Journal of Computer and System Sciences 82 (2016) 551–563

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Sorting nine inputs requires twenty-five comparisons

Michael Codish [a], Luís Cruz-Filipe [b,*], Michael Frank [a], Peter Schneider-Kamp [b]

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

ARTICLE INFO

Article history:
Received 27 October 2014
Received in revised form 10 November 2015
Accepted 30 November 2015
Available online 11 December 2015

Keywords:
Sorting networks
SAT solving
Computer-assisted proofs
Symmetry breaking

ABSTRACT

This paper describes a computer-assisted non-existence proof of 9-input sorting networks consisting of 24 comparators, hence showing that the 25-comparator sorting network found by Floyd in 1964 is optimal. As a corollary, the 29-comparator network found by Waksman in 1969 is optimal when sorting 10 inputs.
This closes the two smallest open instances of the optimal-size sorting network problem, which have been open since the results of Floyd and Knuth from 1966 proving optimality for sorting networks of up to 8 inputs.

1. Introduction

General-purpose sorting algorithms are based on comparing, and possibly 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.

Sorting networks are a classical formal model for such algorithms [7,17,23], where n inputs are fed into networks of n channels, which are 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. Consecutive comparators can be viewed as a "parallel layer" if no two touch 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 for specific given numbers of inputs: optimal size (minimum number of comparators) as well as optimal depth (minimum number of layers) networks. In this paper we focus on optimal-size sorting networks.

Optimal-size and optimal-depth sorting networks for $n \leq 8$ can already be found in Section 5.3.4 of [17]. For optimal depth, in 1991 Parberry [21] proved optimality results for $n = 9$ and $n = 10$, which in 2014 were extended by Bundala and Závodný [8] to $11 \leq n \leq 16$. Both approaches are based on breaking symmetries among the first (two) layers of comparators.

For optimal size, the case of $n = 9$ has been the smallest open problem ever since Floyd and Knuth's result for optimal-size sorting networks [15] in 1966. At first, this might be surprising: is optimal size really harder than optimal depth? However, a comparison of the sizes of the search spaces for the optimal-size and optimal-depth problems for $n = 9$ sheds some light on the issues. The smallest known sorting network for 9 inputs has size 25. For proving/disproving its opti-

* Corresponding author.
E-mail addresses: mcodish@cs.bgu.ac.il (M. Codish), lcf@imada.sdu.dk (L. Cruz-Filipe), frankm@cs.bgu.ac.il (M. Frank), petersk@imada.sdu.dk (P. Schneider-Kamp).

http://dx.doi.org/10.1016/j.jcss.2015.11.014
0022-0000/

mality, we need to consider all comparator networks of 24 comparators. There are $36 = (9 \times 8)/2$ possibilities to place each comparator on 2 out of 9 channels. Thus, the search space for the optimal-size problem on 9 inputs consists of $(36)^{24} \approx 2.2 \times 10^{37}$ comparator networks.

In comparison, to show that the optimal-depth sorting network for 9 inputs is 7, one must show that there are no sorting networks of depth 6. The number of ways to place comparators in an n-channel layer corresponds to the number of matchings in a complete graph with n nodes [8], and for $n = 9$ this number is 2620. Thus, the search space for the optimal-depth problem on 9 inputs is "just" $2620^6 \approx 3.2 \times 10^{20}$. In addition, the layering allows for some beautiful symmetry breaking [8,11] on the first two layers, reducing the search space further to approx. 10^{15} comparator networks.

For the optimal-depth problem, all recent attempts we are aware of [8,20] have used encodings to the satisfiability problem of propositional logic (SAT). Likewise, in this paper we describe a SAT encoding for the optimal-size problem. This SAT encoding was able to reproduce all known results for $n \leq 6$. Unfortunately, the SAT encoding alone did not scale to $n = 9$, with state-of-the-art SAT solvers making no discernible progress even after weeks of operation.

To solve the open problem of optimality for $n = 9$, we had to invent symmetry breaking techniques for reducing the search space to a manageable size. The general idea is similar to the one taken in [8,11] for the optimal-depth sorting network problem, but involves the generation of minimal sets of non-redundant comparator networks for a given number of comparators, one comparator at a time. Redundant networks (i.e., networks that sort less than others of the same size or that are equivalent to another network already in the set) are pruned. For each pruned network, a witness is logged, which can be independently verified. Our symmetry breaks benefit from the idea of partitioning binary sequences according to the number of 1s contain, which has been discussed e.g. in [6].

