Totally colouring a graph with maximum degree four

Kathryn F. Jones and Jennifer Ryan

Department of Mathematics
University of Colorado
Denver, CO 80204
U.S.A.

Abstract. Behzad has conjectured that a simple graph G can always be totally coloured using two more colours than the maximum degree in G. The conjecture has been verified for several special classes of graphs by Behzad, Chartrand and Cooper, Rosenfeld, and Meyer, and by Vijayaditya for graphs with maximum degree less than or equal to 3. We show algorithmically that the conjecture is true for graphs with maximum degree 4.

Introduction

Definition: A *total colouring* of a graph is a colouring of the vertices and edges such that no two adjacent or incident objects receive the same colour.

The *Total Colouring Conjecture* (Behzad [1]) states that a graph G can always be totally coloured using $\Delta(G) + 2$ colours, where $\Delta(G)$ is the maximum degree of any vertex in G. It is clear that the minimum number of colours needed is $\Delta(G) + 1$, since the vertex having maximum degree and all its incident edges must receive distinct colours. The conjecture has been verified for complete graphs by Behzad, Chartrand, and Cooper [2], for bipartite, complete tripartite and complete balanced k-partite graphs by Rosenfeld [8], for graphs which are composed of stable sets of equal size arranged in a cycle by Meyer [6], and for graphs whose maximum degree is 3 by Vijayaditya [9]. We show that the conjecture is true for graphs with maximum degree 4.

Theorem 1 (König [5]). *A graph with maximum degree Δ can be imbedded in a Δ-regular graph.*

We present an algorithm that totally colours a 4-regular graph using 6 colours. It will then follow from Theorem 1 that any graph with maximum degree 4 can be totally coloured with 6 colours. The following definitions and theorems will be needed for the development to follow.

Definitions: A *2-factor* of a graph G is a spanning subgraph F of G such that each vertex in $V(G)$ has degree 2 in F. If G can be represented as the edge sum of 2-factors, then G is *2-factorable*.

Note that a 2-factor consists of a collection of vertex disjoint cycles.

Theorem 2 (Petersen [7]). *A non-empty graph G is 2-factorable if and only if G is $2k$-regular for some $k \geq 1$.*

Thus any 4-regular graph can be expressed as the edge sum of two 2-factors.

Theorem 3 (Brooks [4]). *Any connected graph which is neither an odd cycle nor complete can always be vertex coloured using at most $\Delta(G)$ colours.*

Totally Colouring a 4-Regular Graph

Let G be a 4-regular graph. The Total Colouring Conjecture has been shown to be true for complete graphs in [2], so we assume that G is not complete. By Theorem 2 above, any 4-regular graph can be decomposed into two 2-factors, F_1 and F_2. Each 2-factor is the disjoint union of cycles. The algorithm to totally colour a 4-regular graph works in three stages.

The first stage colours the vertices legally using only four colours. (This is always possible by Theorem 3.) The second stage colours the edges of F_1 using the same four colours used to colour the vertices, without recolouring any vertex, so the vertices remain legally coloured for the original graph. The third and final stage, which is the most complex part of the algorithm, uses the fifth and sixth colours to colour the edges of F_2. It is often necessary to recolour previously coloured edges and vertices in this stage.

```
Begin
  For each cycle C = {v_1, e_1, ..., v_n, e_n, v_1} of F_1 do
  Begin
    If colour(v_{i-1}) = colour(v_{i+1}) for all i = 1, ..., n, then
      {the cycle is even and uses only two colours}
      colour the edges of C alternately with the remaining two colours
    Else
      {there exists k with colour(v_{k-1}) ≠ colour(v_{k+1})}
      Begin
        Relabel C so that v_k becomes v_1;
        Colour e_1 with colour(v_n) which ≠ colour(v_2) or colour(v_1)
        For i = 2, ..., n - 1 {e_{i-1}, v_i, and v_{i+1} have been coloured}
          Colour e_i with the one colour free at e_i;
        {Now e_1, e_{n-1}, v_n and v_1 are all coloured but from above
        colour(e_1) = colour(v_n)}
        Colour e_n with the (at least) one colour free at e_n;
      End {Else}
  End {For}
End.
```

Given that the vertices of G have been legally coloured using four colours, this algorithm colours the edges of each cycle of F_1 without recolouring any vertex. A cycle will be described $\{v_1, e_1, v_2, \cdots, v_n, e_n, v_1\}$, where e_i is the edge connecting the nodes v_i and v_{i+1}. For ease of notation we will consider $v_{n+1} \equiv v_1$.

