Decidability Classes for Mobile Agents Computing

Pierre Fraigniaud
CNRS and University Paris Diderot

GRASTA-MAC 2015

October 19-23, 2015 - Montréal

*Joint work with Andrzej Pelc, Université du Québec en Outaouais, Canada

Algorithmic achievements in mobile computing

Many algorithms for
'construction/coordination' tasks:

- rendezvous
- exploration
- intruder detection/search/capture
- fault-tolerance (byzantine agents)
- 'black-hole' search
- etc.

but…

Verification

1. Designing a program together with its proof
2. Verifying a given program a posteriori
3. Verifying the execution at runtime:

Runtime verification

Results I would love to see in the context of mobile computing

Theorem (Naor&Stockmeyer, 1995).
If there exists a distributed ***randomized*** <u>construction</u> algorithm for $\mathcal{L}$ running in $O(1)$ rounds, then there exists a distributed ***deterministic*** <u>construction</u> algorithm for $\mathcal{L}$ running in $O(1)$ rounds.

******* Require $\mathcal{L} \in \mathrm{LD}$ to be locally decidable! *******

Is the system satisfying predicate P?

Construction vs. Decision

Language: $\mathcal{L}$ = {w ∈ {0,1}* satisfying predicate P}

Construction: Given x, compute y s.t. (x,y) ∈ $\mathcal{L}$

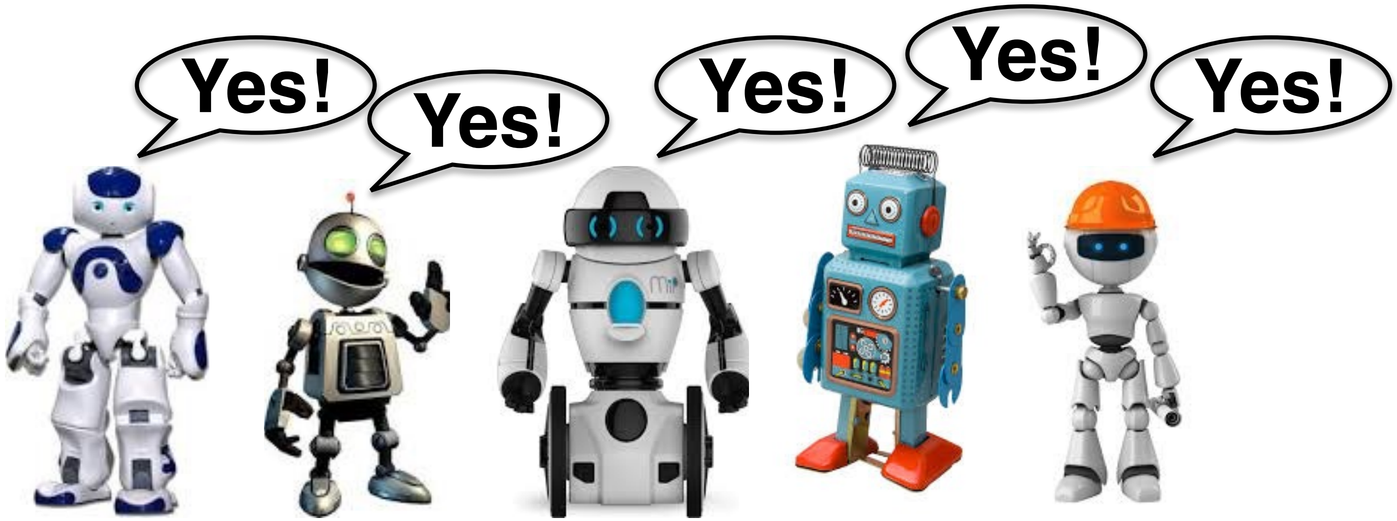
Decision: Given x, decide whether x ∈ $\mathcal{L}$ (yes/no)

Applications:

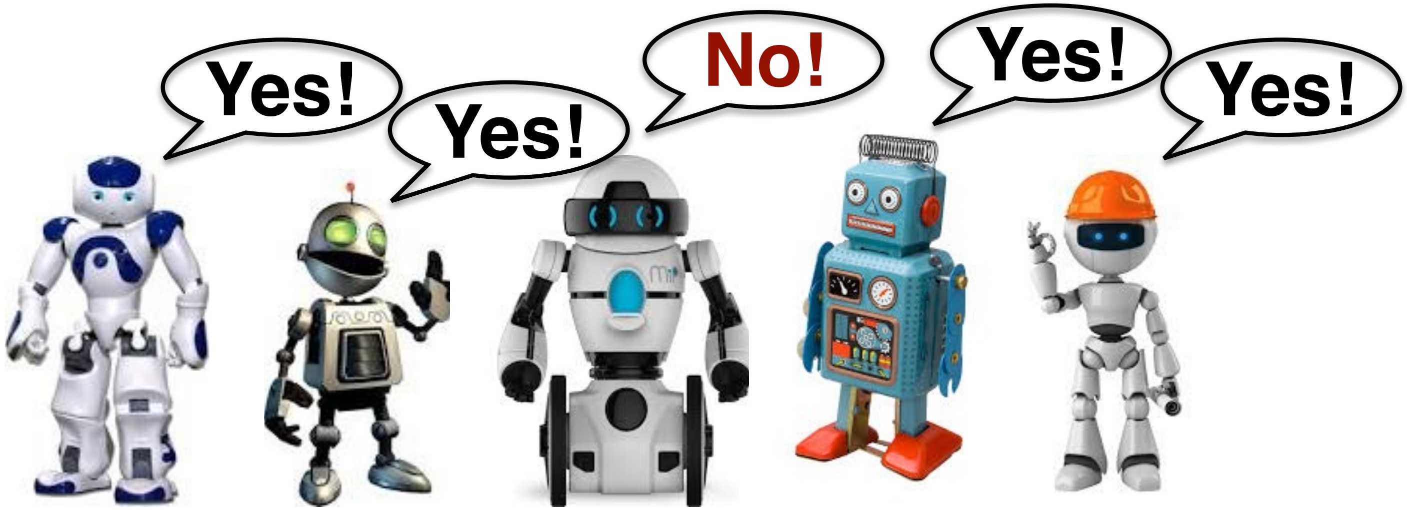
- Self-reducibility for NPC languages in sequential computing
- Derandomization theorems in distributed computing
- Monitoring (distributed) systems

