Limits of Distributed Quantum Computing

GRAN SASSO SCIENCE INSTITUTE

Francesco d'Amore

Based on [STOC '24, STOC '25a, STOC '25b, SODA '26]

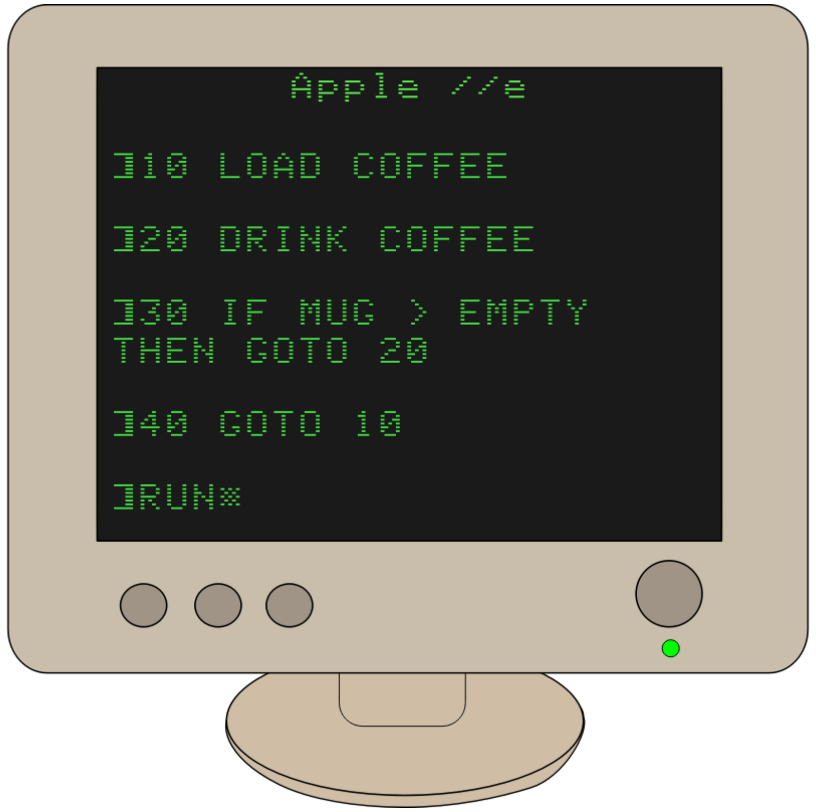
Joint works with Amirreza Akbari, Alkida Balliu, Sebastian Brandt, Filippo Casagrande, Xavier Coiteux-Roy, Massimo Equi, Rishikesh Gajjala, Barbara Keller, Fabian Kuhn, François Le Gall, Henrik Lievonen, Darya Melnyk, Augusto Modanese, Dennis Olivetti, Shreyas Pai, Marc-Olvier Renou, Václav Rozhon, Gustav Schmid, Jukka Suomela, Lucas Tendick, Isadora Veeren

Table of content

Table of content

Distributed algorithms

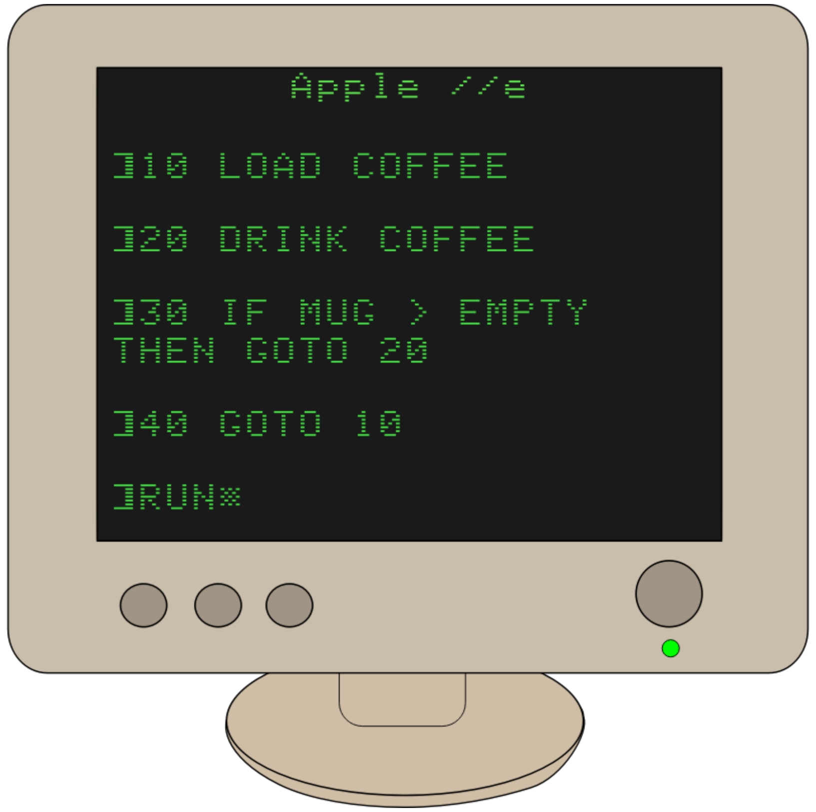
- Write a program $\mathcal{A}$ for a single computer (e.g., for $(\Delta + 1)$-coloring a graph)
  - use commands like *send a message through this communication port*, etc.



Distributed algorithms

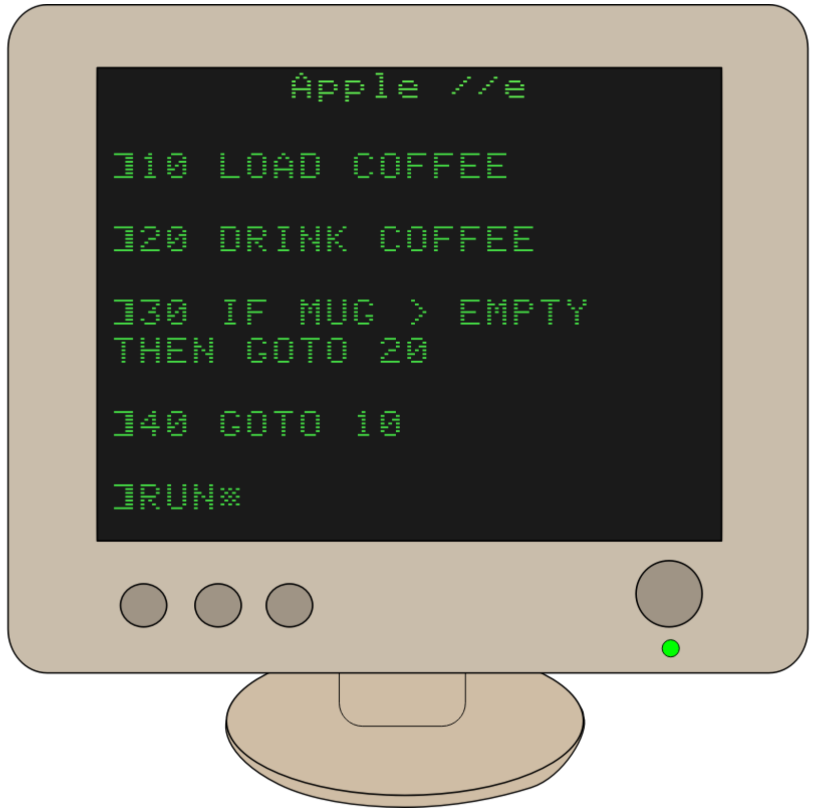
- Write a program $\mathcal{A}$ for a single computer (e.g., for $(\Delta+1)$-coloring a graph)
  - use commands like *send a message through this communication port*, etc.

- Adversary constructs a network of n computers, all running program $\mathcal{A}$
  - **goal**: solve $(\Delta+1)$-coloring on the network

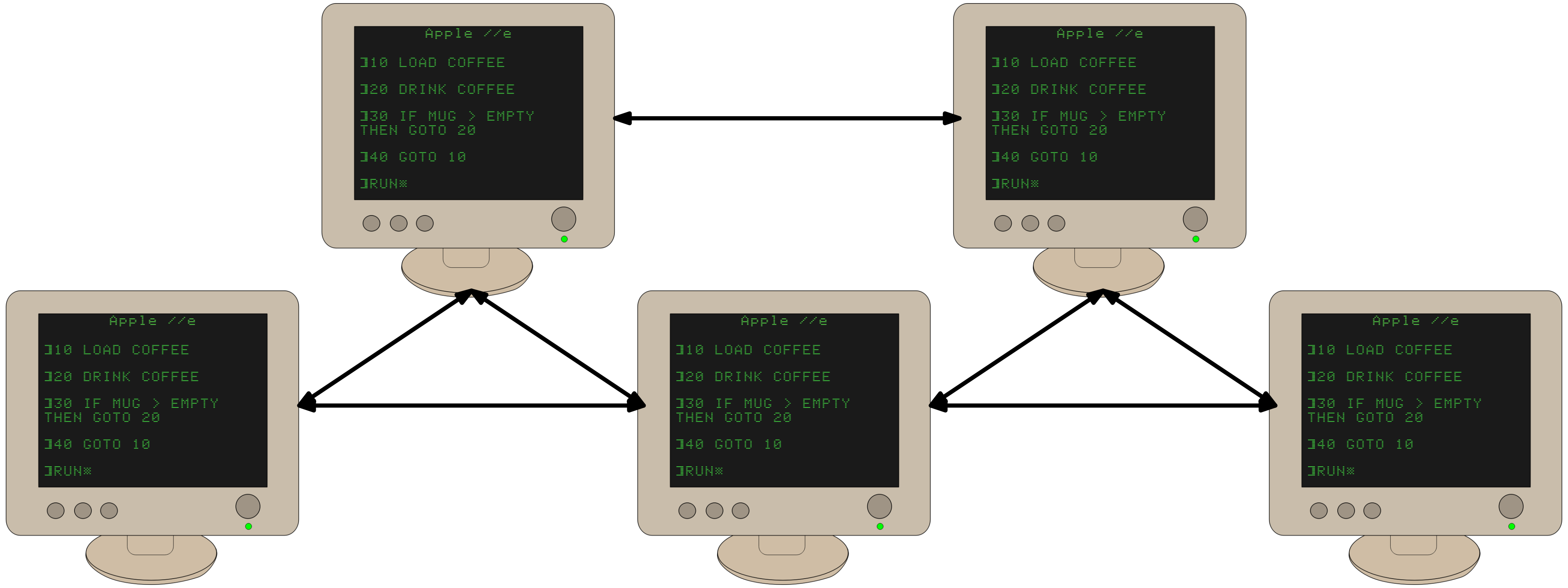


Distributed algorithms

- Write a program $\mathcal{A}$ for a single computer (e.g., for $(\Delta + 1)$-coloring a graph)
  - use commands like *send a message through this communication port*, etc.