Referring to the comments in the text of the algorithm, it is easily seen that the algorithm correctly colours the edges of F_1 without recolouring any vertex, thus the vertices remain legally coloured in G.

Now we want to extend this colouring to a total colouring of G by colouring the edges of F_2. Let Z be a cycle component of F_2. If Z is an even cycle, then the edges of Z can be coloured alternately with the 2 colours which have not been used so far. If Z is an odd cycle, then we will show the following:

If F_1 is coloured with the colours $\{a, b, c, d\}$, then every odd cycle Z of F_2 can be coloured using only the additional colours e and f and recolouring at most one vertex, v, of Z with e (or f). If such a vertex v is coloured e (or f), then one edge incident to v will be coloured f (or e).

Let $v_1, \ldots, v_n$ be the consecutive vertices of Z and let E_i be the set of colours used on the edges in F_1 that are incident to v_i. If $E_i \cup E_{i+1} = \{a, b, c, d\}$ for all i, then $E_i = E_{i+2}$ for all i, where we consider $E_{n+1} = E_1$, etc.. Since Z is an odd cycle, $n-1$ is even and we have $E_n = E_2 = \cdots = E_{n-1} = E_1 = \cdots$. This is a contradiction. Hence there exist v_i and v_{i+1} such that $E_i \cup E_{i+1} \neq \{a, b, c, d\}$. Consider the edge v_iv_{i+1} and suppose that $a \notin E_i \cup E_{i+1}$. If neither v_i nor v_{i+1} is coloured a, then we can colour v_iv_{i+1} with a and the rest of the edges of Z alternately with e and f. Since the edges and vertices of F_1 are coloured using only $\{a, b, c, d\}$, this will give a good total colouring.

Now suppose one of the vertices v_i and v_{i+1} is coloured a. Without loss of generality, suppose that v_i is coloured a. If Z is the first cycle of F_2 to be coloured, then colour v_iv_{i+1} with colour a, colour v_i with colour e and colour the remaining edges of Z alternately with colours e and f starting with e. Since both v_i and v_{i+1} cannot be coloured a, this gives a good colouring of Z in which exactly one vertex has been recoloured with e. However, if several cycles of F_2 have already been coloured, it may be the case that v_i has neighbours which have been recoloured with e. Note that since each vertex is only on one cycle of F_2, none of the vertices of Z will have been recoloured, so such a neighbour is not on Z. Also notice that as we colour cycles of F_2, we may allow a vertex to recoloured e or f, but this doesn't affect the colours on vertices of cycles which have not yet been coloured. If f does not appear at a neighbour of v_i then interchange the roles of e and f in the above colouring of Z.

We now suppose that v_i's two neighbours, x and y, not on Z are coloured with e and f respectively. Since (inductively) at most one vertex in an odd cycle is coloured with e or f, x and y are in different odd cycle components, Z_1 and Z_2, of F_2. Also (inductively, given our colouring procedure), one of the edges incident to x (y) in F_2 is coloured f (e). If x (or y) does not have a neighbour that is coloured f (or e), then we could interchange e and f in Z_1 (or Z_2), making e (or f) available to colour v_i as above. Note that this interchange would not affect the rest of G, since x (or y) is the only vertex coloured e (or f) in Z_1 (or Z_2).

Thus we have managed to colour Z unless x and y have neighbours coloured f and e respectively. The rest of our algorithm is devoted to handling this case.

We summarize again the situation:

The colour $a \notin E_i \cup E_{i+1}$, v_i is coloured a and is adjacent to vertices x and y in F_1 which are coloured e and f respectively. Further, x has a neighbour coloured f and y has a neighbour coloured e.

Let the edges in F_1 adjacent to v_i be coloured b and c. If neither v_{i-1} nor v_{i+1} is coloured d, then we could recolour v_i with d, colour $v_i v_{i+1}$ with a and the remainder of Z alternately with e and f. Thus we suppose that at least one of v_{i-1} and v_{i+1} is coloured d, and so either b or c is not used at v_{i-1} nor at v_{i+1}. We assume that b is this colour, so that if we are able to recolour the edge $v_i x$, which is currently coloured b, we will be able to use the colour b to colour v_i. (If c is the colour not at v_{i-1} or v_{i+1} the argument below proceeds using Z_2 instead of Z_1.) Finally, let $\{w_1 = x, w_2, \cdots, w_n\}$ be the vertices of Z_1, and let e_i denote the edge from w_i to w_{i+1}, where $w_{n+1} = w_1$. Since, inductively, only one edge of Z_1 has been coloured with a colour other than e or f, we have that one of e_1 or e_n is coloured f and the other is coloured some colour $\alpha \neq b$. Let $\gamma \neq b$ (and $\neq \alpha$) be the colour of the second edge of F_1 incident to w_1. (So the edge coloured γ has w_1 as one endpoint and its other endpoint is coloured f.)

