Journal of Computer and System Sciences ••• (••••) •••–•••

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Optimal-depth sorting networks

Daniel Bundala [a], Michael Codish [b], Luís Cruz-Filipe [c,*], Peter Schneider-Kamp [c], Jakub Závodný [a]

[a] *Department of Computer Science, University of Oxford, United Kingdom*
[b] *Department of Computer Science, Ben-Gurion University of the Negev, Israel*
[c] *Department of Mathematics and Computer Science, University of Southern Denmark, Denmark*

ARTICLE INFO

Article history:
Received 6 February 2015
Received in revised form 20 August 2016
Accepted 22 September 2016
Available online xxxx

Keywords:
Sorting networks
SAT solving
Symmetry breaking

ABSTRACT

We solve a 40-year-old open problem on depth optimality of sorting networks. In 1973, Donald E. Knuth detailed sorting networks of the smallest depth known for $n \leq 16$ inputs, quoting optimality for $n \leq 8$ (Volume 3 of "*The Art of Computer Programming*"). In 1989, Parberry proved optimality of networks with $9 \leq n \leq 10$ inputs. We present a general technique for obtaining such results, proving optimality of the remaining open cases of $11 \leq n \leq 16$ inputs. Exploiting symmetry, we construct a small set R_n of two-layer networks such that: if there is a depth-k sorting network on n inputs, then there is one whose first layers are in R_n. For each network in R_n, we construct a propositional formula whose satisfiability is necessary for the existence of a depth-k sorting network. Using an off-the-shelf SAT solver we prove optimality of the sorting networks listed by Knuth. For $n \leq 10$ inputs, our algorithm is orders of magnitude faster than prior ones.

1. Introduction

General-purpose sorting algorithms are based on comparing and exchanging pairs of inputs. If the order of these comparisons is predetermined by the number of inputs to sort and does not depend on their concrete values, then the algorithm is said to be data oblivious. Such algorithms are well-suited for e.g. parallel sorting or secure multi-party computations, unlike standard sorting algorithms, such as QuickSort, MergeSort or HeapSort, where the order of comparisons performed depends on the input data.

Sorting networks are a classical formal model for data-oblivious algorithms [19], where n inputs are fed into networks of n channels connected pairwise by comparators. Each comparator takes the two inputs from its two channels, compares them, and outputs them sorted back to the same two channels. A set of consecutive comparators can be viewed as a "parallel layer" if no two comparators act on the same channel. A comparator network is a sorting network if the output on the n channels is always the sorted sequence of the inputs.

Ever since sorting networks were introduced, there has been a quest to find optimal sorting networks: optimal size (minimal number of comparators), as well as optimal depth (minimal number of layers) networks. In their celebrated result, Ajtai, Komlós and Szemerédi [1] give a construction for sorting networks with $O(n \log n)$ comparators in $O(\log n)$ parallel levels. These AKS sorting networks are a classical example of an algorithm optimal in theory, but highly inefficient

[*] Corresponding author. Fax: +45 6550 2373.
E-mail addresses: bundala@gmail.com (D. Bundala), mcodish@cs.bgu.ac.il (M. Codish), lcf@imada.sdu.dk (L. Cruz-Filipe), petersk@imada.sdu.dk (P. Schneider-Kamp), jakub.zavodny@oxfordalumni.org (J. Závodný).

http://dx.doi.org/10.1016/j.jcss.2016.09.004
0022-0000/© 2016 Elsevier Inc. All rights reserved.

in practice. Although they attain the theoretically optimal $O(n \log n)$ number of comparisons and $O(\log n)$ depth, the AKS networks are infamous for the large constants hidden in the big-O notation. On the other hand, already in 1968, Batcher [4] gave a simple recursive construction that, even though it creates networks of depth $O(\log^2 n)$, is superior to AKS networks for all practical values of n.

It is of particular interest to construct optimal sorting networks (both in size and in depth) for specific small numbers of inputs. Such networks can be used as building blocks to construct more efficient networks on larger numbers of inputs, for example by serving as base cases in recursive constructions such as Batcher's odd-even construction.

Already in the fifties and sixties various constructions appeared for small sorting networks on few inputs. In the 1973 edition of *"The Art of Computer Programming"* [19] (vol. 3, Section 5.3.4), Knuth detailed the smallest sorting networks known at the time with $n \leq 16$ inputs.

