Counting-Based Impossibility Proofs for Renaming and Set Agreement

Hagit Attiya and Ami Paz

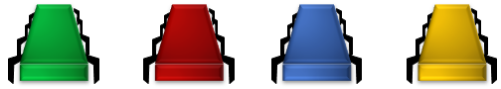
Technion - Israel Institute of Technology

DISC 2012

This Work

Wait-free algorithms for n processes, using only reads and writes

I.e., all but one process may crash

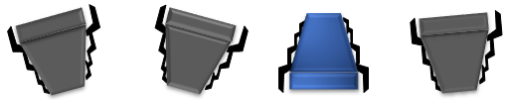


New proofs for the impossibility of:

- $(n-1)$-set agreement
- $(2p - \left\lceil \frac{p}{n-1} \right\rceil)$-adaptive renaming
- $(2n-2)$-renaming

This Work

Wait-free algorithms for n processes, using only reads and writes
I.e., all but one process may crash



New proofs for the impossibility of:

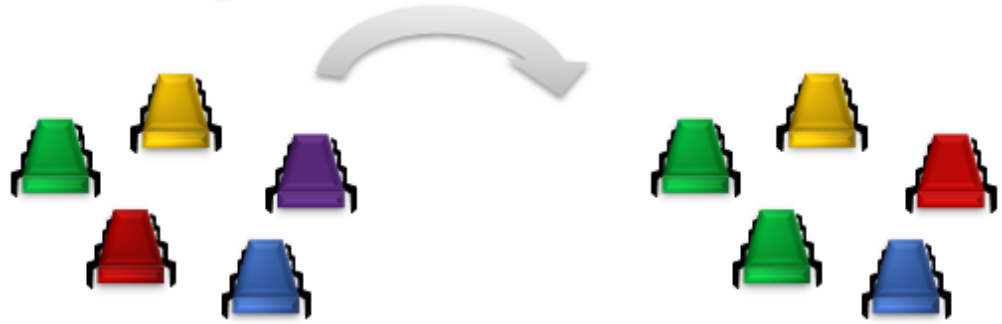
- $(n-1)$-set agreement
- $\left(2p - \left\lceil \frac{p}{n-1} \right\rceil\right)$-adaptive renaming
- $(2n-2)$-renaming

This Work

Wait-free algorithms for n processes, using only reads and writes
I.e., all but one process may crash

New proofs for the impossibility of:

- $(n-1)$-set agreement
- $(2p - \left\lceil \frac{p}{n-1} \right\rceil)$-adaptive renaming
- $(2n-2)$-renaming



This Work

Wait-free algorithms for n processes, using only reads and writes
I.e., all but one process may crash

New proofs for the impossibility of:

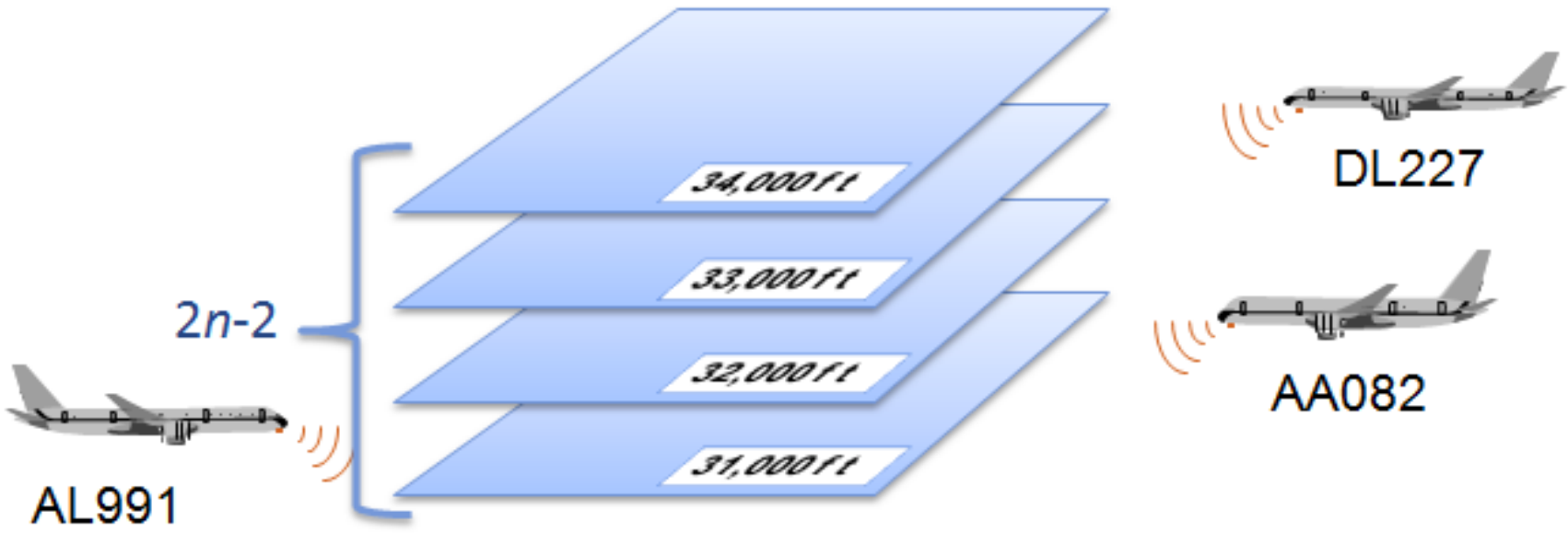
- $(n-1)$-set agreement
- $(2p - \left\lceil \frac{p}{n-1} \right\rceil)$-adaptive renaming
- $(2n-2)$-renaming

Renaming [Attiya et al.]

Processes start with unique inputs; must decide on unique new names from a smaller range

Renaming [Attiya et al.]

Processes start with unique inputs; must decide on unique new names from a smaller range



Renaming [Attiya et al.]

Processes start with unique inputs; must decide on unique new names from a smaller range

- $2n-1$ names are sufficient [Attiya et al., Borowsky and Gafni, Attiya and Fouren, Gafni and Rajsbaum]
- $n+1$ names are necessary [Attiya et al.]
- $2n-1$ names are necessary [Herlihy and Shavit, Herlihy and Rajsbaum, Attiya and Rajsbaum]
- $2n-1$ names are necessary for some values of n, but not for others [Castañeda and Rajsbaum]

Theorem: $(2n-2)$-renaming solvable $\Leftrightarrow \binom{n}{1}, \ldots, \binom{n}{n-1}$ relatively prime $\Leftrightarrow n$ is not a prime power

This talk: n is a prime power $\Rightarrow (2n-2)$-renaming is not solvable

Renaming [Attiya et al.]

Processes start with unique inputs; must decide on unique new names from a smaller range

- $2n-1$ names are sufficient [Attiya et al., Borowsky and Gafni, Attiya and Fouren, Gafni and Rajsbaum]
- $n+1$ names are necessary [Attiya et al.]
- $2n-1$ names are necessary [Herlihy and Shavit, Herlihy and Rajsbaum, Attiya and Rajsbaum]

- $2n-1$ names are necessary for some values of n, but not for others [Castañeda and Rajsbaum]

Theorem: $(2n-2)$-renaming solvable $\Leftrightarrow \binom{n}{1}, \ldots, \binom{n}{n-1}$ relatively prime $\Leftrightarrow n$ is not a prime power

This talk: n is a prime power $\Rightarrow$ $(2n-2)$-renaming is not solvable

Renaming [Attiya et al.]

Processes start with unique inputs; must decide on unique new names from a smaller range

- $2n-1$ names are sufficient [Attiya et al., Borowsky and Gafni, Attiya and Fouren, Gafni and Rajsbaum]
- $n+1$ names are necessary [Attiya et al.]
- $2n-1$ names are necessary [Herlihy and Shavit, Herlihy and Rajsbaum, Attiya and Rajsbaum]

- $2n-1$ names are necessary for some values of n, but not for others [Castañeda and Rajsbaum]

Theorem: $(2n-2)$-renaming solvable $\Leftrightarrow \binom{n}{1}, \ldots, \binom{n}{n-1}$ relatively prime $\Leftrightarrow$ n is not a prime power

This talk: n is a prime power $\Rightarrow$ $(2n-2)$-renaming is not solvable

Renaming [Attiya et al.]

