A Glimpse at the AKS Network

There exists a sorting network with depth $O(\log n)$. 

Ajtai, Komlós, Szemerédi (1983)

Quite elaborate construction, and involves huge constants.

Perfect Halver

Perfect halver of depth $\log 2 n$ exist $\Rightarrow$ yields sorting networks of depth $\Theta((\log n)^2)$.

Approximate Halver

We will prove that such networks can be constructed in constant depth!
A Glimpse at the AKS Network

There exists a sorting network with depth $O(\log n)$.

Ajtai, Komlós, Szemerédi (1983)

Perfect Halver

A perfect halver is a comparator network that, given any input, places the $n/2$ smaller keys in $b_1, \ldots, b_{n/2}$ and the $n/2$ larger keys in $b_{n/2+1}, \ldots, b_n$. 

Approximate Halver

We will prove that such networks can be constructed in constant depth!
A Glimpse at the AKS Network

Ajtai, Komlós, Szemerédi (1983)

There exists a sorting network with depth $O(\log n)$.

Perfect Halver

A perfect halver is a comparator network that, given any input, places the $n/2$ smaller keys in $b_1, \ldots, b_{n/2}$ and the $n/2$ larger keys in $b_{n/2+1}, \ldots, b_n$.

Perfect halver of depth $\log_2 n$ exist $\leadsto$ yields sorting networks of depth $\Theta((\log n)^2)$. 
A Glimpse at the AKS Network

Ajtai, Komlós, Szemerédi (1983)

There exists a sorting network with depth \( O(\log n) \).

Perfect Halver

A \textit{perfect halver} is a comparator network that, given any input, places the \( n/2 \) smaller keys in \( b_1, \ldots, b_{n/2} \) and the \( n/2 \) larger keys in \( b_{n/2+1}, \ldots, b_n \).

Approximate Halver

An \((n, \epsilon)\)-\textit{approximate halver}, \( \epsilon < 1 \), is a comparator network that for every \( k = 1, 2, \ldots, n/2 \) places at most \( \epsilon k \) of its \( k \) smallest keys in \( b_{n/2+1}, \ldots, b_n \) and at most \( \epsilon k \) of its \( k \) largest keys in \( b_1, \ldots, b_{n/2} \).
A Glimpse at the AKS Network

There exists a sorting network with depth $O(\log n)$.

A perfect halver is a comparator network that, given any input, places the $n/2$ smaller keys in $b_1, \ldots, b_{n/2}$ and the $n/2$ larger keys in $b_{n/2+1}, \ldots, b_n$.

An $(n, \epsilon)$-approximate halver, $\epsilon < 1$, is a comparator network that for every $k = 1, 2, \ldots, n/2$ places at most $\epsilon k$ of its $k$ smallest keys in $b_{n/2+1}, \ldots, b_n$ and at most $\epsilon k$ of its $k$ largest keys in $b_1, \ldots, b_{n/2}$.

We will prove that such networks can be constructed in constant depth!
Expander Graphs

A bipartite \((n, d, \mu)\)-expander is a graph with:
- \(G\) has \(n\) vertices (\(n/2\) on each side)
- the edge-set is the union of \(d\) matchings
- For every subset \(S \subseteq V\) being in one part,
  \[|N(S)| \geq \min\{\mu \cdot |S|, n/2 - |S|\}\]
A bipartite \((n, d, \mu)\)-expander is a graph with:

- \(G\) has \(n\) vertices (\(n/2\) on each side)
- the edge-set is the union of \(d\) matchings
- For every subset \(S \subseteq V\) being in one part,
  \[
  |N(S)| \geq \min\{\mu \cdot |S|, n/2 - |S|\}
  \]
Expander Graphs

A bipartite \((n, d, \mu)\)-expander is a graph with:

- \(G\) has \(n\) vertices \((n/2\) on each side)\
- the edge-set is the union of \(d\) matchings\
- For every subset \(S \subseteq V\) being in one part,

\[
|N(S)| \geq \min\{\mu \cdot |S|, n/2 - |S|\}
\]
A bipartite \((n, d, \mu)\)-expander is a graph with:

- \(G\) has \(n\) vertices \((n/2\) on each side) 
- the edge-set is the union of \(d\) matchings 
- For every subset \(S \subseteq V\) being in one part, 
  \[ |N(S)| \geq \min\{\mu \cdot |S|, n/2 - |S|\]
A bipartite \((n, d, \mu)\)-expander is a graph with:
- \(G\) has \(n\) vertices \((n/2\) on each side)
- the edge-set is the union of \(d\) matchings
- For every subset \(S \subseteq V\) being in one part,
  \[|N(S)| \geq \min\{\mu \cdot |S|, n/2 - |S|\}\]

Expander Graphs:
- **probabilistic construction** "easy": take \(d\) (disjoint) random matchings
- **explicit construction** is a deep mathematical problem with ties to number theory, group theory, combinatorics etc.
- **many applications** in networking, complexity theory and coding theory
From Expanders to Approximate Halvers
From Expanders to Approximate Halvers
From Expanders to Approximate Halvers

I. Sorting Networks
Batcher's Sorting Network
From Expanders to Approximate Halvers

I. Sorting Networks

Batcher’s Sorting Network
From Expanders to Approximate Halvers

I. Sorting Networks

Batcher's Sorting Network
From Expanders to Approximate Halvers

I. Sorting Networks

Batcher's Sorting Network
From Expanders to Approximate Halvers

I. Sorting Networks

Batcher’s Sorting Network
From Expanders to Approximate Halvers

I. Sorting Networks

Batcher's Sorting Network
Existence of Approximate Halvers

Proof:

Let \( X \) be the set of wires with the \( k \) smallest inputs.

Let \( Y \) be the set of wires in the lower half with \( k \) smallest outputs.

For every \( u \in N(Y) \):

There exists a comparator \((u, v)\).

Let \( u_t, v_t \) be their keys after the comparator.

Let \( u_d, v_d \) be their keys at the output.

Note that \( v_d \in Y \subseteq X \).

Further:

\[ u_d \leq u_t \leq v_t \leq v_d \]

\[ \Rightarrow u_d \in X \]

Since \( u \) was arbitrary:

\[ |Y| + |N(Y)| \leq k \]

Since \( G \) is a bipartite \((n, d, \mu)\)-expander:

\[ |Y| + |N(Y)| \geq |Y| + \min\{|Y|, n/2 - |Y|\} \]

\[ = \min\{|(1 + \mu)|Y|, n/2\} \]

Combining the two bounds above yields:

\[ (1 + \mu)|Y| \leq k \]

The same argument shows that at most \( \epsilon \cdot k \), \( \epsilon := \frac{1}{\mu + 1} \), of the \( k \) largest input keys are placed in \( b_1, \ldots, b_{n/2} \).

