8-19-2011

Isomorph-free generation of 2-connected graphs with applications

Derrick Stolee
University of Nebraska-Lincoln, s-dstolee1@math.unl.edu

Isomorph-free generation of 2-connected graphs with applications

Derrick Stolee*
Department of Mathematics
Department of Computer Science
University of Nebraska–Lincoln
s-dstolee1@math.unl.edu

August 19, 2011

Abstract

Many interesting graph families contain only 2-connected graphs, which have ear decompositions. We develop a technique to generate families of unlabeled 2-connected graphs using ear augmentations and apply this technique to two problems. In the first application, we search for uniquely K_r-saturated graphs and find the list of uniquely K_4-saturated graphs on at most 12 vertices, supporting current conjectures for this problem. In the second application, we verify the Edge Reconstruction Conjecture for all 2-connected graphs on at most 12 vertices. This technique can be easily extended to more problems concerning 2-connected graphs.

1 Introduction

If a connected graph G has a vertex x so that $G - x$ is disconnected or a single vertex, then G is *separable*. Otherwise, G is 2-*connected*, and there is no single vertex whose removal disconnects the graph. Many interesting graph families contain only 2-connected graphs, so we devise a generation technique that exploits the structure of 2-connected graphs.

A fundamental and well known property of 2-connected graphs is that they have an *ear decomposition*. An *ear* is a path $x_0, x_1, \ldots, x_k$ so that x_0 and x_k have degree at least three and x_i has degree exactly two for all $i \in \{1, \ldots, k-1\}$. An *ear augmentation* on a graph G is the addition of a path with at least one edge between two vertices of G. The augmentation process is also invertible: an *ear deletion* takes an ear $x_0, x_1, \ldots, x_k$ in a graph and deletes all vertices $x_1, \ldots, x_{k-1}$ (or the edge x_0x_1 if $k = 1$). Every 2-connected graph G has a sequence of subgraphs $G_1 \subset \cdots \subset G_\ell = G$ so that G_1 is a cycle and for all $i \in \{1, \ldots, \ell - 1\}$, G_{i+1} is the result of an ear augmentation of G_i [35].

In Section 2, we describe a method for generating 2-connected graphs using ear augmentations. While we wish to generate unlabeled graphs, any computer implementation must store an explicit labeling of the graph. Without explicitly controlling the number of times an isomorphism class appears, a singe unlabeled graph may appear up to $n!$ times. An *isomorph-free* generation scheme for a class of combinatorial objects visits each isomorphism class exactly once. To achieve this goal, our strategy will make explicit use of isomorphisms, automorphisms, and orbits. The technique used in this work is an implementation of McKay's isomorph-free generation technique [19], which is sometimes called "canonical augmentation" or "canonical deletion". See [14] for a discussion of

*The author is supported in part by the National Science Foundation grants CCF-0916525 and DMS-0914815.

similar techniques. We implement this technique to generate only 2-connected graphs using ear augmentations.

Almost all graphs are 2-connected [32], even for graphs with a small number of vertices[1], so as a method of generating all 2-connected graphs, this method cannot significantly reduce computation compared to generating all graphs and ignoring the separable graphs. The strength of the method lies in its application to search over ear-monotone properties and to use the structure of the search to reduce computation. These strengths are emphasized in two applications presented in this work.

In Section 3, we search for graphs that are uniquely K_r-saturated. These graphs contain no K_r and adding any edge from the complement creates a unique copy of K_r. This pair of constraints reduces the number of graphs that are visited while searching for uniquely K_r-saturated graphs. The graphs found with this method support the current conjectures on these graphs.

In Section 4, we verify the Edge Reconstruction Conjecture on small 2-connected graphs. The structure of the search allows for a reduced number of pairwise comparisons between edge decks. Also, it is known that the Reconstruction Conjecture holds if all 2-connected graphs are reconstructible. Since graphs with more than $1 + \log(n!)$ edges are edge-reconstructible, we focus only on 2-connected graphs with at most this number of edges, providing a sparse set of graphs to examine. This verifies the conjecture on all 2-connected graphs up to 12 vertices, extending previous results [18].

1.1 Notation

In this work, H and G are graphs, all of which will are simple: there are no loops or multi-edges. For a graph G, $V(G)$ is the vertex set and $E(G)$ is the edge set. The number of vertices is denoted $n(G)$, while $e(G)$ is the number of edges. The complement graph $\overline{G}$ is the graph on vertices $V(G)$ with a vertex pair xy in $E(\overline{G})$ if and only if $xy \notin E(G)$.

For a 2-connected graph, a vertex of degree at least three is a *branch vertex*. Vertices of degree two are *internal* vertices, as they are contained between the endpoints of an ear. Ears will be denoted with ε. For an ear ε, the *length* of ε is the number of edges between the endpoints and its *order* is the number of internal vertices between the endpoints. We will focus on the order of an ear. An ear of order 0 (length 1) is a single edge, called a *trivial* ear. Ears of larger order are *non-trivial*.

Given a graph G and an ear $\varepsilon = x_0, x_1, \ldots, x_k$, the *ear deletion* $G - \varepsilon$ is the graph $G - x_1 - x_2 - \cdots - x_{k-1}$, where all internal vertices of ε are removed. For an ear $\varepsilon = x_0, x_1, \ldots, x_{k-1}, x_k$ where $x_0, x_k \in V(G)$ but $x_1, x_2, \ldots, x_{k-1}$ are not vertices in G, the *ear augmentation* $G + \varepsilon$ is given by adding the internal vertices of ε to G and adding the edges $x_i x_{i+1}$ for $i \in \{0, \ldots, k-1\}$.

2 Isomorph-Free Generation via Ear Decompositions

In this section, we describe a general method for performing isomorph-free generation in specific families of 2-connected graphs.

