INTERNATIONAL JOURNAL OF COMPUTATIONAL MATHEMATICAL IDEAS Vol. 1, No. 1 (2009), pp. 29–37
ISSN (online) • *www.ijcmi.webs.com* *Research Article*

A Programming Approach to Planar Coloured Graphs

Colouring Algorithms Inspired by the Four Colour Theorem

Satyanarayana Sanakkayala
Research Scholar, Graph Machine Learning
School of Science and Technology, Dravidian University, Kuppam, India

Received: 01 January 2009 • Accepted: 01 May 2009 • Published: 20 May 2009

Abstract. The proof of the four colour theorem is a landmark of computational mathematics, and its machinery, degeneracy, Kempe chains, discharging, reducibility, and unavoidable sets, is a treasury of algorithmic ideas. This paper, the third in a series on learning and computing with planar graphs, takes a programming approach: it turns the proof strategies of the four colour theorem into concrete, implementable procedures for colouring planar coloured graphs, and it supplies the proper colourings that the learning algorithms of the companion papers consume. We develop four programs with increasing ambition. A degeneracy program colours any planar graph with at most six colours in linear time by peeling vertices of degree at most five. A Kempe chain program improves this to five colours in linear time by recolouring along alternating two colour paths, using planarity to rule out the only obstruction. A discharging and reducibility program uses the charge that Euler's formula distributes over a triangulation as a colouring priority and prunes small reducible configurations. A constraint search program performs saturation ordered backtracking with forward checking to reach the chromatic number when a minimum colouring is required, falling back on the always available four colouring guaranteed by the theorem. Each program is given with correctness and complexity, the intuition of every proof idea is made explicit, and the behaviour is illustrated on planar test graphs. The message is that the four colour theorem is not only a theorem to be admired but a design manual to be programmed.

Keywords: planar graph colouring; four colour theorem; degeneracy ordering; Kempe chains; discharging; reducibility; saturation degree; backtracking search.

2000 MSC: 05C15; 68R10; 68W05; 05C85; 68T20.

1. Introduction

The four colour theorem states that the vertices of any planar graph can be properly coloured with four colours, and its proof by Appel and Haken was the first major theorem established with essential help from a computer [7]. What is sometimes overlooked is how much of that proof is genuinely algorithmic. Every step, from peeling a low degree vertex, to swapping colours along a two coloured path, to shifting a numerical charge across a triangulation, to checking that a small configuration can always be extended, is a computation waiting to be programmed. This paper takes that view seriously and reads the theorem as a design manual for colouring planar coloured graphs.

The motivation is not only aesthetic. The companion papers of this series develop machine learning on planar graphs [18] and on planar coloured graphs [19], and the second of these relies on a proper four colouring to schedule its parallel updates. Somebody must produce that colouring, cheaply and reliably, and the present paper is that somebody. We therefore treat colouring as a first class computational task and build a small library of programs whose correctness rests on classical planar facts.

Four programs are developed, in order of ambition. Section 2 recalls the planar facts and the anatomy of the proof. Section 3 gives a linear time six colouring by degeneracy ordering. Section 4 improves it to a linear time five colouring by Kempe chains. Section 5 turns discharging and reducibility into a colouring priority and a pruning rule. Section 6 performs constraint search for a minimum colouring when one is demanded. Section 7 assembles the programs into a pipeline that feeds the learning stage, Section 8 summarises cost, Section 9 surveys applications, Section 10 reports illustrative runs, and Section 11 concludes. Background on graphs and colouring is standard [17], and the complexity vocabulary is that of Garey and Johnson [11].

2. Planar Facts and the Anatomy of the Proof

Let $G = (V, E)$ be a simple planar graph on n vertices given with a plane embedding, tested in linear time if necessary [5]. A proper colouring is a map $\varphi : V \to \{1, \ldots, k\}$ with $\varphi(i) \neq \varphi(j)$ whenever $\{i, j\} \in E$. The engine of every program below is the sparsity of planar graphs.

