Definition 1

A 0-1 sequence S is bitonic if it can be written as the concatenation of subsequences S_1 and S_2 such that either

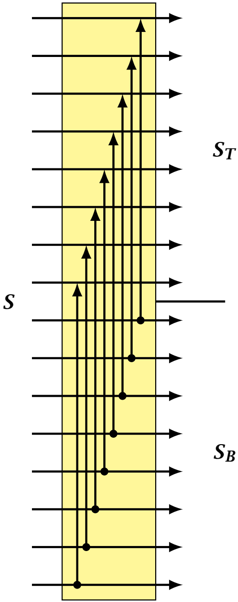
- S_1 is monotonically increasing and S_2 monotonically decreasing, or
- S_1 is monotonically decreasing and S_2 monotonically increasing.

Note, that this just defines bitonic 0-1 sequences. Bitonic sequences are defined differently.

Bitonic Merger

If we feed a bitonic 0-1 sequence S into the network on the right we obtain two bitonic sequences S_T and S_B s.t.

1. $S_B \leq S_T$ (element-wise)
2. S_B and S_T are bitonic



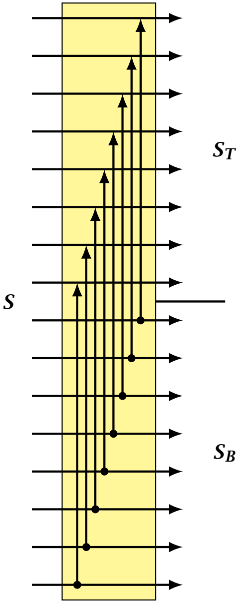
Proof:

- assume wlog. S more 1's than 0's.
- assume for contradiction two 0s at same comparator $(i, j = i + 2^d)$
  - everything 0 btw i and j means we have more than 50% zeros $(\lightning)$.
  - all 1s btw. i and j means we have less than 50% ones $(\lightning)$.
  - 1 btw. i and j and elsewhere means S is not bitonic $(\lightning)$.

Bitonic Merger

If we feed a bitonic 0-1 sequence S into the network on the right we obtain two bitonic sequences S_T and S_B s.t.

1. $S_B \leq S_T$ (element-wise)
2. S_B and S_T are bitonic



Proof:

- assume wlog. S more 1's than 0's.
- assume for contradiction two 0s at same comparator $(i, j = i + 2^d)$
  - everything 0 btw i and j means we have more than 50% zeros (↯).
  - all 1s btw. i and j means we have less than 50% ones (↯).
  - 1 btw. i and j and elsewhere means S is not bitonic (↯).

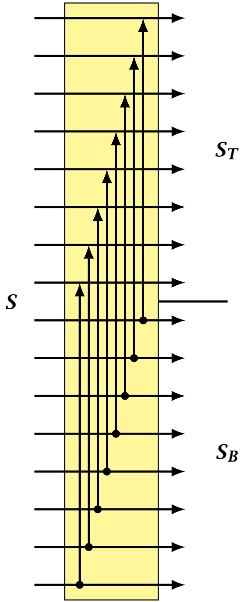
Bitonic Merger

If we feed a bitonic 0-1 sequence S into the network on the right we obtain two bitonic sequences S_T and S_B s.t.

1. $S_B \leq S_T$ (element-wise)
2. S_B and S_T are bitonic

Proof:

- assume wlog. S more 1's than 0's.
- assume for contradiction two 0s at same comparator $(i, j = i + 2^d)$
  - everything 0 btw i and j means we have more than 50% zeros (↯).
  - all 1s btw. i and j means we have less than 50% ones (↯).
  - 1 btw. i and j and elsewhere means S is not bitonic (↯).



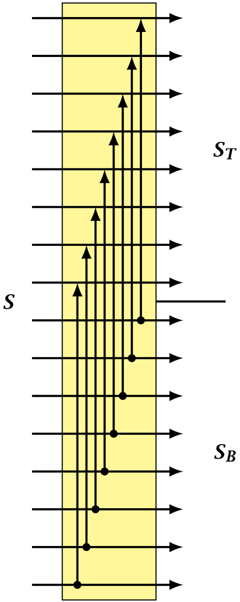
Bitonic Merger

If we feed a bitonic 0-1 sequence S into the network on the right we obtain two bitonic sequences S_T and S_B s.t.

1. $S_B \leq S_T$ (element-wise)
2. S_B and S_T are bitonic

Proof:

- assume wlog. S more 1's than 0's.
- assume for contradiction two 0s at same comparator $(i, j = i + 2^d)$
  - everything 0 btw i and j means we have more than 50% zeros (↯).
  - all 1s btw. i and j means we have less than 50% ones (↯).
  - 1 btw. i and j and elsewhere means S is not bitonic (↯).