2.1 The search space and ear augmentation

Consider a family $\mathcal{F}$ of unlabeled 2-connected graphs. We say $\mathcal{F}$ is *deletion-closed* if every graph G in $\mathcal{F}$ which is not a cycle has an ear ε so that the ear deletion $G - \varepsilon$ is also in $\mathcal{F}$. For an integer $N \geq 3$, $\mathcal{F}_N$ is the set of graphs in $\mathcal{F}$ with at most N vertices.

[1] To see the overwhelming majority of 2-connected graphs, compare the number of unlabeled graphs [26] to the number of unlabeled 2-connected graphs [27].

(a) A 2-connected graph G and an ear ε.

(b) $G - \varepsilon$, separable.

Figure 1: A 2-connected graph G and an ear ε whose removal makes $G - \varepsilon$ separable.

This requirement implies that for every graph $G \in \mathcal{F}$, there exists a sequence $G \supset G_1 \supset G_2 \cdots$ of ear deletions $G_{i+1} = G_i - \varepsilon_i$ where each graph G_i is in $\mathcal{F}$ and the sequence $\{G_i\}$ terminates at some cycle $C_k \in \mathcal{F}$. By selecting an ear deletion which is invariant to the representation of each G_i, we define a canonical sequence of ear-deletions that terminates at such a cycle. While generating graphs of $\mathcal{F}$, we shall only follow augmentations that correspond to these canonical deletions, giving a single sequence of augmentations for each isomorphism class in $\mathcal{F}$. This allows us to visit each isomorphism class in $\mathcal{F}$ exactly once using a backtracking search and without storing a list of previously visited graphs.

The search structure is that of a rooted tree: the root node is an empty graph, with the first level of the tree given by each cycle C_k in $\mathcal{F}_N$. Each subsequent search node is extended upwards by all canonical ear augmentations. Since the search does not require a list of previously visited graphs, disjoint subtrees are independent and can be run concurrently without communication. This leads to a search method which can be massively parallelized without a significant increase in overhead.

Note that being deletion-closed does not imply that *every* ear ε in G has $G - \varepsilon$ in the family. In fact, this does not even hold for the family of 2-connected graphs, as removing some ears leave the graph separable. See Figure 1 for an example of such an ear deletion.

Also, if $\mathcal{F}$ is deletion-closed, then so is $\mathcal{F}_N$. While the algorithms described could operate over $\mathcal{F}$, a specific implementation will have a bounded number (N) of vertices to consider. Operating over $\mathcal{F}_N$ allows for a finite number of possible ear augmentations at each step.

To augment a given labeled graph G, enumerate all pairs of vertices $x, y \in V(G)$ and orders $r \geq 0$ so that $|V(G)| + r \leq N$ and attempt adding an ear between x and y of order r. If an edge exists between x and y, then adding an ear of order 0 will immediately fail. However, all other orders produce valid 2-connected graphs. We then test if the augmentation $G + \varepsilon$ is in $\mathcal{F}$, discarding graphs which are not in the family.

2.2 Augmenting by orbits

By considering the automorphisms of a given graph, we can reduce the number of attempted ear augmentations. First, note that between a given pair of vertices, multiple ears of the same order are in orbit with each other. Second, if ε_1 is an ear between x_1 and y_1 and ε_2 is an ear between x_2 and y_2, then ε_1 and ε_2 are in orbit if and only if they have the same order and the vertex sets $\{x_1, y_1\}$, $\{x_2, y_2\}$ are in orbit under the automorphism group of G. Third, if the sets of vertices $\{x_1, y_1\}$ and $\{x_2, y_2\}$ are in orbit under the automorphism group of G, then the augmentations formed by adding an ear of order r between x_1 and y_1 is isomorphic to adding an ear of order r between x_2 and y_2.

Algorithm 1 Delete$_{\mathcal{F}}G$) — The Default Canonical Deletion in $\mathcal{F}$

```
minOrder ← n(G)
minLabel ← n(G)^2
bestEar ← null
for all vertices x ∈ V(G) with deg x ≥ 3 do
  for all ears e incident to x do
    Let y be the opposite endpoint of e
    label ← min{n(G)π_G(x) + π_G(y), n(G)π_G(y) + π_G(x)}
    r ← order of e
    if G − e ∈ F_N then
      if r < minOrder then
        minOrder ← r
        minLabel ← label
        bestEar ← (x, y, r)
      else if r = minOrder and label < minLabel then
        minLabel ← label
        bestEar ← (x, y, r)
      end if
    end if
  end for
end for
return bestEar
```

This redundancy under graphs with non-trivial automorphism group is removed by computing the orbits of vertex pairs, then only augmenting ears between a single representative of a pair orbit. Pair orbits are computed by applying the generators of the automorphism group of G to the set of vertex pairs.

2.3 Canonical deletion of ears

While augmenting by orbits reduces the number of generated graphs, a *canonical deletion* is defined to guarantee that each unlabeled graph in $\mathcal{F}_N$ is enumerated exactly once. This selects a unique ear $\varepsilon = \text{Delete}_{\mathcal{F}}(G)$ so that $G - \varepsilon$ is in $\mathcal{F}$ and ε is invariant to the labeling of G. That is, if G_1 and G_2 are isomorphic graphs with deletions $\text{Delete}_{\mathcal{F}}(G_1) = \varepsilon_1$ and $\text{Delete}_{\mathcal{F}}(G_2) = \varepsilon_2$, then there is an isomorphism π from G_1 to G_2 so that π maps ε_1 to ε_2.

