How False is Kempe's Proof of the Four Color Theorem? (Part II)

Ellen Gethner
ellen.gethner@cudenver.edu

Bopanna Kallichanda
bkallich@ouray.cudenver.edu

Alexander Mentis
alexander.mentis@usma.edu

Sarah Braudrick Sumeet Chawla Andy Clune Rachel Drummond
Panagiota Evans William Roche Nao Takano

University of Colorado Denver
Department of Computer Science
Department of Mathematics
Denver, Colorado
23 November 2008

Abstract

In this paper we continue the investigation of A.B. Kempe's flawed proof of the Four Color Theorem from a computational and historical point of view. Kempe's "proof" gives rise to an algorithmic method of coloring planar graphs that sometimes yields a proper vertex coloring requiring four or fewer colors. We investigate a recursive version of Kempe's method and a modified version based on the work of I. Kittell. Then we empirically analyze the performance of the implementations on a variety of historically-motivated benchmark graphs and explore the usefulness of simple randomization in four-coloring small planar graphs. We end with a list of open questions and future work.

1 Introduction

The Four Color Theorem for Planar Graphs states that, given a planar graph G, the vertices of G can be properly colored with at most four colors. While the Four Color Theorem was proven in 1977 through the use of a computer and irreducible sets [AH77, AH77a, AH77b, AHK77, RSST96, RSST97], no proof has been found that can be verified by a human without the use of a computer. Alfred Kempe seemingly came close to accomplishing this in 1879 when he presented a "proof" of the Four Color Theorem in his paper "On the Geographical Problem of the Four Colours"; however, his proof contained a flaw, published by P. J. Heawood in 1890 and discovered independently by de la Vallée Poussin in 1896 [Wil02a]. Although Kempe was unable to repair the flaw, his innovation of Kempe chains and Kempe chain switches remain useful to graph theorists, and it is interesting to explore the boundaries of his technique [GS03]. In particular, we focus our attention on the work of Errera, who was the first person to study the importance of the order in which the vertices are labeled [Err21]. For a comprehensive history of the Four Color Theorem, see [Wil02b, Wil02a, Ore67, FF98, BLW86].

Following [Err21, HW98, GS03, Wag], we implemented Kempe's method of proof as a recursive algorithm (Algorithm Kempe) on different vertex labelings for some well-known graphs of nine vertices or more. For labelings resulting in the algorithm's inability to properly four-color the graphs, we identify vertices that cause irrevocable Kempe chain failures (the source of the flaw in Kempe's proof), and quantify the graphs' failure rates. In acknowledgement that Algorithm Kempe sometimes correctly four-colors the vertices of a planar graph, we explore some improvements to Algorithm Kempe including random selection among all Kempe chain choices and using random Kempe-Kittell chain switches to overcome irrevocable Kempe chain tangles, following [Kit35, HW98, Wag02, Wag, AS86, MS91]. While there may be different flaws that also result in failure to four-color a planar graph, our improvements focus solely on circumventing the flaw identified by Heawood and Poussin, since that is the flaw addressed by our implementation of Kittell's approach. Where Kempe-Kittell chain switches allow Algorithm Kempe to continue, we correlate the identified vertex with the number of Kempe-Kittell chain switches required to overcome the tangle.

2 Definitions and Algorithm

It is important to understand Kempe's alleged proof and the flaw that led to our investigations. For completeness and ease of reference, the following definitions and algorithm are taken directly from [GS03]. In all of the following, C_x refers to color C_x from the set of four possible colors C_1, C_2, C_3, and C_4.

Definition 2.1 (C_1C_2-Kempe Chain): Let G be a planar graph whose vertices have been properly colored and suppose $v \in V(G)$ is colored C_1. Define the C_1C_2-*Kempe chain containing* v to be the maximal connected component of G that

- **contains v, and**
- **contains only vertices that are colored with elements from $\{C_1, C_2\}$.**

Importantly, the maximality of the set of colored vertices in a C_1C_2-Kempe Chain guarantees that interchanging all occurrences of C_1 and C_2 preserves the proper coloring of G. Thus, we define

Definition 2.2 (C_1C_2-Kempe Chain Switch): Let KC be a C_1C_2-Kempe chain. A C_1C_2-*Kempe chain switch* interchanges all values of C_1 and C_2 in KC.

Last, but hardly least, to illustrate the potential flaw in Kempe's method, we will define the notion of *irrevocable Kempe chain tangle*. To help visualize the set-up, see Figure 1.

Definition 2.3 (Irrevocable Kempe Chain Tangle): Let G be a plane graph, all of whose vertices, with the exception of one, have been properly colored with four colors. Denote the exceptional vertex by v. Suppose further that $degree(v) = 5$, and that the five neighbors of v (denoted v_1,v_2,v_3,v_4,v_5) are colored in cyclic counterclockwise order (endowed by the planar embedding) by G_1RG_2BY respectively. Moreover, assume that the RB-Kempe chain containing v_2 also contains v_4, and that the RY-Kempe chain containing v_2 also contains v_5.

Denote the GB-Kempe chain containing v_1 by KC_1 and the GY-Kempe chain containing v_3 by KC_2. We say that Algorithm Kempe causes an *irrevocable Kempe chain tangle on vertex* v if either

- **following a G_1B-Kempe chain switch on KC_1 by a G_2Y-Kempe chain switch on KC_2 causes v_5 to be re-colored *Green*, or else**
- **following a G_2Y-Kempe chain switch on KC_2 by a G_1B-Kempe chain switch on KC_1 causes v_4 to be re-colored *Green*.**

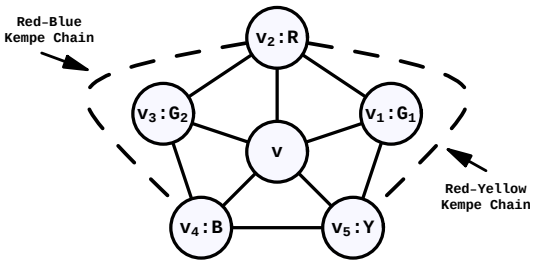

Figure 1: **Set-up for the faulty case in Kempe's proof.**

In particular, at least one of the original barriers afforded by either the RB-Kempe chain containing v_2 and v_4, or the RY-Kempe chain containing v_2 and v_5 has been broken by two successive GX-Kempe chain switches, where $X \in \{Y, B\}$. Moreover, the second GX-Kempe chain contains two vertices in the neighborhood of v, which reintroduces a vertex colored *Green* as a neighbor of v; thus the procedure has not made *Green* available for vertex v.

