Stably Decidable Graph Languages by Mediated Population Protocols

Othon Michail
Joint work with:
Ioannis Chatzigiannakis
Paul Spirakis

Research Academic Computer Technology Institute (RACTI)

SSS 2010
September 2010

Mediated Population Protocol Model

[CMS '09]

- The idea is the existence of a mediator with limited storage capacity
- Simplification: The communication links are constant storages

Mediated Population Protocol Model

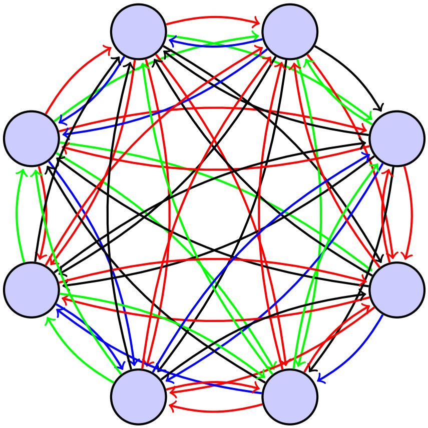
[CMS '09]

- The idea is the existence of a mediator with limited storage capacity
- Simplification: The communication links are constant storages

Mediated Population Protocol Model

[CMS '09]

- The idea is the existence of a mediator with limited storage capacity
- Simplification: The communication links are constant storages



MPP Model

A Mediated Population Protocol

- is a PP that additionally has
  - a finite set of edge states S
  - and an extended transition function

$$\delta : Q \times Q \times S \rightarrow Q \times Q \times S$$

MPP Model

A Mediated Population Protocol

- is a PP that additionally has
  - a finite set of edge states S
  - and an extended transition function

$$\delta : Q \times Q \times S \rightarrow Q \times Q \times S$$

MPP Model

A Mediated Population Protocol

- is a PP that additionally has
  - a finite set of edge states S
  - and an extended transition function

$$\delta : Q \times Q \times S \rightarrow Q \times Q \times S$$

Decidability of Graph Properties

- Here we disregard inputs to study the ability of MPPs to stably decide graph properties.
  - The corresponding model is the Graph Decision Mediated Population Protocol model (GDMPP).

Decidability of Graph Properties

- Here we disregard inputs to study the ability of MPPs to stably decide graph properties.
    - The corresponding model is the Graph Decision Mediated Population Protocol model (GDMPP).

GDMPP Model

GDMPPs

- binary output alphabet $Y = \{0, 1\}$
- set of agent states Q
- output function $O : Q \to Y$
- set of edge states S
- transition function $\delta : Q \times Q \times S \to Q \times Q \times S$
- initial agent state $q_0 \in Q$
- initial edge state $s_0 \in S$

GDMPP Model

GDMPPs

- binary output alphabet $Y = \{0, 1\}$
- set of agent states Q
- output function $O : Q \to Y$
- set of edge states S
- transition function $\delta : Q \times Q \times S \to Q \times Q \times S$
- initial agent state $q_0 \in Q$
- initial edge state $s_0 \in S$

GDMPP Model

GDMPPs

- binary output alphabet $Y = \{0, 1\}$
- set of agent states Q
- output function $O : Q \to Y$
- set of edge states S
- transition function $\delta : Q \times Q \times S \to Q \times Q \times S$
- initial agent state $q_0 \in Q$
- initial edge state $s_0 \in S$

GDMPP Model

GDMPPs

- binary output alphabet $Y = \{0, 1\}$
- set of agent states Q
- output function $O : Q \to Y$
- set of edge states S
- transition function $\delta : Q \times Q \times S \to Q \times Q \times S$
- initial agent state $q_0 \in Q$
- initial edge state $s_0 \in S$

GDMPP Model

GDMPPs

- binary output alphabet $Y = \{0, 1\}$
- set of agent states Q
- output function $O : Q \to Y$
- set of edge states S
- transition function $\delta : Q \times Q \times S \to Q \times Q \times S$
- initial agent state $q_0 \in Q$
- initial edge state $s_0 \in S$

