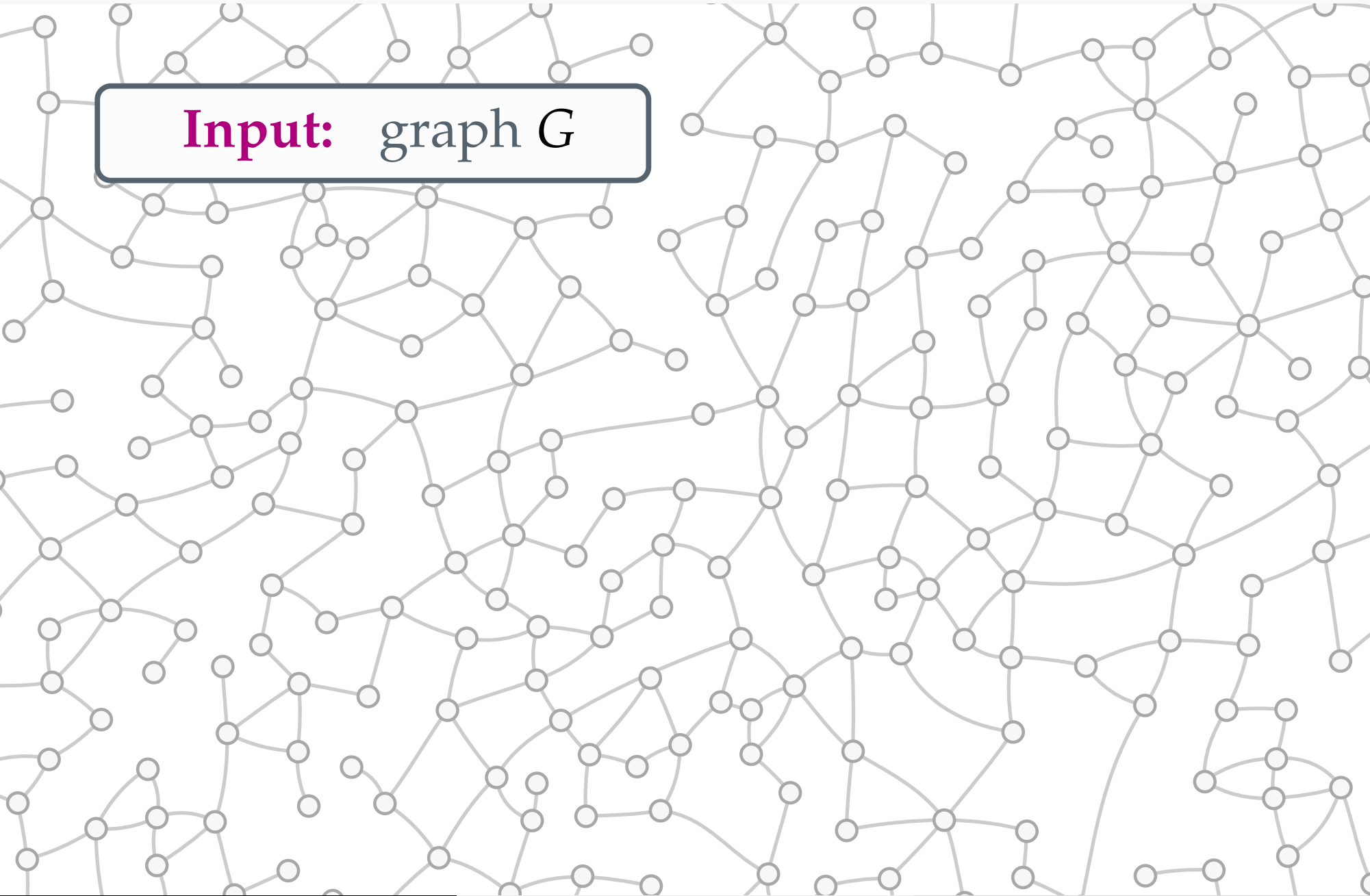
What can be decided locally without identifiers?

