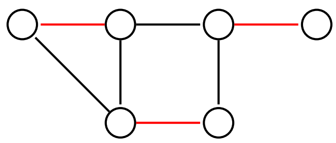


Figure 2.5: An example of a perfect matching.

Perfect matchings are an interesting topic of study when related to *bipartite graphs*. We observe that if a bipartite graph G, bipartitioned into (A, B), admits a perfect matching M, it must be that $|A| = |B|$. This is because every matched edge must connect one vertex from A to one vertex from B, and by definition M matched *each* vertex exactly once. However, the converse is not true.

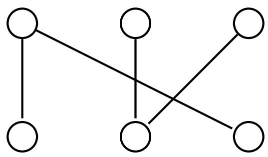


Figure 2.6: For instance, this bipartite graph, bipartitioned into (A, B) such that A is the uppermost row of vertices, even if $|A| = |B|$ this graph does not admit a perfect matching.

2.2.1 Hall's theorem

Because of this characterization on bipartite graphs, in addition to the concept of *perfect matching*, if $|A| \neq |B|$ we can define a *weaker* version of "perfect".

Definition 2.8: A-perfect matching

Given a bipartite graph G, bipartitioned through (A, B) such that $|A| \leq |B|$, we say that a matching M is an **A-perfect matching** if it covers all the vertices in A.

Note that we can always assume that $|A| \leq |B|$, without loss of generality. The following theorem, known as the Hall's marriage theorem — proved by Hall [5] in 1935 — shows the conditions that guarantee that an A-perfect matching exists.

Theorem 2.4: Hall's marriage theorem

Given a bipartite graph G, bipartitioned into (A, B), then G admits an A-perfect matching if and only if

$$\forall S \subseteq A \quad |S| \leq |\mathcal{N}(S)|$$

Proof. The direct implication is trivially true by definition of matching. We will prove the converse implication by contrapositive. Therefore, suppose that the bipartite graph G does not admit any A-perfect matching, and consider a minimum vertex cover V^* of G. Then, by Theorem 2.3, since G is bipartite we know that for any maximum matching M^* of G it holds that $|M^*| = |V^*|$. However, since we are assuming that there are no A-perfect matchings of G, any maximum matching must have size strictly less than $|A|$, meaning that $|V^*| = |M^*| < |A|$.

Now, consider the set $S := A - V^*$; since G is bipartite, it must hold that

$$\mathcal{N}(S) \subseteq V^* \cap B \implies |\mathcal{N}(S)| \leq |V^* \cap B| = |V^*| - |A \cap V^*|$$

Moreover, observe that

$$V^* \cap A = A - (A - V^*) = A - S \implies |V^* \cap A| = |A| - |S|$$

therefore, we have that

$$|\mathcal{N}(S)| \leq |V^*| - |V^* \cap A| = |V^*| - |A| + |S|$$

Lastly, since $|V^*| < |A| \iff |V^*| - |A| < 0$, we conclude that

$$|\mathcal{N}(S)| \leq |V^*| - |A| + |S| < 0 + |S| = |S| \implies |\mathcal{N}(S)| < |S|$$

which proves that there is at least one set S for which the *Hall condition* does not hold. □

2.2.2 Tutte's theorem

In the general case, what obstructs the existence of a perfect matching in a graph? Consider a graph G; clearly, if n is odd, the graph does not admit perfect matchings, since each edge matches exactly two nodes, meaning that there will always be a free vertex in the graph.

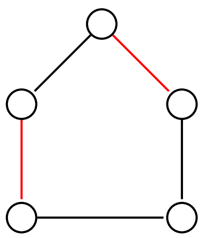
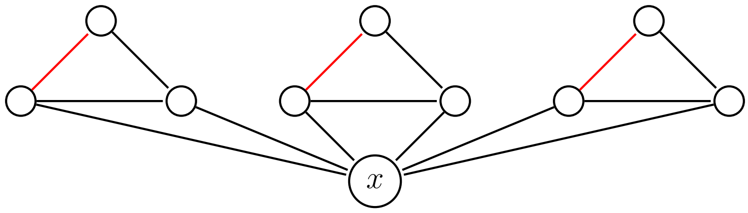


Figure 2.7: For instance, no matching of C_5 can be perfect.

For the same reasoning, if G has a connected component with an odd number of vertices, G does not admit perfect matchings. Hence, if G admits a perfect matching, there cannot be any connected component with an odd number of vertices. But is the converse true as well? Consider the following graph



This graph is connected, and has 10 vertices, but it does not admit perfect matchings, meaning that the converse does not hold. Can we find a condition that guarantees that a given graph always admits a perfect matching?

Given a graph G, let $\mathcal{O}(G)$ be the number of components with an odd number of vertices. The next theorem, proved by Lovász et al. [7] in 1947 shows that the following property

$$\forall S \subseteq V(G) \quad \mathcal{O}(G[V(G) - S]) \leq |S|$$

which we will refer to as **Tutte condition** — is a *necessary* and *sufficient* condition to guarantee that a graph admits a perfect matching.