We use the adjective **irrevocable** in Definition 2.3 because under the initial hypotheses, a Kempe chain tangle might occur: that is, one of either the RB-Kempe chain or the RY-Kempe chain may be "broken" by the two successive GX-Kempe chain switches, but the procedure need not force any of the neighbors of v to be re-colored with *Green*. In that case, v will be properly colored with *Green*.

With these definitions, we now have the vocabulary to describe Algorithm Kempe.

2.1 Algorithm Kempe

INPUT:

- A connected plane graph G with n vertices, labeled (*in some order*) with distinct elements from $\{1, \ldots, n\}$.
- A set of four colors $C = \{C_1, C_2, C_3, C_4\}$ with which to (attempt to) properly color the vertices of G.
- For the purpose of recursion, let $G_0 := G$

OUTPUT:

- Either a proper vertex coloring of G that uses at most four colors, or
- the message "Kempe's algorithm has encountered an irrevocable Kempe chain tangle at vertex V and hence has failed to properly 4-color G."

Gadget Relabel (relabel the vertices):

- Search G_0 for the first occurrence of a vertex of degree five or less; the existence of such a vertex is guaranteed by Euler's Formula. The first occurrence is dictated by the given ordering of the vertices. Call this vertex v_1.
- Find the first occurrence of a vertex of degree five or less in $G_1 := G \setminus v_1$ and label it v_2.
- Find the first occurrence of a vertex of degree five or less in $G_2 := G_1 \setminus v_2$ and label it v_3.
- $\ddots$
- Find the first occurrence of a vertex of degree five or less in $G_i := G_{i-1} \setminus v_i$ and label it v_{i+1}.
- Continue this procedure until the n vertices of G have been labeled.

Gadget Greed (color greedily whenever possible):

- Color v_n in G_{n-1} with the available color of lowest index from C. In this case, since no colors have been used, v_n will be colored C_1.
- Color v_{n-1} in G_{n-2} with the available color of lowest index in C; if v_{n-1} is not adjacent to v_n, then v_{n-1} is colored C_1. On the other hand, if v_{n-1} is adjacent to v_n, color v_{n-1} is colored C_2.
- In general (if possible) color v_i in G_{i-1} with the available color of lowest index from C.

Gadget 4 (perform Kempe chain switches on degree four vertices):

- We encounter a vertex v_i of degree four that cannot be greedily colored. That is, suppose $degree(v_i) = 4$ and the neighbors are colored R, G, B, and Y in counterclockwise order. See Figure 2.
- If there is a RB-Kempe chain containing both the *Red* and *Blue* neighbors of v_i, then there cannot be a YG-Kempe chain that contains *both* of the *Yellow* and *Green* neighbors of v_i. In that case a YG-Kempe chain switch leaves a color available for v_i.
- Otherwise, if there is no RB-Kempe chain containing both the *Red* and *Blue* neighbors of v_i, perform a RB-Kempe chain switch to make a color available for v_i.

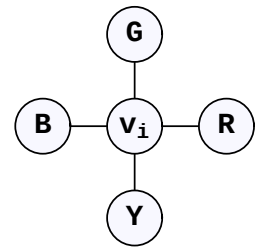

Figure 2: **All four colors are used on the neighbors of v_i.**

Gadgets 5_1 and 5_2 (perform Kempe chain switches on degree five vertices):

- We encounter a vertex v_i of degree five that cannot be greedily colored; a priori, one color is used exactly twice and the other three are used exactly once on the five neighbors of v_i. Without loss of generality, suppose the twice-used color is *Green*. There are only two (up to rotation and reflection) configurations in which this case can occur.
  1. Either the two *Green* neighbors of v_i are next to one another in the embedding, or
  2. the two *Green* neighbors of v_i are separated by one other neighbor of v_i (see Figure 3).

Gadget 5_1 ($degree(v_i) = 5$; two *Green* neighbors next to each other):

- In Configuration 1, a gadget much like *Gadget* 4 will succeed in coloring v_i. In particular, suppose the five neighbors of v_i are colored, in counterclockwise order, by $GGYBR$ (see Figure 3 I).
- If there is no YR-Kempe chain containing both *Yellow* and *Red* neighbors of v_i, then a YR-Kempe chain switch will leave a color available for v_i.
- Therefore, assume there is a YR-Kempe chain containing both *Yellow* and *Red* neighbors of v_i. Thus a BG-Kempe chain containing the *Blue* neighbor of v_i contains neither of the *Green* neighbors of v_i.
- In that case a BG-Kempe chain switch makes *Blue* available for v_i.
- In all cases v_i can be properly colored.

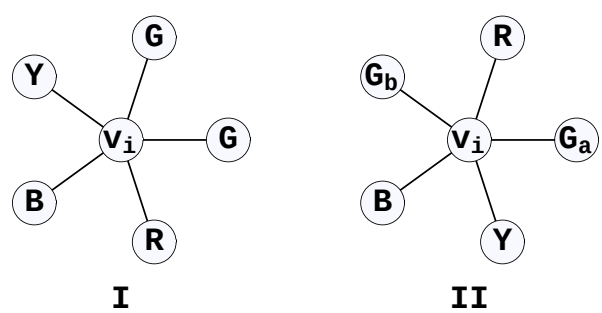

Figure 3: **Two *Green* neighbors of v_i are either next to each other (I) or separated by one other neighbor (II).**

Gadget 5_2 ($degree(v_i) = 5$; two *Green* neighbors are separated by another neighbor of v_i):

- Configuration 2 is the case in which an irrevocable Kempe chain tangle might occur, causing Algorithm Kempe to halt before completing a proper 4-coloring of the graph. Suppose the neighbors of v_i are colored in counterclockwise order by G_aRG_bBY (at this point, it is helpful to distinguish between the two *Green* vertices). See Figure 3 II.

