Sorting Networks: the End Game[*]

Michael Codish[1], Luís Cruz-Filipe[2], and Peter Schneider-Kamp[2]

[1] Department of Computer Science
Ben-Gurion University of the Negev
PoB 653, Beer-Sheva, Israel 84105
`mcodish@cs.bgu.ac.il`
[2] Department of Mathematics and Computer Science,
University of Southern Denmark
Campusvej 55, 5230 ODENSE M, Denmark
`{lcf,petersk}@imada.sdu.dk`

Abstract. This paper studies properties of the back end of a sorting network and illustrates the utility of these in the search for networks of optimal size or depth. All previous works focus on properties of the front end of networks and on how to apply these to break symmetries in the search. The new properties help shed understanding on how sorting networks sort and speed-up solvers for both optimal size and depth by an order of magnitude.

Keywords: sorting networks, SAT solving, symmetry breaking

1 Introduction

In the last year, new results were obtained regarding optimality of sorting networks, concerning both the optimal depth of sorting networks on 11 to 16 channels [2] and the optimal size of sorting networks on 9 and 10 channels [3]. Both these works apply symmetry-breaking techniques that rely on analyzing the structure at the front of a sorting network in order to reduce the number of candidates to test in an exhaustive proof by case analysis.

In this work, we focus on the dual problem: what does the *end* of a sorting network look like? To the best of our knowledge, this question has never been studied in much detail. Batcher [1] characterizes a particular class of networks that can be completed to a sorting network in a systematic way, but his work only applies to the search for efficient sorting networks. Parberry [8] establishes a necessary condition to avoid examining the last two layers of a candidate prefix in his proof of optimality of the depth 6 sorting network on 9 channels, but its application requires fixing the previous layers (although it has similarities to the idea behind our proof of Theorem 11 below).

We show that the comparators in the last layer of a sorting network are of a very particular form, and that the possibilities for the penultimate layer are also

limited. Furthermore, we show how to control redundancy of a sorting network in a very precise way in order to restrict its last two layers to a significantly smaller number of possibilities, and we study the impact of this construction in the SAT encodings used in the proofs of optimality described in [2, 3].

The analysis, results, and techniques in this paper differ substantially from the work done on the first layers: that work relies heavily on symmetries of sorting networks to show that the comparators in those layers *may* be restricted to be of a particular form. Our results show that the comparators in the last layers *must* have a particular form. When working with the first layers it suffices to work up to renaming of the channels, as there are very general results on how to apply permutations to the first layers of any sorting network and obtain another sorting network of the same depth and size. On the last layer, this is not true: permuting the ending of a sorting network will not, in general, yield the ending of another sorting network. We formalize the fact that, as inputs go through a sorting network, the number of channels between pairs of unsorted values gets smaller, until, at the last layer, all occurrences of unsorted pairs of values are on adjacent channels. To the best of our knowledge, this surprising fact has never been observed before, and it influences the possible positions of comparators in the last layers. This intuition about the mechanism of sorting networks is formally expressed by the notion of k-block and Theorem 11, which is the main contribution of this paper.

2   Preliminaries on Sorting Networks

A *comparator network* C with n channels and depth d is a sequence $C = L_1; \ldots; L_d$ where each *layer* L_k is a set of comparators (i, j) for pairs of channels $i < j$. At each layer, every channel may occur in at most one comparator. The *depth* of C is the number of layers d, and the *size* of C is the total number of comparators in its layers. If C_1 and C_2 are comparator networks, then $C_1; C_2$ denotes the comparator network obtained by concatenating the layers of C_1 and C_2; if C_1 has m layers, it is an *m-layer prefix* of $C_1; C_2$.

An input $\bar{x} \in \{0, 1\}^n$ propagates through C as follows: $\bar{x}_0 = \bar{x}$, and for $0 < k \leq d$, $\bar{x}_k$ is the permutation of $\bar{x}_{k-1}$ obtained as follows: for each comparator $(i, j) \in L_k$, the values at positions i and j of $\bar{x}_{k-1}$ are reordered in $\bar{x}_k$ so that the value at position i is not larger than the value at position j. The output of the network for input $\bar{x}$ is $C(\bar{x}) = \bar{x}_d$, 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). The zero-one principle (e.g. [6]) implies that a sorting network also sorts any other totally ordered set, e.g. integers.

