Self-Stabilizing Leader Election (SSLE) in Population Protocols over Arbitrary Communication Graphs

J. BEAUQUIER, P. BLANCHARD, J. BURMAN

Basics

Population Protocol (PP)

- Network of mobile agents
- Constant memory (e.g. no ids)
- Protocol = set of transition rules for 2 meeting agents.

Basics

Communication Graph

- node = agent
- edge (u, v) = possible meeting of u and v
- connected

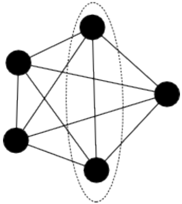
Population Protocol

- family of communication graphs $\mathcal{F}$
- state space Q, input space I
- finite set of rules : $(p, i)(q, j) \longrightarrow p', q'$

Basics

Example : state space $\{\circ, \bullet\}$, input space $\{\perp\}$

$$\bullet\bullet \longrightarrow \circ\bullet$$



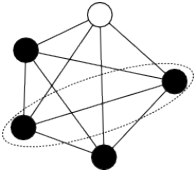
$$\underbrace{(\gamma_1, \alpha_1)}$$

configuration, input

Basics

Example : state space $\{\circ, \bullet\}$, input space $\{\perp\}$

$$\bullet\bullet \longrightarrow \circ\bullet$$

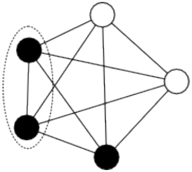


$$(\gamma_1, \alpha_1) \xrightarrow{\text{edge } e_1} (\gamma_2, \alpha_2)$$

Basics

Example : state space $\{\circ, \bullet\}$, input space $\{\perp\}$

$$\bullet\bullet \longrightarrow \circ\bullet$$

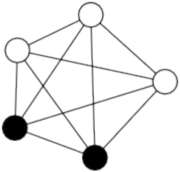


$$(\gamma_1, \alpha_1) \xrightarrow{e_1} (\gamma_2, \alpha_2) \xrightarrow{e_2} (\gamma_3, \alpha_3)$$

Basics

Example : state space $\{\circ, \bullet\}$, input space $\{\perp\}$

$$\bullet\bullet \longrightarrow \circ\bullet$$



$$(\gamma_1, \alpha_1) \xrightarrow{e_1} (\gamma_2, \alpha_2) \xrightarrow{e_2} (\gamma_3, \alpha_3) \longrightarrow \dots$$

Basics

Definition (Global Fairness)

If (γ, α) occurs infinitely often and $(\gamma, \alpha) \rightarrow \gamma'$ then γ' occurs infinitely often.

- Mobility of the agents is non-deterministic.
- As commonly in PP, we consider only globally fair executions.

SSLE

Basic Idea :

- Each agent is leader or non-leader
- Eventually, unique and static leader agent

Agents are **NOT** required to :

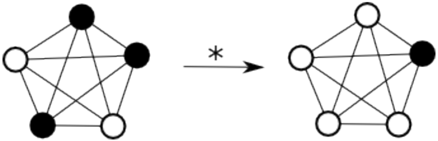
- Know when the leader is elected (no termination)
- Know who the leader is (no id)

Self-stabilization

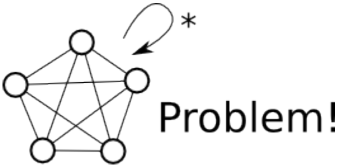
- Arbitrary initial configuration
- Convergence after a finite period of time

SSLE

On *complete* graphs, let's try ●● $\longrightarrow$ ●○



SSLE



- Previous protocol is not self-stabilizing
- Need to create leaders (e.g. ○○ ⟶ ●○)
- Need to stop creating leaders

Oracle $\Omega?$

(Angluin et al., 2005).

SSLE impossible on complete graphs.

Fischer and Jiang (distributed) oracle $\Omega?$

- For each agent x, output $\Omega?_x$ as input to x
- Constantly observe the leader bit in the configurations
- Forever 0 leader $\Rightarrow$ eventually output 0 in *some* agent
- Forever at least 1 leader $\Rightarrow$ eventually output 1 in *every* agent

N.B. : The observed bit is not necessarily interpreted as "leader".

Oracle $\Omega ?$

Main ideas for using $\Omega ?$ to solve *SSLE* :

- Leader detector.
- Create leader at x iff $\Omega ?_x = 0$.
- Each leader tries to “kill” other leaders ...
- ... without killing himself.

