AUSTRALASIAN JOURNAL OF COMBINATORICS
Volume **58(2)** (2014), Pages 320–332

A colour-sliding problem on graphs

Landan Hicks Marc Lipman Matt Walsh

Indiana-Purdue University Fort Wayne
Fort Wayne, Indiana
U.S.A.

Abstract

We investigate several aspects of a family of problems involving restoring a scrambled proper colouring on the vertices of a graph. Specifically, the restoration is accomplished by removing the colour from some number of vertices and sliding the remaining colours into the resulting "voids"; our principal interest in this paper is the number of voids required for different classes of graphs.

1 Sliding colours

For our purposes, graphs are simple and finite without loops. A *proper colouring* of (the vertices of) a graph G is a function $f : V(G) \to [k]$ such that $f(v) \neq f(w)$ whenever $vw \in E(G)$ (where $[k] = \{1, \ldots, k\}$); we assume for convenience that for any $i \in [k]$ the set $f^{-1}(i)$ is nonempty. The *colour profile* of a colouring is the multiset of colours it employs; two colourings feature the same colour profile if they employ the same colours the same number of times.

Suppose that we have a proper colouring of G which is then scrambled by some villainous person or persons. (By scrambled we mean that a permutation $\pi : V(G) \to V(G)$ is composed with the colouring.) We wish to restore the colouring to propriety (although not necessarily to the exact same colouring as we started with); however, unlike in Clark *et al.* [1] ours is not the power of our enemy to move colours around arbitrarily. What we can do is delete the colours from one or more vertices, resulting in a partial colouring containing *voids*; given such a partial colouring, we may then evolve it into a new colouring by finding a pair of adjacent vertices v, w such that v is coloured and w is not and switching the colour with its adjacent void, in effect sliding the colour along the joining edge.

For the purposes of restoring a proper (partial) colouring we treat voids as irrelevant to the constraints: that is, two adjacent voids do not count against propriety. We treat the deletion of colours as an expensive operation compared to colour sliding, and thus our principal question is: given a graph G, what is the minimum number of voids that must be deployed to guarantee that any scrambled colouring of G can be restored to a proper partial colouring by sliding colours? Let us call this minimum the *sliding number* of G, denoted $\xi(G)$. (Note that the colouring being scrambled is not required to be minimum or balanced; all we ask is that it is proper before scrambling.) We do not require that the voids be filled in (with the original deleted colours, in some order) at the end of the process; we also allow the restorer to choose exactly which vertices become voids in the original scrambled colouring. Strengthening either of these requirements may lead to situations where extra voids are required.

In the following section we give an obvious upper bound for $\xi(G)$ and show that there are infinite classes of graphs which require this number of voids. The following three sections determine $\xi(G)$ (or at least very tight bounds on $\xi(G)$) for all (connected) graphs G: trees in Section 3, 2-connected graphs in Section 4, and general graphs in Section 5. We subsequently show that, given a known proper colouring of G, the number of slides required is polynomial in the order of G.

2 Sliding and edge covers

If our goal is to minimize the number of voids required, one obvious strategy gives a clear upper bound: conflicts can only happen between adjacent vertices of the same colour, so if the voids were arranged so that we never had two coloured vertices adjacent then we would have a proper partial colouring. In such a case the voids would form an edge cover (sometimes edge-by-vertex cover) of G, whence:

Lemma 2.1. *For any graph G, $\xi(G) \leq \beta'(G)$ where $\beta'(G)$ denotes the size of a minimum edge cover of G.*

Note that we can achieve this upper bound in several ways. One example that does so is the graph consisting of n disjoint copies of K_n; our villain could scramble the colours so that all instances of a given colour lie in the same component, and hence we can do no better than uncolouring all vertices but one in each clique. Clearly, we can generalize this to disconnected graphs in general; for this reason, henceforth we shall devote our attention to connected graphs. Even here we can find examples requiring the edge cover number of voids.

Theorem 2.2. $\xi(P_n) = \lfloor \frac{n-1}{2} \rfloor$.