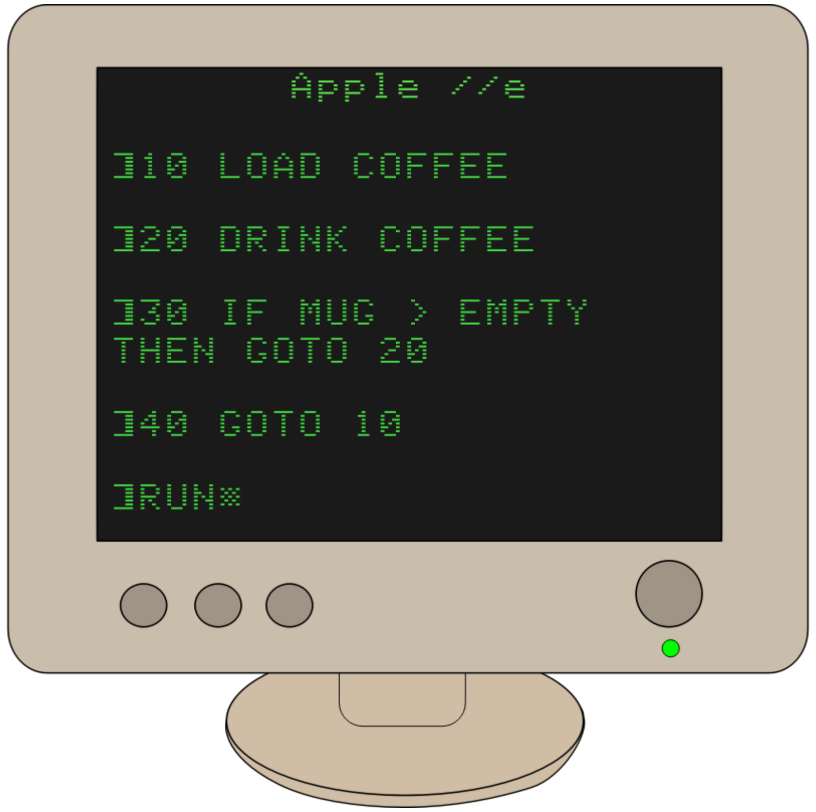


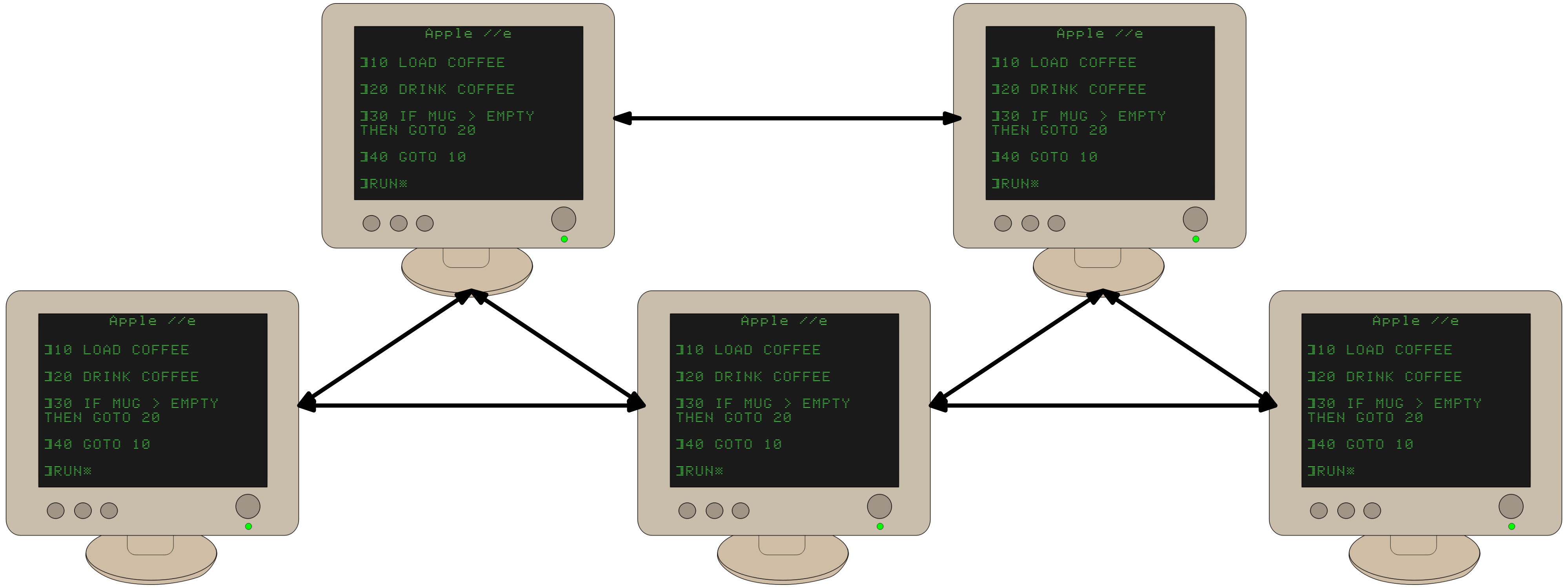
- Adversary constructs a network of n computers, all running program $\mathcal{A}$
  - **goal**: solve $(\Delta + 1)$-coloring on the network



Distributed algorithms

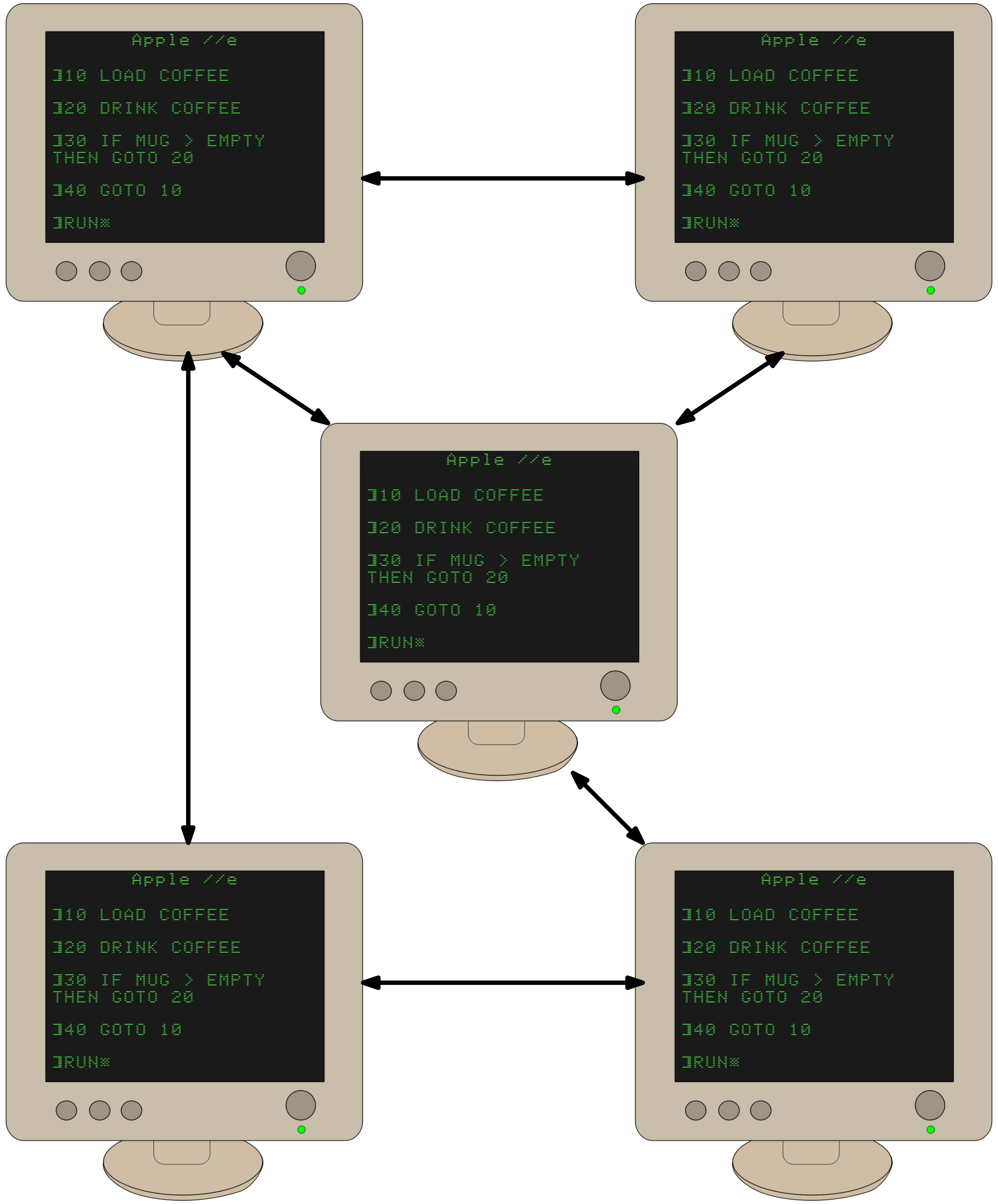
- Write a program $\mathcal{A}$ for a single computer (e.g., for $(\Delta + 1)$-coloring a graph)
  - use commands like *send a message through this communication port*, etc.
- Adversary constructs a network of n computers, all running program $\mathcal{A}$
  - **goal**: solve $(\Delta + 1)$-coloring on the network
- **Switch everything on**, see what happens