Processes start with unique inputs; must decide on unique new names from a smaller range

- $2n-1$ names are sufficient [Attiya et al., Borowsky and Gafni, Attiya and Fouren, Gafni and Rajsbaum]
- $n+1$ names are necessary [Attiya et al.]
- $2n-1$ names are necessary [Herlihy and Shavit, Herlihy and Rajsbaum, Attiya and Rajsbaum]

- $2n-1$ names are necessary for some values of n, but not for others [Castañeda and Rajsbaum]

Theorem: $(2n-2)$-renaming solvable $\Leftrightarrow$ $\binom{n}{1}, \ldots, \binom{n}{n-1}$ relatively prime $\Leftrightarrow$ n is not a prime power

This talk: n is a prime power $\Rightarrow$ $(2n-2)$-renaming is not solvable

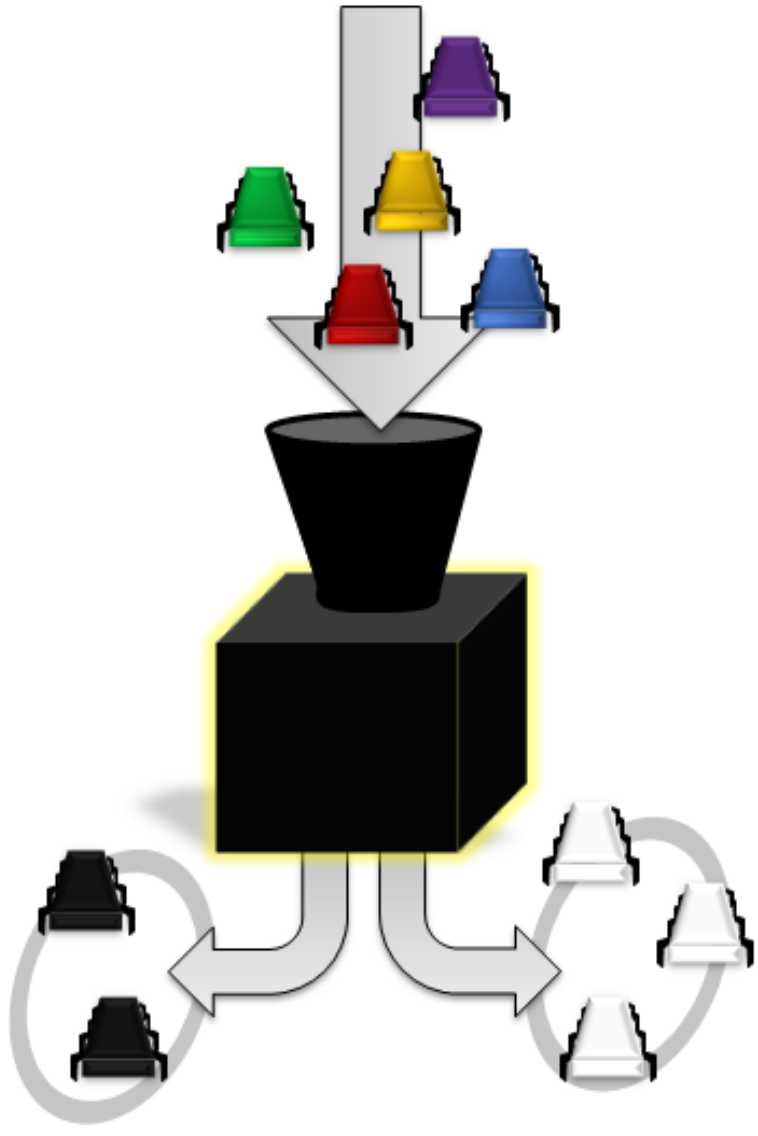
Weak Symmetry Breaking (WSB)

Processes start with unique input values
Must decide on a single bit s.t.:

- If all terminate,
  both 0 and 1 are decided

If $(2n-2)$-renaming is solvable,
then so is WSB

Suffices to show that WSB is unsolvable



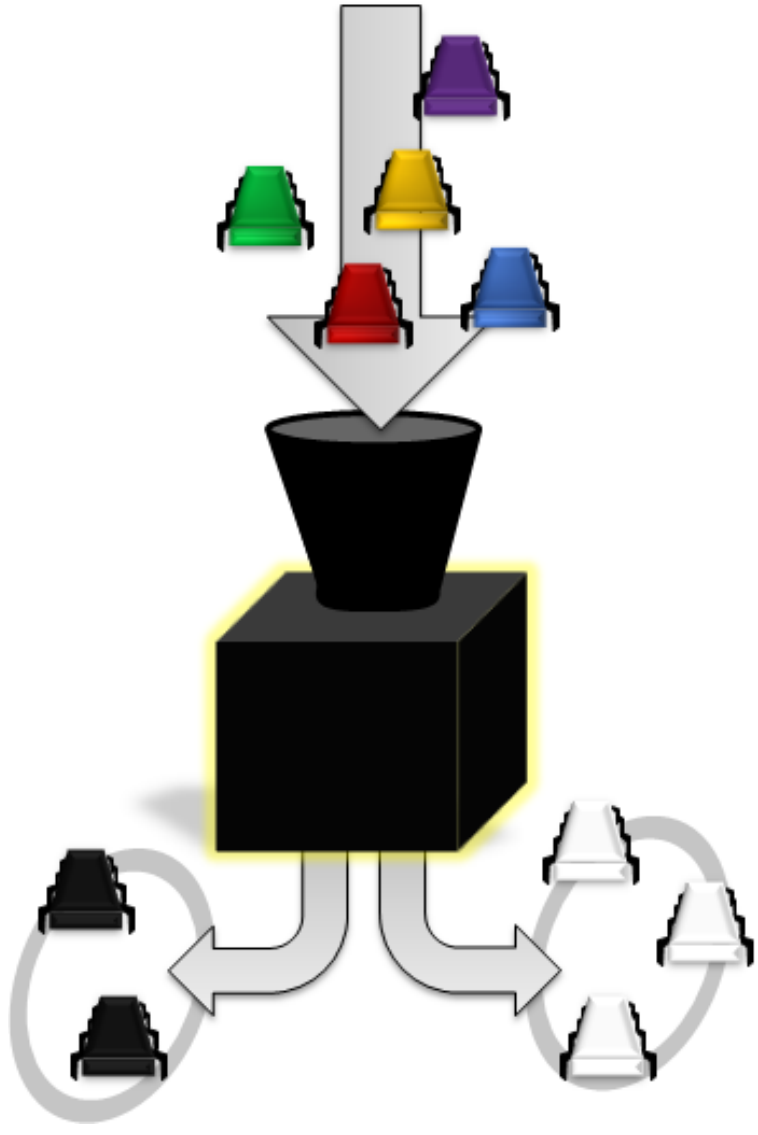
Weak Symmetry Breaking (WSB)

Processes start with unique input values
Must decide on a single bit s.t.:

- If all terminate,
  both 0 and 1 are decided

If $(2n-2)$-renaming is solvable,
then so is WSB

Suffices to show that WSB is unsolvable



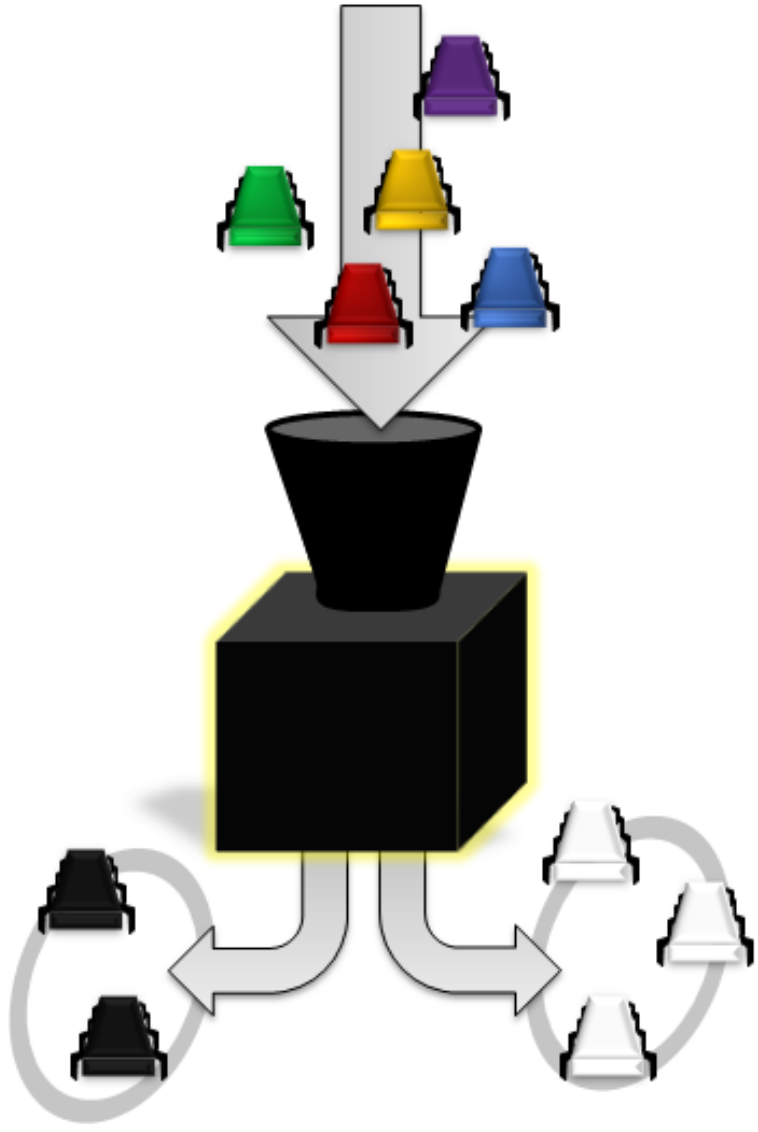
Weak Symmetry Breaking (WSB)

Processes start with unique input values
Must decide on a single bit s.t.:

- If all terminate,
  both 0 and 1 are decided

If $(2n-2)$-renaming is solvable,
then so is WSB

Suffices to show that WSB is unsolvable



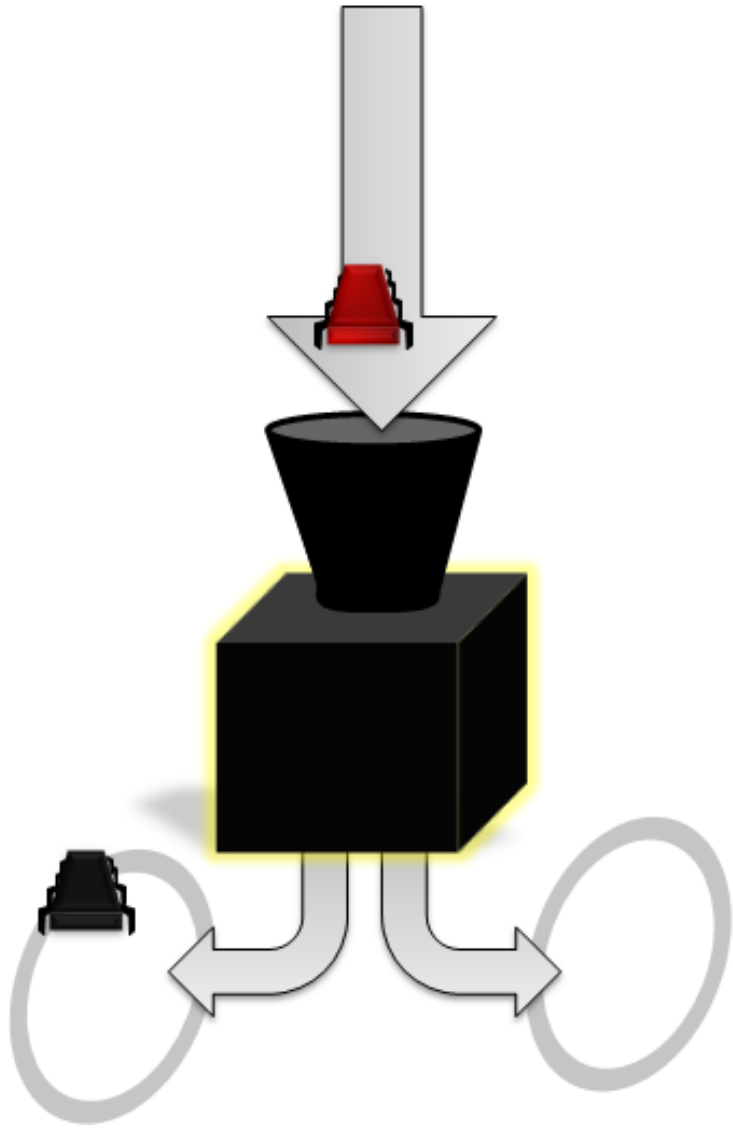
Strong Symmetry Breaking (SSB)

Processes start with unique input values
Must decide on a single bit s.t.:

- If all terminate,
  both 0 and 1 are decided
- In any execution, 1 is decided

If $(2p - \left\lfloor \frac{p}{n-1} \right\rfloor)$-adaptive renaming is solvable, then so is SSB

Suffices to show that SSB is unsolvable



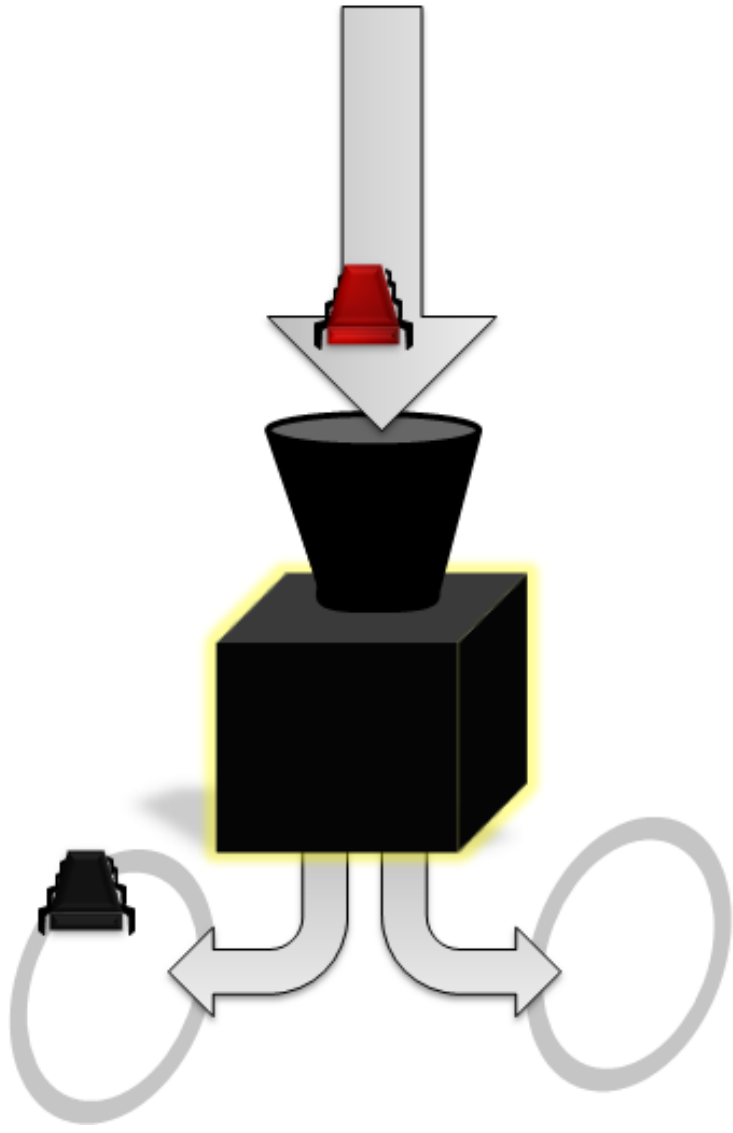
Strong Symmetry Breaking (SSB)

Processes start with unique input values
Must decide on a single bit s.t.:

- If all terminate,
  both 0 and 1 are decided
- In any execution, 1 is decided

If $(2p - \left\lceil \frac{p}{n-1} \right\rceil)$-adaptive renaming is solvable, then so is SSB

Suffices to show that SSB is unsolvable



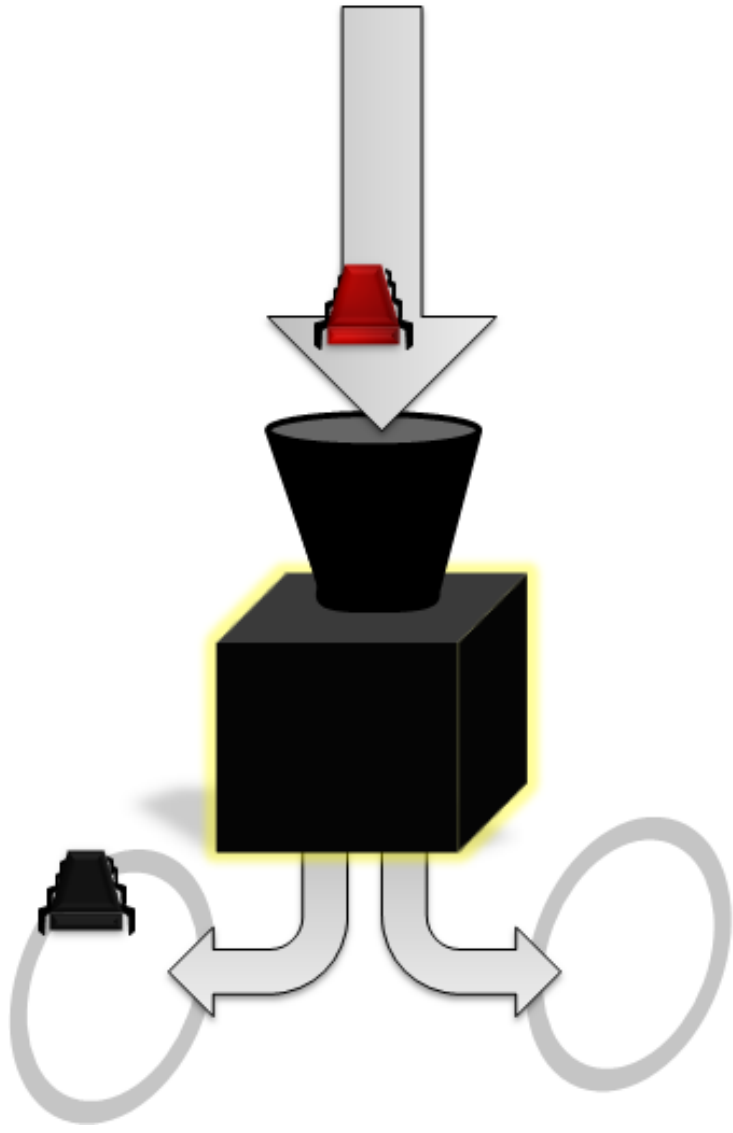
Strong Symmetry Breaking (SSB)

Processes start with unique input values
Must decide on a single bit s.t.:

- If all terminate,
  both 0 and 1 are decided
- In any execution, 1 is decided

If $(2p - \left\lceil \frac{p}{n-1} \right\rceil)$-adaptive renaming is solvable, then so is SSB

Suffices to show that SSB is unsolvable



SSB and WSB are Impossible: Proof Overview

Assume to the contrary, that some algorithm solves SSB / WSB
W.l.o.g, a process alternates between write and scan (read all)

Consider only its immediate atomic snapshot (IAS) executions

Show that one of them is univalued, e.g., all processes decide 0,
by counting the number of such executions

$\Rightarrow$ The algorithm has a univalued execution

SSB and WSB are Impossible: Proof Overview

Assume to the contrary, that some algorithm solves SSB / WSB
W.l.o.g, a process alternates between write and scan (read all)

Consider only its immediate atomic snapshot (IAS) executions

Show that one of them is univalued, e.g., all processes decide 0,
by counting the number of such executions