However, showing their optimality has proved to be extremely challenging. For $n \leq 8$ inputs, optimality was established by Knuth and Floyd [16] in 1973. No further progress had been made on the problem until 1989, when Parberry [28,29] showed that the networks given for $n = 9$ and $n = 10$ are also optimal. Parberry obtained this result by implementing an exhaustive search with pruning based on symmetries in the first two parallel steps in the sorting networks, and executing the algorithm on a Cray-2 supercomputer. Despite the great increase in available computational power in the two and a half decades since, his algorithm would still not be able to handle the case $n = 11$ or bigger. More recently, there were additional attempts [27] at solving the $n = 11$ case, but we are not aware of any successful one.

In this paper, some 40 years after the publication of the networks by Knuth, we finally prove their optimality by settling the remaining open cases of $11 \leq n \leq 16$ inputs. Our approach combines two methodologies: symmetry breaking and Boolean satisfiability.

Symmetry Breaking. We show how to construct a small set R_n of two-layer networks on n channels such that: if there is a sorting network on n channels of a given depth, then there is one whose first two layers are in this set. We first show how each two-layer network can be represented by a graph, with isomorphic graphs corresponding to equivalent networks. By defining a notion of "relative strength" between networks that takes into account their effects on the inputs, we further restrict the set of two-layer networks. We show how to characterize the strongest networks using context-free grammars, which enables us to construct the sets R_n for up to $n = 40$ inputs within two hours of computation. For example, R_{11} consists of 28 networks, enabling us to solve the optimal-depth problem for $n = 11$ in terms of only 28 independent cases, as opposed to over one billion cases of all two-layer networks on 11 channels. Similarly, we show that $|R_{13}| = 117$.

Boolean Satisfiability. With the first two layers restricted to a small set, we construct a family of propositional formulas whose satisfiability is necessary for the existence of sorting networks of a given size. Using an off-the-shelf SAT solver we show that all the constructed formulas are unsatisfiable, and hence we conclude that for $n \leq 16$ inputs the networks listed in [19] are indeed optimal. A similar construction, without restricting the first two layers, is able to find optimal-depth sorting networks for $n \leq 10$ inputs and prove them optimal, thus providing independent confirmation of the previously known results.

We obtained all our results using an off-the-shelf SAT solver running under Linux on commodity hardware. It is noteworthy that our algorithm required a few seconds to prove the optimality of networks with $n \leq 10$ inputs, whereas for $n = 10$ the algorithm described in [28] was estimated to take hundreds of hours on a supercomputer, and the algorithm described in [27] took more than three weeks on a desktop computer.

The work we describe is another success in the history of computer-assisted proofs. Since the proof of the four-color theorem [2,3], in 1976, several mathematical results have been proven with the help of a computer. Nearly all of these have only been proved by exhaustively analyzing an extremely large search space and using clever reduction techniques, as in our case. SAT-solving has been a key tool in some recent successes in this area, such as the proof of Erdős' discrepancy conjecture for $C = 2$ [20], the proof that the Ramsey number $R(3, 3, 4)$ is equal to 30 [14], and the solution to the Boolean Pythagorean Triples Problem [18].

This paper is an extended version of [5] and [12]. The first paper presents the theory and experiments for calculating optimal sorting networks. In the current paper we construct even smaller sets of "non-isomorphic" two-layer networks using a much faster algorithm (the construction in [5] does not scale beyond $n = 13$ inputs). This new algorithm is a culmination of the work presented in the second paper [12]. However, that paper deals only with computing the sets of "relevant" two-layer networks, and not with computing the optimal sorting networks as we do in this paper.

2. Preliminaries on sorting networks

An example of a comparator network on 4 channels is shown in Fig. 1. The figure introduces the graphical notation used throughout the paper to depict comparator networks. Channels are indicated as horizontal lines (with channel 4 at the bottom), comparators are indicated as vertical lines connecting a pair of channels, and layers are separated by dashed lines. The figure further shows how the inputs $\langle 5, 2, 0, 7\rangle$ and $\langle 0, 1, 0, 1\rangle$ propagate from left to right through the network.

Formally, a *comparator network* C with n *channels* and *depth* d is a sequence $C = L_1; \ldots; L_d$ of d *layers*. Each layer L_k is a set of comparators (i, j), joining the channels i and j, with $1 \leq i < j \leq n$. In every layer L_k, each channel i is used by at

Please cite this article in press as: D. Bundala et al., Optimal-depth sorting networks, J. Comput. Syst. Sci. (2016), http://dx.doi.org/10.1016/j.jcss.2016.09.004