Proof. When $n \leq 2$ the result is clear, since P_n is complete in these cases. For $n \geq 3$, note that we can consider the scrambled colouring as successive clusters of colours, and the highly limited connectivity of the path ensures that we can never get one colour past another; hence, for any scrambled colouring the best we can do is to select voids to form an edge cover in each colour-cluster. It follows that the worst-case scenario must then involve a 2-colouring that has been sorted into two colour-clusters, thus maximizing the number of conflicts on edges; the size of the clusters must be $\lceil \frac{n}{2} \rceil$ and $\lfloor \frac{n}{2} \rfloor$, and a case analysis of the parities confirms the stated result. □

When n is odd, then, the edge-cover bound is achieved by P_n.

Corollary 2.3. $\xi(C_n) = \lfloor \frac{n-2}{2} \rfloor$.

Proof. The argument for paths holds here as well, since we can still only break up clusters with voids (not having any way of sliding one colour into another cluster). When n is even the sliding number for C_n equals that of P_n by a parallel argument; when n is even we save a void by virtue of the necessity for a third colour. □

Of course, we can find further examples of graphs that satisfy the edge cover bound among those with small edge cover numbers.

Lemma 2.4. $\xi(K_{1,n}) = 1$.

Recall that a double-star is a tree with exactly two vertices of degree greater than 1. We use the notation $D_{m,n}$ to refer to the double-star with vertices of degree $m+1$ and $n+1$; m and n then count the number of leaves adjacent to each support vertex.

Lemma 2.5. *For positive integers $m \leq n$,*

$$\xi(D_{m,n}) = \begin{cases} 1 & \textit{if } m+n \leq 3 \\ 2 & \textit{if } m+n \geq 4 \end{cases}$$

Proof. Let x and y be the vertices of degree $m+1$ and $n+1$, respectively. Clearly $\xi(D_{m,n}) \leq 2 = \beta'(D_{m,n})$ by Lemma 2.1. We note that $D_{m,n}$ has a unique proper 2-colouring, with colour classes of size $m+1$ and $n+1$.

Let $n \geq 3$. Then consider a 2-colouring that assigns the same colour to x, y, and one of each of their leaf-neighbours. (Since $n \geq 3$ we know that one of the colour classes must have size at least 4.) This gives an induced monochromatic P_4, which will require two vertices to be either uncoloured or recoloured to fix. Therefore one void is clearly inadequate. (Even with sliding the void around, in a tree with a single void the best we can do is shift the colours of vertices along a path by one vertex. In this colouring this means that even after such a shift we must still have two adjacent vertices of the same colour.) We can take the same tack with $D_{2,2}$, with the proviso that we must use three colours in order to get a colour class of size 4 to work with.

Note that $D_{1,1} = P_4$ and is therefore covered by Theorem 2.2. For $D_{1,2}$ we can always use one void: if the graph is 2-coloured, then the three vertices in the larger colour class will either induce P_3 or $P_2 \cup P_1$, and in either case a single void is sufficient to obviate the conflict(s). If the two vertices of the smaller colour class are adjacent, then we can always choose the void to be adjacent to one of those vertices, thus allowing a slide to fix the last conflict. (Similar remarks hold in the case of a 3-colouring, and of course with more colours at most one void would be needed.) □

As a side note to the previous proof, notice that any scrambled *minimum* colouring of $D_{2,2}$ requires only one void to fix. To show this, let x and y be the vertices of degree 3, with u, v the private neighbours of x and w, z those of y. If u, v, x are all the same colour then we can uncolour x and slide y's colour onto it. If v and u are both coloured differently from x (whether or not they share a common colour), then any conflict must involve y so uncolouring y gives a resolution. Finally, if v and x share a colour (say 1) and u is coloured differently (say 2), then we uncolour u and slide the 1 from x to u. If y is also coloured 1 then we have achieved a proper partial colouring (since w and z must have colour 2); otherwise we can now slide the colour 2 from y to x.

3 The sliding numbers of trees

We begin our consideration of trees in general by concentrating on those structures that lack vertices of degree 2; our justification is that such vertices can create "bottlenecks" in the tree that will require extra voids to work around. Without such bottlenecks, we can unscramble arbitrary colourings with a small number of voids.

Theorem 3.1. *Let T be a tree containing no vertices of degree 2. Then $\xi(T) \leq 3$.*

Proof. We begin by considering the smallest such tree not covered by Lemma 2.5: the tree T' with three vertices of degree 3 and five of degree 1, where the three 3-valent vertices form a path. In this tree we can uncolour the three high-degree nodes, which gives us an edge-cover solution to the problem. More usefully, we also note that the colours on the remaining five (leaf) nodes can be permuted freely with each other. (That is, we can derive the full group S_5 of permutations of the labels of these five vertices.)