Optimal sorting network problems are about finding the smallest depth and the smallest size of a sorting network for a given number of channels n. Figure 1 shows a sorting network on 5 channels that has optimal size (9 comparators) and optimal depth (5 layers). It also shows how the network sorts the input 10101.

In order to determine the minimal depth of an optimal sorting network on n channels, one needs to consider all possible ways in which such a network
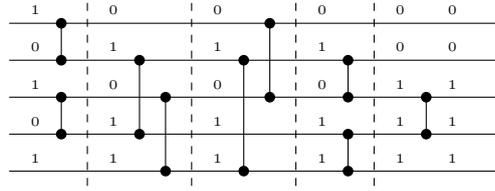
Fig. 1. An optimal-depth, optimal-size sorting network on 5 channels, operating on the input 10101. The channels are numbered from top to bottom, with a comparator (i, j) represented as a vertical line between two channels; each comparator moves its smallest input to its top channel. The layers are separated by a vertical dashed line.

can be built. Parberry [8] shows that the first layer of a depth-optimal sorting network on n channels can be assumed to consist of the comparators $(2k-1, 2k)$ for $1 \leq k \leq \lfloor \frac{n}{2} \rfloor$. Parberry and later Bundala and Závodný pursued the study of the possibilities for the second layer and demonstrated the impact of this on the search for optimal sorting networks.

The following two observations will be be instrumental for proofs in later sections. We write $\bar{x} \leq \bar{y}$ to denote that every bit of x is less than or equal to the corresponding bit of y, and $\bar{x} < \bar{y}$ for $\bar{x} \leq \bar{y}$ and $x \neq y$.

Lemma 1. *Let C be a comparator network and $\bar{x}$ be a sorted sequence. Then $\bar{x}$ is unchanged by every comparator in C.*

Lemma 2 (Theorem 4.1 in [1]). *Let C be a comparator network and $\bar{x}, \bar{y} \in \{0,1\}^n$ be such that $\bar{x} \leq \bar{y}$. Then $C(\bar{x}) \leq C(\bar{y})$.*

3   The Last Layers of a Sorting Network

In this section we analyze the last two layers of a sorting network and derive some structural properties that will be useful both for restricting the search space in proofs of optimality, and as a tool to understand how a sorting network works.

We begin by recalling the notion of redundant comparator (Exercise 5.3.4.51 of [6], credited to R.L. Graham). Let $C; (i, j); C'$ be a comparator network. The comparator (i, j) is *redundant* if $x_i \leq x_j$ for all sequences $x_1 \ldots x_n \in \mathsf{outputs}(C)$. If D' is a comparator network obtained by removing every redundant comparator from D, then D' is a sorting network iff D is a sorting network: from the definition it follows that $D(\bar{x}) = D'(\bar{x})$ for every input $\bar{x} \in \{0,1\}^n$. This result was already explored in the proof of optimality of the 25-comparator sorting network on 9 channels [3]. We will call a sorting network without redundant comparators *non-redundant*. In this section we focus on non-redundant sorting networks.

Lemma 3. *Let C be a non-redundant sorting network on n channels. Then all comparators in the last layer of C are of the form $(i, i+1)$.*

Proof. Let C be as in the premise with a comparator $c = (i, i+2)$ in the last layer. We can assume it is the last comparator. Since c is not redundant, there is an input $\bar{x}$ such that channels i to $i+2$ before applying c look like (a) or (b) on the right.

Suppose $\bar{x}$ is a word yielding case (a), and let $\bar{y}$ be any word obtained by replacing one 0 in $\bar{x}$ by a 1. Since C is a sorting network, $C(\bar{y})$ is sorted, but since $\bar{x} < \bar{y}$ the value in channel i before applying c must be a 1 (Lemma 2), hence $\bar{y}$ yields situation (b). Dually, given $\bar{y}$ yielding (b), we know that any $\bar{z}$ obtained by replacing one 1 in $\bar{y}$ by a 0 will yield (a).