- If there is a RB-Kempe chain that does not contain both the *Red* and *Blue* neighbors of v_i then a RB-Kempe chain switch leaves a color available for v_i.

- If there is a RY-Kempe chain that does not contain both *Red* and *Yellow* neighbors of v_i then a RY-Kempe chain switch makes a color available for v_i.

- Otherwise, we must attempt both a G_aB-Kempe chain switch followed by a G_bY Kempe chain switch (or vice versa).

- If no irrevocable Kempe chain tangle occurs, then we successfully color v_i with *Green* and move on to vertex v_{i-1}.

- Otherwise, **halt** and return an error message that the offending vertex is v_i.

BEGIN

Step 1: Use *Gadget Relabel* to label the vertices of G.
Step 2: for i=n **downto** 1 **do**
Attempt to color v_i in graph G_{i-1}:

- (a) **if** v_i can be greedily colored in graph G_{i-1} by *Gadget Greed* **then** do so, **else**

- (b) **if** $degree(v_i) = 4$ in G_i **then** color v_i using *Gadget* 4 **else**

- (c) **if** $degree(v_i) = 5$ in G_{i-1} in configuration 1 **then** color v_i using *Gadget* 5_1 **else**

- (d) **if** $degree(v_i) = 5$ in G_{i-1} in configuration 2 **then** try to color v_i using *Gadget* 5_2 employing two Kempe chain switches: try both orders if necessary

END

Thus, it is obvious that it is possible to color vertices of degree three or less with no more than a fourth color, and it has been shown that it is always possible to color vertices of degree four through the use of Kempe chain switches [Hea90]. Algorithm Kempe only encounters difficulties upon vertices of degree five or more, but it has been shown that Algorithm Kempe will always succeed in properly four-coloring any graph containing eight or fewer vertices (which may contain vertices of degree five or more) [GS03]. In light of the fact that Kempe's method of proof works in some, but not all cases, we were interested in identifying patterns of when the algorithm halts without producing a proper four-coloring on our benchmark graphs. In particular, we explore the usefulness of simple randomization when used with Kempe-Kittell chain switches to improve its success on small planar graphs.

3 Results

3.1 Identification of Vertex Failures

We first implemented Algorithm Kempe and explored its success in properly four-coloring nine well-known graphs: the Fritsch [FF98], Soifer [Soi97], Poussin [Wil02b], Errera [Err21, HW98, Wag], Heawood [Hea90], Dodecahedral, Icosahedral, Octahedral, and Cubical graphs. The first five are known counterexamples for Algorithm Kempe (on at least one labeling)[GS03]. Because the Dodecahedral, Octahedral, and Cubical graphs all have vetices of degree less than or equal to four, Algorithm Kempe is known to always successfully color them and they served as benchmarks for our implementations. Further, although the Icosahedral graph is 5-regular, we did not expect Kempe's method to fail on any labeling of vertices, and thus that graph served as a benchmark graph as well. See also problem 4 in Section 4.

Five groups worked independently of one another to implement Algorithm Kempe and test it on these graphs. For the graphs containing nine vertices or fewer, each group explored Algorithm Kempe's results for all $n!$ labelings. For the graphs containing more than nine vertices, each group independently tested a random subset of at least 9! labelings.

While we expected different failure rates for the graphs with more than nine vertices due to the use of different labeling subsets among the groups, we expected the failure rates for Fritsch, Soifer, and the four benchmark graphs to be the same. Instead, while the benchmark graphs produced no failures, as expected, failure rates did vary for Fritsch and Soifer due to differences in the individual implementations or failure rate calculations. In the case of Group 2, when Gadget 5_2 is required the implementation only tests one of the two possible Kempe chain switch orders, resulting in a higher failure rate. This difference in implementation, however, gives us an idea of how many Kempe chain tangles can be "fixed" by changing the order in which the switches are performed (Table 1).

Table 1: **Kempe method failure rates by graph.**

	Group 1			Group 2			Group 3			Group 4			Group 5		
	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max
Fritsch	13.947	13.947	13.947	14.031	14.083	14.131	3.598	3.599	3.600	13.687	13.687	13.687	13.630	13.635	13.637
Soifer	1.692	1.692	1.692	1.783	1.832	1.859	0.520	0.523	0.525	1.635	1.635	1.635	1.620	1.620	1.620
Octahedron	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Cube	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Icosahedron	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Dodecahedron	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Poussin	0.610	0.620	0.627	0.149	0.153	0.156	0.014	0.017	0.019	0.165	0.165	0.165	0.180	0.186	0.190
Errera	9.730	9.755	9.772	3.350	3.383	3.402	8.140	8.168	8.186	7.302	7.302	7.302	10.680	10.798	10.866
Heawood	7.089	7.124	7.153	0.372	0.382	0.390	1.089	1.097	1.111	0.387	0.387	0.387	3.199	3.203	3.210

We initially compared the vertices that caused irrevocable Kempe chain tangles for each implementation on all graphs. Because each group tested all 9! labelings for the two 9-vertex graphs (Fritsch and Soifer), each implementation agreed on the vertices that caused failures for Fritsch and Soifer, as expected. An interesting and unpredicted discovery, however, was that despite the differences in the labeling subsets tested by each group for the graphs containing more than nine vertices, there was considerable consensus among the groups on the vertices that cause failures.