Lemma 2.1

Given a graph G, consider a graph G' obtained by adding edges to G; if G satisfies the Tutte condition, then G' satisfies the Tutte condition as well.

Proof. By contrapositive, suppose that G' fails the Tutte condition, meaning that there exists a set $S \subseteq V(G') = V(G)$ such that $\mathcal{O}(G[V(G') - S]) > |S|$. Since G' is obtained by adding edges to G, the number of odd components in G' can either remain the same, or decrease if two different components having an odd number of vertices became connected in G'. Therefore, we have that

$$|S| < \mathcal{O}(G[V(G') - S]) \leq \mathcal{O}(G[V(G) - S])$$

meaning that G fails the Tutte condition as well. $\square$

We are now ready to prove Tutte's theorem.

Theorem 2.5: Tutte's theorem

A graph G admits a perfect matching if and only if for any $S \subseteq V(G)$ it holds that $\mathcal{O}(G[V(G) - S]) \leq |S|$ — meaning that it satisfies the Tutte condition.

Proof. TODO $\square$

todo

2.3 Stable matching

Matchings in bipartite graphs are particularly useful as they can be applied to model various types of problems across different fields. One well-known example is the stable matching problem, which arises in scenarios like job assignments, college admissions, and matchmaking systems. In this problem, the goal is to find a *stable pairing* between two sets of entities — such as students and universities — where no two unmatched entities would prefer each other over their current assignments.

Definition 2.9: Stable matching

Given a graph G, bipartitioned through (A, B), and a matching M of G, consider a family of *preference* functions $\{w_v\}_{v \in V(G)}$ that for each vertex $v \in V(G)$, assign a value to the edges $vu \in E(G)$, for all $u \in \mathcal{N}(v)$

$$\forall v \in V(G) \quad w_v : \mathcal{N}(v) \to \mathbb{R}$$

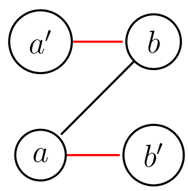
We say that M is **stable** if for each $ab \in E(G)$ it does not happen that

$$(a \text{ free } \vee (\exists ab' \in M \quad w_a(b) > w_a(b')))$$

$$\wedge$$

$$(b \text{ free } \vee (\exists a'b \in M \quad w_b(a) > w_b(a')))$$

For instance, consider the following bipartite graph G, and a matching M — outlined in *red*:



If the family of preference functions $\{w_v\}_{v \in V}$ is such that

$$w_a(b) > w_b(b') \wedge w_b(a) > w_b(a')$$

which implies that both a and b would prefer to *switch* their current matched vertex — then M is *not* stable. Note that the empty matching is *not* stable, by definition.

The following theorem, proved by Gale et al. [4] in 1962, proves that a stable matching can be always constructed in a bipartite graph, regardless of the preference functions.

Theorem 2.6: Gale-Shapley theorem

Given a bipartite graph G, and a family of preference functions $\{w_v\}_{v \in V(G)}$, there exists a stable matching of G.

Proof. Before proving the theorem, we need to introduce some definitions.

Consider a bipartite graph G, bipartitioned through (A, B), and consider a matching M. Given two vertices $a \in A$ and $b \in B$, we say that a is **acceptable to** b if

- b is free, or
- b is matched to a vertex a', and $w_b(a) > w_b(a')$

Moreover, we say that a vertex $a \in A$ is **happy** if either

- a is free, or
- $ab \in M$ and for each b' such that a is acceptable to b', it holds that $w_a(b) \geq w_a(b')$

Observe that in the empty matching every vertex is happy.

Claim: Consider a matching M such that each vertex is happy; if M is not stable, then there exists a vertex $a \in A$ such that a is free and acceptable to some $b \in B$.

Proof of the Claim. By instability of M, there must be an edge $ab \notin M$ such that a is free, or it prefers b to its current partner, and vice versa. By way of contradiction, suppose that a is matched to some $b' \in B$, hence by instability of M through ab we know that $w_a(b) > w_a(b')$

- If b is free, then by definition a is acceptable to b; however, by happiness of a it must be that $w_a(b') \geq w_a(b)$ $\lightning$
- If b is matched by some edge $a'b \in M$, by instability of M through ab we know that $w_b(a) > w_b(a')$, hence a is acceptable to b, and by happiness of a it must be that $w_a(b') \geq w_a(b)$ $\lightning$

This implies that a must be free; therefore, we have that

- If b is free as well, then by definition a is acceptable to b
- If b is matched by some edge $a'b \in M$, by instability of M through ab we know that $w_b(a) > w_b(a')$, hence a is still acceptable to b

$\square$

Now, consider another matching M' of G; we say that M is **better than** M' if

- $\forall a'b \in M' \quad \exists ab \in M \quad w_b(a) \geq w_b(a')$, meaning that every vertex $b \in B$ prefers its match in M at least as much as its match in M', and
- $\exists a'b \in M', ab \in M \quad w_b(a) > w_b(a')$, meaning that there is at least one vertex b that strictly perfers its match in M over its match in M'

In other words, M is better than M' if no match in M is worse than in M', and at least one match is strictly better.

