A Graph Isomorphism Algorithm

Glen Woodworth

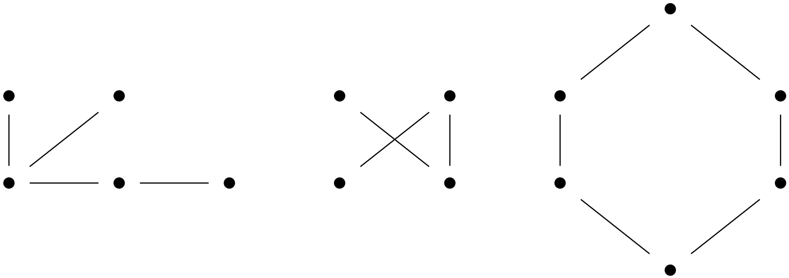
Department of Mathematics and Statistics
University of Alaska Fairbanks

March 2, 2023

Graphs

Definition

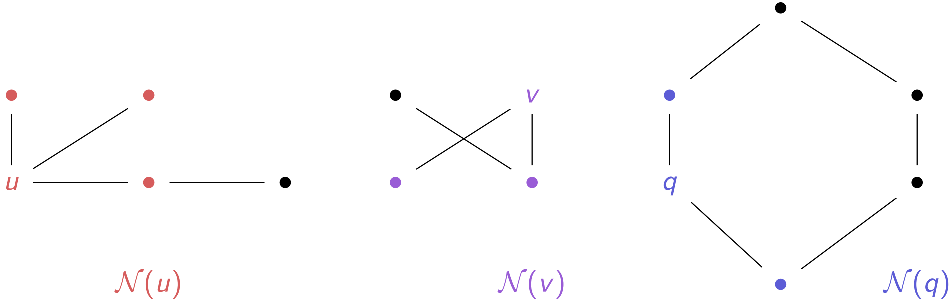
An undirected, simple graph $G = \{V, E\}$ is a set of vertices V, and a set of edges, E, where E is a set of unordered pairs of vertices from V such that for all $u \in V$, $\{u, u\} \notin E$.



Neighbors

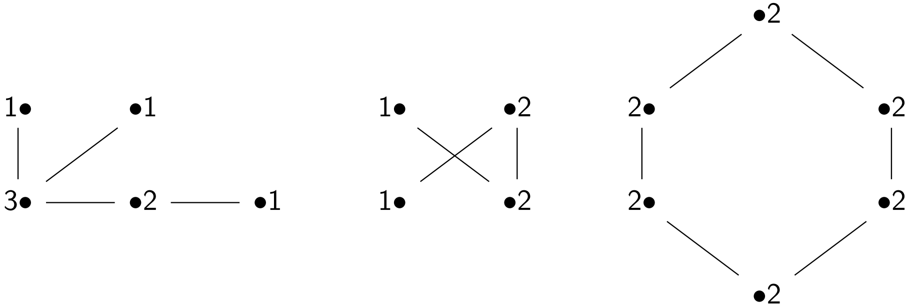
Definition

For a graph $G = \{V, E\}$, if $\{u, v\} \in E$, we say u and v are *adjacent* or *neighbors*. The *neighborhood* of $v \in G$, denoted $\mathcal{N}(v)$ is the set of all vertices $u \in G$ such that v and u are neighbors.

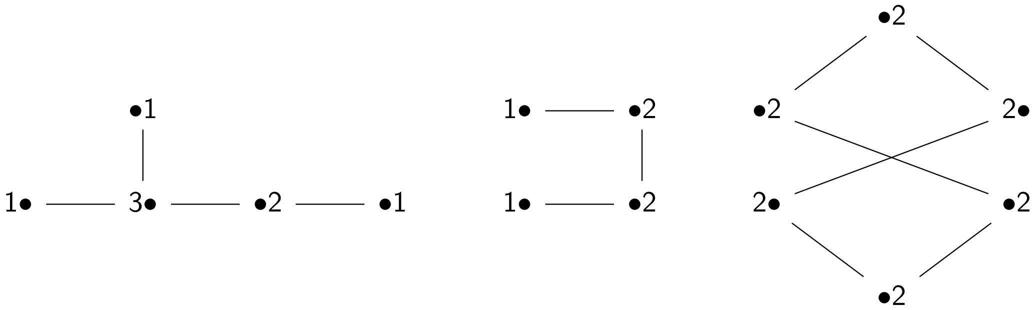


Labelling

We want to talk about properties of specific vertices, so we give a label to each vertex. The vertices in the graphs below are labelled using the cardinality of their neighborhoods, also called the *degree* of a vertex.



Rearrnging the vertices does not change a graph. We want to know when two graphs are the same, even though they may not look like it. The graphs below are the same as the graphs we have been looking at.



Similar or the same?

Before formalizing what we mean by two graphs $G = \{V, E\}$ and $G' = \{V', E,\}$ being the same, here are some obvious properties G and G' must share if they are the same, such as

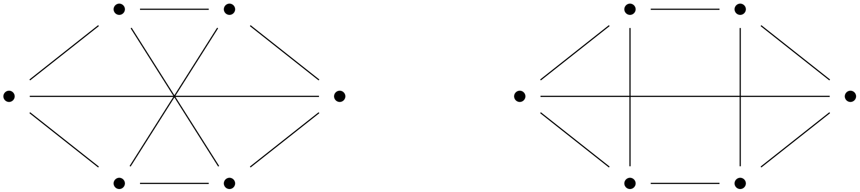
- the same number of vertices, $|V| = |V'|$
- the same number of edges, $|E| = |E'|$
- vertices with mathcing degrees, i.e., G has two vertices of degree 3 iff G' does too
- similar neighborhoods, i.e., $v \in V$ has a neighborhood containing 2 vertices of degree 2 iff there exists $v' \in V$ such that $\mathcal{N}(v')$ has the same property

Remark

These are necessary conditions that follow if G and G' are the same. Are they sufficient?

