The Bitonic Sort Algorithm: wrap-up

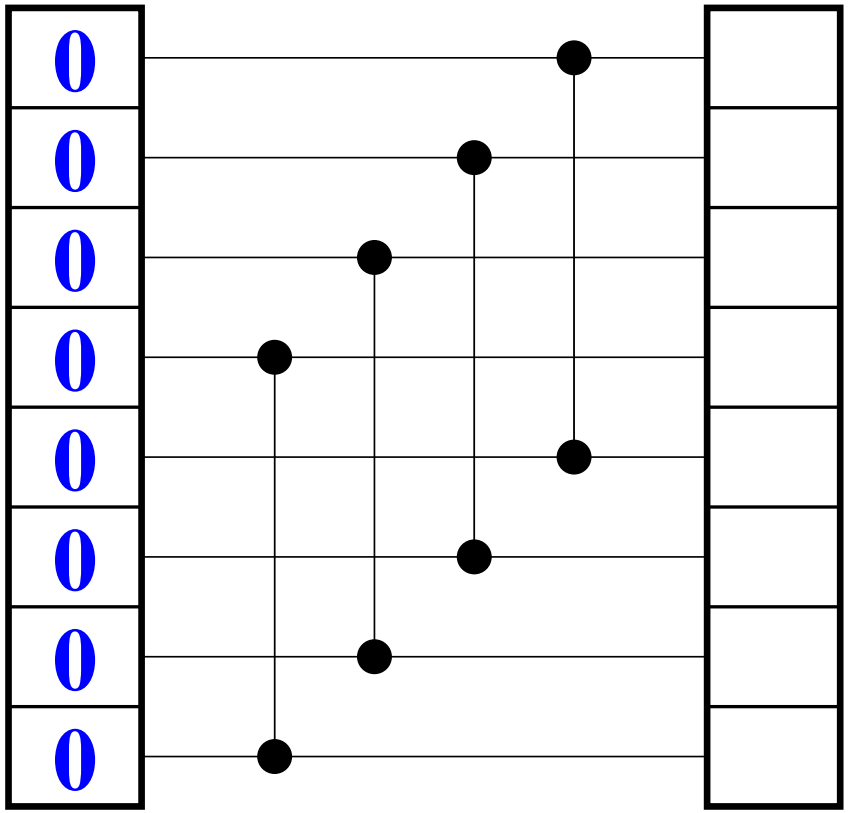
Mark Greenstreet

CpSc 418 – October 24, 2018

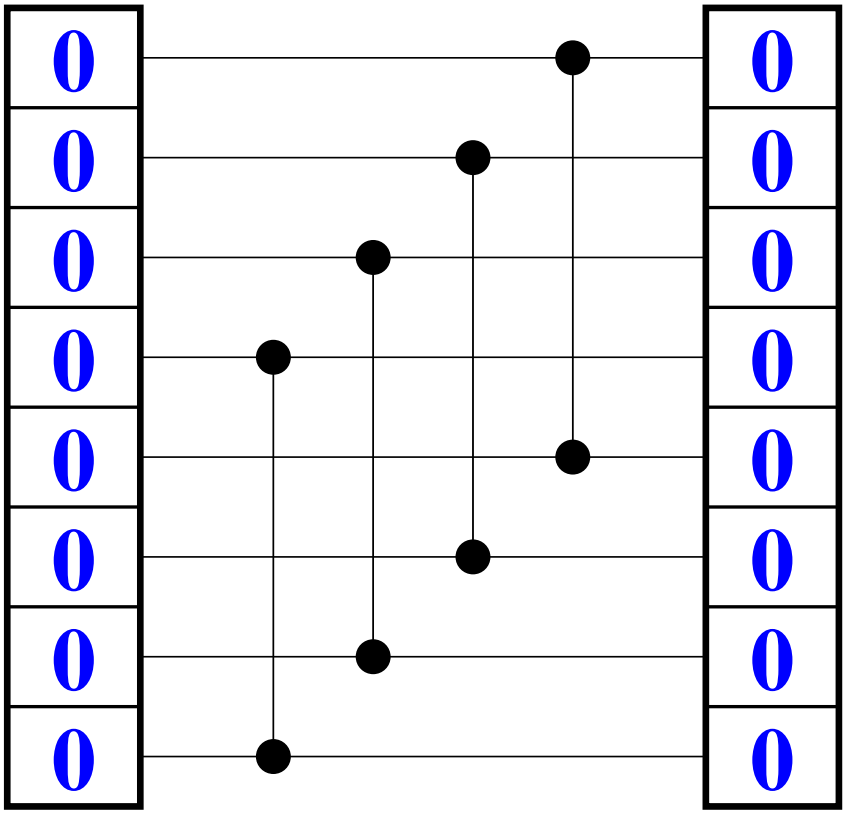
Outline

- Bitonic Sort
  - The easy cases
  - The general case
  - The whole algorithm

