Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

Asymmetric graphs

Bachelor Thesis

2022
Simona Dubeková

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

Asymmetric graphs

Bachelor Thesis

Study Programme:	Computer Science
Field of Study:	Computer Science
Department:	Department of Computer Science
Supervisor:	doc. RNDr. Tatiana Jajcayová, PhD.
Consultant:	Mgr. Dominika Mihálová

Bratislava, 2022
Simona Dubeková

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Simona Dubeková
Študijný program: aplikovaná informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: anglický
Sekundárny jazyk: slovenský

Názov: Asymmetric graphs
Asymetrické grafy

Anotácia:

Vedúci: doc. RNDr. Tatiana Jajcayová, PhD.
Katedra: FMFI.KAI - Katedra aplikovanej informatiky
Vedúci katedry: prof. Ing. Igor Farkaš, Dr.

Dátum zadania: 01.10.2021

Dátum schválenia: 06.10.2021 doc. RNDr. Damas Gruska, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

THESIS ASSIGNMENT

Name and Surname:	Simona Dubeková
Study programme:	Applied Computer Science (Single degree study, bachelor I. deg., full time form)
Field of Study:	Computer Science
Type of Thesis:	Bachelor´s thesis
Language of Thesis:	English
Secondary language:	Slovak

Title: Asymmetric graphs

Annotation: We study non-oriented simple graphs. We call a graph symmetric, if there exists a non-identical permutation of its vertices, which leaves the graph invariant, i.e. a graph is called symmetric if the group of its automorphisms is not trivial. A graph which is not symmetric will be called asymmetric. The degree of symmetry of a symmetric graph is measured by the size of its group of automorphisms. We will measure the degree of asymmetry of an asymmetric graph by the number of vertices which we have to delete to obtain a symmetric graph.

Supervisor:	doc. RNDr. Tatiana Jajcayová, PhD.
Department:	FMFI.KAI - Department of Applied Informatics
Head of department:	prof. Ing. Igor Farkaš, Dr.

Assigned: 01.10.2021

Approved: 06.10.2021 doc. RNDr. Damas Gruska, PhD.
Guarantor of Study Programme

Student Supervisor

Acknowledgments: I would like to thank my supervisor RNDr. Tatiana Jajcayová, PhD. for her great patience and support throughout the creation of the bachelor's thesis. I would also like to thank my consultant Mgr. Dominika Mihálová for her detailed explanation of system for computational discrete algebra.

Abstrakt

V tejto bakalárskej práci skúmame neorientované jednoduché grafy. Graf nazývame symetrický, ak existuje neidentická permutácia jeho vrcholov, ktorá ponechá graf invariantný, t.j. graf sa nazýva symetrický, ak grupa jeho automorfizmov nie je triviálna. Graf, ktorý nie je symetrický, nazývame asymetrický. Stupeň asymetrie asymetrického grafu meriame počtom vrcholov, ktoré musíme odstrániť, aby sme získali graf symetrický. Neorientované jednoduché asymetrické grafy vytvárame pomocou programovacieho jazyka Kotlin a pre jednoduchšiu predstavu ich vykresľujeme pomocou systému vytvoreného pre výpočtovú diskrétnu algebru - GAP. Z týchto asymetrických grafov postupne, po jednom, odstraňujeme všetky vrcholy a všetky hrany. Pri tomto postupnom odstraňovaní vrcholov a hrán z týchto asymetrických grafov skúmame, ako sa mení symetria týchto grafov oproti grafom pôvodným. Stupeň asymetrie asymetrického grafu potom môžeme vypočítať pomocou počtu vrcholov, ktoré musíme odstrániť, aby sme získali graf symetrický.

Kľúčové slová: graf, vrchol, hrana, asymetrický graf, stupeň asymetrie

Abstract

In this bachelor thesis we study non-oriented simple graphs. A graph is called symmetric if there is a non-identical permutation of its vertices that leaves the graph invariant, i. a graph is called symmetric if its group of automorphisms is not trivial. A graph that is not symmetric is called asymmetric. We measure the degree of asymmetry of an asymmetric graph by the number of vertices which we have to delete to obtain a symmetric graph. We create non-oriented simple asymmetric graphs using the Kotlin programming language and we draw them for a better visualization using a system created for computational discrete algebra - GAP. From these asymmetric graphs, one by one, we remove all vertices and all edges. In this gradual deletion of vertices and edges from these asymmetric graphs, we study how the symmetry of these graphs changes from the original graphs. The degree of asymmetry of an asymmetric graph can be measured by the number of vertices which we have to delete to obtain a symmetric graph.