Results

Some previous results for *SSLE* :

- (Fischer, Jiang) solution, oracle $\Omega?$, over complete graphs
- (Fischer, Jiang) solution, oracle $\Omega?$, over all rings
- (Angluin et al.) For each k, solution, no oracle, over rings of size $\notin k\mathbb{Z}$
- (Caneda, Potop-Butucaru) impossibility of *silent* self-stabilizing leader election even with the help of $\Omega?$
- (Cai, Izumi, Wada) solution, no oracle, over a complete graph of size n using state space of size n

Results

Our results :

- Leader election over *arbitrary graphs without oracles, with uniform initialization* (non self-stabilizing)
- family $\{\Omega?[k,d]\}_{k,d\in\mathbb{N}}$ that generalizes Fischer and Jiang's oracle $\Omega?$.
- a *self-stabilizing* protocol over *arbitrary graphs using* $\Omega?[2,1] = 2$ instances of $\Omega?$.
- $\Omega?$ cannot be implemented using *SSLE* over arbitrary graphs.

Results

Our results :

- Leader election over *arbitrary graphs without oracles, with uniform initialization* (non self-stabilizing)
- family $\{\Omega?[k, d]\}_{k,d \in \mathbb{N}}$ that generalizes Fischer and Jiang's oracle $\Omega?$.
- **a *self-stabilizing* protocol over *arbitrary graphs using* $\Omega?[2, 1]$ = 2 instances of $\Omega?$.**
- $\Omega?$ cannot be implemented using *SSLE* over arbitrary graphs.

Results

Protocol $\mathcal{A}$, each agent

- is either non-leader, white leader ($\circ$), black leader ($\bullet$)
- holds either no token, white token ($\triangleright$), black token ($\blacktriangleright$)

$\Omega?[2,1]$ = two instances of $\Omega?$

- leader detector, $\Omega?^l$, observes the presence of leaders
- token detector, $\Omega?^t$, observes the presence of tokens

Results

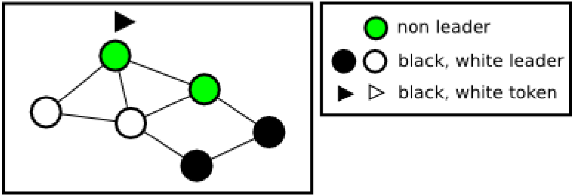
Rules of $\mathcal{A}$

- $\Omega?^l = 0$: creates a black leader
- $\Omega?^t = 0$: creates a black token
- tokens move (by swapping)
- black token kills white leader $\blacktriangleright, \circ \rightarrow \perp$
- white token whitens black leader $\triangleright, \bullet \rightarrow \circ$
- if a token meets a leader with the same color, they both flip their colors
- two colliding tokens : one disappears

Results

Why does it work ?

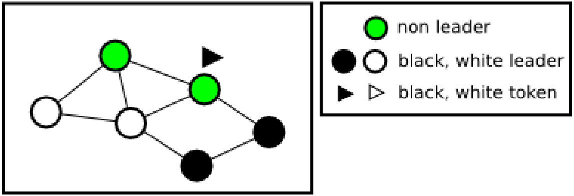
- With $\Omega?^t$: eventually a unique token (changing color)
- Synchronized pair of leader and token : same color
- Invariant : if a config contains a synchronized pair, then the next config also contains a synchronized pair



Results

Why does it work ?

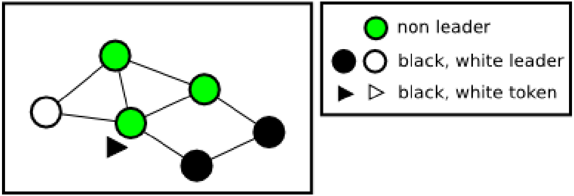
- With $\Omega?^t$: eventually a unique token (changing color)
- Synchronized pair of leader and token : same color
- Invariant : if a config contains a synchronized pair, then the next config also contains a synchronized pair



Results

Why does it work ?