Distributed Decision Rules

✔



✖

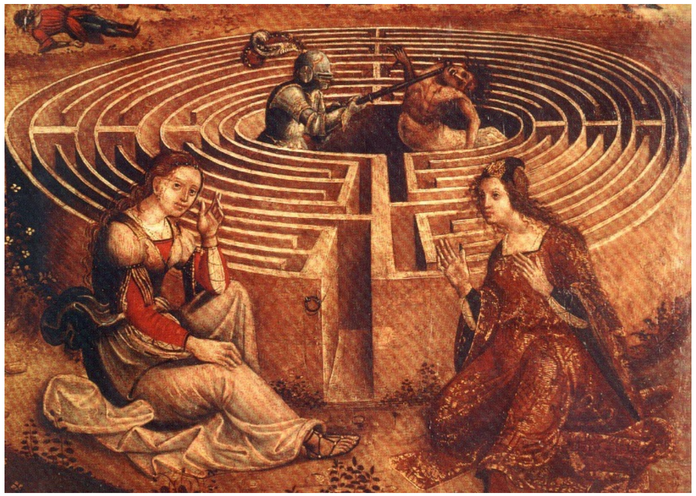


Decision tasks

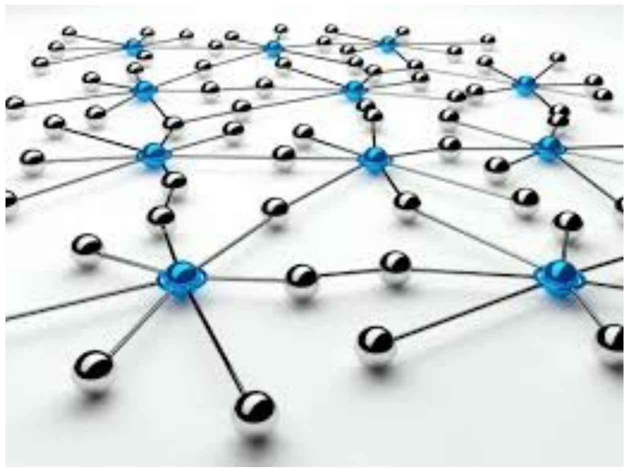
Is there an intruder
in this building?

Is this network
planar?

Is there an exit
in this labyrinth?



Network monitoring



Decision classes (computability)

Configuration: $C = (G,S,x)$ with $S \subseteq V(G)$ and $x: S \longrightarrow \{0,1\}^*$

Language: $\mathcal{L}$ = { configurations }

MAD = **M**obile **A**gent **D**ecision

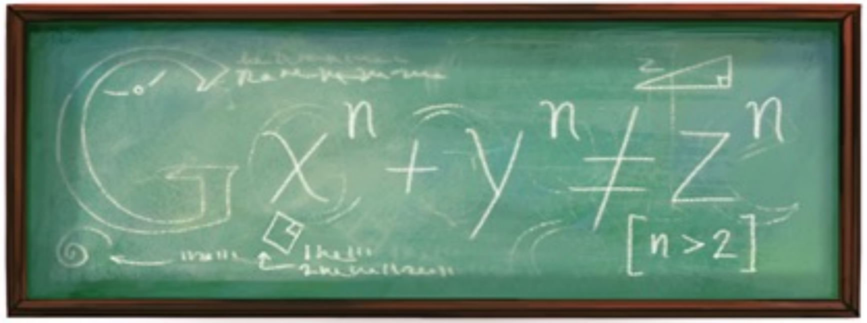
- MAD = { $\mathcal{L}$ | $\exists$ mobile agent algorithm A deciding $\mathcal{L}$ }
- A decide $\mathcal{L}$ if and only if, for every configuration C:

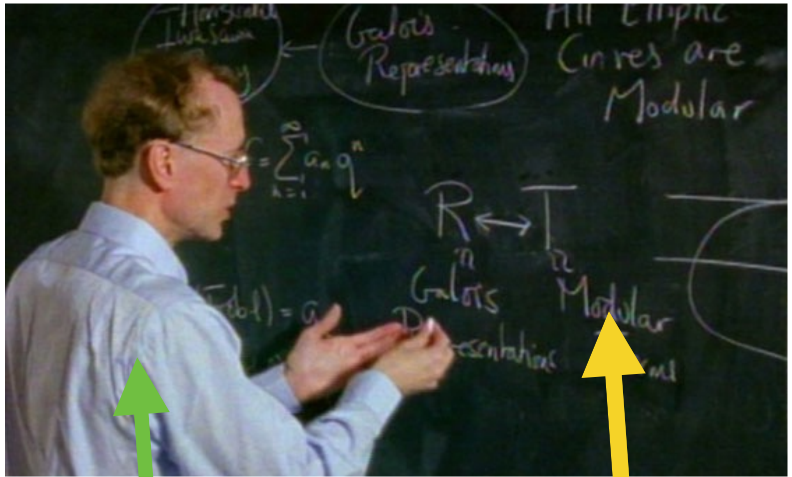
$C \in \mathcal{L} \Leftrightarrow$ every agent outputs yes

Deciding vs. verifying

Wiles' proof

Fermat's conjecture





Decide

Verify

Oracle

Certificate
or Proof

P vs. NP

NP = Non-deterministic Polynomial

$\mathcal{L} \in$ NP iff there is a poly-time algorithm A such that:

- $x \in \mathcal{L} \Rightarrow \exists c$, A(x,c) accepts
- $x \notin \mathcal{L} \Rightarrow \forall c$, A(x,c) rejects

c is the certificate, or the proof.

