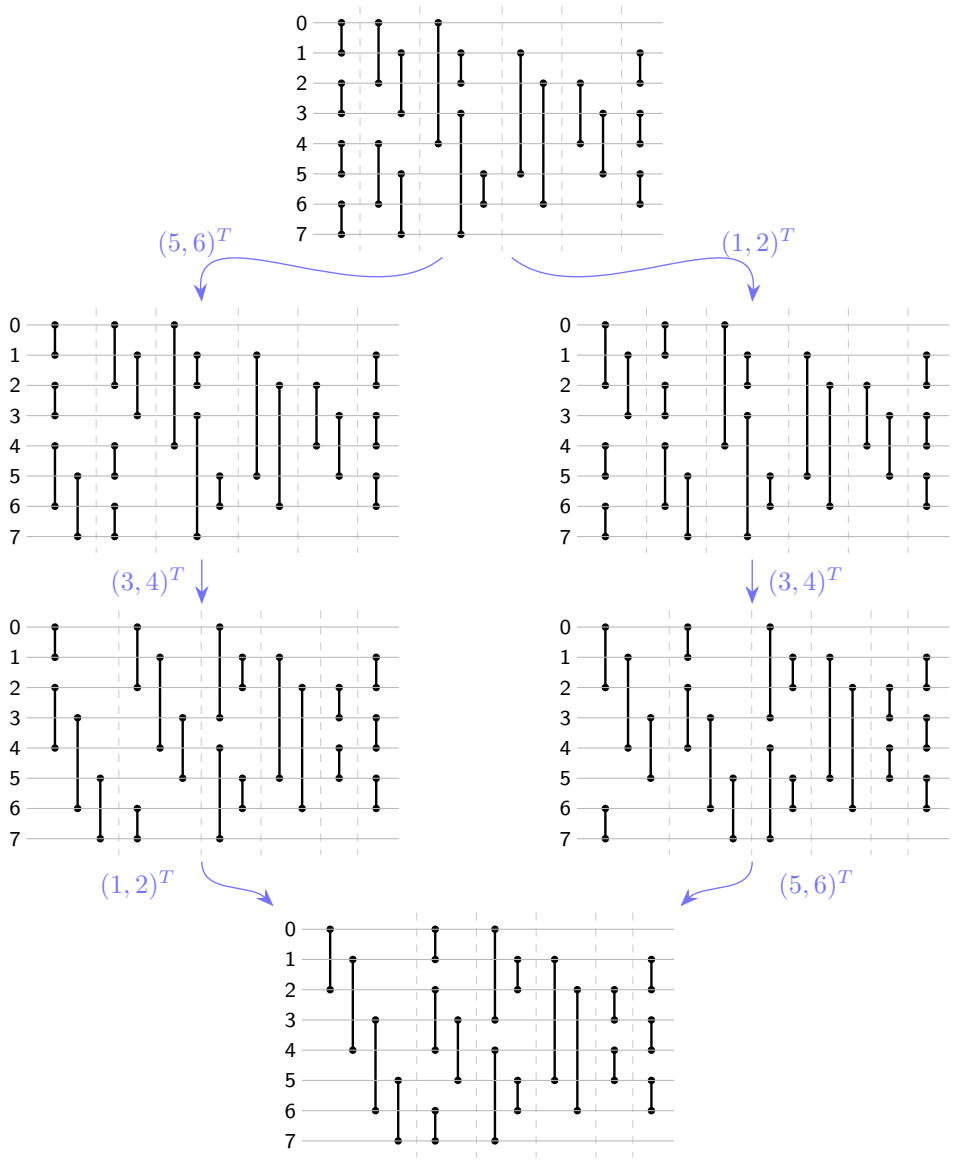
Known-Optimal Sorting Networks for $n = 8$ Inputs

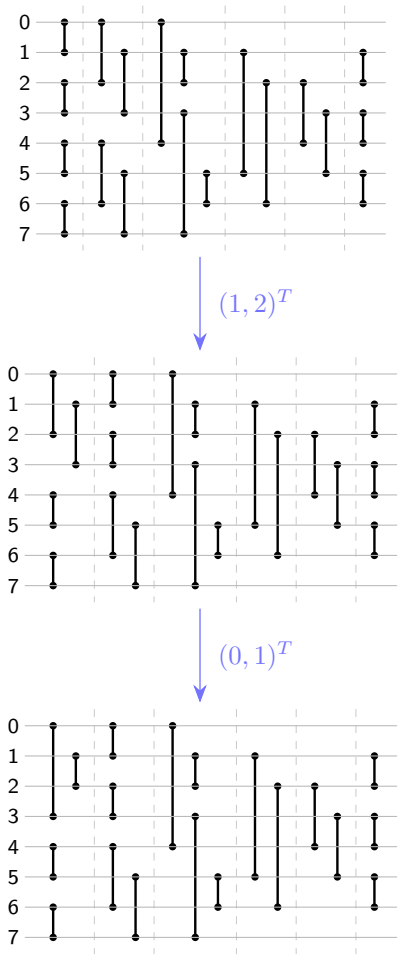
A catalogue with compatible-relabel group forms