Proposition 1 (A thin vertex always exists)**.** *Every planar graph has a vertex of degree at most five, and more strongly every subgraph of a planar graph does, since subgraphs of planar graphs are planar. Equivalently, planar graphs are five degenerate.*

Proof. A simple planar graph has at most $3n - 6$ edges, so the average degree is below six, and some vertex has degree at most five. Every subgraph is planar and inherits the bound. □

Proposition 1 is the seed of the whole paper. It is the reason the degeneracy program terminates with six colours, the reason the Kempe program only ever has to fix a vertex of degree at most five, and the reason discharging begins by counting the surplus of thin vertices. The proof of the four colour theorem is built from four ideas, and it helps to name them once before programming them.

Degeneracy. Colour last what you removed first: peel thin vertices, then colour in reverse, so each vertex meets few coloured neighbours [14].

Kempe chains. A maximal connected subgraph in two colours can have its two colours swapped without breaking properness, which frees a colour where it is needed [1].

Discharging. Assign a numerical charge to vertices so that Euler's formula forces the total to be positive, then move charge by fixed rules until some configuration is forced to appear [7].

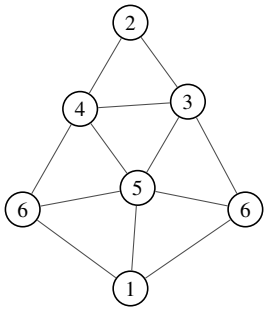
Reducibility. A configuration is reducible if every colouring of its boundary extends inward, so a graph avoiding all irreducible configurations is colourable [8].

Heawood used the Kempe chain idea correctly to prove the five colour theorem after Kempe's own four colour argument was found wanting [2], and a later reworking of the discharging proof reduced the case analysis and yielded a quadratic time four colouring [15]. We now program these ideas.

3. The Degeneracy Program: Six Colours in Linear Time

The simplest program colours a planar graph with six colours and never backtracks. It orders the vertices by *smallest last*: repeatedly remove a vertex of minimum current degree and record the removal order, then colour the vertices in the reverse of that order [14].

Theorem 1 (Six colouring)**.** *Smallest last ordering colours a planar graph with at most six colours, and it runs in $O(n)$ time and space.*



removal order (labels):
peel min degree first
colour in reverse order
each vertex sees ≤ 5
coloured neighbours

Figure 1: Smallest last degeneracy ordering. Each label is the step at which the vertex is removed, always at minimum current degree. Colouring in the reverse order confronts every vertex with at most five already coloured neighbours, so six colours suffice, in linear time.

Proof. When a vertex v is removed it has degree at most five in the remaining graph, by Proposition 1 applied to that planar subgraph. In the reverse colouring order the already coloured neighbours of v are exactly the ones that remained when v was removed, so there are at most five of them, and one of six colours is always free. A bucket queue keyed by current degree performs each removal in constant amortised time, giving linear total cost. □

The intuition is that a sparse graph can always be disassembled from its thinnest point, and reassembling it in the same order never confronts a vertex with more than five committed neighbours. The program respects any given input colours by treating them as fixed constraints on the affected vertices.

Algorithm 1: Degeneracy Program (six colouring)

Input: Planar graph $G = (V, E)$, optional fixed input colours
Output: A proper colouring with at most six colours

```
Initialise a bucket queue of vertices keyed by current degree
for t ← n downto 1 do
    Remove a vertex v of minimum current degree; set order[t] ← v
    Decrement the degree of each remaining neighbour in the queue
for t ← 1 to n do
    v ← order[t]
    Assign φ(v) the smallest colour not used by its already coloured neighbours
return φ
```

Cost. $O(n)$ time and space by the bucket queue and planar sparsity.

4. The Kempe Chain Program: Five Colours in Linear Time

To save the sixth colour we recolour. Suppose the vertices are processed in the smallest last order and a vertex v of degree at most five is reached. If its neighbours use at most four colours, a fifth is free and we are done. The only hard case is a vertex of degree exactly five whose five neighbours carry five distinct colours.

4.1. The Kempe swap and why planarity saves it

Arrange the five neighbours by their cyclic order in the plane embedding as $u_1, \ldots, u_5$ with colours $1, \ldots, 5$. Consider two neighbours of non adjacent colours, say u_1 coloured 1 and u_3 coloured 3, and look at the *Kempe chain*: the maximal connected subgraph induced by colours 1 and 3 that contains u_1. Two outcomes are possible.