Here we used that \( k \leq n/2 \).
Existence of Approximate Halvers

Proof:
- $X :=$ wires with the $k$ smallest inputs
Existence of Approximate Halvers

Proof:  
- $X :=$ wires with the $k$ smallest inputs
- $Y :=$ wires in lower half with $k$ smallest outputs

Proof Strategy:
1. $|N(Y)| > |Y|
2. $|N(Y)| \leq |X| = k$
Existence of Approximate Halvers

Proof:
- \( X := \) wires with the \( k \) smallest inputs
- \( Y := \) wires in lower half with \( k \) smallest outputs
- For every \( u \in N(Y) \): \( \exists \) comparator \( (u, v), v \in Y \)
Existence of Approximate Halvers

Proof:

- \( X := \) wires with the \( k \) smallest inputs
- \( Y := \) wires in lower half with \( k \) smallest outputs
- For every \( u \in N(Y) \): \( \exists \) comparator \((u, v)\)
- Let \( u_t, v_t \) be their keys after the comparator
  - Let \( u_d, v_d \) be their keys at the output
Existence of Approximate Halvers

Proof:

- $X$ := wires with the $k$ smallest inputs
- $Y$ := wires in lower half with $k$ smallest outputs
- For every $u \in N(Y)$: $\exists$ comparator $(u, v)$
- Let $u_t, v_t$ be their keys after the comparator
- Let $u_d, v_d$ be their keys at the output

Note that $v_d \in Y \subseteq X$

Further:

$u_d \leq u_t \leq v_t \leq v_d \Rightarrow u_d \in X$

Since $u$ was arbitrary:

$|Y| + |N(Y)| \leq k$.

Since $G$ is a bipartite $(n, d, \mu)$-expander:

$|Y| + |N(Y)| \geq |Y| + \min\{\mu \cdot |Y|, \frac{n}{2} - |Y|\} = \min\{\left(1 + \frac{\mu}{\mu + 1}\right) |Y|, \frac{n}{2}\}$.

Combining the two bounds above yields:

$\left(1 + \frac{\mu}{\mu + 1}\right) |Y| \leq k$.

The same argument shows that at most $\epsilon \cdot k$, $\epsilon := \frac{1}{\mu + 1}$, of the $k$ largest input keys are placed in $b_1, \ldots, b_n/2$.

Here we used that $k \leq n/2$. 

I. Sorting Networks

Batcher’s Sorting Network 22
Existence of Approximate Halvers

Proof:

- $X$ := wires with the $k$ smallest inputs
- $Y$ := wires in lower half with $k$ smallest outputs
- For every $u \in N(Y)$: $\exists$ comparator $(u, v)$
- Let $u_t, v_t$ be their keys after the comparator
  Let $u_d, v_d$ be their keys at the output
- Note that $v_d \in Y \subseteq X$

Diagram showing the comparison process and the keys before and after the comparator.
Existence of Approximate Halvers

Proof:

- \( X := \) wires with the \( k \) smallest inputs
- \( Y := \) wires in lower half with \( k \) smallest outputs
- For every \( u \in N(Y) \): \( \exists \) comparator \((u, v)\)
- Let \( u_t, v_t \) be their keys after the comparator
  - Let \( u_d, v_d \) be their keys at the output
- Note that \( v_d \in Y \subseteq X \)
- Further: \( u_d \leq u_t \leq v_t \leq v_d \)
Existence of Approximate Halvers

Proof:

- \( X := \) wires with the \( k \) smallest inputs
- \( Y := \) wires in lower half with \( k \) smallest outputs
- For every \( u \in N(Y) \): \( \exists \) comparator \((u, v)\)
- Let \( u_t, v_t \) be their keys after the comparator
- Let \( u_d, v_d \) be their keys at the output
- Note that \( v_d \in Y \subseteq X \)
- Further: \( u_d \leq u_t \leq v_t \leq v_d \Rightarrow u_d \in X \)
- Since \( u \) was arbitrary:

\[
|Y| + |N(Y)| \leq k.
\]
Existence of Approximate Halvers

Proof:

- \( X := \) wires with the \( k \) smallest inputs
- \( Y := \) wires in lower half with \( k \) smallest outputs
- For every \( u \in \mathcal{N}(Y) \): \( \exists \) comparator \((u, v)\)
- Let \( u_t, v_t \) be their keys after the comparator
  - Let \( u_d, v_d \) be their keys at the output
- Note that \( v_d \in Y \subseteq X \)
- Further: \( u_d \leq u_t \leq v_t \leq v_d \Rightarrow u_d \in X \)
- Since \( u \) was arbitrary:
  \[
  |Y| + |\mathcal{N}(Y)| \leq k.
  \]

- Since \( G \) is a bipartite \((n, d, \mu)\)-expander:
Existence of Approximate Halvers

Proof:

- \( X := \) wires with the \( k \) smallest inputs
- \( Y := \) wires in lower half with \( k \) smallest outputs
- For every \( u \in N(Y) \): \( \exists \) comparator \((u, v)\)
- Let \( u_t, v_t \) be their keys after the comparator
  - Let \( u_d, v_d \) be their keys at the output
- Note that \( v_d \in Y \subseteq X \)
- Further: \( u_d \leq u_t \leq v_t \leq v_d \Rightarrow u_d \in X \)
- Since \( u \) was arbitrary:
  \[ |Y| + |N(Y)| \leq k. \]
- Since \( G \) is a bipartite \((n, d, \mu)\)-expander:
  \[ |Y| + |N(Y)| \leq k. \]
Existence of Approximate Halvers

Proof:
- $X :=$ wires with the $k$ smallest inputs
- $Y :=$ wires in lower half with $k$ smallest outputs
- For every $u \in N(Y)$: $\exists$ comparator $(u, v)$
- Let $u_t, v_t$ be their keys after the comparator
  Let $u_d, v_d$ be their keys at the output
- Note that $v_d \in Y \subseteq X$
- Further: $u_d \leq u_t \leq v_t \leq v_d \Rightarrow u_d \in X$
- Since $u$ was arbitrary:
  \[ |Y| + |N(Y)| \leq k. \]
- Since $G$ is a bipartite $(n, d, \mu)$-expander:
  \[ |Y| + |N(Y)| \geq |Y| + \min\{\mu|Y|, n/2 - |Y|\}. \]
Existence of Approximate Halvers

Proof:

- \( X \) := wires with the \( k \) smallest inputs
- \( Y \) := wires in lower half with \( k \) smallest outputs
- For every \( u \in N(Y) \): \( \exists \) comparator \( (u, v) \)
- Let \( u_t, v_t \) be their keys after the comparator
  - Let \( u_d, v_d \) be their keys at the output
  - Note that \( v_d \in Y \subseteq X \)
  - Further: \( u_d \leq u_t \leq v_t \leq v_d \Rightarrow u_d \in X \)
  - Since \( u \) was arbitrary:
    \[
    |Y| + |N(Y)| \leq k.
    \]

- Since \( G \) is a bipartite \((n, d, \mu)\)-expander:
  \[
  |Y| + |N(Y)| \geq |Y| + \min\{\mu|Y|, n/2 - |Y|\} = \min\{(1 + \mu)|Y|, n/2\}.
  \]
Existence of Approximate Halvers

Proof:

- \( X := \) wires with the \( k \) smallest inputs
- \( Y := \) wires in lower half with \( k \) smallest outputs
- For every \( u \in N(Y) \): \( \exists \) comparator \((u, v)\)
- Let \( u_t, v_t \) be their keys after the comparator
- Let \( u_d, v_d \) be their keys at the output
- Note that \( v_d \in Y \subseteq X \)
- Further: \( u_d \leq u_t \leq v_t \leq v_d \Rightarrow u_d \in X \)
- Since \( u \) was arbitrary:
  \[ |Y| + |N(Y)| \leq k. \]
- Since \( G \) is a bipartite \((n, d, \mu)\)-expander:
  \[ |Y| + |N(Y)| \geq |Y| + \min\{\mu|Y|, n/2 - |Y|\} \]
  \[ = \min\{(1 + \mu)|Y|, n/2\}. \]
- Combining the two bounds above yields:
  \[ (1 + \mu)|Y| \leq k. \]
Existence of Approximate Halvers

Proof:
- Let $X$ be the wires with the $k$ smallest inputs.
- Let $Y$ be the wires in lower half with $k$ smallest outputs.
- For every $u \in N(Y)$: there exists a comparator $(u, v)$.
- Let $u_t, v_t$ be their keys after the comparator.
- Let $u_d, v_d$ be their keys at the output.
- Note that $v_d \in Y \subseteq X$.
- Further: $u_d \leq u_t \leq v_t \leq v_d \Rightarrow u_d \in X$.
- Since $u$ was arbitrary:
  \[ |Y| + |N(Y)| \leq k. \]
- Since $G$ is a bipartite $(n, d, \mu)$-expander:
  \[ |Y| + |N(Y)| \geq |Y| + \min\{\mu|Y|, n/2 - |Y|\} \]
  \[ \geq \min\{1 + \mu|Y|, n/2\}. \]
- Combining the two bounds above yields:
  \[ (1 + \mu)|Y| \leq k. \]

Here we used that $k \leq n/2$. 

I. Sorting Networks

Batcher's Sorting Network
Existence of Approximate Halvers

Proof:

- $X :=$ wires with the $k$ smallest inputs
- $Y :=$ wires in lower half with $k$ smallest outputs
- For every $u \in N(Y)$: $\exists$ comparator $(u, v)$
- Let $u_t, v_t$ be their keys after the comparator
  Let $u_d, v_d$ be their keys at the output
- Note that $v_d \in Y \subseteq X$
- Further: $u_d \leq u_t \leq v_t \leq v_d \Rightarrow u_d \in X$
- Since $u$ was arbitrary:
  $$|Y| + |N(Y)| \leq k.$$
- Since $G$ is a bipartite $(n, d, \mu)$-expander:
  $$|Y| + |N(Y)| \geq |Y| + \min\{\mu |Y|, n/2 - |Y|\}$$
  $$= \min\{(1 + \mu) |Y|, n/2\}.$$
- Combining the two bounds above yields:
  $$(1 + \mu) |Y| \leq k.$$
- The same argument shows that at most $\epsilon \cdot k$, $\epsilon := 1/(\mu + 1)$, of the $k$ largest input keys are placed in $b_1, \ldots, b_{n/2}$. $$\square$$
Donald E. Knuth (Stanford)

“Batcher’s method is much better, unless $n$ exceeds the total memory capacity of all computers on earth!”

Richard J. Lipton (Georgia Tech)

“The AKS sorting network is galactic: it needs that $n$ be larger than $2^{78}$ or so to finally be smaller than Batcher’s network for $n$ items.”
Siblings of Sorting Network

- Sorting Networks
  - sorts any input of size $n$
  - special case of Comparison Networks
Siblings of Sorting Network

Sorting Networks
- sorts any input of size $n$
- special case of Comparison Networks

Switching (Shuffling) Networks
- creates a random permutation of $n$ items
- special case of Permutation Networks
Siblings of Sorting Network

- **Sorting Networks**
  - sorts any input of size $n$
  - special case of Comparison Networks

- **Switching (Shuffling) Networks**
  - creates a random permutation of $n$ items
  - special case of Permutation Networks

- **Counting Networks**
  - balances any stream of tokens over $n$ wires
  - special case of Balancing Networks
Outline

Outline of this Course

Introduction to Sorting Networks

Batcher's Sorting Network

Counting Networks
Counting Network

Distributed Counting

Processors collectively assign successive values from a given range.
Counting Network

Distributed Counting

Processors collectively assign successive values from a given range.

Values could represent addresses in memories or destinations on an interconnection network.
Counting Network

Distributed Counting

Processors collectively assign successive values from a given range.

Balancing Networks

- constructed in a similar manner like sorting networks
- instead of comparators, consists of balancers
- balancers are asynchronous flip-flops that forward tokens from its inputs to one of its two outputs alternately (top, bottom, top, ...)

I. Sorting Networks  Counting Networks  26
Counting Network

Distributed Counting

Processors collectively assign successive values from a given range.

Balancing Networks

- constructed in a similar manner like sorting networks
- instead of comparators, consists of balancers
- balancers are asynchronous flip-flops that forward tokens from its inputs to one of its two outputs alternately (top, bottom, top, ...)

---

I. Sorting Networks  Counting Networks  26
Counting Network

Distributed Counting

Processors collectively assign successive values from a given range.

Balancing Networks

- constructed in a similar manner like sorting networks
- instead of comparators, consists of balancers
- balancers are asynchronous flip-flops that forward tokens from its inputs to one of its two outputs alternately (top, bottom, top, ...)

![Diagram of a balancer network]
Counting Network

Distributed Counting

Processors collectively assign successive values from a given range.

Balancing Networks

- constructed in a similar manner like sorting networks
- instead of comparators, consists of balancers
- balancers are asynchronous flip-flops that forward tokens from its inputs to one of its two outputs alternately (top, bottom, top, ...)