Heath Hunnicutt and **Iyun** of Ruach Tov Collective

Overview

Pictured below are an algebraic group of six (6) already-known optimal sorting networks for $n = 8$. The arrows indicate (sub-)group action of the non-idempotent operators $(\sigma_{(1,2)}, \sigma_{(3,4)}, \sigma_{(5,6)})$.




0
1
2
3
4
5
6
7
$(1,2)^T$
0
1
2
3
4
5
6
7
$(0,1)^T$
0
1
2
3
4
5
6
7

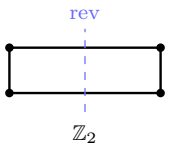
Overview

At $n = 8$ the optimal size is 19 comparators in 6 layers (Knuth; confirmed by exhaustive search). This catalogue holds the distinct optimal networks the collective has generated or verified, plus the classical bitonic network for contrast. Each is verified sorting by the ROBDD reachable-0-1-order decider. For each we give the layer program and the **compatible-relabel group** $G(N) = \{\pi : \pi(N) \text{ still sorts}\}$ — the input-wire renumberings that regenerate an (m, d)-equal network. That such relabelings exist and act on the layers is the general fact underlying the search that settled optimal-depth sorting for $n \leq 16$: Bundala and Závodný (Lemmas 5–7) prove that a maximal first layer is reachable via a compatible relabeling, with an analogous restricted argument for the second layer. The catalogue instantiates that theorem concretely, enumerating each network's orbit directly rather than leaving it abstract.

A primer: symmetry groups

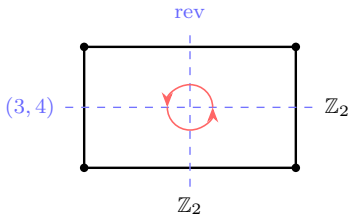
Each compatible-relabel group is a **symmetry group**: the rigid motions fixing a shape, under composition. Reflection axes are dashed, rotations curved in red, set elements (corners) filled.

$\mathbb{Z}_2$



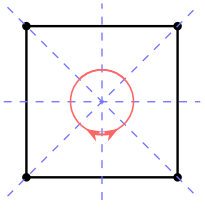
Reflection ($\mathbb{Z}_2$). A single mirror axis; the reflection is an involution ($\sigma^2 = \mathrm{id}$), giving the order-2 group $\{\mathrm{id}, \sigma\}$. For a network the reflection is the **reversal**, the remapping $k' \leftarrow n - k - 1$; every network is symmetric under it.

$\boldsymbol{V_4 = \mathbb{Z}_2 \times \mathbb{Z}_2}$



Two commuting reflections ($V_4 = \mathbb{Z}_2 \times \mathbb{Z}_2$). A rectangle has two commuting reflections whose product is the $180°$ rotation: four elements, **abelian**. Being a **direct** product, the two factors are independent — here the reversal $k' \leftarrow n - k - 1$ and an independent compatible transposition.

$\boldsymbol{D_4 = (\mathbb{Z}_2)^2 \rtimes \mathbb{Z}_2}$



$D_4 = (\mathbb{Z}_2)^2 \rtimes \mathbb{Z}_2$ (non-abelian)

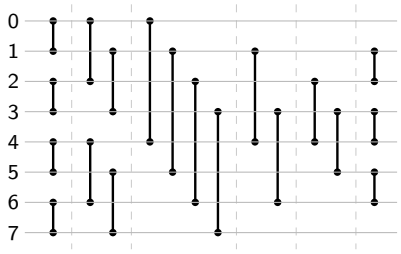
Reflections and rotations ($D_4 = (\mathbb{Z}_2)^2 \rtimes \mathbb{Z}_2$). Four reflections and four rotations, order 8, and **non-abelian**: a reflection inverts a rotation, $s\,r\,s^{-1} = r^{-1}$. One factor **acting** on the other rather than commuting with it is the **semidirect** product $\rtimes$, drawn as two opposite-handed rotation arcs. The larger relabel groups here are all generalized dihedral: the reversal $k' \leftarrow n-k-1$ acting by mirror on a set of commuting transpositions.

