Rochester Institute of Technology
Department of Computer Science

MS Project Proposal
Graph Reconstruction Numbers

Brian McMullen
bmm3056@rit.edu

May 18, 2005

Abstract

The project we propose is an extension of another project completed by Jennifier Baldwin at the Rochester Institute of Technology during the summer of 2004. In it the existential and universal graph reconstruction numbers were calculated for an exhaustive list of isomorphically unique graphs with more than two and less than nine vertices. One aim of this project is to calculate reconstruction numbers for significantly more graphs while verifying earlier results. We hope that, by analyzing reconstruction numbers for several graphs, trends can be identified that may provide some insight into the Reconstruction Conjecture. In addition, the scope of this project will be extended to find graphs that are 2-nonreconstructible when considering k-reconstructibility.

1 Overview

1.1 Background

In 1942, Kelly and Ulam proposed the Reconstruction Conjecture [9]. It states that every finite, undirected graph with three or more vertices is uniquely reconstructible up to isomorphism given a multiset of subgraphs where each subgraph is created by removing a single, unique vertex of the original graph. Given a graph G, this multiset of subgraphs is known as the *deck* of G and denoted as *D(G)*. Each graph in a deck is referred to as a *card*. The set of vertices of graph G is denoted as $V(G)$, and the deck of G contains $|V(G)|$ cards, where $|V(G)|$ is the number of vertices of G. So assuming the Reconstruction Conjecture is true, two graphs G and H have the same deck *iff* G and H are isomorphic.

Although unproven in the general case, the Reconstruction Conjecture has been proven to hold true for several classes of graphs. These include trees, regular graphs and disconnected graphs [4]. The proofs for the reconstructibility of these classes of graphs are direct results of Kelly's Lemma [4], which tells us that the number of edges and the degree sequences of a graph are reconstructible from a deck.

Kelly proposed that the Reconstruction Conjeture can be generalized by considering the reconstructibility of graphs with some number k vertex-deleted subgraphs instead of just one [8]. In this project we will also look for graphs that are 2-nonreconstructible. Because of findings by Kelly and Nýdl [14] we know that there are graphs which are not reconstructible by their 2-vertex-deleted subgraphs, but it is not clear which graphs these are. By using computation to identify these graphs, we hope to shed some light on what type of graphs tend to be 2-nonreconstructible.

An issue related to the Reconstruction Conjecture is that of *reconstruction numbers*. While the Reconstruction Conjecture is concerned with the possibility of reconstructing graphs from a deck, reconstruction numbers can be considered a measure of how easily a graph can be reconstructed from a deck. In most cases, a given graph G is reconstructible from a small subset of cards from *D(G)*. In fact, Bollobás proved probabilistically that almost all graphs can be reconstructed with only three cards of a deck [3].

This project is concerned with two types of reconstruction numbers – the *existential reconstruction number* and the *universal reconstruction number*. The existential reconstruction number of G, $\exists rn(G)$, is the minimum number of vertex-deleted subgraphs of G required to uniquely reconstruct G up to isomophism. The universal reconstruction number, $\forall rn(G)$, is the minimum number n for which all subsets of $D(G)$ of size n uniquely reconstruct G up to isomorphism.

In the project presented by Baldwin, both existential and universal reconstruction numbers were computed for every graph G where $3 \leq |V(G)| \leq 8$. When calculating the reconstruction numbers for graphs with nine vertices, however, the time to calculate reconstruction numbers would take anywhere from a second to a minute to complete [2]. When analyzing the earlier algorithms used, we can see there are many areas in which the computations can be made to be much more efficient. A major aim of this project is to use an improved algorithm to compute the reconstruction numbers for all graphs with at least ten vertices and, ideally, those with eleven as well.

1.2 Previous Algorithm

In preparation for computing reconstrction numbers, several files were created. Some of these files held the decks of the graphs for which they were calculating reconstruction numbers. The remaining files were used for all singly vertex-extended graphs, or *extension graphs*, of relevant graphs. The set of extension graphs of a given graph G contains all isomporphically unique graphs that can each create G by deleting a single one of their vertices.

With these graphs generated, first $\exists rn(G)$ then $\forall rn(G)$ were calculated. The algorithms are shown in diagrams **Algorithm 1** and **Algorithm 2**.

