The Pairwise Sorting Network

Ian Parberry*
Center for Research in
Parallel and Distributed Computing
Department of Computer Sciences
University of North Texas

Abstract

A new sorting network with exactly the same size and depth as the odd-even sorting network is presented. This sorting network is designed using the zero-one principle, and proceeds by first sorting pairs of bits and sorting the pairs into lexicographic order.

1 Introduction

Sorting networks are a popular model of comparison-based parallel sorting that has been studied for over two decades (for example, see Knuth [5], or Parberry [7]). Sorting networks with optimal size are known for $n \leq 8$ (Knuth [5]), and sorting networks with optimal depth are known for $n \leq 10$ (Parberry [8]). For $n \leq 16$, Knuth [5] contains the sorting networks with best known size and depth. For all practical values of $n > 16$, the best known sorting network is the odd-even sorting network of Batcher [3], which is constructed recursively and has depth $(\log n)(\log n + 1)/2$ and size $n(\log n)(\log n - 1)/4 + n - 1$. For extremely large n (currently approximately 2^{6100}, Paterson [11]), the asymptotically optimal sorting network of Ajtai, Komlós and Szemerédi [1, 2] has superior size and depth, but this is unlikely to be of practical interest.

We will construct a new sorting network, called the *pairwise sorting network*, that has exactly the same size and depth as the odd-even sorting network. Sorting networks with depth $O(\log^2 n)$ are no longer remarkable; for example, an infinite number of $O(\log^2 n)$ depth sorting networks can be constructed using the k-ary version (Parker and Parberry [10]) of Columnsort (Leighton [6]). Other $O(\log^2 n)$ depth sorting networks are known, for example, the periodic balanced sorting network of Dowd *et al.* [4]. However, the pairwise sorting network differs from these in that it is the first sorting network to be competitive with odd-even sort for all values of n. The value of the pairwise sorting network is not that it is

*Research supported by NSF Grant CCR–8801659. Author's address: Department of Computer Sciences, University of North Texas, P.O. Box 13886, Denton, TX 76203-3886, U.S.A. Electronic mail: ian@ponder.csci.unt.edu.

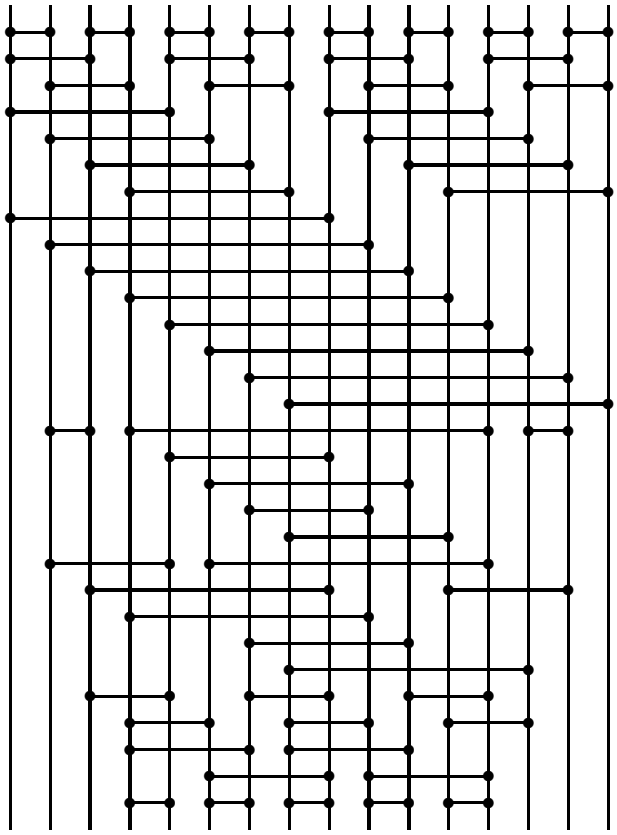

Figure 1: The 16-input sorting network with best known depth.

superior to the odd-even sorting network in any sense, but that it is the first serious rival to appear in over 20 years.

The pairwise sorting network has a somewhat interesting structure, with comparators in the first $\log n$ layers of an n-input network comparing values which are successively increasing powers of two apart. It shares this property with the 16-input sorting network with the smallest known depth, shown in Figure 1. In our diagrams the values being sorted travel downwards along vertical lines to which comparators, depicted as horizontal lines, are attached. Heavy dots are used to emphasize the end-points of comparators. Unsorted data is presented at the top of the network and emerges sorted in nondecreasing order from left to right at the bottom.

The remainder of this paper consists of three short sections. The first contains a high-level description of the algorithm, the second contains more details, and the final section contains the analysis.