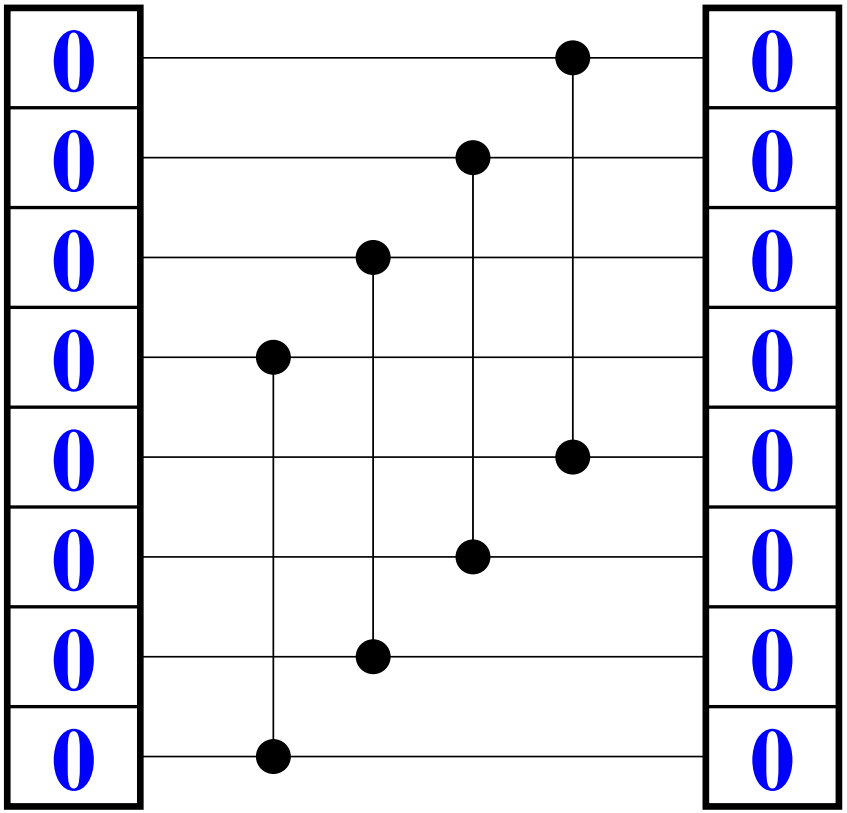
Bitonic Merge Step: Bottom Half All 0s

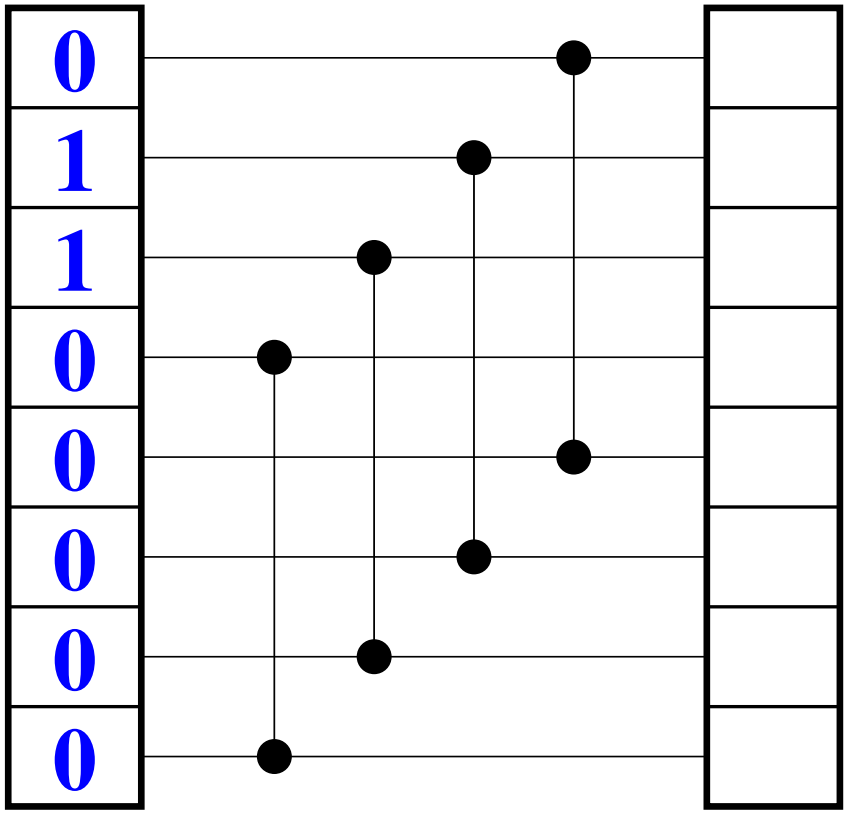


Bitonic Merge Step: Bottom Half All 0s

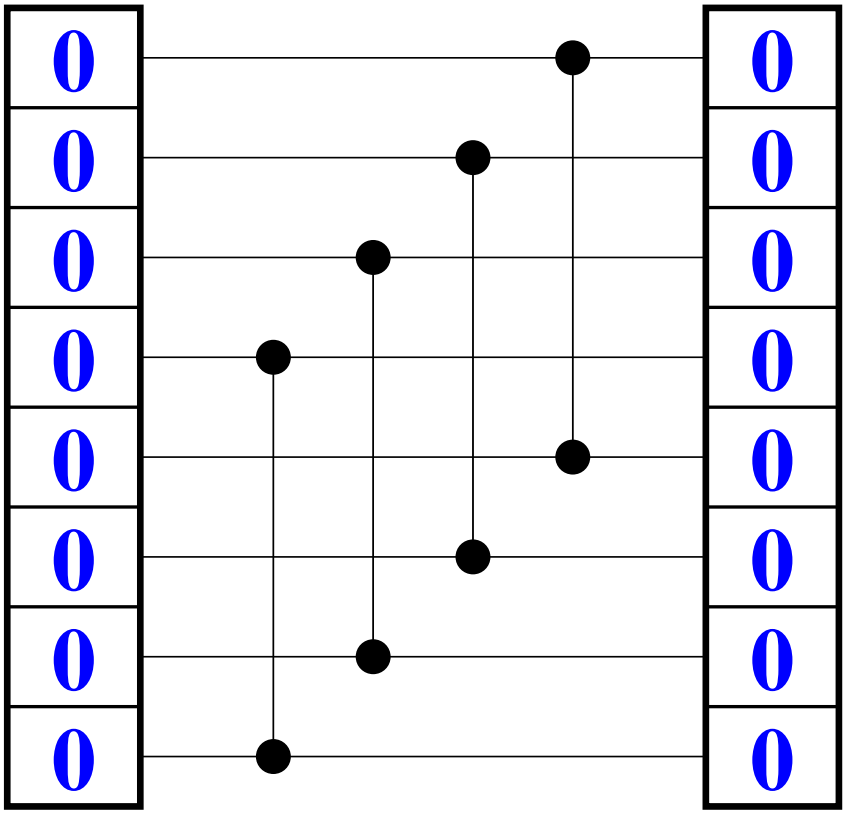


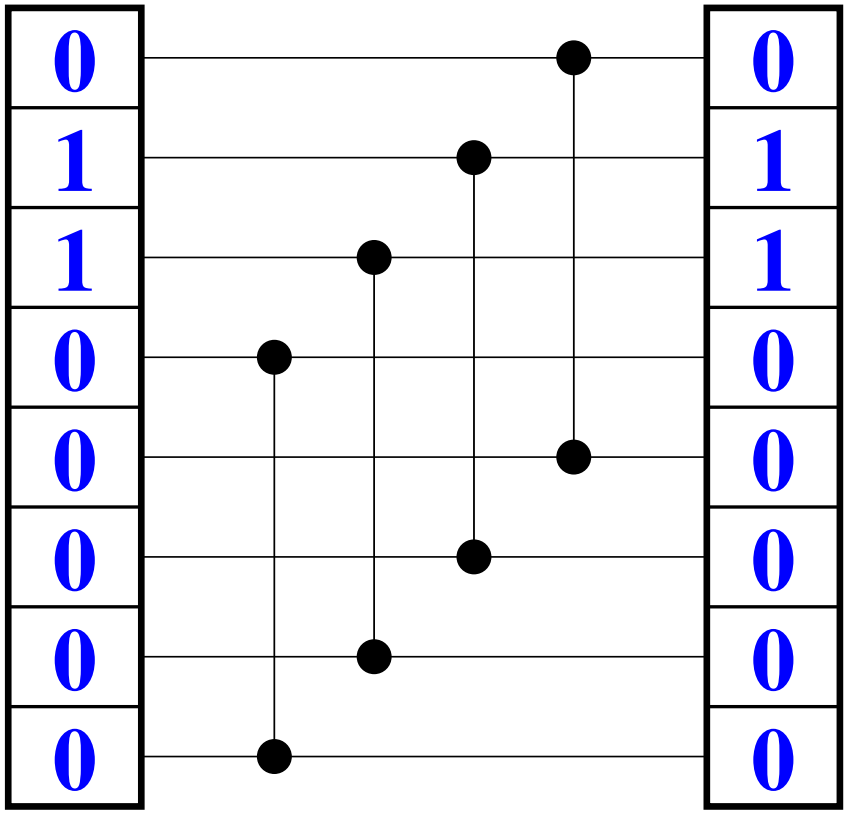