Thus all inputs with the same number of zeroes as $\bar{x}$ or $\bar{y}$ must yield either (a) or (b), in particular sorted inputs, contradicting Lemma 1. The same reasoning shows that c cannot have the form $(i, i+k)$ with $k > 2$, thus it has to be of the form $(i, i+1)$. □

Corollary 4. *Suppose that C is a sorting network with no redundant comparators that contains a comparator (i, j) at layer d, with $j > i + 1$. Then at least one of channels i and j is used in a layer d' with $d' > d$.*

Proof. If neither i nor j are used after layer d, then the comparator (i, j) can be moved to the last layer without changing the function computed by C. By the previous lemma C can therefore not be a sorting network. □

Lemma 3 restricts the number of possible comparators in the last layer in a sorting network on n channels to $n - 1$, instead of $n(n-1)/2$ in the general case.

Theorem 5. *The number of possible last layers in an n-channel sorting network with no redundancy is $L_n = F_{n+1} - 1$, where F_n denotes the Fibonacci sequence.*

Proof. Denote by L_n^+ the number of possible last layers on n channels, where the last layer is allowed to be empty (so $L_n = L_n^+ - 1$). There is exactly one possible last layer on 1 channel, and there are two possible last layers on 2 channels (no comparators or one comparator), so $L_1^+ = F_2$ and $L_2^+ = F_3$.

Given a layer on n channels, there are two possibilities. Either the first channel is unused, and there are L_{n-1}^+ possibilities for the remaining $n - 1$ channels; or it is connected to the second channel, and there are L_{n-2}^+ possibilities for the remaining $n - 2$ channels. So $L_n^+ = L_{n-1}^+ + L_{n-2}^+$, whence $L_n^+ = F_{n+1}$. □

Even though L_n grows quickly, it grows slower than the number G_n of possible layers in general [2]; in particular, $L_{17} = 2583$, whereas $G_{17} = 211{,}799{,}312$.

To move (backwards) beyond the last layer, we introduce an auxiliary notion.

Definition 6. *Let C be a depth d sorting network without redundant comparators, and let $k < d$. A k-block of C is a set of channels B such that $i, j \in B$ if and only if there is a sequence of channels $i = x_0, \ldots, x_\ell = j$ where (x_i, x_{i+1}) or (x_{i+1}, x_i) is a comparator in a layer $k' > k$ of C.*

Note that for each k the set of k-blocks of C is a partition of the set of channels.

Given a comparator network of depth d, we will call its $(d-1)$-blocks simply *blocks* – so Lemma 3 states that a block in a sorting network C is either a channel unused at the last layer of C or two adjacent channels connected by a comparator at the last layer of C.

Example 7. Recall the sorting network shown in Figure 1. Its 4-blocks, or simply blocks, are $\{1\}$, $\{2\}$, $\{3,4\}$ and $\{5\}$, its 3-blocks are $\{1\}$, $\{2,3,4,5\}$, and for $k < 3$ there is only the trivial k-block $\{1,2,3,4,5\}$.

Lemma 8. *Let C be a sorting network of depth d on n channels, and $k < d$. For each input $\bar{x} \in \{0,1\}^n$, there is at most one k-block that receives a mixture of $0s$ and $1s$ as input.*

Proof. From the definition of k-block, there is no way for values to move from one k-block to another. Therefore, if there is an input for which two distinct k-blocks receive both 0s and 1s as inputs, the output will not be sorted. □

Lemma 9. *Let C be a depth d sorting network on n channels without redundant comparators. Then all comparators in layer $d-1$ connect adjacent blocks of C.*

Proof. The proof is similar to that of Lemma 3, but now considering blocks instead of channels. Let c be a comparator in layer $d-1$ of C that does not connect adjacent blocks of C. Since c is not redundant, there must be some input $\bar{x}$ that provides c with input 1 on its top channel and 0 on its bottom channel. The situation is depicted below, where A and C are blocks, and B is the set of channels in between. According to Lemma 8, there are five possible cases for A, B and C, depending on the number of 0s in $\bar{x}$.

