Information Sciences 190 (2012) 178–191

Contents lists available at SciVerse ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

On the complexity of min–max sorting networks

Giuseppe Campobello [a], Giuseppe Patanè [b], Marco Russo [c,*]

[a] Department of Matter Physics and Electronic Engineering, University of Messina, C.da di Dio, 98166 Messina, Italy
[b] DMCE Danube Mobile Communications Engineering GmbH & Co KG (majority owned by Intel Mobile Communications), RF System Engineering, Freistaedter Strasse 400, 4040 Linz, Austria
[c] Department of Physics and Astronomy, Univeristy of Catania and National Institute of Nuclear Physics (INFN) Catania Section, Viale A. Doria, 6, 95125 Catania, Italy

ARTICLE INFO

Article history:
Received 15 July 2009
Received in revised form 30 November 2011
Accepted 2 December 2011
Available online 24 December 2011

Keywords:
Min–max circuits
Sorting networks
Merging networks
FPGA

ABSTRACT

This paper extends previous work on sorting networks (SNs) based on min/max circuits. In particular, we have identified the complexity of both min/max-based sorting and merging networks showing that, depending on design choice, the time complexity of this kind of SN ranges from $O(1)$ to $O(\log(n))$ and spatial complexity from $O(n2^n)$ to $O(n^2)$, respectively. Moreover, we show that both AT and AT^2 metrics of the proposed SN are better than those of Batcher's SNs also for SNs with several hundreds of inputs.

In addition to these results we show how to design a fast digital, serial, pipelined sorting network using FPGA technology. As expected, FPGA synthesis results confirm our theoretical analysis.

1. Introduction

Sorting algorithms are extensively studied in Computer Science [20,2,10] and their hardware implementations, known as sorting networks (SNs), are very important in different applications, for instance, in high-energy physics experiments [12], ATM switching [39,5], radio-astronomy [8,7] and parallel databases [13,9].

Usually, SNs are implemented using an appropriate interconnection of Comparison Elements (CEs) [20], which are two-input two-output devices able to sort two elements.

Obviously, in the digital domain, the word size (i.e. the number of bits used to represent the elements) and the technology used for realizing CEs affect implementation costs (i.e. the number of transistors or logic gates needed to implement the SN) and also the maximum achievable speed.

To avoid technology dependent and implementation specific factors, SNs are compared in terms of two main metrics: time (or depth) complexity (T) and spatial (or area) complexity (A), both expressed in terms of the number of inputs to sort, n, using the big-O notation [20]. For instance, a spatial complexity $A = O(n\log^2(n))$ means that the implementation cost can be upper-bounded by $k \cdot n\log^2(n)$ for some real constant k.

It is worth observing that due to the fact that the big-O notation hides constant factors, it follows that the complexity is the same if, instead of the number of CEs, the number of gates or transistors or maximum/minimum circuits needed for the implementation are considered.

The same observations can be made for time complexity, representing the maximum number of CEs (or gates or transistors) that must be traversed from the generic input to the generic output of a SN.

Sometimes a combined metric, such as AT or AT^2, is preferred for comparison purposes.

* Corresponding author. Tel.: +39 095 378 53 26.
E-mail addresses: gcampobello@unime.it (G. Campobello), giuseppe.patane@intel.com (G. Patanè), marco.russo@ct.infn.it (M. Russo).

0020-0255/$ - see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2011.12.008

One of the most cited works in this field is that of Batcher [3]. In his paper in 1948, Batcher proposed two SNs, named respectively bitonic sorting and odd–even merging networks, which have area and time complexities of $A = O(n\log^2(n))$ and $T = O(\log^2(n))$, respectively.

For many years since Batcher's work, researchers have been investigating the existence of faster SNs without success.

Although it has been proved that optimal SNs implemented with CEs have $A = O(n\log(n))$ and $T = O(\log(n))$ [19], today the only known optimal SN is the AKS [26] and its refinements, mainly due to the work of Leighton [21], Paterson [30] and Chvátal [6].

However, the big O notation can hide large constant factors and this is the case of AKS where the hidden constant in the time complexity is nearly 6000 in [30] and nearly 2000 in [6].