Bitonic Merge Step: Bottom Half All 0s



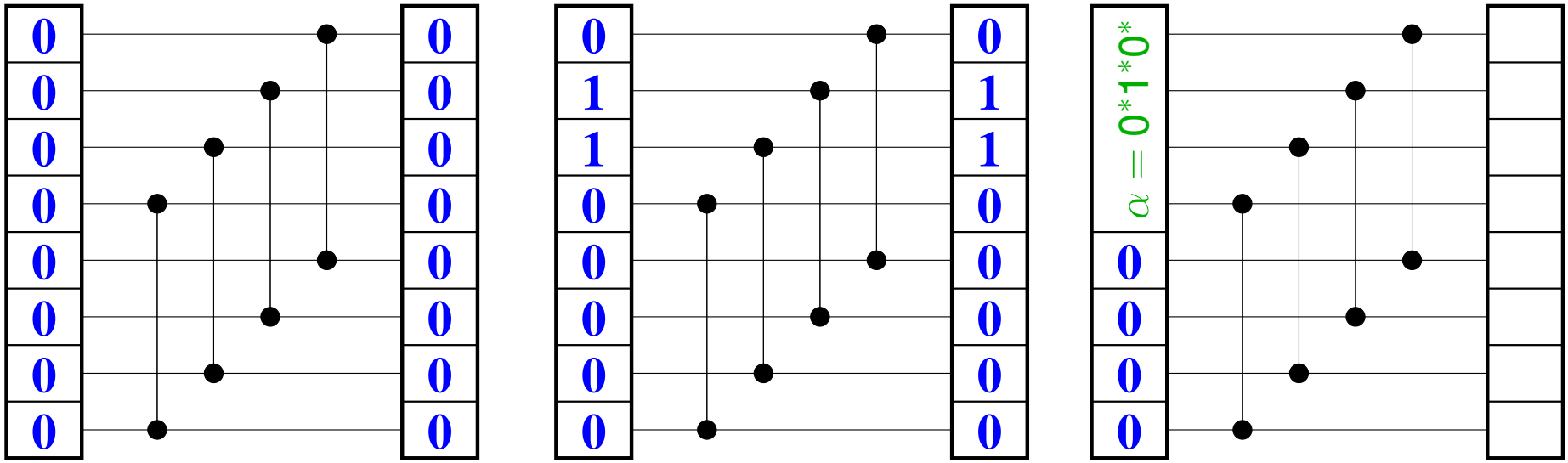


Bitonic Merge Step: Bottom Half All 0s

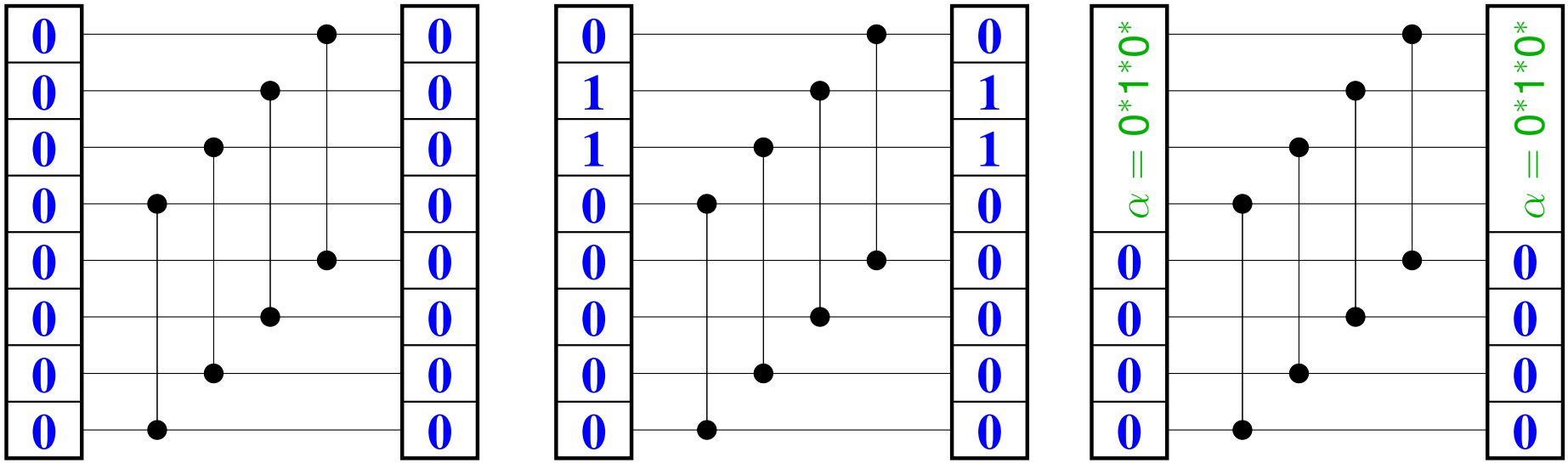




Bitonic Merge Step: Bottom Half All 0s

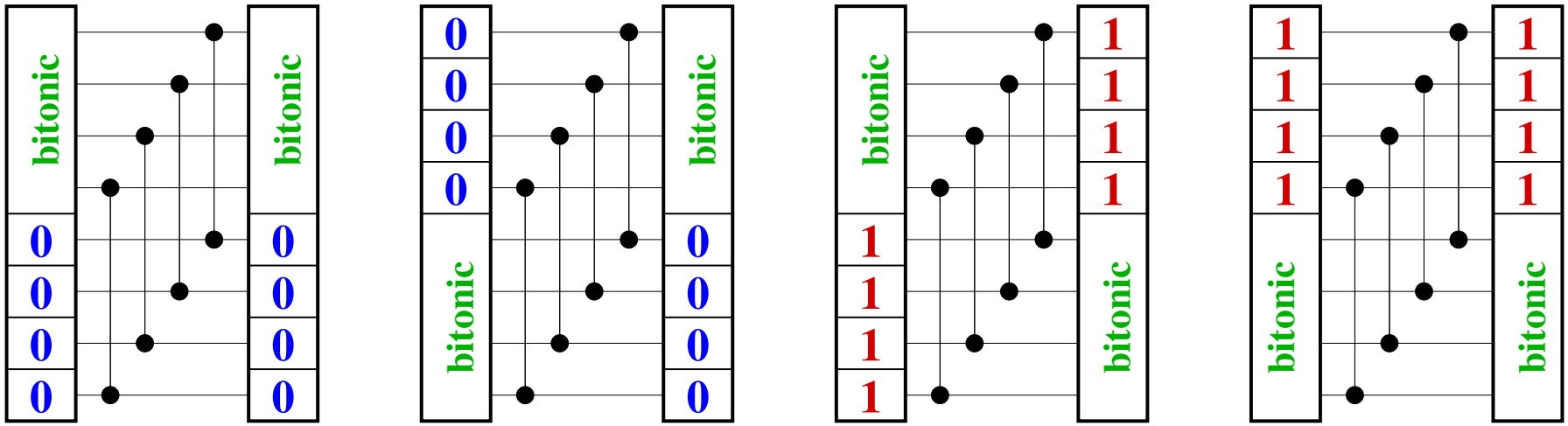