We employ this insight as follows: let T be a tree on at least eight vertices containing no nodes of degree 2, and suppose that $f_0 : V(T) \to V(T)$ is a scrambled colouring of T with f^* a corresponding unscrambled version of f_0. Remove the colours of three vertices in f_0; for the sake of definiteness choose colours that are at or near the centre of T in f^*. We then move the voids to occupy the central vertices. Call this partial colouring g_0.

Choose a vertex $v \in T$ maximally distant from the centre of T with the property that $g_0(v) \neq f^*(v)$. Find the closest instance of the colour $f^*(v)$ in g_0 that is at least as central as v; call the vertex sporting this colour w. (We may assume that, if w and v are equally eccentric, then $f^*(w) \neq g_o(w)$.) We can evolve the colouring g_0 into a new colouring g_1 which is identical save that $g_1(v) = g_0(w)$ and vice-versa in the following way:

1. If $d_T(v, w) \leq 4$ and w does not lie on the path between v and the centre, then there is a subtree T^* of T that is isomorphic to T' and contains v and w as leaves; we can move the voids in from the centre, take advantage of the availability of permutations to swap the colours on v and w, and then move the voids back to the centre. This new partial colouring is g_1.

2. Otherwise, move the three voids so that they lie on the path between v and w, with one adjacent to w. (It is possible that this will require moving the colour already on w, if w lies between v and the centre. If this happens call w_0 the vertex where w's colour lands; if this doesn't happen then $w_0 = w$.)

3. We can now identify a subtree T_0^* of T isomorphic to T' which contains w_0 as a leaf and the three voids as the central vertices. If v is in T_0^* then we can swap the colours on v and w_0 and reset our voids; if not, then find a vertex w_1 in T_0^* that is closer to v but not on the path between v and w_0, and swap the colours on w_0 and w_1.

4. Now move the voids to be between w_1 and v (or as many of them as is possible; we might end up with voids hanging off the side, if $d_T(v, w_1) < 4$). We can

now pick out a subtree T_1^* isomorphic to T' and containing w_1. As before, if $v \in V(T_1^*)$ then we can complete our swap and move the voids back to the centre; if not, then we pick out another intermediary vertex w_2 and swap its colour with w_1, subsequently identifying the tree T_2^*. We repeat this process until we can swap the original colour on w onto v, at which point we shuttle the voids back to the centre. g_1 is the resulting partial colouring.

(Note that in the second case above, involving iterated colour swaps, we can always carefully retrace our steps to return to effectively the original colouring but for the single swap we wished to accomplish; the process of moving voids to a desired location can be seen to be self-reversing if we move the voids back along the same route from which they came.)

In either case, the colouring g_1 is "closer" to f^* than g_0 is, in the sense that either the maximum distance of badly-coloured vertices from the centre is less in g_1 or else that this distance is the same but the number of badly-coloured vertices at this distance is smaller in g_1. Iterating this process leads therefore to a sequence $g_0, g_1, g_2, \ldots$ of colourings that approaches f^*, resulting in a colouring g that agrees with f^* on every coloured vertex. This is a proper partial colouring. □

We have as an easy consequence of this result that any two partial scrambled colourings of such a T with at least three voids and the same colour distribution can be transformed into each other by shifting, as there is nothing in the proof that requires that our reference colouring f^* be proper.

In the presence of degree-2 vertices, our approach above runs into some snags; specifically, if we need to exchange colours that are on opposite sites of a long string of degree-2 vertices, then we will naturally need extra voids in order to maintain transportation. In what follows, let b_T represent the *bottleneck number* of T, defined as the largest number of degree-2 vertices in direct sequence (i.e. if we took the induced subgraph of T on all vertices of degree 2, b_T would be the order of the largest component). Using substantially the same method as in the previous proof, we find:

Corollary 3.2. *For any tree T, $\xi(T) \leq 3 + b_T$.*

This bound is sharp in a great many cases, but under some circumstances it may be possible to do better than this; specifically, imagine a situation where all of the vertices of degree 2 are along a single path, at one end of which is a leaf. (We shall call such a feature a *tail* of T, fearlessly mixing animal and vegetable metaphors.) In this case we don't really think of this as a bottleneck, since we're not trying to use them for transportation; instead, we could use a relatively small number of voids to fix the rest of the tree and then park them in the tail to solve it the way we solved paths in the previous section (i.e. with an edge cover). Such a strategy assumes that there are either no internal bottlenecks, or at least that those present are small compared with the number of tail-packing voids required.