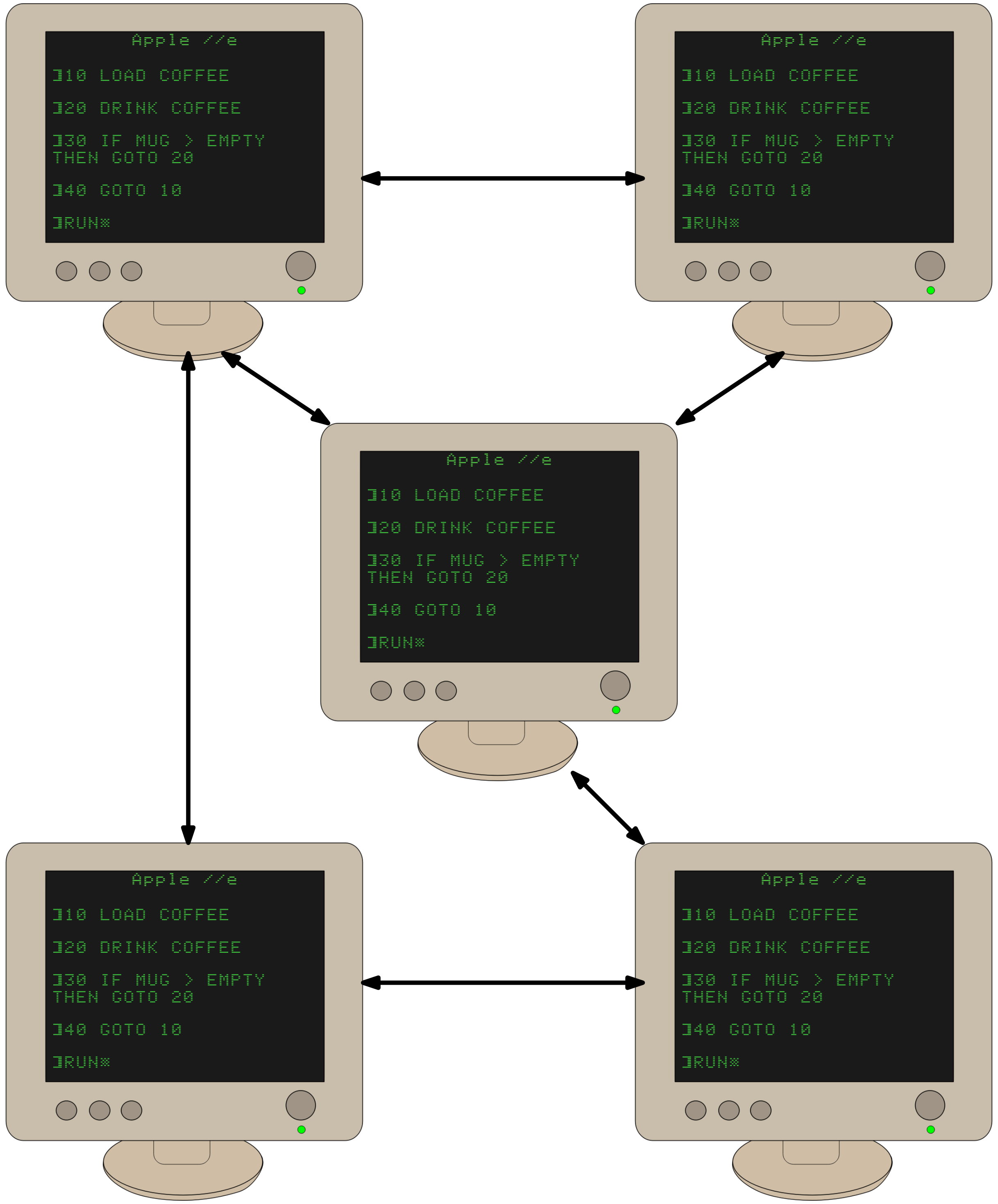
Distributed algorithms

- **Abstractions**
  - identical computers



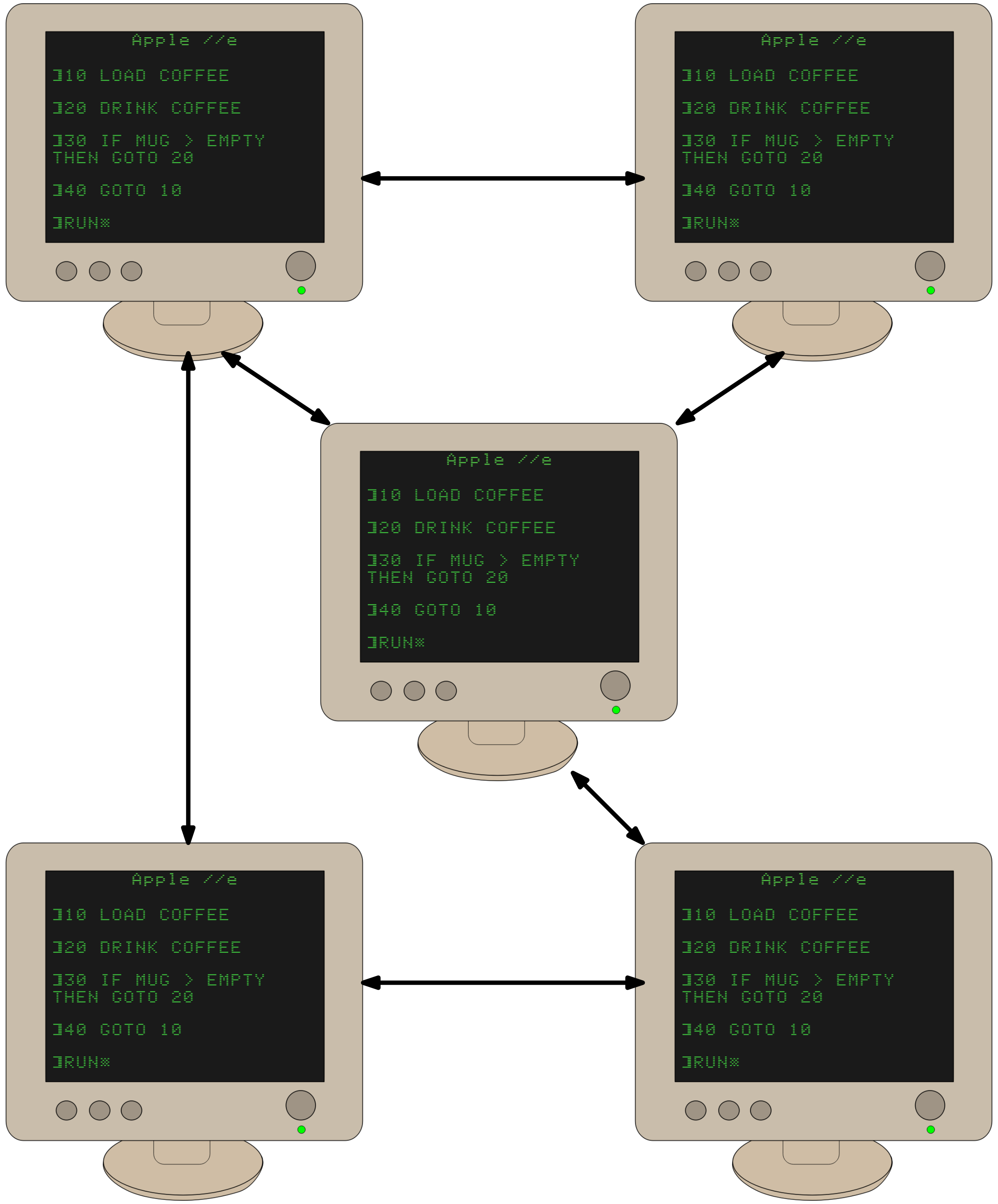
Distributed algorithms

- **Abstractions**
  - identical computers
  - synchronous rounds



Distributed algorithms

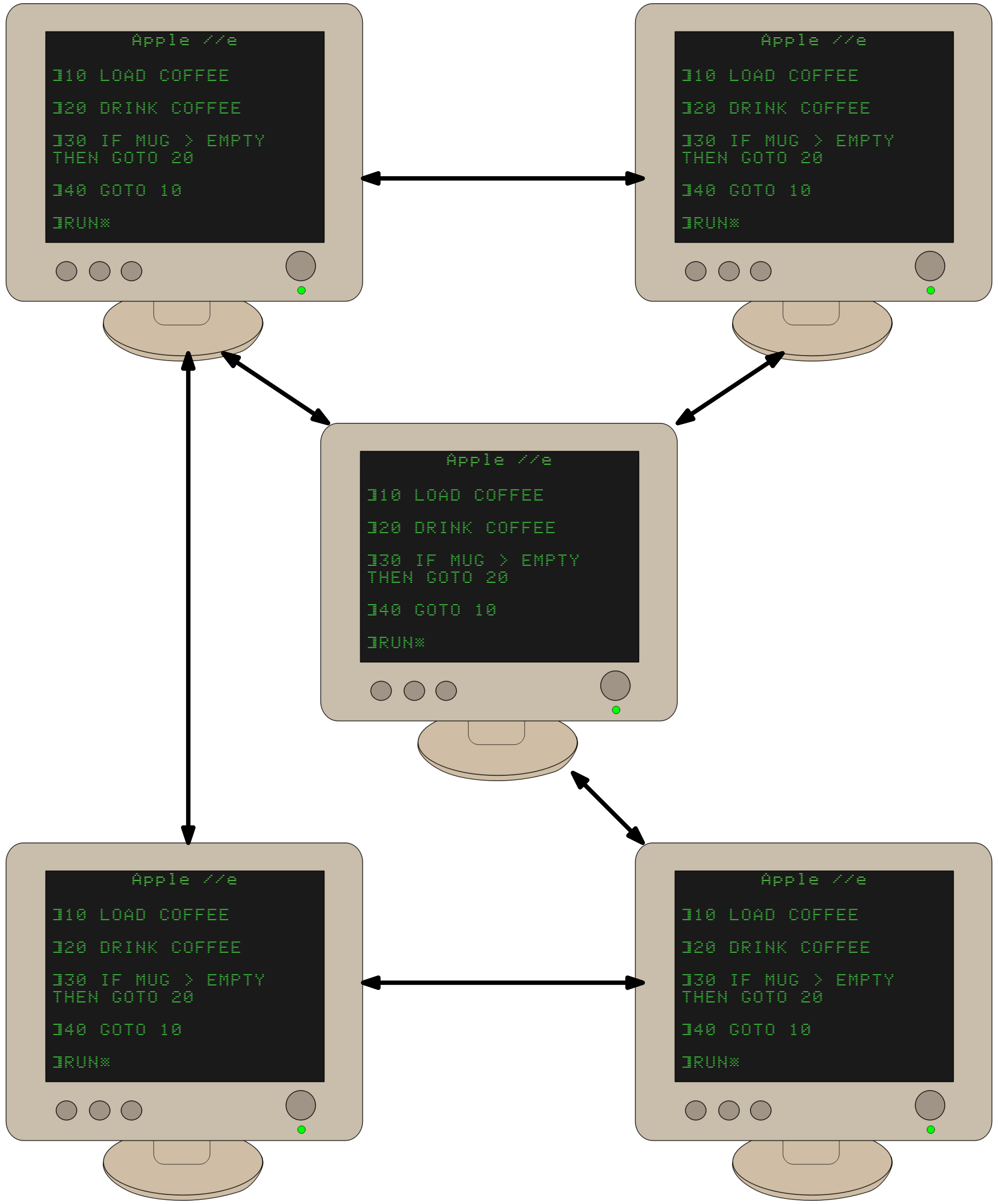
- **Abstractions**
  - identical computers
  - synchronous rounds
  - each round:
    - local computation
    - send/receive messages to/from all neighbors