GDMPP Model

GDMPPs

- binary output alphabet $Y = \{0, 1\}$
- set of agent states Q
- output function $O : Q \to Y$
- set of edge states S
- transition function $\delta : Q \times Q \times S \to Q \times Q \times S$
- initial agent state $q_0 \in Q$
- initial edge state $s_0 \in S$

GDMPP Model

GDMPPs

- binary output alphabet $Y = \{0, 1\}$
- set of agent states Q
- output function $O : Q \to Y$
- set of edge states S
- transition function $\delta : Q \times Q \times S \to Q \times Q \times S$
- initial agent state $q_0 \in Q$
- initial edge state $s_0 \in S$

GDMPP Model

- Initially all agents are in q_0 and all edges in s_0
- A fair adversary scheduler picks ordered pairs of agents that interact according to δ
- A graph universe $\mathcal{U}$ is a set containing all possible communication graphs on which the protocol may run
- Given some $\mathcal{U}$ a graph language L is any subset of $\mathcal{U}$
    - We are interested in those that can be described by some succinct property
- A GDMPP protocol $\mathcal{A}$ stably decides a graph language $L \subseteq \mathcal{U}$ iff for any $G \in L$ all agents eventually accept and for any $G \in \mathcal{U} - L$ all agents eventually reject
- $L \in \mathcal{U}$ is stably decidable if $\exists$ GDMPP protocol that stably decides it

GDMPP Model

- Initially all agents are in q_0 and all edges in s_0
- A fair adversary scheduler picks ordered pairs of agents that interact according to δ
- A graph universe $\mathcal{U}$ is a set containing all possible communication graphs on which the protocol may run
- Given some $\mathcal{U}$ a graph language L is any subset of $\mathcal{U}$
  - We are interested in those that can be described by some succinct property
- A GDMPP protocol $\mathcal{A}$ stably decides a graph language $L \subseteq \mathcal{U}$ iff for any $G \in L$ all agents eventually accept and for any $G \in \mathcal{U} - L$ all agents eventually reject
- $L \in \mathcal{U}$ is stably decidable if $\exists$ GDMPP protocol that stably decides it

GDMPP Model

- Initially all agents are in q_0 and all edges in s_0
- A fair adversary scheduler picks ordered pairs of agents that interact according to δ
- A graph universe $\mathcal{U}$ is a set containing all possible communication graphs on which the protocol may run
- Given some $\mathcal{U}$ a graph language L is any subset of $\mathcal{U}$
  - We are interested in those that can be described by some succinct property
- A GDMPP protocol $\mathcal{A}$ stably decides a graph language $L \subseteq \mathcal{U}$ iff for any $G \in L$ all agents eventually accept and for any $G \in \mathcal{U} - L$ all agents eventually reject
- $L \in \mathcal{U}$ is stably decidable if $\exists$ GDMPP protocol that stably decides it

GDMPP Model

- Initially all agents are in q_0 and all edges in s_0
- A fair adversary scheduler picks ordered pairs of agents that interact according to δ
- A graph universe $\mathcal{U}$ is a set containing all possible communication graphs on which the protocol may run
- Given some $\mathcal{U}$ a graph language L is any subset of $\mathcal{U}$
  - We are interested in those that can be described by some succinct property
- A GDMPP protocol $\mathcal{A}$ stably decides a graph language $L \subseteq \mathcal{U}$ iff for any $G \in L$ all agents eventually accept and for any $G \in \mathcal{U} - L$ all agents eventually reject
- $L \in \mathcal{U}$ is stably decidable if $\exists$ GDMPP protocol that stably decides it