Keywords: graph, vertex, edge, asymmetric graph, degree of asymmetry

Contents

1 Preliminaries **1**

1.1 Graph theory 1

1.1.1 Graphs in Real Life 1

1.1.2 Graphs basics 2

1.1.3 Bipartite graphs 3

1.1.4 Subgraphs 4

1.1.5 Paths, circuits, and reachability in graphs 4

1.1.6 Trees 5

1.1.7 Spanning trees 6

1.1.8 Prim's and Kruskal's algorithm 6

1.2 Group theory 6

1.2.1 Group isomorphism 7

1.2.2 Group automorphism 7

1.3 Theory of symmetries 8

1.3.1 Graph isomorphism 8

1.3.2 Symmetric graphs 9

1.3.3 Partial symmetries of graphs 10

1.3.4 Asymmetric graphs 10

List of Figures

1.1 Tram Lines 1
1.2 Constellation Graph 2
1.3 Examples of bipartitegraphs 3
1.4 Examples of trees 5
1.5 Kruskal's and Prim's algorithms 6
1.6 Example group isomorphism 7
1.7 Example of symmetric and asymmetric graph 8
1.8 Petersen graph 9
1.9 The 18 minimal asymmetric graphs 11

List of Tables

1.1 Example group isomorphism 8

Chapter 1

Preliminaries

This chapter is devoted to necessary knowledge of graph theory, group theory and theory of symmetries which will be used in this thesis.

1.1 Graph theory

This section contains basic definitions of graph theory and graphs in real life. It is divided into several subsections, where each subsection describes graphs and terms which are connected with this topic. We will use figures and tables for better understanding of graph theory.

Figure 1.1: Tram Lines

1.1.1 Graphs in Real Life

Many people think of graphs just as a series of connected or unconnected dots, but they do not realize all the possibilities which graph theory offers. One of the most common example of graph in real life is road network. Individual cities represent vertices and the roads represent the edges of the graph. It is the best example of a graph, such as tram network in Figure 1.1. Another example could be constellations, which are not immediately thought of as an easy example of a graph. As seen in Figure 1.2, constellations consist of stars, which are the vertices, and their imaginary connections

- edges. Among other uses of graphs belong cardiovascular system, friends of social media or the internet itself. Graphs can be used for quite different things and they are effective visual tools because they present information quickly and easily.

