Wait-free Solvability of Equality Negation Tasks

Éric Goubault[1] Marijana Lazić[2] **Jérémy Ledent**[1] Sergio Rajsbaum[3]

[1]École Polytechnique, Palaiseau, France

[2]TU München, Munich, Germany

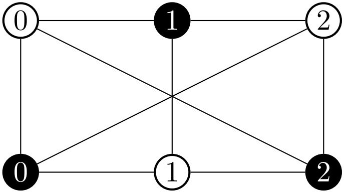
[3]UNAM, Mexico City, Mexico

DISC 2019
Budapest, Hungary
October 16th, 2019

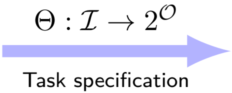
Equality Negation

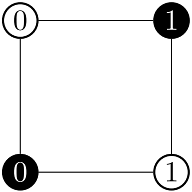
(Lo and Hadzilacos, *Nondeterministic wait-free hierarchies are not robust*, 2000)

- Two processes P, Q (represented in black and white).
- Three possible inputs values $i_P, i_Q \in \{0, 1, 2\}$.
- Binary decision values $d_P, d_Q \in \{0, 1\}$.
- **Goal:** $i_P = i_Q \iff d_P \neq d_Q$.



Input complex $\mathcal{I}$



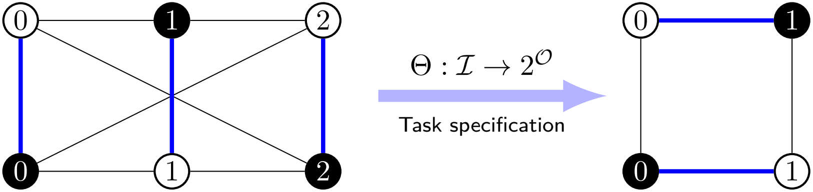


Output complex $\mathcal{O}$

Equality Negation

(Lo and Hadzilacos, *Nondeterministic wait-free hierarchies are not robust*, 2000)

- Two processes P, Q (represented in black and white).
- Three possible inputs values $i_P, i_Q \in \{0, 1, 2\}$.
- Binary decision values $d_P, d_Q \in \{0, 1\}$.
- **Goal:** $i_P = i_Q \iff d_P \neq d_Q$.



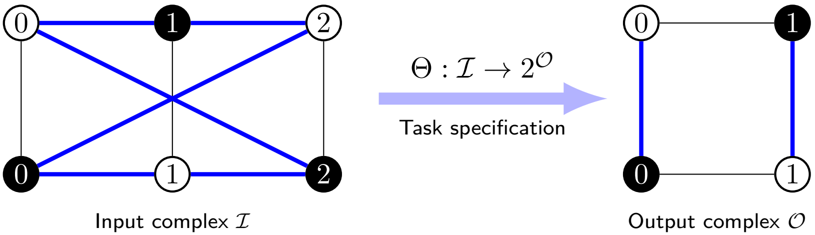
Input complex $\mathcal{I}$

Output complex $\mathcal{O}$

Equality Negation

(Lo and Hadzilacos, *Nondeterministic wait-free hierarchies are not robust*, 2000)

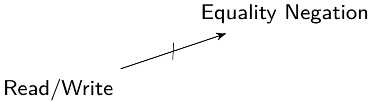
- Two processes P, Q (represented in black and white).
- Three possible inputs values $i_P, i_Q \in \{0, 1, 2\}$.
- Binary decision values $d_P, d_Q \in \{0, 1\}$.
- **Goal:** $i_P = i_Q \iff d_P \neq d_Q$.



Equality Negation (2)

Facts: (Lo and Hadzilacos, 2000)

(1) EN is not wait-free solvable using read/write registers.

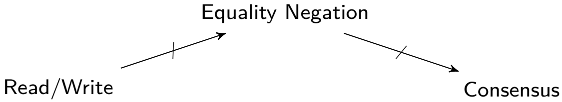


Equality Negation (2)

Facts: (Lo and Hadzilacos, 2000)

(1) EN is not wait-free solvable using read/write registers.

(2) Consensus is not wait-free solvable using EN objects.



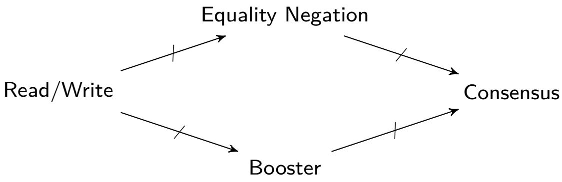
Equality Negation (2)

Facts: (Lo and Hadzilacos, 2000)

(1) EN is not wait-free solvable using read/write registers.

(2) Consensus is not wait-free solvable using EN objects.