In Figure 4, vertices shown in red are those vertices that all groups found to result in an irrevocable Kempe chain tangle for at least one labeling. Vertices shown in yellow are those vertices that at least one group, but not all, found to fail. Vertices shown in white were not found by any groups to cause a failure on the labelings tested. As one can see, the failure patterns of the vertices are highly symmetrical for all graphs except the Poussin graph, which itself is a fairly asymmetrical graph. For the Fritsch and Soifer graphs, since all $n!$ labelings were tested, we know that the vertices shown in white will never cause Kempe chain tangles for any labeling. For the remaining graphs, we predict that the vertices shown in yellow would eventually become red as more labelings are

explored. We cannot predict anything for the vertices shown in white of degree five or more - they may eventually fail, or they may not. Nevertheless, these results lead us to ask the question: are there commonalities among these vertices that can be exploited to improve Algorithm Kempe? We leave this as an open question.

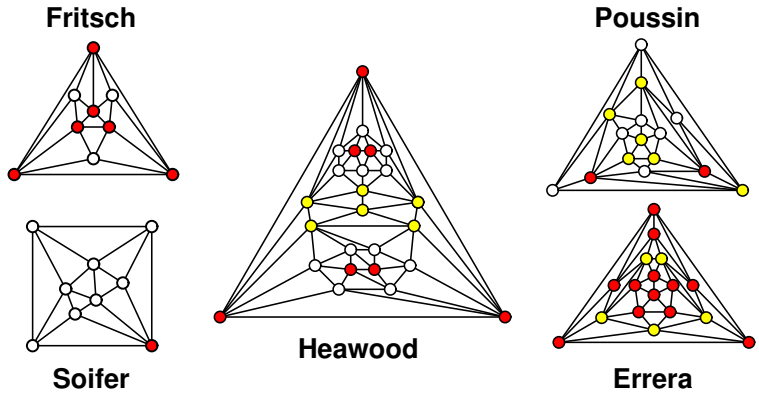

Figure 4: **Graph vertex failures.**

The next step was to add randomization, studied in [Kit35, HW98, Wag02, Wag, AS86, MS91], through the application of Kempe-Kittell chain switches [Kit35] and the use of randomization of the choice of Kempe or Kempe-Kittell chain switches, rather than heuristics, at various stages of the algorithm. In contrast to the study of randomization for large graphs in [AS86, MS91], we continue to investigate small, historically-significant benchmark graphs.

3.2 Randomization Implementation

In our recursive implementation of Kempe's method, there are several decision points in the algorithm where we must choose among multiple Kempe chains upon which to perform a switch. In both Gadgets 4 and 5_1, in the case that the vertex cannot be greedily colored, there will be up to four Kempe chains from which to choose; our implementation randomly chooses one that results in a successful coloring. In Gadget 5_2, two Kempe chain switches must be performed, but the order of the switches is not specified in the algorithm. Theorem 3.1 shows that the order in which the switches are performed can influence the success of the operation. In light of this knowledge, we randomize the choice of which Kempe chain switch to perform first and perform the alternative order only if the first order fails.

Theorem 3.1 (Gadget 5_2 is Order Dependent) *In Algorithm Kempe, Gadget 5_2 is sometimes noncommutative. That is, the order in which one chooses to execute the Kempe Chain switches on KC_1 and KC_2 may matter; in one order an irrevocable Kempe Chain Tangle could occur and in the other order, no Kempe Chain Tangle occurs.*

Proof It suffices to exhibit a plane graph G and a labeling of the vertices of G that cause Algorithm Kempe to execute Gadget 5_2 in the following way: upon that execution, one of the two choices of Kempe chain switch orders causes an irrevocable Kempe chain tangle while the other does not. To this end, we call upon the Fritsch graph, which we denote by F. In Figure 5, the labeling of the vertices in F (the uppermost graph) leads to a successful four-coloring with the exception of vertex 1, whereupon Gadget 5_2 must be invoked. Following the arrows marked "**A**," one choice of Kempe chain switch order has been executed successfully, and vertex 1 is colored Green. Following the arrows marked "**B**," the other Kempe Chain switch order has been followed, leading to an irrevocable Kempe chain tangle. ∎

In the case that both orders fail, we encounter the previously defined irrevocable Kempe chain tangle and turn to Kempe-Kittell chain switches in an attempt to solve the impasse. Kempe-Kittell

chains present yet another opportunity for randomization of choices. To better understand these choices, we first define the eight Kempe-Kittell chains identified by Kittell in [Kit35].

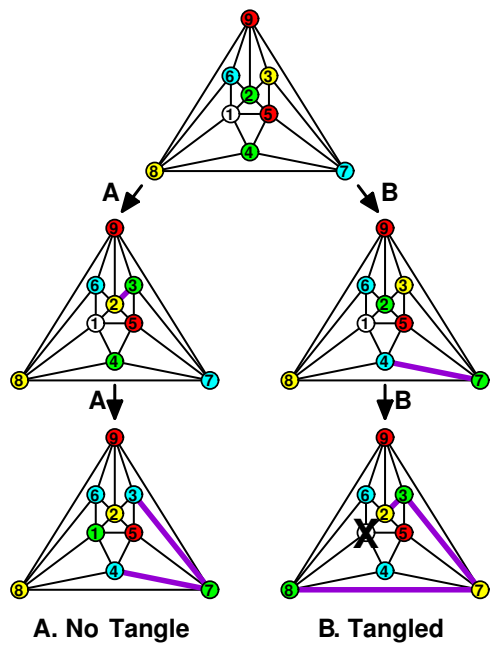

Figure 5: **Gadget 5_2 in Algorithm Kempe does not commute. The thick purple lines highlight the current Kempe chain switch.**

We use exactly the notation and Kempe Chain switches as suggested by Kittell in [Kit35]. The new gadget, called "Gadget Kittell," is invoked only when Gadget 5_2 is called upon in Algorithm Kempe and fails. For reference, see Figure 1.