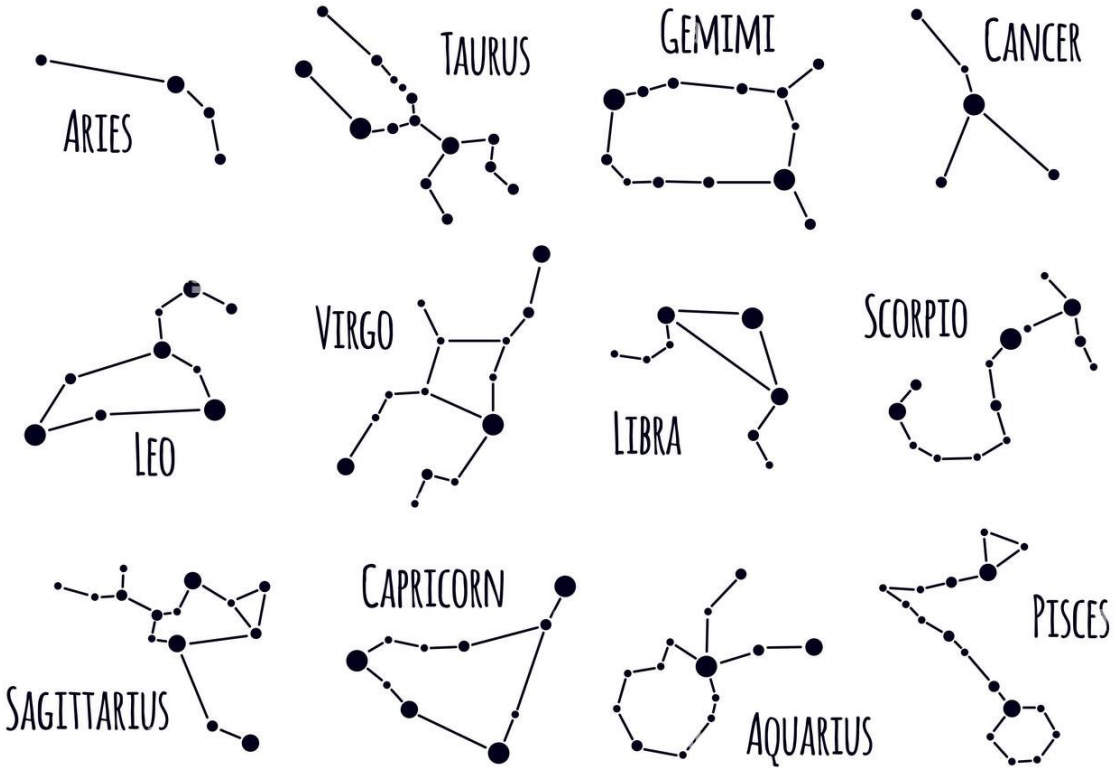

Figure 1.2: Constellation

1.1.2 Graphs basics

Definition. A *general graph* is a triple $G = (V, E, \phi)$, where V is a nonempty set of vertices (or nodes) of G, and E is set of edges of G, and ϕ, called the edgemap, is a function $\phi : E \to \mathcal{P}(V)$, where $|\phi(e)| = 1$ or 2, for each $e \in E$. The vertices in $\phi(e)$ are called endpoints of the edge e. An edge e having only one endpoint (i.e., $|\phi(e)| = 1$) is called a self-loop. Two edges, e_1, e_2 that have the same endpoints (i.e., $\phi(e_1) = \phi(e_2)$) are called parallel edges or multiedges. [4]

In fact, we don't need all the graphs for this work. All we need are non-oriented simple graphs, because they are all we need for the purposes of analysing asymmetry.

Definition. A *simple graph* is an ordered pair of sets $G = (V, E)$, where V is a nonempty set of vertices (or nodes) of G, and E is set of edges of G is a set of two-element pairs (2-combinations) of vertices. Thus, each edge of G can be expressed as $/e, v/$ are called the endpoints of the of the edge. The edge $\{u, v\}$ is said to join u and v, and the edge

is said to be incident to either of its endpoints. Any two vertices in in G that are joined by an edge are said to be adjacent, and are called neighbors. A vertex with no neighbors is called isolated. [4]

Basically, a graph with no loops and no parallel edges is called a simple graph. The maximum number of edges possible in a single graph with n vertices is $n(n-1)/2$ and the number of simple graphs possible with n vertices is