Algorithm 1 Computing $\exists rn(G)$

```
1: Input graph $G$
2: Read $D(G)$ from vertex-deletion files
3: Read all 1-vertex extensions of each card of $D(G)$ from vertex-extension files
4: Read the deck of each extension graph from vertex-deletion files
5: for $subdeck\_size = 2$ to $n - 1$ do
6:   for each unique subdeck of $G$ of size $subdeck\_size$ do
7:     $found$ = false
8:     for each extension graph read do
9:       if extension deck contains current subdeck of $G$ then
10:        $found$ = true
11:        break out of innermost for loop
12:      end if
13:    end for
14:    if $found$ = false then
15:      break out of outermost for loop
16:    end if
17:  end for
18: end for
19: The value of $subdeck\_size$ is $\exists rn(G)$
```

The idea behind these algorithms is to take the graph G in question and consider all graphs that share at least one card in their decks in common with the cards of $D(G)$. These graphs are found by looking up the extension graphs of each card in $D(G)$. We then look at increasing values for variable *subdeck_size*, checking if each possible subdeck of G of this size is unique in comparison to the decks of the relevant extension graphs. Once we find such a subdeck unique to G, we know the current value of *subdeck_size* is $\exists rn(G)$. Using the value for $\exists rn(G)$ as a starting point, a similar method is used for calculating $\forall rn(G)$.

Algorithm 2 Computing $\forall rn(G)$

```
1:  for $subdeck\_size = \exists rn(G)$ to $n - 1$ do
2:      $found$ = false
3:      for each unique subdeck of $G$ of size $subdeck\_size$ do
4:          for each extension graph from $\exists rn(G)$ algorithm do
5:              if extension deck contains current subdeck of $G$ then
6:                  $found$ = true
7:                  break for on line 3
8:              end if
9:          end for
10:     end for
11:     if $found$ = false then
12:         break for
13:     end if
14: end for
15: The value of $subdeck\_size$ is $\forall rn(G)$
```

In both algorithms, matches between the cards of $D(G)$ and the decks of G's extension graphs are made by first setting up a *relation matrix*. So the extension graphs read in step 3 of **Algorithm 1** are the graphs we are checking against to find reconstruction numbers since they each share at least one card in their decks in common with $D(G)$. Each row of the relation matrix serves to show which cards the deck of an extension graph have in common with $D(G)$. For each card in $D(G)$, there is a column in the relation matrix.

So before the first **for** loop of **Algorithm 1**, the values of each row in the relation matrix are assigned a one where the deck of the extension graph in question has a card equivalent to the card of the intersecting column and a zero otherwise. (Given that the decks involved are multisets, it is important to note that a card of an extension graph deck must be used once and only once as a match to a card of $D(G)$). The relation matrix is used to see if a subdeck of $D(G)$ for graph G is also a subdeck of at least one of the extension graph decks by reading down the proper columns for a given subdeck and checking the values for each extension graph row. This same relation matrix is used for calculating both existential and universal reconstruction numbers. Obviously, $\exists rn(G)$ is calculated first since its value is used as the first value to check as being $\forall rn(G)$.

1.3 Previous Results

Baldwin [2] cited two interesting results from her reconstruction number calculations. One of these was that the dominant value for $\forall rn(G)$ seemed to be four [2]. Although the number of graphs with a universal reconstruction number of four seemed to be large, it was not clear whether this value would be true for a majority of graphs as proven by Bollobás [3] for $\exists rn(G)$ or whether it was apparent the trend would continue for graphs with more vertices.

The second result noted was that, of the analyzed graphs having an odd number of vertices, none had an existential reconstruction number greater than three [4]. However, we know from Myrvold [12] that for every disconnected graph G made up of p copies of K_c (where K_c is a complete graph with c vertices), $\exists rn(G) = c + 2$. And since a disconnected graph G with nine vertices made up of three copies of K_3 is possible (resulting in $\exists rn(G) = 5$) we know this will not hold for all graphs with nine vertices. However, this does not rule out the possibility that all graphs with a *prime* number of vertices do not have existential reconstruction numbers greather than three, since splitting them up into disconnected graphs with an equal number of vertices of two or more is not possible. This is a compelling reason to find reconstruction numbers for all graphs with eleven vertices.

