Computing Chromatic Polynomials of Large Graphs I

Gary Haggard

Bucknell University
Lewisburg, Pennsylvania
U.S.A. 17837

Abstract. An efficient algorithm for calculating the chromatic polynomial of large graphs relative to the tree basis is presented. As an application of this algorithm, the degree thirty-two chromatic polynomial of the dual of the truncated icosahedron is calculated. Before this algorithm, only the by-hand calculations of Hall, Siry, and Vanderslice completed in 1965 had produced this chromatic polynomial.

In 1965 Hall, Siry, and Vanderslice [2] announced the result of an extensive manual calculation to determine the chromatic polynomial of the dual of the planar map of the truncated icosahedron (TI). They completed this computation over a period of several years. The work depended on the reduction of the original problem to the determination of the chromatic polynomial of each of a set of thirty-five graphs with twenty-six or fewer vertices [7]. The calculation involved the determination of a monic polynomial of degree thirty-two with maximum coefficient having magnitude of order 10^{19}. The whole computation stands as an amazing record of a successful, complex, manual computation. No algorithm of more than theoretical value had been developed to calculate such a large chromatic polynomial until the work described herein was completed. However, on a smaller scale, Read [5] developed an algorithm for calculating chromatic polynomials that he used in calculating the chromatic polynomial of all graphs with fewer than eleven vertices.

The starting point for the algorithm developed is the algorithm for computing chromatic polynomials given in [4]. This algorithm uses the delete-contract procedure repeatedly to reduce the graphs considered until each branch of the computation tree is reduced to the problem of computing the chromatic polynomial of a tree. Without substantial improvements, this algorithm is unable to deal effectively with graphs the size of the TI. Two major improvements to this algorithm are presented. The first introduces a new termination condition that relies on recognizing the graph being processed before it becomes a tree. The second generalizes classical reduction results to delete vertices of small degree.

Chromatic Polynomials

Let $G = (V, E)$ be a graph with $|V| = n$ where $n > 0$. As usual, denote the number of ways to color the vertices of G with λ colors as $P(G, \lambda)$ where $\lambda = 0, 1, 2, \ldots$. The well known properties of chromatic polynomials that will be used are found in [3] and [6].

JCMCC 13 (1993), pp. 175-186

Definition 1 (Delete-contract graphs): Let $U \subseteq E$. The deletion graph $G - U$ is defined as the graph with vertex set V and edge set $E - U$. If U is a single edge e, this graph is denoted as $G - e$. For $v \in V$, the graph $G - v$ will denote the graph with vertex set $V - \{v\}$ and edge set consisting of all the edges of G that are not incident to v. Let $W \subseteq V$. The contraction graph G/W is defined to be the graph generated from G by identifying the vertices in W as a single vertex. If $W = \{v, w\}$ and $e = (v, w) \in E$, the contraction, denoted as G/e denotes the graph $(G - \{(v, w)\})/\{v, w\}$ that first deletes e and then contracts the vertices that were its ends.

Much of the current literature only considers the operations $G - e$ and G/e. For the work here, the more general notions of deletion and contraction are necessary.

It will be enough to consider only connected graphs in this paper. The problem of calculating the chromatic polynomial of any graph only requires the multiplication of chromatic polynomials for the connected components of a graph. The algorithms presented here for computing chromatic polynomials do require and preserve the property that the graph is connected.

The next two well known results about chromatic polynomials are needed later.

Proposition 1. *If two graphs G_1 and G_2 intersect in a complete graph on k vertices where $k \geq 1$, then*

$$P(G_1 \cup G_2, \lambda) = \frac{P(G_1, \lambda) \cdot P(G_2, \lambda)}{\lambda(\lambda - 1) \cdots (\lambda - k + 1)}.$$

Proposition 2. *Let T be any tree with k vertices where $k \geq 1$, then the chromatic polynomial of T is*

$$P(T, \lambda) = \lambda(\lambda - 1)^{k-1}.$$

A Basic Algorithm

The algorithm described in [4] for calculating chromatic polynomials is a starting point. A version of that algorithm is given in Table I.