A network's compatible-relabel group is the symmetry group of one of these shapes; the same figures recur beside the networks throughout.

Optimal 19/6 (Batcher-derived, shared by Dobbelaere / systematic / Heath hand-derivation)

Size 19, depth 6.

Discharge-verified. The provable-order ledger reaches $28 = \binom{8}{2}$ settled pairs — every pair provably ordered — so the network sorts. Machine-checked in Lean (discharge calculus, not a 2^n scan).



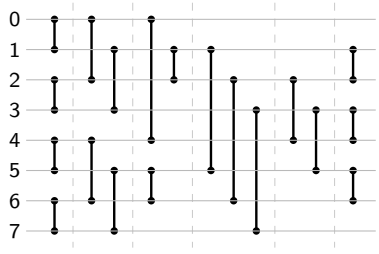
layer	comparators			
L_1	(0,1)	(2,3)	(4,5)	(6,7)
L_2	(0,2)	(1,3)	(4,6)	(5,7)
L_3	(0,4)	(1,5)	(2,6)	(3,7)
L_4	(1,4)	(3,6)		
L_5	(2,4)	(3,5)		
L_6	(1,2)	(3,4)	(5,6)	

Group form: $(\mathbb{Z}_2)^3 \rtimes \mathbb{Z}_2$ ($|G| = 16$), generators $(1,2), (3,4), (5,6), \mathrm{rev}$. Distinct networks generated: 4 (reversal is a self-symmetry, halving the orbit). **Source:** 3 compatible interior transpositions + reversal (reversal acts by the mirror involution).

Batcher odd-even 19/6 (canonical merge)

Size 19, depth 6.

Discharge-verified. The provable-order ledger reaches $28 = \binom{8}{2}$ settled pairs — every pair provably ordered — so the network sorts. Machine-checked in Lean (discharge calculus, not a 2^n scan).



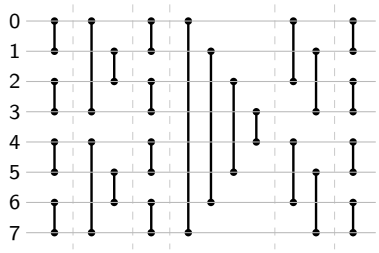
layer	comparators
L_1	(0,1) (2,3) (4,5) (6,7)
L_2	(0,2) (1,3) (4,6) (5,7)
L_3	(0,4) (1,2) (5,6)
L_4	(1,5) (2,6) (3,7)
L_5	(2,4) (3,5)
L_6	(1,2) (3,4) (5,6)

Group form: $\mathbb{Z}_2^2$ (Klein four-group) ($|G| = 4$), generators $(3,4), \mathrm{rev}$. Distinct networks generated: 2 (reversal is a self-symmetry, halving the orbit). **Source:** 1 compatible interior transposition + reversal (reversal acts by the mirror involution).

Bitonic 24/6

Size 24, depth 6.

Discharge-verified. The provable-order ledger reaches $28 = \binom{8}{2}$ settled pairs — every pair provably ordered — so the network sorts. Machine-checked in Lean (discharge calculus, not a 2^n scan).

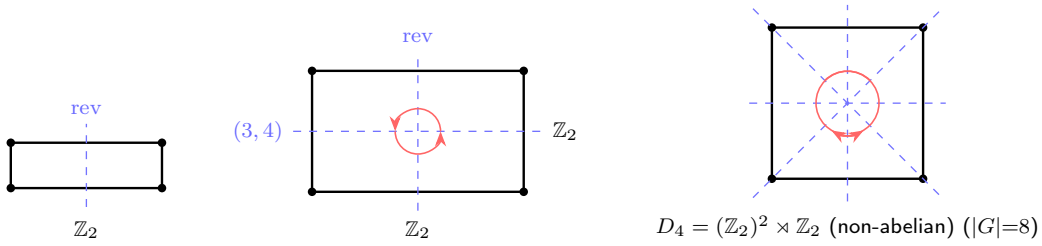


layer	comparators
L_1	(0,1) (2,3) (4,5) (6,7)
L_2	(0,3) (1,2) (4,7) (5,6)
L_3	(0,1) (2,3) (4,5) (6,7)
L_4	(0,7) (1,6) (2,5) (3,4)
L_5	(0,2) (1,3) (4,6) (5,7)
L_6	(0,1) (2,3) (4,5) (6,7)

Group form: $\mathbb{Z}_2$ ($|G| = 2$), generators rev. Distinct networks generated: 1 (reversal is a self-symmetry, halving the orbit). **Source:** 0 compatible interior transpositions + reversal (reversal acts by the mirror involution).

