A Linear Algorithm for Colouring Planar Graphs with Five Colours

M. H. WILLIAMS
Department of Computer Science, Heriot-Watt University, 79 Grassmarket, Edinburgh EH1 2HJ

A linear algorithm for colouring planar graphs with at most five colours has recently been published. However, this algorithm, which operates by recursive reduction of the graph, is unnecessarily complicated. An alternative method which is much simpler is presented in this paper.

1. INTRODUCTION

Chiba, Nishizeki and Saito[1] recently put forward a linear algorithm for colouring a planar graph which employs at most five colours. However, their method is unnecessarily complicated and a much simpler algorithm is presented in this paper.

The nomenclature which is used here is the same as that followed in Refs. 1 and 2. All graphs considered here are assumed to be connected and have no multiple edge. The number of vertices in a graph, G, is denoted by n. For any vertex v, the set of all vertices adjacent to v is termed the *neighbourhood of* v,$N(v)$, and the number of vertices in this set is termed the *degree of* v,$d(v)$. The basic operations used in the algorithm are *deletion of a vertex* v which removes v from G, and *identification or merging of vertices* u *and* v which removes u and adds to $N(v)$ any vertices from $N(u)$ which are not already in $N(v)$. When a vertex is removed all edges incident upon that vertex are discarded.

2. THE ALGORITHM

This algorithm is based on successively removing vertices of degree less than six from the graph until at most 5 vertices remain. These vertices can then be coloured with at most five colours. The vertices which had been removed are then reinstated one by one and coloured appropriately.

Two basic transformations are used in reducing the graph, viz.

(1) if $d(v) \leqslant 4$ then remove v (type 1 reduction)
(2) if $d(v) = 5$ then remove v and identify a pair of non-adjacent vertices in $N(v)$ (type 2 reduction).

In order to implement this, the graph G is represented by an adjacency list $L[v]$ for each $v \varepsilon V$ while two queues, Q_4 and Q_5, defined as

$$Q_4 = \{v | d(v) \leqq 4\}$$

$$Q_5 = \{v | d(v) = 5\}$$

are used to contain all vertices available for reduction. Both the adjacency lists and the queues are doubly linked, as are the two copies of each edge (u, v) in $L[u]$ and $L[v]$.

In addition a stack S is used to contain vertices removed from the graph. Each element of S consists of two fields:

(*a*) the vertex v removed, and
(*b*) either a pointer to the adjacency list for v at the time v was removed, or a vertex u with which v was identified.

Before reducing a vertex of degree 5, its neighbours are checked to find two neighbouring vertices with sufficiently low degree which can be identified. For the purposes of the algorithm the vertices selected must have degree less than some constant k. The value of k will be defined later.

The process of identifying two vertices involves the merging of their two neighbour sets, while taking care to avoid the creation of multiple edges. To this end a Boolean array MARK[v] is used, the elements of which are all set to false initially. Each member of the neighbour set of vertex 1 is 'marked' by setting the corresponding entry of MARK to true. Then each 'unmarked' member of the neighbour set of vertex 2 is added to the neighbour set of vertex 1 while each 'marked' member is simply ignored.

Two additional arrays used are: DEG[v] stores the degree of each vertex v, while DP[v] contains a pointer to an element v in Q_i ($4 \leqq i \leqq 5$) if v is present in Q_i.