4 The sliding numbers of blocks

In this section we deal with graphs that are 2-connected but not cycles (which were dispensed with earlier). We start with the simplest such graphs, consisting of a cycle with a subdivided chord; we call such a graph a *theta*, for obvious reasons. Let G be a theta with $V(G) = \{h, x, 1, \ldots, i, a_1, \ldots, a_j, b_1, \ldots, b_k\}$, $i \leq j \leq k$; h and x are vertices of degree 3, with three paths between them consisting of the numerical vertices, that a-vertices, and the b-vertices respectively. (See Figure 1 for an example with $i = 2, j = 3, k = 4$; note that we count upwards from h for the numerical vertices, but from x for the other two sequences.)

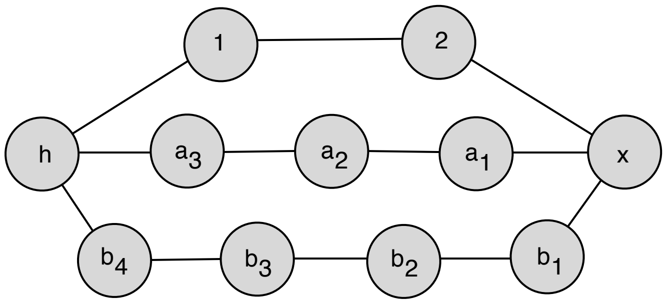

Figure 1: A theta graph

Theorem 4.1. *If G is a theta then $\xi(G) = 1$.*

Proof. Let f be a scrambled proper colouring of G with a corresponding proper colouring f^*. We delete one copy of the colour $f^*(h)$ in f and then slide this void to occupy h; this partial colouring g is our starting point.

Note that any sequence of slides that ends with the void in its starting place gives a permutation on the colours of the (coloured) vertices. Our strategy here is to show that the permutation group available to us in any theta is broad enough that we can unscramble any scrambled proper colouring.

Consider the result of taking the partial colouring g and sliding the void along the following walk: $h, a_j, a_{j-1}, \ldots, a_1, x, i, i-1, \ldots, 1, h$; we denote by A the permutation this gives on the labels of $V(G) \backslash \{h\}$. Then $A = (12 \ldots ixa_1a_2 \ldots a_j)$, a cycle of length $i + j + 1$. Likewise, let B denote the permutation resulting from pushing the void along the walk $h, b_k, b_{k-1}, \ldots, b_1, x, i, i-1, \ldots, 1, h$. Then $B = (12 \ldots ixb_1b_2 \ldots b_k)$, a cycle of length $i + k + 1$. Let $\Gamma = \langle A, B \rangle$. Every element of Γ corresponds to a sequence of slides that can be performed in G that begins and ends with the void at vertex h. We will show that Γ is large enough to unscramble g to a proper partial colouring.

It is easy to show that Γ is primitive on $V(G) \setminus \{h\}$. Suppose that S is a block of Γ containing at least two vertices. Since A is a cycle on the set $\mathcal{A} = \{1, 2, \ldots, i, x, a_j, a_{j-1}, \ldots, a_1\}$ and hence primitive on $\mathcal{A}$, S cannot be a proper subset of $\mathcal{A}$. Analogously, the cycle B shows that S cannot be a proper subset of $\mathcal{B} = \{1, 2, \ldots, i, x, b_j, b_{j-1}, \ldots, b_1\}$. Γ leaves neither $\mathcal{A}$ nor $\mathcal{B}$ intact; therefore $S \neq \mathcal{A}$ and

$S \neq \mathcal{B}$. Hence S must contain at least one a_r and one b_s. If there is some $a_t \notin S$ then some power of A moves a_r to a_t while leaving b_s fixed, which is impossible. Therefore S contains every a_r and b_s. The action of A then forces x and the numbered vertices to also be in S. Therefore, Γ is primitive.