Bitonic Merge Step: Bottom Half All 0s



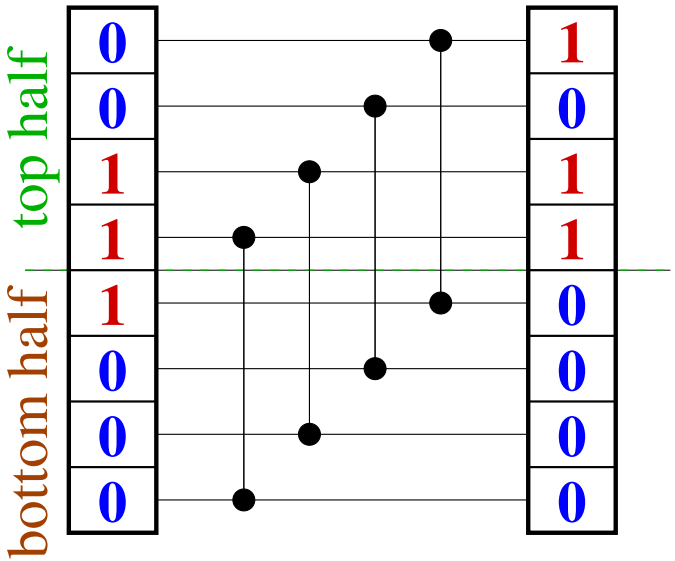
- If the bottom half of the input is all zeros
  - The output is the same as the input.
  - The bottom half is all zeros and the top half is bitonic.