We will proceed by analyzing the possible cases given the current situation. The following subsection describes a situation that is common to several of the cases, and an operation ("shifting") to deal with such a situation. The final subsection gives a colouring of Z in all possible cases, possibly using the shifting procedure.

2.1 The Shifting Procedure

Given the setting described above, let β be the colour of w_2. In general, it is possible that β is the same as either γ or α, but the shifting procedure will only be called when α, β, and γ are distinct and are all not equal to b. Thus we assume for now that $\{\alpha, \beta, \gamma\} = \{a, c, d\}$. The shifting procedure will recolour the edges of Z_1 and all vertices but w_1 without using the colour α on e_1 or e_n. The colourings in the following subsection will make use of the fact that α has been "freed up" to both colour w_1 and Z.

Define $m(w_i) \equiv \{a, b, c, d\} - colour(w_i) - colour$(two F_1 edges incident to w_i), for $i = 1, \ldots, n$. So $|m(w_i)| = 1$ for $i = 2, \ldots, n$ and $|m(w_1)| = 2$. The shifting procedure, given below in algorithmic form, calls a subroutine "BACKUP", whose statement will follow that of the shifting procedure. Let $c(z) \equiv colour(z)$, for all $z \in V \cup E$.

```
Begin {SHIFTING}
  Uncolour w_1 and all edges of Z_1
  Let i = 2 and let done = false.
  { Note: β = c(w_2) ∈ m(w_1) }
  While i ≤ n and done = false do
  Begin
    { Note: By induction c(w_i) ∈ m(w_{i-1}) }
    Case 1: If w_i is not adjacent to both a vertex coloured e and one coloured f,
    then
    Begin
      c(e_{i-1}) ← c(w_i) ∈ m(w_{i-1});
      c(w_i) ← e (or f);
      colour remaining edges in Z_1 f and e (or e and f) alternately;
      done ← true
    End{If}
    Else
    Case 2: w_i has neighbours coloured e and f.
    Begin
      If m(w_i) = c(w_{i+1}) then i ← i + 1
      Else m(w_i) ≠ c(w_{i+1})
      Begin
        If m(w_i) = c(w_{i-1}), then call BACKUP(i);
        done ← true
        Else {m(w_i) ≠ c(w_{i-1})}
        Begin
          c(e_{i-1}) ← c(w_i) ∈ m(w_{i-1});
          c(w_i) ← m(w_i) ≠ c(w_{i-1}) or c(w_{i+1});
          colour remaining edges of Z_1 e and f alternately ending with e_n
          coloured f;
          done ← true
        End{else}
      End{Else}
    End{Else (Case 2)}
```

End{While}

{ Now, $i = n + 1$, *done* = *false*, and $m(w_{n-1}) = c(w_n)$.}

If $m(w_n) \neq c(w_{n-1})$, then

Begin

$c(e_{n-1}) \leftarrow c(w_n)$;

$c(w_n) \leftarrow m(w_n)$;

colour remaining edges of Z_1 f and e alternately, with e_1 coloured f

End{If}

Else { $m(w_n) = c(w_{n-1})$ }

Call BACKUP(n)

End{SHIFTING}

Given the fact that, by induction, $c(w_i) \in m(w_{i-1})$ at each stage of the algorithm, it is not hard to see that Z_1 is legally coloured when BACKUP is not called. (The algorithm checks colours of w_{i-1} and w_{i+1}).) Note that SHIFTING terminates immediately after returning from BACKUP. We now describe BACKUP.

BACKUP is called with parameter k, when $m(w_k) \neq c(w_{k+1})$, but $m(w_k) = c(w_{k-1})$. Also, for all $2 \leq i \leq k$, $c(w_i) \in m(w_{i-1})$.