Definition 3.2 (Gadget Kittell)

1. ***Chain*** α**: Perform a RB-Kempe Chain switch beginning either on v_2 or v_4.**
2. ***Chain*** β**: Perform a RY-Kempe Chain switch beginning either on v_2 or v_5.**
3. ***Chain*** γ**: Perform a GY-Kempe Chain switch beginning either on v_1 or v_5.**
4. ***Chain*** δ**: Perform a GB-Kempe Chain switch beginning either on v_3 or v_4.**
5. ***Chain*** ϵ**: Perform a BY-Kempe Chain switch beginning either on v_4 or v_5.**
6. ***Chain*** ζ**: Perform a GB-Kempe Chain switch beginning either on v_1 or v_4.**
7. ***Chain*** η**: Perform a GY-Kempe Chain switch beginning either on v_3 or v_5.**
8. ***Chain*** θ**:Perform a RG-Kempe Chain switch beginning on any of v_2 or v_1.**

Upon encountering an irrevocable Kempe chain tangle, we randomly choose one of the eight Kempe-Kittell chains in *Gadget Kittell* and continue to randomly execute switches from that list until we reach a coloration of the graph that allows us to successfully color the vertex causing the impasse or until a fixed number of Kempe-Kittell chain switches have failed (we chose an upper limit of 100 Kempe-Kittell chain switches).

Thus Algorithm Kempe is modified as follows to create Algorithm Kempe-Kittell:

BEGIN

Step 1: Use *Gadget Relabel* to label the vertices of G.
Step 2: for i=n **downto** 1 **do**
Attempt to color v_i in graph G_{i-1}:

- (a) **if** v_i can be greedily colored in graph G_{i-1} by *Gadget Greed* **then** do so, **else**
- (b) **if** $degree(v_i) = 4$ in G_i **then** color v_i using *Gadget* 4 on a randomly selected viable Kempe chain **else**
- (c) **if** $degree(v_i) = 5$ in G_{i-1} in configuration 1 **then** color v_i using *Gadget* 5_1 on a randomly selected viable Kempe chain **else**
- (d) **if** $degree(v_i) = 5$ in G_{i-1} in configuration 2 **then** try to color v_i using *Gadget* 5_2 employing two Kempe chain switches: randomly select an order in which to perform the switches; try both orders if necessary

- (e) **if** *irrevocable Kempe chain tangle* **then** select a random Kempe-Kittell chain using *Gadget Kittell* until v_i successfully colored or 100 attempts have failed

END

The fixed limit on the number of failures is required because it is unknown if there is always a series of one or more Kempe-Kittell chain switches that will result in successful resolution of the impasse. The set of possible Kempe-Kittell chain switch combinations that can affect the five vertices adjacent to v_i, called the *impasse group*, is known to have a lower bound of 120 [Kit35], but it is impractical to determine and check the upper bound for even a small arbitrary graph. The use of heuristics to guide the search of the impasse group has been studied for large graphs [MS91], but our interest was in determining algorithm performance when executing a purely random sequence of Kempe-Kittell chain switches to color a small graph, as this could provide an easy way to improve the performance of Algorithm Kempe for those cases.

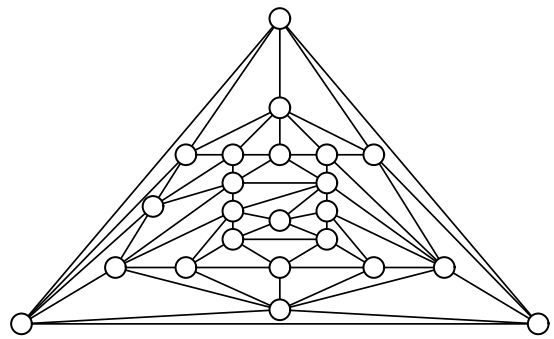

Figure 6: **Kittell Graph.**

Our randomized recursive implementation of Kempe's method always succeeded in four-coloring the graphs we tested. We ran the algorithm 500 times for each of the nine graphs tested in the original implementation and, additionally, the Kittell Graph (Figure 6)[Kit35]. We kept track of the number of times a Kempe-Kittell chain switch was required to solve an impasse (Table 2) and the vertices causing the impasse (Appendix A).

	Kempe-Kittell Switches (Max)	Max Failure Vertices[a]
Fritsch	9	i
Soifer	7	c
Octahedron	0	
Cube	0	
Icosahedron	0	
Dodecahedron	0	
Poussin	6	d
Errera	73	a
Heawood	6	a, b, l, t, u
Kittell	11	n, r

Table 2: **Maximum number of Kempe-Kittell chain switches required for any vertex.**

[a]Appendix A

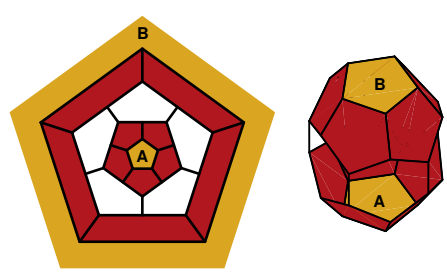

Figure 7: **Errera map: planar representation and coordinatized as a Fullerine C_{30} in $\mathbb{R}^3$.**

Our fixed upper limit of 100 for Kempe-Kittell chain switches was more than sufficient since, for most impasses encountered in our tests, eleven or fewer randomly-chosen Kempe-Kittell chain switches were sufficient to achieve a successful four-coloring. The exception to this was the Errera graph, which contained two vertices that required over 70 randomly-chosen Kempe-Kittell chain switches on the regions marked **A** and **B** in Figure 7 to achieve successful four-coloring. We observe

that these two vertices are the only two vertices in the Errera graph that do not have any neighbors of degree greater than five, and they are the polar regions of Errera's 17-country counterexample when described as a spherical map as shown in [HW98, Figure 2][Wag] and as a Fullerine C_{30} in our Figure 7.

3.3 Comparison of Original and Randomized Implementations

We achieve proper four-coloring of all of our graphs on 100% of our runs through the inclusion of randomly-selected Kempe-Kittell chain switches. In addition to this, the percentage of times that *Gadget Kittell* was required in our algorithm indicates the percentage of irrevocable Kempe chain tangles encountered by our randomized algorithm, which we can compare to the failure rates from the original five groups' implementations (Table 3 and Appendix A).