$$2^{\frac{n\cdot(n-1)}{2}} \tag{1.1}$$

1.1.3 Bipartite graphs

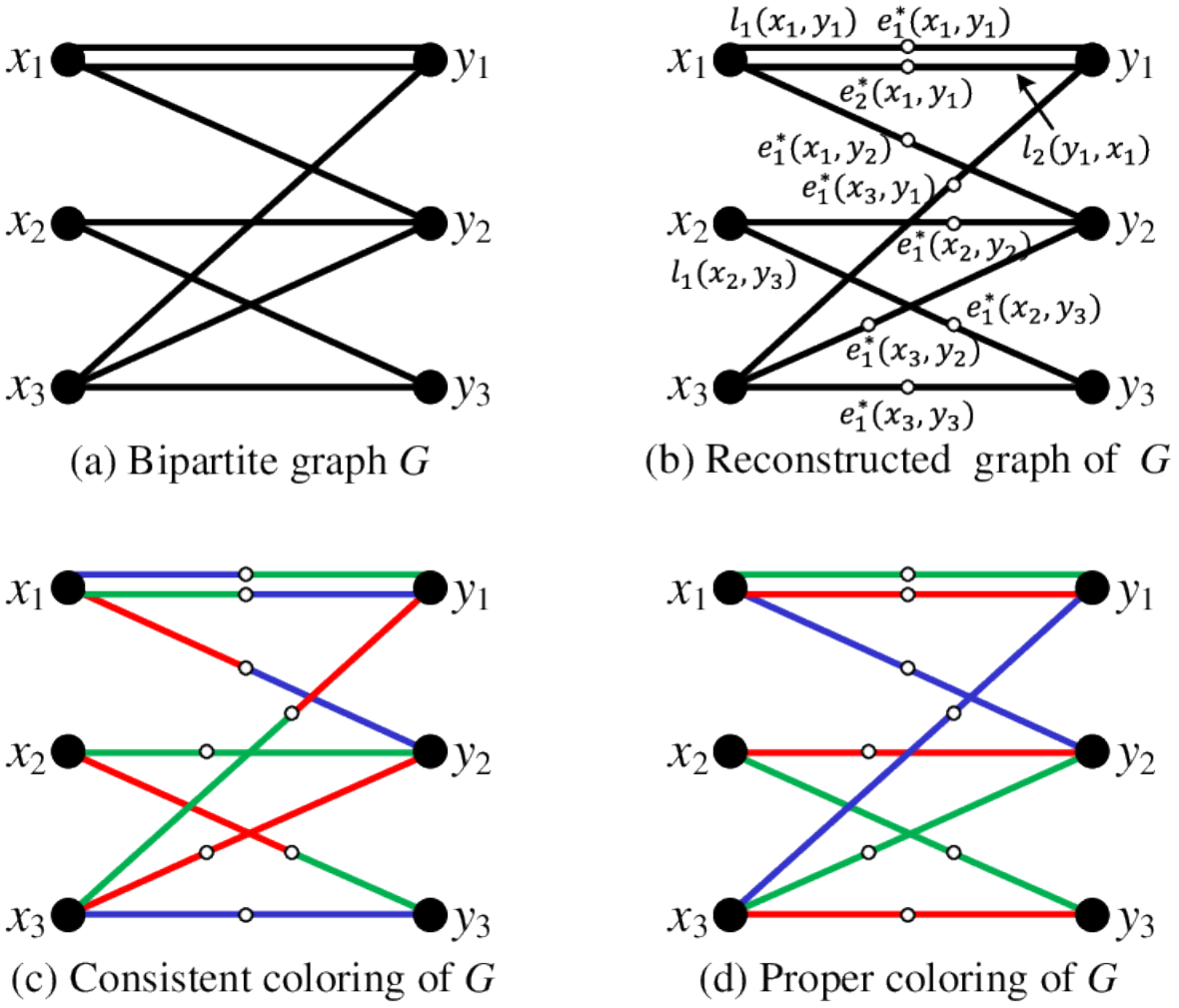

Figure 1.3: Bipartite Graphs: a) bipartite graph G, b) reconstructed graph of G, c) consistent colouring of G, b) proper colouring of G

An important class of graphs are those in which all vertices can be painted "black" or "red" in such a way, that adjacent vertices will never have the same colour. For example, if we consider the graph whose vertices are the men and women at the ball,

and whose edges link each man to the women whom he danced with during the evening, then this graph will fulfill this property if we paint the man vertices "black" and the women vertices "red".

Definition. A graph G is bipartite if the vertex set V can be partitioned into two subsets: $V = U \cup W$, such that each edge of G has one endpoint in U and one endpoint W. The pair U, W is called a (vertex) bipartition of G. [4]

In other words, the graph is a bipartite when the vertex set of can be partitioned into two disjoint and independent sets and all the edges from the edge set have one endpoint vertex from the set and another endpoint vertex from the set.

1.1.4 Subgraphs

For example, as there is a subset in set theory, there is a subgraph in graph theory.

Definition. Suppose that $G = (V, E)$ is a graph. Another graph $H = \left(V', E'\right)$ is said to be a *subgraph* of G, if $V' \subseteq V$ and $E' \subseteq E$. More generally, we use this terminology, or simply say G contains copy of H if the vertices of H can be relabelled with some of G's vertex labels in such a way that all of the edges of H are edges of G. [4]

Simplified, subgraph is only part of the graph.

1.1.5 Paths, circuits, and reachability in graphs

Definition. Suppose that $G = (V, E)$ is a graph, and $v, w \in V$ are pair of vertices. A *path* in G from v to w is an alternating sequence of vertices and edges:

$$P = \langle v_0, e_1, v_1, e_2, v_2, ..., v_{k-1}, e_k, v_k \rangle \,, \tag{1.2}$$

such that the endpoints of edge e_i are vertices $\{v_{i-1}, v_i\}$, for $1 \leq i \leq k, v_0 = v$, and $v_k = w$. We say the path P passes through the vertices $v_0, v_1, v_2, ..., v_{k-1}, v_k$, and traverses the edges $e_1, e_2, ..., e_k$, and that the path has length k, since it traverses k edges. If such a path exists, we say that the vertex w is *reachable* from the vertex v in G. A path having positive length ($k > 0$) from any vertex to itself is called a *circuit*. A path (or circuit) is called simple if it never traverses the same edge twice. A simple circuit that does not pass through the same vertex twice (except for the initial and final vertex) is called a cycle. [4]

Observe that in the case of a simple graph, the above path can be specified by the sequence of vertices $\langle v_0, v_1, v_2, ..., v_{k-1}, v_k \rangle$, since the corresponding edges are uniquely determined. Also note, that any vertex is reachable from itself by the path $\langle v_0 \rangle$ of length zero. [4]

1.1.6 Trees

A connected simple graph with no cycles is called a tree.

Definition. If T is a simple graph on n vertices, then the following statements are logically equivalent:

1. T is a tree.
2. T is connected and has $n-1$ edges.
3. T has no cycles and has $n-1$ edges.
4. Any two vertices of T are joined by a unique simple path in T.
5. T is connected, and every edge is a bridge (i.e., deleting an edge renders the graph disconnected).
6. T has no cycles and if we add any new edge to T (between two of its vertices) the resulting simple graph will have a unique cycle. [4]

Trees are very important graphs, and they have been used in numerous applications. The most familiar example is a family tree. Almost everyone has drawn a family tree already in playschool. The folder structure in any computer operating system or network has also the structure of a tree. Trees are the basis for some of the most efficient search and sorting algorithms. For these resons trees are the most important graphs in computer science and that is why we chose them in this thesis.