Distributed algorithms

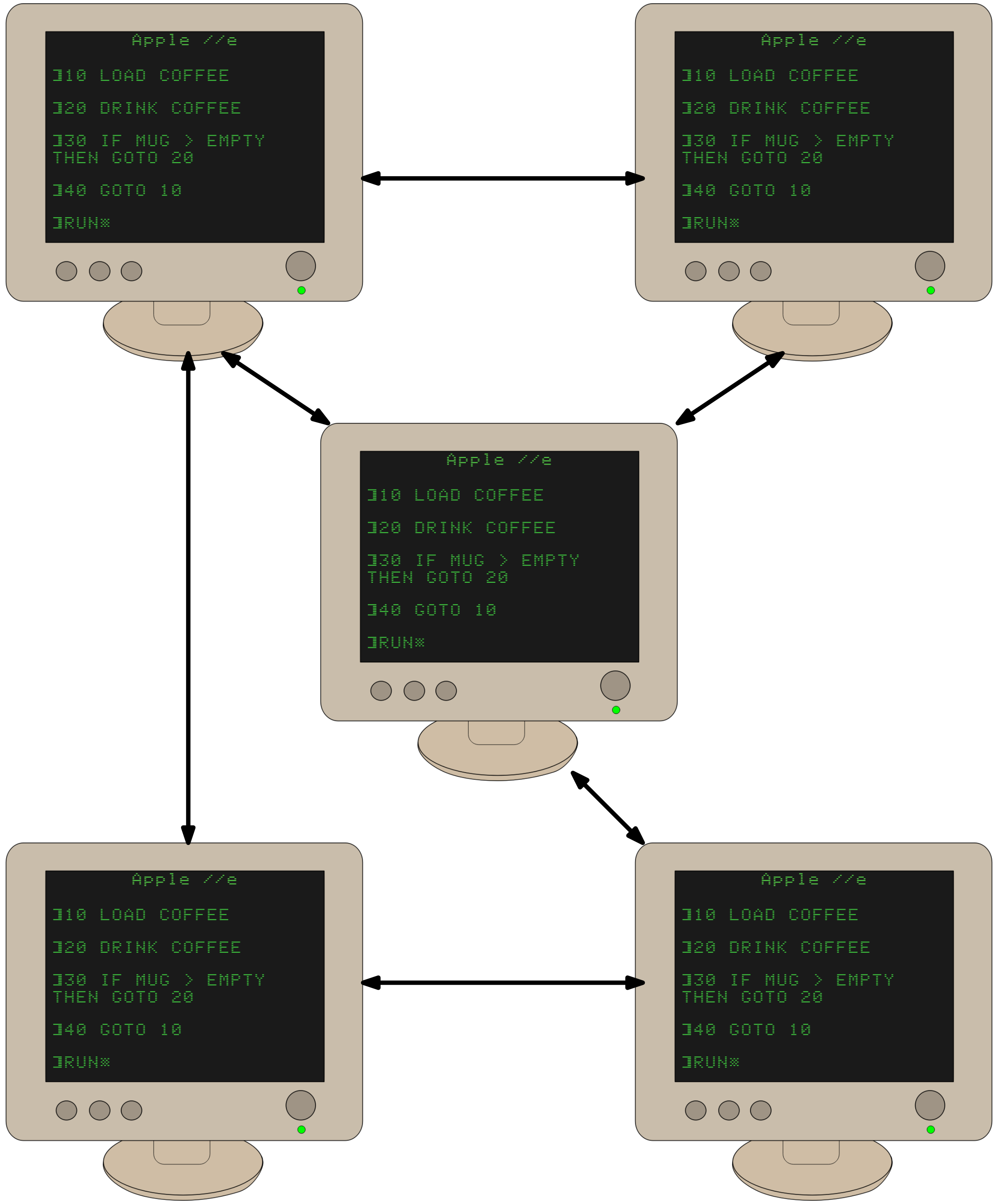
- **Abstractions**
  - identical computers
  - synchronous rounds
  - each round:
    - local computation
    - send/receive messages to/from all neighbors

- **Running time**: number of communication rounds



Distributed algorithms

- **Abstractions**
  - identical computers
  - synchronous rounds
  - each round:
    - local computation
    - send/receive messages to/from all neighbors
- **Running time**: number of communication rounds
- **Challenge**: what to do in the middle of a network?



Computability with identical computers

- **Problem**: $\mathbf{2}$-coloring $\mathbf{2}$-paths

Computability with identical computers

- **Problem**: $\mathbf{2}$-coloring $\mathbf{2}$-paths
- *Promise*: the graph constructed is a path of length $\mathbf{1}$

Computability with identical computers

- **Problem**: $\mathbf{2}$-coloring $\mathbf{2}$-paths
- *Promise*: the graph constructed is a path of length $\mathbf{1}$

- *What program?*

Computability with identical computers

- **Problem**: 2-coloring 2-paths
- *Promise*: the graph constructed is a path of length 1

- *What program?* * **Impossible** if the two nodes are truly identical

Computability with identical computers

- **Problem**: 2-coloring 2-paths
- *Promise*: the graph constructed is a path of length 1

- *What program?* * **Impossible** if the two nodes are truly identical
- **Assumption**: *identifiers*. Each node has a unique identifier in $\{1, \ldots, \text{poly}(n)\}$

Computability with identical computers

- **Problem**: 2-coloring 2-paths
- *Promise*: the graph constructed is a path of length 1

- *What program?* * **Impossible** if the two nodes are truly identical
- **Assumption**: *identifiers*. Each node has a unique identifier in $\{1, \ldots, \text{poly}(n)\}$
- **Algorithm**: - exchange identifier

  - if my identifier $>$ received identifier, output blue

  - otherwise output red

Computability with identical computers

- **Problem**: 2-coloring 2-paths
- *Promise*: the graph constructed is a path of length 1

- *What program?* * **Impossible** if the two nodes are truly identical
- **Assumption**: *identifiers*. Each node has a unique identifier in $\{1, \ldots, \mathrm{poly}(n)\}$
- **Algorithm**: - exchange identifier

  - if my identifier > received identifier, output blue

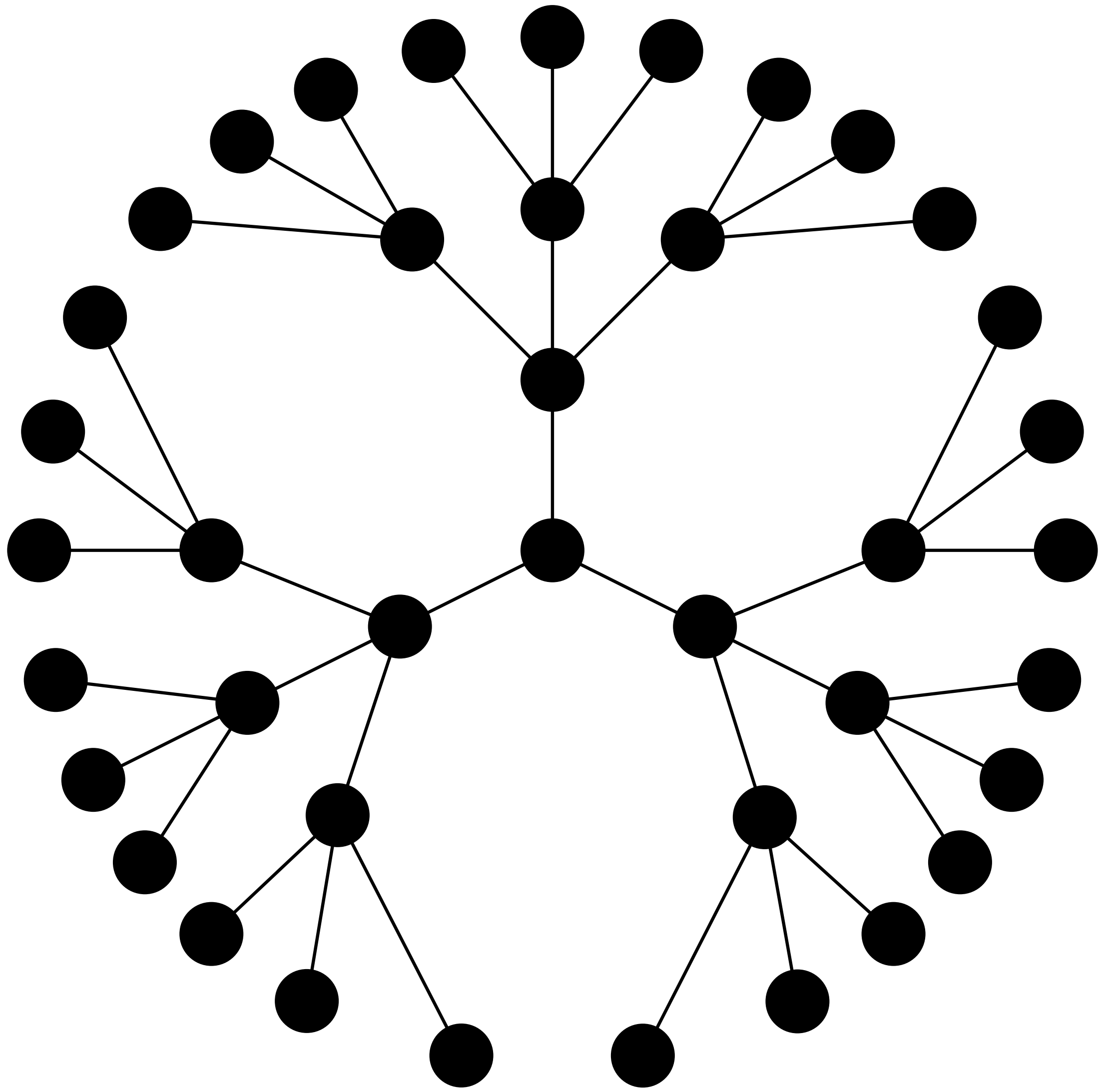
  - otherwise output red
- **Other possibility**: *randomness*. Each node has access to independent source of randomness