When we make this comparison to the average failure rate of the original five implementations, we see that our algorithm outperforms the average original algorithm's performance on the two graphs for which all $n!$ labelings were tested (Fritsch and Soifer graphs). In fact, our randomized implementation nearly matches the lowest failure rates observed among the original six implementations: 3.60% and 0.52% for Fritsch and Soifer, respectively (Table 1 and Table 3).

On the Errera and Heawood graphs, randomization results in a higher rate of irrevocable Kempe chain tangles than the average rate of the original algorithm, but it is still within the range of the minimum and maximum failure rates of the original implementations. On the asymmetrical Poussin graph, our algorithm results in significantly more Kempe chain tangles than the average (Table 3), but this is mitigated by the success of the Kempe-Kittell chain switches in coloring the graph. We exclude the graphs of the platonic solids, as they cause no failures for either algorithm.

	Original Algorithm			Randomized Algorithm
	Min % Failure	Max % Failure	Avg % Failure	% Failure
Fritsch	3.598	14.131	11.790	3.60
Soifer	0.520	1.859	1.461	0.55
Poussin	0.014	0.627	0.228	1.18
Errera	3.350	10.866	7.881	9.28
Heawood	0.372	7.153	2.439	4.83

Table 3: **Comparison of original Algorithm Kempe to randomized version with Kempe-Kittell chains.**

4 Conclusions

We evaluated the performance of the version of Algorithm Kempe in [GS03] and its performance after the addition of Kempe-Kittell chain switches, which successfully overcame all irrevocable Kempe chain tangles in our benchmark graphs. We have proven that the order in which Kempe chain switches are performed affects the outcome of the algorithm and shown that the application of randomization to the selection of Kempe and Kempe-Kittell chain switches in this algorithm is a useful method for making the choice of which switch to perform first. In 500 test runs on each of ten benchmark graphs, the use of randomized chain choices resulted in successful four-coloring of the graphs with fewer than 12 random choices for any vertex most of the time. When compared to the original algorithm, there appears to be a performance trade-off in that randomization causes the use of *Gadget Kittell* in the Poussin graph more often than would have been required by the non-randomized version. Finally, we discovered that some vertices appear to be more likely to cause

irrevocable Kempe Chain tangles than others, and we identify those vertices in the hopes of being able to characterize them.

The following are open questions and future work:

1. (a) Given a planar graph G with e edges and n vertices, what is the minumum value of e for which Kempe's method is provably guaranteed to succeed in properly coloring G?

   (b) What percentage of all planar graphs on nine vertices serve as counterexamples to Kempe's method?

2. Do the vertices that cause irrevocable Kempe chain tangles or require high numbers of Kempe-Kittell chain switches share properties which can be exploited to improve Algorithm Kempe-Kittell?

3. Let G be a planar graph on n vertices. It follows from West [Wes01] (exercise 6.1.9) that, when $n \leq 11$, there is some labeling for which Algorithm Kempe succeeds in properly four-coloring G. What is the smallest value of $n > 11$ for which Algorithm Kempe-Kittell is provably guaranteed to succeed?

4. It is not difficult to show that the Icosahedral Graph will be properly 4-colored by Kempe's algorithm regardless of the labeling of the vertices (and this is confirmed by Table A in the appendix). Characterize all planar graphs that will be properly 4-colored by Kempe's algorithm under all possible orderings of the vertices. Short of that potentially difficult goal, find interesting families of planar graphs (with at least 11 vertices and whose average vertex degree is at least 5) for which Kempe's algorithm will always succeed.

5 Addendum

Stan Wagon reports the discovery that the planar graph corresponding to the contiguous 48 United States plus Lake Michigan and the oceanic waters admits a labeling that leads to a Kempe impasse at the great state of Illinois [Wag08]; for complete details, see [Wag].

References

[AH77a] Kenneth Appel and Wolfgang Haken. The solution of the four-color-map problem. *Sci. Amer.*, 237(4):108–121, 152, 1977.

[AH77b] Kenneth Appel and Wolfgang Haken. Supplement to: "Every planar map is four colorable. I. Discharging" (Illinois J. Math. **21** (1977), no. 3, 429–490) by Appel and Haken; "II. Reducibility" (ibid. **21** (1977), no. 3, 491–567) by Appel, Haken and J. Koch. *Illinois J. Math.*, 21(3):1–251. (microfiche supplement), 1977.

[AH77] Kenneth Appel and Wolfgang Haken. Every planar map is four colorable. *J. Recreational Math.*, 9(3):161–169, 1976/77.

[AHK77] Kenneth Appel, Wolfgang Haken, and John Koch. Every planar map is four colorable. Part II. Reducibility. *Illinois J. Math.*, 21:491–567, 1977.

[AS86] Raymond A. Archuleta and Henry D. Shapiro. A fast probabilistic algorithm for four-coloring large planar graphs. In *ACM '86: Proceedings of 1986 ACM Fall joint computer conference*, pages 595–600, Los Alamitos, CA, USA, 1986. IEEE Computer Society Press.

[BLW86] Norman L. Biggs, E. Keith Lloyd, and Robin J. Wilson. *Graph Theory. 1736–1936.* The Clarendon Press Oxford University Press, New York, 1986.

[Err21] Alfred Errera. *Du Colorage des Cartes et de Quelques d´Anaysis Situs.* PhD thesis, Gauthier-Villars, Paris, 1921.

[FF98] Rudolf Fritsch and Gerda Fritsch. *The Four-Color Theorem.* Springer-Verlag, New York, 1998.

[GS03] Ellen Gethner and William M. Springer, II. How false is Kempe's proof of the four color theorem? *Congr. Numer.*, 164:159–175, 2003.

[Hea90] P. J. Heawood. Map colour theorem. *Quart. J. Pure Appl. Math.*, 24:332–338, 1890.

[HW98] Joan P. Hutchinson and Stan Wagon. Kempe revisited. *Amer. Math. Monthly*, 105(2):170–174, 1998.

[Kit35] Irving Kittell. A group of operations on a partially colored map. *Bull. Amer. Math. Soc.*, 41:407–413, 1935.