A theorem of Jordan (as found in [2], among other places) states that any primitive permutation group X operating on a set $\mathcal{X}$ that contains a 3-cycle contains the full alternating group on $\mathcal{X}$. If in addition X contains an odd permutation, then X is the full symmetric group on $\mathcal{X}$.

Since Γ is primitive on $V(G) \setminus \{h\}$, it is useful to determine when Γ contains a 3-cycle. Let $C = A^{-1}B = (a_j a_{j-1} \dots a_1 x b_1 b_2 \dots b_k)$, which corresponds to sliding the void along the walk $h, b_k, \dots, b_1, x, a_1, \dots, a_j, h$. Let $D = ABA^{-1}B^{-1}$. We examine Γ by cases, and show that except for exactly one case Γ contains a 3-cycle.

Case $i = 0$**.** Then $1 \leq j \leq k$, and $D = (x a_1 b_1)$.

Case $i = 1$**.** Then $D = (1x)(a_1 b_1)$. If also $j = 1$ then $A = (1 x a_1)$. So assume that $j \geq 2$.

If we also have $k \geq 3$:

$$\begin{aligned} CDC^{-1} &= (1b_1)(xb_2) \\ C^2DC^{-2} &= (1b_2)(b_1b_3) \\ AC^2DC^{-2}A^{-1} &= (xb_2)(b_1b_3) \end{aligned}$$

Then $CDC^{-1} \cdot AC^2DC^{-2}A^{-1} = (1b_1)(xb_2) \cdot (xb_2)(b_1b_3) = (1b_1b_3)$.

The case $i = 1, j = k = 2$ is exceptional and discussed below.

Case $i \geq 2$**.** Then $D = (12)(a_1 b_1)$.

Suppose $i = j = k = 2$:

$$\begin{aligned} CDC^{-1} &= (12)(xb_2) \\ C^2DC^{-2} &= (12)(b_1a_2) \\ DC^2DC^{-2} &= (a_1b_1a_2) \end{aligned}$$

If $k = 3$ then $C^3DC^{-3} = (12)(a_1b_2)$; then $DC^3DC^{-3} = (a_1b_2b_1)$.

Finally if $k \geq 4$ then $C^3DC^{-3} = (12)(b_2b_4)$ and $DC^3DC^{-3} = (xb_2b_4)$.

Therefore, except in the case where $i = 1, j = k = 2$, Γ is either the full symmetric group or the full alternating group on $V(G) \setminus \{h\}$. In the former case, Γ can slide g to any desired partial colouring by selecting the proper permutation and constructing it from A and B. In the latter case, if every colour occurs only once in g then the partial colouring is trivially proper. Otherwise, let y and z be vertices such that $g(y) = g(z)$; then for any vertex $w \notin \{h, y, z\}$ the 3-cycle $(wyz) \in \Gamma$. This results in a permutation that exactly switches the colours of w and y. Jordan's theorem implies that Γ acts as the full symmetric group on the set of partial colourings of G, and hence can unscramble G.

For the remaining case G_θ with $i = 1, j = k = 2$, we constructed Γ with the help of the SAGE software package [3]. In this case $|\Gamma| = 120$. Using brute force, we may show directly that we can unscramble every partial colouring. Details for the strong of stomach are available in the appendix. □

Corollary 4.2. *Let G be a theta not isomorphic to G_θ, and let g_1, g_2 be partial colourings of G with one void and the same colour profile where at least one colour appears twice. Then there is a sequence of moves that converts g_1 to g_2.*

Proof. Given the starting colouring g_1, our first operation is to move the void to one of the degree-3 vertices, which we label h. From here, we have full access to the alternating group of permutations of the colours, if not the full symmetric group; this allows us to arrange the colours arbitrarily using our trick from the previous proof. If g_2 has its void at h, then we can construct it directly; if the void is at some other vertex v, then we construct a colouring with the property that, once we move the void along a chosen path from h to v, the resulting partial colouring is g_2. □

Lemma 4.3. *Let G be a 2-connected graph with $\Delta(G) \geq 3$. Any two vertices of G lie on some subgraph of G isomorphic to a theta.*

Proof. Let $u, v \in V(G)$. We know that u and v must lie on a common cycle C_1, by the definition of 2-connectedness; some vertex (say w) on this cycle must have degree greater than 2. Let x, y, z be neighbours of w with $xw, wy \in E(C_1)$. The edges wy and wz must also be in a common cycle C_2. Then the set of edges given by $C_2 - C_1$ contains one or more paths; let P indicate the path in $C_2 - C_1$ containing the edge wz. Then $C_1 \cup P$ is a theta-subgraph of G. □