In order to compute a representative $\text{Delete}_{\mathcal{F}}(G)$ that is invariant to the labels of G, a canonical labeling of $V(G)$ is computed. A *canonical labeling* is a map $\text{lab}(G)$ which maps graphs G to permutations $\pi_G : V(G) \to \{0, 1, 2, \ldots, |V(G)| - 1\}$ so that for every labeled graph $G' \cong G$, the map $\phi : V(G) \to V(G')$ given by $\phi(v) = \pi_{G'}^{-1}(\pi_G(v))$ for each $v \in V(G)$ is an isomorphism from G to G'. In this sense, the map π_G is invariant to the labels of $V(G)$. McKay's `nauty` library [20, 11] is used to compute this canonical labeling.

Once the canonical labeling is computed, the canonical deletion can be chosen by considering all ears ε whose deletion $(G - \varepsilon)$ remains in $\mathcal{F}_N$, and selecting the ear with (a) minimum length, and (b) lexicographically-least canonical label of branch vertices. Algorithm 1 details this selection procedure.

Algorithm 2 Search$_{\mathcal{F}}(G, N)$ — Search all canonical augmentations of G in $\mathcal{F}_N$

```
if Prune_F(G) = true then
  return
end if
if G is a solution then
  Store G
end if
R ← N − n(G)
for all vertex-pair orbits O do
  {x, y} ← representative pair of O
  for all orders r ∈ {0, 1, ..., R} do
    G' ← G + Ear(x, y, r)
    (x', y', r') ← Delete_F(G')
    if r = r' and {x', y'} ∈ O then
      Search_F(G', N)
    end if
  end for
end for
return
```

2.4 Full implementation

This isomorph-free generation scheme is formalized by the recursive algorithm Search$_{\mathcal{F}}(G, N)$, given in Algorithm 2. The full algorithm Search$_{\mathcal{F}}(N)$ searches over all graphs of order at most N in $\mathcal{F}$ and is initialized by calling Search$_{\mathcal{F}}(C_k, N)$ for each $k \in \{3, 4, \ldots, N\}$. Since the recursive calls to Search$_{\mathcal{F}}(G, N)$ are independent, they can be run concurrently without communication.

For some applications, it is possible to determine that no solutions are reachable under any sequence of ear augmentations. In such a case, the algorithm can stop searching at the current node to avoid computing all augmentations and canonical deletions. Let Prune$_{\mathcal{F}}(G)$ be the subroutine which detects if such a pruning is possible.

The framework for Algorithm 2 was implemented in the TreeSearch library[2] [29], a C++ library for managing a distributed search using the Condor scheduler [30]. This implementation was executed on the Open Science Grid [23] using the University of Nebraska Campus Grid [33]. Performance calculations in this paper are based on the accumulated CPU time over this heterogeneous set of computation servers. For example, the nodes available on the University of Nebraska Campus Grid consist of Xeon and Opteron processors with a speed range of 2.0-2.8 GHz. All code and documentation written for this paper are available in a GitHub repository[3].

2.5 Generating all 2-connected graphs

Using the isomorph-free generation scheme of canonical ear deletions, we can generate all unlabeled 2-connected graphs on N vertices or graphs on N vertices with exactly E edges.

Definition. Let N and E be integers. Set g_N to be the number of unlabeled 2-connected graphs on N vertices and $g_{N,E}$ to be the number of unlabeled 2-connected graphs on N vertices and E edges.

[2]The TreeSearch library is available at https://github.com/derrickstolee/TreeSearch
[3]The EarSearch library is available at https://github.com/derrickstolee/EarSearch

N	g_N	CPU time
5	10	0.01s
6	56	0.11s
7	468	0.26s
8	7123	10.15s
9	194066	5m 17.27s
10	9743542	7h 39m 28.47s
11	900969091	71d 22h 22m 49.12s

Table 1: Comparing g_N and the time to generate $\mathcal{G}_N$.

$\mathcal{G}_N$ is the family of 2-connected graphs on up to N vertices. $\mathcal{G}_{N,E}$ is the family of 2-connected graphs on up to N vertices and up to E edges.

Robinson [25] computed the values of g_N and $g_{N,E}$, listed in [27, 24]. Note that $\mathcal{G}_N$ and $\mathcal{G}_{N,E}$ are deletion-closed families, and can be searched using isomorph-free generation via ear augmentations. We revisit the three main behaviors of the algorithm: canonical deletion, pruning, and determining solutions.

Canonical Deletion: The canonical deletion algorithm in Algorithm 1 suffices for the class of 2-connected graphs. Recall this algorithm selects from ears ε so that $G - \varepsilon$ remains 2-connected, selecting one of minimum length and breaking ties by using the canonical labels of the endpoints.

Pruning: If the number of edges is fixed to be E, a graph with more than E edges should be pruned. Also, a graph on $n(G) < N$ vertices must add at least $N - n(G) + 1$ edges during ear augmentations in order to achieve N total vertices. If $e(G) + (N - n(G) + 1) > E$, then no graph on N vertices with at most E edges can be reached by ear augmentations from G. In this case, the node can be pruned.

Solutions: A 2-connected graph is a solution if and only if $n(G) = N$, and if E is specified then $e(G) = E$ must also hold.

Table 1 compares the number of 2-connected graphs of order N and the CPU time to enumerate all such graphs. Both the computation times and the sizes of the sets grow exponentially. Since the number of 2-connected graphs on N vertices grows so quickly, to test the performance for larger orders, the number of edges was also fixed to be slightly more than N. Table 2 shows these computation times.

