Brief Announcement: Linial's Lower Bound Made Easy

Juhana Laurinharju
Helsinki Institute for Information Technology HIIT,
Department of Computer Science,
University of Helsinki, Finland
juhana.laurinharju@cs.helsinki.fi

Jukka Suomela
Helsinki Institute for Information Technology HIIT,
Department of Information and Computer Science,
Aalto University, Finland
jukka.suomela@aalto.fi

ABSTRACT

Linial's seminal result shows that any deterministic distributed algorithm that finds a 3-colouring of an n-cycle requires at least $\log^*(n)/2 - 1$ communication rounds. We give a new simpler proof of this theorem.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems; F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems

Keywords

Cycle; distributed algorithm; graph coloring; iterated logarithm; LOCAL model; lower bound

1. INTRODUCTION

Linial's lower bound for 3-colouring directed cycles [2] is one of the most celebrated results in the area of distributed graph algorithms. It is cited in hundreds of papers and the proof has been reproduced in textbooks and lecture notes [1, 3–5]. Yet it it seems that typical presentations of this result either follow the structure of Linial's original proof [1, 3, 5], or rely on some prior knowledge of Ramsey's theorem [4].

In this work we give a simpler, self-contained version of Linial's proof. This version of the proof is easy to explain to a student on a whiteboard in fifteen minutes. We do not need to refer to neighbourhood graphs, line graphs, and chromatic numbers.

2. PROBLEM FORMULATION

Fix a natural number n. We are interested in deterministic distributed algorithms that find a proper 3-colouring of any directed n-cycle. We use the LOCAL model [3]: in each round each node can send arbitrarily large messages to each of its neighbours. The nodes are labelled with unique identifiers from the set $\{1, 2, \ldots, n\}$. Each node must pick its own colour from the set $\{1, 2, 3\}$. The orientation of the edges is just auxiliary information that the algorithm can use; messages can be sent in both directions.

If we have a distributed algorithm with a running time of T communication rounds, then each node has to pick its own colour based on the information that is available within distance T from it; see Figure 1. Moreover, two nodes that are adjacent to each other must pick different colours. Hence the algorithm is a function A with $2T + 1$ arguments that satisfies

$$A(x_1, x_2, \ldots, x_{2T+1}) \in \{1, 2, 3\},$$
$$A(x_1, x_2, \ldots, x_{2T+1}) \neq A(x_2, x_3, \ldots, x_{2T+2})$$

whenever $x_1, x_2, \ldots, x_{2T+2}$ are distinct identifiers from the set $\{1, 2, \ldots, n\}$.

Function $\log^* x$ is the iterated logarithm of x, defined as follows: $\log^* x = 0$ if $x \leq 1$, and $\log^* x = 1 + \log^* \log_2 x$ otherwise. Linial's famous result shows that no matter which algorithm A we pick, we must have

$$T \geq \frac{1}{2} \log^*(n) - 1. \tag{1}$$

We will now give a simple proof of this theorem.

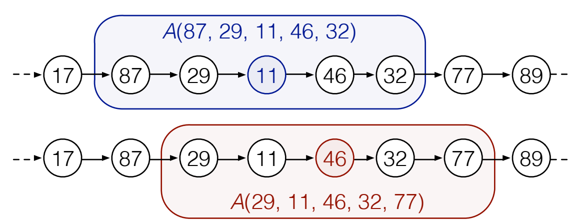

Figure 1: Colouring directed cycles in time $T = 2$. For example, the output of node 11 only depends on its radius-T neighbourhood, $(87, 29, 11, 46, 32)$. We can interpret algorithm A as a k-ary function, $k = 2T + 1 = 5$, that maps each local neighbourhood to a colour. As it is possible that adjacent nodes have neighbourhoods $(87, 29, 11, 46, 32)$ and $(29, 11, 46, 32, 77)$, function A must satisfy $A(87, 29, 11, 46, 32) \neq A(29, 11, 46, 32, 77)$.

PODC'14, July 15–18, 2014, Paris, France.
ACM 978-1-4503-2944-6/14/07.
http://dx.doi.org/10.1145/2611462.2611505.