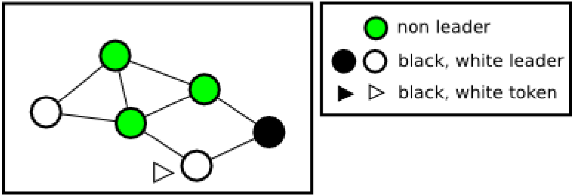
- With $\Omega?^t$: eventually a unique token (changing color)
- Synchronized pair of leader and token : same color
- Invariant : if a config contains a synchronized pair, then the next config also contains a synchronized pair



Results

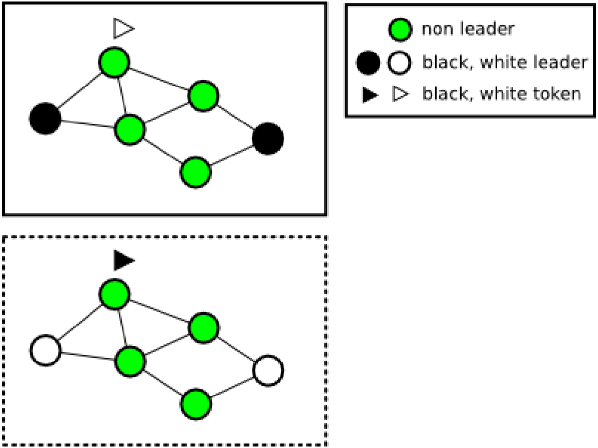
Why does it work ?

- With $\Omega?^t$: eventually a unique token (changing color)
- Synchronized pair of leader and token : same color
- Invariant : if a config contains a synchronized pair, then the next config also contains a synchronized pair



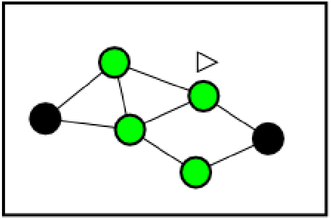
Results

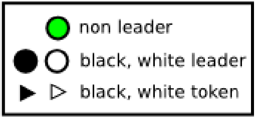
What about configuration without synchronized pair ?

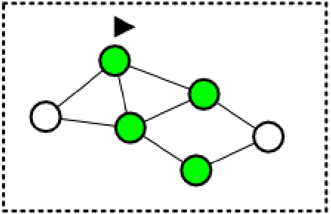


Results

What about configuration without synchronized pair ?

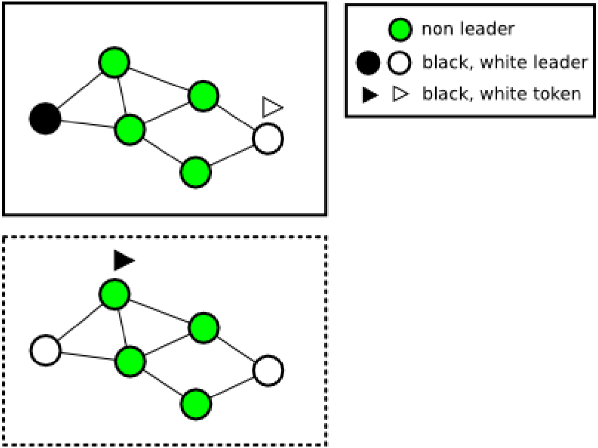






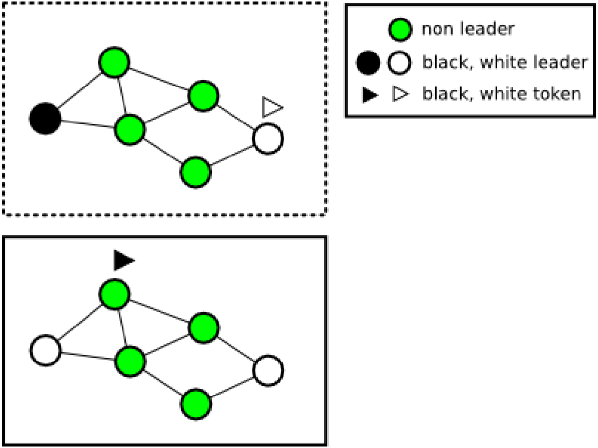
Results

What about configuration without synchronized pair ?



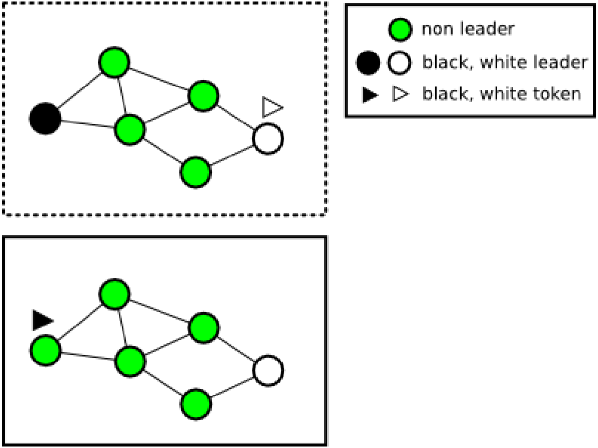
Results

What about configuration without synchronized pair ?