3 Application 1: Uniquely H-Saturated Graphs

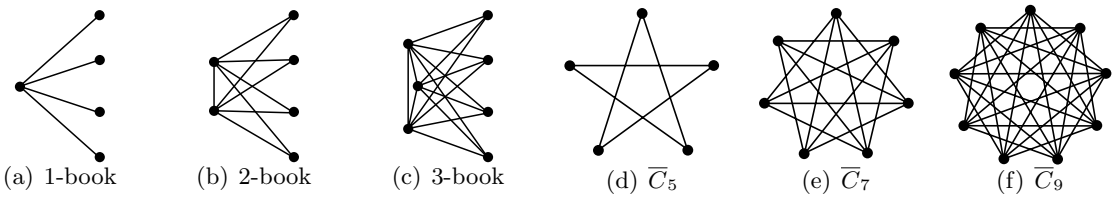
Our first application forbids certain subgraphs, which decreases the number of graphs to enumerate. We investigate *uniquely H-saturated* graphs.

Definition. Let H and G be graphs. G is *H-saturated* if G contains no copy of H and for every edge $e \in E(\overline{G})$ there is at least one copy of H in $G + e$. G is *uniquely H-saturated* if G contains no copy of H and for every edge $e \in E(\overline{G})$, there is a unique copy of H in $G + e$.

While it is easy to see that H-saturated graphs always exist, being uniquely H-saturated is a very strict condition. In fact, not all H admit *any* graph which is uniquely H-saturated. For $k \in \{6, 7, 8\}$, no uniquely C_k-saturated graphs exist [34]. For other graphs H, there is a very limited list of uniquely H-saturated graphs. If G is uniquely C_3-saturated, then G is either a star ($K_{1,n}$) or a Moore graph of diameter two and girth five: G has no triangles and every pair of non-adjacent vertices have exactly one common neighbor. There are at least three Moore graphs:

N	$E=11$	$E=12$	$E=13$	$E=14$	$E=15$	$E=16$	$E=17$	$E=18$	$E=19$	$E=20$
10	9	121	1034	5898	23370	69169	162593	317364	530308	774876
	0.01	0.16	1.73	12.99	65.88	167.12	472.68	972.62	2048.85	3631.71
11		11	189	2242	17491	94484	380528	1212002	3194294	7197026
		0.02	0.38	5.52	56.10	260.53	1212.89	4069.09	13104.24	32836.53
12			13	292	4544	46604	334005	1747793	7274750	24972998
			0.03	0.86	17.56	286.00	1226.71	6930.00	33066.80	125716.68
13				15	428	8618	113597	1031961	6945703	36734003
				0.05	1.83	44.64	469.02	5174.92	39018.15	227436.84
14					18	616	15588	257656	2925098	24532478
					0.08	3.82	90.51	1573.81	21402.18	183482.70
15						20	855	26967	519306	7654299
						0.12	7.56	198.84	4567.43	76728.79
16							23	1176	44992	1111684
							0.18	15.56	498.20	13176.05

Table 2: Comparing $g_{N,E}$ (above) and the time to generate $\mathcal{G}_{N,E}$ (below, in seconds).



(a) 1-book (b) 2-book (c) 3-book (d) $\overline{C}_5$ (e) $\overline{C}_7$ (f) $\overline{C}_9$

Figure 2: The $(r-2)$-books and complemented $(2r-1)$-cycles are uniquely K_r saturated.

C_5, the Petersen graph, the Hoffman-Singleton graph, and possibly some 57-regular graphs on 3250 vertices [13]. There are exactly ten uniquely C_4-saturated graphs [3]. If G is uniquely C_5-saturated, then G is either a friendship graph (every pair of vertices have exactly one common neighbor) or one of a finite number of other examples [34]. The only friendship graphs are the *windmills*: $\frac{n-1}{2}$ triangles sharing a common vertex [7].

3.1 Uniquely K_r-saturated graphs

Historically, the first host graph H where the extremal problems on H-saturated graphs were solved was the complete graph K_r [31, 6]. However, uniquely K_r-saturated graphs have evaded attempts at classification. Only empty graphs are uniquely K_2-saturated, and uniquely K_3-saturated graphs are stars and Moore graphs (since $K_3 \cong C_3$). There are two known infinite families of uniquely K_r-saturated graphs: books and cycle complements.

The t-book on n vertices is a complete graph K_t (the *spine*) joined with an independent set on $n-t$ vertices (the *pages*). The $(r-2)$-book has cliques of size at most $r-1$ and all non-edges are in the independent set. Adding any edge in the independent set forms exactly one K_r by using the two endpoints and the $r-2$ vertices in the spine. Figures 2(a), 2(b), and 2(c) are examples of $(r-2)$-books for $r \in \{3,4,5\}$. For $r=3$, note that the $(r-2)$-book with n pages is isomorphic to the star $K_{1,n}$ with n leaves.

The complement of the $(2r-1)$-cycle is also uniquely K_r-saturated. All pairs of vertices in a clique of $\overline{C}_{2r-1}$ are at distance at least two in the original cycle. Such a set must have size at most $r-1$. However, adding an edge from the cycle to its complement creates a unique copy of

K_r. Figures 2(d), 2(e), and 2(f) are examples of cycle complements for $r \in \{3, 4, 5\}$. Note that for $r = 3$, the complemented $(2r-1)$-cycle is isomorphic to C_5, one of the Moore graphs.

The cycle complement construction differs from the book in that it gives exactly one uniquely K_r-saturated graph for each r. Also of note is that the cycle complement has no dominating vertex (a vertex adjacent to all other vertices) and is regular.

Given a uniquely K_r-saturated graph G, adding a dominating vertex to G results in a uniquely K_{r+1}-saturated graph. This process is reversible: given a uniquely K_r-saturated graph with a dominating vertex, remove that vertex to find a uniquely K_{r-1}-saturated graph. Repeating this process will eventually result in a graph with no dominating vertex. Starting with the t-book, this process terminates in an independent set, which is uniquely K_2-saturated. This motivates the question: which uniquely K_r-saturated graphs have no dominating vertex?