$\Rightarrow$ The algorithm has a univalued execution

SSB and WSB are Impossible: Proof Overview

Assume to the contrary, that some algorithm solves SSB / WSB
W.l.o.g, a process alternates between write and scan (read all)

Consider only its immediate atomic snapshot (IAS) executions

Show that one of them is univalued, e.g., all processes decide 0, by counting the number of such executions

- Estimate their number with the univalued signed count
- Define a trimmed version of the algorithm, preserving the univalued signed count
- Prove that the univalued signed count of the trimmed algorithm is $\neq 0$

$\Rightarrow$ The algorithm has a univalued execution

SSB and WSB are Impossible: Proof Overview

Assume to the contrary, that some algorithm solves SSB / WSB
W.l.o.g, a process alternates between write and scan (read all)

Consider only its immediate atomic snapshot (IAS) executions

Show that one of them is univalued, e.g., all processes decide 0,
by counting the number of such executions

- Estimate their number with the univalued signed count
- Define a trimmed version of the algorithm,
  preserving the univalued signed count
- Prove that the univalued signed count of the trimmed algorithm is $\neq 0$

$\Rightarrow$ The algorithm has a univalued execution

SSB and WSB are Impossible: Proof Overview

Assume to the contrary, that some algorithm solves SSB / WSB
W.l.o.g, a process alternates between write and scan (read all)

Consider only its immediate atomic snapshot (IAS) executions

Show that one of them is univalued, e.g., all processes decide 0, by counting the number of such executions

- Estimate their number with the univalued signed count
- Define a trimmed version of the algorithm, preserving the univalued signed count
- Prove that the univalued signed count of the trimmed algorithm is $\neq 0$

$\Rightarrow$ The algorithm has a univalued execution

SSB and WSB are Impossible: Proof Overview

Assume to the contrary, that some algorithm solves SSB / WSB
W.l.o.g, a process alternates between write and scan (read all)

Consider only its immediate atomic snapshot (IAS) executions

Show that one of them is univalued, e.g., all processes decide 0, by counting the number of such executions

- Estimate their number with the univalued signed count
- Define a trimmed version of the algorithm, preserving the univalued signed count
- Prove that the univalued signed count of the trimmed algorithm is $\neq 0$

$\Rightarrow$ The univalued signed count of the algorithm is $\neq 0$

$\Rightarrow$ The algorithm has a univalued execution

SSB and WSB are Impossible: Proof Overview

Assume to the contrary, that some algorithm solves SSB / WSB
W.l.o.g, a process alternates between write and scan (read all)

Consider only its immediate atomic snapshot (IAS) executions

Show that one of them is univalued, e.g., all processes decide 0, by counting the number of such executions

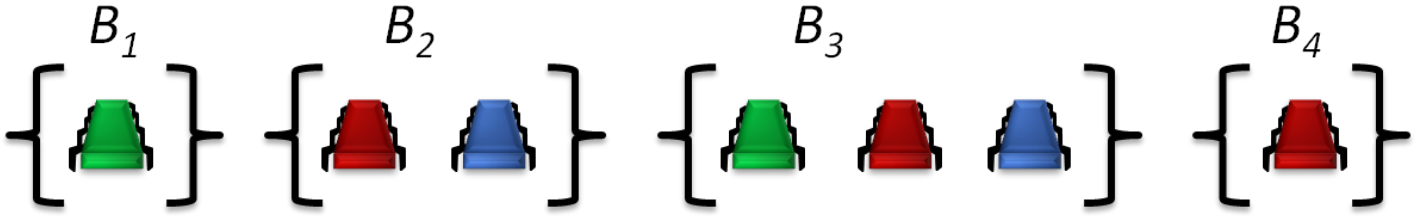
- Estimate their number with the univalued signed count
- Define a trimmed version of the algorithm, preserving the univalued signed count
- Prove that the univalued signed count of the trimmed algorithm is $\neq 0$

$\Rightarrow$ The univalued signed count of the algorithm is $\neq 0$

$\Rightarrow$ The algorithm has a univalued execution

Immediate Atomic Snapshot Executions

- Induced by a sequence of blocks of processes
- The processes in each block, first write, then scan



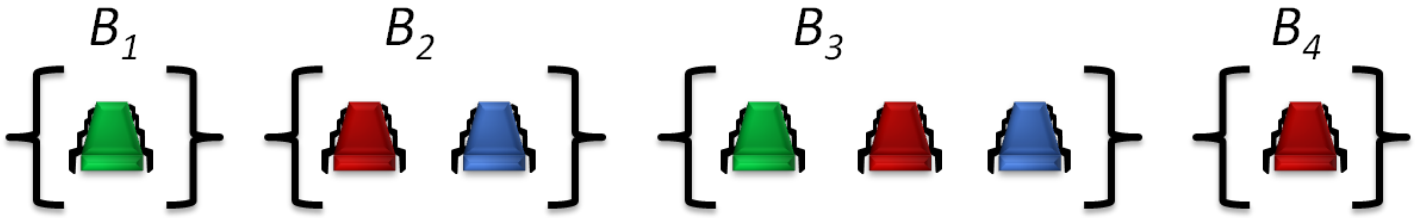
Lemma [Attiya and Rajsbaum, extended]

Immediate snapshot executions can be paired (α, α') s.t.:

- Exactly one process distinguishes between α and α'

Immediate Atomic Snapshot Executions

- Induced by a sequence of blocks of processes
- The processes in each block, first write, then scan



$$sign(\alpha) = \begin{cases} +1 & \alpha \text{ has an even number of even-sized blocks} \\ -1 & \alpha \text{ has an odd number of even-sized blocks} \end{cases}$$

- Odd-sized blocks do not affect the sign

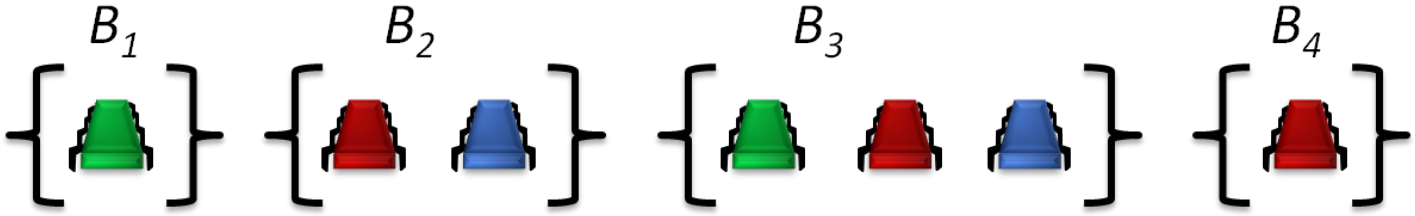
Lemma [Attiya and Rajsbaum, extended]

Immediate snapshot executions can be paired (α, α') s.t.:

- Exactly one process distinguishes between α and α'
- $sign(\alpha) = -sign(\alpha')$

Immediate Atomic Snapshot Executions

- Induced by a sequence of blocks of processes
- The processes in each block, first write, then scan



$$\text{sign}(\alpha) = \begin{cases} +1 & \alpha \text{ has an even number of even-sized blocks} \\ -1 & \alpha \text{ has an odd number of even-sized blocks} \end{cases}$$

- Odd-sized blocks do not affect the sign

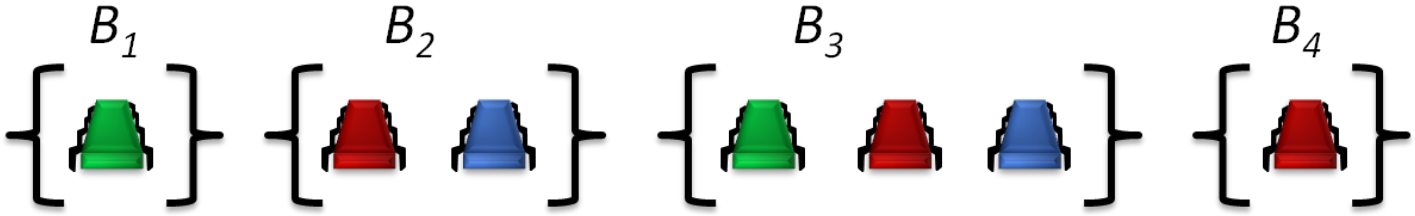
Lemma [Attiya and Rajsbaum, extended]

Immediate snapshot executions can be paired (α, α') s.t.:

- Exactly one process distinguishes between α and α'
- $\text{sign}(\alpha) = -\text{sign}(\alpha')$

Immediate Atomic Snapshot Executions

- Induced by a sequence of blocks of processes
- The processes in each block, first write, then scan



$$\text{sign}(\alpha) = \begin{cases} +1 & \alpha \text{ has an even number of even-sized blocks} \\ -1 & \alpha \text{ has an odd number of even-sized blocks} \end{cases}$$

- Odd-sized blocks do not affect the sign

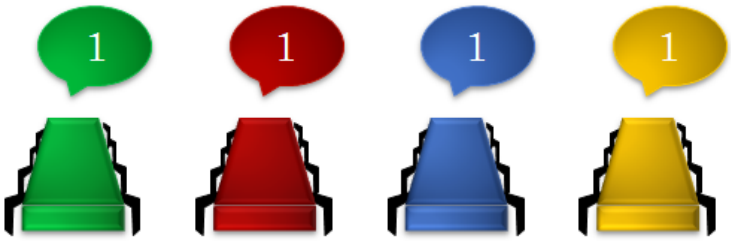
Lemma [Attiya and Rajsbaum, extended]

