Introduction

Definition 1.1: An undirected, simple graph $G = \{V, E\}$ is a set of vertices V, and a set of edges, E, where E is a set of unordered pairs of vertices from V such that for all $u \in V$, $\{u, u\} \notin E$.

Example 1.1: GRAPHS

Remark: Graphs may have any number of vertices. However, unless otherwise specified, suppose any graph presented in this paper has a finite vertex set.

Definition 1.2: For a graph $G = \{V, E\}$, if $\{u, v\} \in E$, we say u and v are *adjacent*.

Example 1.2: adjacent and non-adjacent vertices.

Definition 1.3: Two graphs, $G = \{V, E\}$ and $G' = \{V', E'\}$, are *isomorphic*, denoted $G \cong G'$, if there exists a bijection $f : V \to V'$ such that for every $u, v \in V$, $\{u, v\} \in E$ if and only if $\{f(u), f(v)\} \in E'$. If such a function f exists, we call f a *graph isomorphism* from G to G'.

Example 1.3: isomorphic and nonisomorphic graphs

1.1 The Graph Isomorphism Problem

Definition 1.4: Given two graphs $G = \{V, E\}$ and $G' = \{V', E'\}$, the *graph isomorphism problem* is to determine if there exists a graph isomorphism from G to G'.

Example 1.4: isomorphic and non-isomorphic graphs

The graph isomorphism problem (GI) has a straightforward statement. Given any two graphs, $G = \{V, E\}$ and $G' = \{V', E'\}$, solving the problem requires proving the existence of

a bijection from V onto V' satisfying the conditions in Definition 1.3 (perhaps by producing such a bijeciton) or showing no such function exists. There are are some instances where solving the GI is as simple as the statement of the problem. If $V = V$ and $E = E'$, then the identity map is a graph isomorphism. On the other hand, since it is impossible to produce a bijection between two sets of different finite sizes, if $|V| \neq |V'|$ then G is not isomorphic to G'. Nearly all graph properties we care about must be shared between two graphs for there to exist a graph isomorphism between them. To understand some properties of graphs that must be preserved by a graph isomorphism, the following definition is useful.

Definition 1.5: A *walk* is a list $v_0, e_1, v_1, \ldots, e_k, v_k$ of vertices and edges such that for $1 \leq i \leq k$, the edge e_i has endpoints v_{i-1} and v_i. (West, 2002)

A $u, v-$*path* in a graph $G = \{V, E\}$ is a walk from vertex u to vertex v that has no repeated vertices. A graph is *connected* if there exists a $u, v-$path for all $u, v \in V$. If G is connected and G' is not connected, then G and G' are not isomorphic. The length of a $u, v-$path is the number of edges it contains. Thus, the longest $u, v-$path in a graph is the one containing the most edges. If the longest path in G is not the same length as that of G', then G and G' are not isomorphic. When the given graphs share common properties (e.g.,connected) and standard measures (e.g., longest path), solving the GI solution is more difficult. In fact, the computational complexity of GI is currently unsettled. The most recent advance is Babai (2018), which established a *quasi-polynomial time*, $n^{(\log n)^{\mathcal{O}(1)}}$, isomorphism test, where n denotes the number of vertices of the input graphs. Although GI is thought to be of polynomial time complexity, Babai's bound is the best established. Luks (1982) established a divide and conquer method, which laid the groundwork for Babai's (2018) result. Luks' paper shows that GI is polynomial time reducible to the *color automorphism problem*. On the other hand, there are tools for efficiently solving the GI for two graphs, even if they have thousands of vertices (see Nauty). There are also algorithms that are useful for giving a sense as to how similar two graphs are, even if they are not isomorphic. The next section

is dedicated to such an algorithm, the Weisfeler Lehman Algorithm (WL). WL is also successful on nearly all types of graphs at determining that two graphs are not isomorphic.

Weisfeler Lehman Algorithm

The Weisfeler Lehman Algorithm is useful for determining that two graphs are not isomorphic. We focus on what is called the one dimensional Weisfeler Lehman algorithm.

References