The LOCAL model

[Linial FOCS '87 & SICOMP '92]

- **Distributed network** of n processors/nodes

  - graph $G = (V, E)$ with $|V| = n$
  - E: communication links
  - each node in V runs the same algorithm



The LOCAL model

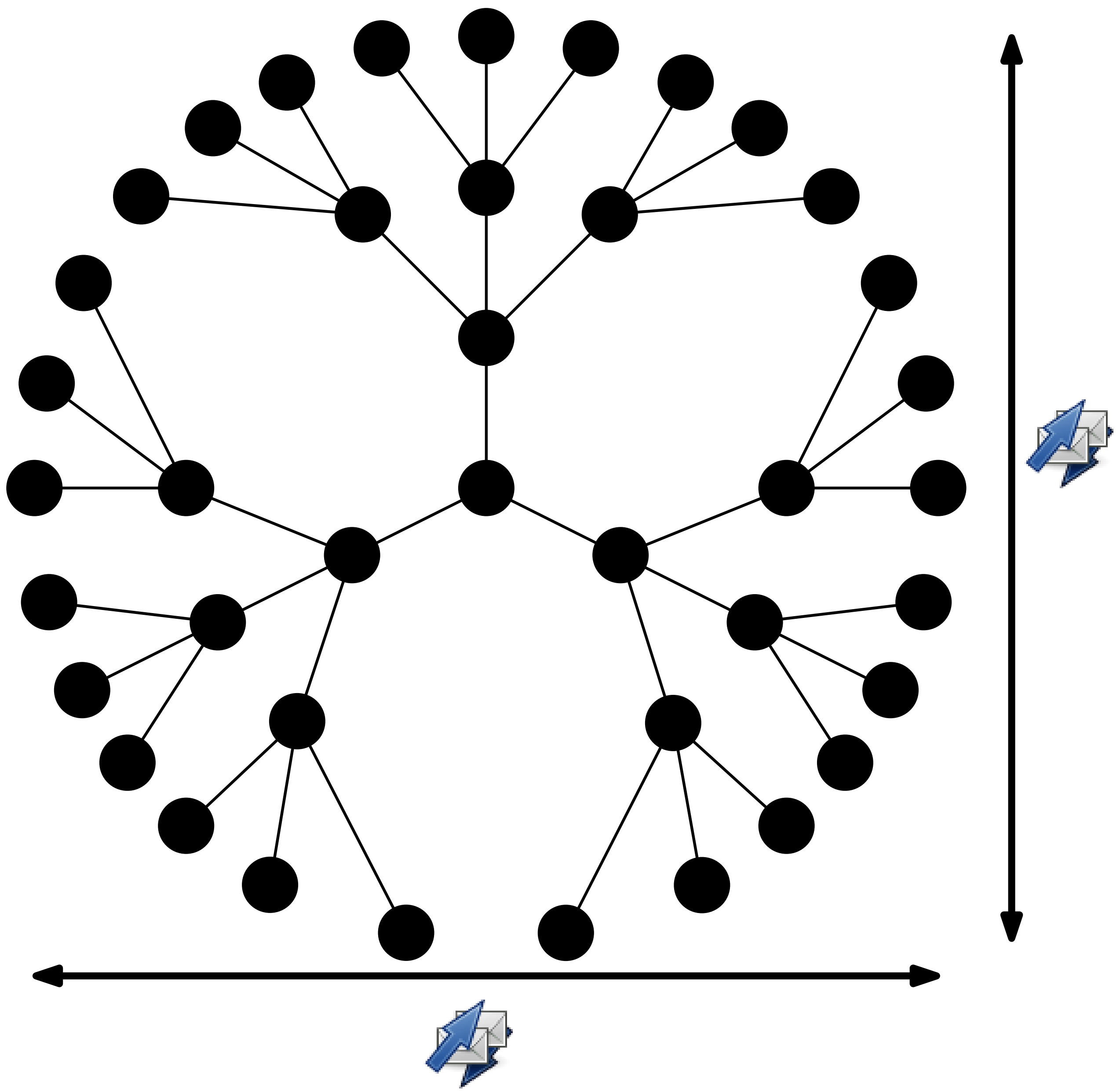
[Linial FOCS '87 & SICOMP '92]

- **Distributed network** of n processors/nodes

  - graph $G = (V, E)$ with $|V| = n$
  - E: communication links
  - each node in V runs the same algorithm

- **Time is synchronous**: nodes alternate

  - arbitrary local computation & update of state variables
  - sending of messages to all neighbors
    * no bandwidth constraints



The LOCAL model

[Linial FOCS '87 & SICOMP '92]

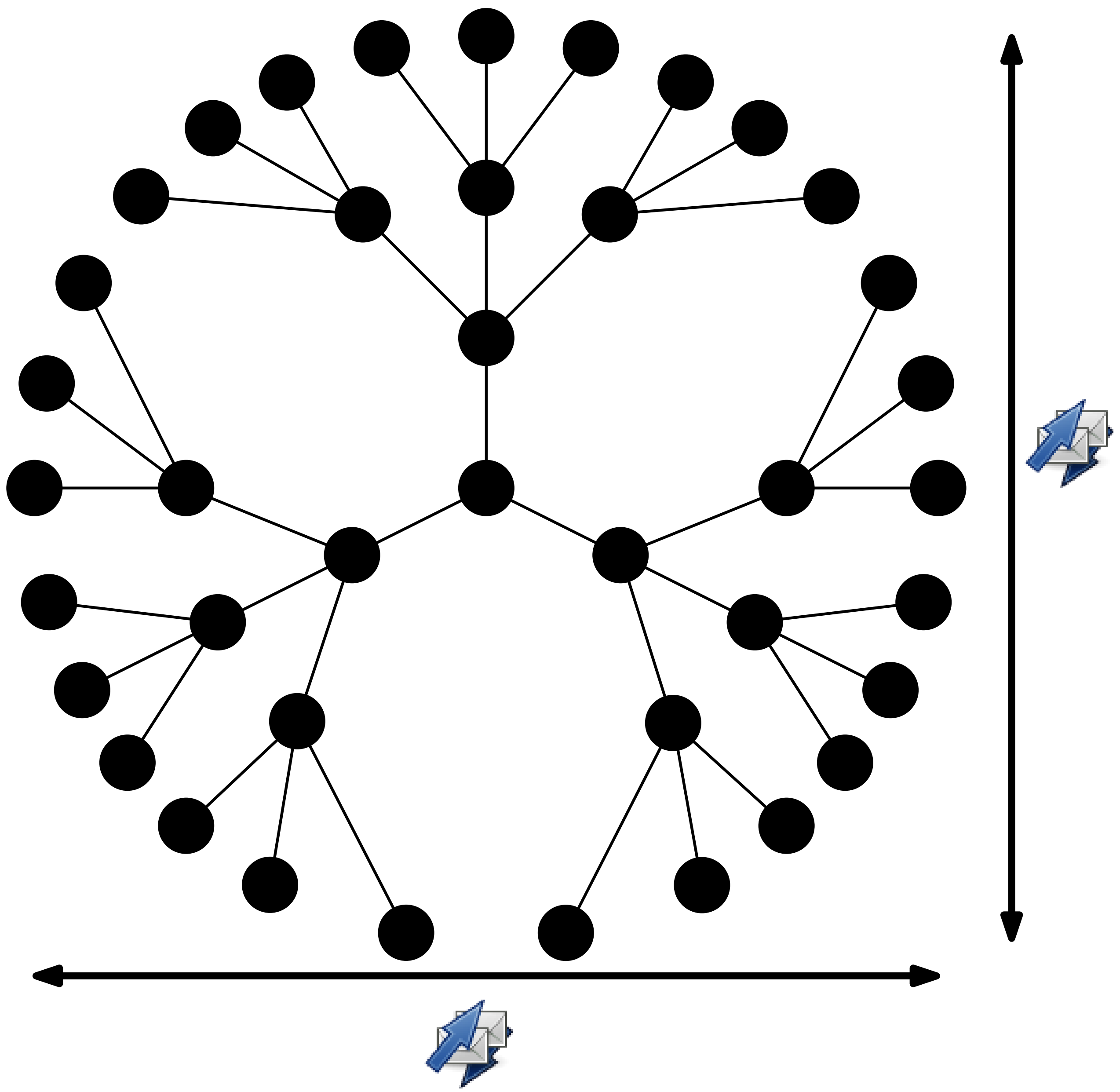
- **Distributed network** of n processors/nodes

  - graph $G = (V, E)$ with $|V| = n$
  - E: communication links
  - each node in V runs the same algorithm

- **Time is synchronous**: nodes alternate

  - arbitrary local computation & update of state variables
  - sending of messages to all neighbors
    * no bandwidth constraints