I. Sorting Networks  Counting Networks  26
Counting Network

Distributed Counting

Processors collectively assign successive values from a given range.

Balancing Networks

- constructed in a similar manner like sorting networks
- instead of comparators, consists of balancers
- balancers are asynchronous flip-flops that forward tokens from its inputs to one of its two outputs alternately (top, bottom, top, ...)

![Diagram of balancing network](image)
Counting Network

Distributed Counting

Processors collectively assign successive values from a given range.

Balancing Networks

- constructed in a similar manner like sorting networks
- instead of comparators, consists of balancers
- balancers are asynchronous flip-flops that forward tokens from its inputs to one of its two outputs alternately (top, bottom, top, ...)

![Diagram of a balancing network]
Counting Network

Distributed Counting

Processors collectively assign successive values from a given range.

Balancing Networks

- constructed in a similar manner like sorting networks
- instead of comparators, consists of balancers
- balancers are asynchronous flip-flops that forward tokens from its inputs to one of its two outputs alternately (top, bottom, top, . . .)
Counting Network

Distributed Counting

Processors collectively assign successive values from a given range.

Balancing Networks

- constructed in a similar manner like sorting networks
- instead of comparators, consists of balancers
- balancers are asynchronous flip-flops that forward tokens from its inputs to one of its two outputs alternately (top, bottom, top, ...)

![Diagram of balancer network]
Counting Network

**Distributed Counting**

Processors collectively assign successive values from a given range.

**Balancing Networks**

- constructed in a similar manner like sorting networks
- instead of comparators, consists of balancers
- balancers are asynchronous flip-flops that forward tokens from its inputs to one of its two outputs alternately (top, bottom, top, ...)

I. Sorting Networks

Counting Networks 26
Counting Network

Distributed Counting
Processors collectively assign successive values from a given range.

Balancing Networks
- constructed in a similar manner like sorting networks
- instead of comparators, consists of balancers
- balancers are asynchronous flip-flops that forward tokens from its inputs to one of its two outputs alternately (top, bottom, top, ...)

I. Sorting Networks
Counting Networks
26
Counting Network

Distributed Counting

Processors collectively assign successive values from a given range.

Balancing Networks

- constructed in a similar manner like sorting networks
- instead of comparators, consists of balancers
- balancers are asynchronous flip-flops that forward tokens from its inputs to one of its two outputs alternately (top, bottom, top, ...)

![Balancing Network Diagram]
Counting Network

Distributed Counting

Processors collectively assign successive values from a given range.

Balancing Networks

- constructed in a similar manner like sorting networks
- instead of comparators, consists of balancers
- balancers are asynchronous flip-flops that forward tokens from its inputs to one of its two outputs alternately (top, bottom, top, ...)

Number of tokens differs by at most one
Bitonic Counting Network

Counting Network (Formal Definition)

1. Let $x_1, x_2, \ldots, x_n$ be the number of tokens (ever received) on the designated input wires
2. Let $y_1, y_2, \ldots, y_n$ be the number of tokens (ever received) on the designated output wires
Bitonic Counting Network

Counting Network (Formal Definition)

1. Let $x_1, x_2, \ldots, x_n$ be the number of tokens (ever received) on the designated input wires
2. Let $y_1, y_2, \ldots, y_n$ be the number of tokens (ever received) on the designated output wires
3. In a quiescent state: $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$
4. A counting network is a balancing network with the step-property:
   
   $0 \leq y_i - y_j \leq 1$ for any $i < j$.

(4, 4, 4, 3, 3, 3, 3, 3)
Bitonic Counting Network

Counting Network (Formal Definition)

1. Let $x_1, x_2, \ldots, x_n$ be the number of tokens (ever received) on the designated input wires
2. Let $y_1, y_2, \ldots, y_n$ be the number of tokens (ever received) on the designated output wires
3. In a quiescent state: $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$
4. A counting network is a balancing network with the step-property:

   $$0 \leq y_i - y_j \leq 1 \text{ for any } i < j.$$

Bitonic Counting Network: Take Batcher’s Sorting Network and replace each comparator by a balancer.
Correctness of the Bitonic Counting Network

**Facts**

Let $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ have the step property. Then:

1. We have $\sum_{i=1}^{n/2} x_{2i-1} = \lceil \frac{1}{2} \sum_{i=1}^{n} x_i \rceil$ and $\sum_{i=1}^{n/2} x_{2i} = \lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \rfloor$
2. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$, then $x_i = y_i$ for $i = 1, \ldots, n$.
3. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i + 1$, then $\exists! j = 1, 2, \ldots, n$ with $x_j = y_j + 1$ and $x_i = y_i$ for $j \neq i$. 

---

I. Sorting Networks
Counting Networks
Correctness of the Bitonic Counting Network

Facts

Let \( x_1, \ldots, x_n \) and \( y_1, \ldots, y_n \) have the step property. Then:

1. We have \( \sum_{i=1}^{n/2} x_{2i-1} = \lceil \frac{1}{2} \sum_{i=1}^{n} x_i \rceil \) and \( \sum_{i=1}^{n/2} x_{2i} = \lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \rfloor \)

2. If \( \sum_{i=1}^{n/2} x_{2i-1} = \sum_{i=1}^{n/2} x_{2i} \), then \( x_i = y_i \) for \( i = 1, \ldots, n \).

3. If \( \sum_{i=1}^{n/2} x_{2i-1} = \sum_{i=1}^{n/2} x_{2i} + 1 \), then \( \exists j = 1, 2, \ldots, n \) with \( x_j = y_j + 1 \) and \( x_i = y_i \) for \( j \neq i \).

Key Lemma

Consider a \textsc{Merger}[n]. Then if the inputs \( x_1, \ldots, x_{n/2} \) and \( x_{n/2+1}, \ldots, x_n \) have the step property, then so does the output \( y_1, \ldots, y_n \).

Proof (by induction on \( n \))
Correctness of the Bitonic Counting Network

\[ \text{Facts} \]

Let \( x_1, \ldots, x_n \) and \( y_1, \ldots, y_n \) have the step property. Then:

1. We have \( \sum_{i=1}^{n/2} x_{2i-1} = \left\lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \right\rfloor \), and \( \sum_{i=1}^{n/2} x_{2i} = \left\lceil \frac{1}{2} \sum_{i=1}^{n} x_i \right\rceil \).

2. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \), then \( x_i = y_i \) for \( i = 1, \ldots, n \).

3. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i + 1 \), then \( \exists! j = 1, 2, \ldots, n \) with \( x_j = y_j + 1 \) and \( x_i = y_i \) for \( j \neq i \).

