The University of Sydney
Basser Department of Computer Science
Algorithmic Applications Group

Technical Report CS-AAG-2001-05

Brooks' Vertex-Colouring Theorem in Linear Time *

Bradley Baetz[†] and David R. Wood[‡]
Basser Department of Computer Science
The University of Sydney
Sydney NSW 2006
Australia

October 24, 2001

Abstract

Brooks' Theorem [R. L. Brooks, On Colouring the Nodes of a Network, *Proc. Cambridge Philos. Soc.* **37**:194-197, 1941] states that every graph G with maximum degree Δ, has a vertex-colouring with Δ colours, unless G is a complete graph or an odd cycle, in which case $\Delta + 1$ colours are required. Lovász [L. Lovász, Three short proofs in graph theory, *J. Combin. Theory Ser.* **19**:269-271, 1975] gives an algorithmic proof of Brooks' Theorem. Unfortunately this proof is missing important details and it is thus unclear whether it leads to a linear time algorithm. In this paper we give a complete description of the proof of Lovász, and we derive a linear time algorithm for determining the vertex-colouring guaranteed by Brooks' Theorem.

Let $G = (V, E)$ be a simple graph with maximum degree Δ. Undefined graph-theoretic terms can be found in [3]. A *vertex-colouring* of G is a mapping from the vertex set of G to some set of colours such that adjacent vertices receive different colours. For convenience we take the set of colours to be the positive integers $\{1, 2, \dots\}$. A graph is said to be *k-colourable* if it has a vertex-colouring with at most k colours. Minimising the number of colours in a vertex-colouring

*Partially completed while both authors were visiting the School of Computer Science, McGill University, Montréal, Canada.

[†]E-mail: `bbaetz@ug.cs.usyd.edu.au`.

[‡]Supported by the Australian Research Council Large Grant A49906214. E-mail: `davidw@cs.usyd.edu.au`.

of a given graph is a fundamental problem in algorithmic graph theory with applications in register allocation for example. Unfortunately, determining if a given graph is k-colourable is NP-complete [4]. The sequential greedy algorithm, which chooses for each vertex v in turn, the minimum colour not used by a neighbour of v, will use at most $\Delta + 1$ colours, since at each vertex v there is at most Δ different colours assigned to the neighbours of v. Brooks [1] proved the following improvement to this result.

Theorem 1 (Brooks' Theorem [1]). *Every graph G with maximum degree Δ, has a vertex-colouring with Δ colours, unless G is a complete graph or an odd cycle, in which case $\Delta + 1$ colours are required.*

We say a vertex-colouring of graph G with maximum degree Δ is a *Brooks-colouring* if the number of colours is at most Δ, or $\Delta + 1$ if G is a complete graph or an odd cycle.

The original proof by Brooks leads to a quadratic time algorithm for calculating a Brooks-colouring. Since then Ponstein [6] and Lovász [5] (also see Bryant [2]) describe algorithmic proofs of Brooks' Theorem. However, the running time of the resulting algorithms are not analysed, and in the proof by Lovász [5], many important details are omitted.

In this paper, we give a complete description of the proof of Brooks' Theorem due to Lovász [5], and derive a linear time algorithm for computing a Brooks-colouring. We start with the following well-known result, which can be proved by performing a pre-order traversal of the block-cut-forest of the given graph, and possibly swapping two colours in each biconnected component.

Lemma 1. *Given k-colourings of the biconnected components of a graph $G = (V, E)$, a k-colouring of G can be determined in $O(|V| + |E|)$ time.* □

Lemma 1 implies that we need only describe a linear time algorithm for determining a Brooks-colouring in the case of a biconnected graph.

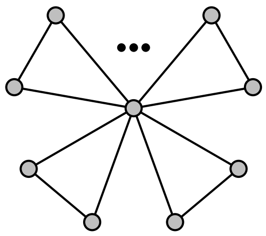
Lemma 2. *If a graph G with maximum degree Δ contains two vertices a and b at distance 2 such that $G \setminus \{a, b\}$ is connected, then a Δ-colouring of G can be determined in $O(|V| + |E|)$ time.*