In this algorithm the edge to be deleted is not in the spanning tree. Deleting such an edge insures that the graph will remain connected and progress will be made towards transforming G into a tree. When forming H_2, the edges designated as a spanning tree in H_1 become a subgraph of H_2 with one cycle. To form a spanning tree in H_2, designate one of the edges in that cycle as not in the spanning tree. When the algorithm terminates, the chromatic polynomial of G is given by

$$P(G, \lambda) = \sum_{i=0}^{|V|-1} (-1)^i CT[i + 1] \lambda(\lambda - 1)^i$$

Algorithm 1

INPUT: Connected graph $G = (V, E)$ with $|V| > 0$.
RESULT: Coefficients of the chromatic polynomial of G relative the tree basis.
Data Structures: Array $CT[1..|V|]$ initialized to have all its entries zero. $CT[i]$ will contain the number of trees with i vertices that terminated branches of the computation for $1 \leq i \leq |V|$. A stack S (last-in-first-out list) to hold graphs generated by the contract operation.

```
find a spanning tree T in G
stop-condition:=false
do
    if G is a tree on i vertices then
      increment CT[i] by 1
      if S ≠ ∅ then
        G =POP(S)
      else
        stop-condition:=true
    else
      H1 = G − e where e ∉ T
      H2 = G/e
      form a spanning tree in H2
      PUSH(H2)
      G = H1
until (stop-condition = true)
```

Table I

when $|V|$ is odd and

$$P(G,\lambda) = \sum_{i=0}^{|V|-1} (-1)^{i+1} CT[i+1]\lambda(\lambda-1)^i$$

when $|V|$ is even. The computation time for calculating a chromatic polynomial using **Algorithm 1** will be proportional to the sum of the entries in CT since this sum represents the number of trees that terminate branches of the computation tree.

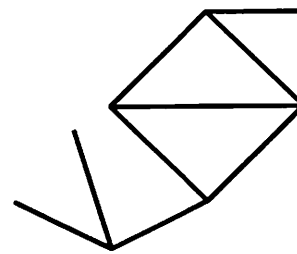
The terms

$$\lambda, \lambda(\lambda-1), \lambda(\lambda-1)^2, \ldots, \lambda(\lambda-1)^{|V|-1}$$

are called the *tree basis* for the chromatic polynomial. The improvements to **Algorithm 1** speed the computation of the coefficients for a chromatic polynomial relative the tree basis.

Small graphs

Rather than wait for G to become a tree to terminate a branch of the computation, it is possible to make a contribution to the final answer when a "small" graph is recognized. It is not necessary that every graph with n vertices that is used to terminate a branch of the computation have the same chromatic polynomial. What is necessary, however, is that every graph that causes termination of a branch of the computation tree be easy to recognize and to have a known chromatic polynomial. One problem that stands in the way of the implementation of such a termination condition is the generation of tree-like subgraphs as a result of deleting edges. For example, a graph of the form shown in Fig. 1 could easily result from the delete-contract process.



Tree-like subgraphs
Figure 1

Clearly the "core" graph in Fig. 1 is the graph without the "trees attached." The "core" graph, however, will not be recognized because of the tree-like subgraphs attached to it. The key to the solution of this problem is found in Theorem 1.

Theorem 1. *Let* $G = (V, E)$ *be a graph. Let* $e = (v, w) \in E$ *such that* $\deg(v) = 1$. *The chromatic polynomial of* G *is*

$$P(G, \lambda) = (\lambda - 1)P(G - v, \lambda).$$

Proof: The result follows from Proposition 1 with $k = 1$. ∎

It is instructive to interpret Theorem 1 in terms of the tree basis. Let $v \in V$ such that $\deg(v) = 1$. Let $CT_1[1..|V| - 1]$ be the coefficients of the chromatic polynomial of $G - v$ relative the tree basis. Let $CT[1..|V|]$ be the coefficients of the chromatic polynomial of G relative the tree basis. The entries in CT_1 and CT are related as indicated below.

$$CT[i] = \begin{cases} 0 & i = 1 \\ CT_1[i-1] & i = 2, 3, \ldots, |V|. \end{cases}$$

Thus, a vertex of degree one can have its contribution to a chromatic polynomial incorporated into the tree basis representation by means of a simple shift operation. The number of times this simple shift operation needs to be applied to remove the tree-like subgraphs of a graph will be the number of edges that must be removed to leave a graph with minimum degree two.

By checking at each step of the computation for the presence of vertices of degree one, vertices of degree one can be eliminated as they are generated. The algorithm must make sure to pass the number of degree one vertices deleted to all the nodes that are descendants of a given node. When the contribution of a graph to the chromatic polynomial of the original graph is made, the total number of degree one vertices deleted in processing that graph will determine the number of shifting operations needed. By removing the tree-like subgraphs and forming a graph with minimum degree two, the termination of a branch of the computation becomes the problem of trying to identify what graph remains. Using the table in Harary [3], a procedure is developed that causes the algorithm to terminate as soon as any one of the seventy-six, connected graphs with six or fewer vertices and minimum degree two is encountered. In Table II, **Algorithm 1** is modified to incorporate this new termination condition.