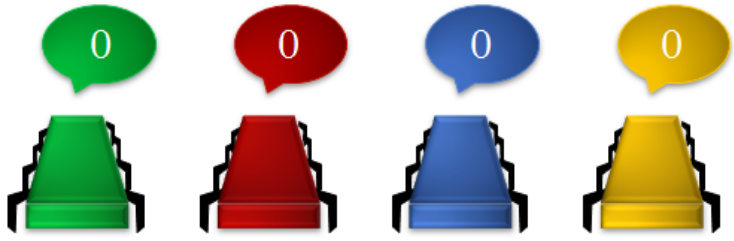
Immediate snapshot executions can be paired (α, α') s.t.:

- Exactly one process distinguishes between α and α'
- $\text{sign}(\alpha) = -\text{sign}(\alpha')$

The Univalued Signed Count of an Algorithm

$$(-1)^{n-1} \cdot \sum_{\text{only 1 is decided in } \alpha} \text{sign}(\alpha) \; + \sum_{\text{only 0 is decided in } \alpha} \text{sign}(\alpha)$$

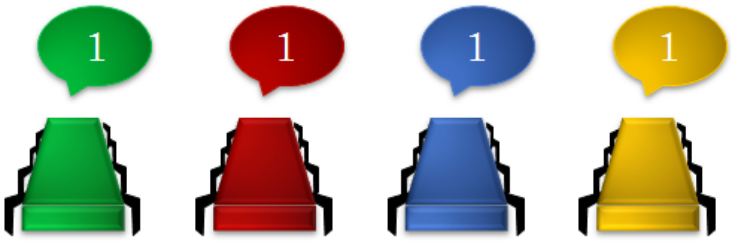


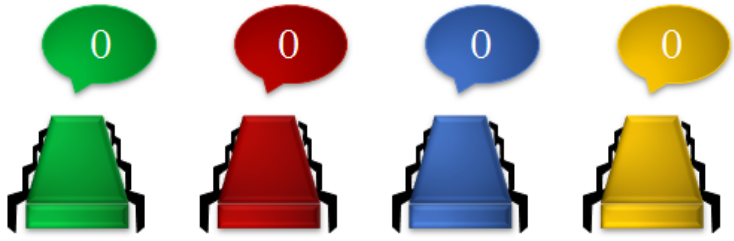


In WSB and in SSB, the univalued signed count has to be 0

The Univalued Signed Count of an Algorithm

$$(-1)^{n-1} \cdot \sum_{\text{only 1 is decided in } \alpha} \text{sign}(\alpha) \; + \sum_{\text{only 0 is decided in } \alpha} \text{sign}(\alpha)$$

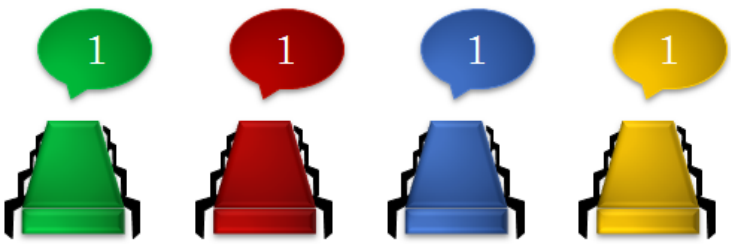


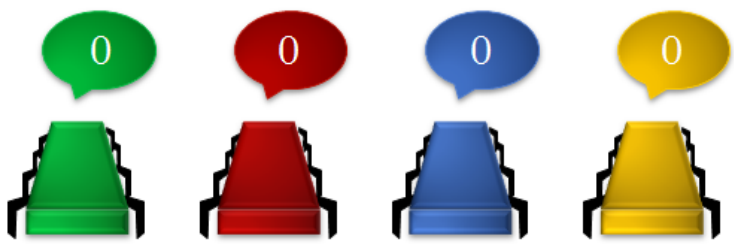


In WSB and in SSB, the univalued signed count has to be 0

The Univalued Signed Count of an Algorithm

$$(-1)^{n-1} \cdot \sum_{\text{only 1 is decided in } \alpha} \text{sign}(\alpha) \; + \sum_{\text{only 0 is decided in } \alpha} \text{sign}(\alpha)$$





In WSB and in SSB, the univalued signed count has to be 0

The Univalued Signed Count of an Algorithm

$$(-1)^{n-1} \cdot \sum_{\text{only 1 is decided in } \alpha} \text{sign}(\alpha) \; + \sum_{\text{only 0 is decided in } \alpha} \text{sign}(\alpha)$$

In WSB and in SSB, the univalued signed count has to be 0

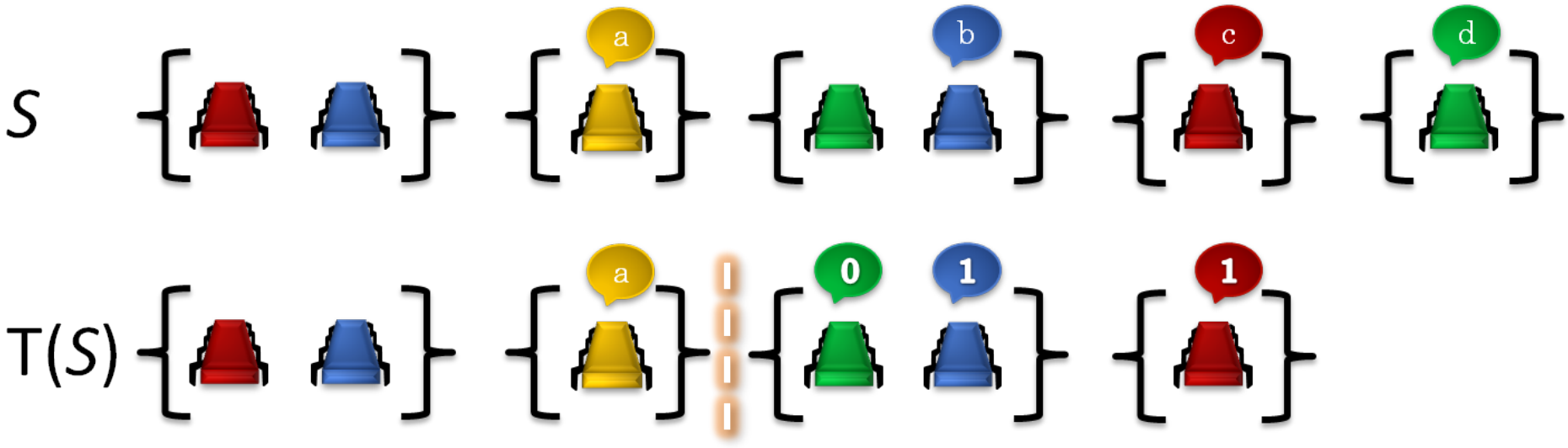
A Trimmed Version T(S), of an SSB Algorithm S

- Write your input and scan
  - If all processes have arrived, decide 0
- Repeat: Simulate a step of S, write and scan
  - If all processes have arrived, decide 1
  - If S decides, decide the same

Does not claim to solve SSB!

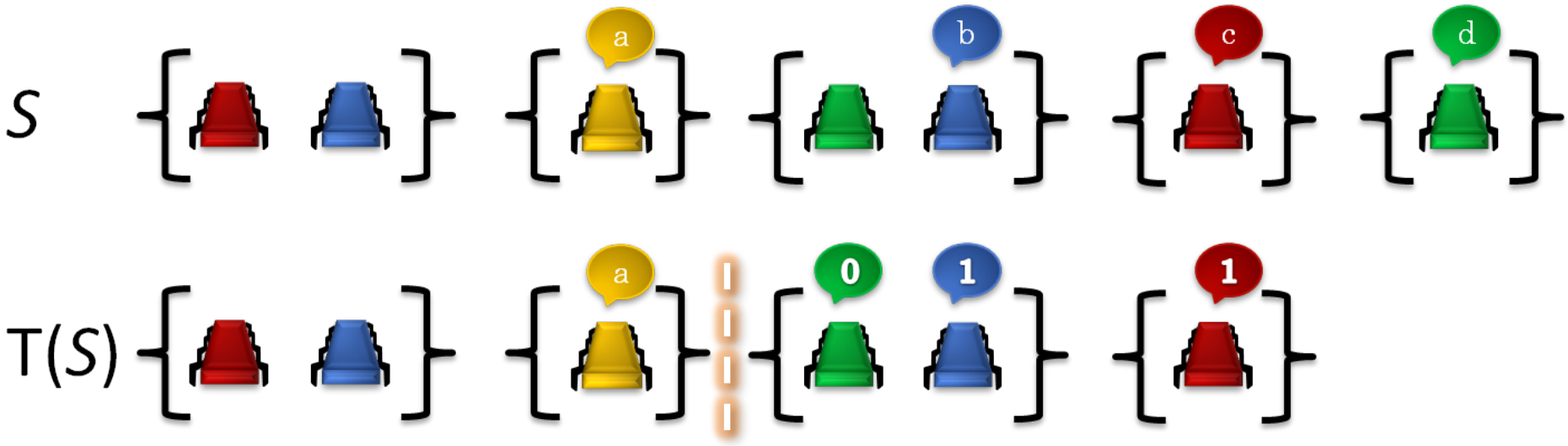
A Trimmed Version T(S), of an SSB Algorithm S

- Write your input and scan
  - If all processes have arrived, decide 0
- Repeat: Simulate a step of S, write and scan
  - If all processes have arrived, decide 1
  - If S decides, decide the same