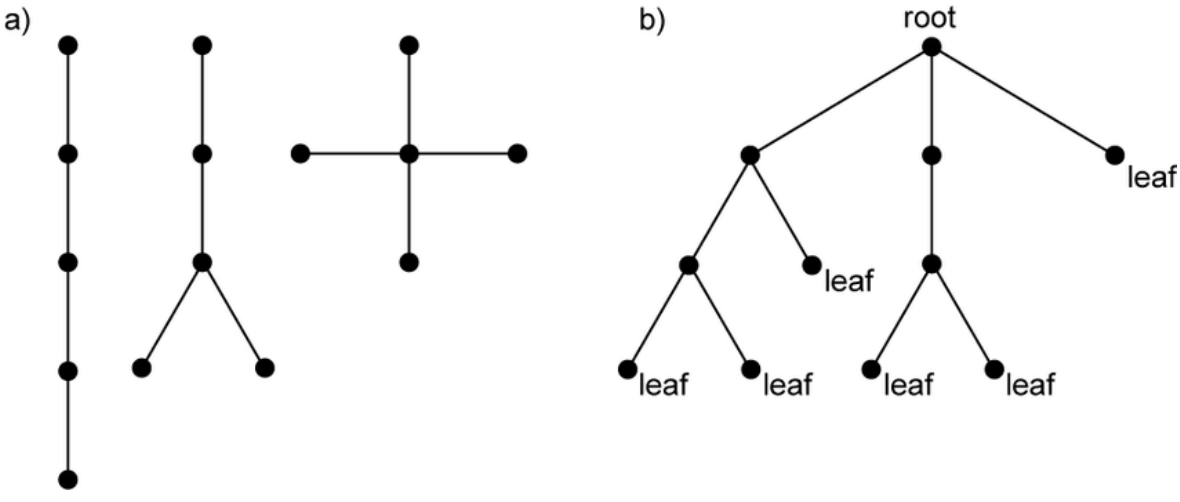

Figure 1.4: Trees: a) examples of non-isomorphic trees, b) a tree with a root and six leaves

1.1.7 Spanning trees

A spanning tree is the maximum possible tree in a graph, such that adding an edge would create a cycle. It is defined as: A spanning tree of a graph on n vertices is a subset of $n-1$ edges that form a tree. A tree consists of leaves, that only have 1 adjacent vertex, and inner vertices.

1.1.8 Prim's and Kruskal's algorithm

The algorithms of Kruskal and Prim are both minimum spanning tree algorithms. The difference between Prim's and Kruskal's algorithm is in specific rule that talks about how to determine a safe edge.

```
MST-KRUSKAL(G, w)
1  A = ∅
2  for each vertex v ∈ G.V
3      MAKE-SET(v)
4  sort the edges of G.E into nondecreasing order by weight w
5  for each edge (u, v) ∈ G.E, taken in nondecreasing order by weight
6      if FIND-SET(u) ≠ FIND-SET(v)
7          A = A ∪ {(u, v)}
8          UNION(u, v)
9  return A
```

$A = \{(v, v.\pi) : v \in V - \{r\}\}$.

```
MST-PRIM(G, w, r)
1  for each u ∈ G.V
2      u.key = ∞
3      u.π = NIL
4  r.key = 0
5  Q = G.V
6  while Q ≠ ∅
7      u = EXTRACT-MIN(Q)
8      for each v ∈ G.Adj[u]
9          if v ∈ Q and w(u, v) < v.key
10             v.π = u
11             v.key = w(u, v)
```

Figure 1.5: Kruskal's and Prim's algorithms

In Kruskal's algorithm, the set A is a forest whose vertices are all those of the given graph. The safe edge added to A is always a least-weight edge in the graph that connects two distinct components. In Prim's algorithm, the set A forms a single tree. The safe edge added to A is always a least-weight edge connecting the tree to a vertex not in the tree. [5] Kruskal's algorithm's time complexity in worst case is $O(E \log E)$, this is because we need to sort the edges. Prim's algorithm's time complexity in worst case is $O(E \log V)$ with priority queue or even better, $O(E + V \log V)$ with Fibonacci Heap. We should use Kruskal's algorithm when the graph has small number of edges, like $E = O(V)$, when the edges are already sorted or if we can sort them in linear time. In other hand, we should use Prim's algorithm when the graph's number of edges is high, like $E = O(V^2)$.