A	all 0	all 0	all 0	all 0	mixed
B	all 0	all 0	mixed	all 1	all 1
C	mixed	all 1	all 1	all 1	all 1
	(a)	(b)	(c)	(d)	(e)

Suppose that $\bar{x}$ yields (a). By changing the appropriate number of 0s in $\bar{x}$ to 1s, we can find a word $\bar{y}$ yielding case (b), since again by monotonicity of C this cannot bring a 0 to the top input of c. Likewise, we can reduce (e) to (d). But now we can move between (b), (c) and (d) by changing one bit of the word at a time. By Lemma 2, this must keep either the top 1 input of c or the lower 0, while the other input is kept by the fact that C is a sorting network. Again this proves that this configuration occurs for all words with the same number of 0s, which is absurd since it cannot happen for the sorted input. □

Combining this result with Lemma 3 we obtain the explicit configurations that can occur in a sorting network.

Corollary 10. *Let C be a depth d sorting network on n channels without redundant comparators. Then every comparator (i,j) in layer $d-1$ of C satisfies $j - i \leq 3$. Furthermore, if $j = i + 2$, then either $(i, i+1)$ or $(i+1, i+2)$ occurs in the last layer; and if $j = i + 3$, then both $(i, i+1)$ and $(i+2, i+3)$ occur in the last layer.*
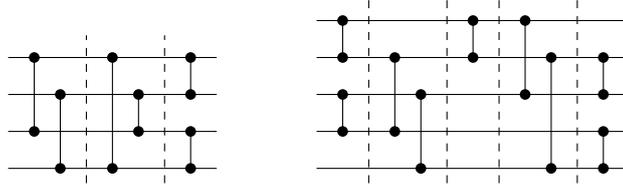
Fig. 2. Sorting networks containing a comparator $(i, i + 3)$ in their penultimate layer.

The sorting networks in Figure 2 show that the bound $j - i \leq 3$ is tight. We can also state a more general form of Lemma 9, proved exactly in the same way.

Theorem 11. *If C is a sorting network on n channels without redundant comparators, then every comparator at layer k of C connects adjacent k-blocks of C.*

When considering the last n comparators instead of the last k layers, induction on n using Theorem 11 yields the following result.

Corollary 12. *Every k-block with n comparators of a sorting network without redundant comparators uses at most $n + 1$ channels.*

4   Co-saturation

The results in the previous sections allow us to reduce the search space of all possible sorting networks of a given depth simply by identifying necessary conditions on the comparators those networks may have. However, the successful strategies in [2, 3, 8] all focus on finding *sufficent* conditions on those comparators: identifying a (smaller) set of networks that must contain one sorting network of depth d (or size k), if such a network exists at all.

We now follow this idea pursuing the idea of saturation in [2]: how many (redundant) comparators can we safely add to the last layers of a sorting network? We will show how to do this in a way that reduces the number of possibilities for the last two layers to a minimum. Note that we are again capitalizing on the observation that redundant comparators do not change the function represented by a comparator network and can, thus, be removed or added at will.

Lemma 13. *Let C be a sorting network on n channels. There is a sorting network N of the same depth whose last layer: (i) only contains comparators between adjacent channels; and (ii) does not contain two adjacent unused channels.*

Proof. We first eliminate all redundant comparators from C to obtain a sorting network S. By Lemma 3 all comparators in the last layer of S are then of the form $(i, i+1)$. Let j be such that j and $j+1$ are unused in the last layer of S; since S is a sorting network, this means that the comparator $(j, j + 1)$ is redundant and we can add it to the last layer of S. Repeating this process for $j = 1, \ldots, n$ we obtain a sorting network N that satisfies both desired properties. $\qquad\square$

We say that a sorting network satisfying the conditions of Lemma 13 is in *last layer normal form* (llnf).

Theorem 14. *The number of possible last layers in llnf on n channels is $K_n = P_{n+5}$, where P_n denotes the Padovan sequence, defined as $P_0 = 1$, $P_1 = P_2 = 0$ and $P_{n+3} = P_n + P_{n+1}$.*

Proof. Let K_n^+ be the number of layers in llnf that begin with the comparator $(1, 2)$, and K_n^- the number of those where channel 1 is free. Then $K_n = K_n^+ + K_n^-$. Let $n > 3$. If a layer in llnf begins with a comparator, then there are K_{n-2} possibilities for the remaining channels; if it begins with a free channel, then there are K_{n-1}^+ possibilities for the remaining channels. Therefore $K_n = K_n^+ + K_n^- = K_{n-2} + K_{n-1}^+ = K_{n-2} + K_{n-3}$. There exist one last layer on 1 channel (with no comparator), one on 2 channels (with one comparator between them) and two on 3 channels (one comparator between either the top two or the bottom two channels), so $K_1 = P_6$, $K_2 = P_7$ and $K_3 = P_8$. From the recurrence it follows that $K_n = P_{n+5}$. □

Note that K_n grows much slower than the total number L_n of non-redundant last layers identified in Theorem 5. For example, $K_{17} = 86$ instead of $L_{17} = 2583$.