Theorem 4.4. *Let G be a 2-connected graph with $\Delta(G) \geq 3$. Then $\xi(G) = 1$.*

Proof. Suppose that f is a scrambled colouring of the vertices of G, with a corresponding proper colouring f^*. If f colours all vertices uniquely then we are done, so assume that at least one colour appears at least twice. Remove a copy of the colour $f^*(h)$ for some maximum-degree vertex h, and move the resulting void to h. By Lemma 4.3, every vertex appears on some theta with h, and hence we can use Corollary 4.2 to unscramble f into f^* piecemeal as long as the theta-subgraph in question is not G_θ. (Note that if we need a colour from outside of the theta we're looking at, we can find a sequence of thetas with pairwise common vertices to transfer the colour through.)

If there are two vertices y, z whose only common theta-subgraph containing h is isomorphic to G_θ, then one of them (say y) must correspond to the vertex we would label 1 in our previous discussion of thetas. □

Corollary 4.5. *Let G be a 2-connected graph with $\Delta(G) \geq 3$ not isomorphic to G_θ, and let g_1, g_2 be partial colourings with one void and the same colour profile where at least one colour appears twice. Then there is a sequence of moves that converts g_1 to g_2.*

Proof. Since we have effective access to the whole group, this follows easily. Another way to approach it is to find some partial proper colouring g^* sharing the colour profile of g_1, which we know we can solve for; we then go from g_1 to g^* and thence to g_2 by reversing a solution for g_2. □

5 General connected graphs

Our strategy for general connected graphs is to combine our methods for (nontrivial) blocks and for trees. Recall that a block decomposition of a graph G is a partition of $E(G)$ into subgraphs which are either single (cut-)edges or maximally 2-connected; by a *block-tree decomposition* we mean the result of taking a block decomposition and merging connected sets of trivial blocks (i.e. edges) into unified components (trees).

Lemma 5.1. *Let G be a graph consisting of two cycles C_1, C_2 that intersect at a single vertex x. Then $\xi(G) = 2$. Further, with two voids we can permute the remaining colours arbitrarily.*

Proof. It is easy to see that $\xi(G) > 1$: let $|C_1| \le |C_2|$ and consider a 2-colouring that assigns the same colour to all vertices in $C_1 - x$. None of these colours can be slid unless the void is positioned at x, and when it is we can only slide them to other vertices in C_1. Since we can never remove these colours from C_1 we clearly cannot replace them with the other colour (which is only in C_2).

Now consider the situation with two voids. We shall use u_1, v_1, u_2, v_2 to refer to the neighbours of x, with subscripts indicating the containing block. Without loss of generality suppose that we have positioned the voids at x and at v_2; then we can rotate the colours in C_1 around to a point where a colour we wish to dispose of is positioned at v_1, from which it can be taken to v_2 in two slides. This now leaves voids at x and v_1, and we can rotate the colours in C_1 in such a way as to position the voids between any two colours in the sequence. This allows us to carry out successive exchanges between C_1 and C_2 to achieve any permutation we wish of the colours available to us; specifically, we can achieve an unscrambled colouring. □

Theorem 5.2. *Let G be a graph with $\kappa'(G) \ge 2$ and $\Delta(G) \ge 3$. Then $\xi(G) \le 2$.*

Proof. We have already dealt with the case where $\kappa(G) \ge 2$ in Theorem 4.4, so assume that G contains at least one cut-vertex x. If G_1, G_2 are blocks of G containing x, then we can find a cycle C_i in each G_i and use Lemma 5.1 to set up an interchange between C_1 and C_2. This allows us to move colours between blocks; if we want to trade colours that are in nonadjacent blocks then we can repeat this process through a sequence of cutpoints $x, x', x'', \ldots$ that lie on a path between the two target vertices.