That is the reason why the AKS network has, for a long time, been considered highly impractical [33,27].

The increasing number of transistors available in modern VLSI technology have made AKS feasible. For instance in [24] Lin and Olariu presented a VLSI architecture that efficiently implements Leighton's improvement of the AKS network (i.e. Columnsort) showing that for small and moderate numbers of inputs (i.e. $n \leqslant 16$), when the AT^2 metric is considered, the proposed architecture is superior to the existing designs.

Another implementation of the AKS algorithm was proposed in [25] where the authors showed how to implement an AKS network in a multibutterfly network. However, as stated by the same authors, their algorithm is not practical. More recently, Seiferas et al. [34] have simplified the description of the AKS algorithm but with no further reduction in complexity.

So, as also stated in recent papers, *"the merge-sort networks of Batcher are still the best practical sorting networks"* [22]. This justifies why Batcher's SNs are the most used SNs even when fast sorting circuits are required [14,11].

Moreover, it is worth mentioning that the area complexity of Batcher's bitonic sorting networks has been further improved and reduced to $O(n \log(n))$ [18] (instead its time complexity still remains $O(\log^2(n))$).

Obviously, $O(\log(n))$ time complexity with a low constant factor can be achieved at the cost of higher spatial complexity. For instance, in [15] a class of SNs with $O(\log(n))$ time complexity and $O\left(\frac{n^2}{\log(n)}\right)$ spatial complexity was proposed.

Another example of very fast SNs is given in [32] where in 1996 a novel class of SNs with $O(1)$ time complexity was presented. More precisely, in the first part of the paper the authors show how to sort n items using a two layer SN each layer implemented by minimum or maximum circuits. This means that the time complexity of this solution is $O(1)$ with an approximated, but commonly used, cost model where the delay introduced by a min/max elementary block is not related to the number of its inputs. The authors clearly showed through numerical examples that the solution presented was the fastest in literature but were not able to evaluate its spatial complexity. In the second part of the paper, the authors introduced min–max merging networks, consisting of two layers of minimum and maximum elementary blocks which are able to sort two or more pre-ordered vectors. In particular, the authors showed that using more layers of mergering networks it is possible to design large sorting networks with $O(\log(n))$ time complexity at a substantially lower cost in comparison to the previous solution. In these cases too, all results reported regarding spatial complexity were empirical and based on an oversimplified analog hardware implementation.

Recently min–max networks have been rediscovered by Levi and Litman [16]. In particular they have shown that the min–max model of computation outperforms the comparator model used to derive Batcher's SNs [23] and that using min–max networks it is possible to accelerate specific outputs of Batcher's sorting networks [22]. However the overall time complexity of the proposed SN remains the same as Batcher's, i.e. $O(\log^2(n))$. Moreover, spatial complexity is not discussed.

Obviously in order to correctly compare Batcher's SNs with min–max SNs, it is necessary to take spatial complexity into consideration.

In this paper we finally derive the spatial complexities of both sorting and merging networks based on min–max circuits originally proposed in [32].

In particular, we prove that when the number of elements to sort is below a few hundred and AT or AT^2 metrics are considerd, min–max SNs outperform Batcher's SNs.

Finally, we show how to design a fast digital, serial, pipelined SN using FPGA technology. Synthesis results for FPGA confirm our theoretical analysis.

The paper is organized as follows: in Section 2 notation, main definitions and a brief summary of the theorems demonstrated in [32] are presented; in Section 3 the complexity of sorting and merging networks is approached from an entirely analytical point of view; in Section 4 the obtained analytical results are compared with empirical ones in [32]; in Section 5 an FPGA-based implementation is illustrated and synthesis results are provided; in Section 6 some considerations regarding the model are given. Lastly, the VHDL source codes used to obtain synthesis results are reported in the Appendix.

2. Notation and definitions

In this section, firstly we introduce the notation that will be used throughout the paper; afterwards, max/min based sorting and merging networks are formally defined and related theorems, already presented in [32], will be summarized.

2.1. Notation

Here, we present a brief outline of the notations employed in the paper. More symbols will be introduced, when necessary, in the related sections.