(3) The "Booster" object also has properties (1) and (2).



Equality Negation (2)

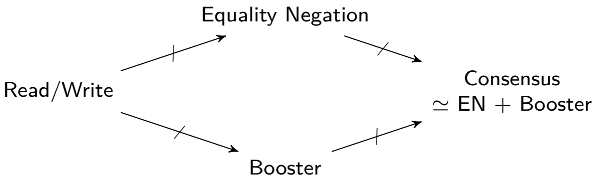
Facts: (Lo and Hadzilacos, 2000)

(1) EN is not wait-free solvable using read/write registers.

(2) Consensus is not wait-free solvable using EN objects.

(3) The "Booster" object also has properties (1) and (2).

(4) But EN + Booster can implement consensus!



Why is Equality Negation interesting?

Our goal: understand sub-consensus tasks better.

Why is Equality Negation interesting?

Our goal: understand sub-consensus tasks better.

Equality negation shares characteristics with two important tasks:

- Consensus: if inputs are different, the processes must *agree*.
- Symmetry breaking: if inputs are equal, they must *disagree*.

Why is Equality Negation interesting?

Our goal: understand sub-consensus tasks better.

Equality negation shares characteristics with two important tasks:

- Consensus: if inputs are different, the processes must *agree*.
- Symmetry breaking: if inputs are equal, they must *disagree*.

We have two papers about this task:

(1) *A Dynamic Epistemic Logic Analysis of the Equality Negation Task*, DaLi'19.
$\longrightarrow$ The reason why EN is not solvable cannot be expressed in the language of epistemic logic.

Why is Equality Negation interesting?

Our goal: understand sub-consensus tasks better.

Equality negation shares characteristics with two important tasks:

- Consensus: if inputs are different, the processes must *agree*.
- Symmetry breaking: if inputs are equal, they must *disagree*.

We have two papers about this task:

(1) *A Dynamic Epistemic Logic Analysis of the Equality Negation Task*, DaLi'19.
$\longrightarrow$ The reason why EN is not solvable cannot be expressed in the language of epistemic logic.

(2) This talk:
$\longrightarrow$ Extend the task to n processes and study its solvability.

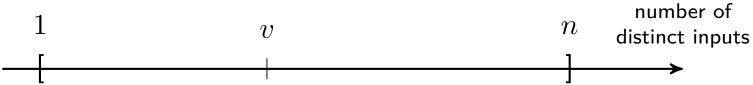
Equality Negation for n processes

- A fixed number n of processes $P_0, \ldots, P_{n-1}$.
- At least n possible input values $\{0, 1, \ldots, n-1\}$.
- Binary decision values $\{0, 1\}$.

Equality Negation for n processes

- A fixed number n of processes $P_0, \ldots, P_{n-1}$.
- At least n possible input values $\{0, 1, \ldots, n-1\}$.
- Binary decision values $\{0, 1\}$.

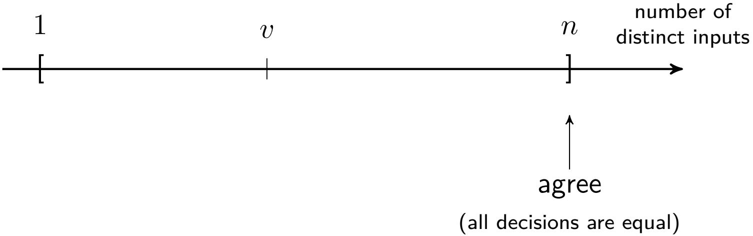
Let $1 \leq v \leq n$ denote the number of distinct input values.



Equality Negation for n processes

- A fixed number n of processes $P_0, \ldots, P_{n-1}$.
- At least n possible input values $\{0, 1, \ldots, n-1\}$.
- Binary decision values $\{0, 1\}$.

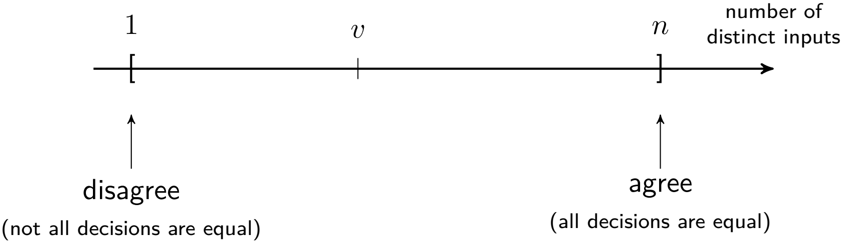
Let $1 \leq v \leq n$ denote the number of distinct input values.



Equality Negation for n processes

- A fixed number n of processes $P_0, \ldots, P_{n-1}$.
- At least n possible input values $\{0, 1, \ldots, n-1\}$.
- Binary decision values $\{0, 1\}$.