Proof. Let $(v_1, v_2, \ldots, v_n)$ be an ordering of the vertices of $G \setminus \{a, b\}$ such that v_1 is a vertex adjacent to both a and b, and for every $i \geq 2$, the vertex v_i has at least one neighbour v_j with $j < i$. Since $G \setminus \{a, b\}$ is connected, such an ordering can be determined by a depth-first search of $G \setminus \{a, b\}$. Let the colour of a and b be 1. This is valid since a and b are not adjacent. Now, for each i, $i = n, n-1, \ldots, 1$, colour the vertex v_i with the minimum positive integer which is different from the colours assigned to the neighbours of v_i which are already coloured. For each i, $1 \leq i \leq n-1$, the colour assigned to v_i is at most Δ since there are at most $\Delta - 1$ neighbours of v_i already coloured. The colour assigned to v_a is at most Δ since v_1 has two neighbours (namely, a and b) receiving the same colour. Thus, we have a vertex-colouring of G with at most Δ colours. This procedure can be implemented in $O(|V| + |E|)$ time using standard depth-first search algorithms. □

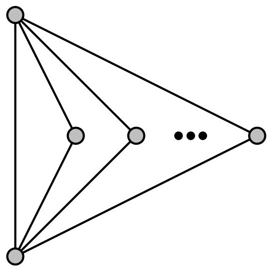
Lemma 3. *Let $G = (V, E)$ be a biconnected graph which is not a complete graph or a cycle. Then vertices a and b at distance 2 in G can be found in $O(|V| + |E|)$ time such that $G \setminus \{a, b\}$ is connected.*

Proof. Let $n = |V|$. Since G is biconnected and not a 3-cycle, $n \geq 4$.

Suppose every vertex v has degree 2 or degree $n-1$. Since G is not a cycle, at least one vertex has degree $n-1$. Thus, and since G is biconnected, at least two vertices have degree $n-1$, as otherwise G would be the 1-connected graph shown in Figure (a). Since G is not a complete graph there is at least one vertex of degree 2, which implies there is exactly two vertices of degree $n-1$; that is, G is $K_{1,1,n-2}$, as shown in Figure (b). Since G is not a 3-cycle there are at least two vertices of degree 2. Let a and b be any two degree 2 vertices. Then $G \setminus \{a, b\}$ is connected, and we are done. Clearly this case can be recognised and the vertices a and b determined in $O(|V| + |E|)$ time.



(a)



(b)

Otherwise G has a vertex x with $3 \leq \deg(x) \leq n-1$. We now consider two cases depending on the biconnectivity of $G \setminus x$. First, suppose $G \setminus x$ is biconnected. Let $a = x$ and b be any vertex at distance 2 from x. Then $G \setminus \{a, b\}$ is connected, and we are done. Second, suppose $G \setminus x$ is not biconnected. Since G is biconnected, $G \setminus x$ is connected. Let B_1 and B_2 be end-blocks of $G \setminus x$ with respective cut-points z_1 and z_2. (An end-block corresponds to a leaf of the block-cut-tree, and since a tree has at least two leaves, B_1 and B_2 exist.) Since G is biconnected but $G \setminus x$ is not biconnected, x must be adjacent to vertices in B_1 and B_2 which are not z_1 and z_2. Let a and b be these vertices. The only vertex adjacent to both a and b is x, and since $\deg(x) \geq 3$, $G \setminus \{a, b\}$ is connected. Again, this algorithm can be implemented in $O(|V|+|E|)$ time using depth-first search algorithms for determining biconnectivity and the biconnected components of a graph [7]. $\square$

Of course cycles (both odd and even) and complete graphs can be recognised and Brooks-colourings determined in linear time. Combining this observation with Lemma 1, Lemma 2 and Lemma 3, we obtain the following result.

Theorem 2. *There is an algorithm to determine a Brooks-colouring of a given graph $G = (V, E)$ in $O(|V| + |E|)$ time.* $\square$

References

[1] R. L. Brooks, On colouring the nodes of a network. *Proc. Cambridge Philos. Soc.*, **37**:194–197, 1941.

[2] V. Bryant, A characterisation of some 2-connected graphs and a comment on an algorithmic proof of Brooks' theorem. *Discrete Math.*, **158**:279–281, 1996.

[3] G. Chartrand and L. Lesniak, *Graphs & Digraphs*. Chapman and Hall, 1996.

[4] R. M. Karp, Reducibility among combinatorial problems. In *Complexity of Computer Communications*, pp. 85–103, Plenum Press, 1972.

[5] L. Lovász, Three short proofs in graph theory. *J. Combin. Theory Ser. B*, **19**:269–271, 1975.

[6] J. Ponstein, A new proof of Brooks's chromatic number theorem for graphs. *J. Combinatorial Theory*, **7**:255–257, 1969.

[7] R. Tarjan, Depth-first search and linear graph algorithms. *SIAM J. Comput.*, **1(2)**:146–160, 1972.