The algorithm is as follows:

```
procedure COLOUR5;
  procedure CHECK (w);
  if DEG[w] = 5 then add w to Q5
  else if DEG[w] = 4 then move w from Q5 to Q4;
  procedure DELETE(v);
    begin
    for wεL[v] do
      begin
      delete v from L[w];
      DEG[w]: = DEG [w]−1;
      CHECK (w)
      end;
    push (v, pointer to L[v]) on to S;
    delete from Qi the node pointed to by DP [v];
    noofvert: = noofvert − 1
    end;
  procedure IDENTIFY (u, v);
    begin
    for wεL[v] do MARK [w]: = true;
    for wεL[u] do
      begin
      delete u from L[w];
      if not MARK[w]
      then begin (*w is adjacent to u but not v*)
        add w to L[v]; add v to L[w];
        DEG[v]: = DEG[v]+1;
        if DEG[v] = 6 then delete v from Q5
        else if DEG[v] = 5 then move v from Q4 to Q5
        end
      else begin
        DEG[w]: = DEG[w]−1;
        CHECK (w)
        end
      end;
```

```
    for wεL[v] do MARK[w]: = false;
    delete u from Q1;
    push (u, v) on to S;
    noofvert: = noofvert − 1
    end;
  procedure REDUCE;
  while noofvert > 5 do
  if Q4 ≠ Φ then DELETE (top entry from Q4)
  else begin
    take top entry v from Q5;
    if two non-adjacent vertices x,yεN(v) can be found such
    that DEG[x] < k and DEG[y] < k
    then begin
      DELETE(v);
      IDENTIFY (x, y)
      end
    else return v to rear of Q5
    end;
  procedure COLOUR;
    begin
    colour remaining vertices (i.e. those on Q4);
    while S not empty do
      begin
      pop top element (x, y) from S;
      if y is pointer to adjacency list L then colour x with colour
      different from vertices on L
      else colour x with same colour as vertex y
      end
    end;
  begin
  scan graph and set up arrays MARK, DEG, Q4, Q5 and DP;
  noofvert: = n;
  REDUCE;
  COLOUR
  end (*OF COLOUR5*)
```

3. TIME COMPLEXITY

Initially Q_4 is set up to contain those vertices with degree less than or equal to four, Q_5 to contain those with degree five. That both queues cannot be empty follows from the fundamental property of planar graphs that the average degree of vertices in a planar graph is less than 6. We now establish the following theorem concerning the queues Q_i.

Theorem 1

At the beginning of each iteration of the loop within the procedure REDUCE the queues Q_4 and Q_5 will contain all vertices v with $d(v) < 6$ in the graph.

Proof. Proof is by induction. Assume that the theorem holds at some point in the execution of REDUCE. The only operations performed by REDUCE which can alter the state of the graph or the queues (other than their order) are as follows

(*a*) DELETE (top entry from Q_4) or DELETE(v). In this case a vertex is deleted from the graph and from Q_i. It is deleted from the adjacency lists of each of its neighbours and the degree of each neighbour is decreased by one. After doing so the degree of each neighbour is checked and if it is less than 6, the neighbour is added to Q_5 or moved from Q_5 to Q_4 if necessary. No other vertex of the graph will be affected by the deletion. Thus after the deletion the Q_i will contain all vertices v with $d(v) < 6$.

(*b*) IDENTIFY (x, y). In this case vertex x is deleted from the graph and from Q_i, and each of the neighbours of x which is not a neighbour of y will be connected to y. As in (*a*), the degree of each affected vertex is checked so that after identification of x and y the Q_i will contain all vertices v with $d(v) < 6$.

Since Q_4 and Q_5 are set up initially to contain all vertices v with d(v) < 6, the theorem is proved. ∎

Since the average degree of vertices in a planar graph is less than six, these queues can never both be empty as long as there are vertices in the graph.

Noofvert is a variable which represents the number of vertices in the graph. It is initially set to have value n and is reduced by 1 in each of the procedures DELETE and IDENTIFY when a vertex is removed from the graph.

Theorem 2

The procedure COLOUR5 will colour any planar graph G with at most five colours.

Proof. Each reduction step removes either one or two vertices from the graph until it has no more than five vertices. These can be coloured with at most five colours.

In the final phase each of the vertices removed from the graph is unstacked from the stack S and assigned a colour. If at the time that it was removed from the graph a vertex v had:

(*a*) fewer than five neighbours

then v will be connected to vertices coloured with at most four colours;

(*b*) five neighbours

then S will contain two entries: an entry E_1 identifying one of the neighbours of v, x, with another, y, and below it an entry E_2 containing a pointer to an adjacency list with five vertices (including x and y). At this point a colour will have been assigned to vertex y, and entry E_1 can be unstacked and vertex x assigned the same colour. Then E_2 can be unstacked, and v will be connected to vertices coloured with at most four colours.