Bitonic Merge Step: Either Half Clean



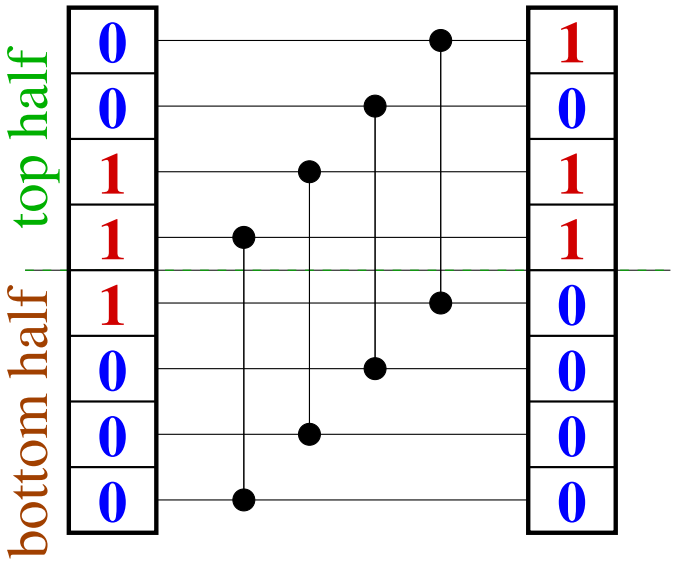
- If a sequence is all 0s or all 1s, we say that it is **clean**.
  - Note: a clean sequence is trivially bitonic.
- If the top or bottom half of the input is clean:
  - Then the top or bottom half of the output is clean.
  - The other half is bitonic.
  - Every element of the bottom half is less than or equal to every element of the top half.
- We'll now show that these properties apply to any bitonic input.