Vertex reduction

The classical results about the calculation of chromatic polynomials [1] use reduction theorems to eliminate regions with a small number of bounding edges (small degree in the planar dual for planar graphs). Since chromatic polynomials can be calculated for both planar and non-planar graphs, there is no need to restrict reductions to planar graphs. The more general reduction theorems are given for certain vertices of degree two, three, four, five, and six. For these theorems the graph need not be connected.

Theorem 2. *Let* $G = (V, E)$ *be a graph. Let* $v \in V$ *such that* $\deg(v) = 2$ *and the neighbors of* v *are adjacent. The chromatic polynomial of* G *is*

$$P(G, \lambda) = (\lambda - 2)P(G - v, \lambda).$$

Proof: Use Proposition 1 with $k = 2$. ∎

To incorporate this result into **Algorithm 2**, the actual contribution to $P(G, \lambda)$ is made using a two step process. The procedure first represents the computation as:

$$(\lambda - 2)P(G - v, \lambda) = (\lambda - 1)P(G - v, \lambda) - P(G - v, \lambda).$$

Since $P(G - v, \lambda)$ is represented relative the tree basis, the first calculation represents a shifting right one position of each element in $CT[*]$. The second calculation is accomplished by adding $P(G - v, \lambda)$ to the polynomial that results

Algorithm 2

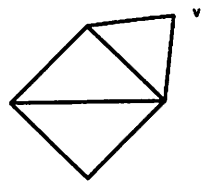
INPUT: Connected graph $G = (V,E)$ with $|V| > 0$.
RESULT: Coefficients of the chromatic polynomial of G relative the tree basis.
Data Structures: Array $CT[1..|V|]$ initialized to have all its entries zero. $CT[i]$ will contain the coefficients of $\lambda(\lambda - 1)^{i-1}$ in the chromatic polynomial for G for $1 \leq i \leq |V|$. A stack S (last-in-first-out list) to hold graphs generated by the contract operation.

```
find a spanning tree T in G
stop-condition:=false
do
    if G is a tree on i vertices or |V| ≤ 6 then
      contribute to CT[*]
      if S ≠ ∅ then
        G =POP(S)
      else
        stop-condition:=true
    else
      H1 = G − e where e ∉ T
      H2 = G/e
      form a spanning tree in H2
      PUSH(H2)
      G = H1
until (stop-condition = true)
```

Table II

from the shifting operation. The signs are actually unnecessary during the computation because all the contributions to a particular final coefficient will have the same sign. The actual plus and minus signs can be attached to the final polynomial when it is output. An example of this shift and add computation is shown in Fig. 2.

The reductions for vertices of degree greater than two are a bit more complex because they not only reduce the graph but also, in some cases, generate additional graph(s) that must be put on the stack so that the(ir) chromatic polynomial(s) can be calculated later.



$$\begin{aligned}
P(G,\lambda) &= (\lambda-2)P(G-v,\lambda) \\
&= (\lambda-1)P(G-v,\lambda) - P(G-v,\lambda) \\
&= (\lambda-1)(\lambda(\lambda-1) - 2\lambda(\lambda-1)^2 + \lambda(\lambda-1)^3) \\
&\quad - (\lambda(\lambda-1) - 2\lambda(\lambda-1)^2 + \lambda(\lambda-1)^3) \\
&= -\lambda(\lambda-1) + 3\lambda(\lambda-1)^2 - 3\lambda(\lambda-1)^3 + \lambda(\lambda-1)^4
\end{aligned}$$