If the last layer is required to be in llnf, we can also study the previous layer. By Lemma 9, we know that every block can only be connected to the adjacent ones; again we can *add* redundant comparators to reduce the number of possibilities for the last two layers.

Lemma 15. *Let C be a sorting network of depth d in llnf. Let $i < j$ be two channels that are unused in layer $d-1$ and that belong to different blocks. Then adding the comparator (i, j) to layer $d - 1$ of C still yields a sorting network.*

Proof. Suppose there is an input $\bar{x}$ such that channel i carries a 1 at layer $d-1$, and channel j carries a 0 at that same layer. Since neither channel is used, their corresponding blocks will receive these values. But then $C(\bar{x})$ has a 1 in a channel in the block containing i and a 0 in the block containing j, and since $i < j$ this sequence is not sorted by C. Therefore the comparator (i, j) at layer $d - 1$ of C is redundant, and can be added to this network. □

Lemma 16. *Let C be a sorting network of depth d in llnf. Suppose that there is a comparator $(i, i+1)$ in the last layer of C, that channel $i+2$ is used in layer $d-1$ but not in layer d, and that channels i and $i+1$ are both unused in layer $d-1$ of C (see Figure 3, left). Then there is a sorting network C' of depth d in llnf such that channels $i+1$ and $i+2$ are both used in layers $d-1$ and d.*

Proof. Since channels i and $i+1$ are unused in layer $d-1$, comparator $(i, i+1)$ can be moved to that layer without changing the behaviour of C; then the redundant comparator $(i+1, i+2)$ can be added to layer d, yielding the sorting network C' (Figure 3, left). If $i > 1$ and channel $i-1$ is not used in the last layer of C, then C' must also contain a comparator $(i-1, i)$ in its last layer (Figure 3, right). □
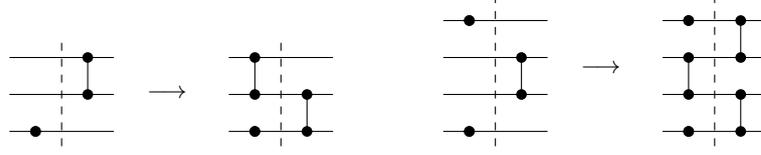
Fig. 3. Transformations in the proof of Lemma 16.

Lemma 16 can also be applied if channel $i - 1$ (instead of $i + 2$) is used at layer $d - 1$ and unused in layer d.

Definition 17. *A sorting network of depth d is co-saturated if: (i) its last layer is in llnf, (ii) no two consecutive blocks at layer $d-1$ have unused channels, and (iii) if $(i, i + 1)$ is a comparator in layer d and channels i and $i + 1$ are unused in layer $d - 1$, then channels $i - 1$ and $i + 2$ (if they exist) are used in layer d.*

Theorem 18. *If C is a sorting network on n channels with depth d, then there is a co-saturated sorting network N on n channels with depth d.*

Proof. Assume C is given. Apply Theorem 14 to find a sorting network S in llnf, containing no redundant comparators except possibly in the last layer.

Let $B_1, \ldots, B_k$ be the $(d - 1)$-blocks in S. For $i = 1, \ldots, k - 1$, if blocks B_i and B_{i+1} have a free channel, add a comparator between them. (Note that it may be possible to add *two* comparators between these blocks, namely if they both have two channels and none is used in layer $d - 1$.) Let N be the resulting network. By Lemma 15, all the comparators added from S to N are redundant, so N is a sorting network; by construction, N satisfies (ii).

If N does not satisfy (iii), then applying Lemma 16 transforms it into another sorting network N' that does. $\square$

Table 1 shows the number of possibilities for the last two layers of a co-saturated sorting network on n channels for $n \leq 17$, obtained by a representation of these suffixes similar to the one described in [4].