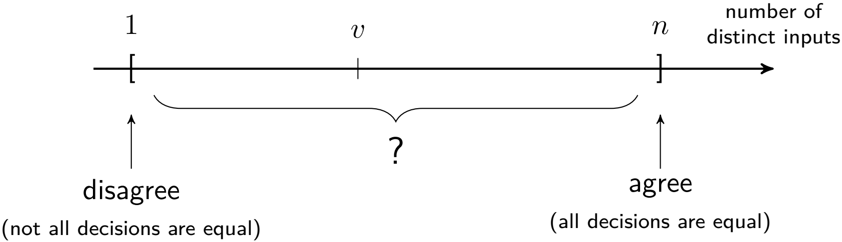
Let $1 \leq v \leq n$ denote the number of distinct input values.



Equality Negation for n processes

- A fixed number n of processes $P_0, \ldots, P_{n-1}$.
- At least n possible input values $\{0, 1, \ldots, n-1\}$.
- Binary decision values $\{0, 1\}$.

Let $1 \leq v \leq n$ denote the number of distinct input values.

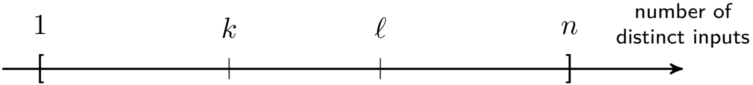


Equality Negation for n processes

- A fixed number n of processes $P_0, \ldots, P_{n-1}$.
- At least n possible input values $\{0, 1, \ldots, n-1\}$.
- Binary decision values $\{0, 1\}$.

Let $1 \leq v \leq n$ denote the number of distinct input values.

Fix two parameters $1 \leq k < \ell \leq n$.

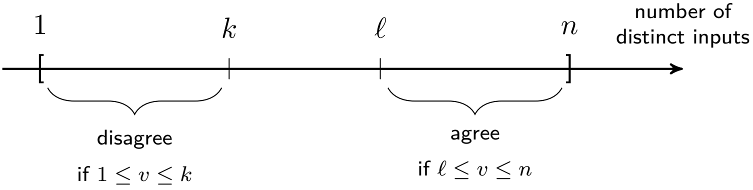


Equality Negation for n processes

- A fixed number n of processes $P_0, \ldots, P_{n-1}$.
- At least n possible input values $\{0, 1, \ldots, n-1\}$.
- Binary decision values $\{0, 1\}$.

Let $1 \leq v \leq n$ denote the number of distinct input values.

Fix two parameters $1 \leq k < \ell \leq n$.

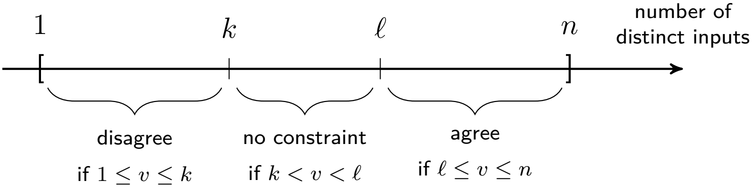


Equality Negation for n processes

- A fixed number n of processes $P_0, \ldots, P_{n-1}$.
- At least n possible input values $\{0, 1, \ldots, n-1\}$.
- Binary decision values $\{0, 1\}$.

Let $1 \leq v \leq n$ denote the number of distinct input values.

Fix two parameters $1 \leq k < \ell \leq n$.

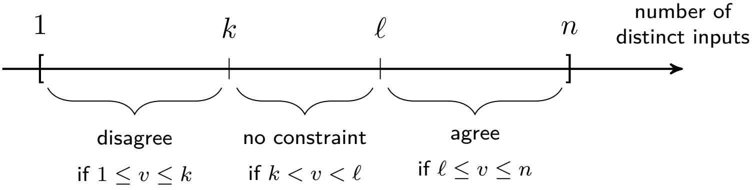


Equality Negation for n processes

- A fixed number n of processes $P_0, \ldots, P_{n-1}$.
- At least n possible input values $\{0, 1, \ldots, n-1\}$.
- Binary decision values $\{0, 1\}$.

Let $1 \leq v \leq n$ denote the number of distinct input values.

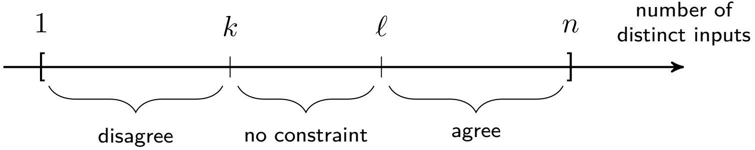
Fix two parameters $1 \leq k < \ell \leq n$.