Conjecture 3.1.1 ([4]). *For each r, there are a finite number of uniquely K_r-saturated graphs with no dominating vertex.*

In an effort to generate more evidence for this conjecture, examples of such graphs are generated. All known examples happen to be regular, which motivates the following conjecture.

Conjecture 3.1.2 ([4]). *For each r, a uniquely K_r-saturated graph with no dominating vertex is regular.*

For $r \geq 3$, a uniquely K_r-saturated graph has diameter two, and is 2-connected. We apply our generation technique with an application-specific pruning mechanism to find these graphs.

3.2 The Search

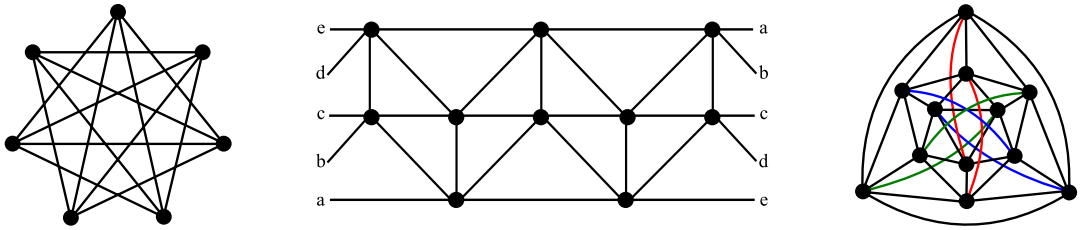
In order to apply isomorph-free generation using ear augmentations, we must show that uniquely K_r-saturated graphs are 2-connected. In fact, we prove a stronger statement using k-connectivity. A graph G is *k-connected* if there exists no set S of $k-1$ vertices so that either $G-S$ is disconnected or $G - S$ consists of a single vertex.

Proposition 3.2.1. *For all $r \geq 4$, if G is a K_r-saturated graph on at least $r+1$ vertices, then G is $(r-2)$-connected.*

Proof. If G is not $(r-2)$-connected, there is a set $S = \{x_1, \ldots, x_{r-3}\}$ of $r-3$ vertices so that $G-S$ has at least two components. Let u and v be vertices in two different components. Then, uv is not an edge in G. Since there is a copy of K_r in $G+uv$, then there is a clique $\{y_1, y_2, \ldots, y_{r-2}\}$ of order $r-2$ so that each vertex y_i in the clique is adjacent to both u and v. At least one of the vertices y_i is not in S, so in $G - S$, u and v are in the same component. This contradicts the assumption that $G - S$ is disconnected, and hence G is $(r-2)$-connected. ∎

Let $\mathcal{U}^r$ be the class of 2-connected graphs G with no copy of K_r as a subgraph and for every edge $e \in \overline{G}$, there is at most one copy of K_r in $G + e$. These constraints are *ear-monotone* in that every G satisfying the constraints and any ear ε has $G - \varepsilon$ satisfying the constraints (except possibly 2-connectedness). To enumerate $\mathcal{U}^r$, we use the default canonical deletion, $\text{Delete}_{\mathcal{U}}(G)$. Since this deletion always removes a deletable ear of minimum length, and we are searching for uniquely K_r-saturated graphs with no dominating vertex, we can prune whenever our graph has a dominating vertex. Further, since $\mathcal{U}^r$ is defined by an ear-monotone property, we prune whenever that property is violated.

The cases for $r \in \{2, 3\}$ are solved, outside of the missing Moore graph of degree 57. Hence, we run our search for $r \in \{4, 5, 6\}$, where we are guaranteed to have at least one uniquely K_r-saturated



(a) (b) (c)

Figure 3: The uniquely K_4-saturated graphs on at most 12 vertices with no dominating vertex.

N	CPU time for $r = 4$	CPU time for $r = 5$	CPU time for $r = 6$
8	1.01s	7.90s	8.80s
9	31.51s	4m 12.75s	4m 14.90s
10	29m 31.46s	5h 24m 38.29s	8h 0m 47.43s
11	1d 8h 13m 59.16s	44d 20h 39m 34.66s	63d 13h 31m 24.30s
12	155d 7h 52m 36.51s		

Table 3: The time to search for uniquely K_r-saturated graphs with at most N vertices.

graph with no dominating vertex of order at most 12. Enumerating $\mathcal{U}_{12}^4$, $\mathcal{U}_{11}^5$, and $\mathcal{U}_{11}^6$ resulted in the following theorems.

Theorem 3.2.2. *There are exactly three uniquely K_4-saturated graphs of order at most 12 without a dominating vertex:*

1. *$\overline{C}_7$, on 7 vertices of degree 4 (Figure 3(a)).*
2. *A triangulation of the Möbius strip, on 10 vertices of degree 5 (Figure 3(b)).*
3. *The icosahedron with antipodal vertices joined, on 12 vertices of degree 6 (Figure 3(c)).*

Theorem 3.2.3. *There is exactly one uniquely K_5-saturated graph of order at most 11 without a dominating vertex: $\overline{C}_9$, on 9 vertices of degree 6 (Figure 2(f)).*

Theorem 3.2.4. *There is exactly one uniquely K_6-saturated graph of order at most 11 without a dominating vertex: $\overline{C}_{11}$, on 11 vertices of degree 8.*

While these graphs were known, an exhaustive search had previously been completed for up to only nine vertices [2].

The search required about 155 days of computation time to search for uniquely K_4-saturated graphs on up to 12 vertices. Timing statistics for smaller N are available in Table 3. Notice that as r increases, the uniquely K_r-saturated graphs become more dense and the restriction on $\mathcal{U}^r$ requires more graphs to be generated, leading to longer search times. This caused the generation of uniquely K_5-saturated and uniquely K_6-saturated graphs on twelve vertices to be left incomplete.