In the next sections, we show how we can capitalize on these results to improve the proofs of optimal depth and optimal size of sorting networks.

5   Implications for Optimal Depth SAT encodings

In this section, we describe how SAT encodings in the spirit of [2] can profit from the results in Sections 3 and 4. We detail the boolean variables in the model of the

Table 1. Number of distinct co-saturated two-layer suffixes on n channels, for $n \leq 17$.

n	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
#	4	4	12	26	44	86	180	376	700	1,440	2,892	5,676	11,488	22,848	45,664

encoding, and express our contribution in terms of those. The remaining details of this construction are immaterial to this paper. The encoding represents an n-channel comparator network of depth d by $d \times n(n-1)$ Boolean variables

$$\mathtt{V}_n^d = \left\{ \; c_{i,j}^\ell \mid 1 \leq \ell \leq d, 1 \leq i < j \leq n \; \right\}$$

where the intention is that $c_{i,j}^\ell$ is true if and only if the network contains a comparator between channels i and j at depth ℓ. Further, to facilitate a concise and efficient encoding of our new results, we introduce an additional set of $d \cdot n$ Boolean variables capturing which channels are "used" at a given layer

$$\mathtt{U}_n^d = \left\{ \; u_k^\ell \mid 1 \leq \ell \leq d, 1 \leq k \leq n \; \right\}$$

where the intention is that u_k^ℓ is true if and only if there is some comparator on channel k at level ℓ. Using these variables, previous work describes how the search for an n-channel sorting network of depth d is encoded by a formula φ_0 satisfiable if and only if there is such a network. If φ_0 is satisfiable, the network found can be reconstructed from the assignment of the variables $\mathtt{V}_n^d$.

5.1   Encoding Necessary Conditions

The results of Section 3 represent necessary conditions for non-redundant sorting networks. Thus, we can just add them to the SAT encoding as further restrictions of the search space without losing solutions. We start by looking at the last layer, i.e., the layer at depth d, and then continue to consider layer $d-1$.

Consider first Lemma 3, which states that non-redundant comparators in the last layer have to be of the form $(i, i+1)$. Seen negatively, we can simply forbid all comparators (i, j) where $j > i + 1$, that connect non-adjacent channels. This restriction can be encoded straightforwardly by adding the following $(n-1)(n-2)$ unit clauses φ_1 to the SAT encoding:

$$\varphi_1 = \left\{ \; \neg c_{i,j}^d \mid 1 \leq i, i+1 < j \leq n \; \right\}$$

The restriction from Lemma 3 is generalized by Corollary 4, which states that whenever a comparator at any layer connects two non-adjacent channels, necessarily one of these channels is used at a later layer. Similarly to φ_1 we can encode this by adding one clause for each of the $(n-1)(n-2)/2$ non-adjacent comparator at any given depth ℓ using $\varphi_1(\ell)$:

$$\varphi_1(\ell) = \left\{ \; c_{i,j}^\ell \rightarrow \bigvee_{\ell < k \leq d} u_i^k \vee u_j^k \mid 1 \leq i, i+1 < j \leq n \; \right\}$$

Note that indeed $\varphi_1(d) = \varphi_1$, as there is no depth k with $\ell < k \leq d$.

We now move on to consider the penultimate layer $d-1$. According to Corollary 10, no comparator at this layer can connect two channels more than 3 channels apart. Similar to Lemma 3, we encode this restriction by adding unit clauses for each of the $(n-3)(n-4)/2$ comparators more than 3 channels apart:

$$\varphi_2 = \left\{ \; \neg c_{i,j}^{d-1} \mid 1 \leq i, i+3 < j \leq n \; \right\}$$

Corollary 10 also states that the existence of a comparator $(i, i + 3)$ on the penultimate layer implies the existence of the two comparators $(i, i + 1)$ and $(i+2, i+3)$ on the last layer. This is straightforwardly encoded using additional $2(n-3)$ implication clauses:

$$\varphi_3 = \left\{ \ c^{d-1}_{i,i+3} \to c^{d}_{i,i+1} \right) \land \left(c^{d-1}_{i,i+3} \to c^{d}_{i+2,i+3} \mid 1 \le i \le n - 3 \right\}$$