For $n = 9$, we used this method, which we call generate-and-prune, to reduce the search space from approx. 2.2×10^{37} to approx. 3.3×10^{21} comparator networks, all of which can be obtained by extending one of 914,444 representative 14-comparator networks. This process took a little over one week of computation, and all of the resulting problems could be handled efficiently by our SAT encoding in less than 12 hours (in total). All computations, if not specified otherwise, were performed on a cluster with a total of 144 Intel E8400 cores clocked at 2 GHz each, able to run a total of 288 parallel threads.

The generate-and-prune method can also be used in isolation to decide this open problem: amongst the set of all comparator networks (modulo equivalence and non-redundancy) there is only one single sorting network, and it is of size 25. To obtain this result, we continued running the generate-and-prune method for five more days in order to check the validity of the results obtained through the SAT encoding independently, thereby instilling a higher level of trust into the computer-assisted proof. This paper presents both techniques: the first one based completely on the generate-and-prune approach, and the second, hybrid, method combining generate-and-prune with SAT encoding. It is the second approach that solves the nine-input case in the least amount of time, and also shows the potential to scale.

Once determining that 25 comparators is optimal for 9 inputs, we move on to consider the case of 10 inputs. Using a result of Van Voorhis from 1971 [26], we know that the minimum number of comparators for sorting 10 inputs is larger by at least 4 than for 9 inputs. As a sorting network with 29 comparators on ten inputs (attributed to Waksman) is known since 1969 [17], our result implies its optimality.

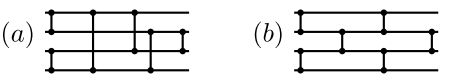
The next section introduces the relevant concepts on sorting networks together with some notations. The generate-and-prune algorithm is introduced in Section 3, while its optimization and parallelization are discussed in detail in Section 4. The SAT encoding is explained and analyzed in Section 5. In Section 6 we reflect on the validity of the proof, and we conclude in Section 7. This paper is an extended version of [10].

2. Preliminaries on sorting networks

A *comparator network* C with n channels and size k is a sequence of *comparators* $C = (i_1, j_1); \ldots; (i_k, j_k)$ where each comparator (i_ℓ, j_ℓ) is a pair of channels $1 \leq i_\ell < j_\ell \leq n$. The *size* of a comparator network is the number of its comparators. If C_1 and C_2 are comparator networks with n channels, then $C_1; C_2$ denotes the comparator network obtained by concatenating C_1 and C_2; if C_1 has m comparators, it is a *size-m prefix* of $C_1; C_2$. An input $\bar{x} = x_1 \ldots x_n \in \{0,1\}^n$ propagates through C as follows: $\bar{x}^0 = \bar{x}$, and for $0 < \ell \leq k$, $\bar{x}^\ell$ is the permutation of $\bar{x}^{\ell-1}$ obtained by interchanging $\bar{x}^{\ell-1}_{i_\ell}$ and $\bar{x}^{\ell-1}_{j_\ell}$ whenever $\bar{x}^{\ell-1}_{i_\ell} > \bar{x}^{\ell-1}_{j_\ell}$. The output of the network for input $\bar{x}$ is $C(\bar{x}) = \bar{x}^k$, and $\mathsf{outputs}(C) = \{\, C(\bar{x}) \mid \bar{x} \in \{0,1\}^n \,\}$. The comparator network C is a *sorting network* if all elements of $\mathsf{outputs}(C)$ are sorted (in ascending order).

Our focus on binary sequences is justified by the zero-one principle (e.g. [17]), which states that a comparator network sorts all elements of $\{0,1\}^n$ iff it sorts all sequences of length n over any totally ordered set, e.g. integers.

Images (a) and (b) on the right depict sorting networks on 4 channels, each consisting of 6 comparators. The channels are indicated as horizontal lines (with channel 4 at the bottom), comparators are indicated as vertical lines connecting a pair of channels, and input values are assumed to propagate from left to right. The sequence of comparators associated with a picture representation is obtained by a left-to-right, top-down traversal. For example the networks depicted above are: (a) $(1,2); (3,4); (1,4); (1,3); (2,4); (2,3)$ and (b) $(1,2); (3,4); (2,3); (1,2); (3,4); (2,3)$.



The optimal-size sorting network problem is about finding the smallest size, $S(n)$, of a sorting network on n channels. In [15], Floyd and Knuth present sorting networks of optimal size for $n \leq 8$ and prove their optimality. Until today, the minimal size $S(n)$ of a sorting network on n channels was known only for $n \leq 8$; for greater values of n, there are upper bounds