$\lambda - 2$ computation
Figure 2

Theorem 3. *Let* $G = (V,E)$ *be a graph. Let* $v, w_1, w_2, w_3 \in V$ *such that* $\deg(v) = 3$ *and the neighbors of* v *are* w_1, w_2, *and* w_3. *Suppose that the neighbors of* v *are contained on a path of length two,* w_1, e_1, w_2, e_2, w_3 *where* $e_1, e_2 \in E$. *Then*

$$P(G,\lambda) = (\lambda-3)P(G-v,\lambda) + P((G-v)/\{w_1,w_3\},\lambda).$$

Corollary. *If* $(w_1,w_3) \in E$, *then*

$$P(G,\lambda) = (\lambda-3)P(G-v,\lambda).$$

When incorporating the reduction of Theorem 3 into the algorithm, the graph $(G-v)/\{w_1,w_3\}$ is put on the stack for later use and the graph $G-v$ is used in the next step of the computation. The factor $(\lambda-3)$ is incorporated into the final value of $P(G,\lambda)$ as follows:

$$(\lambda-3)P(G-v,\lambda) = (\lambda-1)P(G-v,\lambda) - P(G-v,\lambda) - P(G-v,\lambda).$$

This representation of $(\lambda-3)P(G-v,\lambda)$ indicates that the computation consists of a shift operation followed by two add operations.

The reduction of a vertex of degree four generates up to three additional graphs to be put on the stack. In general, the number of graphs to be put on the stack depends on the structure of the subgraph induced by the neighbors of the vertex that is being deleted.

Theorem 4. *Let* $G = (V,E)$ *be a connected graph. Let* $v, w_1, w_2, w_3, w_4 \in V$ *such that* $\deg(v) = 4$ *and the neighbors of* v *are* w_1, w_2, w_3, *and* w_4. *Suppose*

that the neighbors of v *are contained on a path of length three,* $w_1, e_1, w_2, e_2, w_3, e_3, w_4$, *where* $e_1, e_2, e_3 \in E$. *Then*

$$P(G,\lambda) = (\lambda - 4)P(G - v,\lambda) + P((G - v)/\{w_1, w_3\},\lambda) + P((G - v)/\{w_1, w_4\},\lambda) + P((G - v)/\{w_2, w_4\},\lambda).$$

Again, the result can be simplified if the subgraph induced by $\{w_1, w_2, w_3, w_4\}$ contains any of the edges (w_1, w_3), (w_1, w_4), and (w_2, w_4).

Reductions can also be formulated for vertices of degree five.

Theorem 5. *Let* $G = (V, E)$ *be a graph. Let* $v, w_1, w_2, w_3, w_4, w_5 \in V$ *such that* $\deg(v) = 5$ *and the neighbors of* v *are* w_1, w_2, w_3, w_4, *and* w_5. *Suppose that the neighbors of* v *are contained on a path of length four,*

$$w_1, e_1, w_2, e_2, w_3, e_3, w_4, e_4, w_5,$$

where $e_1, e_2, e_3, e_4 \in E$. *Then*

$$\begin{aligned} P(G,\lambda) = {} & (\lambda - 5)P(G - v,\lambda) + P((G - v)/\{w_1, w_3\},\lambda) \\ & + P((G - v)/\{w_1, w_4\},\lambda) + P((G - v)/\{w_1, w_5\},\lambda) \\ & + P((G - v)/\{w_2, w_4\},\lambda) + P((G - v)/\{w_2, w_5\},\lambda) \\ & + P((G - v)/\{w_3, w_5\},\lambda) - P((G - v)/\{w_1, w_3, w_5\},\lambda). \end{aligned}$$

The reduction for a vertex of degree six is presented because it is useful in making a reduction to the task of finding the chromatic polynomial of the TI. In general, the reduction of a vertex of degree six causes the stacking of so many graphs of about the same size as the original graph that it may take longer to calculate a chromatic polynomial using the reduction than without using it. Even the reduction for a vertex of degree five may not always be a useful reduction.

Theorem 6. *Let* $G = (V, E)$ *be a graph. Let* $v, w_1, w_2, w_3, w_4, w_5, w_6 \in V$ *such that* $\deg(v) = 6$ *and the neighbors of* v *are* w_1, w_2, w_3, w_4, w_5, *and* w_6. *Suppose that the neighbors of* v *are contained on a path of length five,*

$$w_1, e_1, w_2, e_2, w_3, e_3, w_4, e_4, w_5, e_5, w_6,$$

where $e_1, e_2, e_3, e_4, e_5 \in E$. *Then*

$$\begin{aligned} P(G,\lambda) = {} & (\lambda - 6)P(G - v,\lambda) + P((G - v)/\{w_1, w_3\},\lambda) \\ & + P((G - v)/\{w_1, w_4\} + P((G - v)/\{w_1, w_5\},\lambda) \\ & + P((G - v)/\{w_1, w_6\},\lambda) + P((G - v)/\{w_2, w_4\},\lambda) \\ & + P((G - v)/\{w_2, w_5\},\lambda) + P((G - v)/\{w_2, w_6\},\lambda) \\ & + P((G - v)/\{w_3, w_5\},\lambda) + P((G - v)/\{w_3, w_6\},\lambda) \\ & + P((G - v)/\{w_4, w_6\},\lambda) - P((G - v)/\{w_1, w_3, w_5\},\lambda) \\ & - P((G - v)/\{w_1, w_3, w_6\},\lambda) - P((G - v)/\{w_1, w_4, w_6\},\lambda) \\ & - P((G - v)/\{w_2, w_4, w_6\},\lambda) \end{aligned}$$