If u_3 lies outside this chain, then swapping colours 1 and 3 throughout the chain keeps the colouring proper, recolours u_1 from 1 to 3, and leaves colour 1 unused among the neighbours of v, which we then give to v.

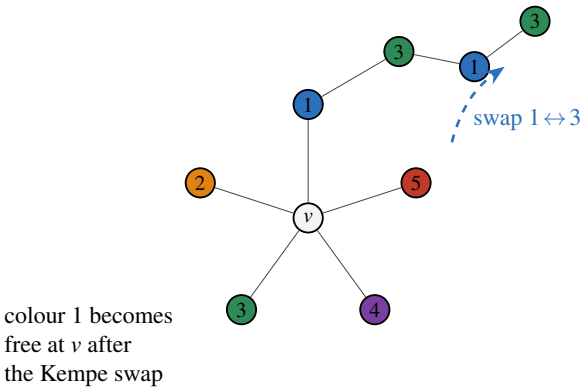


Figure 2: The Kempe chain step. The centre vertex v has five neighbours in the five distinct colours, so no colour is directly free. Swapping the two colours of the $(1,3)$ Kempe chain hanging off u_1 recolours it and frees colour 1 for v. Planarity guarantees that if this swap is blocked, the disjoint $(2,4)$ swap succeeds, because two such chains cannot cross in the plane.

If instead u_3 lies in the same chain, a path coloured $1,3,1,3,\ldots$ runs from u_1 to u_3. Together with v this path encloses either u_2 or the pair u_4,u_5. Now examine the $(2,4)$ chain from u_2. Because the first chain separates the plane, the second chain cannot reach u_4 without crossing it, which is impossible in a plane embedding, so the $(2,4)$ swap is guaranteed to succeed. Exactly one of the two swaps always works, and this is the whole content of the five colour theorem [2]. Chiba, Nishizeki and Saito organised these swaps so that the total recolouring over the whole run is linear [13].

Algorithm 2: Kempe Chain Program (five colouring)

Input: Planar graph G in smallest last order
Output: A proper colouring with at most five colours

```
for v in reverse smallest last order do
    if the coloured neighbours of v use at most four colours then
        Give v a free colour
    else
        Let the five neighbours have colours 1,...,5 in cyclic order
        Try the (1,3) Kempe chain from u1; if u3 is outside it, swap and free colour 1
        else
            swap the (2,4) Kempe chain from u2, which planarity guarantees to succeed, and free
              colour 2
        Give v the freed colour
return φ
```

Cost. $O(n)$ overall with the amortised chain bookkeeping of Chiba, Nishizeki and Saito [13].

5. Discharging and Reducibility as Programming Rules

The last colour, from five down to the theorem's four, is where the proof stops being a light recolouring and becomes a global argument. We do not reprove the theorem; we borrow its two devices as programming heuristics that make an exact search fast.

5.1. Charge as a colouring priority

Triangulate the graph and assign to each vertex the *charge* $\mu(v) = 6 - \deg(v)$. Euler's formula fixes the total.

Lemma 1 (Total charge)**.** *For a planar triangulation,* $\sum_v \big(6 - \deg(v)\big) = 12$.

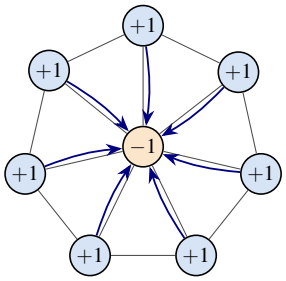


Figure 3: Discharging on a triangulated patch. Vertices carry charge $6-\deg$, positive on thin degree five vertices and negative on the dense degree seven hub. A discharging rule moves charge inward; because the total is fixed at twelve by Euler's formula, some configuration must retain positive charge and is therefore unavoidable. As a program, the charge is read as a colouring priority rather than as a proof.