The strategy for unscrambling is as follows: choose a terminal block B_1 (that is, a block with interior vertices on no path between other blocks) and determine what colours we wish it to contain (and in what positions). We can use Lemma 5.1 repeatedly in the method just described to bring the colours to (or just outside of) B_1; once we have the desired colours in place we can use either Corollary 4.5 or

Lemma 5.1 to give B_1 its correct colouring. (We may need the lemma if B_1 does not meet the requirements of the corollary: that is, if B_1 is either a cycle or isomorphic to G_θ. In either of these cases, we can swap "dummy colours" from a block adjacent to B_1 in and out to achieve the required permutations.) Once B_1 is fixed then we never need to disturb it again (except possibly for its cut-vertex); we then find another terminal block B_2 in (the nontrivial component of) the graph $G - E(B_1)$ and repeat. We continue recursively until we are left with either one block that satisfies the premises of Corollary 4.5 or else two adjacent blocks that don't; in the latter case we can use Lemma 5.1 to achieve the final colouring. □

Theorem 5.3. *Let G be a connected graph with the property that no tree in its block-tree decomposition contains a vertex of degree 2. Then $\xi(G) \leq 3$ unless G is a cycle.*

Proof. If the block-tree decomposition of G contains no trees at all then Theorem 5.2 applies here, so assume that there is at least one tree in the decomposition. We distinguish between a *terminal* tree (which shares only a single vertex with one or more blocks) and a *transitional* tree (which provides connectivity between multiple blocks). In either case, if T_1 is a tree in the decomposition that shares a vertex x with a block B_1, then we may in some sense consider the neighbours of x in B_1 as being part of T_1 for the purposes of running our procedures.

As in the previous result, our strategy is recursive: arrange the colours required in a terminal segment (block or tree) which we can subsequently ignore. With a terminal block, we can use a mix of Theorems 5.2 and 3.1 to move colours into it (and then to arrange them as required). With a terminal tree, we use the same two theorems to transition colours into the neighbourhood of the tree, and then Theorem 3.1 (treating the linking cut-vertex as the root) to arrange the colours in the tree. □

We define the bottleneck number b_G of a general graph as:

$$b_G = \max\{b_T : T \text{ is a tree in the block-tree decomposition of } G\}$$

(We can ignore degree 2 vertices in the nontrivial blocks of the graph, since they can easily be routed around.) As in our earlier discussion, in order to maintain transportation we need additional voids to deal with bottlenecks; likewise as before, we can move the voids between bottlenecks as we need to get through.

Corollary 5.4. *For any connected graph G that is not a cycle, $\xi(G) \leq 3 + b_G$.*

6 Computational complexity

In reckoning the complexity of our procedures, we assume that the solver has a proper colouring of the graph (with the same colour profile as the scrambled colouring) for reference; without such a colouring to guide us, solving a sliding-colours problem could be as difficult as determining the chromatic number of G.

Given a reference colouring, we measure the complexity of a solution (unscrambling to a partial analogue of the given colouring) in terms of the number of slides required. We make no claim that our methods as described above are optimal by this metric; they are, however, good enough for polytime. In the following results n is the number of vertices in the graph under discussion.

Theorem 6.1. *Unscrambling a tree T containing no vertices of degree 2 can be accomplished in $O(n^2)$ slides.*

Proof. The process of switching the colours on two vertices is $O(n)$, since permuting two labels within a copy of T' is $O(1)$ and we may be required to do this a number of times equal to a linear function of the tree's diameter. Since each switch that we make only guarantees that we reduce the incorrectly coloured vertex count by one or two, the number of such swaps is again $O(n)$. □

Corollary 6.2. *Unscrambling a tree T with bottleneck number b_T can by accomplished in $O(n^2 b_T)$ slides using $3 + b_T$ voids.*

Proof. The analysis parallels the one above; the main difference is that switching vertices within a subdivided T' (i.e. a segment containing bottlenecks) could require up to $O(b_T)$ slides rather than $O(1)$. □

Theorem 6.3. *Unscrambling a 2-connected graph G can be accomplished in $O(n^4)$ slides with a minimum number of voids.*

Proof. Assume that G contains a theta, since otherwise we have a cycle that requires no slides at all using the edge-cover bound. Executing the generating cycles of the permutations in a theta on k vertices is $O(k)$, and from the generating cycles we can construct any 3-cycle in $O(k)$ operations. We can construct any required (even) permutation on a theta with $O(k)$ 3-cycles; therefore fixing a theta on k vertices requires $O(k^3)$ slides. The number of distinct thetas that we might need to work with will be $O(n)$, each with $O(n)$ vertices, giving us the $O(n^4)$ bound for the number of slides required with a single void. □