4 Application 2: The Edge Reconstruction Conjecture

In the second application, we restrict the search to sparse 2-connected graphs and utilize the structure of the search tree in order to minimize pairwise comparisons among the list of generated graphs.

4.1 Background

The Reconstruction Conjecture and Edge Reconstruction Conjecture are two of the oldest unsolved problems in graph theory. Given a graph G, the *vertex deck* of G is the multiset of unlabeled graphs given by the vertex-deleted subgraphs $\{G - v : v \in V(G)\}$. The *edge deck* of G is the multiset of unlabeled graphs given by the edge-deleted subgraphs $\{G - e : e \in E(G)\}$. A graph G is *reconstructible* if all graphs with the same vertex deck are isomorphic to G. G is *edge reconstructible* if all graphs with the same edge deck are isomorphic to G.

Conjecture 4.1.1 (The Reconstruction Conjecture)**.** *Every graph on at least three vertices is reconstructible.*

Conjecture 4.1.2 (The Edge Reconstruction Conjecture)**.** *Every graph with at least four edges is edge reconstructible.*

Bondy's survey [1] discusses many classic results on this topic. Greenwell [9] showed that the vertex deck is reconstructible from the edge deck, so a reconstructible graph is also edge reconstructible. Therefore, the Edge Reconstruction Conjecture is weaker than the Reconstruction Conjecture.

Yang [36] showed that the Reconstruction Conjecture can be restricted to 2-connected graphs.

Theorem 4.1.3 (Yang [36])**.** *If all 2-connected graphs are reconstructible, then all graphs are reconstructible.*

The proof considers a separable graph G and tests if the complement $\overline{G}$ is 2-connected. If $\overline{G}$ is 2-connected, $\overline{G}$ is reconstructible (by hypothesis) and since the vertex deck of $\overline{G}$ is reconstructible from the vertex deck of G, G is also reconstructible. If $\overline{G}$ is not 2-connected, Yang reconstructs G directly using a number of possible cases for the structure of G. There has been work to make Yang's theorem unconditional by reconstructing separable graphs such as trees [16], cacti [8, 21], and separable graphs with no vertices of degree one [17], but separable graphs with vertices of degree one have not been proven to be reconstructible.

Verifying the Reconstruction Conjecture requires that every pair of non-isomorphic graphs have non-isomorphic decks. Running a pair-wise comparison on every pair of isomorphism classes on n vertices is quickly intractable. McKay [18] avoided this issue and verified the conjecture on graphs up to 11 vertices by incorporating the vertex deck as part of the canonical deletion. McKay used vertex augmentations to generate the graphs, so a canonical deletion in this search is essentially selecting a canonical vertex-deleted subgraph. His technique selects the deletion based only on the vertex deck, so two graphs with the same vertex deck would be immediate siblings in the search tree. With this observation, only siblings require pairwise comparison, making the verification a reasonable computation. We use a modification of McKay's technique within the context of 2-connected graphs to test the Edge Reconstruction Conjecture on small graphs. This strategy was first proposed in unpublished work of Hartke, Kolb, Nishikawa, and Stolee [10].

4.2 The Search Space

To search for pairs of non-isomorphic graphs with the same edge deck, we adapt McKay's sibling-comparison strategy as well as a density argument. If a graph has sufficiently high density, then the graph is edge reconstructible.

Theorem 4.2.1 (Lovász, Müller [15, 22])**.** *A graph on N vertices and E edges with either $E > \frac{1}{2}\binom{N}{2}$ or $E > 1 + \log_2(N!)$ is edge reconstructible.*

Note that for all $N \geq 11$, $1 + \log_2(N!) < \frac{1}{2}\binom{N}{2}$.

Definition. Let $\mathcal{R}_N$ be the class of 2-connected graphs G with at most N vertices and at most $1 + \log_2(N!)$ edges.

Note that this definition of $\mathcal{R}_N$ bounds the number of edges as a function of N which is independent of the number of vertices of a specific graph.

Corollary 4.2.2. *For $N \geq 11$, all 2-connected graphs G with at most N vertices and $G \notin \mathcal{R}_N$ are edge reconstructible.*

We shall use $\mathcal{R}_N$ as our search space. It is deletion-closed, since removing an ear will always decrease the number of edges.

Within the context of the ear-augmentation generation algorithm, we generate 2-connected graphs. When trivial ears are added, these are the same as edge-augmentations. We will show that if a non-trivial ear is added, then the resulting graph is edge reconstructible and its edge deck does not need to be compared to other edge decks. Hence, an edge deck must be compared only when the final augmentation that generated the graph is an edge augmentation, where the canonical deletion can be selected using the edge deck.

We begin by discussing graphs which are known to be reconstructible or edge reconstructible.

Proposition 4.2.3. *A 2-connected graph G is edge reconstructible if any of the following hold:*

1. *There is an ear with at least two internal vertices.*
2. *There is a branch vertex v which is incident to only non-trivial ears.*
3. *G is regular.*

Proof. **(1)** By reconstructing the degree sequence, we recognize that all vertices have degree at least two. Since there is an ear with at least two internal vertices, there is an edge internal to that ear with endpoints of degree two. In that edge-deleted card, there are exactly two vertices of degree one, which must be connected by the missing edge, giving G.
(2) Let d be the degree of v. By reconstructing the vertex deck, we can recognize that the card for $G - v$ is missing a vertex of degree d and that there are d vertices of degree one in $G - v$. Attaching v to these vertices reconstructs G.
(3) For a d-regular graph G, every edge-deleted subgraph $G - e$ has exactly two vertices of degree $d - 1$ corresponding to the endpoints of e. ∎

Graphs satisfying any of the conditions of Proposition 4.2.3 are called *detectably edge reconstructible* graphs.

4.3 Canonical deletion in $\mathcal{R}_N$

In this section, we describe a method for selecting a canonical ear to delete from a graph in $\mathcal{R}_N$.