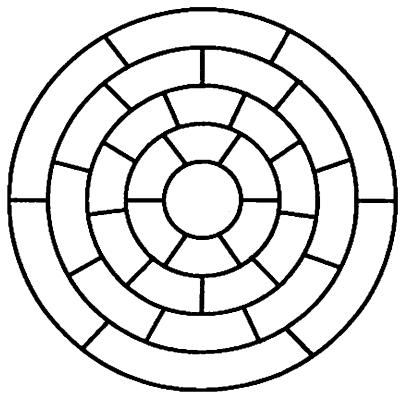
The condition that the neighbors of the vertex to be deleted must be on a path makes it easy to prove that the graph resulting from the deletion will be connected. It is a well known result of graph theory that the removal of an edge of a connected graph disconnects the graph if and only if the edge is not contained in a circuit. A problem of ongoing interest is how to generate other efficient reductions in the size of the problem while maintaining the property of connectedness. The path condition imposed clearly resulted in an efficient reduction process.

Overflow

Each vertex reduction results in the determination of a factor of the chromatic polynomial of the form $\lambda - k$ where k is the degree of the vertex removed. To "multiply" the chromatic polynomial of the resulting graph by these factors requires shifting and adding operations applied to $CT[*]$. The shift and add operations must be monitored carefully so that no overflow results. For the *TI* this is a particularly important consideration because the coefficients of its chromatic polynomial relative the tree basis vary in size up to 10^{17}.

Truncated Icosahedron

Incorporating the reduction of vertices of small degree into **Algorithm 2** gives rise to **Algorithm 3** shown in Table III. Figure 3 shows the planar map of the truncated icosahedron.



The truncated icosahedron
Figure 3

The "contribute to CT" step involves the shift and add process with arbitrary precision for each of the reductions applied.

Figure 4 gives the chromatic polynomial of the truncated icosahedron relative the tree basis as well as relative the standard basis.

Algorithm 3

INPUT: Connected graph $G = (V, E)$ with $|V| > 0$.
RESULT: Coefficients of the chromatic polynomial of G relative the tree basis.
Data Structures: Array $CT[1..|V|]$ initialized to have all its entries zero. $CT[i]$ will contain the coefficient of $\lambda(\lambda - 1)^{i-1}$ in the chromatic polynomial for G for $1 \leq i \leq |V|$. A stack S (last-in-first-out list) to hold graphs generated by the contract operation.

```
find a spanning tree T in G
stop-condition:=false
do
    remove vertices of small degree from G
    if G is a tree on i vertices or |V| ≤ 6 then
      contribute to CT[*]
           (include contributions from deleted vertices)
      if S ≠ ∅ then
        G =POP(S)
      else
        stop-condition:=true
    else
      H1 = G − e where e ∉ T
      H2 = G/e
      form a spanning tree in H2
      PUSH(H2)
      G = H1
until (stop-condition = true)
```

Table III

A shortcut to the *TI*

Substantial reduction in run time for a direct calculation of the chromatic polynomial of the TI can be achieved by applying a reduction process to the vertex of degree six that represents the "outside" region of the planar map representing the truncated icosahedron. The reduction of this degree six vertex yields the chromatic polynomial of the TI in terms of twelve graphs with fewer vertices than the TI. The reduction of the set of fifteen graphs that result from the reduction of a vertex of degree six to just twelve graphs result from the presence of the edge