Similar but not the same

Those conditions are *not* sufficient for us to call two graphs the same. Here is an example:



- same number of vertices and edges (6 vertices, 9 edges each)✓
- every vertex has degree 3 in each graph ✓
- every neighborhood looks the same ✓
- *not* the same ×

Why not?

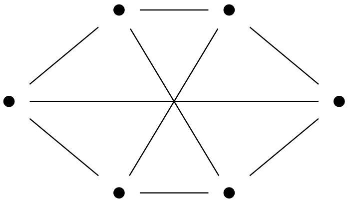
In order for two graphs to be the same, virtually *every* graph property we are interested in must be shared between the two graphs. To distunguish these graphs, we can use the following definition.

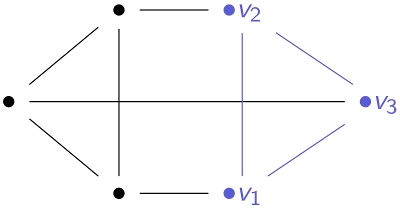
Definition

A *cycle* in a graph $G = \{V, E\}$ is an ordered list of vertices $c = (v_1, v_2, \ldots, v_n)$ where $v_i \in V$ are all unique, $\{v_1, v_n\} \in E$, and $\{v_i, v_{i+1}\} \in E$ for $1 \leq i \leq n$. Since c contains n vertices, we call c a n cycle.

Different cycles

Another graph property then is the number and type of cylces in the graph. The graph on the right has a 3 cycle (highlighted) but the graph on the left doesn't :(





Graph Isomorphism

Two graphs being the same is a high bar. The following definition formalizes exactly 'two graphs are the same' means.

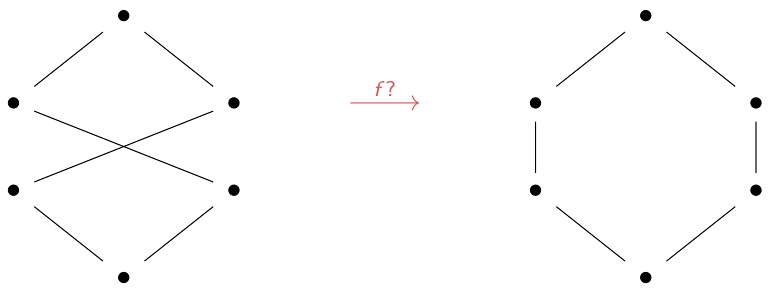
Definition

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'.

A hard problem (maybe?)

Problem

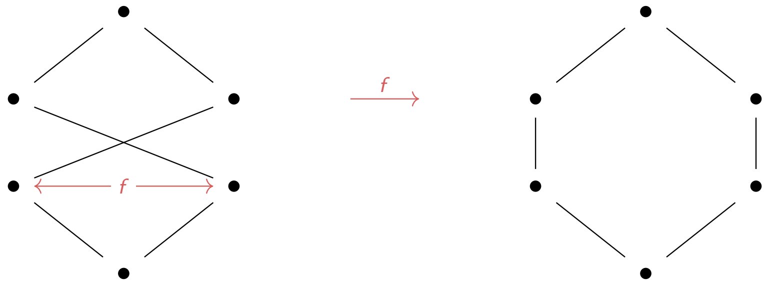
Given two graphs $G = \{V, E\}$ and $G' = \{V', E'\}$, the *graph isomorphism problem* (GI) is to determine if there exists a graph isomorphism from G to G'.



The solution

Solution

Given two graphs G and G' solving GI requires proving the existence of a graph isomorphism from G to G' (perhaps by producing such a funciton) or showing no such function exists.



Complexity of GI

The complexity class of solving GI for any two finite simple graphs is an open problem (also known as the graph isomorphism disease).

- since it is straightforward to check if some function f is a graph isomorphism between two graphs or not, GI is in NP (the class of problems for which a possible solution can be checked nondeterministically in polynomial time)
- lowest established upper bound is
$$\mathcal{O}(2^{(\log n)^c})$$
where n is the number of vertices and $c \in \mathbb{N}$ (fixed), called *quasi-polynomial* time (Babai 2018)

Applied GI solvers

Althought the complexity of GI is an open problem, in reality, powerful (open source) software exists for classifying graphs efficently (i.e., they can efficiently proudce the automorphism group containing the set of all graph isomorphisms from a graph to itself).

- Weisfeiler-Leman (color refinement)
- nauty
- saucy
- bliss
- Traces

A bad method

The following solution method probably makes sense. Most efficient software for solving GI does *not* use this approach becasue it is not suitable for collections of graphs or finding graphs in a database (Mckay 2013)

Solution method

input: $G = \{V, E\}$, $G' = \{V', E'\}$
iterate: check functions $f : V \to V'$
output: true or false "G is isomorphic to G'"

The practical approach

Rather than check for isomorphism dirctly, most efficent GI solvers use an approach called "canonical labelling", along with another algorithm (e.g. hash function) where the vertex set is iteratively relabeled, producing partitions in such a way that isomorphic graphs are identical after relabelling (Mckay 2013).

Solution method

input: $G = \{V, E\}$, $G' = \{V', E'\}$
iterate: produce canonical labelling
output: true or false "G is isomorphic to G'"

frame name

Definition

text of thm

frame name

Definition

text of thm

this appears after thm