Finally, Corollary 10 also states that the existence of a comparator $(i, i+2)$ on the penultimate layer implies the existence of either of the comparators $(i, i+1)$ or $(i + 1, i + 2)$ on the last layer. This can be encoded using $n - 2$ clauses:

$$\varphi_4 = \left\{ \ c^{d-1}_{i,i+2} \to c^{d}_{i,i+1} \lor c^{d}_{i+1,i+2} \mid 1 \le i \le n - 2 \right\}$$

Empirically, we have found that using $\varphi = \varphi_0 \land \varphi_1 \land \varphi_2 \land \varphi_3 \land \varphi_4$ instead of just φ_0 decreases SAT solving times dramatically. In contrast, adding $\varphi_1(\ell)$ for $\ell < d$ has not been found to have a positive impact.

5.2   Symmetry Breaking using Sufficient Conditions

The restrictions encoded so far were necessary conditions for non-redundant sorting networks. In addition, we can break symmetries by using the sufficient conditions from Section 4, essentially forcing the SAT solver to *add redundant comparators*.

According to Lemma 13 (ii) we can break symmetries by requiring that there are no adjacent unused channels in the last layer, i.e., that the network is in llnf.

$$\psi_1 = \left\{ \ u^{d}_i \lor u^{d}_{i+1} \mid 1 \le i < n \right\}$$

Essentially, this forces the SAT solver to add a (redundant) comparator between any two adjacent unused channels on the last layer.

The next symmetry break is based on a consideration of two adjacent blocks. There are four possible cases: two adjacent comparators, a comparator followed by an unused channel, an unused channel followed by a comparator, and two unused channels. The latter is forbidden by the symmetry break ψ_1 (and thus not regarded further).

The case of two adjacent comparators is handled by formula ψ_2^a:

$$\psi_2^a = \left\{ \ c^{d}_{i,i+1} \land c^{d}_{i+2,i+3} \to \left(u^{d-1}_i \land u^{d-1}_{i+1} \right) \lor \left(u^{d-1}_{i+2} \land u^{d-1}_{i+3} \right) \mid 1 \le i \le n - 3 \right\}$$

This condition essentially forces the SAT solver to add a (redundant) comparator on layer $d - 1$, if both blocks have an unused channel in that layer.

The same idea of having to add a comparator at layer $d - 1$ is enforced for the two remaining cases of a comparator followed by an unused channel or its dual by ψ_2^b and ψ_2^c, respectively:

$$\psi_2^b = \left\{ \ c^{d}_{i,i+1} \land \neg u^{d}_{i+2} \to \left(u^{d-1}_i \land u^{d-1}_{i+1} \right) \lor u^{d-1}_{i+2} \mid 1 \le i \le n - 2 \right\}$$
$$\psi_2^c = \left\{ \ \neg u^{d}_i \land c^{d}_{i+1,i+2} \to u^{d-1}_i \lor \left(u^{d-1}_{i+1} \land u^{d-1}_{i+2} \right) \mid 1 \le i \le n - 2 \right\}$$

Table 2. SAT solving for n-channel, depth 8 sorting networks with $|R_n|$ 2-layer filters. The table shows the impact of the restrictions on the last two layers in the size of the encoding and the solving time (in seconds) for the slowest unsatisfiable instance, as well as the total time for all $|R_n|$ instances.

n	$|R_n|$	**unrestricted** last layer: φ_0				**restricted** last layer: ψ			
		slowest instance			total	slowest instance			total
		#clauses	#vars	time	time	#clauses	#vars	time	time
15	262	278312	18217	754.74	**130551.42**	335823	25209	148.35	**19029.26**
16	211	453810	27007	1779.14	**156883.21**	314921	22901	300.07	**24604.53**

The final symmetry break is based on Lemma 16, i.e., on the idea of moving a comparator from the last layer to the second last layer. We encode that such a situation cannot occur, i.e., that whenever we have a comparator on the last layer d following or followed by an unused channel, one of the channels of the comparator is used on layer $d - 1$:

$$\psi_3^a = \left\{\, c_{i,i+1}^d \wedge \neg u_{i+2}^d \to u_i^{d-1} \vee u_{i+1}^{d-1} \mid 1 \le i \le n - 2 \,\right\}$$

$$\psi_3^b = \left\{\, c_{i,i+1}^d \wedge \neg u_{i-1}^d \to u_i^{d-1} \vee u_{i+1}^{d-1} \mid 2 \le i \le n - 1 \,\right\}$$