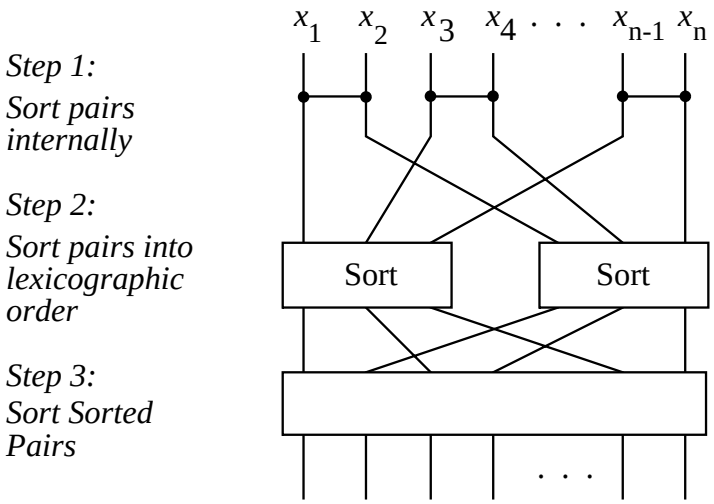

Figure 2: Recursive construction of the pairwise sorting network.

2 The Pairwise Sorting Network

The pairwise sorting network is constructed recursively as follows (see Figure 2). The construction will be guided by the *zero-one principle* (see Knuth [5] or Parberry [7]), which states that it is sufficient to sort sequences of bits. To sort n bits where n is a power of 2:

1. Divide the input into pairs of bits, and sort each pair internally.

2. Sort the pairs into lexicographic order (that is, all 00 pairs before all 01 pairs before all 11 pairs).

3. Sort the sorted pairs into nondecreasing order.

Step 1 is easy to achieve. Suppose we are given a string of n zeros and ones, $x = (x_1, x_2, \ldots, x_n)$, where n is a power of 2. Simply compare x_{2i-1} with x_{2i} and swap them if the former is greater than the latter, for $1 \leq i \leq n/2$.

Step 2 is also relatively easy. Recursively sort $(x_1, x_3, \ldots, x_{n-1})$ and $(x_2, x_4, \ldots, x_n)$ to give $y = (y_1, \ldots, y_{n/2})$ and $z = (z_1, \ldots, z_{n/2})$ respectively. Then z has at least as many ones as y, and hence the string $(y_1, z_1, y_2, z_2, \ldots, y_{n/2}, z_{n/2})$ consists of sorted pairs.

Step 3 is slightly more complicated, and is the subject of the next section.

3 Sorting Sorted Pairs

If $A \in \{0,1\}^{2k}$ is a string of k sorted pairs of bits, that is,

$$A = (a_1, b_1, a_2, b_2, \ldots, a_k, b_k),$$

where $a_i = 0$ for $1 \le i \le p+m$ and 1 otherwise, $b_i = 0$ for $1 \le i \le p$ and 1 otherwise, where $m \ge 0$, then A said to have *prefix* p, *amplitude* m, and *suffix* $k-m-p$. We will without loss of generality assume k is a power of two.

Suppose A is a string of k sorted pairs of bits with prefix p, suffix r, and amplitude $r-p+1 \le m$, where m is a power of 2. For $1 \le i \le k-m/2$, compare b_i with $a_{i+m/2}$ and swap them if the former is larger than the latter. We claim that the resulting string has amplitude at most $m/2$. If $r-p < m/2$ then the comparisons have no effect; in this case the claimed result holds, since A has amplitude at most $m/2$. Otherwise $m/2 \le r-p \le m-1$ and $b_p, \ldots, b_{r-m/2}$ are compared to $a_{p+m/2}, \ldots, a_r$ respectively, where the former are all one and the latter are all zero. Suppose that the result of the comparisons is $A' = (a'_1, b'_1, a'_2, b'_2, \ldots, a'_k, b'_k)$. Then $a'_i, b'_i = 0$ for $1 \le i < p$, $a'_i, b'_i = 1$ for $r < i \le k$, and

$$\begin{array}{ll} b'_i = 0 & \text{for } p \le i \le r - m/2 \\ b'_i = 1 & \text{for } r - m/2 < i \le r \\ a'_i = 0 & \text{for } p \le i < p + m/2 \\ a'_i = 1 & \text{for } p + m/2 \le i \le r. \end{array}$$