- **Unique identifiers** to nodes in the set $1, \ldots, \text{poly}(n)$
  * adversarially chosen   * n is known to the nodes
  - needed to solve even basic problems (2-coloring a 2-path)



The LOCAL model

[Linial FOCS '87 & SICOMP '92]

- **Distributed network** of n processors/nodes

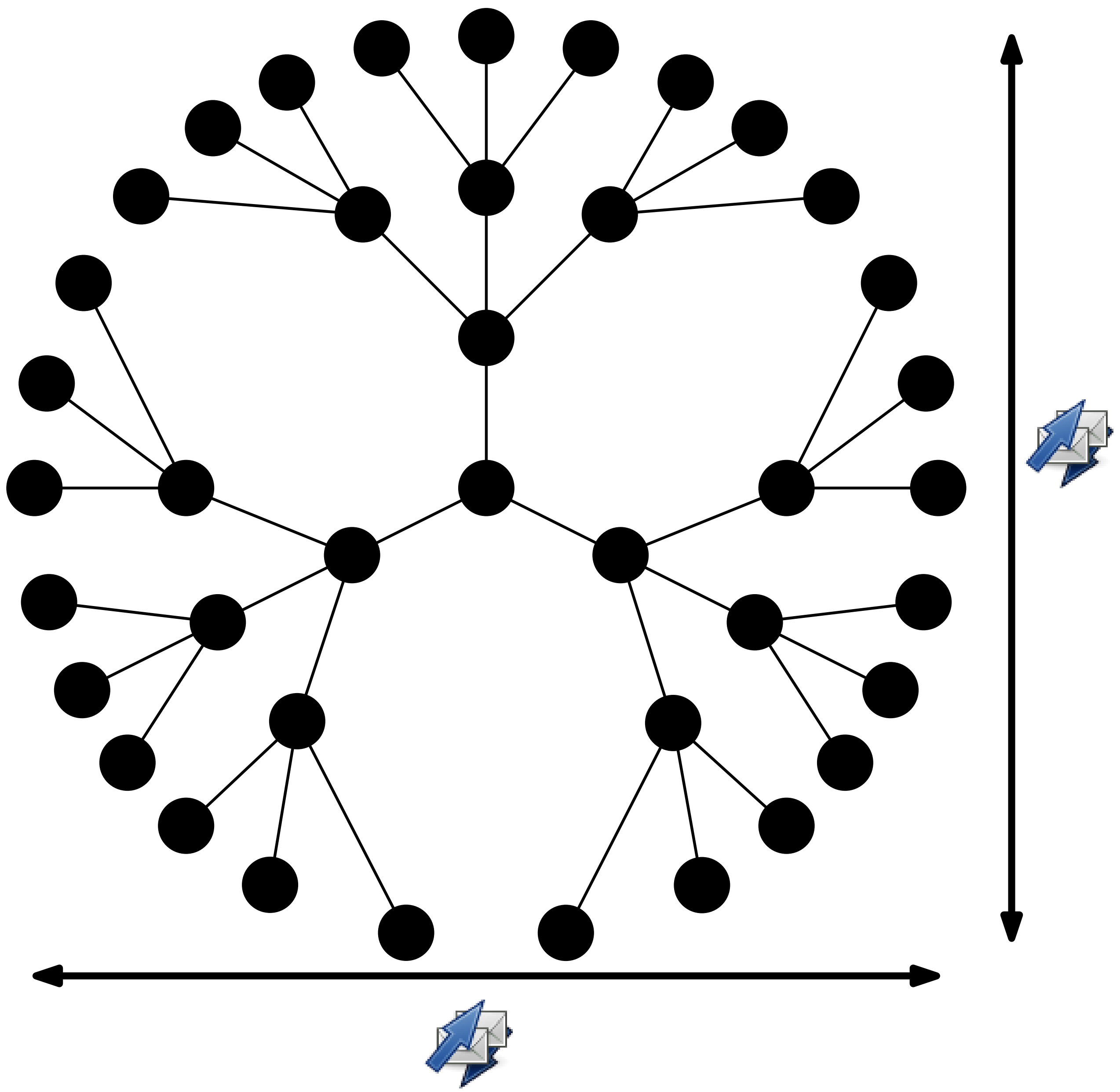
  - graph $G = (V, E)$ with $|V| = n$
  - E: communication links
  - each node in V runs the same algorithm

- **Time is synchronous**: nodes alternate

  - arbitrary local computation & update of state variables
  - sending of messages to all neighbors
    * no bandwidth constraints

- **Unique identifiers** to nodes in the set $1, \ldots, \mathrm{poly}(n)$
  * adversarially chosen * n is known to the nodes
  - needed to solve even basic problems (2-coloring a 2-path)

- **Port numbering**: adversarially chosen in $\{1, \ldots, \Delta\}$



The LOCAL model

[Linial FOCS '87 & SICOMP '92]

- **Distributed network** of n processors/nodes

  - graph $G = (V, E)$ with $|V| = n$
  - E: communication links
  - each node in V runs the same algorithm

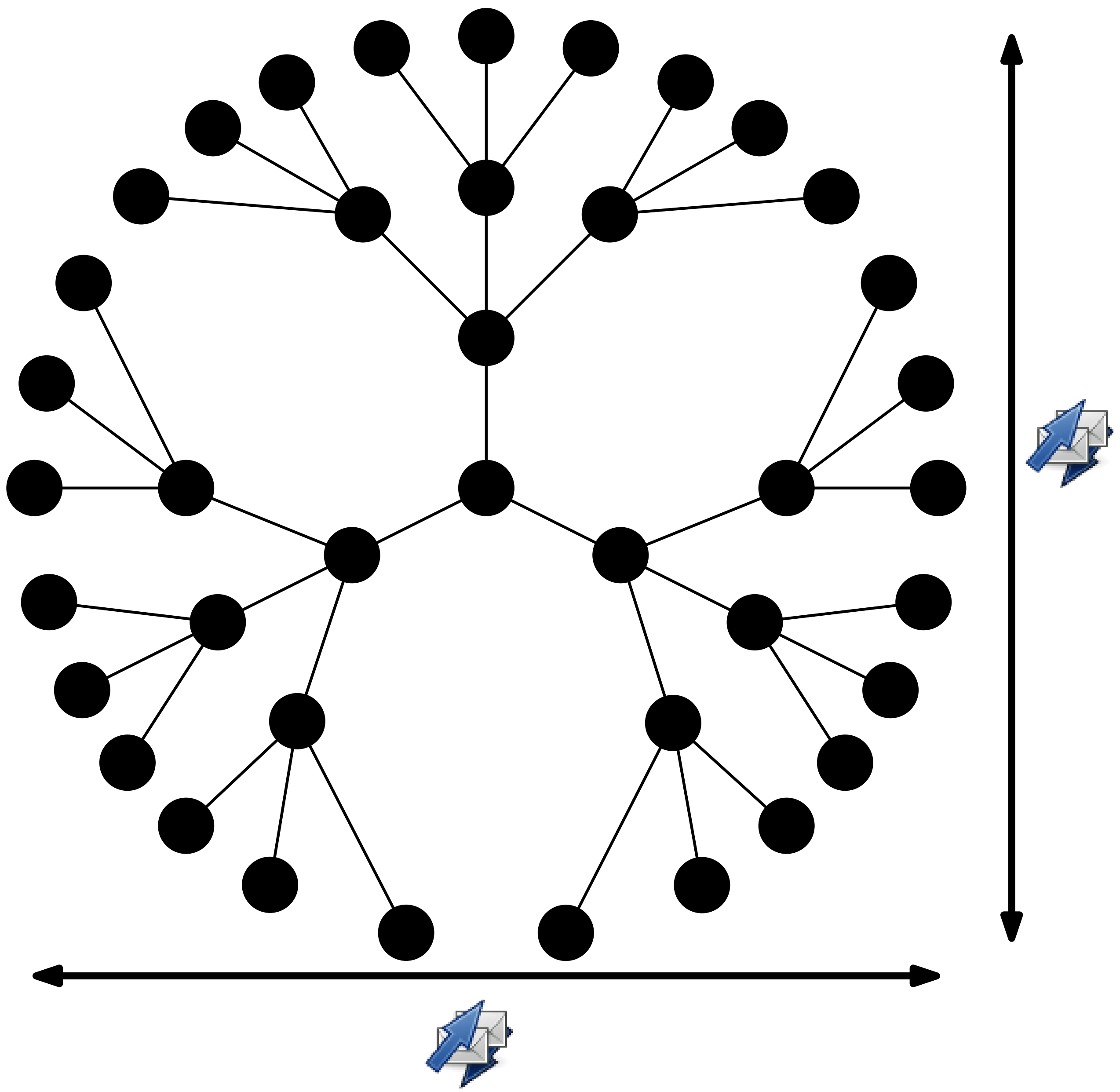
- **Time is synchronous**: nodes alternate

  - arbitrary local computation & update of state variables
  - sending of messages to all neighbors
    * no bandwidth constraints

- **Unique identifiers** to nodes in the set $1, \ldots, \text{poly}(n)$
  * adversarially chosen * n is known to the nodes
  - needed to solve even basic problems (2-coloring a 2-path)

- **Port numbering**: adversarially chosen in $\{1, \ldots, \Delta\}$

- **Possible randomness**: i.i.d. infinite random bit strings to nodes



The LOCAL model

[Linial FOCS '87 & SICOMP '92]

- **Distributed network** of n processors/nodes

  - graph $G = (V, E)$ with $|V| = n$
  - E: communication links
  - each node in V runs the same algorithm