Let M_k be an unstable matching such that every vertex is happy; therefore, by the previous claim we know that there exists a vertex $a \in A$ such that a is free and and acceptable to some $b \in B$. We will construct a matching M_{k+1} as follows:

$$b^* \in \underset{\substack{b \in \mathcal{N}(a): \\ a \text{ acceptable to } b}}{\arg\max} \; w_a(b) \implies M_{k+1} := (M_k \cup \{ab^*\}) - \{a'b^* \in M_k \mid a' \in A\}$$

meaning that M_{k+1} is obtained from M_k by adding the edge ab^*, where b^* maximizes $w_a(b^*)$, and removing the edge $a'b^*$ from M_k, if present — the last set is either $\{a'b^*\}$ or $\varnothing$.

Claim: M_{k+1} is better than M_k, and if M_k is ensures that every vertex is happy, M_{k+1} does as well.

Proof of the Claim. First, we prove that M_{k+1} is better than M_k. The first condition that M_{k+1} has to satisfy in order to be better than M_k is true simply because we removed the edge $a'b^*$ if it was present in M_k, and the rest of the matching has not been altered. Moreover, if b^* was free then the second condition is vacuously true, otherwise if the edge $a'b^*$ was present in M_k, in M_{k+1} there we added the edge ab^* and we know that $w_{b^*}(a) > w_{b^*}(a')$ because a is acceptable to b^* by definition.

Now, assume that M_k is such that every vertex is happy. Since b^* is the neighbor of a that maximizes $w_a(b^*)$, a is happy by definition. Now, if b^* was free in M_k, then we only added ab^* to M_{k+1}, hence every vertex is happy w.r.t. M_{k+1}. Otherwise, if b^* was not free in M_k, i.e. there was an edge $a'b^* \in M_k$, by definition M_{k+1} will not contain $a'b^*$, meaning that a' is free w.r.t. M_{k+1}, hence a' is happy by definition. □

Now, consider the empty matching $M_0 := \varnothing$; we already observed that M_0 is not stable, but is such that each vertex is happy, therefore by the previous claim we can extend M_0 into M_1 through some vertex $a \in A$ that satisfied the condition of the first claim. In particular, since we proved that this process preserves the happiness of the vertices, we can repeat this process as long as the current matching M_i is still unstable.

TODO □

scrivi bene parte finale

2.4 Exercises

Problem 2.1

Let G be a k-regular bipartite graph, bipartitioned through (A, B). Prove that

1. $|A| = |B|$
2. G has a perfect matching

Proof. Since G is k-regular, it holds that the number of edges that have an endpoint in A is precisely $k\,|A|$, and the number of edges that have an endpoint in B is exactly $k\,|B|$; moreover, since G is bipartite, we have that

$$k\,|A| = k\,|B| \implies |A| = |B|$$

We will prove the second statement by using Theorem 2.4. By way of contradiction, assume that there is a set $S \subseteq V(G)$ such that $|S| > |\mathcal{N}(S)| \implies k\,|S| > k\,|\mathcal{N}(S)|$. Applying the same argument as before, the number of edges ab such that $a \in S, b \in \mathcal{N}(S)$ is exactly $k\,|S|$, hence by the pigeonhole principle there must be at least one vertex in $\mathcal{N}(S)$ that has more than k vertices, contradicting the k-regularity of G. $\square$

Bibliography

[1] Claude Berge. “TWO THEOREMS IN GRAPH THEORY”. In: *Proceedings of the National Academy of Sciences* 43.9 (Sept. 1957), 842–844. ISSN: 1091-6490. DOI: 10.1073/pnas.43.9.842. URL: http://dx.doi.org/10.1073/pnas.43.9.842.

[2] Wikipedia contributors. *Seven Bridges of Königsberg*. Jan. 2025. URL: https://en.wikipedia.org/wiki/Seven_Bridges_of_K%C3%B6nigsberg.

[3] Leonhard Euler. “Solutio problematis ad geometriam situs pertinentis”. In: *Commentarii academiae scientiarum Petropolitanae* (1741), pp. 128–140.

[4] David Gale et al. “College admissions and the stability of marriage”. In: *The American mathematical monthly* 69.1 (1962), pp. 9–15.

[5] P. Hall. “On Representatives of Subsets”. In: *Journal of the London Mathematical Society* s1-10.1 (Jan. 1935), 26–30. ISSN: 0024-6107. DOI: 10.1112/jlms/s1-10.37.26. URL: http://dx.doi.org/10.1112/jlms/s1-10.37.26.

[6] Denés Konig. *Gráfok és mátrixok. Matematikai és Fizikai Lapok, 38: 116–119, 1931*.

[7] L. Lovász et al. *Matching Theory*. AMS Chelsea Pub., 2009. URL: https://books.google.it/books?id=OaoJBAAAQBAJ.