$\longrightarrow$ We get a family of tasks EN(k, ℓ).

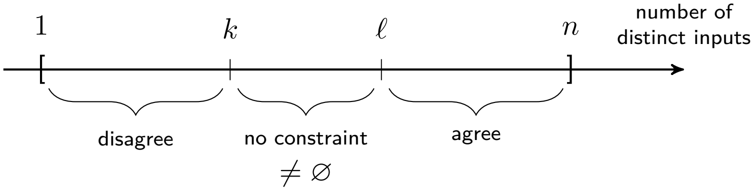
Solvable cases

Reminder: parameters $1 \leq k < \ell \leq n$.



Solvable cases

Reminder: parameters $1 \leq k < \ell \leq n$.

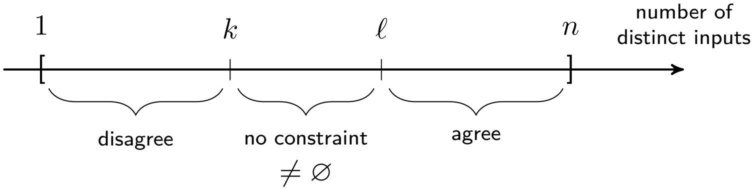


Theorem

If $k + 2 \leq \ell$, the task EN(k,ℓ) is wait-free solvable using read/write.

Solvable cases

Reminder: parameters $1 \leq k < \ell \leq n$.



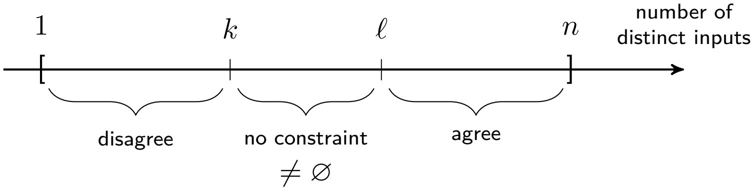
Theorem

If $k+2 \leq \ell$, the task EN(k,ℓ) is wait-free solvable using read/write.

- Very simple algorithm (one round of immediate-snapshot).

Solvable cases

Reminder: parameters $1 \leq k < \ell \leq n$.



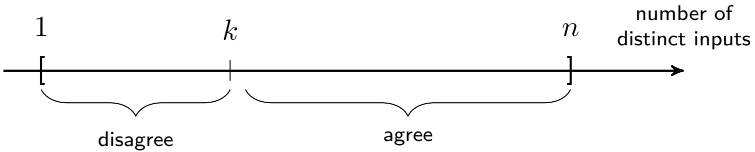
Theorem

If $k+2 \leq \ell$, the task EN(k,ℓ) is wait-free solvable using read/write.

- Very simple algorithm (one round of immediate-snapshot).
- Not anonymous!

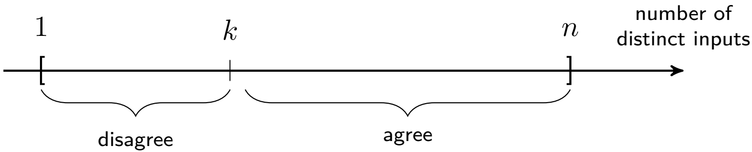
Unsolvable cases

Parameter $1 \leq k < n$.



Unsolvable cases

Parameter $1 \leq k < n$.



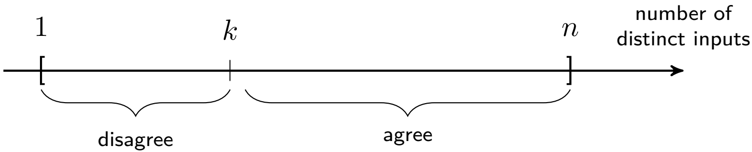
Theorem

If $k \leq n/2$, the task EN(k,$k+1$) is not solvable using registers.

- Uses Sperner's Lemma

Unsolvable cases

Parameter $1 \leq k < n$.



Theorem

If $k \leq n/2$, the task EN(k,$k+1$) is not solvable using registers.

- Uses Sperner's Lemma

Theorem

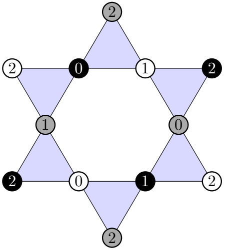
If $n-k$ is odd, the task EN(k,$k+1$) is not solvable using registers.

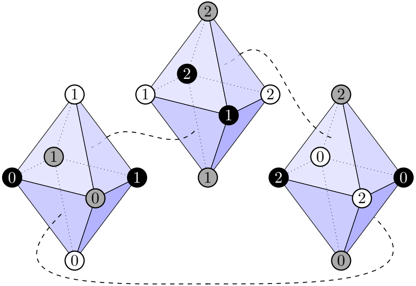
- Uses the Index Lemma

Proof sketch for $n = 3$, $k = 2$

- Three processes: Black, Gray, White.
- Three inputs: 0, 1, 2.