Pierre Fraigniaud — *University Paris Diderot & CNRS*
Mika Göös — *University of Toronto*
Amos Korman — *University Paris Diderot & CNRS*
Jukka Suomela — *University of Helsinki & HIIT*

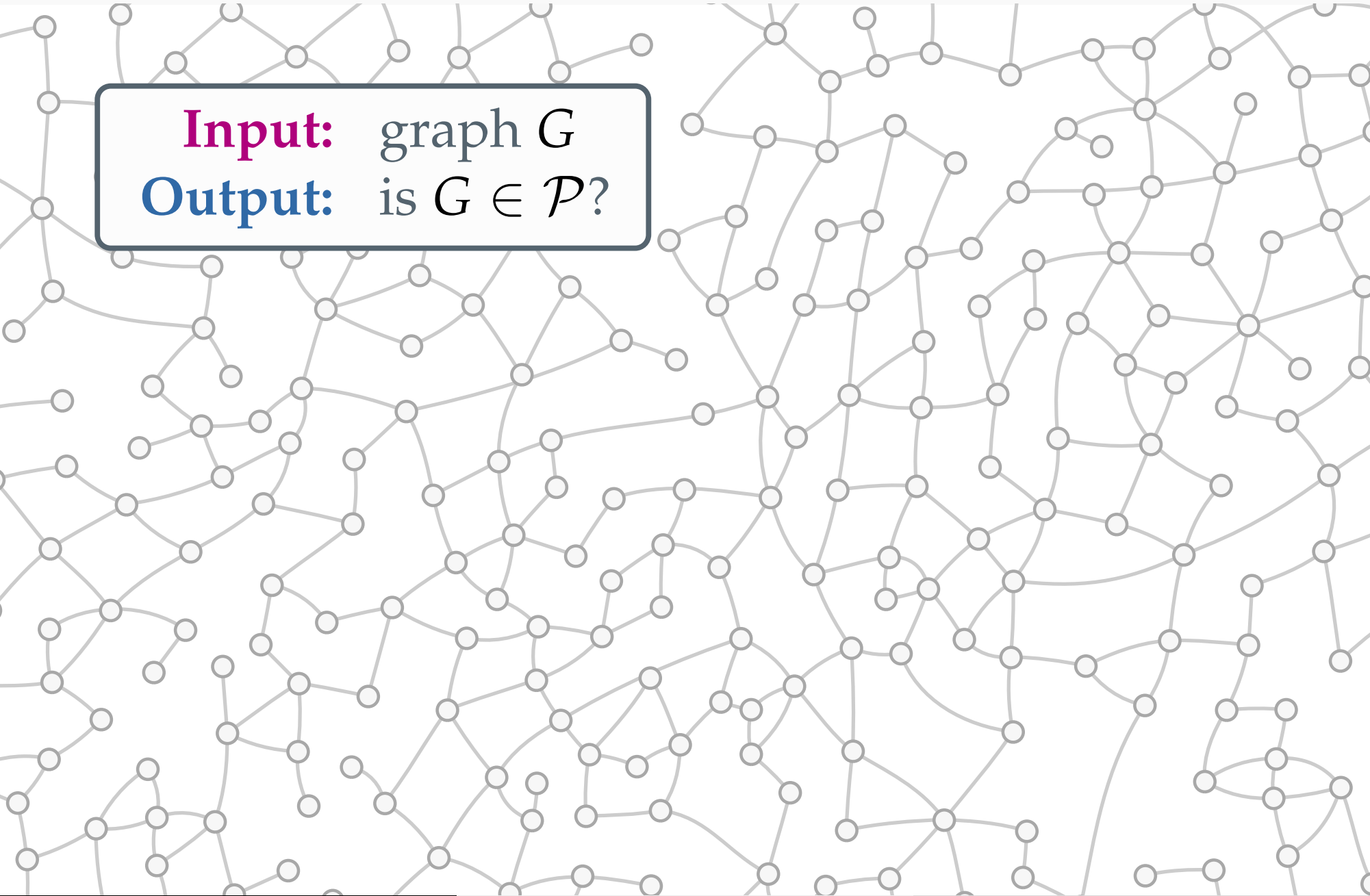
Local decision

[FKP **FOCS'11**]



Local decision

[FKP **FOCS'11**]