Proof (by induction on \( n \))
Correctness of the Bitonic Counting Network

Facts

Let \( x_1, \ldots, x_n \) and \( y_1, \ldots, y_n \) have the step property. Then:

1. We have \( \sum_{i=1}^{n/2} x_{2i-1} = \left\lceil \frac{1}{2} \sum_{i=1}^{n} x_i \right\rceil \), and \( \sum_{i=1}^{n/2} x_{2i} = \left\lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \right\rfloor \)
2. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \), then \( x_i = y_i \) for \( i = 1, \ldots, n \).
3. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i + 1 \), then there exists \( j = 1, 2, \ldots, n \) with \( x_j = y_j + 1 \) and \( x_i = y_i \) for \( j \neq i \).

Proof (by induction on \( n \))

- Case \( n = 2 \) is clear, since \textsc{Merger}[2] is a single balancer
Correctness of the Bitonic Counting Network

Let $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ have the step property. Then:

1. We have $\sum_{i=1}^{n/2} x_{2i-1} = \left\lceil \frac{1}{2} \sum_{i=1}^{n} x_i \right\rceil$, and $\sum_{i=1}^{n/2} x_{2i} = \left\lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \right\rfloor$.
2. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$, then $x_i = y_i$ for $i = 1, \ldots, n$.
3. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i + 1$, then $\exists! j = 1, 2, \ldots, n$ with $x_j = y_j + 1$ and $x_i = y_i$ for $j \neq i$.

Proof (by induction on $n$)

- Case $n = 2$ is clear, since MERGER[2] is a single balancer
- $n > 2$: 

![Diagram of a bitonic counting network]
Correctness of the Bitonic Counting Network

Let \( x_1, \ldots, x_n \) and \( y_1, \ldots, y_n \) have the step property. Then:

1. We have \( \sum_{i=1}^{n/2} x_{2i-1} = \lceil \frac{1}{2} \sum_{i=1}^{n} x_i \rceil \), and \( \sum_{i=1}^{n/2} x_{2i} = \lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \rfloor \).
2. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \), then \( x_i = y_i \) for \( i = 1, \ldots, n \).
3. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i + 1 \), then \( \exists! j = 1, 2, \ldots, n \) with \( x_j = y_j + 1 \) and \( x_i = y_i \) for \( j \neq i \).

Proof (by induction on \( n \))

- Case \( n = 2 \) is clear, since \( \text{MERGER}[2] \) is a single balancer
- \( n > 2 \): Let \( z_1, \ldots, z_{n/2} \) and \( z'_1, \ldots, z'_{n/2} \) be the outputs of the \( \text{MERGER}[n/2] \) subnetworks
Correctness of the Bitonic Counting Network

Let $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ have the step property. Then:

1. We have $\sum_{i=1}^{n/2} x_{2i-1} = \lceil \frac{1}{2} \sum_{i=1}^{n} x_i \rceil$, and $\sum_{i=1}^{n/2} x_{2i} = \lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \rfloor$.

2. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$, then $x_i = y_i$ for $i = 1, \ldots, n$.

3. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i + 1$, then $\exists j = 1, 2, \ldots, n$ with $x_j = y_j + 1$ and $x_i = y_i$ for $j \neq i$.

Proof (by induction on $n$)

- Case $n = 2$ is clear, since $\text{MERGER}[2]$ is a single balancer.
- $n > 2$: Let $z_1, \ldots, z_{n/2}$ and $z'_1, \ldots, z'_{n/2}$ be the outputs of the $\text{MERGER}[n/2]$ subnetworks.
Correctness of the Bitonic Counting Network

Let $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ have the step property. Then:

1. We have $\sum_{i=1}^{n/2} x_{2i-1} = \lceil \frac{1}{2} \sum_{i=1}^{n} x_i \rceil$, and $\sum_{i=1}^{n/2} x_{2i} = \lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \rfloor$.
2. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$, then $x_i = y_i$ for $i = 1, \ldots, n$.
3. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i + 1$, then $\exists! j = 1, 2, \ldots, n$ with $x_j = y_j + 1$ and $x_i = y_i$ for $j \neq i$.

Proof (by induction on $n$)

- Case $n = 2$ is clear, since MERGER[2] is a single balancer
- $n > 2$: Let $z_1, \ldots, z_{n/2}$ and $z'_1, \ldots, z'_{n/2}$ be the outputs of the MERGER[n/2] subnetworks
**Correctness of the Bitonic Counting Network**

Let \( x_1, \ldots, x_n \) and \( y_1, \ldots, y_n \) have the step property. Then:

1. We have \( \sum_{i=1}^{n/2} x_{2i-1} = \lceil \frac{1}{2} \sum_{i=1}^{n} x_i \rceil \) and \( \sum_{i=1}^{n/2} x_{2i} = \lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \rfloor \).

2. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \), then \( x_i = y_i \) for \( i = 1, \ldots, n \).

3. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i + 1 \), then \( \exists! j = 1, 2, \ldots, n \) with \( x_j = y_j + 1 \) and \( x_i = y_i \) for \( j \neq i \).

**Proof (by induction on \( n \))**

- Case \( n = 2 \) is clear, since MERGER[2] is a single balancer.
- \( n > 2 \): Let \( z_1, \ldots, z_{n/2} \) and \( z'_1, \ldots, z'_{n/2} \) be the outputs of the MERGER[n/2] subnetworks.
Let \( x_1, \ldots, x_n \) and \( y_1, \ldots, y_n \) have the step property. Then:

1. We have
   \[
   \sum_{i=1}^{n/2} x_{2i-1} = \left\lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \right\rfloor, \quad \text{and} \quad \sum_{i=1}^{n} x_{2i} = \left\lceil \frac{1}{2} \sum_{i=1}^{n} x_i \right\rceil
   \]

2. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \), then \( x_i = y_i \) for \( i = 1, \ldots, n \).

3. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i + 1 \), then there exists \( j = 1, 2, \ldots, n \) with \( x_j = y_j + 1 \) and \( x_i = y_i \) for \( i \neq j \).

Proof (by induction on \( n \))

- Case \( n = 2 \) is clear, since MERGER[2] is a single balancer
- \( n > 2 \): Let \( z_{1, \ldots, n/2} \) and \( z'_{1, \ldots, n/2} \) be the outputs of the MERGER[n/2] subnetworks
Correctness of the Bitonic Counting Network

Facts

Let $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ have the step property. Then:

1. We have $\sum_{i=1}^{n/2} x_{2i-1} = \left\lceil \frac{1}{2} \sum_{i=1}^{n} x_i \right\rceil$, and $\sum_{i=1}^{n/2} x_{2i} = \left\lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \right\rfloor$
2. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$, then $x_i = y_i$ for $i = 1, \ldots, n$.
3. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i + 1$, then $\exists! j = 1, 2, \ldots, n$ with $x_j = y_j + 1$ and $x_i = y_i$ for $j \neq i$.