1.2 Group theory

Group is This section contains basic definitions of group theory, more precisely definitions of group isomorphism and group automorphism. A group is a pair, where the

first item is a finite or infinite set of elements and the second item is a binary operation. This binary operation is a function, which combines two elements (not necessarily distinct) from the set, forming a third one, which also has to be in that specific set. This is a fundamental property of groups and it is called closure. Groups also have other properties, which distinguish them from other types of structures in graph theory. The other three properties that a group has to satisfy are associativity, existence of identity and existence of inverse. If a group also satisfies a condition called commutativity, it is referred to as an abelian group, but sometimes it is referred to as a commutative group.

1.2.1 Group isomorphism

An *isomorphism* ϕ from a group G to a group $\overline{G}$ is one-to-one mapping (or function) from G onto $\overline{G}$ that preserves the group operation. That is,

$$\phi(ab) = \phi(a)\phi(b) \tag{1.3}$$

for all a, b in G. If there is an isomorphism from G onto $\overline{G}$, we say that G and $\overline{G}$ are *isomorphic* and write $G \approx \overline{G}$.

This definition can be visualized as shown in Figure 1.6. The pairs of dashed arrows represent the group operations.

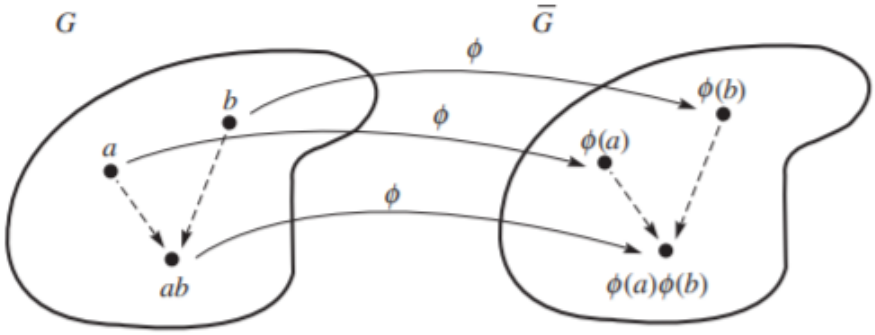

Figure 1.6: Group Isomorphism

It is implicit in the definition of isomorphism that isomorphic groups have the same order. It is also implicit in the definition of isomorphism that the operation on the left side of the equal sign is that of G, whereas the operation on the right side is that of $\overline{G}$. The four cases involving $\cdot$ and $+$ are shown in Table 1.1.

[1]

1.2.2 Group automorphism

An isomorphism from a group G onto itself is called an *automorphism* of G. [1]

Table 1.1: Group Isomorphism

G Operation	$\overline{G}$ Operation	Operation Preservation
$\cdot$	$\cdot$	$\phi(a \cdot b) = \phi(a) \cdot \phi(b)$
$\cdot$	$+$	$\phi(a \cdot b) = \phi(a) + \phi(b)$
$+$	$\cdot$	$\phi(a + b) = \phi(a) \cdot \phi(b)$
$+$	$+$	$\phi(a + b) = \phi(a) + \phi(b)$

1.3 Theory of symmetries

Most people think of symmetry as beauty of form arising from balanced proportions. But in mathematics it means the property of being symmetrical, especially in correspondence in size, shape, and relative position of parts on opposite sides of a dividing line or median plane or about a centre or axis.

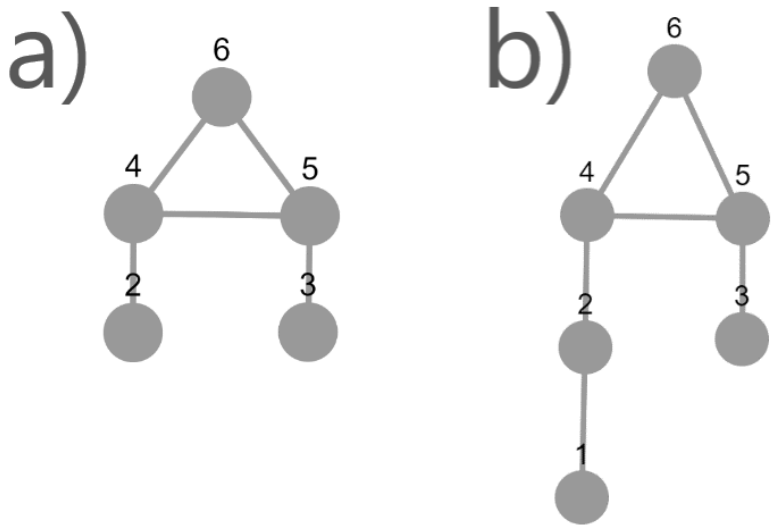

Figure 1.7: Graphs: a) example of symmetric graph, b) example of asymmetric graph

1.3.1 Graph isomorphism

Definition. Suppose that $G = (V, E)$ and $G' = \left(V', E'\right)$ are simple graphs, that $f : V \to V'$ is one-to-one (vertex) function.

