Master’s Project Proposal
Graph Reconstruction Numbers

Jennifer Baldwin
jlb5851@cs.rit.edu

Rochester Institute of Technology
Department of Computer Science
102 Lomb Memorial Drive
Rochester, NY 14623-5608
Overview

The Reconstruction Conjecture was first proposed by S.M. Ulam and P.J. Kelly in 1941. The conjecture states that every graph with at least 3 vertices is reconstructible [3, 4, 1, 8]. In order to understand what reconstructible means, one must first understand what a deck and a card is. Given a graph $G$ a card represents the subgraph created by deleting a single vertex of $G$. A deck for graph $G$ is the multi-set containing all the vertex-deleted subgraphs of $G$ [3, 8, 4]. The deck is considered a multi-set because $G - v_i$ could be isomorphic to $G - v_j$ but they were created differently and therefore are viewed as unique. A graph $G$ is reconstructible if there exists some multi-subset of the deck that can be used to create a unique graph $H$ that is isomorphic to $G$. $H$ is considered unique if all the graphs that can be formed from the multi-subset have the same canonical labeling and are therefore isomorphic.

Since the conjecture has been put forth many special classes of the conjecture have been proven as well as useful corollaries and lemmas. One useful piece of information proven by Kelly is that the number of edges and the degree sequence of the original graph can be determined from the deck [3]. Using this information Kelly proved that trees, disconnected graphs, and regular graphs are reconstructible [3]. The proof that regular graphs, graphs for which the minimum and maximum degree is the same, are reconstructible is the easiest to understand. Since the degree sequence can be determined from the deck, it is known that $G$, the original graph, is a regular graph with $|\text{deck}| + 1$ vertices. As a result all reconstructions of $G$ will be $k$-regular, where $k = |\text{deck}| + 1$. All reconstructions using the deck will be isomorphic since each reconstruction can be obtained via the addition of a vertex with edges connecting to all vertices of degree $k - 1$. Thus $G$ is reconstructible [3, 4, 8]. The deck for trees is composed of a slightly different set of cards. The only vertices that are deleted from the tree are those that are the leaves of the tree [3, 8].

There has also been work conducted to determine the validity of the conjecture in a computational manner. Brendan McKay created an algorithm that is able to generate isomorph-free sets of graphs [5]. McKay used this algorithm to generate graphs with $n$ vertices and compared their isomorph-reduced decks. If any of the decks were the same, then the conjecture would prove to be false since the graphs reconstructible by that deck would not be unique modulo isomorphism. Overall, McKay was unable to find duplicate decks for the following classes of graphs: all graphs with at most 11 vertices, all graphs of order 12 with maximum degree 5, all triangle-free graphs with maximum order of 14, and all square-free graphs with maximum order 15 [5]. As a result these classes of graphs support the reconstruction conjecture.

One new area of research related to the reconstruction conjecture is determining the reconstruction number of a graph. By convention the deck of a graph $G$ is a multi-set composed of all vertex-deleted subgraphs of $G$. Each subgraph within the multi-set is called a card. The reconstruction number of a graph, denoted by $rn$, is the number of cards required to reconstruct the original graph up to isomorphism. The conjecture proposed by Harary and Plantholt is that the reconstruction number of any graph is at most $\frac{n}{2} + 1$ [7]. Bollobás proved in [2] that almost all graphs have a minimum reconstruction number of 3. Wendy Myrvold proved that disconnected graphs with at least two non-isomorphic components have a reconstruction number of 3. She determined a way to choose 3 cards based on the properties of disconnected graphs and their vertex-deleted subgraphs. Myrvold
also determined an upper bound for the reconstruction number of disconnected graphs with all isomorphic components. She determined that the bound was \( c + 2 \) where \( c \) represents the order of the component [7].

My project will focus on computationally determining minimum reconstruction numbers as well as the minimum sub-deck size such that all sub-decks of that size are able to reconstruct \( G \).