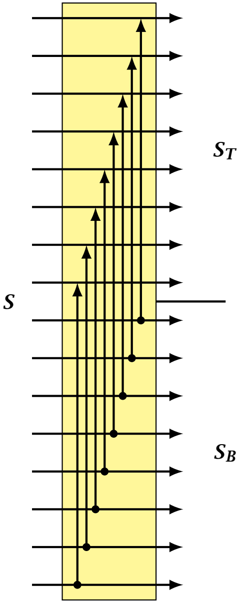
Bitonic Merger

If we feed a bitonic 0-1 sequence S into the network on the right we obtain two bitonic sequences S_T and S_B s.t.

1. $S_B \leq S_T$ (element-wise)
2. S_B and S_T are bitonic

Proof:

- assume wlog. S more 1's than 0's.
- assume for contradiction two 0s at same comparator $(i, j = i + 2^d)$
  - everything 0 btw i and j means we have more than 50% zeros (↯).
  - all 1s btw. i and j means we have less than 50% ones (↯).
  - 1 btw. i and j and elsewhere means S is not bitonic (↯).



Bitonic Merger

Bitonic Merger B_d
The bitonic merger B_d of dimension d is constructed by combining two bitonic mergers of dimension $d-1$.

If we feed a bitonic 0-1 sequence into this, the sequence will be sorted.

(actually, any bitonic sequence will be sorted, but we do not prove this)

Bitonic Merger

Bitonic Merger B_d

The bitonic merger B_d of dimension d is constructed by combining two bitonic mergers of dimension $d-1$.

If we feed a bitonic 0-1 sequence into this, the sequence will be sorted.

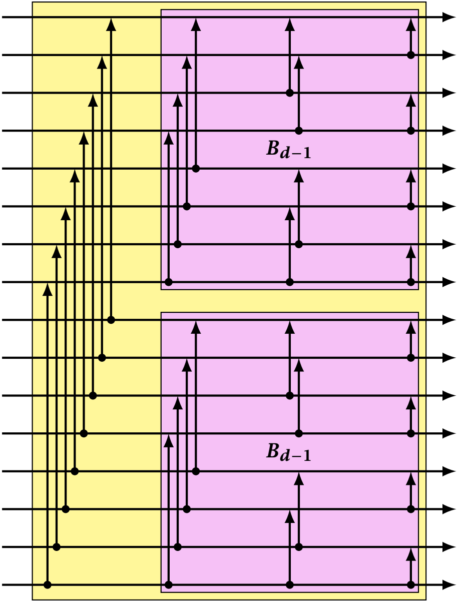
(actually, any bitonic sequence will be sorted, but we do not prove this)

Bitonic Merger

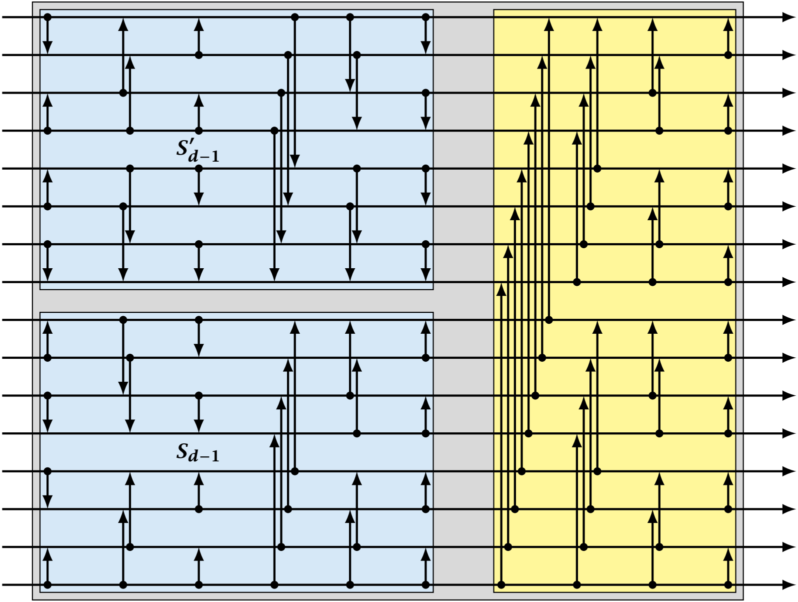
Bitonic Merger B_d
The bitonic merger B_d of dimension d is constructed by combining two bitonic mergers of dimension $d-1$.

If we feed a bitonic 0-1 sequence into this, the sequence will be sorted.