(a) The function f is said to be preserve adjacency if for any pair of vertices $u, v \in V$, we have $\{u, v\} \in E \Rightarrow \{f(u), f(v)\} \in E'$. In other words, if u and v are neighbours in G, then their images $f(u)$ and $f(v)$ must be neighbours in G'.

(b) The function f is said to be preserve non-adjacency if for any pair of vertices $u, v \in V$, we have $\{u, v\} \notin E \Rightarrow \{f(u), f(v)\} \notin E'$. In other words, if u and v are not neighbours in G, then their images $f(u)$ and $f(v)$ must not be neighbours in G'.

(c) The function f is said to be a graph isomorphism from G to G' if it is bijective, and preserves both adjacency and non-adjacency. In this case we that the graphs G to G' are isomorphic, and write this as $G \cong G'$. If no such isomorphism exists, we say that G and G' are not isomorphic and write $G \ncong G'$. [4]

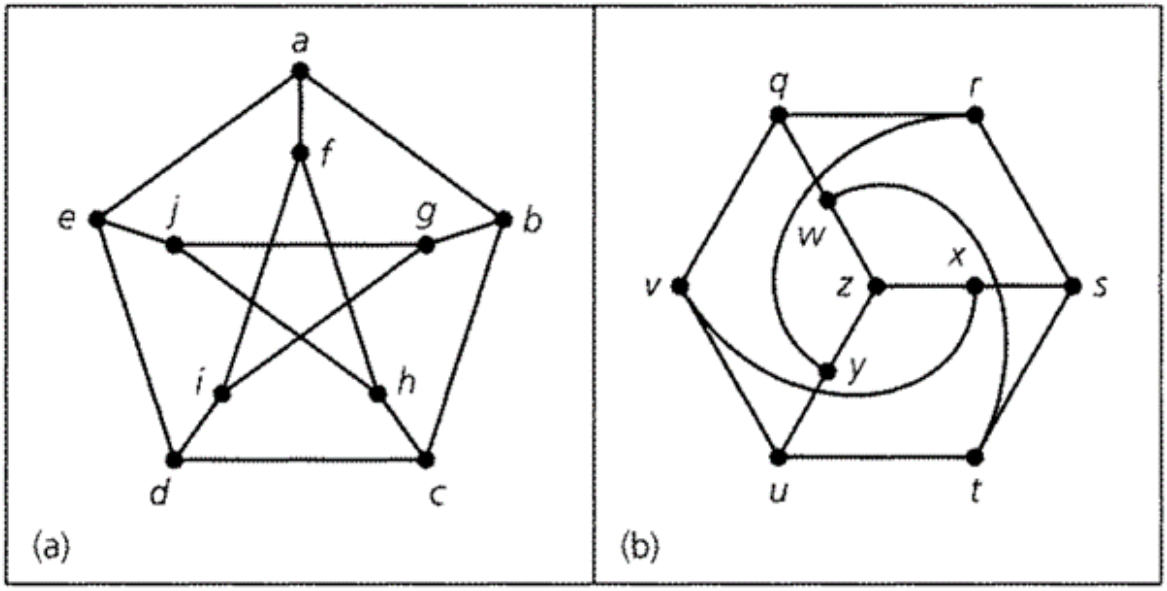

Figure 1.8: Two isomorphic representations of the Petersen graph

1.3.2 Symmetric graphs

The concept of symmetry can be defined as a transformation of a mathematical structure, of a specified kind, that leaves specified properties of the structure unchanged. There are different types of symmetries, such as rotation and reflection. Graph symmetries are directly related to graph automorphisms, since the structure of a graph remains the same. Identity, as it is a trivial automorphism, is also a form of a symmetry. Identity can be referred to as a 0° rotation or a 360° rotation. The automorphism groups of a graph characterize its symmetries. Firstly, it is clearly visible that the structure of a graph remained the same. Although the position of vertices has changed, if the graph on the left would rotate around the centre of the graph by 60° clockwise, the result of that rotation would be the original automorphism. In Figure 1.8, another type of symmetry is shown. It is a reflection around the vertical axis, where all vertices and edges from the left side had been projected to the right side and vice versa. As a

property related to automorphism groups, composition of two symmetric translations creates another graph with the same structure, for example combining rotation and reflection. These examples showed the coherence between graph automorphism and graph symmetries. This means that graph automorphism can be helpful in the study of graphs symmetries.

1.3.3 Partial symmetries of graphs

A large number of graphs, whether symmetric or asymmetric, contain subgraphs that have symmetries. Such graphs are known partially symmetric.