GDMPP Model

- Initially all agents are in q_0 and all edges in s_0
- A fair adversary scheduler picks ordered pairs of agents that interact according to δ
- A graph universe $\mathcal{U}$ is a set containing all possible communication graphs on which the protocol may run
- Given some $\mathcal{U}$ a graph language L is any subset of $\mathcal{U}$
  - We are interested in those that can be described by some succinct property
- A GDMPP protocol $\mathcal{A}$ stably decides a graph language $L \subseteq \mathcal{U}$ iff for any $G \in L$ all agents eventually accept and for any $G \in \mathcal{U} - L$ all agents eventually reject
- $L \in \mathcal{U}$ is stably decidable if $\exists$ GDMPP protocol that stably decides it

GDMPP Model

- Initially all agents are in q_0 and all edges in s_0
- A fair adversary scheduler picks ordered pairs of agents that interact according to δ
- A graph universe $\mathcal{U}$ is a set containing all possible communication graphs on which the protocol may run
- Given some $\mathcal{U}$ a graph language L is any subset of $\mathcal{U}$
  - We are interested in those that can be described by some succinct property
- A GDMPP protocol $\mathcal{A}$ stably decides a graph language $L \subseteq \mathcal{U}$ iff for any $G \in L$ all agents eventually accept and for any $G \in \mathcal{U} - L$ all agents eventually reject
- $L \in \mathcal{U}$ is stably decidable if $\exists$ GDMPP protocol that stably decides it

GDMPP Model

- Initially all agents are in q_0 and all edges in s_0
- A fair adversary scheduler picks ordered pairs of agents that interact according to δ
- A graph universe $\mathcal{U}$ is a set containing all possible communication graphs on which the protocol may run
- Given some $\mathcal{U}$ a graph language L is any subset of $\mathcal{U}$
  - We are interested in those that can be described by some succinct property
- A GDMPP protocol $\mathcal{A}$ stably decides a graph language $L \subseteq \mathcal{U}$ iff for any $G \in L$ all agents eventually accept and for any $G \in \mathcal{U} - L$ all agents eventually reject
- $L \in \mathcal{U}$ is stably decidable if $\exists$ GDMPP protocol that stably decides it

Basic Question

What graph properties are stably decidable by the GDMPP model?

Closure results

Theorem

The class of stably decidable graph languages is closed under

- *Complement*
- *Union*
- *Intersection*

Closure results

Theorem

The class of stably decidable graph languages is closed under

- *Complement*
- *Union*
- *Intersection*

Closure results

Theorem

The class of stably decidable graph languages is closed under

- *Complement*
- *Union*
- *Intersection*

Weakly Connected Graphs

Let $\mathcal{G}$ be the graph universe consisting of all directed and weakly connected communication graphs

Decidability

Some examples of stably decidable graph languages:

- Node Parity
- Edge Parity
- All nodes have less than $k = \mathcal{O}(1)$ outgoing neighbors (bounded out-degree)
- Some node has more incoming than outgoing neighbors
- G has some directed path of length at least $k = \mathcal{O}(1)$

Decidability

Some examples of stably decidable graph languages:

- Node Parity
- Edge Parity
- All nodes have less than $k = \mathcal{O}(1)$ outgoing neighbors (bounded out-degree)
- Some node has more incoming than outgoing neighbors
- G has some directed path of length at least $k = \mathcal{O}(1)$

Decidability

Some examples of stably decidable graph languages:

- Node Parity
- Edge Parity
- All nodes have less than $k = \mathcal{O}(1)$ outgoing neighbors (bounded out-degree)
- Some node has more incoming than outgoing neighbors
- G has some directed path of length at least $k = \mathcal{O}(1)$

Decidability

Some examples of stably decidable graph languages:

- Node Parity
- Edge Parity
- All nodes have less than $k = \mathcal{O}(1)$ outgoing neighbors (bounded out-degree)
- Some node has more incoming than outgoing neighbors
- G has some directed path of length at least $k = \mathcal{O}(1)$

Decidability

Some examples of stably decidable graph languages:

- Node Parity
- Edge Parity
- All nodes have less than $k = \mathcal{O}(1)$ outgoing neighbors (bounded out-degree)
- Some node has more incoming than outgoing neighbors
- G has some directed path of length at least $k = \mathcal{O}(1)$

Decidability

Some examples of stably decidable graph languages:

- Node Parity
- Edge Parity
- All nodes have less than $k = \mathcal{O}(1)$ outgoing neighbors (bounded out-degree)
- Some node has more incoming than outgoing neighbors
- G has some directed path of length at least $k = \mathcal{O}(1)$

An Impossibility Result

- $2C = \{G \in \mathcal{G} \mid G$ has at least two nodes u, υ s.t. both $(u, \upsilon), (\upsilon, u) \in E(G)\}$ (in other words, G has at least one 2-cycle)

Theorem

$2C$ is not stably decidable by GDMPPs with stabilizing states.

An Impossibility Result

- $2C = \{G \in \mathcal{G} \mid G$ has at least two nodes u, υ s.t. both $(u, \upsilon), (\upsilon, u) \in E(G)\}$ (in other words, G has at least one 2-cycle)

Theorem

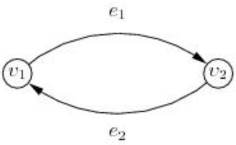
$2C$ is not stably decidable by GDMPPs with stabilizing states.

An Impossibility Result

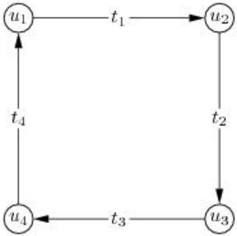
- $2C = \{G \in \mathcal{G} \mid G$ has at least two nodes u, υ s.t. both $(u, \upsilon), (\upsilon, u) \in E(G)\}$.

Theorem

$2C$ is not stably decidable by GDMPPs with stabilizing states.



(a) Graph G



(b) Graph G'

Figure: $G \in 2C$, $G' \notin 2C$.

An Impossibility Result

Lemma

For any GDMPP $\mathcal{A}$ and any computation (infinite fair execution) $C_0, C_1, C_2, \ldots$ of $\mathcal{A}$ on G (Figure 1(a)) there exists a computation $C'_0, C'_1, C'_2, \ldots, C'_i, \ldots$ of $\mathcal{A}$ on G' (Figure 1(b)) s.t.

$$\begin{aligned} C_i(\upsilon_1) &= C'_{2i}(u_1) = C'_{2i}(u_3) \\ C_i(\upsilon_2) &= C'_{2i}(u_2) = C'_{2i}(u_4) \\ C_i(e_1) &= C'_{2i}(t_1) = C'_{2i}(t_3) \\ C_i(e_2) &= C'_{2i}(t_2) = C'_{2i}(t_4) \end{aligned}$$

for any finite $i \geq 0$.

Proof: The proof is by induction on i

An Impossibility Result

Theorem

$2C$ is not stably decidable by GDMPPs with stabilizing states.

Proof

- Assume that a GDMPP $\mathcal{A}$ stably decides $2C$ with stabilizing states
- When $\mathcal{A}$ runs in a fair manner on G, after finitely many steps all agents output the value 1 (i.e. $\mathcal{A}$ accepts G)
- But according to the previous Lemma there exists some unfair execution of $\mathcal{A}$ on G' simulating that of $\mathcal{A}$ on G

An Impossibility Result

Theorem

$2C$ is not stably decidable by GDMPPs with stabilizing states.

Proof

- Assume that a GDMPP $\mathcal{A}$ stably decides $2C$ with stabilizing states
- When $\mathcal{A}$ runs in a fair manner on G, after finitely many steps all agents output the value 1 (i.e. $\mathcal{A}$ accepts G)
- But according to the previous Lemma there exists some unfair execution of $\mathcal{A}$ on G' simulating that of $\mathcal{A}$ on G

An Impossibility Result

Theorem

$2C$ is not stably decidable by GDMPPs with stabilizing states.