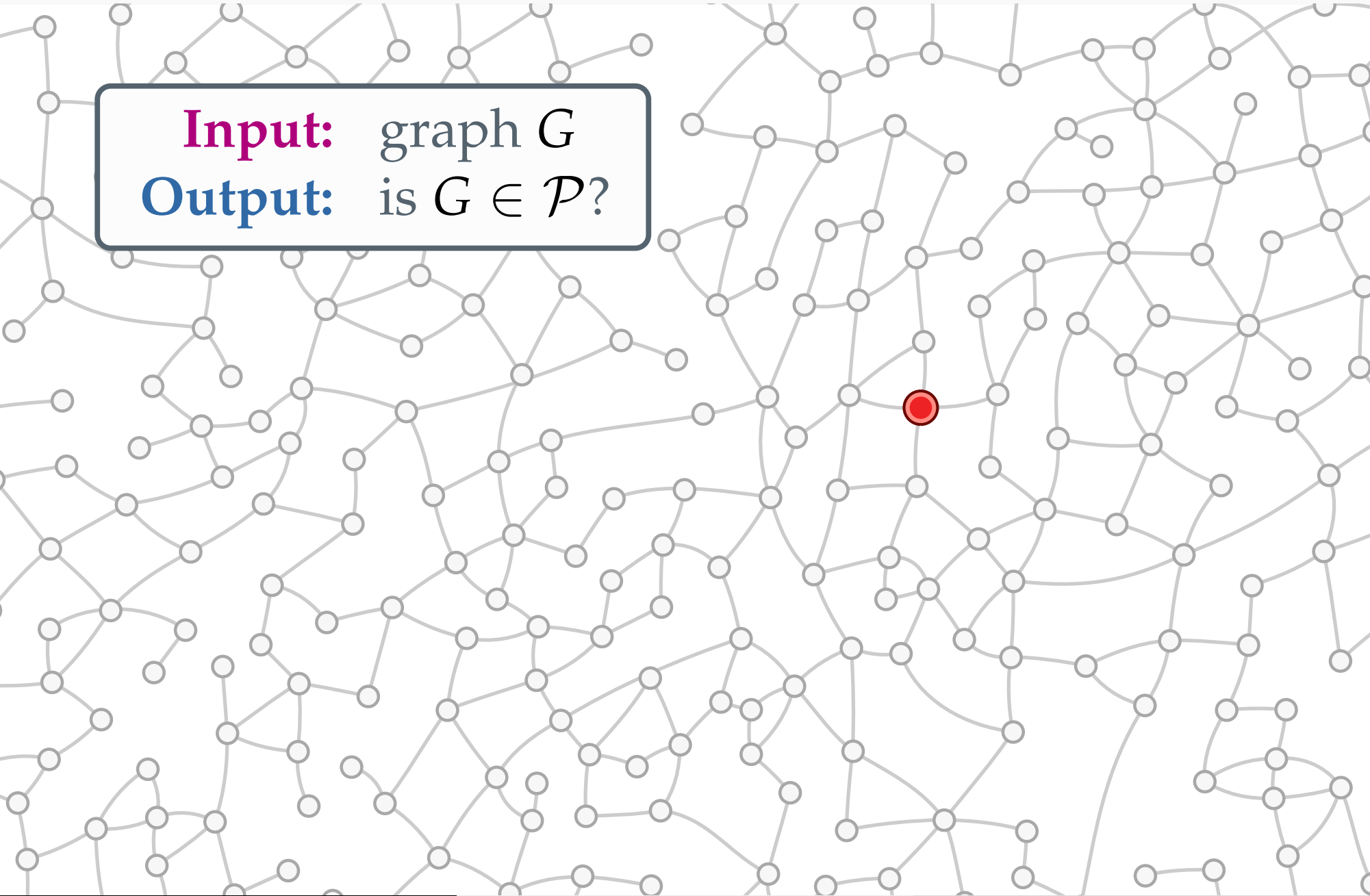


Input: graph G

Output: is $G \in \mathcal{P}$?

Local decision

[FKP **FOCS'11**]



Local decision

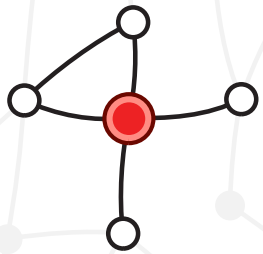
[FKP **FOCS'11**]

Input: graph G
Output: is $G \in \mathcal{P}$?

Local decision

[FKP **FOCS'11**]

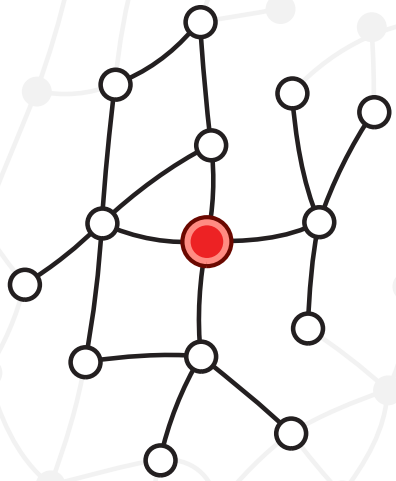
Input: graph G
Output: is $G \in \mathcal{P}$?