**Definitions:**

- \( r_{n_{\min}} \) is the minimum reconstruction number
- \( r_{n_{\max}} \) is the minimum sub-deck size such that all sub-decks of this size reconstruct \( G \)

In order to accomplish this goal I will be using programs developed by Brendan McKay as a part of the *nauty* package. The size of the graphs will at first be restricted to order eleven or less. Other specific types of graphs such as trees and disconnected graphs will be examined in hopes of finding a pattern for the reconstruction number. If patterns emerge they will be analyzed, conjectures will be formulated and attempted to be proven. Given a graph, the cards will be created and stored as a deck. Subsets of the deck will be tested from smallest to largest in size in order to determine \( r_{n_{\min}} \) and \( r_{n_{\max}} \).

**Functional Specification**

My approach to solving this problem will be split into three modules. The first module will be used primarily to generate test data. The second module will determine the minimum reconstruction number for a given graph. Lastly, the third module will determine the minimum sub-deck size such that all sub-decks of that size reconstruct the original graph.

To determine the minimum reconstruction number of a given graph first I need to determine if the sub-deck reconstructs the graph. It’s been proven that 2 cards will not reconstruct a graph so the minimum reconstruction number for the graph is at least 3 [3]. The general idea of this portion of the project is to loop through all sub-decks of size \( n \) and see if any reconstruct the original graph. Reconstructing the original graph is not enough to qualify the sub-deck as a reconstructing sub-deck. All possible reconstructed graphs created from the sub-deck must be unique modulo isomorphism. Once a sub-deck that satisfies these requirements is found the program will proceed to determine \( r_{n_{\max}} \).

Once the minimum reconstruction number is determined \( r_{n_{\max}} \) will be ascertained. Sub-decks of size smaller than the size of the minimum reconstruction set will not be considered when finding \( r_{n_{\max}} \) since it will be a waste of resources and time. Every possible sub-deck of size \( n \) will be checked to see if it reconstructs the original graph. If all sub-decks of size \( n \) reconstruct the original, then \( r_{n_{\max}} = n \). At some point both \( r_{n_{\min}} \) and \( r_{n_{\max}} \) will be known this will be handled accordingly.

Each module will not consist solely of a larger for-loop. Optimization algorithms and possible data structures will be used to further enhance to program.

My project will use the *gtools* package created by Brendan McKay [6] to facilitate the creation of graphs and the identification and removal of isomorphic graphs. The program *geng* will be used to generate graphs with 32 or fewer vertices [6]. Due to large number of

2
graphs that can be created, I will limit generation of graphs to 11 or fewer vertices. I will also use shorty to remove isomorphisms from the graphs generated by geng. As a result there will be no duplication of work for already solved problems. Gtools also provides a program, labelg, that canonically labels graphs. This will be used to help identify isomorphisms when reconstructing graphs based on a given sub-deck.

1 Foreseeable Problems

As a result of the large number of graphs possible with many vertices many data files will be generated. Handling all of these files may pose a challenge in the completion of this project. There needs to be some simplification of the data files in order to make using them easier.

Schedule

<table>
<thead>
<tr>
<th>Task</th>
<th>Expected Completion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal approved</td>
<td>3/12/04</td>
</tr>
<tr>
<td>Generate test graphs</td>
<td>3/24/04</td>
</tr>
<tr>
<td>Implement program</td>
<td>4/12/04</td>
</tr>
<tr>
<td>Unit test implementation</td>
<td>4/19/04</td>
</tr>
<tr>
<td>Run experiments</td>
<td>4/28/04</td>
</tr>
<tr>
<td>Analysis of experimental results</td>
<td>5/7/04</td>
</tr>
<tr>
<td>Complete writeup</td>
<td>5/21/04</td>
</tr>
<tr>
<td>Project defense</td>
<td>5/28/04</td>
</tr>
</tbody>
</table>

Deliverables

- Formal Project Writeup
  - Overview of the topic area
    * Background
    * Current research and solutions
    * Proposed research area
  - Overview of programs utilized by this project
    * $rr_{min}$ algorithm
    * $rr_{max}$ algorithm
    * nauty
    * geng
    * shortg
  - Description of implementation
  - Analysis of results

- Program Source Code
References