If we are able to determine that G is edge reconstructible, then the canonical deletion does not need to be generated from the edge deck. In such a case, we default to the canonical deletion algorithm $\text{Delete}_{\mathcal{F}}(G)$, where the canonical labeling of G gives the lex-first ear ε of minimum length so that $G - \varepsilon$ 2-connected.

If G is not detectably edge reconstructible, then all ears of G have at most one internal vertex, and every branch vertex is incident to at least one trivial ear. These properties allow us to find either a trivial ear or an ear of order one whose deletion remains 2-connected. Compute the minimum r

N	$g(N)$	$\lvert\mathcal{R}_N\rvert$	Diff 1	Diff 2	Diff 3	CPU time
8	16	4804	145	177	187	8.01s
9	19	111255	6.19×10^3	5.72×10^3	4.77×10^3	5m 33.85s
10	22	3051859	7.13×10^5	6.00×10^5	4.21×10^5	6h 33m 40.59s
11	26	308400777	9.44×10^7	7.28×10^7	3.83×10^7	32d 20h 38m 08.16s
12	29	25615152888	12.00×10^9	9.60×10^9	4.47×10^9	10y 362d 13h 05m 39.13s

Table 4: Comparing $\lvert\mathcal{R}_N\rvert$ and the time to check $\mathcal{R}_N$. Here, $g(N) = 1 + \lfloor \log_2(N!) \rfloor$.

so that there exists an ear ε in G of order r so that $G - \varepsilon$ is 2-connected. We prefer to select a trivial ear when available.

Out of the choices of possible order-r ear deletions, count the multiplicities for the degree set of the ear endpoints. Find the pair $\{d_1, d_2\}$ of endpoint degrees which has minimum multiplicity over all deletable ears of order r in G breaking ties by using the lexicographic order. Out of the deletable ears of order r and endpoint degrees $\{d_1, d_2\}$, we must select a canonical ear using the edge deck. If $r = 0$, any trivial deletable ear ε corresponds to the edge-deleted subgraph $G - \varepsilon$. By computing the canonical labels of these cards and selecting the lexicographically-least canonical string, we can select a canonical edge. If $r = 1$, there are two edges in the ear that can be deleted to form edge-deleted subgraphs with a single vertex of degree 1 connected to a 2-connected graph. We compute the canonical labels of both cards, select the lexicographically-least canonical string, then find the lex-least string of those strings.

Due to the nature of the reconstruction problem, this canonical deletion procedure is not perfect. There are graphs G containing trivial ears $\varepsilon_1, \varepsilon_2$ whose deletions $G - \varepsilon_1$ and $G - \varepsilon_2$ are isomorphic, but ε_1 and ε_2 are not in orbit within G. If the edge-deleted subgraph $G - \varepsilon_1$ is selected as the canonical edge card, the deletion algorithm must accept both ε_1 and ε_2 as canonical deletions. This leads to a duplication of G in the search tree, but only in the limited case of a graph G which is not detectably edge reconstructible *and* such ambiguity appears. A similar concern occurs for the vertex-deletion case, but is not explained in [18].

To compare graphs with the same canonical deletion, we use three comparisons. The first compares the degree sequences. The second compares a custom reconstructible invariant[4], which is based on the degree sequence of the neighborhood of each vertex. The third and final check compares the sorted list of canonical strings for the edge-deleted subgraphs. During the search, there was no pair of graphs which satisfied all three of these checks.

4.4 Results

With the canonical deletion $\text{Delete}_{\mathcal{R}}(G)$, $\mathcal{R}_N$ was generated and checked for collisions in the edge decks of graphs which are not detectably reconstructible. Table 4 describes the computation time for $N \in \{8, \ldots, 12\}$.

With this computation, we have the following theorem.

Theorem 4.4.1. *All 2-connected graphs on at most* 12 *vertices are edge reconstructible.*

This computation extends the previous result that all graphs of order at most 11 are vertex reconstructible [18]. To remove the 2-connected condition of Theorem 4.4.1, there are three possible methods. First, prove Yang's Theorem (Theorem 4.1.3) for the edge reconstruction problem.

[4] This invariant is not theoretically interesting, but is available in the source code. See the `GraphData::computeInvariant()` method.

Second, Yang's Theorem could be made unconditional by proving that separable graphs are reconstructible or edge reconstructible. Third, a second stage of search could be designed to combine a list of two-connected graphs to form sparse separable graphs and test edge reconstruction on those cases.

5 Conclusion

Generating 2-connected graphs by ear augmentations and removing isomorphs by canonical ear deletion is an effective and general technique. The computation times show that the technique is more effective in the case of ear-monotone properties such as uniquely K_4-saturated graphs or when the structure of the ear decomposition is essential to the problem at hand, such as verifying the edge reconstruction conjecture on small graphs.

A forthcoming work [28] applies the generation technique to search for dense graphs with a fixed number of perfect matchings (see [5] and [12] for background). Previous work [12] classified the infinite family of graphs into a particular combination of finite pieces, which can be found through our generation process. This results in exact structure theorems for a larger class of parameters, where the exact structure is computationally generated. Our implementation is general enough to allow for such extensions to generate other families of 2-connected graphs, and is concurrent to allow for large computations to be run quickly in real time.

Acknowledgements

The author thanks Stephen G. Hartke, Hannah Kolb, Jared Nishikawa, Kathryn T. Stolee, and Paul Wenger for interesting discussions concerning the problems addressed in this work and for improving the quality of this work. Special thanks to Joshua Cooper for Figure 3(b).