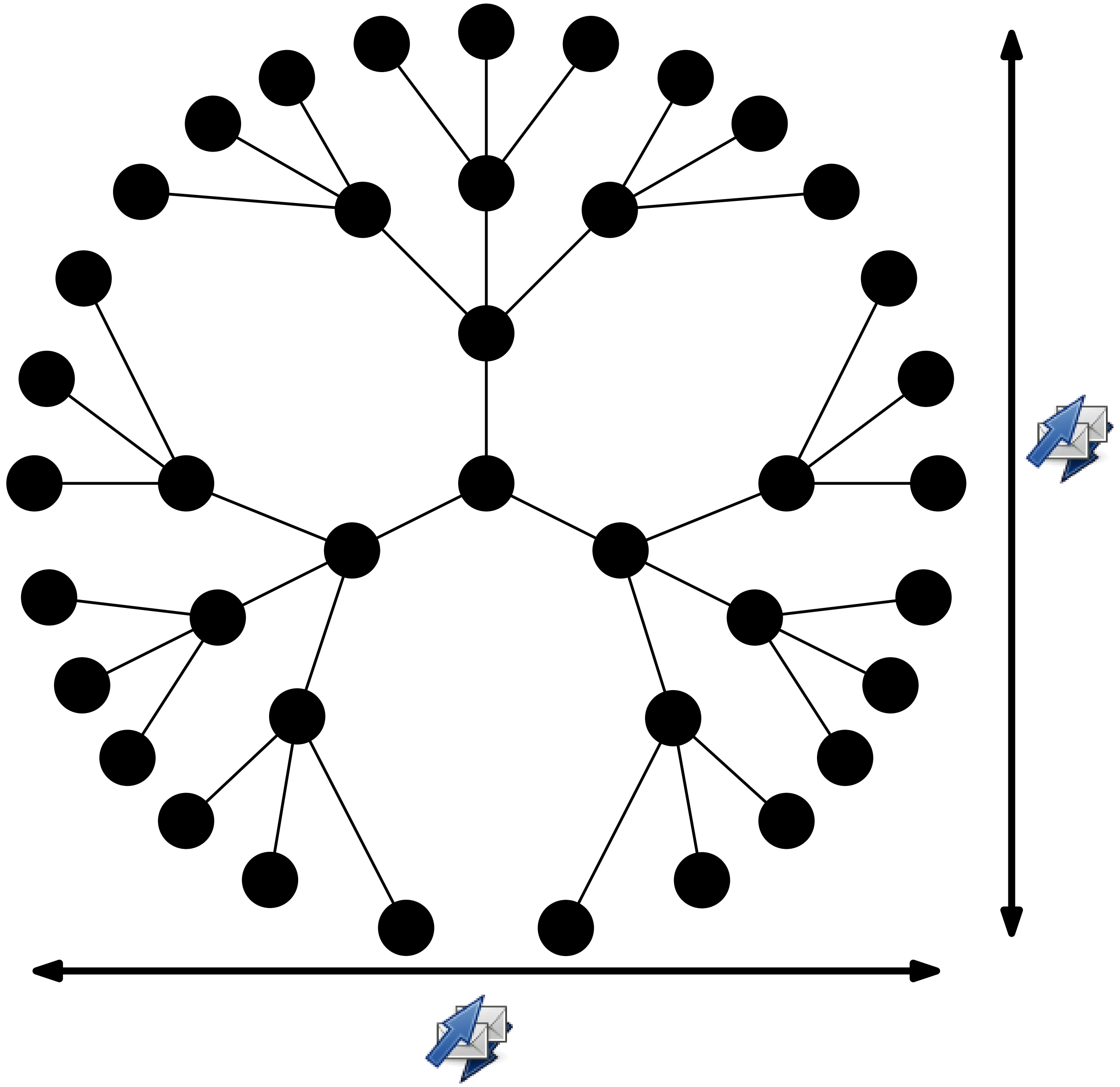
- **Time is synchronous**: nodes alternate

  - arbitrary local computation & update of state variables
  - sending of messages to all neighbors
    * no bandwidth constraints

- ~~**Unique identifiers** to nodes in the set $1, \ldots, \mathrm{poly}(n)$~~
  * adversarially chosen * n is known to the nodes
  - needed to solve even basic problems (2-coloring a 2-path)

- **Port numbering**: adversarially chosen in $\{1, \ldots, \Delta\}$

- **Possible randomness**: i.i.d. infinite random bit strings to nodes



The LOCAL model

[Linial FOCS '87 & SICOMP '92]

- **Distributed network** of n processors/nodes

  - graph $G = (V, E)$ with $|V| = n$
  - E: communication links
  - each node in V runs the same algorithm

- **Time is synchronous**: nodes alternate

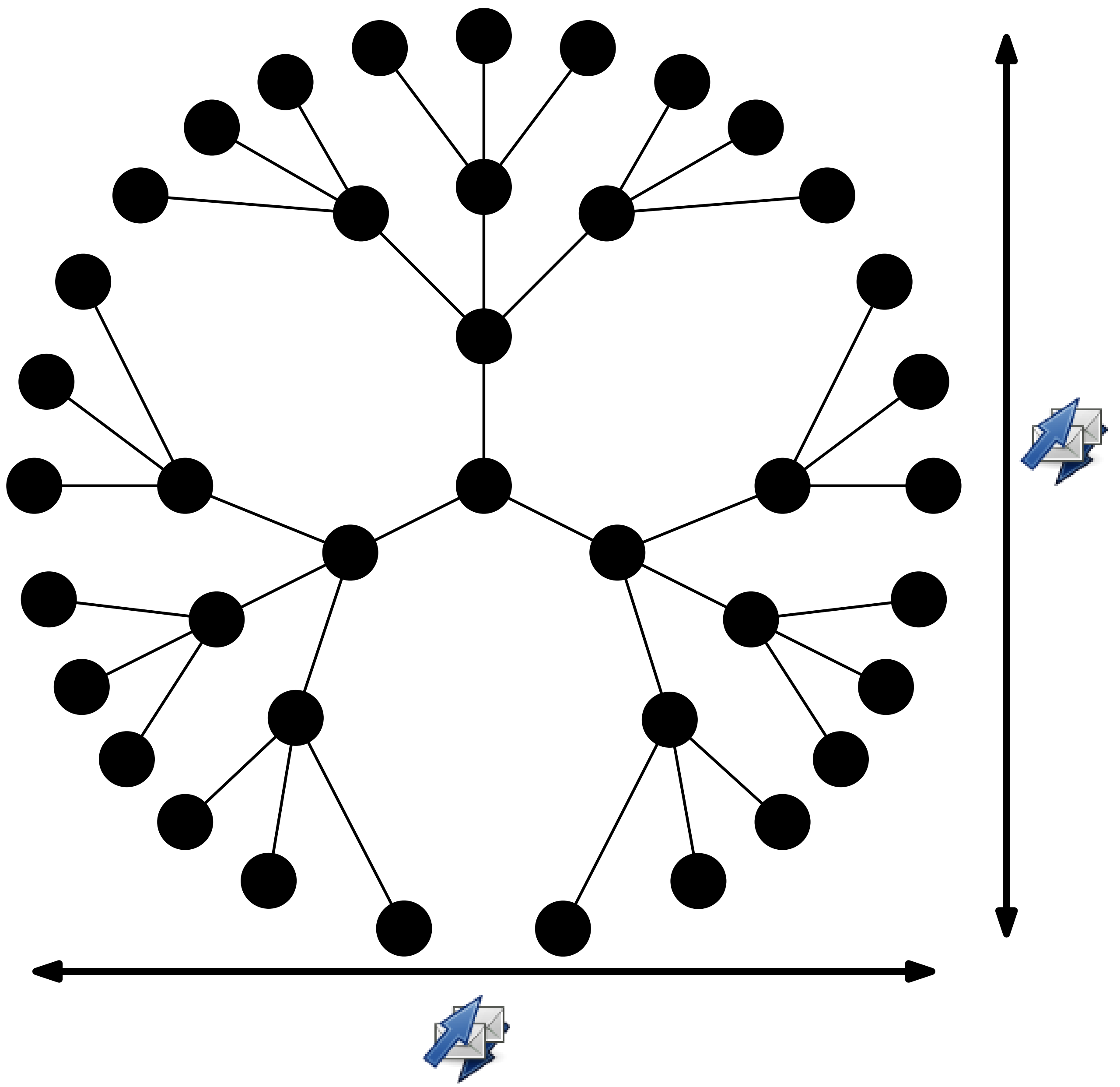
  - arbitrary local computation & update of state variables
  - sending of messages to all neighbors
    * no bandwidth constraints

- ~~**Unique identifiers** to nodes in the set $1, \ldots,$ poly(n)~~
  * adversarially chosen * n is known to the nodes
  - needed to solve even basic problems (2-coloring a 2-path)