A Trimmed Version T(S), of an SSB Algorithm S

- Write your input and scan
  - If all processes have arrived, decide 0
- Repeat: Simulate a step of S, write and scan
  - If all processes have arrived, decide 1
  - If S decides, decide the same



By properties of IAS executions and the extended AR Lemma:

Lemma: S and T(S) have the same univalued signed count

A Trimmed Version T(S), of an SSB Algorithm S

- Write your input and scan
  - If all processes have arrived, decide 0
- Repeat: Simulate a step of S, write and scan
  - If all processes have arrived, decide 1
  - If S decides, decide the same

- No execution in which only 1 is decided
  Since last process to arrive always decide 0
- All processes show up together $\Rightarrow$ only 0 is decided
- No other execution in which only 0 is decided

A Trimmed Version T(S), of an SSB Algorithm S

- Write your input and scan
  - If all processes have arrived, decide 0
- Repeat: Simulate a step of S, write and scan
  - If all processes have arrived, decide 1
  - If S decides, decide the same

- No execution in which only 1 is decided
  Since last process to arrive always decide 0
- All processes show up together $\Rightarrow$ only 0 is decided
- No other execution in which only 0 is decided:
  If a process takes a simulation step, some process decides 1

A Trimmed Version T(S), of an SSB Algorithm S

- Write your input and scan
  - If all processes have arrived, decide 0
- Repeat: Simulate a step of S, write and scan
  - If all processes have arrived, decide 1
  - If S decides, decide the same

- No execution in which only 1 is decided
  Since last process to arrive always decide 0
- All processes show up together $\Rightarrow$ only 0 is decided
- No other execution in which only 0 is decided:
  If a process takes a simulation step, some process decides 1

$\Rightarrow$ The univalued signed count of T(S) is $\neq 0$

By the lemma, S and T(S) have the same univalued signed count

Theorem: There is no algorithm solving SSB

Recall, SSB and WSB

Weak Symmetry Breaking (WSB)

Unique input values

Must decide on a single bit s.t.:

- If all terminate,
  both 0 and 1 are decided

Strong Symmetry Breaking (SSB)

Unique input values

Must decide on a single bit s.t.:

- If all terminate,
  both 0 and 1 are decided
- In any execution,
  1 is decided

What about a WSB algorithm, W?

- Write your input and scan
  - If all processes have arrived, decide 0
- Repeat: Simulate a step of W, write and scan
  - If all processes have arrived, decide 1
  - If W decides, decide the same

Lemma: W and $\mathsf{T}(W)$ have the same univalued signed count

- No execution in which only 1 is decided
- All processes show up together $\Rightarrow$ only 0 is decided
- But...
  there could be other executions in which only 0 is decided!

Assume the algorithm is rank based

What about a WSB algorithm, W?

- Write your input and scan
  - If all processes have arrived, decide 0
- Repeat: Simulate a step of W, write and scan
  - If all processes have arrived, decide 1
  - If W decides, decide the same

Lemma: W and $\mathsf{T}(W)$ have the same univalued signed count

- No execution in which only 1 is decided
- All processes show up together $\Rightarrow$ only 0 is decided
- But...
  there could be other executions in which only 0 is decided!

Assume the algorithm is rank based

What about a WSB algorithm, W?

- Write your input and scan
  - If all processes have arrived, decide 0
- Repeat: Simulate a step of W, write and scan
  - If all processes have arrived, decide 1
  - If W decides, decide the same

Lemma: W and $\mathsf{T}(W)$ have the same univalued signed count

- No execution in which only 1 is decided
- All processes show up together $\Rightarrow$ only 0 is decided
- But...
  there could be other executions in which only 0 is decided!

Assume the algorithm is rank based

What about a WSB algorithm, W?

- Write your input and scan
  - If all processes have arrived, decide 0
- Repeat: Simulate a step of W, write and scan
  - If all processes have arrived, decide 1
  - If W decides, decide the same

Lemma: W and $\mathsf{T}(W)$ have the same univalued signed count

- No execution in which only 1 is decided
- All processes show up together $\Rightarrow$ only 0 is decided
- But...
  there could be other executions in which only 0 is decided!

Assume the algorithm is rank based

Rank-Based Algorithms

Have the same outputs in symmetric executions:

- Same block structure
- The ranks of all processes in all blocks are the same

Note: Symmetric executions have the same sign

Rank-Based Algorithms

Have the same outputs in symmetric executions:

- Same block structure
- The ranks of all processes in all blocks are the same

Note: Symmetric executions have the same sign

Rank-Based Algorithms

Have the same outputs in symmetric executions:

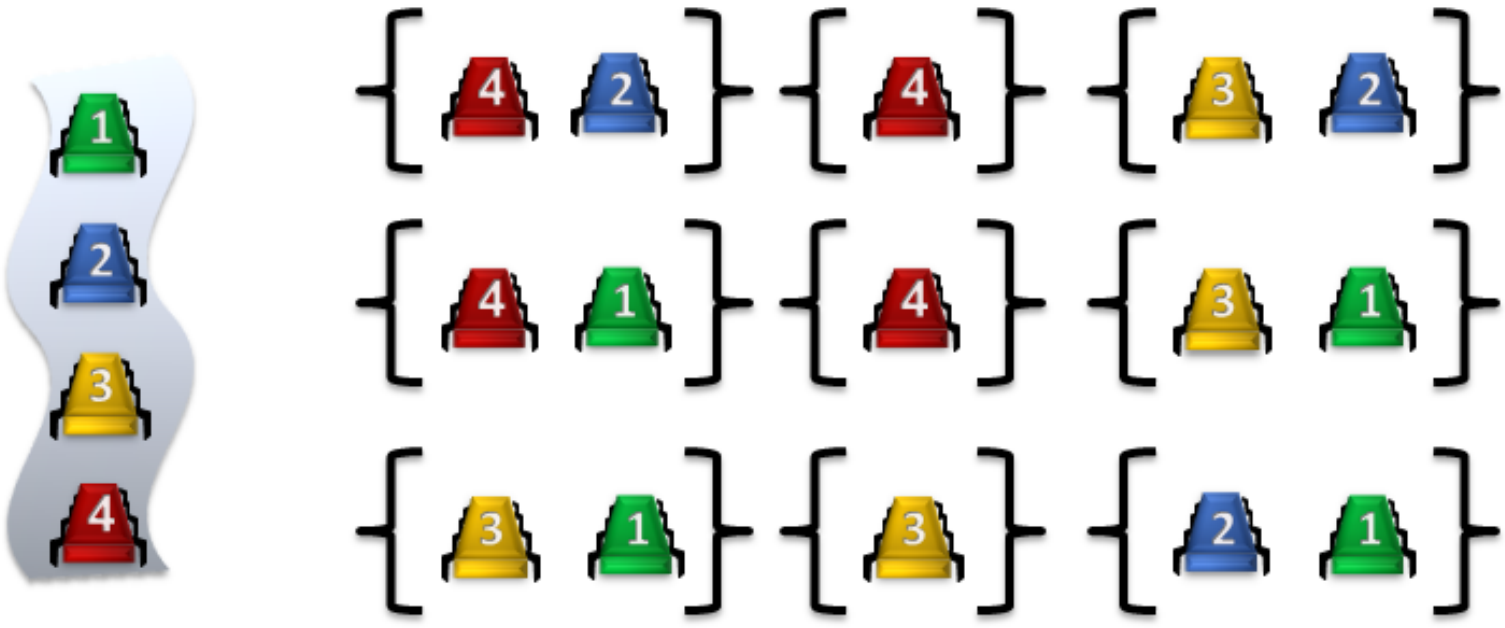
- Same block structure
- The ranks of all processes in all blocks are the same

Note: Symmetric executions have the same sign

Rank-Based Algorithms

Have the same outputs in symmetric executions:

- Same block structure
- The ranks of all processes in all blocks are the same



Note: Symmetric executions have the same sign

The univalued signed count

$$(-1)^{n-1} \cdot \sum_{\text{only 1 is decided in } \alpha} \text{sign}(\alpha) + \sum_{\text{only 0 is decided in } \alpha} \text{sign}(\alpha)$$

Executions deciding only 0

The univalued signed count of $T(W)$ is $\pm 1 \pmod{p} \Rightarrow \neq 0$.

The univalued signed count of $\mathsf{T}(W)$

$$(-1)^{n-1} \cdot \sum_{\text{only 1 is decided in } \alpha} \operatorname{sign}(\alpha) + \sum_{\text{only 0 is decided in } \alpha} \operatorname{sign}(\alpha)$$

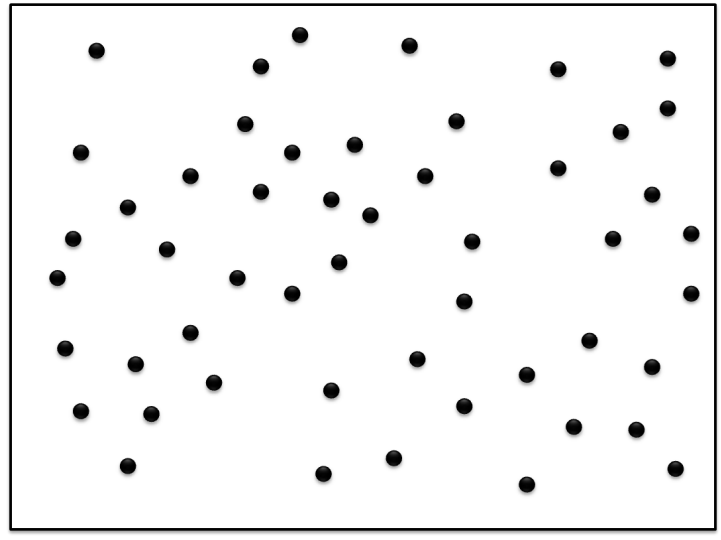
Executions deciding only 0

The univalued signed count of $\mathsf{T}(W)$ is $\pm 1 \pmod{p} \Rightarrow \neq 0$.