$P(TI,\lambda) =$	
+1	$\lambda(\lambda-1)^{31}$
−59	$\lambda(\lambda-1)^{30}$
+1710	$\lambda(\lambda-1)^{29}$
−32450	$\lambda(\lambda-1)^{28}$
+453403	$\lambda(\lambda-1)^{27}$
−4973197	$\lambda(\lambda-1)^{26}$
+44583400	$\lambda(\lambda-1)^{25}$
−335797000	$\lambda(\lambda-1)^{24}$
+2167755011	$\lambda(\lambda-1)^{23}$
−12175221281	$\lambda(\lambda-1)^{22}$
+60183865818	$\lambda(\lambda-1)^{21}$
−264193106900	$\lambda(\lambda-1)^{20}$
+1037166520075	$\lambda(\lambda-1)^{19}$
−3661052024851	$\lambda(\lambda-1)^{18}$
+11665962055746	$\lambda(\lambda-1)^{17}$
−33646105485699	$\lambda(\lambda-1)^{16}$
+87942380056419	$\lambda(\lambda-1)^{15}$
−208269347521537	$\lambda(\lambda-1)^{14}$
+446095183473635	$\lambda(\lambda-1)^{13}$
−861047769728938	$\lambda(\lambda-1)^{12}$
+1489229660477423	$\lambda(\lambda-1)^{11}$
−2289676883015802	$\lambda(\lambda-1)^{10}$
+3096554498357805	$\lambda(\lambda-1)^{9}$
−3633794047947863	$\lambda(\lambda-1)^{8}$
+3636161286009548	$\lambda(\lambda-1)^{7}$
−3033198826142253	$\lambda(\lambda-1)^{6}$
+2046265622980682	$\lambda(\lambda-1)^{5}$
−1069414679790850	$\lambda(\lambda-1)^{4}$
+405031279488050	$\lambda(\lambda-1)^{3}$
−98611241495930	$\lambda(\lambda-1)^{2}$
+11551226205884	$\lambda(\lambda-1)$

(a) Tree basis

$P(TI,\lambda) =$	
+1	λ^{32}
−90	λ^{31}
+3945	λ^{30}
−112200	λ^{29}
+2327268	λ^{28}
−37516324	λ^{27}
+489095520	λ^{26}
−5298021900	λ^{25}
+48618908986	λ^{24}
−383467527324	λ^{23}
+2628112750438	λ^{22}
−15783975098870	λ^{21}
+83613089708150	λ^{20}
−392625990680270	λ^{19}
+1640349520092620	λ^{18}
−6113181533318121	λ^{17}
+20353936625560620	λ^{16}
−60579703683784392	λ^{15}
+161106440132538877	λ^{14}
−382247148887646201	λ^{13}
+806916662571437272	λ^{12}
−1509208695276387615	λ^{11}
+2486158497706802584	λ^{10}
−3577950146822172066	λ^{9}
+4449317065382987055	λ^{8}
−4710499040596052917	λ^{7}
+4160720925588200156	λ^{6}
−2980545888313780582	λ^{5}
+1661171405749474098	λ^{4}
−674454446771952908	λ^{3}
+176998670388733410	λ^{2}
−22463193324569220	λ

(b) Standard basis

Chromatic polynomial of the truncated icosahedron
Figure 4

(w_1, w_6) where w_1 and w_6 are the ends of the path of length five that joins the vertices adjacent to the vertex being removed. It can be shown that only seven

of these graphs are isomorphically distinct. Consequently, the computation can be reduced to the calculation of the chromatic polynomials of these seven graphs followed by the necessary linear combination of their chromatic polynomials to determine the chromatic polynomial of the truncated icosahedron(TI).

Open questions

An investigation of how to eliminate putting the same graph on the stack more than once is needed, but probably very hard. A related problem involves finding a way to identify subgraphs, not necessarily "small" subgraphs, for which the chromatic polynomial has been calculated so that the graph need not be put on the stack.

References

1. G.D. Birkhoff and D.C. Lewis, *Chromatic Polynomials*, Proceedings of the American Mathematical Society **16** (1965), 355–451.
2. D.W. Hall, J.W. Siry, and B.R. Vanderslice, *The Chromatic Polynomial of the Truncated Icosahedron*, Proceedings of the American Mathematical Society **16** (1965), 620–628.
3. F. Harary, "Graph Theory", Addison-Wesley Publ. Co., Reading, MA, 1969.
4. A. Nijenhuis and H. Wilf, "Combinatorial Algorithms", Academic Press, New York, 1975.
5. R. Read, *An improved method for computing the chromatic polynomial of sparse graphs*, Research Report CORR 87-20, University of Waterloo (1987).
6. R. Read, *An Introduction to Chromatic Polynomials*, Journal of Combinatorial Theory **4** (1968), 52–71.
7. J. Siry, *Chromatic Polynomials of Large Maps*, Ph.D. Thesis, University of Maryland (1953).