The input complex $\mathcal{I}$ looks like this (exploded view):

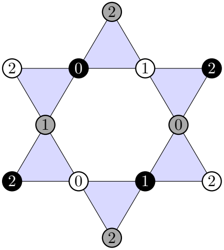




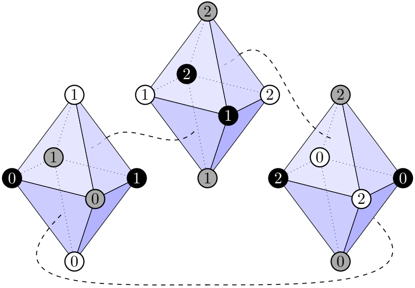
Proof sketch for $n = 3$, $k = 2$

- Three processes: Black, Gray, White.
- Three inputs: 0, 1, 2.

The input complex $\mathcal{I}$ looks like this (exploded view):



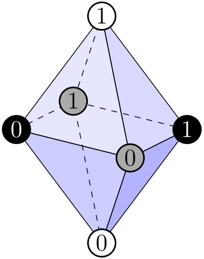
> 2 distinct inputs
$\rightarrow$ agree



≤ 2 distinct inputs
$\rightarrow$ disagree

Proof sketch for $n = 3$, $k = 2$

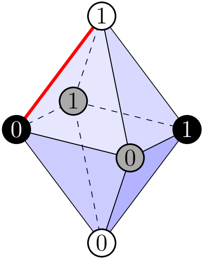
We focus on a subcomplex $S \subseteq \mathcal{I}$ of the input complex:



$S \subseteq \mathcal{I}$
Disagree

Proof sketch for $n = 3$, $k = 2$

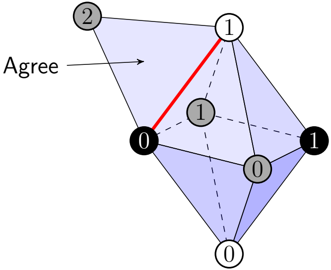
We focus on a subcomplex $S \subseteq \mathcal{I}$ of the input complex:



$S \subseteq \mathcal{I}$
Disagree

Proof sketch for $n = 3$, $k = 2$

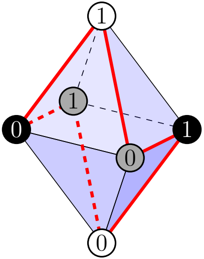
We focus on a subcomplex $S \subseteq \mathcal{I}$ of the input complex:



$S \subseteq \mathcal{I}$
Disagree

Proof sketch for $n = 3$, $k = 2$

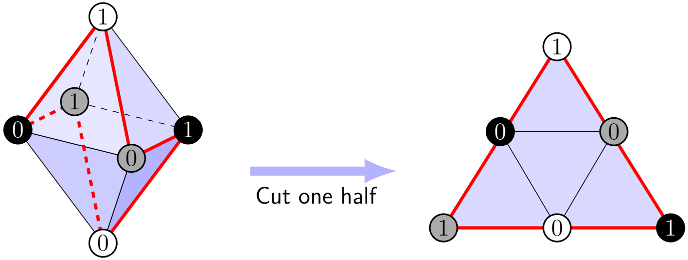
We focus on a subcomplex $S \subseteq \mathcal{I}$ of the input complex:



$S \subseteq \mathcal{I}$
Disagree

Proof sketch for $n = 3$, $k = 2$

We focus on a subcomplex $S \subseteq \mathcal{I}$ of the input complex:

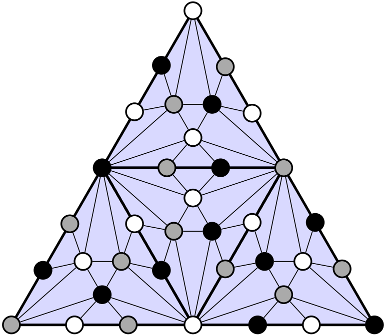


$S \subseteq \mathcal{I}$
Disagree

$H \subseteq S$

Proof sketch for $n = 3$, $k = 2$

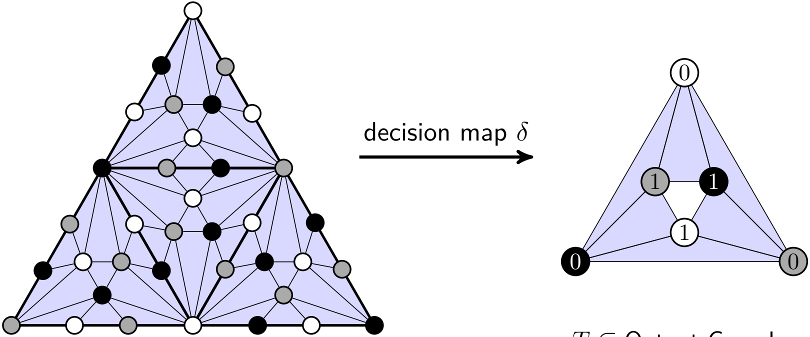
After immediate-snapshot communication (here, one round):