Local decision

[FKP **FOCS'11**]

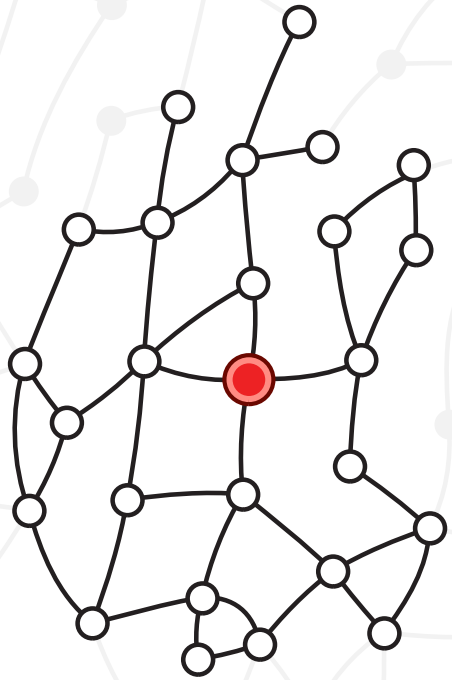
Input: graph G
Output: is $G \in \mathcal{P}$?



Local decision

[FKP **FOCS'11**]

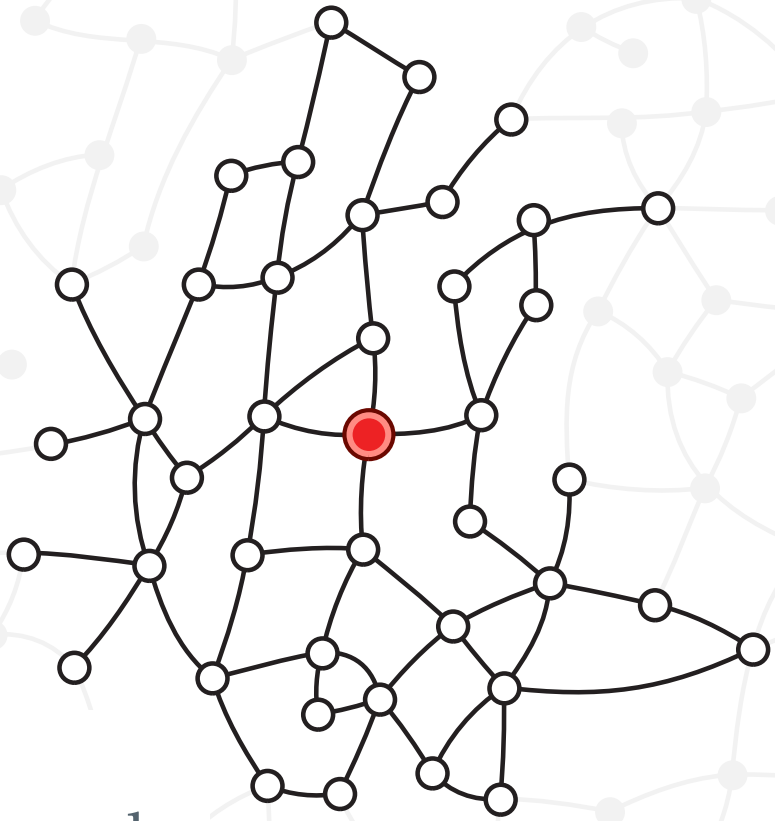
Input: graph G
Output: is $G \in \mathcal{P}$?



Local decision

[FKP **FOCS'11**]

Input: graph G
Output: is $G \in \mathcal{P}$?