MAD vs. MAV

MAV = **M**obile **A**gent **V**erification

$\mathcal{L} \in$ MAV iff there is a mobile agent algorithm A such that:

- $(G,x) \in \mathcal{L} \Rightarrow \exists c$, $A(G,S,x,c)$ leads all agents to accept
- $(G,x) \notin \mathcal{L} \Rightarrow \forall c$, $A(G,S,x,c)$ leads at least one agent to reject

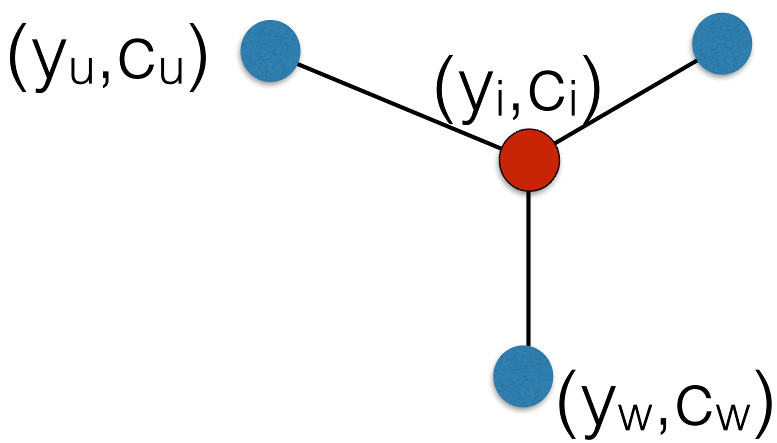
$c: S \longrightarrow \{0,1\}^*$ is the certificate, or the proof

A is a verifier, while the certificates are given by a prover.

Applications

- Composition of algorithms

- Termination (e.g., in self-stabilization)



Oracles

$C^{\mathcal{L}}$ = class C with an oracle for language $\mathcal{L}$

Example:

- P^{SAT} = poly-time with TM using an oracle for SAT.
- Extend to $C^{X} = \cup_{\mathcal{L} \in X} C^{\mathcal{L}}$

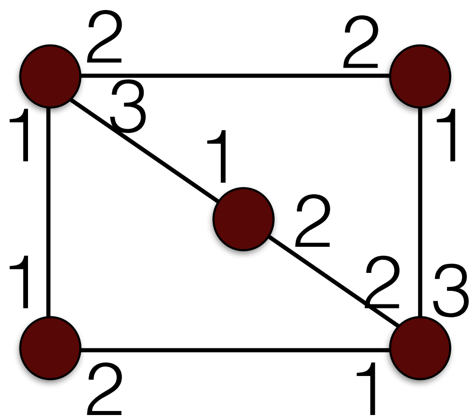
Typical oracles for MAD and MAV:

#nodes #agents upper-bounds on n,k,…

A Scenario of Application

Synchronous Mobile Agents in Anonymous Networks

Network:



Agents:

+

Mobile TM

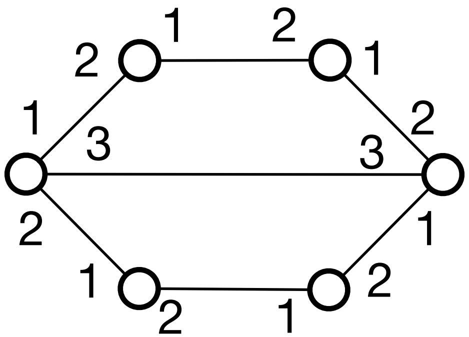
Communication whenever at the same node

MAD vs. MAV & co-MAV

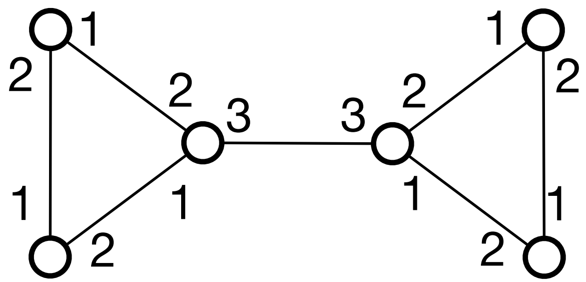
- `treesize` $\in$ MAD (perform DFS for $2(n-1)$ steps)
- `tree` $\notin$ MAD (even `path` $\notin$ MAD_1)
- `tree` $\in$ MAV (certificate = n)
- `nontree` $\in$ co-MAV