SubDiv$(H) \subseteq$ Protocol Complex

Proof sketch for $n = 3$, $k = 2$

After immediate-snapshot communication (here, one round):

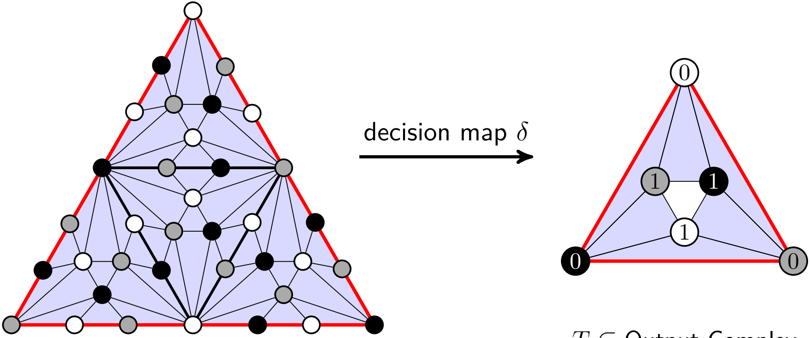


$T \subseteq$ Output Complex

SubDiv$(H) \subseteq$ Protocol Complex

Proof sketch for $n = 3$, $k = 2$

After immediate-snapshot communication (here, one round):

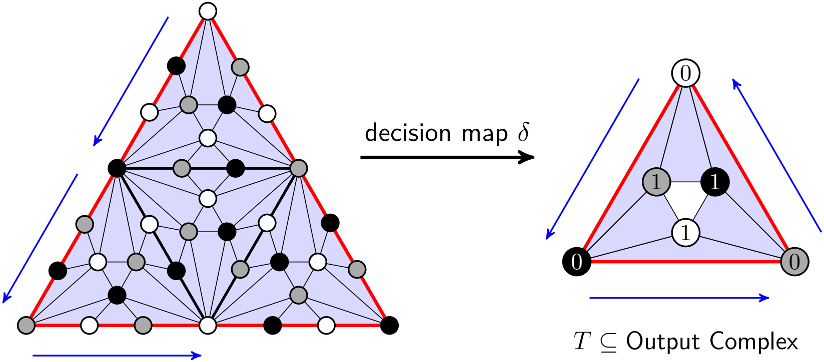


SubDiv$(H) \subseteq$ Protocol Complex

$T \subseteq$ Output Complex

Proof sketch for $n = 3$, $k = 2$

After immediate-snapshot communication (here, one round):

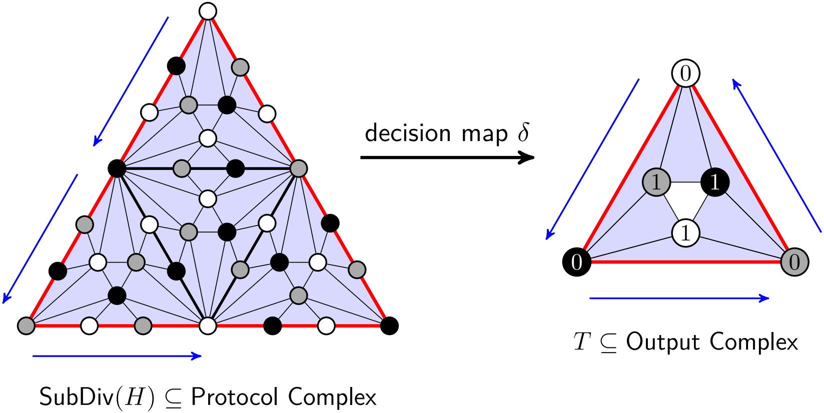


SubDiv$(H) \subseteq$ Protocol Complex

$T \subseteq$ Output Complex

Proof sketch for $n = 3$, $k = 2$

After immediate-snapshot communication (here, one round):



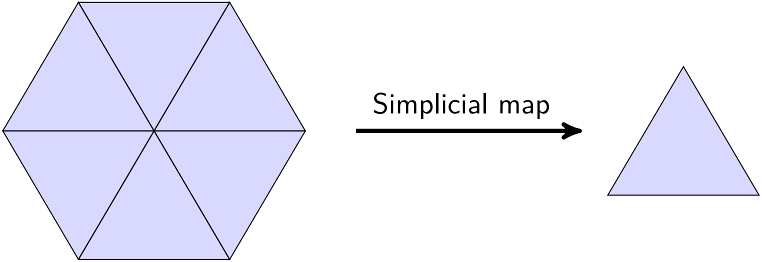
The boundary of SubDiv(H) is winding twice around the boundary of T.