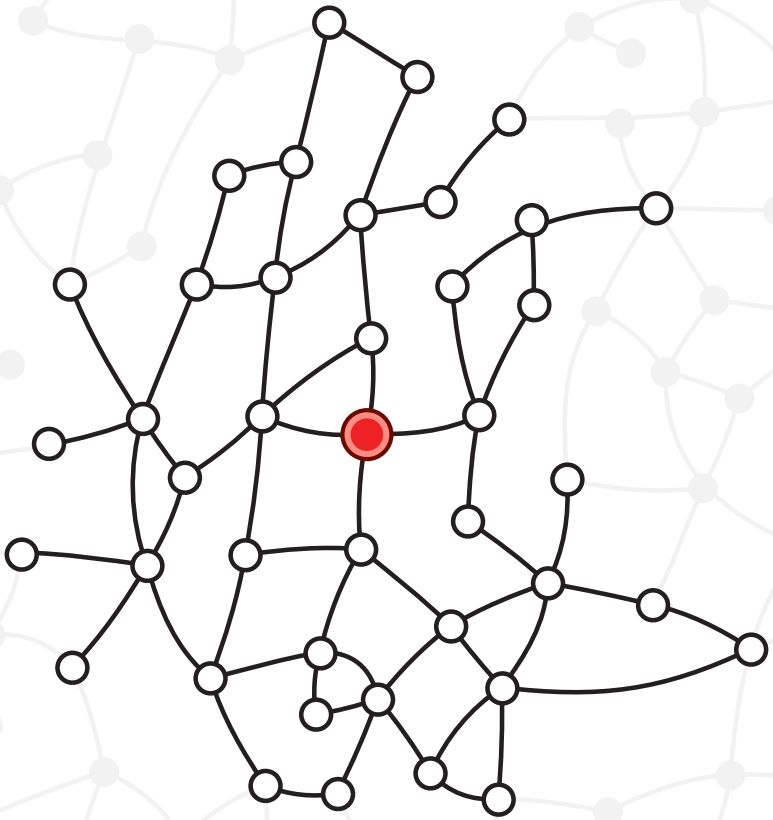


Local algorithm
$\equiv$ $O(1)$ communication rounds
$\equiv$ $O(1)$ radius neighbourhood

Local decision

[FKP **FOCS'11**]

Input: graph G
Output: is $G \in \mathcal{P}$?

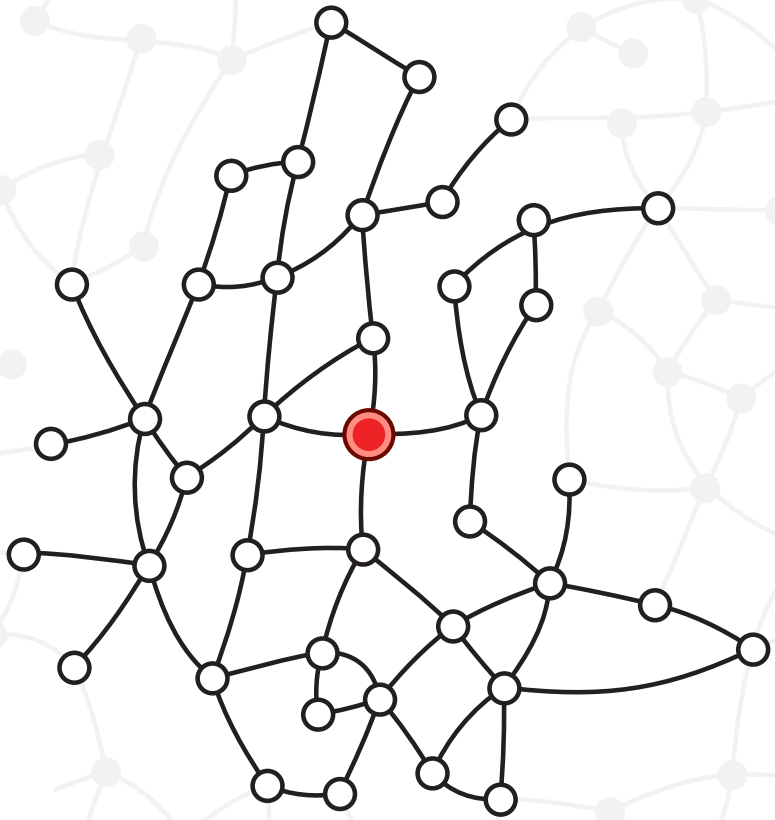


yes / *no*

Local decision

[FKP **FOCS'11**]

Input: graph G
Output: is $G \in \mathcal{P}$?