Thus each vertex can be coloured with one of at most 5 colours. ∎

Now consider in more detail the way in which Q_5 is handled and vertices are selected for identification. Initially Q_5 is set up to contain all vertices with degree 5. In the procedure REDUCE an entry may be removed from the front end of Q_5. If the vertex corresponding to this entry has two non-adjacent neighbours of sufficiently small degree, it will be deleted; otherwise the vertex will be returned to the rear end of the queue (such a vertex will be termed a *blocked vertex*). During the process of a deletion or identification vertices may be removed from any point within the queue or added to the rear end of the queue.

Consider a particular instance of Q_5 at a point where Q_4 is empty. The term *stage* will be used to refer to the period from this point until Q_4 is (once again) empty and all the original entries on this instance of Q_5 have been removed, so that Q_5 consists entirely of new entries and entries which have been returned to the rear of the queue (blocked vertices) or the graph has been collapsed to a point that it contains less than six vertices. The proof of linearity which follows later relies on the fact that the total number of blocked vertices encountered (or, if you prefer, the total number of times a vertex is returned to the end of Q_5) depends linearly on n.

The first result which is required concerns the maximum number of blocked vertices in a stage.

Lemma (Upper bound on number of blocked vertices in a stage.)
If at the beginning of a stage a graph contains n_i vertices of degree $i (i \geq 5)$, and during the stage d type 2 reductions and e type 1 reductions are performed, the maximum number of blocked vertices encountered during the stage (B) cannot exceed

$$\sum_{i=k} in_i/2 + (2k-4)\,d/2.$$

Proof. Consider a vertex v of degree 5 in the graph. For v to be a blocked vertex it must have at least two neighbours of degree at least k (since otherwise it is always possible to find two non-adjacent neighbours each with degree less than k).

If each vertex of degree i ($i \geq k$) has edges in common with i blocked vertices then since each blocked vertex must have edges in common with at least two vertices of degree greater than or equal to k, it follows that, ignoring any new vertices of degree at least k created during the stage, the number of blocked vertices cannot exceed

$$\sum_{i=k} in_i/2.$$

Since for each type 2 reduction performed during the stage, a pair of vertices will be identified, this could create up to d new vertices of degree greater than k. As the degree of each of the vertices identified must be at most $k-1$ before deletion and identification, the largest degree which the resulting vertex may have is $2k-4$. Each type 1 reduction only decreases the degrees of vertices and thus cannot create additional blocked vertices.

Hence the total number of blocked vertices encountered during the stage may not exceed

$$\sum_{i=k} in_i/2 + (2k-4)\,d/2. \qquad \blacksquare$$

For the current purposes k will be taken to be the value 13.

Theorem 3

If at the beginning of a stage a graph contains n_i vertices of degree i ($i \geq 5$) and during the stage R reductions are performed and B blocked vertices are encountered then

$$\frac{B}{R} < 237.$$

Proof. Assume that during the stage d type 2 reductions and e type 1 reductions are performed. For each type 2 reduction one vertex of degree 5 will be deleted and the degree of each of its neighbours decremented (removing the neighbour from Q_5 if necessary). For each type 1 reduction a vertex of degree less than 5 is deleted and the degree of each neighbour decremented (removing it from Q_5 if necessary). Thus altogether $fd+ge$ vertices will be removed from Q_5 during the stage, where $1 \leq f \leq 6$ and $0 \leq g \leq 4$. All other vertices originally on Q_5 must be blocked vertices, i.e.

$$\text{no. of blocked vertices} = B = n_5 - fd - ge.$$

From ref. 2, for a maximal planar graph (i.e. one in which each face is bounded by three edges)[3] the number of vertices of degree 5 initially on Q_5 is given by

$$n_5 = 12 + \sum_{i=1} in_{i+6}.$$

If the graph at the beginning of the stage is not maximal planar and requires the addition of m edges to make it maximal planar, it can easily be shown that

$$n_5 = 12 + 2m + \sum_{i=1} in_{i+6}, \qquad (1)$$

from which it follows that

$$B = 12 + 2m + \sum_{i=1} in_{i+6} - fd - ge. \qquad (2)$$

From the previous lemma

$$12 + 2m + \sum_{i=1} in_{i+6} - fd - ge \leq \sum_{i=k} in_i/2 + (2k-4)\,d/2.$$