Proof

- Assume that a GDMPP $\mathcal{A}$ stably decides $2C$ with stabilizing states
- When $\mathcal{A}$ runs in a fair manner on G, after finitely many steps all agents output the value 1 (i.e. $\mathcal{A}$ accepts G)
- But according to the previous Lemma there exists some unfair execution of $\mathcal{A}$ on G' simulating that of $\mathcal{A}$ on G

An Impossibility Result

- Since the states of $\mathcal{A}$ on G have stabilized there exists no transition to fix the wrong decision that $\mathcal{A}$ has made on G'
- If we allow now the scheduler on G' to decome fair we have a fair execution that also accepts G'
- But this is a contradiction
  - $\mathcal{A}$ stably decides $2C$ so it cannot accept $G' \notin 2C$

$\square$

An Impossibility Result

- Since the states of $\mathcal{A}$ on G have stabilized there exists no transition to fix the wrong decision that $\mathcal{A}$ has made on G'
- If we allow now the scheduler on G' to decome fair we have a fair execution that also accepts G'
- But this is a contradiction
  - $\mathcal{A}$ stably decides $2C$ so it cannot accept $G' \notin 2C$

$\square$

An Impossibility Result

- Since the states of $\mathcal{A}$ on G have stabilized there exists no transition to fix the wrong decision that $\mathcal{A}$ has made on G'
- If we allow now the scheduler on G' to decome fair we have a fair execution that also accepts G'
- But this is a contradiction
  - $\mathcal{A}$ stably decides $2C$ so it cannot accept $G' \notin 2C$

$\square$

An Impossibility Result

- Since the states of $\mathcal{A}$ on G have stabilized there exists no transition to fix the wrong decision that $\mathcal{A}$ has made on G'
- If we allow now the scheduler on G' to decome fair we have a fair execution that also accepts G'
- But this is a contradiction
  - $\mathcal{A}$ stably decides $2C$ so it cannot accept $G' \notin 2C$

$\square$

Allowing Disconnected Graphs

- Here the universe is $\mathcal{H}$ containing all directed graphs
    - also those that are disconnected

Theorem (General Impossibility Result)

Any nontrivial graph language $L \subset \mathcal{H}$ is not stably decidable.

Nontrivial: $L \neq \emptyset$ and $L \neq \mathcal{H}$

Allowing Disconnected Graphs

- Here the universe is $\mathcal{H}$ containing all directed graphs
  - also those that are disconnected

Theorem (General Impossibility Result)

Any nontrivial graph language $L \subset \mathcal{H}$ is not stably decidable.

Nontrivial: $L \neq \emptyset$ and $L \neq \mathcal{H}$

Allowing Disconnected Graphs

- Here the universe is $\mathcal{H}$ containing all directed graphs
  - also those that are disconnected

Theorem (General Impossibility Result)

Any nontrivial graph language $L \subset \mathcal{H}$ *is not stably decidable.*

Nontrivial: $L \neq \emptyset$ and $L \neq \mathcal{H}$

Allowing Disconnected Graphs

- Here the universe is $\mathcal{H}$ containing all directed graphs
  - also those that are disconnected

Theorem (General Impossibility Result)

Any nontrivial graph language $L \subset \mathcal{H}$ is not stably decidable.

Nontrivial: $L \neq \emptyset$ and $L \neq \mathcal{H}$

A General Impossibility Result

Lemma

For any nontrivial graph language L

- $\exists$ *disconnected graph $G \in L$ where at least one component of G does not belong to L, or*
- $\exists$ *disconnected graph $G' \in \overline{L}$ where at least one component of G' does not belong to $\overline{L}$*
- *or both*

A General Impossibility Result

Lemma

For any nontrivial graph language L

- $\exists$ *disconnected graph* $G \in L$ *where at least one component of* G *does not belong to* L*, or*
- $\exists$ *disconnected graph* $G' \in \overline{L}$ *where at least one component of* G' *does not belong to* $\overline{L}$
- *or both*

A General Impossibility Result

Lemma

For any nontrivial graph language L

- $\exists$ *disconnected graph* $G \in L$ *where at least one component of* G *does not belong to* L*, or*
- $\exists$ *disconnected graph* $G' \in \overline{L}$ *where at least one component of* G' *does not belong to* $\overline{L}$
- *or both*

A General Impossibility Result

Lemma

For any nontrivial graph language L

- $\exists$ *disconnected graph* $G \in L$ *where at least one component of* G *does not belong to* L*, or*
- $\exists$ *disconnected graph* $G' \in \overline{L}$ *where at least one component of* G' *does not belong to* $\overline{L}$
- *or both*

A General Impossibility Result

Proof.