Proof (by induction on $n$)

- Case $n = 2$ is clear, since $\text{MERGER}[2]$ is a single balancer
- $n > 2$: Let $z_1, \ldots, z_{n/2}$ and $z'_1, \ldots, z'_{n/2}$ be the outputs of the $\text{MERGER}[n/2]$ subnetworks
- $\text{IH} \Rightarrow z_1, \ldots, z_{n/2}$ and $z'_1, \ldots, z'_{n/2}$ have the step property
Correctness of the Bitonic Counting Network

Let \( x_1, \ldots, x_n \) and \( y_1, \ldots, y_n \) have the step property. Then:

1. We have \( \sum_{i=1}^{n/2} x_{2i-1} = \left\lceil \frac{1}{2} \sum_{i=1}^{n} x_i \right\rceil \) and \( \sum_{i=1}^{n/2} x_{2i} = \left\lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \right\rfloor \)
2. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \), then \( x_i = y_i \) for \( i = 1, \ldots, n \).
3. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i + 1 \), then \( \exists! j = 1, 2, \ldots, n \) with \( x_j = y_j + 1 \) and \( x_i = y_i \) for \( j \neq i \).

Proof (by induction on \( n \))

- **Case** \( n = 2 \) is clear, since MERGER[2] is a single balancer
- **\( n > 2 \):** Let \( z_1, \ldots, z_{n/2} \) and \( z'_1, \ldots, z'_{n/2} \) be the outputs of the MERGER[\( n/2 \)] subnetworks
- \( \text{IH} \Rightarrow z_1, \ldots, z_{n/2} \) and \( z'_1, \ldots, z'_{n/2} \) have the step property
- Let \( Z := \sum_{i=1}^{n/2} z_i \) and \( Z' := \sum_{i=1}^{n/2} z'_i \)
Correctness of the Bitonic Counting Network

Let \( x_1, \ldots, x_n \) and \( y_1, \ldots, y_n \) have the step property. Then:

1. We have \( \sum_{i=1}^{n/2} x_{2i-1} = \left\lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \right\rfloor \) and \( \sum_{i=1}^{n/2} x_{2i} = \left\lceil \frac{1}{2} \sum_{i=1}^{n} x_i \right\rceil \)
2. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \), then \( x_i = y_i \) for \( i = 1, \ldots, n \).
3. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i + 1 \), then \( \exists ! j = 1, 2, \ldots, n \) with \( x_j = y_j + 1 \) and \( x_i = y_i \) for \( i \neq j \).

Proof (by induction on \( n \))

- Case \( n = 2 \) is clear, since MERGER[2] is a single balancer
- \( n > 2 \): Let \( z_1, \ldots, z_{n/2} \) and \( z'_1, \ldots, z'_{n/2} \) be the outputs of the MERGER[\( n/2 \)] subnetworks
- IH \( \Rightarrow z_1, \ldots, z_{n/2} \) and \( z'_1, \ldots, z'_{n/2} \) have the step property
- Let \( Z := \sum_{i=1}^{n/2} z_i \) and \( Z' := \sum_{i=1}^{n/2} z'_i \)
- F1 \( \Rightarrow Z = \left\lfloor \frac{1}{2} \sum_{i=1}^{n/2} x_i \right\rfloor + \left\lfloor \frac{1}{2} \sum_{i=n/2+1}^{n} x_i \right\rfloor \) and \( Z' = \left\lceil \frac{1}{2} \sum_{i=1}^{n/2} x_i \right\rceil + \left\lceil \frac{1}{2} \sum_{i=n/2+1}^{n} x_i \right\rceil \)
Correctness of the Bitonic Counting Network

Let $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ have the step property. Then:

1. We have $\sum_{i=1}^{n/2} x_{2i-1} = \left\lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \right\rfloor$, and $\sum_{i=1}^{n/2} x_{2i} = \left\lceil \frac{1}{2} \sum_{i=1}^{n} x_i \right\rceil$

2. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$, then $x_i = y_i$ for $i = 1, \ldots, n$.

3. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i + 1$, then $\exists! j = 1, 2, \ldots, n$ with $x_j = y_j + 1$ and $x_i = y_i$ for $j \neq i$.

Proof (by induction on $n$)

- Case $n = 2$ is clear, since MERGER[2] is a single balancer
- $n > 2$: Let $z_1, \ldots, z_{n/2}$ and $z'_1, \ldots, z'_{n/2}$ be the outputs of the MERGER[$n/2$] subnetworks
- IH $\Rightarrow z_1, \ldots, z_{n/2}$ and $z'_1, \ldots, z'_{n/2}$ have the step property
- Let $Z := \sum_{i=1}^{n/2} z_i$ and $Z' := \sum_{i=1}^{n/2} z'_i$
- F1 $\Rightarrow Z = \left\lfloor \frac{1}{2} \sum_{i=1}^{n/2} x_i \right\rfloor + \left\lfloor \frac{1}{2} \sum_{i=n/2+1}^{n} x_i \right\rfloor$ and $Z' = \left\lceil \frac{1}{2} \sum_{i=1}^{n/2} x_i \right\rceil + \left\lceil \frac{1}{2} \sum_{i=n/2+1}^{n} x_i \right\rceil$
Correctness of the Bitonic Counting Network

Let $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ have the step property. Then:

1. We have $\sum_{i=1}^{n/2} x_{2i-1} = \left\lceil \frac{1}{2} \sum_{i=1}^{n} x_i \right\rceil$, and $\sum_{i=1}^{n/2} x_{2i} = \left\lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \right\rfloor$
2. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$, then $x_i = y_i$ for $i = 1, \ldots, n$.
3. If $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i + 1$, then $\exists! j = 1, 2, \ldots, n$ with $x_j = y_j + 1$ and $x_i = y_i$ for $j \neq i$.