3. COLOURING FUNCTIONS

The only concept that we need is a *colouring function*. We say that A is a k-ary c-colouring function if

$$\begin{gathered}A(x_1, x_2, \dots, x_k) \in \{1, 2, \dots, c\} \\ \text{for all } 1 \le x_1 < x_2 < \dots < x_k \le n,\end{gathered} \tag{2}$$

$$\begin{gathered}A(x_1, x_2, \dots, x_k) \neq A(x_2, x_3, \dots, x_{k+1}) \\ \text{for all } 1 \le x_1 < x_2 < \dots < x_{k+1} \le n.\end{gathered} \tag{3}$$

Any deterministic distributed algorithm A that finds a proper 3-colouring of an n-cycle defines a k-ary 3-colouring function for $k = 2T + 1$. (Note that the converse is not necessarily true. Colouring functions do not need to be valid algorithms. We do not care what happens if, e.g., $x_1 > x_2$.)

We will show that $k + 1 \ge \log^* n$ for any k-ary 3-colouring function. By plugging in $k = 2T + 1$, we obtain the main result (1).

4. PROOF

We begin with a trivial lemma: if a colouring function only sees 1 identifier, it cannot do much.

Lemma 1. *If A is a 1-ary c-colouring function, we have $c \ge n$.*

Proof. If $c < n$, by the pigeonhole principle there are some $x_1 < x_2$ with $A(x_1) = A(x_2)$, which contradicts (3). $\square$

The key part of the proof is the following observation: given any colouring function A, we can always construct another colouring function B that is "faster" (smaller number of arguments) but "worse" (larger number of colours). Here it is crucial that colouring functions are well-defined for both odd and even values of k.

Lemma 2. *If A is a k-ary c-colouring function, we can construct a $(k-1)$-ary 2^c-colouring function B.*

Proof. We define B as follows:

$$\begin{aligned}&B(x_1, x_2, \dots, x_{k-1}) \\ &\quad = \bigl\{A(x_1, x_2, \dots, x_{k-1}, x_k) : x_k > x_{k-1}\bigr\}.\end{aligned}$$

There are only 2^c possible values of B: all possible subsets of $\{1, 2, \dots, c\}$. These can be represented as integers $\{1, 2, \dots, 2^c\}$, and hence (2) holds.

The interesting part is (3). Let $1 \le x_1 < x_2 < \dots < x_k \le n$. By way of contradiction, suppose that

$$B(x_1, x_2, \dots, x_{k-1}) = B(x_2, x_3, \dots, x_k). \tag{4}$$

Let $\alpha = A(x_1, x_2, \dots, x_k)$.

From the definition of B we have $\alpha \in B(x_1, x_2, \dots, x_{k-1})$. By assumption (4), this implies $\alpha \in B(x_2, x_3, \dots, x_k)$. But then we must have some $x_k < x_{k+1} \le n$ such that

$$\alpha = A(x_2, x_3, \dots, x_{k+1}).$$

That is, A cannot be a colouring function. $\square$

To complete the proof, we will need power towers. Define

$$^{i}2 = 2^{2^{\cdot^{\cdot^{2}}}}$$

with i twos in the power tower. For example, $^{2}2 = 4$ and $^{3}2 = 16$. Now assume that A_1 is a k-ary 3-colouring function. Certainly it is also a k-ary $^{2}2$-colouring function. We can apply Lemma 2 iteratively to obtain

- a $(k-1)$-ary $^{3}2$-colouring function A_2,
- a $(k-2)$-ary $^{4}2$-colouring function A_3,

  ...
- a 1-ary $^{k+1}2$-colouring function A_k.

By Lemma 1, we must have $^{k+1}2 \ge n$, which implies $k + 1 \ge \log^* n$.

5. ACKNOWLEDGEMENTS

This work was supported in part by the Academy of Finland, Grant 252018, and by the Research Funds of the University of Helsinki. Many thanks to Barbara Keller, Stefan Schmid, Jara Uitto, and anonymous reviewers for comments and helpful feedback, and to all students and colleagues who have kindly served as guinea pigs.

6. REFERENCES

[1] Leonid Barenboim and Michael Elkin. *Distributed Graph Coloring: Fundamentals and Recent Developments.* Morgan & Claypool, 2013. doi:10.2200/S00520ED1V01Y201307DCT011.

[2] Nathan Linial. Locality in distributed graph algorithms. *SIAM Journal on Computing*, 21(1):193–201, 1992. doi:10.1137/0221015.

[3] David Peleg. *Distributed Computing: A Locality-Sensitive Approach.* SIAM Monographs on Discrete Mathematics and Applications. Society for Industrial and Applied Mathematics, Philadelphia, 2000.

[4] Jukka Suomela. *A Course on Deterministic Distributed Algorithms.* 2014. Online textbook. http://users.ics.aalto.fi/suomela/dda/.

[5] Roger Wattenhofer. Lecture notes on principles of distributed computing, 2013. http://dcg.ethz.ch/lectures/podc_allstars/.