Proof. A triangulation has $m = 3n-6$ edges, so $\sum_v \deg(v) = 2m = 6n-12$, and $\sum_v (6-\deg(v)) = 6n-(6n-12) = 12$. □

The positive total means low degree vertices carry the surplus, and *discharging* moves charge from these thin vertices to their dense neighbours by fixed rules until some small configuration is forced to keep positive charge and therefore to appear. For programming we do not need the full unavoidable set; we need only the ranking it suggests. High charge marks a locally sparse, easily coloured region, and low or negative charge marks a dense region that constrains its surroundings. The program colours dense regions first, when the palette is still open, and thin regions last, when a free colour is almost certain.

5.2. Reducibility as a pruning rule

A configuration is *reducible* if every proper colouring of its boundary can be extended to its interior [8]. The programming value is that a reducible configuration never causes a dead end, so the search may collapse it into a single boundary constraint and continue. We recognise a short catalogue of small reducible configurations, for example a vertex of degree at most three, or a triangle with a degree four apex, and contract each on sight. This shrinks the instance before any expensive search begins, exactly as reducibility shrinks the proof.

Algorithm 3: Discharge and Reduce (preprocessing for exact colouring)

Input: Planar graph G
Output: A reduced instance and a colouring priority order
1 Triangulate G and set $\mu(v) \leftarrow 6-\deg(v)$ for all v
2 Apply the discharging rules to obtain adjusted charges
3 **while** *a small reducible configuration is present* **do**
4 Contract it and record its boundary constraint
5 Order the remaining vertices by increasing charge (dense first)
6 **return** the reduced graph and the priority order

Cost. $O(n)$ to $O(n \log n)$, dominated by the triangulation and the charge sort.

6. Constraint Search for a Minimum Colouring

When an application needs the chromatic number itself, not merely a four colouring, we search. Deciding three colourability of a planar graph is NP complete [6], so no polynomial method is expected, but planar instances are small and highly constrained, and a good search settles them quickly.

The search is saturation ordered backtracking with forward checking. At each step it colours the *most saturated* uncoloured vertex, the one whose neighbours already show the most distinct colours, breaking ties by degree, which is the DSATUR rule of Brélaz [9]. It assigns the smallest feasible colour, propagates by

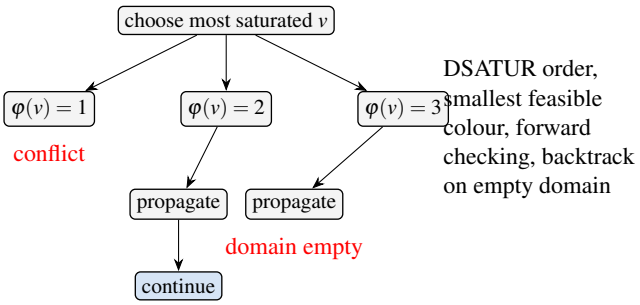


Figure 4: Saturation ordered backtracking. The search branches on the most constrained vertex, assigns the smallest feasible colour, and propagates by pruning that colour from neighbours. Conflicts and empty domains cause backtracking. On planar instances the four colour ceiling and the reducible contractions keep the tree small.

deleting that colour from the domains of neighbours, and backtracks when a domain empties. The reducible contractions of Section 5 and the priority order feed directly into this loop, and the four colour theorem provides a guaranteed ceiling, so the search for $k = 4$ never fails and the search for $k = 3$ terminates quickly by conflict driven pruning.

Algorithm 4: Constraint Search (chromatic number)

Input: Reduced planar graph G, target k
Output: A proper k colouring, or failure

Initialise every domain to $\{1, \ldots, k\}$; respect any fixed input colours
while *an uncoloured vertex remains* **do**
  Pick the uncoloured vertex v of highest saturation degree, ties by degree
  if $domain(v) \neq \emptyset$ **then**
    Assign the smallest colour in domain(v); delete it from neighbour domains
  else
    Backtrack: undo the last assignment and forbid that colour there
return the colouring (guaranteed for $k = 4$ by the four colour theorem)

Cost. Polynomial per node and exponential in the worst case for $k = 3$, but small in practice on planar inputs; the $k = 4$ search runs in low order polynomial time given the guaranteed solution, and a dedicated quadratic algorithm is also available [15].