The group forms as symmetry shapes

The compatible-relabel groups are literally geometric symmetry groups. Each network's group is the symmetry group of the shape below — reflection axes dashed, the rotation shown curved.



Bitonic $24/6$: $\mathbb{Z}_2$ Batcher $19/6$: V_4 (order-8 relabel group)

The bitonic network's only symmetry is the reversal (a single reflection axis, $\mathbb{Z}_2$). Batcher's $19/6$ has the symmetries of a rectangle — reversal and the $(3,4)$ swap as the two flip axes, their product the $180°$ rotation — the Klein four-group. The larger relabel groups are dihedral: their reflection-plus-rotation structure is the symmetry of a square (D_4) and its higher analogues.

Summary: group forms and distinct-network counts

network	size/depth	$\lvert G\rvert$	group	distinct nets	rev self-sym
Optimal 19/6	$19/6$	16	$(\mathbb{Z}_2)^3 \rtimes \mathbb{Z}_2$	4	yes
Batcher odd-even 19/6	$19/6$	4	$\mathbb{Z}_2^2$ (Klein four-group)	2	yes
Bitonic 24/6	$24/6$	2	$\mathbb{Z}_2$	1	yes

Reading. The two optimal $19/6$ networks carry different group forms: the shared Dobbelaere/systematic/hand-derived network admits three compatible interior transpositions $(\mathbb{Z}_2)^3 \rtimes \mathbb{Z}_2$ of order 16, while Batcher's canonical merge admits only one $(\mathbb{Z}_2)^1 \rtimes \mathbb{Z}_2$ (a Klein four-group). Same size, same depth, different symmetry — the group form measures the construction, not the cost. The bitonic $24/6$ is rigid ($\mathbb{Z}_2$, reversal only), the price of its extra five comparators paid partly in lost symmetry.

Enumerating the orbit: generator signatures

Each distinct network in an orbit is reached by a subset of the compatible generators. Writing the independent transpositions as **columns** and the reversal as an idempotent factor, a group element is a signature $(b_0, \ldots, b_{t-1})[\cdot\,\mathrm{rev}]$, where $b_i = 1$ means column i's transposition is applied. Because the reversal is idempotent **on a self-mirror network** (it returns the same network) and because it conjugates the columns by the mirror involution, several signatures collapse to the same network.

Shared optimal $19/6$: columns $[(1,2), (3,4), (5,6)]$

The reversal's mirror action swaps column $0 = (1,2)$ with column $2 = (5,6)$ and fixes the central column $1 = (3,4)$. So each network is reached by four signatures — (b_0, b_1, b_2), its reversal, and the two mirror-images with b_0, b_2 swapped. The 16 group elements collapse to 4 distinct networks:

net	canonical signature	all four signatures	comparators changed from net 1
1	$(0,0,0)$	$(0,0,0),\ (0,0,0)\cdot\mathrm{rev},\ (1,0,1),\ (1,0,1)\cdot\mathrm{rev}$	— (the catalogue entry)
2	$(0,0,1)$	$(0,0,1),\ (0,0,1)\cdot\mathrm{rev},\ (1,0,0),\ (1,0,0)\cdot\mathrm{rev}$	$(1,6)\,(2,5)$
3	$(0,1,0)$	$(0,1,0),\ (0,1,0)\cdot\mathrm{rev},\ (1,1,1),\ (1,1,1)\cdot\mathrm{rev}$	$(0,3)\,(4,7)$
4	$(0,1,1)$	$(0,1,1),\ (0,1,1)\cdot\mathrm{rev},\ (1,1,0),\ (1,1,0)\cdot\mathrm{rev}$	$(0,3)\,(1,6)\,(2,5)\,(4,7)$

The four nets are indexed by two mirror-invariant bits: the central column b_1 (self-mirror, a free bit), and whether the outer pair $\{b_0, b_2\}$ is mirror-symmetric ($00/11$) or asymmetric ($01/10$). Two choices times two: four networks. Reversal never adds a fifth, and the central $(3,4)$ transposition — being its own mirror — toggles a network independently of the reversal, which is why b_1 is a clean free bit.

Batcher $19/6$: column $[(3,4)]$

One column, self-mirror, plus reversal. The signatures (0) and (1) give the two distinct networks; each equals its own reversal, so $\cdot\,\mathrm{rev}$ adds nothing. Klein four-group of order 4, orbit of 2.