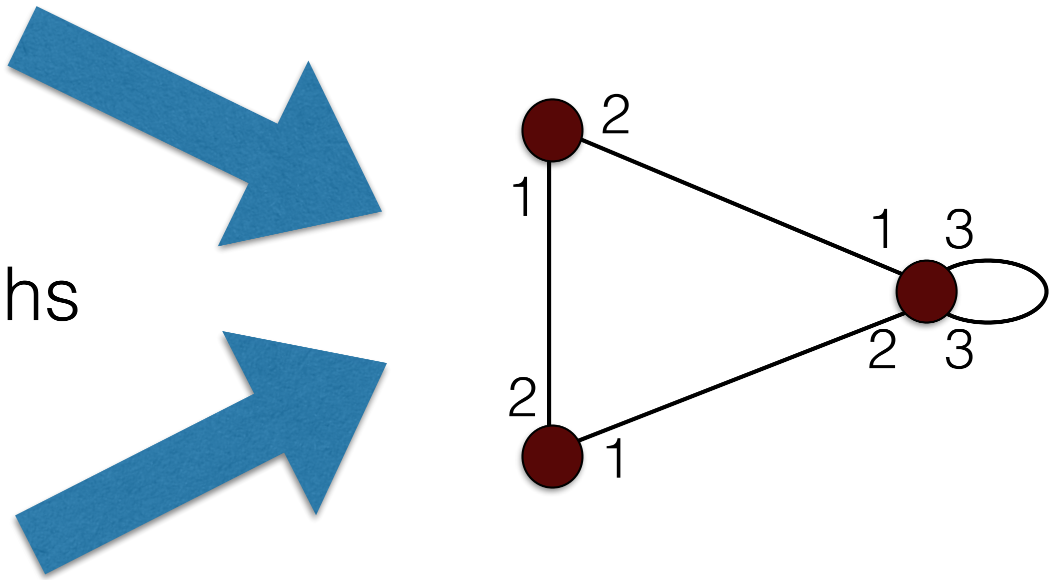
Views and Quotient

quotient(G) = G/view



Non Isomorphic Graphs





Same Quotient

Two Central Languages (i.e., Tasks)

- quotient = { (G,S,H) | G/view = H }
- nonquotient = { (G,S,H) | G/view ≠ H }

nonquotient ∈ MAV (views at distance |G/view|)

- accompanied = {(G,S,x), |S|>1}

accompanied ∈ MAV (lead all nodes to same node)