Proof (by induction on $n$)

- Case $n = 2$ is clear, since MERGER[2] is a single balancer
- $n > 2$: Let $z_1, \ldots, z_{n/2}$ and $z'_1, \ldots, z'_{n/2}$ be the outputs of the MERGER[$n/2$] subnetworks
- IH $\Rightarrow z_1, \ldots, z_{n/2}$ and $z'_1, \ldots, z'_{n/2}$ have the step property
- Let $Z := \sum_{i=1}^{n/2} z_i$ and $Z' := \sum_{i=1}^{n/2} z'_i$
- F1 $\Rightarrow Z = \left\lceil \frac{1}{2} \sum_{i=1}^{n/2} x_i \right\rceil + \left\lfloor \frac{1}{2} \sum_{i=n/2+1}^{n} x_i \right\rfloor$ and $Z' = \left\lceil \frac{1}{2} \sum_{i=1}^{n/2} x_i \right\rceil + \left\lfloor \frac{1}{2} \sum_{i=n/2+1}^{n} x_i \right\rfloor$
- Case 1: If $Z = Z'$, then F2 implies the output of MERGER[$n$] is $y_i = z_{1+[(i-1)/2]}$
Correctness of the Bitonic Counting Network

Let \( x_1, \ldots, x_n \) and \( y_1, \ldots, y_n \) have the step property. Then:

1. We have \( \sum_{i=1}^{n/2} x_{2i-1} = \left\lceil \frac{1}{2} \sum_{i=1}^{n} x_i \right\rceil \) and \( \sum_{i=1}^{n/2} x_{2i} = \left\lfloor \frac{1}{2} \sum_{i=1}^{n} x_i \right\rfloor \)
2. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i \), then \( x_i = y_i \) for \( i = 1, \ldots, n \).
3. If \( \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i + 1 \), then \( \exists ! j = 1, 2, \ldots, n \) with \( x_j = y_j + 1 \) and \( x_i = y_i \) for \( j \neq i \).

Proof (by induction on \( n \))

- Case \( n = 2 \) is clear, since MERGER[2] is a single balancer.
- \( n > 2 \): Let \( z_1, \ldots, z_{n/2} \) and \( z'_1, \ldots, z'_{n/2} \) be the outputs of the MERGER\([n/2]\) subnetworks.

IH \( \Rightarrow \) \( z_1, \ldots, z_{n/2} \) and \( z'_1, \ldots, z'_{n/2} \) have the step property.

- Let \( Z := \sum_{i=1}^{n/2} z_i \) and \( Z' := \sum_{i=1}^{n/2} z'_i \).
- F1 \( \Rightarrow \) \( Z = \left\lceil \frac{1}{2} \sum_{i=1}^{n/2} x_i \right\rceil + \left\lfloor \frac{1}{2} \sum_{i=n/2+1}^{n} x_i \right\rfloor \) and \( Z' = \left\lfloor \frac{1}{2} \sum_{i=1}^{n/2} x_i \right\rfloor + \left\lceil \frac{1}{2} \sum_{i=n/2+1}^{n} x_i \right\rceil \).
- Case 1: If \( Z = Z' \), then F2 implies the output of MERGER\([n]\) is \( y_i = z_{1+\left\lfloor (i-1)/2 \right\rfloor} \).
- Case 2: If \( |Z - Z'| = 1 \), F3 implies \( z_i = z'_i \) for \( i = 1, \ldots, n/2 \) except a unique \( j \) with \( z_j \neq z'_j \). Balancer between \( z_j \) and \( z'_j \) will ensure that the step property holds.
Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2n$, ...
Counting can be done as follows:
Add local counter to each output wire $i$, to assign consecutive numbers $i, i+n, i+2n, ...$.
Counting can be done as follows:

Add local counter to each output wire \(i\), to assign consecutive numbers \(i, i+n, i+2n, \ldots\).
Counting can be done as follows:
Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2n$, ...
Counting can be done as follows: Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2n$, ...
Counting can be done as follows:
Add local counter to each output wire $y_i$, to assign consecutive numbers $i$, $i+n$, $i+2n$, ...
Bitonic Counting Network in Action

Counting can be done as follows:
Add local counter to each output wire i, to assign consecutive numbers i, i+n, i+2·n, ...
Counting can be done as follows: Add local counter to each output wire i, to assign consecutive numbers i, i+n, i+2n,...
Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2\cdot n$, ...
Bitonic Counting Network in Action

Counting can be done as follows: Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2n$, ...
Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i, i+n, i+2n,...$.
Counting can be done as follows: Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2n$, ...
Counting can be done as follows:
Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i + n$, $i + 2 \cdot n$, ...
Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2n$, ...
Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i, i+n, i+2n, ...$.
Counting can be done as follows: Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2\cdot n$, ...
Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2n$, ...
Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i, i+n, i+2n, ...$
Counting can be done as follows:
Add local counter to each output wire $i$, to assign consecutive numbers $i, i+n, i+2n, ...$.
Counting can be done as follows: Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2n$, ...
Counting can be done as follows:

Add local counter to each output wire, to assign consecutive numbers $i, i+n, i+2n, ...$.
Bitonic Counting Network in Action

Counting can be done as follows:
Add local counter to each output wire $i$, to assign consecutive numbers $i, i+n, i+2n, \ldots$
Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i, i+n, i+2n, ...$
Bitonic Counting Network in Action

Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i, i + n, i + 2n, ...$
Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i, i+n, i+2n, ...$
Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2n$, ...
Counting can be done as follows:

Add local counter to each output wire \( i \), to assign consecutive numbers \( i, i+n, i+2n, \ldots \).
Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2n$, ...
Counting can be done as follows:

Add local counter to each output wire \(i\), to assign consecutive numbers \(i, i+n, i+2n, \ldots\).
Counting can be done as follows:
Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i + n$, $i + 2 \cdot n$, ...
Bitonic Counting Network in Action

Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i, i+n, i+2n, ...$.
Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2n$, ...
Bitonic Counting Network in Action

Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2n$, ...
Bitonic Counting Network in Action

Counting can be done as follows: Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2n$, ...
Counting can be done as follows:

Add local counter to each output wire $i$ to assign consecutive numbers $i, i+n, i+2n, ...
Counting can be done as follows:

Add local counter to each output wire $i$, to assign consecutive numbers $i$, $i+n$, $i+2n$, ...
Bitonic Counting Network in Action

Counting can be done as follows:

Add local counter to each output wire \( i \), to assign consecutive numbers \( i, i+n, i+2n, \ldots \)
Bitonic Counting Network in Action

Counting can be done as follows:
Add local counter to each output wire \( i \), to assign consecutive numbers \( i, i+n, i+2\cdot n, ... \).
Counting can be done as follows: Add local counter to each output wire $i$, to assign consecutive numbers $i, i+n, i+2n, ...$.
Counting can be done as follows:
Add **local counter** to each output wire $i$, to assign consecutive numbers $i, i + n, i + 2 \cdot n, \ldots$
A Periodic Counting Network [Aspnes, Herlihy, Shavit, JACM 1994]

Consists of $\log_2 n$ \textsc{lock} networks each of which has depth $\log_2 n$. 
A Periodic Counting Network [Aspnes, Herlihy, Shavit, JACM 1994]

Consists of \( \log n \) \text{BLOCK}[n] networks each of which has depth \( \log n \)
From Counting to Sorting

Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.
From Counting to Sorting

Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.