Reference

D. Bundala and J. Závodný, **Optimal Sorting Networks**, LATA 2014; arXiv:1310.6271. Lemmas 5–7 establish the compatible-relabeling structure this catalogue instantiates.

Appendix A: the group-form computation

Each network's **compatible-relabel group** $G(N)$ above was computed by the following program. It closes the group from its generators — the single-transposition and reversal relabelings that keep N sorting — deciding sorting with the ROBDD reachable-0-1-order decider (not a 2^n scan). The source is reproduced verbatim.

```
#!/usr/bin/env python3
"""groupform.py -- compute the compatible-relabel group of a sorting network.

The *compatible-relabel group* G(N) of a sorting network N on n wires is

    G(N) = { pi in S_n : pi(N) still sorts },

the input-wire renumberings that carry N to another valid sorting network of the same size and depth.
This is the symmetry that the catalogue reports for each network.

We compute it by closure from generators: the single-transposition and reversal relabelings that keep
N sorting generate the whole group (each is an involution or reversal; their closure is G(N)). Sorting
is decided by the ROBDD reachable-0-1-order decider (opus5/order_only) -- not a 2^n scan.

    order = |G(N)|
    is D4 = order 8, non-abelian, contains an order-4 element
    orbit = number of DISTINCT labelled networks pi(N) produces (|G| / |stabiliser|); the reversal is
            often a self-symmetry, halving the orbit.

Usage:
    groupform.py <n> "(0,1)(2,3)..." # comparators in any order
"""
import sys
from itertools import combinations

sys.path.insert(0, "/home/heath/Ruach-Tov/research/sorting-networks")
sys.path.insert(0, "/home/heath/Ruach-Tov/research/sorting-networks/opus5")
import order_only as oo


def relabel(net, phi):
    """Apply wire renumbering phi to every comparator (keeping each min<max)."""
    return [(min(phi[a], phi[b]), max(phi[a], phi[b])) for a, b in net]


def sorts(net, n):
    """Decide sorting via the ROBDD reachable-0-1-order decider."""
    return oo.verify(n, net)["sorts"]


def transposition(n, a, b):
    """The permutation swapping wires a and b, fixing the rest."""
    p = list(range(n))
    p[a], p[b] = p[b], p[a]
    return tuple(p)


def reversal(n):
    """The wire reversal i -> n-1-i."""
    return tuple(n - 1 - i for i in range(n))


def compose(p, q):
    """Permutation composition (p after q)."""
    return tuple(p[q[i]] for i in range(len(p)))


def group(net, n):
    """Return (order, orbit, generators, is_abelian, element_orders) for G(N)."""
```

```
    # generators: every compatible transposition, plus the reversal if compatible
    gens = [transposition(n, a, b) for a, b in combinations(range(n), 2)
            if sorts(relabel(net, transposition(n, a, b)), n)]
    if sorts(relabel(net, reversal(n)), n):
        gens.append(reversal(n))

    identity = tuple(range(n))
    seen = {identity}
    orbit = {tuple(sorted(net))}
    frontier = [identity]
    while frontier:
        g = frontier.pop()
        for h in gens:
            gh = compose(g, h)
            if gh not in seen and sorts(relabel(net, gh), n):
                seen.add(gh)
                orbit.add(tuple(sorted(relabel(net, gh))))
                frontier.append(gh)

    def order_of(g):
        x, k = g, 1
        while x != identity:
            x = compose(x, g)
            k += 1
        return k

    els = list(seen)
    abelian = all(compose(a, b) == compose(b, a) for a in els for b in els)
    return len(seen), len(orbit), gens, abelian, sorted(order_of(g) for g in seen)


def parse(raw):
    out = []
    for tok in raw.replace(")", ") ").split():
        t = tok.strip("(), ")
        if "," in t:
            a, b = t.split(",")
            out.append((int(a), int(b)))
    return out


if __name__ == "__main__":
    n = int(sys.argv[1])
    net = parse(sys.argv[2])
    order, orbit, gens, abelian, ords = group(net, n)
    print(f"|G(N)| = {order}")
    print(f"distinct labelled networks (orbit) = {orbit}")
    print(f"abelian = {abelian}")
    print(f"element orders = {ords}")
    print(f"compatible transpositions = "
          f"{[[(a, b) for a, b in combinations(range(n), 2) if sorts(relabel(net, transposition(n, a, b)), n)
              ]}")
    print(f"reversal compatible = {sorts(relabel(net, reversal(n)), n)}")
```