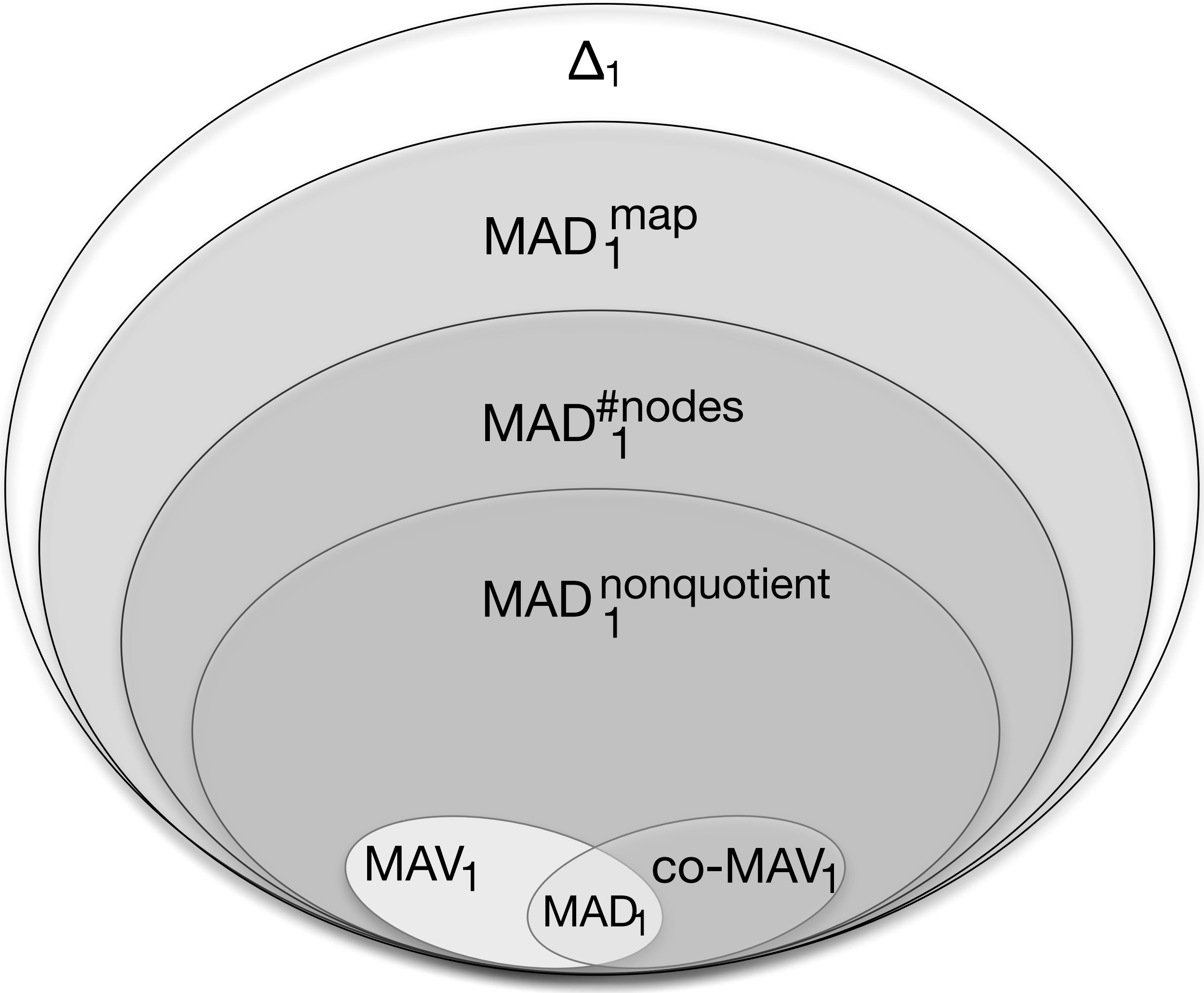
Main Result

$\mathcal{L}_1 \times \mathcal{L}_2 = \{\, (G,S,(i,x)) \mid i \in \{1,2\}$ and $(G,S,x) \in \mathcal{L}_i \,\}$

Theorem (F, Pelc, 2012).
`accompanied` x `nonquotient` is MAV-complete (for 'natural' reduction).

Corollary `nonquotient` is MAV_1-complete.