The univalued signed count of $\mathsf{T}(W)$

$$(-1)^{n-1} \cdot \sum_{\text{only 1 is decided in } \alpha} \operatorname{sign}(\alpha) + \sum_{\text{only 0 is decided in } \alpha} \operatorname{sign}(\alpha)$$

Executions deciding only 0



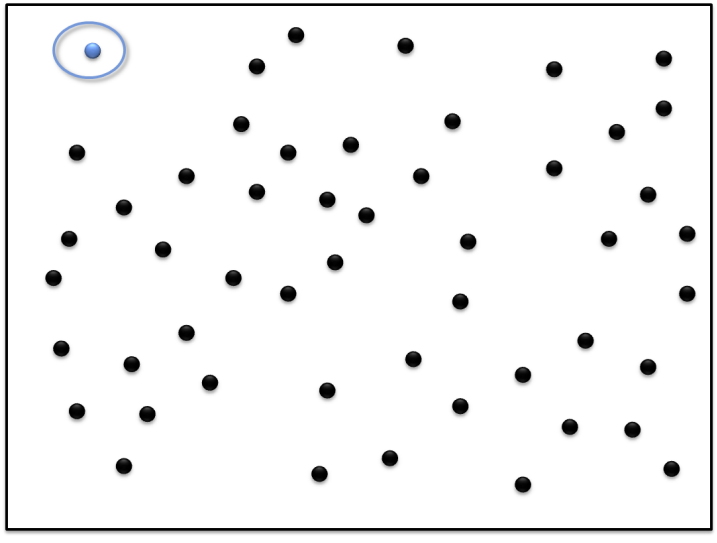
The univalued signed count of $\mathsf{T}(W)$ is $\pm 1 \pmod{p} \Rightarrow \neq 0$.

The univalued signed count of $\mathsf{T}(W)$

$$(-1)^{n-1} \cdot \sum_{\text{only 1 is decided in } \alpha} \operatorname{sign}(\alpha) + \sum_{\text{only 0 is decided in } \alpha} \operatorname{sign}(\alpha)$$

- Single execution in which all processes arrive together
  - Contributes ± 1 to the count

Executions deciding only 0



- Execution in which $m < n$ processes take steps in W
- There are $\binom{n}{m}$ such executions
- When $n = p^e$, they contribute $0 \pmod{p}$ to the count

The univalued signed count of $\mathsf{T}(W)$ is $\pm 1 \pmod{p} \Rightarrow \neq 0$.

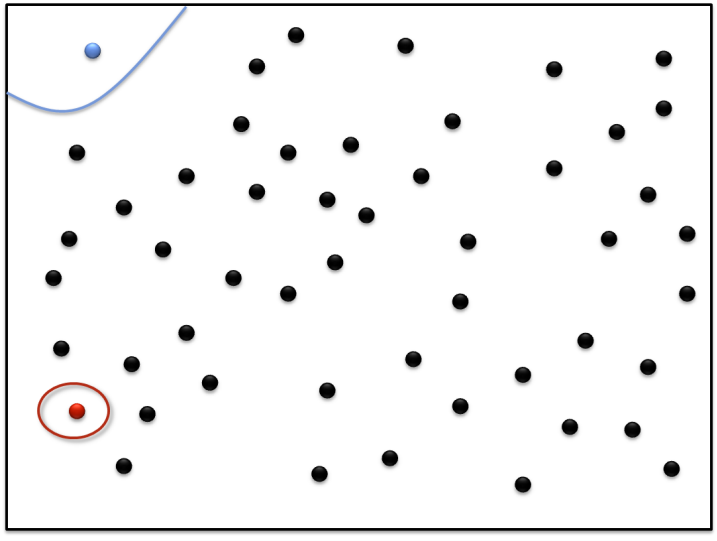
The univalued signed count of $\mathsf{T}(W)$

$$(-1)^{n-1} \cdot \sum_{\text{only 1 is decided in } \alpha} \operatorname{sign}(\alpha) + \sum_{\text{only 0 is decided in } \alpha} \operatorname{sign}(\alpha)$$

- Single execution in which all processes arrive together
  - Contributes ± 1 to the count
- Execution in which $m < n$ processes take steps in W
  - All symmetric executions decide 0 and have the same sign
- There are $\binom{n}{m}$ such executions
- When $n = p^e$, they contribute $0 \pmod p$ to the count

The univalued signed count of $\mathsf{T}(W)$ is $\pm 1 \pmod p \Rightarrow \neq 0$.

Executions deciding only 0

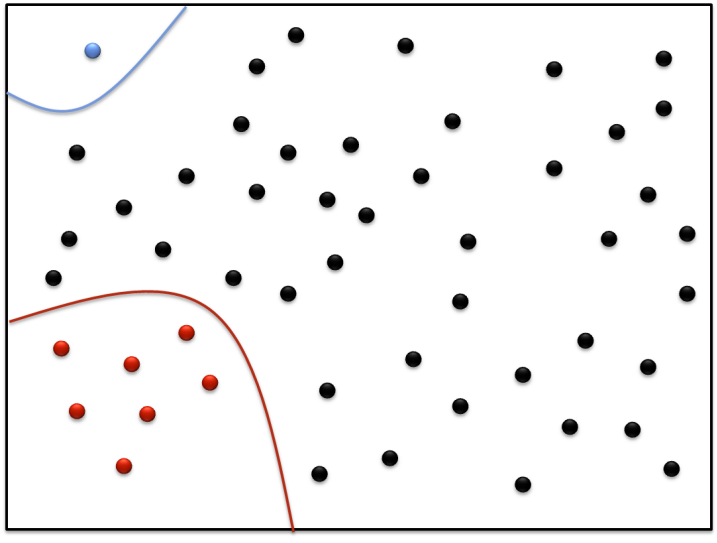


The univalued signed count of $\mathsf{T}(W)$

$$(-1)^{n-1} \cdot \sum_{\text{only 1 is decided in } \alpha} \operatorname{sign}(\alpha) + \sum_{\text{only 0 is decided in } \alpha} \operatorname{sign}(\alpha)$$

- Single execution in which all processes arrive together
  - Contributes ± 1 to the count
- Execution in which $m < n$ processes take steps in W
  - All symmetric executions decide 0 and have the same sign

Executions deciding only 0



- There are $\binom{n}{m}$ such executions
- When $n = p^e$, they contribute $0 \pmod p$ to the count

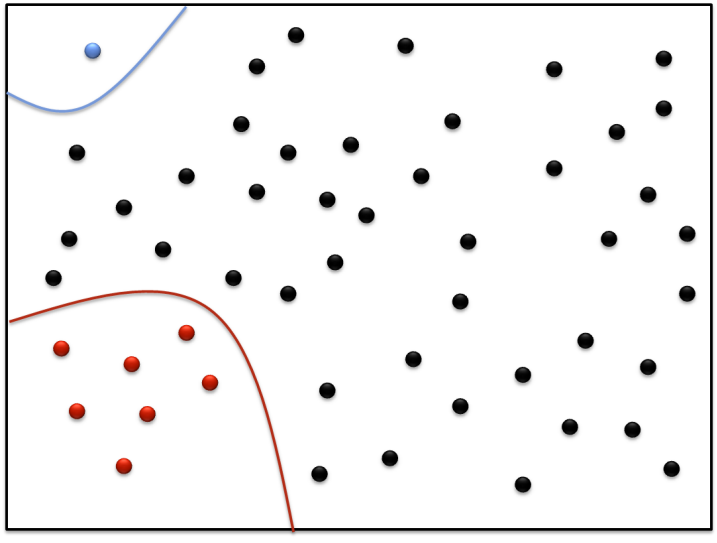
The univalued signed count of $\mathsf{T}(W)$ is $\pm 1 \pmod p \Rightarrow \neq 0$

The univalued signed count of $\mathrm{T}(W)$

$$(-1)^{n-1} \cdot \sum_{\text{only 1 is decided in } \alpha} \operatorname{sign}(\alpha) + \sum_{\text{only 0 is decided in } \alpha} \operatorname{sign}(\alpha)$$

- Single execution in which all processes arrive together
  - Contributes ± 1 to the count
- Execution in which $m < n$ processes take steps in W
  - All symmetric executions decide 0 and have the same sign
- There are $\binom{n}{m}$ such executions
- When $n = p^e$, they contribute $0 \pmod{p}$ to the count

Executions deciding only 0



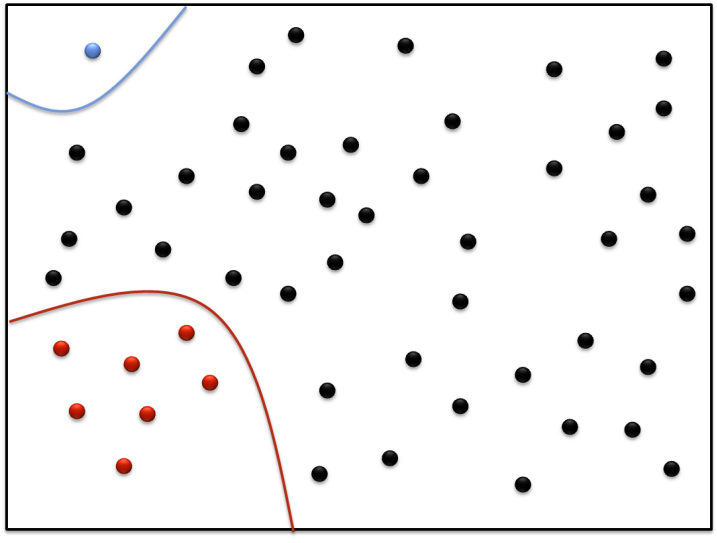
The univalued signed count of $\mathrm{T}(W)$ is $\pm 1 \pmod{p} \Rightarrow \neq 0$.