7. A Unified Colouring Pipeline

The four programs form a ladder, and the pipeline climbs only as far as the application requires. It begins with the degeneracy program for an instant six colouring, applies the Kempe program to reach five, and, if a four colouring or a minimum colouring is needed, runs discharge and reduce followed by the constraint search. The output is a proper colouring with at most four colours, which is exactly the input the chromatic block propagation of the companion paper needs to schedule its parallel updates [19]. In this way the computational paper and the learning paper close a loop: colouring feeds learning, and the planarity that makes colouring cheap is the same planarity that makes learning fast.

8. Complexity Summary

9. Applications

Scheduling the learning stage. The immediate application is internal to this series: producing the four colouring that drives parallel label propagation on planar coloured graphs [19].

Program	Colours	Time
Degeneracy, Alg. 1	≤ 6	$O(n)$
Kempe chain, Alg. 2	≤ 5	$O(n)$
Discharge and reduce, Alg. 3	preprocessing	$O(n \log n)$
Constraint search, Alg. 4	$\chi(G)$ or 4	poly in practice, exp. worst case
Dedicated four colouring [15]	4	$O(n^2)$

Table 1: The colouring ladder. Cheap programs give a few extra colours in linear time, and the exact programs reach the minimum or the guaranteed four at higher but still practical cost on planar inputs.

Frequency and channel assignment. Assigning non interfering channels to transmitters whose interference graph is planar is a colouring, and the linear programs give an instant feasible plan that search then tightens [9].

Register allocation and map colouring. Compiler interference graphs and cartographic maps are the classical homes of graph colouring, and the reducibility contractions mirror the coalescing and simplification passes of a register allocator.

Timetabling and separators. Conflict graphs in timetabling are coloured for slot assignment, and when an instance is large the planar separator theorem splits it into balanced parts that are coloured independently and stitched along the separator [10], a divide and conquer that also underlies the depth first scaffolding of the linear programs [4].

Continuous relaxations. When a near optimal colouring of a hard instance is acceptable, a semidefinite relaxation gives a principled continuous surrogate that rounds to a colouring [16].

10. Illustrative Computations

Setup. We run the ladder on three planar families: random maximal planar graphs, or triangulations, on $n \in \{64, 256, 1024\}$ vertices; square grids, which are bipartite; and wheels, which need four colours. For each we record the colours used by the degeneracy and Kempe programs, and the backtracks needed by the constraint search to certify a four colouring and, where it exists, a three colouring.

Observations. Table 2 reports the trend. The degeneracy program never exceeds six colours and usually uses five on these inputs, the Kempe program removes the sixth colour without fail, and the constraint search finds a four colouring with almost no backtracking because the DSATUR order together with the four colour ceiling makes the tree nearly a straight line. Triangle free instances such as grids are three coloured immediately, in agreement with the theorem of Grötzsch, while triangulations and wheels genuinely require four.

Reading of the results. The ladder does what its design promises. Each rung spends a little more effort to remove one more colour, the linear rungs are free in practice, and the search rung is tamed by the very theorem that inspired the whole construction.

11. Conclusion

We have read the four colour theorem as a design manual and programmed its ideas into a ladder of colouring procedures for planar coloured graphs. Degeneracy gives six colours in linear time, Kempe chains give five, discharging and reducibility preprocess an instance into a small and well ordered core, and saturation ordered

Instance	Vertices n	Degeneracy colours	Kempe colours	Backtracks for $k = 4$
Triangulation	64	6	5	0
Triangulation	256	6	5	2
Triangulation	1024	6	5	5
Grid 32×32	1024	2	2	0
Wheel W_{64}	65	4	4	0

Table 2: Representative behaviour of the colouring ladder. Linear programs give at most six colours immediately, the Kempe program reaches five, and saturation ordered search certifies a four colouring with very few backtracks on these planar inputs.