This work was completed utilizing the Holland Computing Center of the University of Nebraska. Thanks to the Holland Computing Center faculty and staff including Brian Bockelman, Derek Weitzel, and David Swanson. Thanks to the Open Science Grid for access to significant computer resources. The Open Science Grid is supported by the National Science Foundation and the U.S. Department of Energy's Office of Science.

References

[1] J. A. Bondy. A graph reconstructor's manual. In *Surveys in combinatorics, 1991 (Guildford, 1991)*, volume 166 of *London Math. Soc. Lecture Note Ser.*, pages 221–252. Cambridge Univ. Press, Cambridge, 1991.

[2] J. Cooper, 2011. personal communication.

[3] J. Cooper, J. Lenz, T. D. LeSaulnier, P. Wenger, and D. B. West. Uniquely C_4-saturated graphs. *Graphs and Combinatorics*, pages 1–9, 2011.

[4] J. Cooper and P. Wenger, 2011. personal communication.

[5] A. Dudek and J. Schmitt. On the size and structure of graphs with a constant number of 1-factors, 2010. submitted.

[6] P. Erdős, A. Hajnal, and J. W. Moon. A problem in graph theory. *Amer. Math. Monthly*, 71:1107–1110, 1964.

[7] P. Erdős, A. Rényi, and V. T. Sós. On a problem of graph theory. *Studia Sci. Math. Hungar.*, 1:215–235, 1966.

[8] D. Geller and B. Manvel. Reconstruction of cacti. *Canad. J. Math.*, 21:1354–1360, 1969.

[9] D. L. Greenwell. Reconstructing graphs. *Proc. Amer. Math. Soc.*, 30:431–433, 1971.

[10] S. G. Hartke, H. Kolb, J. Nishikawa, and D. Stolee. Edge-reconstruction for small 2-connected graphs, 2009. unpublished.

[11] S. G. Hartke and A. Radcliffe. Mckay's canonical graph labeling algorithm. In *Communicating Mathematics*, volume 479 of *Contemporary Mathematics*, pages 99–111. American Mathematical Society, 2009.

[12] S. G. Hartke, D. Stolee, D. B. West, and M. Yancey. On extremal graphs with a given number of perfect matchings, 2011. in preparation.

[13] A. J. Hoffman and R. R. Singleton. On Moore graphs with diameters 2 and 3. *IBM Journal of Research and Development*, 4(5):497–504, 1960.

[14] P. Kaski and P. R. J. Östergård. *Classification Algorithms for Codes and Designs.* Number 15 in Algorithms and Computation in Mathematics. Springer-Verlag, Berlin Heidelberg, 2006.

[15] L. Lovász. A note on the line reconstruction problem. *J. Combinatorial Theory Ser. B*, 13:309–310, 1972.

[16] B. Manvel. Reconstruction of trees. *Canad. J. Math.*, 22:55–60, 1970.

[17] B. Manvel. On reconstructing graphs from their sets of subgraphs. *Journal of Combinatorial Theory, Series B*, 21(2):156–165, 1976.

[18] B. D. McKay. Small graphs are reconstructible. *Australas. J. Combin.*, 15:123–126, 1997.

[19] B. D. McKay. Isomorph-free exhaustive generation. *J. Algorithms*, 26(2):306–324, 1998.

[20] B. D. McKay. nauty user's guide (version 2.4). *Dept. Computer Science, Austral. Nat. Univ.*, 2006.

[21] S. D. Monson. The reconstruction of cacti revisited. *Congr. Numer.*, 69:157–166, 1989. Eighteenth Manitoba Conference on Numerical Mathematics and Computing (Winnipeg, MB, 1988).

[22] V. Müller. The edge reconstruction hypothesis is true for graphs with more than $n \cdot \log_2 n$ edges. *J. Combinatorial Theory Ser. B*, 22(3):281–283, 1977.

[23] R. Pordes, D. Petravick, B. Kramer, D. Olson, M. Livny, A. Roy, P. Avery, K. Blackburn, T. Wenaus, et al. The Open Science Grid. In *Journal of Physics: Conference Series*, volume 78, pages 12–57. IOP Publishing, 2007.

[24] R. Robinson. Tables. available at http://www.cs.uga.edu/ rwr/publications/tables.pdf.

[25] R. Robinson. Enumeration of non-separable graphs*. *Journal of Combinatorial Theory*, 9(4):327–356, 1970.

[26] N. J. A. Sloane. A001349: Number of connected graphs with n nodes, 2000. http://oeis.org/A002218.

[27] N. J. A. Sloane. A002218: Number of unlabeled nonseparable (or 2-connected) graphs (or blocks) with n nodes, 2000. http://oeis.org/A002218.

[28] D. Stolee. Generating p-extremal graphs, 2011. in preparation.

[29] D. Stolee. TreeSearch user guide, 2011. available at http://github.com/derrickstolee/TreeSearch.

[30] D. Thain, T. Tannenbaum, and M. Livny. Distributed computing in practice: the Condor experience. *Concurrency - Practice and Experience*, 17(2-4):323–356, 2005.

[31] P. Turán. On an extremal problem in graph theory. *Matematikai és Fizikai Lapok*, 48:436–452, 1941.

[32] T. Walsh and E. Wright. The k-connectedness of unlabelled graphs. *Journal of the London Mathematical Society*, 2(3):397, 1978.

[33] D. J. Weitzel. *Campus Grids: A framework to facilitate resource sharing.* Masters thesis, University of Nebraska - Lincoln, 2011.

[34] P. Wenger. *Uniquely C_k-saturated graphs.* PhD dissertation, University of Illinois Urbana–Champaign, Department of Mathematics, 2010.

[35] D. B. West. *Introduction to Graph Theory.* Prentice-Hall, second edition, 2001.

[36] Y. Z. Yang. The reconstruction conjecture is true if all 2-connected graphs are reconstructible. *J. Graph Theory*, 12(2):237–243, 1988.