(actually, any bitonic sequence will be sorted, but we do not prove this)



Bitonic Sorter S_d



Bitonic Merger: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + n/2 \Rightarrow C(n) = \mathcal{O}(n \log n)$.
- depth: $D(n) = D(n/2) + 1 \Rightarrow D(d) = \mathcal{O}(\log n)$.

Bitonic Sorter: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + \mathcal{O}(n \log n) \Rightarrow C(n) = \mathcal{O}(n \log^2 n)$.
- depth: $D(n) = D(n/2) + \log n \Rightarrow D(n) = \mathcal{O}(\log^2 n)$.

Bitonic Merger: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + n/2$

Bitonic Merger: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + n/2 \Rightarrow C(n) = \mathcal{O}(n \log n)$.
- depth: $D(n) = D(n/2) + 1 \Rightarrow D(d) = \mathcal{O}(\log n)$.

Bitonic Sorter: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + \mathcal{O}(n \log n) \Rightarrow C(n) = \mathcal{O}(n \log^2 n)$.
- depth: $D(n) = D(n/2) + \log n \Rightarrow D(n) = \mathcal{O}(\log^2 n)$.

Bitonic Merger: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + n/2 \Rightarrow C(n) = \mathcal{O}(n \log n)$.
- depth: $D(n) = D(n/2) + 1 \Rightarrow D(d) = \mathcal{O}(\log n)$.

Bitonic Sorter: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + \mathcal{O}(n \log n) \Rightarrow C(n) = \mathcal{O}(n \log^2 n)$.
- depth: $D(n) = D(n/2) + \log n \Rightarrow D(n) = \mathcal{O}(\log^2 n)$.

Bitonic Merger: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + n/2 \Rightarrow C(n) = \mathcal{O}(n \log n)$.
- depth: $D(n) = D(n/2) + 1 \Rightarrow D(d) = \mathcal{O}(\log n)$.

Bitonic Sorter: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + \mathcal{O}(n \log n) \Rightarrow C(n) = \mathcal{O}(n \log^2 n)$.
- depth: $D(n) = D(n/2) + \log n \Rightarrow D(n) = \mathcal{O}(\log^2 n)$.

Bitonic Merger: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + n/2 \Rightarrow C(n) = \mathcal{O}(n \log n)$.
- depth: $D(n) = D(n/2) + 1 \Rightarrow D(d) = \mathcal{O}(\log n)$.

Bitonic Sorter: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + \mathcal{O}(n \log n) \Rightarrow C(n) = \mathcal{O}(n \log^2 n)$.
- depth: $D(n) = D(n/2) + \log n \Rightarrow D(n) = \mathcal{O}(\log^2 n)$.

Bitonic Merger: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + n/2 \Rightarrow C(n) = \mathcal{O}(n \log n)$.
- depth: $D(n) = D(n/2) + 1 \Rightarrow D(d) = \mathcal{O}(\log n)$.

Bitonic Sorter: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + \mathcal{O}(n \log n) \Rightarrow$
  $C(n) = \mathcal{O}(n \log^2 n)$.
- depth: $D(n) = D(n/2) + \log n \Rightarrow D(n) = \Theta(\log^2 n)$.

Bitonic Merger: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + n/2 \Rightarrow C(n) = \mathcal{O}(n \log n)$.
- depth: $D(n) = D(n/2) + 1 \Rightarrow D(d) = \mathcal{O}(\log n)$.

Bitonic Sorter: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + \mathcal{O}(n \log n) \Rightarrow$ $C(n) = \mathcal{O}(n \log^2 n)$.
- depth: $D(n) = D(n/2) + \log n \Rightarrow D(n) = \Theta(\log^2 n)$.

Bitonic Merger: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + n/2 \Rightarrow C(n) = \mathcal{O}(n \log n)$.
- depth: $D(n) = D(n/2) + 1 \Rightarrow D(d) = \mathcal{O}(\log n)$.

Bitonic Sorter: ($n = 2^d$)

- comparators: $C(n) = 2C(n/2) + \mathcal{O}(n \log n) \Rightarrow$ $C(n) = \mathcal{O}(n \log^2 n)$.
- depth: $D(n) = D(n/2) + \log n \Rightarrow D(n) = \Theta(\log^2 n)$.

Odd-Even Merge

How to merge two sorted sequences?

$A = (a_1, a_2, \dots, a_n)$, $B = (b_1, b_2, \dots, b_n)$, n even.