Since $r - p \le m - 1$, we know that $r - 1 < p + m/2$ and $p + m/2 > r - m/2$. The former implies that A' has prefix $(r - p + 1 - m/2)$ and the latter implies that it has suffix $(r - p + 1 - m/2)$. Therefore A' has amplitude

$$(r-p+1) - 2(r-p+1-m/2) = (m-1) - (r-p) < m/2.$$

Therefore, a string of k sorted pairs of bits can be sorted in depth $d(k) = \log k$ and size $s(k) = k \log k - k + 1$ by repeating the above procedure on an arbitrary string of sorted pairs with the maximal amplitude descending from $k/2^{i-1}$ to $k/2^i$ with the ith application, for $1 \le i \le \log k$. The ith application uses $k(1 - 1/2^i)$ comparisons, which implies that the size of the network is given by:

$$\sum_{i=1}^{\log k} (k - \frac{k}{2^i}) = k \log k - k + 1.$$

Figure 3 illustrates the construction on 16 inputs. This method for sorting sorted pairs of bits may be useful in other contexts; for example, it is used to reduce the depth of the reduction in Parberry [9].

4 Analysis

It is of some small interest to ask whether the comparator network for sorting sorted pairs (Step 3 of the algorithm) has optimal size and depth. Lower bounds can easily be obtained from the standard lower bounds on the size and depth of merging networks, since the network for sorting sorted pairs can be used to make a merging network, as follows. Suppose we are given two sorted strings of zeros and ones, $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$. Compare x_i with y_i and swap them if the former is larger than the latter, for $1 \le i \le n$. If x has at most as many ones as y, then there is no change. If y has less ones than x, then the net

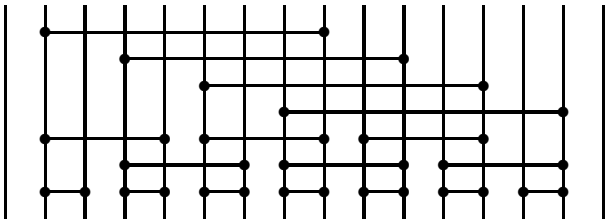

Figure 3: A comparator network for sorting 8 sorted pairs of bits.

effect is to swap x and y. Thus after the sequence of comparisons, x and y remain sorted and y must have at least as many ones as x. Therefore the string $(x_1, y_1, x_2, y_2, \ldots, x_n, y_n)$ is an string of sorted pairs of bits.

Since a network which merges two sorted sequences of n values requires at least $0.5n \log n - O(n)$ comparators (attributed to Floyd by Knuth [5]), we see that any network which sorts a sequence of sorted pairs of bits also requires at least $0.5n \log n - O(n)$ comparators. Thus the comparator network of the previous section has size larger than the optimum by at worst a factor of two. Since a standard argument based on the possible destinations of the smallest element in the right-most sorted subsequence will give a lower-bound of $\log n + 1$ on the depth of a network which merges two sorted sequences of n values, we see that any network which sorts a sequence of n sorted pairs of bits requires depth at least $\log n$. Thus the comparator network of the previous section has optimal depth.

By examination of the third layer one can show that the pairwise sorting network with 16 inputs (Figure 5) is not isomorphic to the 16 input odd-even sorting network (Figure 4), and hence one can prove by induction on n that the n input pairwise sorting network is not isomorphic to the odd-even sorting network. It is also easy to prove that the pairwise sorting network has the same size and depth bounds as Batcher's odd-even sorting network. Let $D(n)$ be the depth of an n-input pairwise sorting network. Then $D(2) = 1$ and for $n \geq 2$,

$$\begin{aligned} D(n) &= D(n/2) + d(n/2) + 1 \\ &= D(n/2) + \log n. \end{aligned}$$

That is, $D(n) = (\log n)(\log n + 1)/2$ (proof by induction on n). Let $S(n)$ be the size of an n-input pairwise sorting network. Then $S(2) = 1$ and for $n > 2$,

$$\begin{aligned} S(n) &= 2S(n/2) + s(n/2) + n/2 \\ &= 2S(n/2) + n(\log n - 1)/2 + 1. \end{aligned}$$

That is, $S(n) = n(\log n)(\log n - 1)/4 + n - 1$ (proof by induction on n).

References

[1] M. Ajtai, J. Komlós, and E. Szemerédi. An $O(n \log n)$ sorting network. *Proc. 15th Ann. ACM Symp. on Theory of Computing*, pages 1–9, April 1983.