If the statement does not hold: Any disconnected graph in L has all its components in L and any disconnected graph in $\overline{L}$ has all its components in $\overline{L}$.

1. All connected graphs belong to L. Then $\overline{L}$ contains at least one disconnected graph (since it is nontrivial) that has all its components in L (contradiction)
2. All connected graphs belong to $\overline{L}$ (contradiction by symmetry)
3. L and $\overline{L}$ contain connected graphs G and G', respectively. Their disjoint union $U = (V \cup V', E \cup E')$ is disconnected, belongs to L or $\overline{L}$ but one of its components belongs to L and the other to $\overline{L}$ (contradiction - by assumption both components should belong to the same language)

□

A General Impossibility Result

Proof.

If the statement does not hold: Any disconnected graph in L has all its components in L and any disconnected graph in $\overline{L}$ has all its components in $\overline{L}$.

1. All connected graphs belong to L. Then $\overline{L}$ contains at least one disconnected graph (since it is nontrivial) that has all its components in L (contradiction)
2. All connected graphs belong to $\overline{L}$ (contradiction by symmetry)
3. L and $\overline{L}$ contain connected graphs G and G', respectively. Their disjoint union $U = (V \cup V', E \cup E')$ is disconnected, belongs to L or $\overline{L}$ but one of its components belongs to L and the other to $\overline{L}$ (contradiction - by assumption both components should belong to the same language)

□

A General Impossibility Result

Proof.

If the statement does not hold: Any disconnected graph in L has all its components in L and any disconnected graph in $\overline{L}$ has all its components in $\overline{L}$.

1. All connected graphs belong to L. Then $\overline{L}$ contains at least one disconnected graph (since it is nontrivial) that has all its components in L (contradiction)
2. All connected graphs belong to $\overline{L}$ (contradiction by symmetry)
3. L and $\overline{L}$ contain connected graphs G and G', respectively. Their disjoint union $U = (V \cup V', E \cup E')$ is disconnected, belongs to L or $\overline{L}$ but one of its components belongs to L and the other to $\overline{L}$ (contradiction - by assumption both components should belong to the same language)

□

A General Impossibility Result

Proof.

If the statement does not hold: Any disconnected graph in L has all its components in L and any disconnected graph in $\overline{L}$ has all its components in $\overline{L}$.

1. All connected graphs belong to L. Then $\overline{L}$ contains at least one disconnected graph (since it is nontrivial) that has all its components in L (contradiction)
2. All connected graphs belong to $\overline{L}$ (contradiction by symmetry)
3. L and $\overline{L}$ contain connected graphs G and G', respectively. Their disjoint union $U = (V \cup V', E \cup E')$ is disconnected, belongs to L or $\overline{L}$ but one of its components belongs to L and the other to $\overline{L}$ (contradiction - by assumption both components should belong to the same language)

□

A General Impossibility Result

Theorem

Any nontrivial graph language $L \subset \mathcal{H}$ is not stably decidable.

Proof

Assume that GDMPP $\mathcal{A}$ stably decides a nontrivial graph language L

- Closure under complement implies that $\exists$ GDMPP $\mathcal{B}$ stably deciding $\overline{L}$
- By previous Lemma, $\exists$ disconnected G in L with some component in $\overline{L}$
- or in $\overline{L}$ with some component in L

A General Impossibility Result

Theorem

Any nontrivial graph language $L \subset \mathcal{H}$ is not stably decidable.