For $k = 13$

$$11d + fd + ge \geq 12 + 2m + \sum_{i=1} in_{i+6} - \sum_{i=13} in_i/2$$

$$= 12 + 2m + \sum_{i=1}^{6} in_{i+6} + \sum_{i=1} in_{i+12}/2. \qquad (3)$$

Now since $(11+f)\,d + ge \leq (11+f)\,(d+e)$

$$\leq 17\,(d+e)$$
$$= 17R,$$

equation (3) becomes

$$17R \geq 12 + 2m + \sum_{i=1}^{6} in_{i+6} + \sum_{i=1} in_{i+12}/2. \qquad (4)$$

Since $f \geq 1$,

$$\frac{11d}{12} \leq \frac{11fd}{12}.$$

Adding $fd/12$ to each side gives

$$fd \geq (11+f)\,d/12$$

and hence

$$(fd+ge) \geq ((11+f)\,d+ge)/12$$

Combining this with equations (2) and (3) gives

$$B \leq 12 + 2m + \sum_{i=1} in_{i+6} - \frac{1}{12}\left(12 + 2m + \sum_{i=1}^{6} in_{i+6} + \sum_{i=1} in_{i+12}/2\right)$$

$$= \frac{11}{12}\left(12 + 2m + \sum_{i=1}^{6} in_{i+6}\right) + \sum_{i=1}\left(\frac{23i}{24} + 6\right) n_{i+12}. \qquad (5)$$

Comparing each term of (5) with the corresponding term of (4) the largest ratio occurs when $i = 13$, namely

$$\frac{167}{24} \Big/ \frac{1}{2} = \frac{167}{12},$$

from which it follows that

$$\frac{B}{17R} \leq \frac{167}{12}$$

or

$$B < 237R. \qquad \blacksquare$$

Theorem 4

The procedure COLOUR5 will colour any planar graph G containing n vertices in $O(n)$ time.

Proof. It can easily be verified that the procedure DELETE deletes a vertex v from the graph in $O(d(v))$ time

where $d(v) \leqslant 5$. Since this procedure can be called at most $n-5$ times the contribution to the total running time is at most $O(n)$.

Likewise the procedure IDENTIFY (x, y) can be seen to operate in time of order $O(d(x)+d(y))$. Since for each call $d(x) < k$ and $d(y) < k$, and since the number of calls can be at most $(n-5)/2$, the total time spent by the procedure IDENTIFY is at most $O(n)$.

On each iteration of the loop in the procedure REDUCE either at least one vertex will be removed from the graph or a blocked vertex will be moved from the front to the rear of the queue. From the previous theorem the total number of blocked vertices encountered in all the stages must be less than 237 times the total number of reductions performed, i.e.

$$\Sigma B < 237 \Sigma R.$$

Since the total number of reductions performed is at most $n-5$, the total number of blocked vertices must be less than 237 $(n-5)$ and hence the number of iterations of the loop in the procedure REDUCE is $O(n)$. Each operation can be shown to operate in time less than or equal to some fixed constant (since even the test for two non-adjacent vertices $x, y \varepsilon N(v)$ such that DEG[x] $< k$ and DEG[y] $< k$ depends only on k).

In the procedure COLOUR the stack S is unstacked element by element and either

(*a*) an adjacency list of up to 5 elements is scanned and a set S′ of colours used is constructed; then a colour not contained in this list is selected (this is always possible otherwise Theorem 2 would not hold), or

(*b*) the colour of vertex y is assigned to vertex x.

Since the stack contains just less than n elements, the time taken for this operation is $O(n)$.

Hence the procedure COLOUR5 operates in $O(n)$ time. ∎

This algorithm has been tested using a set of 3000 planar graphs, each with between 50 and 200 vertices, and an analysis of its performance is given in ref. 4.

REFERENCES

1. N. Chiba, T. Nishizeki and N. Saito, A linear 5-coloring algorithm of graphs, *Journal of Algorithms* **2**, 317–327 (1981).
2. M. H. Williams, Cubic map configurations, *Information Processing Letters* **11**, 180–185 (1980).
3. O. Ore, *The Four-color Problem*. Academic Press, New York (1967).
4. M. H. Williams and K. T. Milne, The performance of algorithms for colouring planar graphs. *The Computer Journal* **27**, 165–170 (1984).