The converse is not true!
Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.

Proof.
Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.

Proof.

- Let $C$ be a counting network, and $S$ be the corresponding sorting network.
From Counting to Sorting

Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.

Proof.

- Let $C$ be a counting network, and $S$ be the corresponding sorting network.
From Counting to Sorting

Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.

Proof.

- Let $C$ be a counting network, and $S$ be the corresponding sorting network.
- Consider an input sequence $a_1, a_2, \ldots, a_n \in \{0, 1\}^n$ to $S$.
From Counting to Sorting

Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.

Proof.

- Let $C$ be a counting network, and $S$ be the corresponding sorting network
- Consider an input sequence $a_1, a_2, \ldots, a_n \in \{0, 1\}^n$ to $S$
From Counting to Sorting

Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.

Proof.

- Let $C$ be a counting network, and $S$ be the corresponding sorting network.
- Consider an input sequence $a_1, a_2, \ldots, a_n \in \{0, 1\}^n$ to $S$.
- Define an input $x_1, x_2, \ldots, x_n \in \{0, 1\}^n$ to $C$ by $x_i = 1$ iff $a_i = 0$.

**Diagram:**

- A counting network $C$ with inputs and outputs.
- A sorting network $S$ with inputs and outputs, demonstrating the sorting of the input sequence.
From Counting to Sorting

Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.

Proof.

- Let $C$ be a counting network, and $S$ be the corresponding sorting network.
- Consider an input sequence $a_1, a_2, \ldots, a_n \in \{0, 1\}^n$ to $S$.
- Define an input $x_1, x_2, \ldots, x_n \in \{0, 1\}^n$ to $C$ by $x_i = 1$ iff $a_i = 0$.
- $C$ is a counting network $\Rightarrow$ all ones will be routed to the lower wires.

\[ \begin{array}{c|c|c|c} 
& 0 & 1 & 1 \\
0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 \\
\end{array} \]

\[ \begin{array}{c|c|c|c} 
& 0 & 1 & 1 \\
0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 \\
\end{array} \]
From Counting to Sorting

Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.

Proof.

- Let $C$ be a counting network, and $S$ be the corresponding sorting network.
- Consider an input sequence $a_1, a_2, \ldots, a_n \in \{0, 1\}^n$ to $S$.
- Define an input $x_1, x_2, \ldots, x_n \in \{0, 1\}^n$ to $C$ by $x_i = 1$ iff $a_i = 0$.
- $C$ is a counting network $\Rightarrow$ all ones will be routed to the lower wires.

![Diagram showing the correspondence between $C$ and $S$]

I. Sorting Networks  
Counting Networks  
31
From Counting to Sorting

Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.

Proof.

- Let $C$ be a counting network, and $S$ be the corresponding sorting network
- Consider an input sequence $a_1, a_2, \ldots, a_n \in \{0, 1\}^n$ to $S$
- Define an input $x_1, x_2, \ldots, x_n \in \{0, 1\}^n$ to $C$ by $x_i = 1$ iff $a_i = 0$.
- $C$ is a counting network $\Rightarrow$ all ones will be routed to the lower wires

![Diagram showing the counting network $C$ and the corresponding sorting network $S$]
From Counting to Sorting

Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.

Proof.

- Let $C$ be a counting network, and $S$ be the corresponding sorting network.
- Consider an input sequence $a_1, a_2, \ldots, a_n \in \{0, 1\}^n$ to $S$.
- Define an input $x_1, x_2, \ldots, x_n \in \{0, 1\}^n$ to $C$ by $x_i = 1$ iff $a_i = 0$.
- $C$ is a counting network $\Rightarrow$ all ones will be routed to the lower wires.

![Diagram](image)

I. Sorting Networks

Counting Networks
From Counting to Sorting

Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.

Proof.

- Let $C$ be a counting network, and $S$ be the corresponding sorting network.
- Consider an input sequence $a_1, a_2, \ldots, a_n \in \{0, 1\}^n$ to $S$.
- Define an input $x_1, x_2, \ldots, x_n \in \{0, 1\}^n$ to $C$ by $x_i = 1$ iff $a_i = 0$.
- $C$ is a counting network $\Rightarrow$ all ones will be routed to the lower wires.

\[\begin{array}{cccccc}
0 & 1 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 & 1 \\
1 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
\end{array}\]

\[\begin{array}{cccccc}
1 & 0 & 0 & 1 & 1 \\
1 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
1 & \min & \max & \min & \max \\
\end{array}\]

$C$ $S$
From Counting to Sorting

Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.

Proof.

- Let $C$ be a counting network, and $S$ be the corresponding sorting network.
- Consider an input sequence $a_1, a_2, \ldots, a_n \in \{0, 1\}^n$ to $S$.
- Define an input $x_1, x_2, \ldots, x_n \in \{0, 1\}^n$ to $C$ by $x_i = 1$ iff $a_i = 0$.
- $C$ is a counting network $\Rightarrow$ all ones will be routed to the lower wires.
- $S$ corresponds to $C \Rightarrow$ all zeros will be routed to the lower wires.
From Counting to Sorting

Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.

Proof.

- Let $C$ be a counting network, and $S$ be the corresponding sorting network.
- Consider an input sequence $a_1, a_2, \ldots, a_n \in \{0, 1\}^n$ to $S$.
- Define an input $x_1, x_2, \ldots, x_n \in \{0, 1\}^n$ to $C$ by $x_i = 1$ iff $a_i = 0$.
- $C$ is a counting network $\Rightarrow$ all ones will be routed to the lower wires.
- $S$ corresponds to $C$ $\Rightarrow$ all zeros will be routed to the lower wires.
From Counting to Sorting

Counting vs. Sorting

If a network is a counting network, then it is also a sorting network.

Proof.

- Let $C$ be a counting network, and $S$ be the corresponding sorting network.
- Consider an input sequence $a_1, a_2, \ldots, a_n \in \{0, 1\}^n$ to $S$.
- Define an input $x_1, x_2, \ldots, x_n \in \{0, 1\}^n$ to $C$ by $x_i = 1$ iff $a_i = 0$.
- $C$ is a counting network $\Rightarrow$ all ones will be routed to the lower wires.
- $S$ corresponds to $C$ $\Rightarrow$ all zeros will be routed to the lower wires.
- By the Zero-One Principle, $S$ is a sorting network.

The converse is not true!