[MS91] Craig A. Morgenstern and Henry D. Shapiro. Heuristics for rapidly four-coloring large planar graphs. *Algorithmica*, 6(6):869–891, 1991.

[Ore67] Oystein Ore. *The Four-Color Problem.* Pure and Applied Mathematics, Vol. 27. Academic Press, New York, 1967.

[RSST96] Neil Robertson, Daniel P. Sanders, Paul Seymour, and Robin Thomas. A new proof of the four-colour theorem. *Electron. Res. Announc. Amer. Math. Soc.*, 2(1):17–25 (electronic), 1996.

[RSST97] Neil Robertson, Daniel P. Sanders, Paul Seymour, and Robin Thomas. The four-colour theorem. *J. Combin. Theory Ser. B*, 70(1):2–44, 1997.

[Soi97] Alexander Soifer. Map coloring in the victorian age: Problems and history. *Mathematics Competitions*, 10:20–31, 1997.

[Wag] Stan Wagon. *Mathematica in Action, 3rd ed.* Springer/TELOS, New York. To appear.

[Wag02] Stan Wagon. A machine resolution of a four-color hoax. In *CCCG*, pages 174–185, 2002.

[Wag08] Stan Wagon. Personal communication. 2008.

[Wes01] Douglas B. West. *Introduction to graph theory.* Prentice Hall Inc., Upper Saddle River, NJ, second edition, 2001.

[Wil02a] Robert A. Wilson. *Graphs, Colourings and the Four-Colour Theorem.* Oxford University Press, Oxford, 2002.

[Wil02b] Robin J. Wilson. *Four Colors Suffice.* Princeton University Press, Princeton, NJ, 2002.

A Algorithm Kempe-Kittell Results

	Node Label	Kempe Kittell Chain Switches Required			Labelings / Trial	Colored Nodes	Kittell Colored Nodes	Kittel Use %
		Min	Max	Avg				
Fritsch	a	3	8	4.52	362880	181440000	1088000	0.599647
Fritsch	b	3	8	4.46	362880	181440000	1088000	0.599647
Fritsch	c	3	8	4.44	362880	181440000	1088000	0.599647
Fritsch	d	0	0	0.00	362880	181440000	0	0.0
Fritsch	e	0	0	0.00	362880	181440000	0	0.0
Fritsch	f	3	8	4.46	362880	181440000	1088000	0.599647
Fritsch	g	3	7	4.49	362880	181440000	1088000	0.599647
Fritsch	h	0	0	0.00	362880	181440000	0	0.0
Fritsch	i	3	9	4.50	362880	181440000	1088000	0.599647
Soifer	a	0	0	0.00	362880	181440000	0	0.0
Soifer	b	0	0	0.00	362880	181440000	0	0.0
Soifer	c	3	7	4.45	362880	181440000	988089	0.544582
Soifer	d	0	0	0.00	362880	181440000	0	0.0
Soifer	e	0	0	0.00	362880	181440000	0	0.0
Soifer	f	0	0	0.00	362880	181440000	0	0.0
Soifer	g	0	0	0.00	362880	181440000	0	0.0
Soifer	h	0	0	0.00	362880	181440000	0	0.0
Soifer	i	0	0	0.00	362880	181440000	0	0.0
Octahedron	a	0	0	0.00	720	360000	0	0.0
Octahedron	b	0	0	0.00	720	360000	0	0.0
Octahedron	c	0	0	0.00	720	360000	0	0.0
Octahedron	d	0	0	0.00	720	360000	0	0.0
Octahedron	e	0	0	0.00	720	360000	0	0.0
Octahedron	f	0	0	0.00	720	360000	0	0.0
Cube	a	0	0	0.00	40320	20160000	0	0.0
Cube	b	0	0	0.00	40320	20160000	0	0.0
Cube	c	0	0	0.00	40320	20160000	0	0.0
Cube	d	0	0	0.00	40320	20160000	0	0.0
Cube	e	0	0	0.00	40320	20160000	0	0.0

	Node Label	Kempe Kittell Chain Switches Required			Labelings / Trial	Colored Nodes	Kittell Colored Nodes	Kittel Use %
		Min	Max	Avg				
Cube	f	0	0	0.00	40320	20160000	0	0.0
Cube	g	0	0	0.00	40320	20160000	0	0.0
Cube	h	0	0	0.00	40320	20160000	0	0.0
Icosahedron	a	0	0	0.00	362880	181440000	0	0.0
Icosahedron	b	0	0	0.00	362880	181440000	0	0.0
Icosahedron	c	0	0	0.00	362880	181440000	0	0.0
Icosahedron	d	0	0	0.00	362880	181440000	0	0.0
Icosahedron	e	0	0	0.00	362880	181440000	0	0.0
Icosahedron	f	0	0	0.00	362880	181440000	0	0.0
Icosahedron	g	0	0	0.00	362880	181440000	0	0.0
Icosahedron	h	0	0	0.00	362880	181440000	0	0.0
Icosahedron	i	0	0	0.00	362880	181440000	0	0.0
Icosahedron	j	0	0	0.00	362880	181440000	0	0.0
Icosahedron	k	0	0	0.00	362880	181440000	0	0.0
Icosahedron	l	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	a	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	b	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	c	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	d	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	e	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	f	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	g	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	h	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	i	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	j	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	k	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	l	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	m	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	n	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	o	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	p	0	0	0.00	362880	181440000	0	0.0