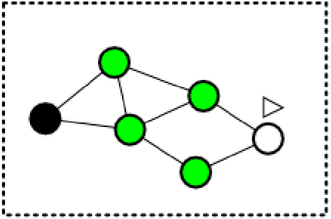
Results

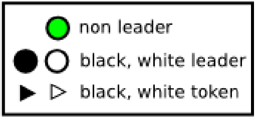
What about configuration without synchronized pair ?

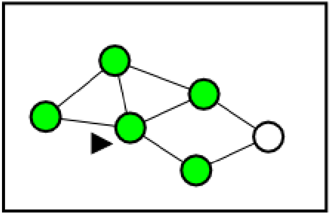


Results

What about configuration without synchronized pair ?

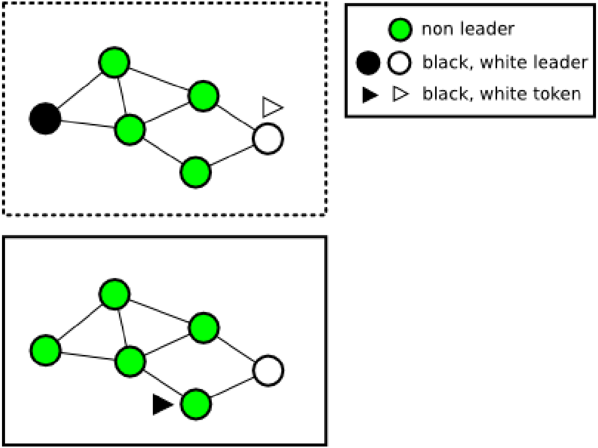






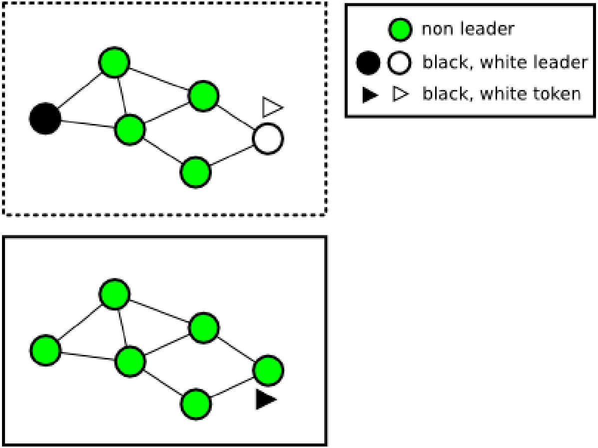
Results

What about configuration without synchronized pair ?



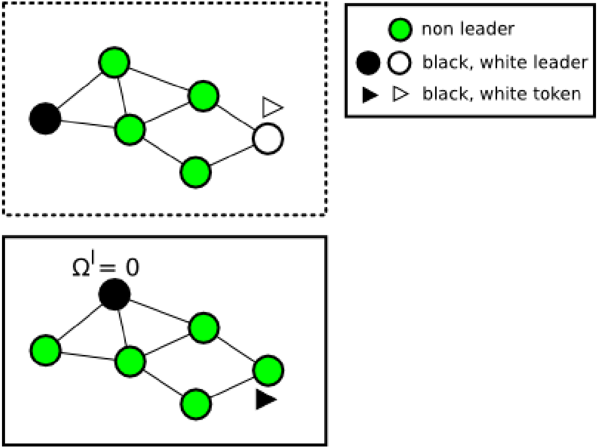
Results

What about configuration without synchronized pair ?



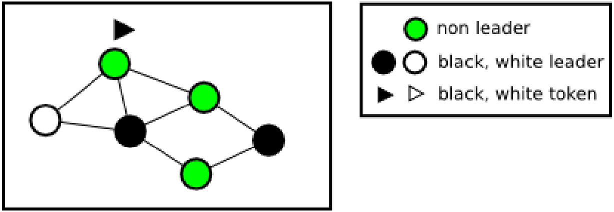
Results

What about configuration without synchronized pair ?