Begin{BACKUP}

Let *temp* $\leftarrow c(w_k)$;

$c(w_k) \leftarrow m(w_k) \neq c(w_{k+1})$;

{ Now *temp* $= m(w_{k-1})$ = previous value of $c(w_k)$ }

$i \leftarrow k$;

While $c(w_i) = c(w_{i-1})$ do

Begin

temp $\leftarrow c(w_{i-1}) = m(w_{i-2})$;

$c(w_{i-1}) \leftarrow m(w_{i-1})$;

$i \leftarrow i - 1$

End{While}

{ Note: Now *temp* $= m(w_{i-1})$ = previous value of $c(w_i)$ and now $c(w_i) \neq c(w_{i-1})$.}

$c(e_{i-1}) \leftarrow$ *temp*

Colour remaining edges of Z_1 with e and f alternately

End{BACKUP}.

We now verify that if SHIFTING terminates after executing BACKUP, the colouring of Z_1 (with w_1 left uncoloured) is legal. First note that BACKUP will terminate with $i \geq 2$, since w_1 is uncoloured. When w_{i-1} is assigned the colour $m(w_{i-1})$ we know that now $c(w_{i-1}) \neq c(w_i)$. Also, by definition of the function m, there is no problem with the F_1 edges incident to w_{i-1}. It is possible that now $c(w_{i-1}) = c(w_{i-2})$, in which case we iterate. When edge e_{i-1} is coloured *temp*, *temp* is $m(w_{i-1})$, so the colouring is legal at w_{i-1}. Further, *temp* is the previous colour of w_i, which is not the current colour of w_i, nor could it be the colour of any edges incident to w_i.

It should be noted here that the shifting procedure may sometimes be called with the indices of the vertices on Z_1 reversed; this has the effect of performing the shift in the reverse direction.

The shifting procedure will result in one of the following situations:

1. If w_2 has no neighbour coloured e (or f), then the procedure stops with $i = 2$ and:

  a) $c(e_1) = \beta$;
  b) $c(w_2) \in \{e, f\}$;
  c) $c(e_n) \in \{e, f\}$; and
  d) $c(w_n)$ is unchanged so that $c(w_n) \notin \{\alpha, e, f\}$.

2. If w_2 has both a neighbour coloured e and one coloured f, but $m(w_2) \neq c(w_3)$, the procedure stops with:

  a) $c(w_2) = m(w_2)$;
  b) $c(w_n)$ unchanged so that $c(w_n) \notin \{\alpha, e, f\}$;
  c) $c(e_1) = \beta$; and
  d) $c(e_n) = f$.

This situation also could result from the call of BACKUP.

3. If the shift occurs at some $i > 2$, the procedure terminates with:

  a) $c(e_1) = e$ (or f); and
  b) $c(e_n) = f$ (or e).

This situation could also result from the call of BACKUP.

2.2 Completing the Colouring

Given the setting described in italics above, we will give colourings of w_1, the edge $w_1 v_i$, v_i, and the edge $v_i v_{i+1}$, as a sequence c_1, c_2, c_3, c_4. (Recall we are assuming that one of v_{i-1} and v_{i+1} is coloured d and that neither of them is coloured b.) The colour c_4 will never be e or f, so after colouring w_1, $w_1 v_i$, v_i and $v_i v_{i+1}$ with such a sequence, the remaining edges of Z can be coloured e and f alternately. (Note that v_i may be recoloured e in which case the edge $v_{i-1} v_i$ must of course be coloured f.)

If $d \notin \{\alpha, \gamma\}$, then the colouring sequence e, d, b, a can be used, so assume that $d \in \{\alpha, \gamma\}$.

Case 1: $\alpha = d$. Then it must be that $\gamma = a$ or c.

If neither w_n or w_2 is coloured $\{a, c\} - \gamma$, then colour $\{a, c\} - \gamma, b, e, a$. Thus assume that at least one of w_n and w_2 is coloured $\{a, c\} - \gamma$. If $c(w_2) = \{a, c\} - \gamma$, then invoke the shifting procedure. If $c(w_n) = \{a, c\} - \gamma$, then invoke the shifting procedure with the indices of the vertices of Z_1 reversed.

If the shift yields situation 1 above, then w_2 (or w_n) is coloured e or f, e_1, (or e_n) is coloured $\{a, c\} - \gamma$, and e_n (or e_1) is coloured e or f. If w_n (or w_2) is not coloured d then colour d, b, e, a. Otherwise colour b, d, e, a.

If the shift yields situation 2, then neither w_1 nor w_n are coloured e or f and together e_1 and e_n have colours $\{a, c\} - \gamma$ and f. In this case we colour e, d, b, a.