Split into odd and even sequences:

$A_{\text{odd}} = (a_1, a_3, a_5, \dots, a_{n-1})$, $A_{\text{even}} = (a_2, a_4, a_6, \dots a_n)$

$B_{\text{odd}} = (b_1, b_3, b_5, \dots, b_{n-1})$, $B_{\text{even}} = (b_2, b_4, b_6, \dots, b_n)$

Let

$$X = \text{merge}(A_{\text{odd}}, B_{\text{odd}}) \text{ and } Y = \text{merge}(A_{\text{even}}, B_{\text{even}})$$

Then

$$S = (x_1, \min\{x_2, y_1\}, \max\{x_2, y_1\}, \min\{x_3, y_2\}, \dots, y_n)$$

Odd-Even Merge

How to merge two sorted sequences?

$A = (a_1, a_2, \ldots, a_n)$, $B = (b_1, b_2, \ldots, b_n)$, n even.

Split into odd and even sequences:

$A_{\text{odd}} = (a_1, a_3, a_5, \ldots, a_{n-1})$, $A_{\text{even}} = (a_2, a_4, a_6, \ldots a_n)$

$B_{\text{odd}} = (b_1, b_3, b_5, \ldots, b_{n-1})$, $B_{\text{even}} = (b_2, b_4, b_6, \ldots, b_n)$

Let

$$X = \mathrm{merge}(A_{\text{odd}}, B_{\text{odd}}) \text{ and } Y = \mathrm{merge}(A_{\text{even}}, B_{\text{even}})$$

Then

$$S = (x_1, \min\{x_2, y_1\}, \max\{x_2, y_1\}, \min\{x_3, y_2\}, \ldots, y_n)$$

Odd-Even Merge

How to merge two sorted sequences?

$A = (a_1, a_2, \dots, a_n)$, $B = (b_1, b_2, \dots, b_n)$, n even.

Split into odd and even sequences:

$A_{\text{odd}} = (a_1, a_3, a_5, \dots, a_{n-1})$, $A_{\text{even}} = (a_2, a_4, a_6, \dots a_n)$

$B_{\text{odd}} = (b_1, b_3, b_5, \dots, b_{n-1})$, $B_{\text{even}} = (b_2, b_4, b_6, \dots, b_n)$

Let

$$X = \mathrm{merge}(A_{\text{odd}}, B_{\text{odd}}) \text{ and } Y = \mathrm{merge}(A_{\text{even}}, B_{\text{even}})$$

Then

$$S = (x_1, \min\{x_2, y_1\}, \max\{x_2, y_1\}, \min\{x_3, y_2\}, \dots, y_n)$$

Odd-Even Merge

How to merge two sorted sequences?

$A = (a_1, a_2, \ldots, a_n)$, $B = (b_1, b_2, \ldots, b_n)$, n even.

Split into odd and even sequences:

$A_{\text{odd}} = (a_1, a_3, a_5, \ldots, a_{n-1})$, $A_{\text{even}} = (a_2, a_4, a_6, \ldots a_n)$

$B_{\text{odd}} = (b_1, b_3, b_5, \ldots, b_{n-1})$, $B_{\text{even}} = (b_2, b_4, b_6, \ldots, b_n)$

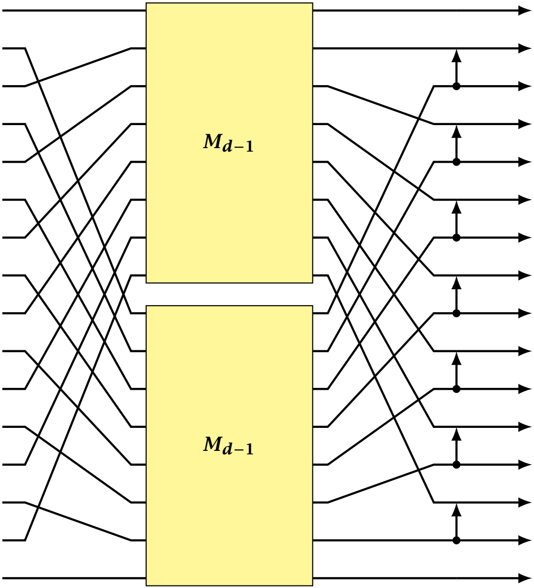
Let

$$X = \text{merge}(A_{\text{odd}}, B_{\text{odd}}) \text{ and } Y = \text{merge}(A_{\text{even}}, B_{\text{even}})$$

Then

$$S = (x_1, \min\{x_2, y_1\}, \max\{x_2, y_1\}, \min\{x_3, y_2\}, \ldots, y_n)$$

Odd-Even Merge



Theorem 2

There exists a sorting network with depth $\mathcal{O}(\log n)$ and $\mathcal{O}(n \log n)$ comparators.