Theoretical Computer Science 610 (2016) 204–217

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Deterministic local algorithms, unique identifiers, and fractional graph colouring

Henning Hasemann [a], Juho Hirvonen [b,c,d], Joel Rybicki [b,c,d], Jukka Suomela [b,c,d,*]

[a] Institute of Operating Systems and Computer Networks, TU Braunschweig, Germany [1,2]
[b] Helsinki Institute for Information Technology HIIT, Finland [1,2]
[c] Department of Information and Computer Science, Aalto University, Finland [1]
[d] Department of Computer Science, University of Helsinki, Finland [2]

ARTICLE INFO

Article history:
Received 1 October 2012
Received in revised form 19 February 2014
Accepted 29 June 2014
Available online 7 July 2014

Keywords:
Distributed algorithms
Fractional domatic partition
Fractional graph colouring
Local algorithms
Unique identifiers

ABSTRACT

In the fractional graph colouring problem, the task is to schedule the activities of the nodes so that each node is active for 1 time unit in total, and at each point of time the set of active nodes forms an independent set.

We show that for any $\alpha > 1$ there exists a deterministic distributed algorithm that finds a fractional graph colouring of length at most $\alpha(\Delta + 1)$ in any graph in one synchronous communication round; here Δ is the maximum degree of the graph. The result is near-tight, as there are graphs in which the optimal solution has length $\Delta + 1$.

The result is, of course, too good to be true. The usual definitions of scheduling problems (fractional graph colouring, fractional domatic partition, etc.) in a distributed setting leave a loophole that can be exploited in the design of distributed algorithms: the size of the local output is not bounded. Our algorithm produces an output that seems to be perfectly good by the usual standards but it is impractical, as the schedule of each node consists of a very large number of short periods of activity.

More generally, the algorithm demonstrates that when we study distributed algorithms for scheduling problems, we can choose virtually any trade-off between the following three parameters: T, the running time of the algorithm, ℓ, the length of the schedule, and κ, the maximum number of periods of activity for any single node. Here ℓ is the objective function of the optimisation problem, while κ captures the "subjective" quality of the solution. If we study, for example, bounded-degree graphs, we can trivially keep T and κ constant, at the cost of a large ℓ, or we can keep κ and ℓ constant, at the cost of a large T. Our algorithm shows that yet another trade-off is possible: we can keep T and ℓ constant at the cost of a large κ.

1. Introduction

In the study of *deterministic distributed algorithms*, it is commonly assumed that there are *unique numerical identifiers* available in the network: in an n-node network, each node is labelled with a unique $O(\log n)$-bit number.

* Corresponding author.
[1] Current affiliations.
[2] Where most of this work was done.

http://dx.doi.org/10.1016/j.tcs.2014.06.044
0304-3975/© 2014 Elsevier B.V. All rights reserved.

In the general case, numerical identifiers are, of course, very helpful—many fast distributed algorithms crucially depend on the existence of numerical identifiers, so that they can use the Cole–Vishkin technique [2] and similar tricks. However, when we move towards the fastest possible distributed algorithms, the landscape looks very different.

1.1. Local algorithms and numerical identifiers

We focus on *local algorithms* [9,12], i.e., distributed algorithms that run in constant time (a constant number of communication rounds), independently of the size of the network. In this context, it is no longer obvious if unique identifiers are of any use:

1. In their seminal work, Naor and Stockmeyer [9] prove that there is a class of problems—so-called LCL problems—that do not benefit from unique numerical identifiers: if an LCL problem can be solved with a local algorithm, it can also be solved with an *order-invariant* local algorithm. Order-invariant algorithms do not exploit the numerical value of the identifier; they merely compare the identifiers with each other and use the relative order of the identifiers.
2. More recently, Göös et al. [3] have shown that for a large class of optimisation problems—so-called PO-checkable problems—local algorithms do not benefit from any kind of identifiers: if a PO-checkable optimisation problem can be approximated with a local algorithm, the same approximation factor can be achieved in anonymous networks if we are provided with a port-numbering and an orientation.

While the precise definitions of LCL problems and PO-checkable problems are not important here, they both share the following seemingly technical requirement: it is assumed that the *size of a local output is bounded by a constant* (here the size refers to the number of bits in the encoding of the local output). That is, for each node in the network, there is only a constant number of possible local outputs, independently of the size of the network. However, previously it has not been known whether this is a necessary condition or merely a proof artefact—while contrived counter-examples exist, natural counter-examples have been lacking.

1.2. Contributions

In this work we provide the missing piece of the puzzle: we show that the condition is necessary, even if we focus on natural graph problems and natural encodings of local outputs. More precisely, we show that there is a classical graph problem—namely, *fractional graph colouring* (see Section 2)—with the following properties:

1. In a natural problem formulation, the local outputs can be arbitrarily large.
2. The problem can be solved with a deterministic local algorithm; the algorithm exploits both numerical identifiers and unbounded local outputs.
3. The problem cannot be solved with a deterministic local algorithm without numerical identifiers.
4. The problem cannot be solved with a deterministic local algorithm if we require that the local outputs are of a constant size.

Moreover, this is not an isolated example. The same holds for many other scheduling problems—for example, *fractional domatic partitions* have similar properties (see Section 7). It is up to the reader's personal taste whether this work should be interpreted as a novel technique for the design of local algorithms, or as a cautionary example of a loophole that needs to be closed.

The present work is an extended and revised version of a preliminary conference report [4]. In comparison with the conference version, the material related to fractional domatic partitions is new.

1.3. Comparison with other graph problems

In the study of local algorithms, one often has to make some assumptions on the graph family [5–8]. The most commonly used assumption is to focus on bounded-degree graphs.

If we have a constant maximum degree Δ, then a constant-size local output is a very natural property that is shared by a wide range of combinatorial graph problems—at least if we use a natural encoding of the solution:

1. *Independent set, vertex cover, dominating set, connected dominating sets, etc.*: The output is a subset $X \subseteq V$ of nodes. Each node outputs 1 or 0, indicating whether it is part of X.
2. *Matching, edge cover, edge dominating set, spanning subgraphs, etc.*: The output is a subset $Y \subseteq E$ of edges. A node of degree d outputs a binary vector of length d, with one bit for each incident edge.
3. *Vertex colouring, domatic partition, minimum cut, maximum cut, etc.*: The output is a partitioning of nodes, $X_1 \cup X_2 \cup \cdots \cup X_k = V$. Each node outputs an integer $i \in \{1, 2, \ldots, k\}$, indicating that it belongs to subset X_i. In most cases, there is a natural constant upper bound on k: for example, a vertex colouring does not need more than $\Delta + 1$ colours, a domatic partition cannot contain more than $\Delta + 1$ disjoint dominating sets, and a cut by definition has $k = 2$.