2 Functional Specification

2.1 Software Tools

There were two tools used in the old project to assist in the calculation of reconstruction numbers. One of these, the Condor package [5], which is installed on several machines in labs at RIT, distributes the computation workload among several machines according to which machines have the most available processing time.

The second tool used was the **nauty** software package written by Brendan McKay [11]. As the most efficient package available for identifying isomorphisms between graphs, it is indispensible for the completion of this project. **Nauty** also offers funcionality for isomorph-free exhaustive generation of graphs, a compact representation of simple graphs, canonization of graphs according to isomorphic equivalence as well as other basic graph manipulation functions. Both Condor and **nauty** will be used in this project as well.

2.2 Proposed Changes to Original Algorithm

The original algorithms serve as helpful guidelines for calculating reconstruction numbers. However, there are several changes that can be made to calculate the numbers much more efficiently. These are listed below...

1. Do not use files to provide extension graphs and decks.
2. Do not use brute force method for calculating $\forall rn(G)$.
3. Cut back on the number of graphs for which we need to calculate reconstruction numbers.
4. Use bitmaps to represent rows in the relation matrix.
5. Try to cut back on the number of extension graphs we need to examine when finding reconstruction numbers.

6. Apply what we know about disconnected graphs to quickly determine their existential reconstruction number values.

The first item is fairly self-explanatory. Although probably intended to reduce processing time, creating files to store decks and extension graphs beforehand leads to a far worse efficiency issue when handling file I/O. A large number of graphs to be stored on disk results in large files to drive through to find the relevant data. And this issue grows exponentially more problematic with the size of graphs being analyzed. It is more efficent to simply calculate the necessary graphs as they are needed.

Item 2 would involve grossly simplifying the calculation of $\forall rn(G)$ for a given graph G. Once the relation matrix is created, it is not necessary to check each possible value for subdeck sizes incrementally. All that is required is to find the maximum number of matches between the cards of $D(G)$ and the extension graphs' decks by looking at the rows of the relation matrix. Adding one to this maximal value yields the universal reconstruction number, since this is the smallest subdeck size, s, where all possible subdecks of $D(G)$ of size s are not present in the extension graph decks considered. This is especially beneficial considering that for most every graph G, $\exists rn(G) = 3$, so not much calculation will be needed for each graph after the relation matrix is created.

It can be easily shown that both $\exists rn(G) = \exists rn(\bar{G})$ and $\forall rn(G) = \forall rn(\bar{G})$ [6], where G and $\bar{G}$ are complementary graphs. Item 3 was created to take advantage of that fact. By calculating reconstruction numbers for a graph and simply assigning these values to its complement, the numbers can be calculated in roughly half the time as they would be otherwise.

In the original algorithm, the relation matrix was created as an array of arrays, with each entry having a value of zero or one. In item 4, the plan is to instead represent each row of the matrix as a bitmap with each bit being the approporiate boolean value. Not only does this save space, but processing time as well. Instead of testing the value of each column one at a time, a bitmask can be created to represent the subset of $D(G)$, for graph G, being tested and a **bitwise and** can test all the values of the relevant row at the same time.

Item 5 is fairly open-ended. Given that we know that for any graph G, $\exists rn(G) \geq 3$ where $|V(G)| \geq 3$ [2], an obvious improvement would be to exclude any extension graph with less than three cards in its deck matching the cards of $D(G)$. However, more consideration can be made on what graphs need to be examined when calculating reconstruction numbers for a given graph.

The sixth item is based on findings made by Myrvold [12]. We know that for every graph G, where G is a disconnected graph consisting of nonisomporphic components, $\exists rn(G) = 3$ [12]. As stated earlier, we can also quickly determine $\exists rn(G)$ where G is a disconnected graph made up of copies of some K_c. However, more research has to be done to see how easily these properties can be determined given a description of a graph and the tools to be used in this project. If identifying certain classes of disconnected graphs cannot be made efficiently, they may have to be calculated in the same manner as other graphs.