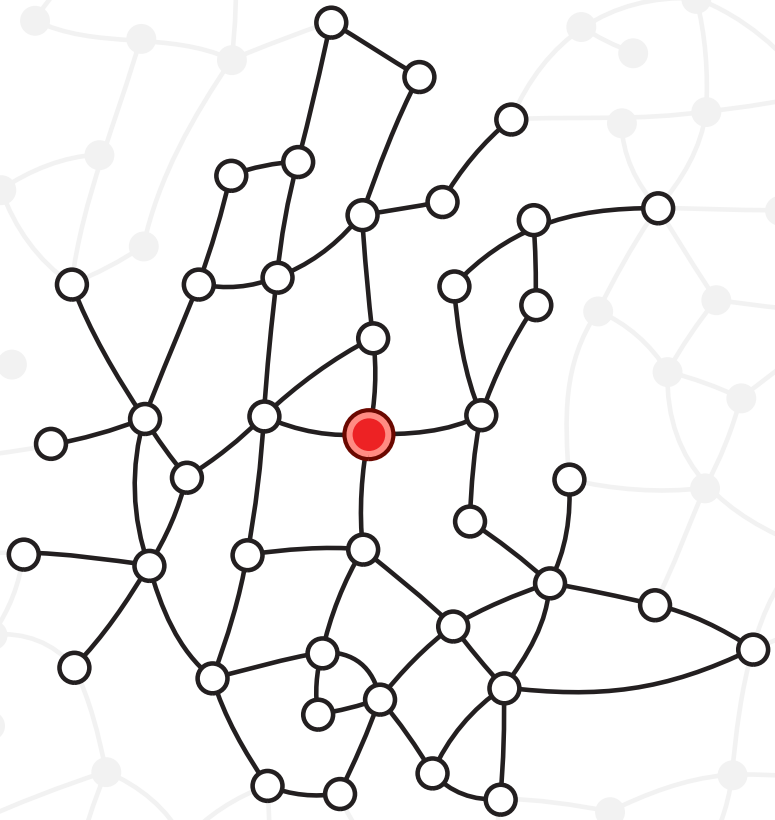


***G* is accepted iff**
all nodes ouput ***yes***

Local decision

[FKP **FOCS'11**]

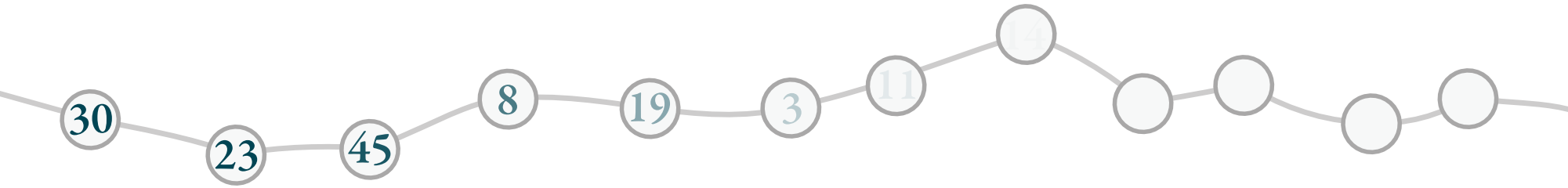
Input: graph G
Output: is $G \in \mathcal{P}$?



Locally decidable $\mathcal{P}$:

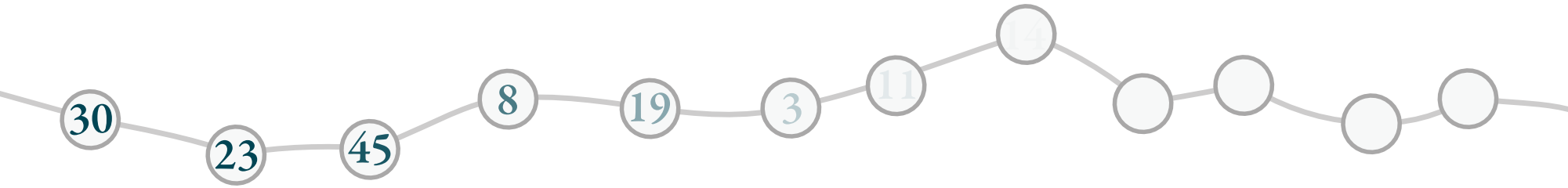
- triangle-freeness
- Eulerian graphs
- line graphs
- Locally checkable labellings (G, ℓ)

Our question



We ask: Do **node identifiers** help in local decision?

