Graph Isomorphism

Franklin van Nes

Tuesday 12th November, 2017

1. Formally define and describe the topic (whether technique, algorithm, model or class) in detail, and discuss its significance.
   Graph Isomorphism is most intuitively understood through its greek etymology: *Iso*, meaning same, and *morphism*, meaning shape; *Graph Isomorhpism* $\rightarrow$ graphs that share the same shape. In more detail, two finite graphs are isomorphic if they share the same number of vertices connected in the same way.

   Formally, by an *isomorphism* we mean from two graphs G_1 to G_2 we have a one-to-one mapping $f : G_1.V \rightarrow G_2.V$ from $G_1.V$ onto $G_2.V$ so that vertices u_1 and v_1 are adjacent in G_1 if and only if the vertices $f(u_1)$ and $f(v_1)$ are adjacent in G_2. If an isomorphism exists between G_1 and G_2, we say they are *isomorphic* [2].

   Graph isomorphism is also an equivalence relation which allows us to say that G_1 and G_2 are isomorhpic if they are equal, and they are equivalent if they are isomorphic. Figure 1 shows how two isomorphic graphs can look very different while still maintaining the same vertex adjacency.

2. How do you show a problem is in the class? The class of problems within graph isomorphism is quite narrow. Determining graph isomorphism means given two graphs, determine if they are isomorphic. There also exists the *Subgraph Isomorphism* problems where given G_1 and G_2, determine if G_2 is a subgraph of G_1. However, Subgraph Isomorphism, which is definitively NP-complete [3], is a generatlization of Graph Isomorphism, so we won't consider this in the same problem set as Graph Isomorphism. Many of the graph isomorphism algorithms can be used to determine subgraph isomorphism. However, it should be noted that determining Graph Isomorphism is complex in its own right. In fact, determining graph isomorphism falls into its own category of problem complexity, called *Graph Isomorphism Complete*. According to Lubiw, it has yet to fall into a typical classification, and is neither P nor NP-complete [4]. There exists no known P algorithm, yet graph isomorphism has not be shown to be NP-complete. This means that either a P algorithm must exist for graph isomorphism but it undiscovered, graph isomorhpism is a problem outside of P and NP, or, as Schning argues, Graph isomorphism problems are in the low hierarchy of NP, which "does not

equal NP unless the polynomial hierarchy collapses to the second level" [6]. Deeper explanation of this exceeds the reaches of this paper, though Schning's research is well understood given the proper background knowlegdge.

3. Which are representative or classic problems in this class?
   While there is really just one representative problem of this class - determine if two graphs are isomorphic - there are a large variety of consequential applications to solving this problem. For instance, database searches that use graphs instead of keywords. A great example of this is the biochecmical search engine described by Bonnici et al. . Given an unknown "biological networks at the molecular, protein, or species level", in a graph representation, how can we determine if it exists? edges of a graph. Attach any metadata, like bond angle, bond type and maybe the element of the atom. With that information, we can perform a graph isomoprhism against the database of graphs existing molecules to determine if our unknown molecule has previously been discovered and what it is commonly known as. Because this is such an expensive process, it would make sense to filter results on other criteria before determining graph isomorphism, or to use graph isomorphism algorithms to eliminate matches as quickly as possible to reduce the search space [1].
   Another application for graph isomorphism is with automata, to determine if two languages (perhaps represented by Turing Machine schema), are the same.

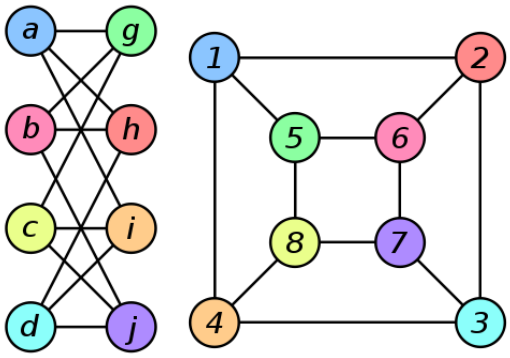
   https://math.stackexchange.com/questions/120408/what-are-the-applications-of-the-isomorphic-graphs

4. How does this class compare to other classes?

5. What techniques are used to solve problems in this class?

1 Figures

Figure 1: Two isomorphic graphs. The colors indicate matching vertices, even if their labels do not match [5]



References

[1] V. Bonnici, R. Giugno, A. Pulvirenti, D. Shasha, and A. Ferro. A subgraph isomorphism algorithm and its application to biochemical data. *BMC Bioinformatics*, 14(7):S13, Apr 2013.

[2] G. Chartrand. *Isomorphic Graphs*, pages 32–40. Dover, 1985.

[3] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158. ACM, 1971.

[4] A. Lubiw. Some np-complete problems similar to graph isomorphism. In *SIAM Journal on Computing*, pages 11–22. Society for Industrial and Applied Mathematics, February 1981.

[5] C. Martin.

[6] U. Schning. A low and a high hierarchy within np. *Journal of Computer and System Sciences*, 27(1):14 – 28, 1983.