	Node Label	Kempe Kittell Chain Switches Required			Labelings / Trial	Colored Nodes	Kittell Colored Nodes	Kittel Use %
		Min	Max	Avg				
Dodecahedron	q	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	r	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	s	0	0	0.00	362880	181440000	0	0.0
Dodecahedron	t	0	0	0.00	362880	181440000	0	0.0
Poussin	a	0	0	0.00	362880	181440000	0	0.0
Poussin	b	0	0	0.00	362880	181440000	0	0.0
Poussin	c	1	4	2.26	362880	181440000	526486	0.290171
Poussin	d	2	6	2.58	362880	181440000	399678	0.220281
Poussin	e	2	5	2.69	362880	181440000	323390	0.178235
Poussin	f	2	5	2.82	362880	181440000	339534	0.187133
Poussin	g	0	0	0.00	362880	181440000	0	0.0
Poussin	h	2	5	2.81	362880	181440000	474890	0.261734
Poussin	i	0	0	0.00	362880	181440000	0	0.0
Poussin	j	0	0	0.00	362880	181440000	0	0.0
Poussin	k	0	0	0.00	362880	181440000	0	0.0
Poussin	l	0	0	0.00	362880	181440000	0	0.0
Poussin	m	0	0	0.00	362880	181440000	0	0.0
Poussin	n	0	0	0.00	362880	181440000	0	0.0
Poussin	o	0	0	0.00	362880	181440000	0	0.0
Errera	a	35	73	47.01	362880	181440000	4937929	2.721522
Errera	b	3	8	4.14	362880	181440000	463228	0.255306
Errera	c	3	7	4.11	362880	181440000	463000	0.255181
Errera	d	3	7	4.05	362880	181440000	463086	0.255228
Errera	e	3	7	4.03	362880	181440000	464390	0.255947
Errera	f	2	5	2.83	362880	181440000	440437	0.242745
Errera	g	2	5	2.82	362880	181440000	440226	0.242629
Errera	h	2	5	2.80	362880	181440000	440679	0.242879
Errera	i	3	6	4.05	362880	181440000	464517	0.256017
Errera	j	2	6	2.85	362880	181440000	439756	0.24237
Errera	k	2	5	2.84	362880	181440000	440513	0.242787
Errera	l	3	7	4.07	362880	181440000	462705	0.255018

	Node Label	Kempe Kittell Chain Switches Required			Labelings / Trial	Colored Nodes	Kittell Colored Nodes	Kittel Use %
		Min	Max	Avg				
Errera	m	3	7	4.07	362880	181440000	462517	0.254915
Errera	n	3	7	4.10	362880	181440000	463978	0.25572
Errera	o	35	71	46.29	362880	181440000	4938719	2.721957
Errera	p	3	7	4.07	362880	181440000	463436	0.255421
Errera	q	3	8	4.06	362880	181440000	461792	0.254515
Heawood	a	2	6	3.22	362880	181440000	775216	0.427257
Heawood	b	2	6	3.29	362880	181440000	1217733	0.671149
Heawood	c	2	5	3.13	362880	181440000	847294	0.466983
Heawood	d	0	0	0.00	362880	181440000	0	0.0
Heawood	e	0	0	0.00	362880	181440000	0	0.0
Heawood	f	0	0	0.00	362880	181440000	0	0.0
Heawood	g	2	5	2.78	362880	181440000	690371	0.380495
Heawood	h	0	0	0.00	362880	181440000	0	0.0
Heawood	i	0	0	0.00	362880	181440000	0	0.0
Heawood	j	1	1	1.00	362880	181440000	6058	0.003339
Heawood	k	1	2	1.56	362880	181440000	6037	0.003327
Heawood	l	2	6	3.36	362880	181440000	907479	0.500154
Heawood	m	3	8	4.60	362880	181440000	1405876	0.774843
Heawood	n	3	7	3.75	362880	181440000	828677	0.456722
Heawood	o	2	5	3.15	362880	181440000	813156	0.448168
Heawood	p	2	5	3.19	362880	181440000	1032339	0.56897
Heawood	q	0	0	0.00	362880	181440000	0	0.0
Heawood	r	0	0	0.00	362880	181440000	0	0.0
Heawood	s	0	0	0.00	362880	181440000	0	0.0
Heawood	t	2	6	3.29	362880	181440000	92157	0.050792
Heawood	u	2	6	3.27	362880	181440000	91564	0.050465
Heawood	v	0	0	0.00	362880	181440000	0	0.0
Heawood	w	0	0	0.00	362880	181440000	0	0.0
Heawood	x	0	0	0.00	362880	181440000	0	0.0
Heawood	y	0	0	0.00	362880	181440000	0	0.0
Kittell	a	2	4	2.79	362880	181440000	636989	0.351074

	Node Label	Kempe Kittell Chain Switches Required			Labelings / Trial	Colored Nodes	Kittell Colored Nodes	Kittel Use %
		Min	Max	Avg				
Kittell	b	2	5	2.75	362880	181440000	478186	0.26355
Kittell	c	0	0	0.00	362880	181440000	0	0.0
Kittell	d	2	6	3.33	362880	181440000	675394	0.372241
Kittell	e	3	8	4.76	362880	181440000	475928	0.262306
Kittell	f	5	9	6.57	362880	181440000	792681	0.436883
Kittell	g	0	0	0.00	362880	181440000	0	0.0
Kittell	h	2	6	3.25	362880	181440000	82375	0.045401
Kittell	i	5	9	6.32	362880	181440000	782074	0.431037
Kittell	j	2	5	2.95	362880	181440000	457668	0.252242
Kittell	k	4	7	4.92	362880	181440000	400408	0.220683
Kittell	l	2	5	2.84	362880	181440000	473726	0.261092
Kittell	m	3	6	3.99	362880	181440000	386798	0.213182
Kittell	n	5	11	6.48	362880	181440000	537869	0.296445
Kittell	o	5	8	5.93	362880	181440000	496916	0.273873
Kittell	p	5	9	6.35	362880	181440000	691822	0.381295
Kittell	q	0	1	0.98	362880	181440000	1676	0.000924
Kittell	r	5	11	6.44	362880	181440000	667972	0.36815
Kittell	s	2	6	3.16	362880	181440000	660014	0.363764
Kittell	t	2	5	2.83	362880	181440000	489486	0.269778
Kittell	u	2	6	3.51	362880	181440000	209896	0.115683
Kittell	v	2	5	2.81	362880	181440000	551059	0.303714
Kittell	w	0	1	0.98	362880	181440000	1966	0.001084