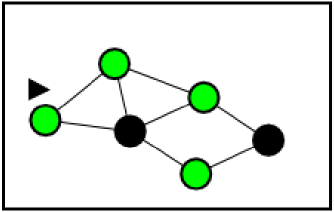
Results

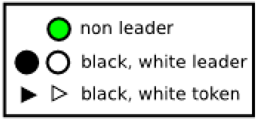
- Unique token synchronized with some leader
- The number of leaders (≥ 1) cannot increase (via $\Omega?^{l}$)
- The number of leaders eventually reaches 1 (by global fairness)



Results

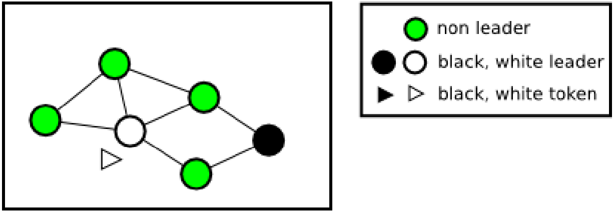
- Unique token synchronized with some leader
- The number of leaders (≥ 1) cannot increase (via $\Omega?^l$)
- The number of leaders eventually reaches 1 (by global fairness)





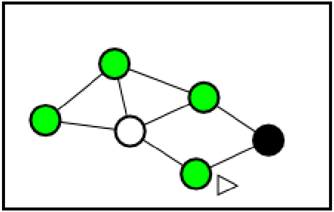
Results

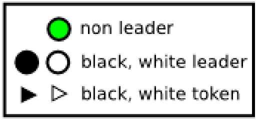
- Unique token synchronized with some leader
- The number of leaders (≥ 1) cannot increase (via $\Omega ?^{l}$)
- The number of leaders eventually reaches 1 (by global fairness)



Results

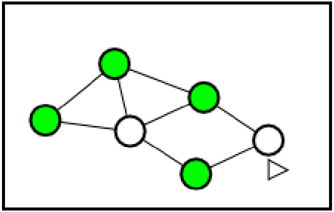
- Unique token synchronized with some leader
- The number of leaders (≥ 1) cannot increase (via $\Omega?^{l}$)
- The number of leaders eventually reaches 1 (by global fairness)

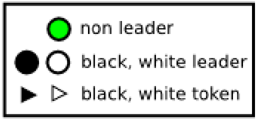




Results

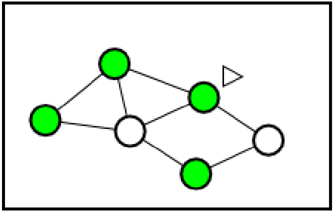
- Unique token synchronized with some leader
- The number of leaders (≥ 1) cannot increase (via $\Omega?^l$)
- The number of leaders eventually reaches 1 (by global fairness)

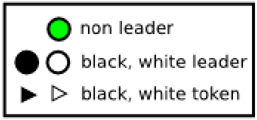




Results

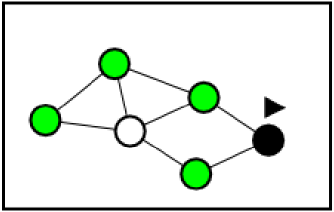
- Unique token synchronized with some leader
- The number of leaders (≥ 1) cannot increase (via $\Omega?^{l}$)
- The number of leaders eventually reaches 1 (by global fairness)

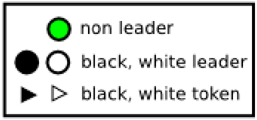




Results

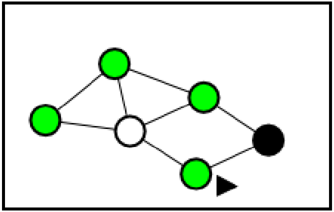
- Unique token synchronized with some leader
- The number of leaders (≥ 1) cannot increase (via $\Omega?^l$)
- The number of leaders eventually reaches 1 (by global fairness)

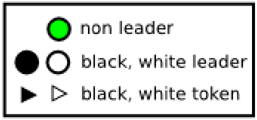




Results

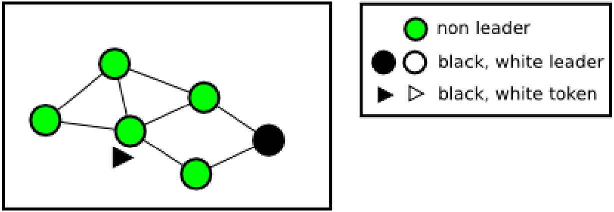
- Unique token synchronized with some leader
- The number of leaders (≥ 1) cannot increase (via $\Omega?^{l}$)
- The number of leaders eventually reaches 1 (by global fairness)





Results

- Unique token synchronized with some leader
- The number of leaders (≥ 1) cannot increase (via $\Omega?^{l}$)
- The number of leaders eventually reaches 1 (by global fairness)



Conclusion

Perspectives

- Is Fischer-Jiang $\Omega?$ enough to solve *SSLE* over arbitrary graphs ?
- Prove the conjecture : there is no solution to *SSLE*, without oracles, over the family of all rings.

Thank you.