Proof

Assume that GDMPP $\mathcal{A}$ stably decides a nontrivial graph language L

- Closure under complement implies that $\exists$ GDMPP $\mathcal{B}$ stably deciding $\overline{L}$
- By previous Lemma, $\exists$ disconnected G in L with some component in $\overline{L}$
- or in $\overline{L}$ with some component in L

A General Impossibility Result

Theorem

Any nontrivial graph language $L \subset \mathcal{H}$ is not stably decidable.

Proof

Assume that GDMPP $\mathcal{A}$ stably decides a nontrivial graph language L

- Closure under complement implies that $\exists$ GDMPP $\mathcal{B}$ stably deciding $\overline{L}$
- By previous Lemma, $\exists$ disconnected G in L with some component in $\overline{L}$
- or in $\overline{L}$ with some component in L

A General Impossibility Result

Theorem

Any nontrivial graph language $L \subset \mathcal{H}$ is not stably decidable.

Proof

Assume that GDMPP $\mathcal{A}$ stably decides a nontrivial graph language L

- Closure under complement implies that $\exists$ GDMPP $\mathcal{B}$ stably deciding $\overline{L}$
- By previous Lemma, $\exists$ disconnected G in L with some component in $\overline{L}$
- or in $\overline{L}$ with some component in L

A General Impossibility Result

1. L contains such a G
   - Since $\mathcal{A}$ stably decides L all agents of G should eventually answer accept
   - But $\mathcal{A}$ runs on a component that belongs to $\overline{L}$ whose agents cannot communicate with the other components
   - Thus $\mathcal{A}$ answers reject in this component (contradiction)
2. $\overline{L}$ contains such a G (contradiction by symmetry)

$\square$

A General Impossibility Result

1. L contains such a G
   - Since $\mathcal{A}$ stably decides L all agents of G should eventually answer accept
   - But $\mathcal{A}$ runs on a component that belongs to $\overline{L}$ whose agents cannot communicate with the other components
   - Thus $\mathcal{A}$ answers reject in this component (contradiction)
2. $\overline{L}$ contains such a G (contradiction by symmetry)

$\square$

A General Impossibility Result

1. L contains such a G
   - Since $\mathcal{A}$ stably decides L all agents of G should eventually answer accept
   - But $\mathcal{A}$ runs on a component that belongs to $\overline{L}$ whose agents cannot communicate with the other components
   - Thus $\mathcal{A}$ answers reject in this component (contradiction)
2. $\overline{L}$ contains such a G (contradiction by symmetry)

$\square$

A General Impossibility Result

1. L contains such a G
   - Since $\mathcal{A}$ stably decides L all agents of G should eventually answer accept
   - But $\mathcal{A}$ runs on a component that belongs to $\overline{L}$ whose agents cannot communicate with the other components
   - Thus $\mathcal{A}$ answers reject in this component (contradiction)
2. $\overline{L}$ contains such a G (contradiction by symmetry)

$\square$

A General Impossibility Result

1. L contains such a G
   - Since $\mathcal{A}$ stably decides L all agents of G should eventually answer accept
   - But $\mathcal{A}$ runs on a component that belongs to $\overline{L}$ whose agents cannot communicate with the other components
   - Thus $\mathcal{A}$ answers reject in this component (contradiction)
2. $\overline{L}$ contains such a G (contradiction by symmetry)

$\square$

An Immediate Consequence

Corollary

Connectivity is not stably decidable.

Open Problems

- An exact characterization of the class of stably decidable graph languages
- An alternative: A general method for impossibility results that suits the GDMPP model
  - Ad-hoc proofs require a lot of effort
  - e.g. Herlihy's method based on simplicial complexes
- In real-life aplications the probability distribution of interactions may change
  - This can cause performance and correctness problems
  - Can we make our protocols adapt to such changes?

Open Problems

- An exact characterization of the class of stably decidable graph languages
- An alternative: A general method for impossibility results that suits the GDMPP model
  - Ad-hoc proofs require a lot of effort
  - e.g. Herlihy's method based on simplicial complexes
- In real-life aplications the probability distribution of interactions may change
  - This can cause performance and correctness problems
  - Can we make our protocols adapt to such changes?