Our question



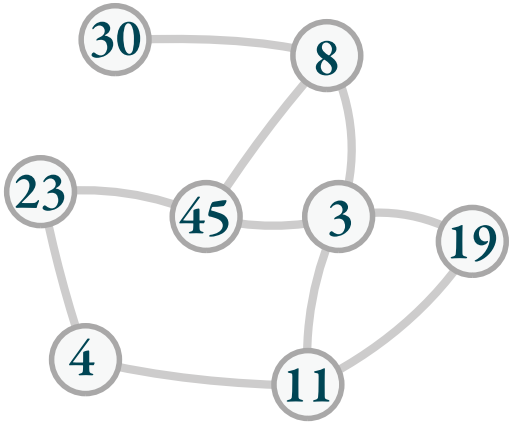
We ask: Do **node identifiers** help in local decision?

IDs do not seem useful...

- Graph properties do not depend on node labels
- **Symmetry breaking** is not needed for decision problems!

Our question—formalised

[FHK **OPODIS'12**]

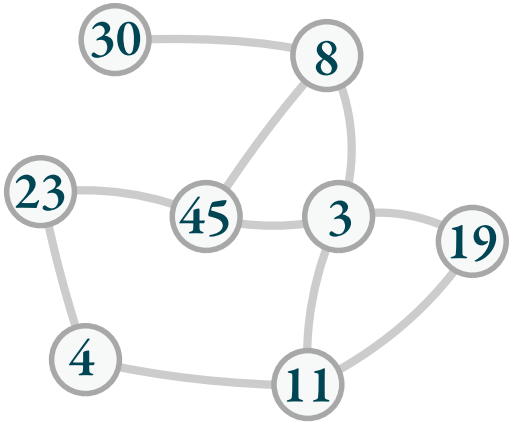


LOCAL model
(deterministic)

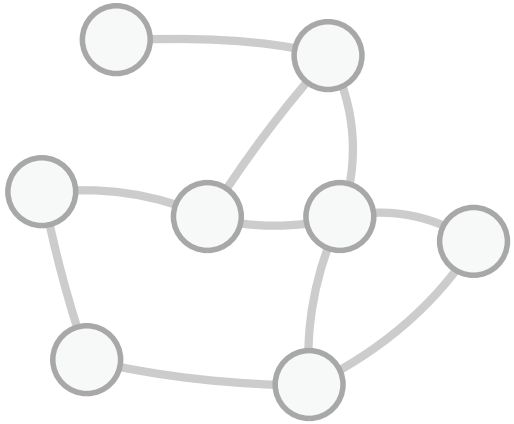
$V(G) \subseteq \{1, 2, 3, \ldots\}$

Our question—formalised

[FHK **OPODIS'12**]



VS.



$\mathcal{LOCAL}$ model

(deterministic)

$V(G) \subseteq \{1, 2, 3, \ldots\}$

ID-oblivious model

Restriction: Output is **invariant under relabelling** the nodes

(i.e., depends only on **topology**)

Easy cases

Warm up!

Under some assumptions:

$$\mathcal{LOCAL} = \text{ID-oblivious}$$

Proof by simulation...

Easy cases

Let A be a $\mathcal{LOCAL}$ decision algorithm

ID-oblivious simulation of A

Input: local neighbourhood (H, v) of G

For each ID-assignment $f : V(H) \to \{1, 2, \ldots, n\}$:

- if $A(f(H, v)) = no$ then **output** no.

Otherwise **output** yes.

Assumptions:

- Nodes know n

Easy cases

Let A be a $\mathcal{LOCAL}$ decision algorithm

ID-oblivious simulation of A

Input: local neighbourhood (H, v) of G

For each ID-assignment $f : V(H) \to \{1, 2, \dots\}$:

- if $A(f(H, v)) = no$ then **output** no.

Otherwise **output** yes.

Assumptions:

- Nodes do not know n

Easy cases

Let A be a $\mathcal{LOCAL}$ decision algorithm

ID-oblivious simulation of A

Input: local neighbourhood (H, v) of G

For each ID-assignment $f(H) \to \{1, 2, \dots\}$:

- if $A(f(H, v)) = no$ then output no.

Otherwise output yes.