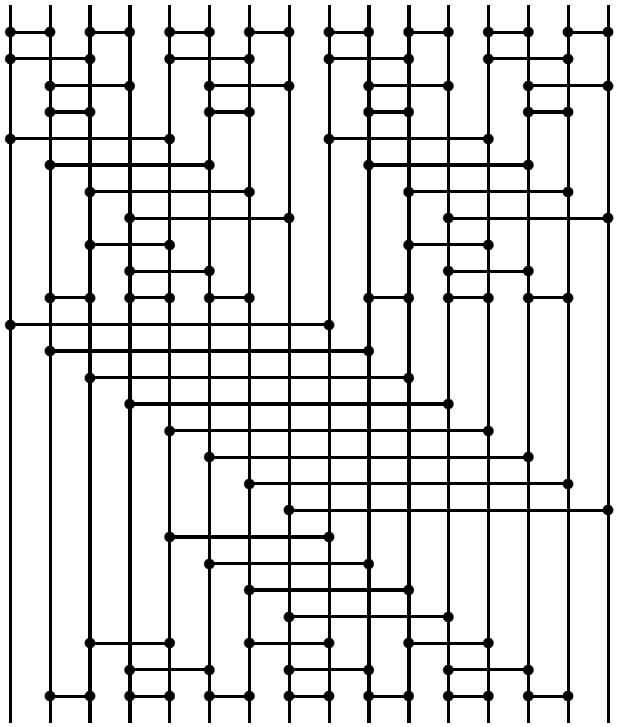

Figure 4: The odd-even sorting network with 16 inputs.

[2] M. Ajtai, J. Komlós, and E. Szemerédi. Sorting in $c \log n$ parallel steps. *Combinatorica*, 3:1–48, 1983.

[3] K. E. Batcher. Sorting networks and their applications. In *Proc. AFIPS Spring Joint Computer Conference*, volume 32, pages 307–314, April 1968.

[4] M. Dowd, Y. Perl, L. Rudolph, and M. Saks. The periodic balanced sorting network. *J. Assoc. Comput. Mach.*, 36(4):738–757, October 1989.

[5] D. E. Knuth. *Sorting and Searching*, volume 3 of *The Art of Computer Programming.* Addison-Wesley, 1973.

[6] F. T. Leighton. Tight bounds on the complexity of parallel sorting. *IEEE Transactions on Computers*, C-34(4):344–354, April 1985.

[7] I. Parberry. *Parallel Complexity Theory.* Research Notes in Theoretical Computer Science. Pitman Publishing, London, 1987.

[8] I. Parberry. A computer-assisted optimal depth lower bound for nine-input sorting networks. *Mathematical Systems Theory*, 24:101–116, 1991.

[9] I. Parberry. On the computational complexity of optimal sorting network verification. In *Proceedings of The Conference on Parallel Architectures and Languages Europe,* in

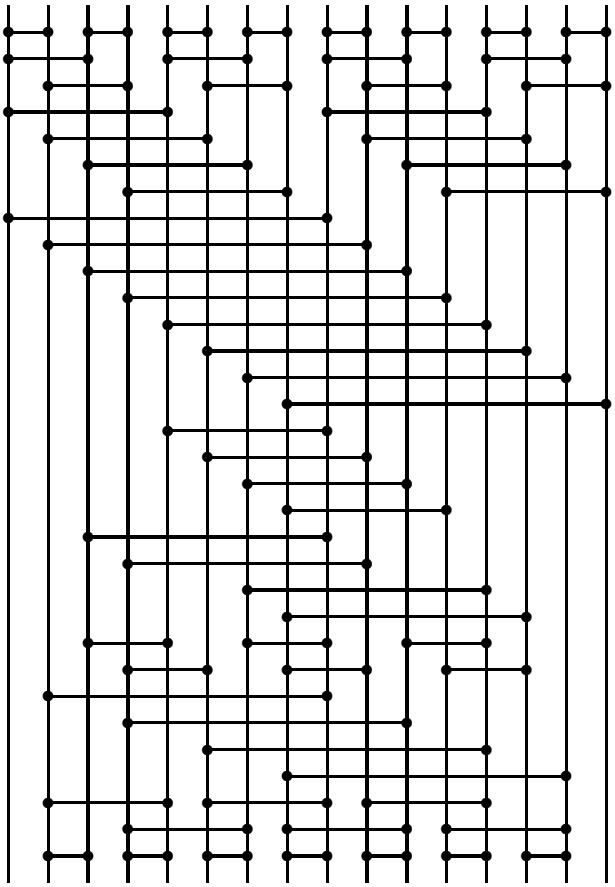

Figure 5: The pairwise sorting network with 16 inputs.

Series *Lecture Notes in Computer Science*, volume 506, pages 252–269. Springer-Verlag, 1991.

[10] B. Parker and I. Parberry. Constructing sorting networks from k-sorters. *Information Processing Letters*, 33(3):157–162, 1989.

[11] M. S. Paterson. Improved sorting networks with $O(\log n)$ depth. *Algorithmica*, 5(4):75–92, 1990.