Definition. A *partial automorphism* of an edge-colored digraph Γ (as well as of a digraph or a graph) is an isomorphism between two vertex-induced subgraphs of Γ, that is, a bijection $\varphi : V_1 \Rightarrow V_2$ between two sets of vertices $V_1, V_2 \subseteq V(\Gamma)$ such that any pair of vertices $u, v \in V_1$ satisfies the condition $(u, v) \in E_c$ if and only if $(\varphi(u), \varphi(v)) \in E_c$ for any color c. The set of all partial automorphisms of Γ together with the operation of the usual composition of partial maps form an inverse monoid, which we denote by $PAut(\Gamma)$. [2]

1.3.4 Asymmetric graphs

A graph is *asymmetric* if it does not have a nontrivial automorphism. In this thesis, we are interested in asymmetric graphs that are as small as possible. An undirected graph G on at least two vertices is *minimal asymmetric* if G is asymmetric and no proper induced subgraph of G on at least two vertices is asymmetric.

There are exactly 18 finite minimal asymmetric undirected graphs up to isomorphism. These are the 18 graphs depicted in 1.9. [3]

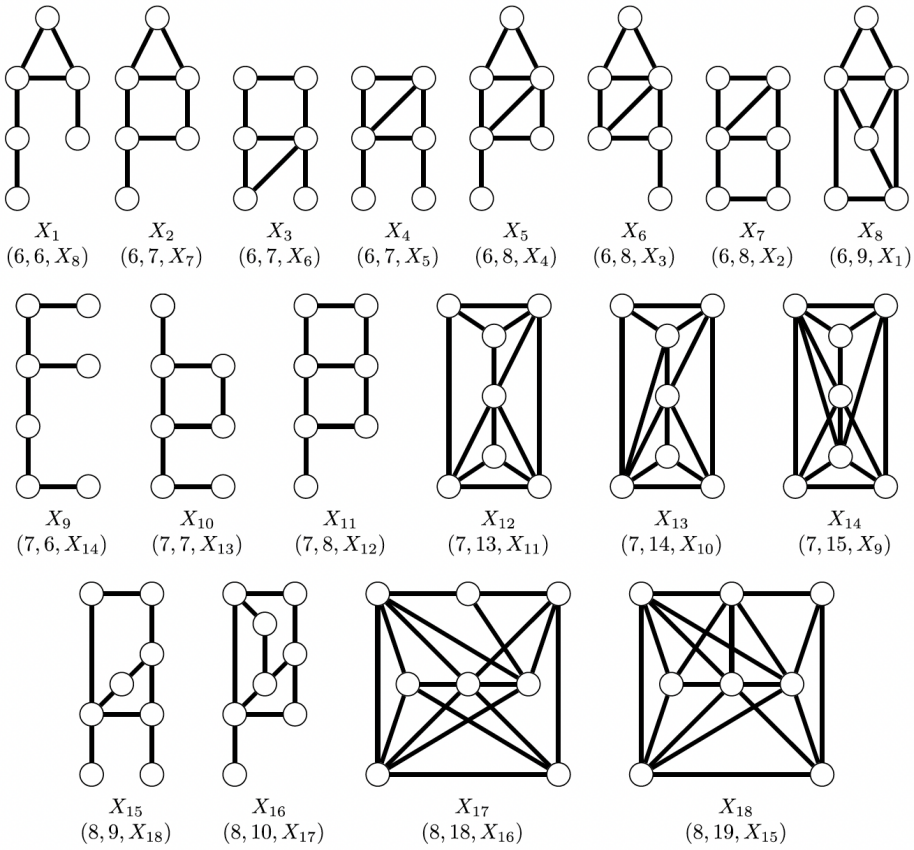

Figure 1.9: These are also the minimal involution-free graphs. For each graph the triple (n, m, co-G), describes the number of vertices, edges and the name of the complement graph, respectively. The graphs are ordered first by number of vertices and second by number of edges.

Bibliography

[1] Joseph A. Gallian. *Contemporary Abstract Algebra*. Brooks/Cole Publishing Co., 2010.

[2] Nóra Szakács Robert Jajcay, Tatiana Jajcayová and Mária B. Szendrei. Inverse monoids of partial graph automorphisms. *Journal of Algebraic Combinatorics*, 53(3):829–849, 2021.

[3] Pascal Schweitzer and Patrick Schweitzer. Minimal asymmetric graphs. *Journal of Combinatorial Theory, Series B*, 127(4):215–227, 2016. [Citované 2021-11-29] Dostupné z `https://www.researchgate.net/publication/301896061_Minimal_Asymmetric_Graphs`.

[4] Alexander Stanoyevitch. *Discrete Structures with Contemporary Applications*. Taylor & Francis Books, 2011.

[5] Ronald L. Rivest Clifford Thomas H. Cormen, Charles E. Leiserson and Stein. *Introduction to Algorithms*. Massachusetts Institute of Technology, 2009.