Assumptions:

- Nodes do not know n
- Nodes are **Turing computable**

Our main result

Main theorem*

$$\mathcal{LOCAL} \neq \text{ID-oblivious}$$

(I.e., there is a locally decidable property
that cannot be decided ID-obliviously)

Assumptions:

- Nodes do not know n
- Nodes are **Turing computable**

Our main result

Main theorem*

$$\mathcal{LOCAL} \neq \text{ID-oblivious}$$

(I.e., there is a locally decidable property
that cannot be decided ID-obliviously)

* Contrary to a conjecture of [FHK'12]

Assumptions:

- Nodes do not know n
- Nodes are **Turing computable**

Our main result

Main theorem*

$\mathcal{LOCAL} \neq$ ID-oblivious

(I.e., there is a locally decidable property
that cannot be decided ID-obliviously)

* Contrary to a conjecture of [FHK'12]

Proof...

Separation under promise

Promise problem

Input:
- $G = (G, M)$ is a labelled $\boldsymbol{n}$-cycle
- M is a Turing machine

Promise:
- If M halts in s steps, then $\boldsymbol{n} \geq s$

Output:
- ***yes*** if M runs forever
- ***no*** if M halts

Separation under promise

Promise problem

Input:
- $G = (G, M)$ is a labelled $\boldsymbol{n}$-cycle
- M is a Turing machine

Promise:
- If M halts in s steps, then $\boldsymbol{n} \geq s$

Output:
- ***yes*** if M runs forever
- ***no*** if M halts

ID-oblivious **Impossible:** Must solve the Halting Problem

Separation under promise

Promise problem

Input:
- $G = (G, M)$ is a labelled $\boldsymbol{n}$-cycle
- M is a Turing machine

Promise:
- If M halts in s steps, then $\boldsymbol{n} \geq s$

Output:
- ***yes*** if M runs forever
- ***no*** if M halts

ID-oblivious **Impossible:** Must solve the Halting Problem

$\mathcal{LOCAL}$ **Possible:** Node v simulates M for $\mathrm{ID}(v)$ steps

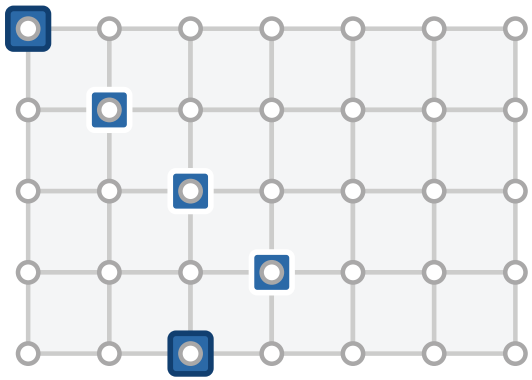
Getting rid of the promise

Promise: • If M halts in s steps, then $\boldsymbol{n} \geq s$

Getting rid of the promise

Promise: • If M halts in s steps, then $n \geq s$

⇓ **Replace!** ⇓



Computation table of M

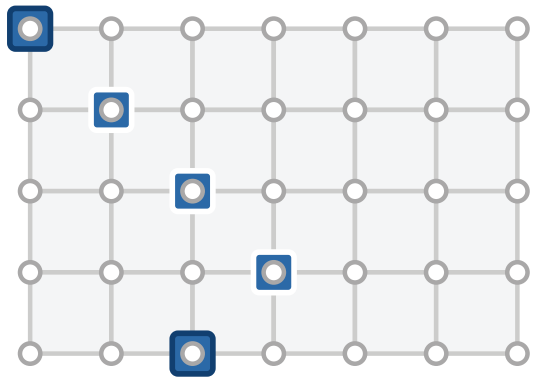
$\subseteq$ G

yes instance

Getting rid of the promise

Promise: • If M halts in s steps, then $n \geq s$

⇓ **Replace!** ⇓



Computation table of M

$\subseteq$ G

yes instance

Interesting bit: Table needs to be **obfuscated!**

Summary

For local decision, we proved:

$$\mathcal{LOCAL} \neq \text{ID-oblivious}$$

Summary

For local decision, we proved:

$$\mathcal{LOCAL} \neq \text{ID-oblivious}$$

	IDs help	IDs don't help
Decision	**This work**	[FHK **OPODIS'12**]
Search	[HHRS **SIROCCO'12**]	[NS **Sicomp'95**] [GHS **PODC'12**]

Summary

For local decision, we proved:

$$\mathcal{LOCAL} \neq \text{ID-oblivious}$$

Randomisation?

- Open problems in **randomised** decision [FKPP **DISC'12**]

Summary

For local decision, we proved:

$$\mathcal{LOCAL} \neq \text{ID-oblivious}$$

Randomisation?

- Open problems in **randomised** decision [FKPP **DISC'12**]

Cheers!