The Index Lemma

A combinatorial version of the notion of degree of a continuous map (or winding number, in dimension 1).

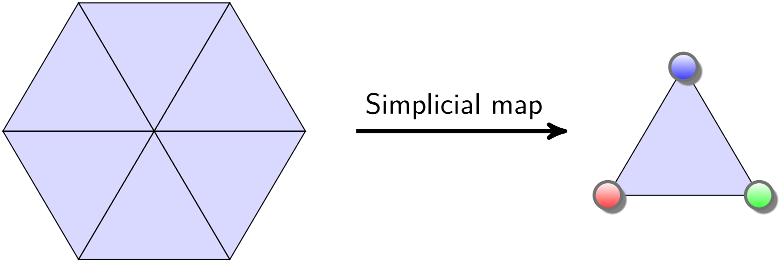
The Index Lemma

A combinatorial version of the notion of degree of a continuous map (or winding number, in dimension 1).



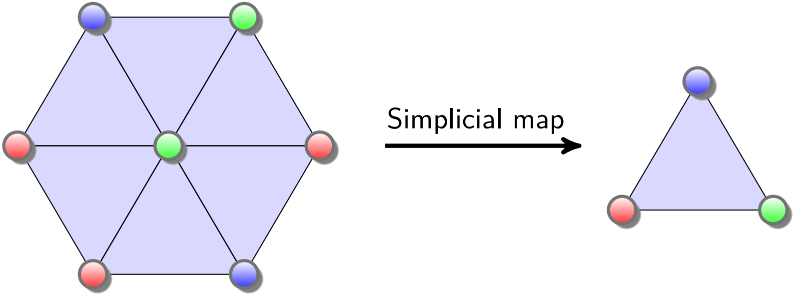
The Index Lemma

A combinatorial version of the notion of degree of a continuous map (or winding number, in dimension 1).



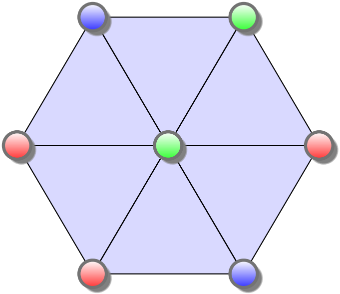
The Index Lemma

A combinatorial version of the notion of degree of a continuous map (or winding number, in dimension 1).



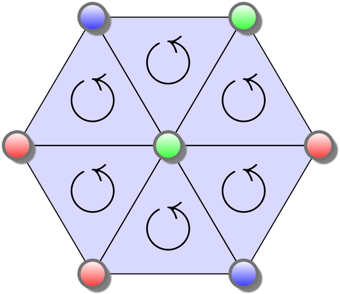
The Index Lemma

A combinatorial version of the notion of degree of a continuous map (or winding number, in dimension 1).



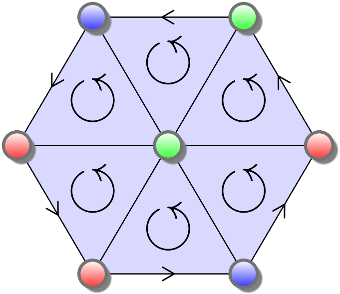
The Index Lemma

A combinatorial version of the notion of degree of a continuous map (or winding number, in dimension 1).



The Index Lemma

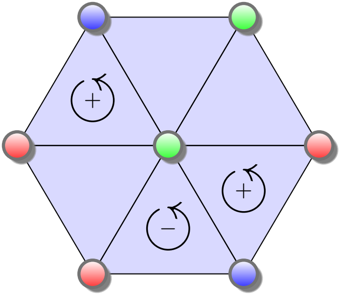
A combinatorial version of the notion of degree of a continuous map (or winding number, in dimension 1).



The Index Lemma

A combinatorial version of the notion of degree of a continuous map (or winding number, in dimension 1).

Content $= 1$

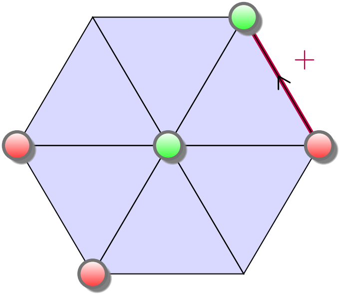


The Index Lemma

A combinatorial version of the notion of degree of a continuous map (or winding number, in dimension 1).