2.3 2-Vertex Reconstruction

At this point, the algorithm to be used for finding graphs that are 2-nonreconstructible is not as clear as it is when calculating reconstruction numbers. As of now, a similar method would be used to generate graphs to test against G to have the same 2-vertex deleted decks as was used to generate graphs to test against for finding reconstruction numbers. The difference would be that $D_2(G)$ would be found as all possible two vertex deletions of G and graphs to test for breaking reconstructibility would be all possible 2-vertex extensions of these graphs. Since this may generate a sizeable number of graphs to test against, especially with graphs with a larger number of vertices, some other method may need to be identified to find a smaller set of graphs to examine.

Since we are only worried about whether or not each graph is 2-reconstructible and do not have to find a reconstruction number, the graphs in question do not have to be stored in a relation matrix; they can either be identified or rejected as exceptions to 2-reconstructibility one by one.

2.4 Results to Look For

As stated earlier, it will be interesting to see if there exists any graph G where $|V(G)| = 11$ and $\exists rn(G) > 3$. If so, we can rule out that all graphs with a prime number of vertices must have an existential reconstruction number of three.

Related to this is the question of just how high the value for $\exists rn(G)$ can be. Because Myrvold [12] established that for every disconnected graph G composed of p copies of K_c $\exists rn(G) = c + 2$, we know that if $|V(G)|$ is divisible by d (where $d \neq |V(G)|$), then $\exists rn(G)$ can be as high as $d + 2$. However if d is the largest such divider of G, we do not know whether or not $\exists rn(G)$ can be greater than $d + 2$. There was no such example from Baldwin's results, and it would be interesting to see if such a case can be found in this project.

Finally the data will be anlyzed for trends in graphs with maximal values for $\exists rn(G)$, $\forall rn(G)$ and any graphs found that are 2-nonreconstructible.

3 Deliverables

- Project Writeup
  - Overview of topic
  - Description of algorithms
  - Data results/analyisis
    - Statistics on reconstruction number counts
    - List of graphs with notable reconstruction numbers
    - List of graphs not reconstructible from 2-vertex deleted decks
    - Analysis of notable graphs
- Program source code

4 Schedule

Task	**Projected Completion**
Proposal approved	11/29/04 - 12/3/04
Code interfaces to **nauty**	12/6/04
Code reconstruction number calculator	1/3/05
Code 2-nonreconstructible calculator	1/10/05
Testing/debugging	1/17/05
Run calculations	1/31/05
Analyze results	2/7/05
Compose final write-up	2/21/05
Project defense	2/24/05

References

[1] K. J. Asciak and J. Lauri, On Disconnected Graph with Large Reconstruction Number. *Ars Combinatoria*, 62:173-181, 2002.

[2] J. Baldwin, *Graph Reconstruction Numbers*, Master's Project, Rochester Institute of Technology, 2004.

[3] B. Bollobás. Almost every graph has reconstruction number three. *Journal of Graph Theory*, 14(1):1-4, 1990.

[4] J. A. Bondy. A Graph Reconstructor's Manual. *Surveys in Combinatoria*, 1991.

[5] Condor Team, University of Wisconsin-Madison, *Condor Version 6.6.7 Manual*, May 2004.

[6] F. Harary and M. Plantholt. The Graph Reconstruction Number. *Jounal of Graph Theory*, 9:451-454, 1985.

[7] E. Hemaspandra, L. Hemaspandra, S. Radziszowski and R.Tripathi. Complexity Results in Graph Reconstruction. *Proceedings of the 29th International Symposium on Mathematical Foundations of Computer Science*. To appear.

[8] P. J. Kelly. A congruence theorem for trees. *Pacific Journal of Mathematics*, 7:961-968, 1957.

[9] P. J. Kelly. *On Isometric Transformations*. PhD thesis, University of Wisconsin, 1942.

[10] B. D. McKay. Isomorph-Free Exhaustive Generation. *Journal of Algorithms*, 26:306-324, 1998.

[11] B. D. McKay. nauty *User's Guide (Version 2.2)*. Computer Science Department, Australian National Universidty, bdm@cs.anu.edu.au.

[12] W. J. Myrvold, The ally reconstruction number of a disconnected graph. *Ars Combinatoria*. 28:123-127, 1989.

[13] V. Nýdl. Finite undirected graphs which are not reconstructible from their large cardinality subgraphs. *Topological, Algebraic and Combinatorial Structures*, 1991.

[14] V. Nýdl. Graph reconstruction from subgraphs. *Discrete Mathematics* 235:335-341, 2001.