Empirically, we found that $\psi = \varphi \wedge \psi_1 \wedge \psi_2^a \wedge \psi_2^b \wedge \psi_2^c \wedge \psi_3^a \wedge \psi_3^b$ further improves the performance of SAT solvers. In order to show optimality of the known depth 9 sorting networks on 15 and 16 channels, it is enough to show that there is no sorting network on those numbers of channels with a depth of 8. Previous work [4] introduces the notion of complete set of prefixes: a set R_n such that if there exists a sorting network on n channels with depth d, then there exists one extending a prefix in R_n. Using this result, it suffices to show that there are no sorting networks of depth 8 that extend an element of R_{15} or R_{16}. Table 2 shows the improvement of using ψ instead of φ_0, detailing for both cases the number of clauses, the number of variables and the time to solve the slowest of the $|R_n|$ instances (which are solved in parallel). We also specify the total solving time (both compilation and SAT-solving) for all $|R_n|$ instances together. The new encodings are larger per same instance (the slowest instances, showed in the table, are different), but, as indicated in the table, the total time required in order to show that the formulas are unsatisfiable is reduced by a factor of around 6.

6   Conclusion

This paper presents the first systematic exploration of what happens at the *end* of a sorting network, as opposed to at the beginning. We present properties of the last layers of sorting networks. In order to assess the impact of our contribution, we show how to integrate them into SAT encodings that search for sorting networks of a given depth [2]. Here, we see an order of magnitude improvement

in solving times, bringing us closer to being able to solve the next open instance of the optimal depth problem (17 channels).

While the paper presents detailed results on the end of sorting networks in the context of proving optimal depth of sorting networks, the necessary properties of the last layers can also be used to prove optimal size. We experimented on adding constraints similar to those in Section 5 for the last three comparators, as well as constraints encoding Corollary 12, to the SAT encoding presented in [3]. Preliminary results based on uniform random sampling of more than 10% of the cases indicate that we can reduce the total computational time used in the proof that 25 comparators are optimal for 9 channels from 6.5 years to just over 1.5 years. On the 288-thread cluster originally used for that proof, this corresponds to reducing the actual execution time from over 8 days to just 2 days.

These results can also be used to improve times for the *search* for sorting networks. In a recent paper [7], the authors introduce an incremental approach to construct sorting networks (iterating between two different SAT problems). They show that, using the first three layers of a Green filter [5], their approach finds a sorting network with 17 channels and depth 10, thus improving the previous best upper bound on the depth of a 17-channel sorting network. Using the same prefix, together with the constraints on the last two layers described in Section 5, we can find a depth 10 sorting network in under one hour of computation. Without these last layer constraints, this procedure times out after 24 hours.

References

1. Baddar, S.W.A.H., Batcher, K.E.: Designing Sorting Networks: A New Paradigm. Springer (2011)
2. Bundala, D., Závodný, J.: Optimal sorting networks. In: LATA 2014. LNCS, vol. 8370, pp. 236–247. Springer (2014)
3. Codish, M., Cruz-Filipe, L., Frank, M., Schneider-Kamp, P.: Twenty-five comparators is optimal when sorting nine inputs (and twenty-nine for ten). In: Proceedings of ICTAI 2014. IEEE (2014), accepted for publication
4. Codish, M., Cruz-Filipe, L., Schneider-Kamp, P.: The quest for optimal sorting networks: Efficient generation of two-layer prefixes. In: Proceedings of SYNASC 2014. IEEE (2014), accepted for publication
5. Coles, D.: Efficient filters for the simulated evolution of small sorting networks. In: Proceedings of GECCO'12. pp. 593–600. ACM (2012)
6. Knuth, D.E.: The Art of Computer Programming, Volume III: Sorting and Searching. Addison-Wesley (1973)
7. Müller, M., Ehlers, T.: Faster sorting networks for 17, 19 and 20 inputs. CoRR abs/1410.2736 (2014), http://arxiv.org/abs/1410.2736
8. Parberry, I.: A computer-assisted optimal depth lower bound for nine-input sorting networks. Mathematical Systems Theory 24(2), 101–116 (1991)