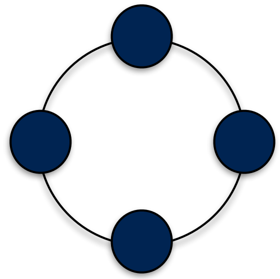
Case of a Single Agent



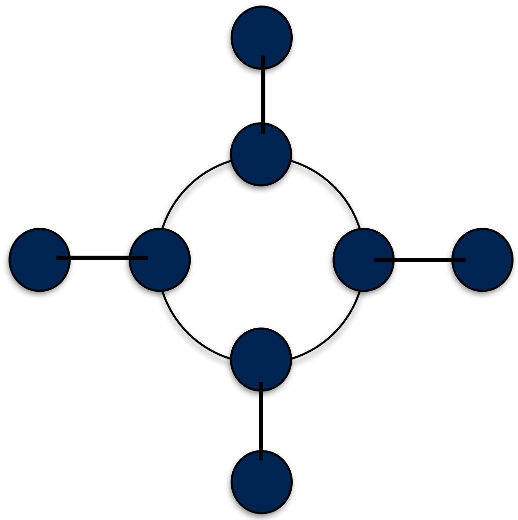
Equalities and Separations

$\mathrm{MAV}_1 \cap \text{co-}\mathrm{MAV}_1 = \mathrm{MAD}_1$ (test all certificates)

$\mathrm{MAV}_1 \cup \text{co-}\mathrm{MAV}_1 \subset \mathrm{MAD}_1^{\texttt{NonQuotient}}$

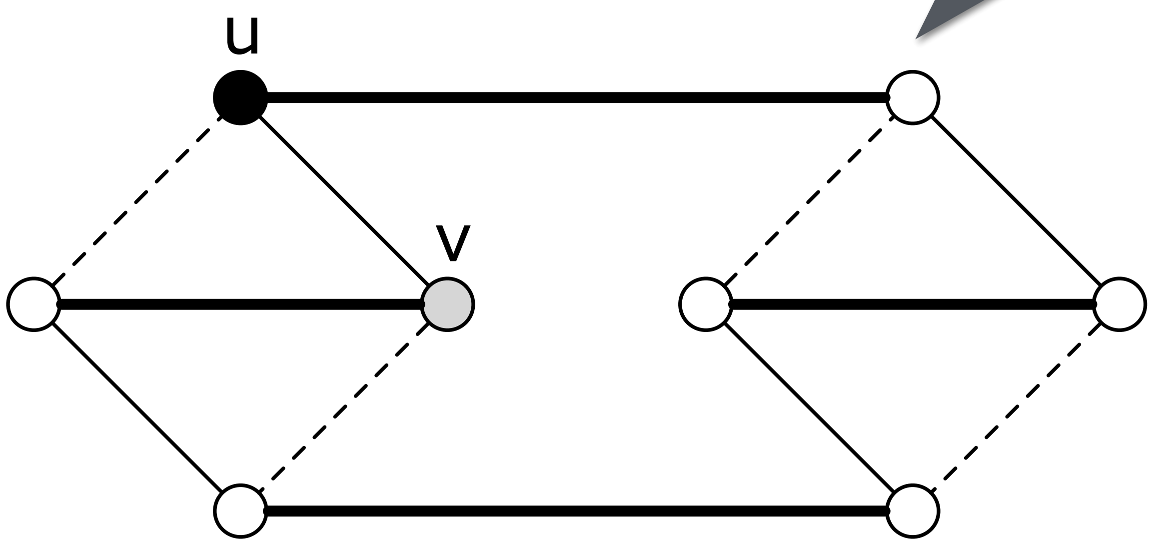


- $\texttt{cycle} \times \texttt{nosun} \notin \mathrm{MAV}_1 \cup \text{co-}\mathrm{MAV}_1$
- $\texttt{cycle} \times \texttt{nosun} \in \mathrm{MAD}_1^{\texttt{NonQuotient}}$



More separations

$\mathrm{MAD}_1^{\texttt{nonquotient}} \subset \mathrm{MAD}_1^{\texttt{\#nodes}} \subset \underbrace{\mathrm{MAD}_1^{\texttt{map}} \subset \mathrm{All}_1}$



Concluding remarks

Objective: developing an embryo of computability theory for mobile agent computing.

Formalize the informal notion of ‘initial knowledge’

Open problems:

- Construction vs. decision for mobile agent computing?
- Complexity theory? (What is the right measure?)
- Role of randomization?

- P. Fraigniaud and A. Pelc, *Decidability Classes for Mobile Agents Computing*, In LATIN 2012.
- E. Bampas and D. Ilcinkas, *Problèmes vérifiables par agents mobiles*, In AlgoTel 2015.