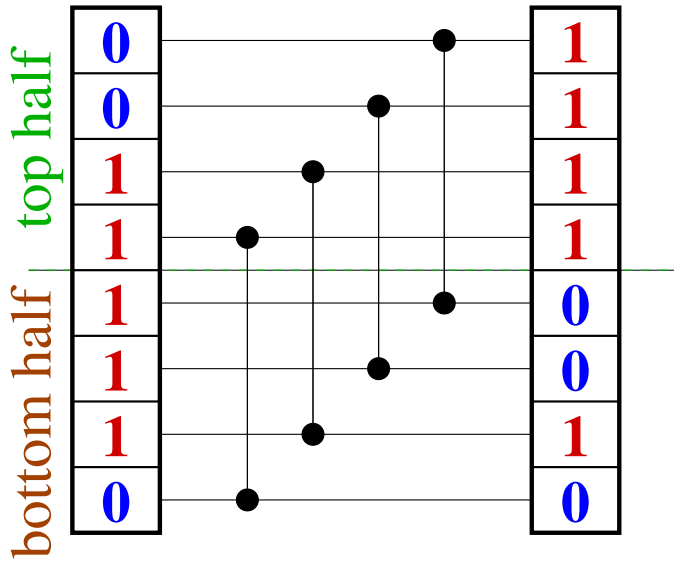
Bitonic Merge Step: Any Bitonic Input



- If an input sequence of length N is bitonic (assume N is even):
- Then
  - Then either the top or bottom half of the output is clean.
  - The other half is bitonic.
  - Every element of the bottom half is less than or equal to every element of the top half.

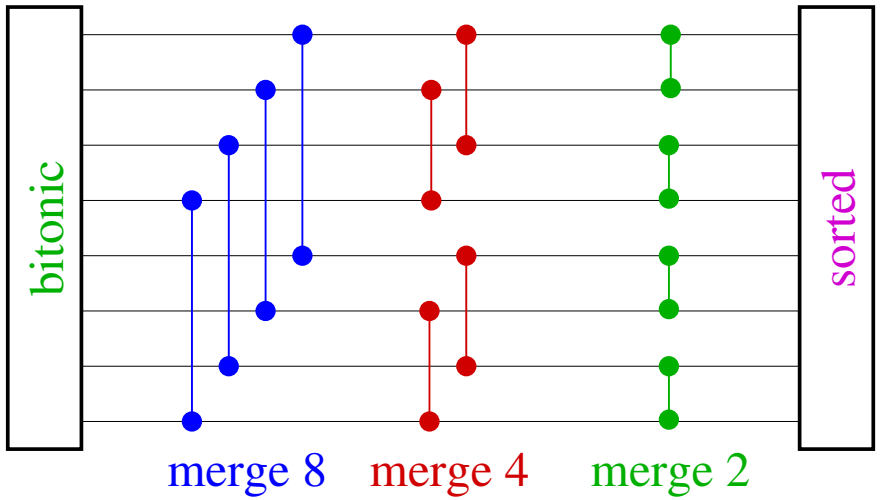
Bitonic Merge Step: Any Bitonic Input





- If an input sequence of length N is bitonic (assume N is even):
- Then
  - Then either the top or bottom half of the output is clean.
  - The other half is bitonic.
  - Every element of the bottom half is less than or equal to every element of the top half.

Bitonic Merge



To sort a bitonic sequence of N values:

- Apply $N/2$ compare and swap operations with stride $N/2$.
  - $Work = N/2$
  - $Span = 1$
- Recursively merge the sequences of length $N/2$
  - $TotalWork = \sum_{k=0}^{(\log_2 N)-1} \frac{N}{2} = \frac{N}{2} \log_2 N$
  - $TotalSpan = \sum_{k=0}^{(\log_2 N)1} = \log_2 N$

Bitonic Sort

- Sort N values – assume N is a power of 2.
- Approach: recursively apply bitonic merge.
  - Input to level k: $N/2^{k-1}$ sorted sequences of length 2^k.
    - ★ Input to first level ($k = 1$): the raw data
  - Output from level k: $N/2^k$ sorted sequences of length k.
    - ★ Output from last level $k = \log_2 N$: sorted sequence.

Bitonic Sort

- Base case $k = 1$:
  - Input: two sequences of length $2^{k-1} = 1$: already sorted.
  - The merge-2 network: a compare-and-swap.
  - Output: a sequence of two, sorted values
- Recursive case:
  - Input: two sequences of length 2^{k-1}: already sorted.
  - Reverse one sequence, concatenate, and apply a 2^k bitonic merge.
  - Output: a sequence of 2^k sorted values.

Bitonic Sort: big-$\mathcal{O}$

Bitonic Sort in Real Life

Bitonic Sort: That's Odd