Open Problems

- An exact characterization of the class of stably decidable graph languages
- An alternative: A general method for impossibility results that suits the GDMPP model
  - Ad-hoc proofs require a lot of effort
  - e.g. Herlihy's method based on simplicial complexes
- In real-life aplications the probability distribution of interactions may change
  - This can cause performance and correctness problems
  - Can we make our protocols adapt to such changes?

Open Problems

- An exact characterization of the class of stably decidable graph languages
- An alternative: A general method for impossibility results that suits the GDMPP model
  - Ad-hoc proofs require a lot of effort
  - e.g. Herlihy's method based on simplicial complexes
- In real-life aplications the probability distribution of interactions may change
  - This can cause performance and correctness problems
  - Can we make our protocols adapt to such changes?

Open Problems

- An exact characterization of the class of stably decidable graph languages
- An alternative: A general method for impossibility results that suits the GDMPP model
  - Ad-hoc proofs require a lot of effort
  - e.g. Herlihy's method based on simplicial complexes
- In real-life aplications the probability distribution of interactions may change
  - This can cause performance and correctness problems
  - Can we make our protocols adapt to such changes?

Open Problems

- An exact characterization of the class of stably decidable graph languages
- An alternative: A general method for impossibility results that suits the GDMPP model
  - Ad-hoc proofs require a lot of effort
  - e.g. Herlihy's method based on simplicial complexes
- In real-life aplications the probability distribution of interactions may change
  - This can cause performance and correctness problems
  - Can we make our protocols adapt to such changes?

Open Problems

- An exact characterization of the class of stably decidable graph languages
- An alternative: A general method for impossibility results that suits the GDMPP model
  - Ad-hoc proofs require a lot of effort
  - e.g. Herlihy's method based on simplicial complexes
- In real-life aplications the probability distribution of interactions may change
  - This can cause performance and correctness problems
  - Can we make our protocols adapt to such changes?

Open Problems

- The GDMPP model that aditionally assumes a unique leader seems to be stronger (no proof exists)
- The Passively mobile Machines (PM) model [CMNPS '10] extends the PP model so that each agent
  - is a multitape Turing machine
  - has unbounded memory (and define space bounded computations)
- What is the class of decidable graph languages by the PM model for each space bound?

Open Problems

- The GDMPP model that aditionally assumes a unique leader seems to be stronger (no proof exists)
- The Passively mobile Machines (PM) model [CMNPS '10] extends the PP model so that each agent
  - is a multitape Turing machine
  - has unbounded memory (and define space bounded computations)
- What is the class of decidable graph languages by the PM model for each space bound?

Open Problems

- The GDMPP model that aditionally assumes a unique leader seems to be stronger (no proof exists)
- The Passively mobile Machines (PM) model [CMNPS '10] extends the PP model so that each agent
  - is a multitape Turing machine
  - has unbounded memory (and define space bounded computations)
- What is the class of decidable graph languages by the PM model for each space bound?

Open Problems

- The GDMPP model that aditionally assumes a unique leader seems to be stronger (no proof exists)
- The Passively mobile Machines (PM) model [CMNPS '10] extends the PP model so that each agent
    - is a multitape Turing machine
    - has unbounded memory (and define space bounded computations)
- What is the class of decidable graph languages by the PM model for each space bound?

Open Problems

- The GDMPP model that aditionally assumes a unique leader seems to be stronger (no proof exists)
- The Passively mobile Machines (PM) model [CMNPS '10] extends the PP model so that each agent
  - is a multitape Turing machine
  - has unbounded memory (and define space bounded computations)
- What is the class of decidable graph languages by the PM model for each space bound?

FRONTS

- This work has been partially supported by the ICT Programme of the European Union under contract number ICT-2008-215270 (FRONTS).
- FRONTS is a joint effort of eleven academic and research institutes in foundational algorithmic research in Europe.
- The effort is towards establishing the foundations of adaptive networked societies of tiny artefacts.

Thank You!