The univalued signed count of $\mathsf{T}(W)$

$$(-1)^{n-1} \cdot \sum_{\text{only 1 is decided in } \alpha} \operatorname{sign}(\alpha) + \sum_{\text{only 0 is decided in } \alpha} \operatorname{sign}(\alpha)$$

- Single execution in which all processes arrive together
  - Contributes ± 1 to the count
- Execution in which $m < n$ processes take steps in W
  - All symmetric executions decide 0 and have the same sign
- There are $\binom{n}{m}$ such executions
- When $n = p^e$, they contribute $0 \pmod{p}$ to the count

Executions deciding only 0



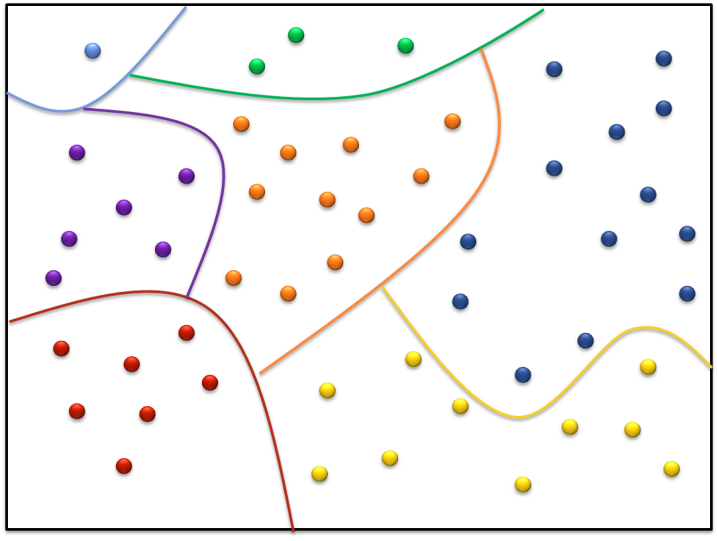
The univalued signed count of $\mathsf{T}(W)$ is $\pm 1 \pmod{p} \Rightarrow \neq 0$.

The univalued signed count of $\mathsf{T}(W)$

$$(-1)^{n-1} \cdot \sum_{\text{only 1 is decided in } \alpha} \operatorname{sign}(\alpha) + \sum_{\text{only 0 is decided in } \alpha} \operatorname{sign}(\alpha)$$

- Single execution in which all processes arrive together
  - Contributes ± 1 to the count
- Execution in which $m < n$ processes take steps in W
  - All symmetric executions decide 0 and have the same sign
- There are $\binom{n}{m}$ such executions
- When $n = p^e$, they contribute $0 \pmod{p}$ to the count

Executions deciding only 0



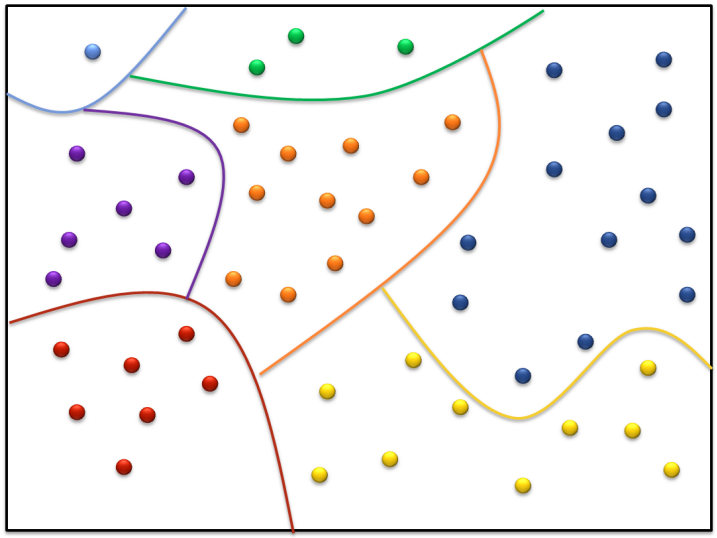
The univalued signed count of $\mathsf{T}(W)$ is $\pm 1 \pmod{p} \Rightarrow \neq 0$.

The univalued signed count of $\mathsf{T}(W)$

$$(-1)^{n-1} \cdot \sum_{\text{only 1 is decided in } \alpha} \operatorname{sign}(\alpha) + \sum_{\text{only 0 is decided in } \alpha} \operatorname{sign}(\alpha)$$

- Single execution in which all processes arrive together
  - Contributes ± 1 to the count
- Execution in which $m < n$ processes take steps in W
  - All symmetric executions decide 0 and have the same sign
- There are $\binom{n}{m}$ such executions
- When $n = p^e$, they contribute $0 \pmod p$ to the count

Executions deciding only 0



The univalued signed count of $\mathsf{T}(W)$ is $\pm 1 \pmod p \Rightarrow \neq 0$.

Wrap Up

- The univalued signed count of $\mathsf{T}(W)$ is $\pm 1 \pmod{p}$

$\Rightarrow$ The univalued signed count of $\mathsf{T}(W)$ is $\neq 0$

- $\mathsf{T}(W)$ and W have same univalued signed count

$\Rightarrow$ The univalued signed count of W is $\neq 0$

$\Rightarrow$ W has a univalued execution

$\Rightarrow$ W does not solve WSB!

Wrap Up

- The univalued signed count of $\mathsf{T}(W)$ is $\pm 1 \pmod{p}$

$\Rightarrow$ The univalued signed count of $\mathsf{T}(W)$ is $\neq 0$

- $\mathsf{T}(W)$ and W have same univalued signed count

$\Rightarrow$ The univalued signed count of W is $\neq 0$

$\Rightarrow$ W has a univalued execution

$\Rightarrow$ W does not solve WSB!

Wrap Up

- The univalued signed count of $\mathsf{T}(W)$ is $\pm 1 \pmod{p}$

$\Rightarrow$ The univalued signed count of $\mathsf{T}(W)$ is $\neq 0$

- $\mathsf{T}(W)$ and W have same univalued signed count

$\Rightarrow$ The univalued signed count of W is $\neq 0$

$\Rightarrow$ W has a univalued execution

$\Rightarrow$ W does not solve WSB!

Wrap Up

- The univalued signed count of $\mathsf{T}(W)$ is $\pm 1 \pmod{p}$

$\Rightarrow$ The univalued signed count of $\mathsf{T}(W)$ is $\neq 0$

- $\mathsf{T}(W)$ and W have same univalued signed count

$\Rightarrow$ The univalued signed count of W is $\neq 0$

$\Rightarrow$ W has a univalued execution

$\Rightarrow$ W does not solve WSB!

Wrap Up

- The univalued signed count of $\mathsf{T}(W)$ is $\pm 1 \pmod{p}$

$\Rightarrow$ The univalued signed count of $\mathsf{T}(W)$ is $\neq 0$

- $\mathsf{T}(W)$ and W have same univalued signed count

$\Rightarrow$ The univalued signed count of W is $\neq 0$

$\Rightarrow$ W has a univalued execution

$\Rightarrow$ W does not solve WSB!

Summary

This paper:

- Simple impossibility proof for $(n-1)$-set agreement
- Simple impossibility proof for $(2p - \left\lceil \frac{p}{n-1} \right\rceil)$-adaptive renaming
- n is a prime-power $\Rightarrow$ $(2n-2)$-renaming impossible

Future research:

- When n is not a prime-power:
  - Explicit algorithm for $(2n-2)$-renaming; complexity
  - $(2n-3)$-renaming and below
- Other models: message passing, partial synchrony
- Other colored tasks?

Summary

This paper:

- Simple impossibility proof for $(n-1)$-set agreement
- Simple impossibility proof for $(2p - \left\lceil \frac{p}{n-1} \right\rceil)$-adaptive renaming
- n is a prime-power $\Rightarrow$ $(2n-2)$-renaming impossible

Future research:

- When n is not a prime-power:
  - Explicit algorithm for $(2n-2)$-renaming; complexity
  - $(2n-3)$-renaming and below
- Other models: message passing, partial synchrony
- Other colored tasks?

Summary

This paper:

- Simple impossibility proof for $(n-1)$-set agreement
- Simple impossibility proof for $(2p - \left\lceil \frac{p}{n-1} \right\rceil)$-adaptive renaming
- n is a prime-power $\Rightarrow$ $(2n-2)$-renaming impossible

Future research:

- When n is not a prime-power:
  - Explicit algorithm for $(2n-2)$-renaming; complexity
  - $(2n-3)$-renaming and below
- Other models: message passing, partial synchrony
- Other colored tasks?