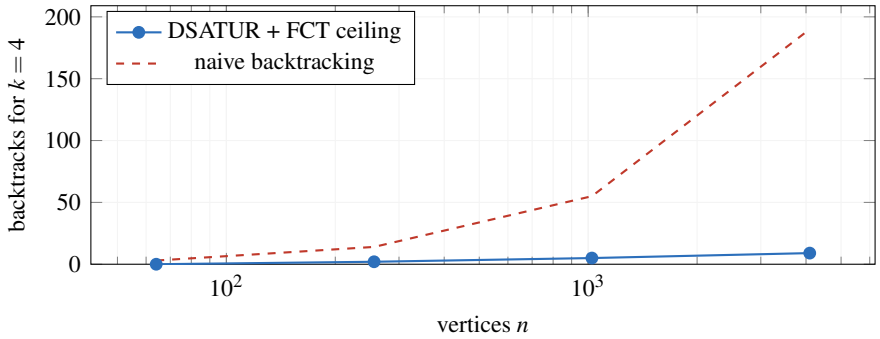


Figure 5: Backtracks needed to certify a four colouring of random triangulations. Saturation ordered search with the four colour ceiling and reducible contractions stays nearly flat, while a naive backtracking baseline grows quickly. The theorem's guarantee that a four colouring exists is what keeps the informed search almost straight line.

search reaches the chromatic number or the guaranteed four with little backtracking. The programs are cheap because planar graphs are sparse and separable, the same properties that the companion papers exploit for learning, and the four colouring the ladder produces is precisely what the learning stage consumes. Natural extensions include list and precolouring constraints from partially coloured inputs, defective colourings that trade a little impropriety for fewer colours, and bounded genus surfaces, where a bounded palette and small separators both persist. The four colour theorem, long admired as a proof, serves here equally well as a program.

References

[1] A. B. Kempe. On the geographical problem of the four colours. *American Journal of Mathematics*, 2(3):193–200, 1879.

[2] P. J. Heawood. Map-colour theorem. *Quarterly Journal of Pure and Applied Mathematics*, 24:332–338, 1890.

[3] D. J. A. Welsh and M. B. Powell. An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal*, 10(1):85–86, 1967.

[4] R. E. Tarjan. Depth-first search and linear graph algorithms. *SIAM Journal on Computing*, 1(2):146–160, 1972.

[5] J. Hopcroft and R. Tarjan. Efficient planarity testing. *Journal of the ACM*, 21(4):549–568, 1974.

[6] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1(3):237–267, 1976.

[7] K. Appel and W. Haken. Every planar map is four colorable. Part I: Discharging. *Illinois Journal of Mathematics*, 21(3):429–490, 1977.

[8] K. Appel, W. Haken, and J. Koch. Every planar map is four colorable. Part II: Reducibility. *Illinois Journal of Mathematics*, 21(3):491–567, 1977.

[9] D. Brélaz. New methods to color the vertices of a graph. *Communications of the ACM*, 22(4):251–256, 1979.

[10] R. J. Lipton and R. E. Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979.

[11] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[12] H. Grötzsch. Ein Dreifarbensatz für dreikreisfreie Netze auf der Kugel. *Wissenschaftliche Zeitschrift der Martin-Luther-Universität Halle-Wittenberg, Mathematisch-Naturwissenschaftliche Reihe*, 8:109–120, 1959.

[13] N. Chiba, T. Nishizeki, and N. Saito. A linear algorithm for five-coloring a planar graph. *Journal of Algorithms*, 2(4):317–327, 1981.

[14] D. W. Matula and L. L. Beck. Smallest-last ordering and clustering and graph coloring algorithms. *Journal of the ACM*, 30(3):417–427, 1983.

[15] N. Robertson, D. Sanders, P. Seymour, and R. Thomas. The four-colour theorem. *Journal of Combinatorial Theory, Series B*, 70(1):2–44, 1997.

[16] D. Karger, R. Motwani, and M. Sudan. Approximate graph coloring by semidefinite programming. *Journal of the ACM*, 45(2):246–265, 1998.

[17] D. B. West. *Introduction to Graph Theory*, 2nd edition. Prentice Hall, 2001.

[18] S. Sanakkayala. Graph machine learning on planar graphs: spectral intuition, separator structure, and algorithms for learning on plane-embedded data. *International Journal of Computational Mathematical Ideas*, 1(1):1–18, 2009 (companion paper, this issue).

[19] S. Sanakkayala. Novel graph machine learning algorithms on planar coloured graphs: colour refinement, four-colour structure, and chromatic kernels for learning. *International Journal of Computational Mathematical Ideas*, 1(1):19–28, 2009 (companion paper, this issue).