Assembling the above pieces in the natural way gives us:

Theorem 6.4. *Unscrambling a general graph G using $3 + b_G$ voids can be accomplished in $O(n^4)$ slides.*

Appendix: Unscrambling the (1,2,2)-theta

Recall that the graph G_θ, the theta-graph with vertex set $\{h, x, 1, a_1, a_2, b_1, b_2\}$, was too small for our general methods to work. We used SAGE to find that its automorphism group Γ has order 120 and includes the element $(a_1b_1)(a_2b_2)$, a fact that reduces the number of cases we are required to consider. We also know that $\alpha(G_\theta) = \chi(G_\theta) = 3$, which reduces the number of partitions of 7 (as distributions of colours in a proper colouring) that we need to consider. In what follows we use the notation zK to indicate that vertex z receives colour K. We remind the reader that our basic operations on thetas in this case take the form $A = (1xa_1a_2)$, $B = (1xb_1b_2)$, and $C = A^{-1}B = (1b_2b_1a_1a_2)$.

Our cases are structured around the number of colours n in the original colouring. $n = 7$ is always proper, and therefore trivial; if $n = 6$ then either the colouring is proper or we may create our void by deleting one of the copies of the duplicate colour.

For $n = 5$, the only possible partitions for the colouring are (3,1,1,1,1) and (2,2,1,1,1). In either case, we can select a vertex to clear that leaves a partial colouring with distribution (2,1,1,1,1); let P represent the duplicate colour. If we move the void to h and the result is not a proper colouring, then there are essentially three possible locations for the conflict (up to the symmetry of the a and the b paths):

- $1P, xP$: applying A^{-1} creates a proper colouring.
- a_1P, xP: applying B creates a proper colouring.
- a_1P, a_2P: applying A creates a proper colouring.

For $n = 4$, the possible colour distributions are (3,2,1,1) and (2,2,2,1); in either case we may delete a colour to get the partial colour distribution (2,2,1,1). Let P and Q be the duplicate colours; as usual, we move the void to h. We can use the methods in the previous case to ensure that we have no adjacent vertices coloured with Q; this leaves us the following possibilities to investigate (up to symmetry, again):

P **vertices**	Q **vertices**	**Move sequence**
$1, x$	a_1, b_1	C
	a_1, b_2	C
	a_2, b_2	A
a_1, x	$1, a_2$	B
	$1, b_1$	B
	$1, b_2$	A^2
	a_2, b_1	B
	a_2, b_2	B
a_1, a_2	Any	A to reduce to a previous case

For $n = 3$, the possible colour distributions are (3,3,1) and (3,2,2); we can always remove a colour to give us a partial colouring with distribution (3,2,1). Let P, Q, R represent the colours in descending order of frequency; as usual, we move to void to h to start with. As in the case where $n = 4$, we can always find a configuration where the only conflicts are between P-vertices. That gives us the following possibilities (once more modulo the symmetry between the as and the bs), where "reduce" means reducing a two-conflict configuration to a one-conflict configuration:

$P-P$ edge count	**P vertices**	**R vertex**	**Move sequence**
1	$1, x, a_2$	Any (b_1 or b_2)	C^{-1}
	x, a_1, b_2	Any	B
	$1, a_1, a_2$	b_1	C
	a_1, a_2, b_1	Any	C
	a_1, a_2, b_2	x	BA
2	$1, x, a_1$	Any (b_1 or b_2)	A to reduce
	x, a_1, b_1	1	A to reduce
		a_2	A to reduce
		b_2	B^{-1} to reduce
	x, a_1, a_2	b_1	A
		b_2	B^{-1}

References

[1] S.A. Clark, J.E. Holliday, S.H. Holliday, P.D. Johnson Jr., J.E. Trimm, R.R. Rubalcaba and M. Walsh, Chromatic villainy in graphs, *Proc. Thirty-Seventh Southeastern Int. Conf. Combin., Graph Theory, Computing*, *Congressus Numerantium* 182 (2006), 171–182.

[2] J.D. Dixon and B. Mortimer, *Permutation Groups*, Springer-Verlag, New York — Heidelberg — Berlin, 1996.

[3] `http://www.sagemath.org/doc/index.html`

(Received 14 May 2013; revised 18 Nov 2013)