Content = 1

Index = 1

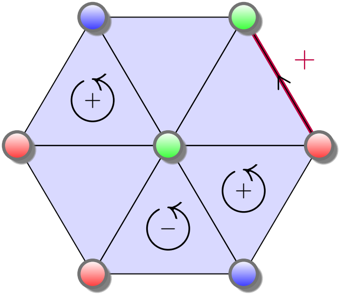


The Index Lemma

A combinatorial version of the notion of degree of a continuous map (or winding number, in dimension 1).

Content = 1

Index = 1



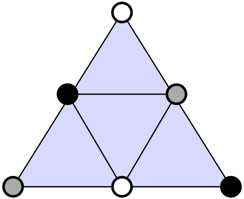
Index Lemma

In a pseudomanifold with boundary, Index $= (-1)^i$ Content.

Proof sketch for $n = 3$, $k = 2$

Back to the subcomplex H of the input complex.
We color the vertices with the value:

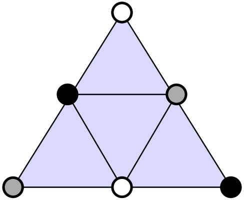
$$\text{process number} + \text{decision value} \mod n$$



Proof sketch for $n = 3$, $k = 2$

Back to the subcomplex H of the input complex.
We color the vertices with the value:

$$\text{process number} + \text{decision value} \mod n$$

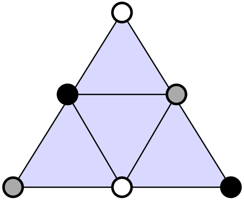


The index of H is 2.

Proof sketch for $n = 3$, $k = 2$

Back to the subcomplex H of the input complex.
We color the vertices with the value:

$$\text{process number} + \text{decision value} \mod n$$

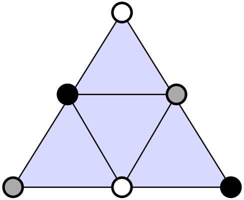


The index of H is 2. Moreover, chromatic subdividions preserve the index, so the index of $\mathrm{SubDiv(H)}$ is also 2.

Proof sketch for $n = 3$, $k = 2$

Back to the subcomplex H of the input complex.
We color the vertices with the value:

$$\text{process number} + \text{decision value} \mod n$$



The index of H is 2. Moreover, chromatic subdividions preserve the index, so the index of SubDiv(H) is also 2. By the Index lemma, the content of SubDiv(H) is ± 2. This implies that there are monochromatic triangles w.r.t. decision values. □

Conclusion

We have studied a family of tasks EN(k,ℓ), for $1 \leq k < \ell \leq n$.

- Solvable if $k + 2 \leq \ell$,
- Unsolvable if $k \leq n/2$ and $\ell = k + 1$,
- Unsolvable if $(n - k)$ is odd and $\ell = k + 1$,

Conclusion

We have studied a family of tasks EN(k,ℓ), for $1 \leq k < \ell \leq n$.

- Solvable if $k + 2 \leq \ell$,
- Unsolvable if $k \leq n/2$ and $\ell = k + 1$,
- Unsolvable if $(n - k)$ is odd and $\ell = k + 1$,
- **Open question** if $k > n/2$ and $(n - k)$ is even and $\ell = k + 1$.

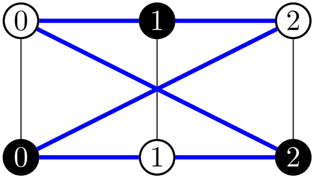
Conclusion

We have studied a family of tasks EN(k,ℓ), for $1 \leq k < \ell \leq n$.

- Solvable if $k+2 \leq \ell$,
- Unsolvable if $k \leq n/2$ and $\ell = k+1$,
- Unsolvable if $(n-k)$ is odd and $\ell = k+1$,
- **Open question** if $k > n/2$ and $(n-k)$ is even and $\ell = k+1$.

Two key ingredients for impossibility:

- The Index lemma, also used for Weak Symmetry Breaking.
- Connectedness of some subcomplex of the input.



Conclusion

We have studied a family of tasks EN(k,ℓ), for $1 \leq k < \ell \leq n$.

- Solvable if $k + 2 \leq \ell$,
- Unsolvable if $k \leq n/2$ and $\ell = k + 1$,
- Unsolvable if $(n - k)$ is odd and $\ell = k + 1$,
- **Open question** if $k > n/2$ and $(n - k)$ is even and $\ell = k + 1$.

Two key ingredients for impossibility:

- The Index lemma, also used for Weak Symmetry Breaking.
- Connectedness of some subcomplex of the input.

Thank you!