If the shift results in situation 3, then e_1 and e_n together have colours e and f. If $b \in \{c(w_n), c(w_2)\}$, then $\{c(w_2), c(w_n), \gamma\}$ intersects with at most two of $\{a, c, d\}$. Use the (at least one) free colour in $\{a, c, d\}$ to colour w_1 and then colour $v_i w_1$, v_i, and $v_i v_{i+1}$ with b, e, a respectively. Otherwise, $b \notin \{c(w_2), c(w_n)\}$ and colour b, d, e, a.

Case 2: $\gamma = d$. Then $\alpha = a$ or c.

The vertex adjacent to w_1 on Z_3 that is coloured f will be denoted z. Note that one edge incident to z must be coloured e (since on any cycle of F_2 we colour at most one edge a colour other than f or e). Denote by $m(z)$, the colour in $\{a, b, c\}$ not on edges incident to z. If $m(z) \in \{\{a, c\} - \alpha, b\}$, colour $w_1 z$ with $m(z)$, then colour e, d, b, a. So assume that the edges incident to z are coloured $\{d, e, \{a, c\} - \alpha, b\}$ and thus $m(z) = \alpha$.

If $\{a, c\} - \alpha$ does not occur on either w_n or w_2, then colour as follows: $\{a, c\} - \alpha, b, e, a$.

If $\{a, c\} - \alpha$ occurs at w_2 then invoke the shifting procedure. If it occurs at w_n, then invoke the shifting procedure with reversed indices.

If the shift yields situation 1, then one of w_2 and w_n is coloured e (or f). Together, the edges e_1 and e_n use the colours $\{a, c\} - \alpha$ and e (or f). If the other of w_2 and w_n is coloured d, then colour α, b, e, a. Otherwise, recalling that α is not used at z, recolour the edge $w_1 z$ with α and colour d, b, e, a.

If the shift yields situation 2, then neither w_1 nor w_n are coloured e. We again recolour the edge $w_1 z$ with α and then colour e, d, b, a.

Finally, suppose the shift produces situation 3. Then e_1 and e_n together use the colours e and f. We recolour $w_1 z$ with α. Now one of $\{\{a, c\} - \alpha, b, d\}$ does not occur on w_n or w_2. Use this missing colour on w_1. Colour $v_i w_1$ with a colour in $\{b, d\} - c(w_1)$. Finally let $c(v_i) = e$ and $c(v_i v_{i+1}) = a$.

We have proved the following:

Theorem 4. *A 4-regular graph can be totally coloured using at most 6 colours.*

Applying Theorem 1, we have

Corollary. *A graph with maximum degree 4 can be totally coloured using at most 6 colours.*

We conclude by noting that our argument for 4-regular graphs does not extend to $2k$-regular graphs with $k \geq 3$. The first place where a generalization of our proof technique breaks down is in the second stage. Given that the vertices of a $2k$-regular graph have been legally coloured, it seems difficult to colour the edges of a $2(k-1)$-regular subgraph without recolouring any vertex and using only $2k$ colours.

Even if this initial problem were overcome, in trying to extend the $2k$-colouring of a graph minus a 2-factor to a colouring of the entire graph, the attempt to colour a cycle as done here fails. If Z is an odd cycle with vertices $\{v_1, \cdots, v_n\}$ it may be the case that $|E_i \cup E_{i+1}| = 2k$ for every i, so that there is no one of the $2k$ colours available to start the colouring of Z.

References

1. M. Behzad, "Graphs and Their Chromatic Numbers", Doctoral Thesis, Michigan State University, 1965.
2. M. Behzad, G. Chartrand and J.K. Cooper Jr., *The Colour Numbers of Complete Graphs*, J. London Math. Soc. **42** (1967), 225–228.
3. M. Behzad, *The Total Chromatic Number of a Graph*,, in "A Survey of Combinatorial Mathematics and its Applications,", (Proc. Conf. Oxford 1969),1-8, Academic Press, London, 1971.
4. R.L. Brooks, *On Colouring the Nodes of a Network*, Proc. Cambridge Philos. Soc. **37** (1941), 194–197.
5. D. König, "Theorie der endlichen und unendlichen Graphen", Leipzig,, 1936.
6. J.C. Meyer, *Nombre chromatique total du joint d'un ensemble stable par un cycle*, Discrete Math. **15** No. 1 (1976), 41–54.
7. J. Petersen, *Die theorie der regulären graphs*, Acta Math. **15** (1891), 193–220.
8. Rosenfeld, *The Total Chromatic Number of Certain Graphs*, Notices Amer. Math. Soc. **15** (1968), p. 478.
9. N. Vijayaditya, *On the Total Chromatic Number of a Graph*, Journal of the London Math. Soc. (2) **3** (1971), 405–408.