- **Port numbering**: adversarially chosen in $\{1, \ldots, \Delta\}$

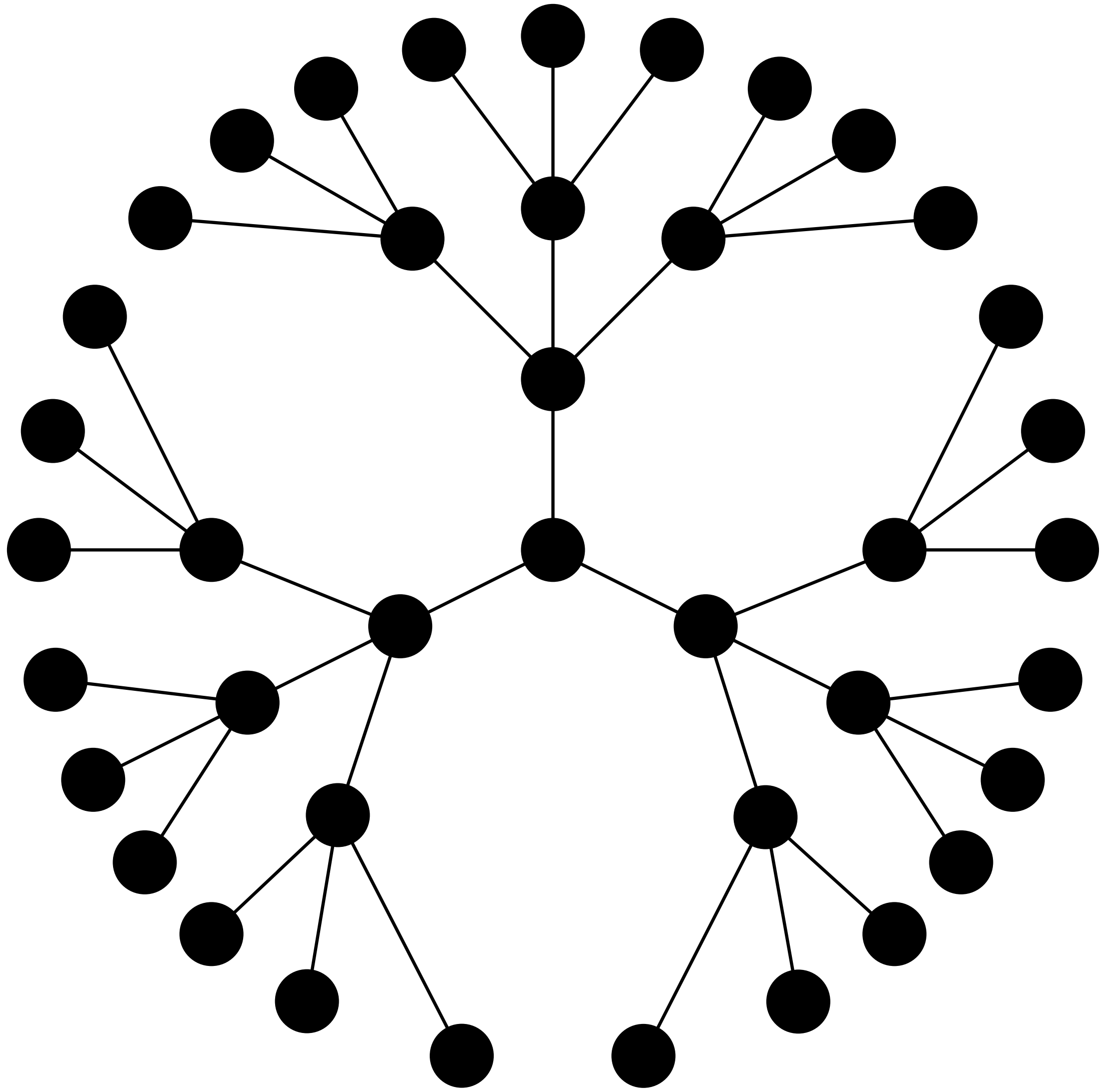
- **Possible randomness**: i.i.d. infinite random bit strings to nodes

- **Complexity measure**: number of communication rounds



Local view

Complexity measure: number of communication rounds

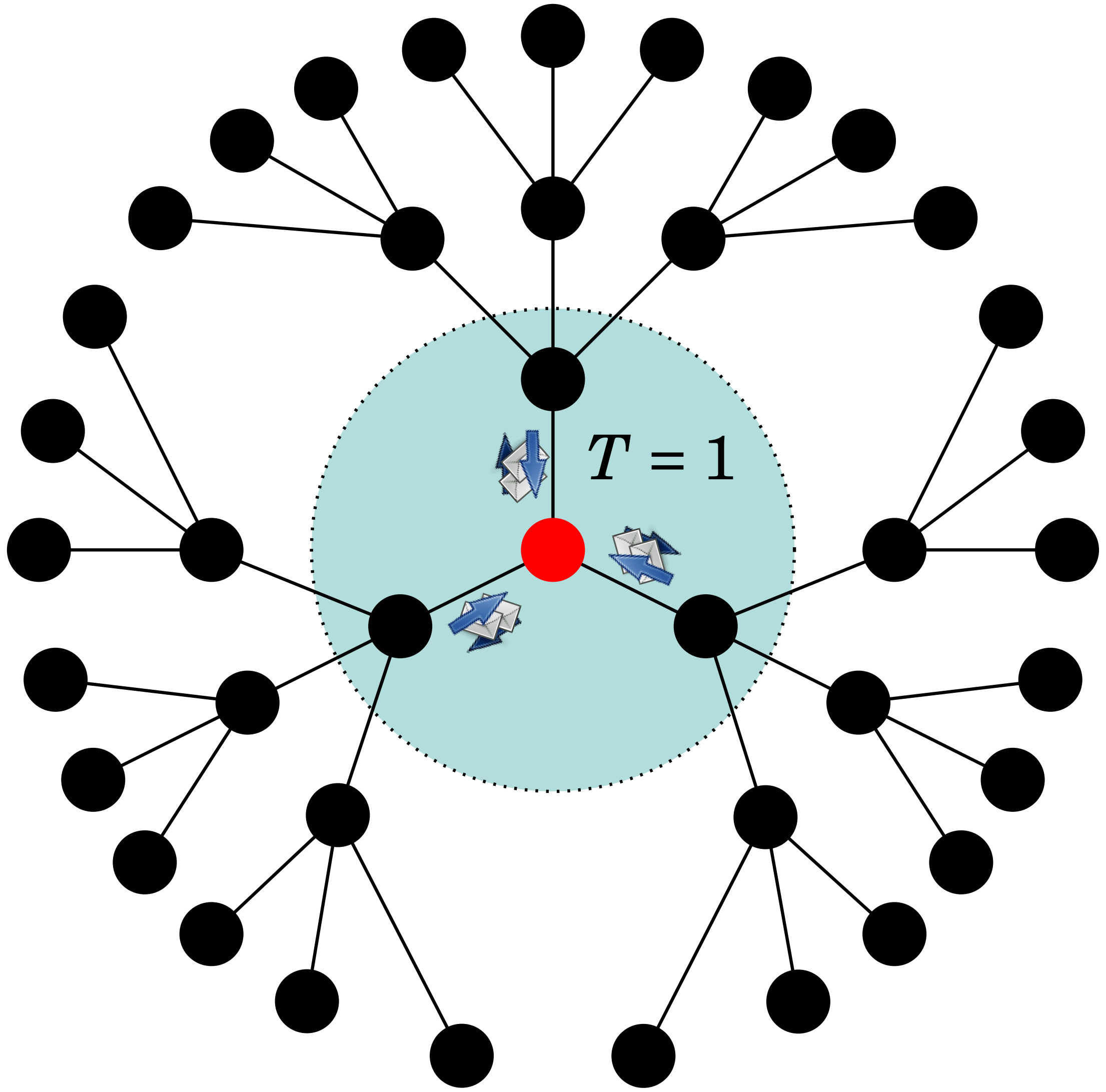


- What do we know after T rounds?

Local view

Complexity measure: number of communication rounds

- What do we know after T rounds?

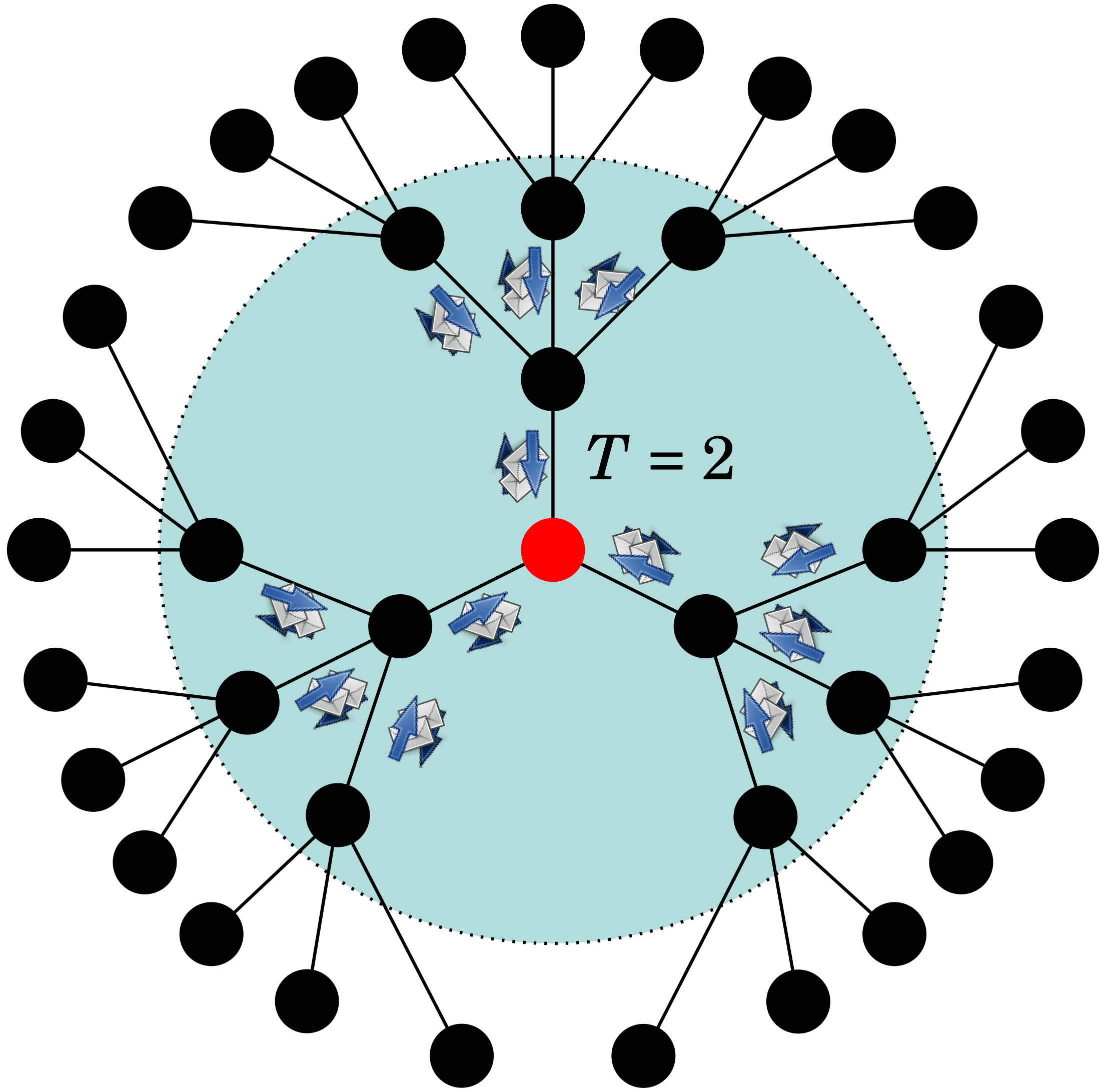


knowledge after T rounds of communication

Local view

Complexity measure: number of communication rounds

- What do we know after T rounds?

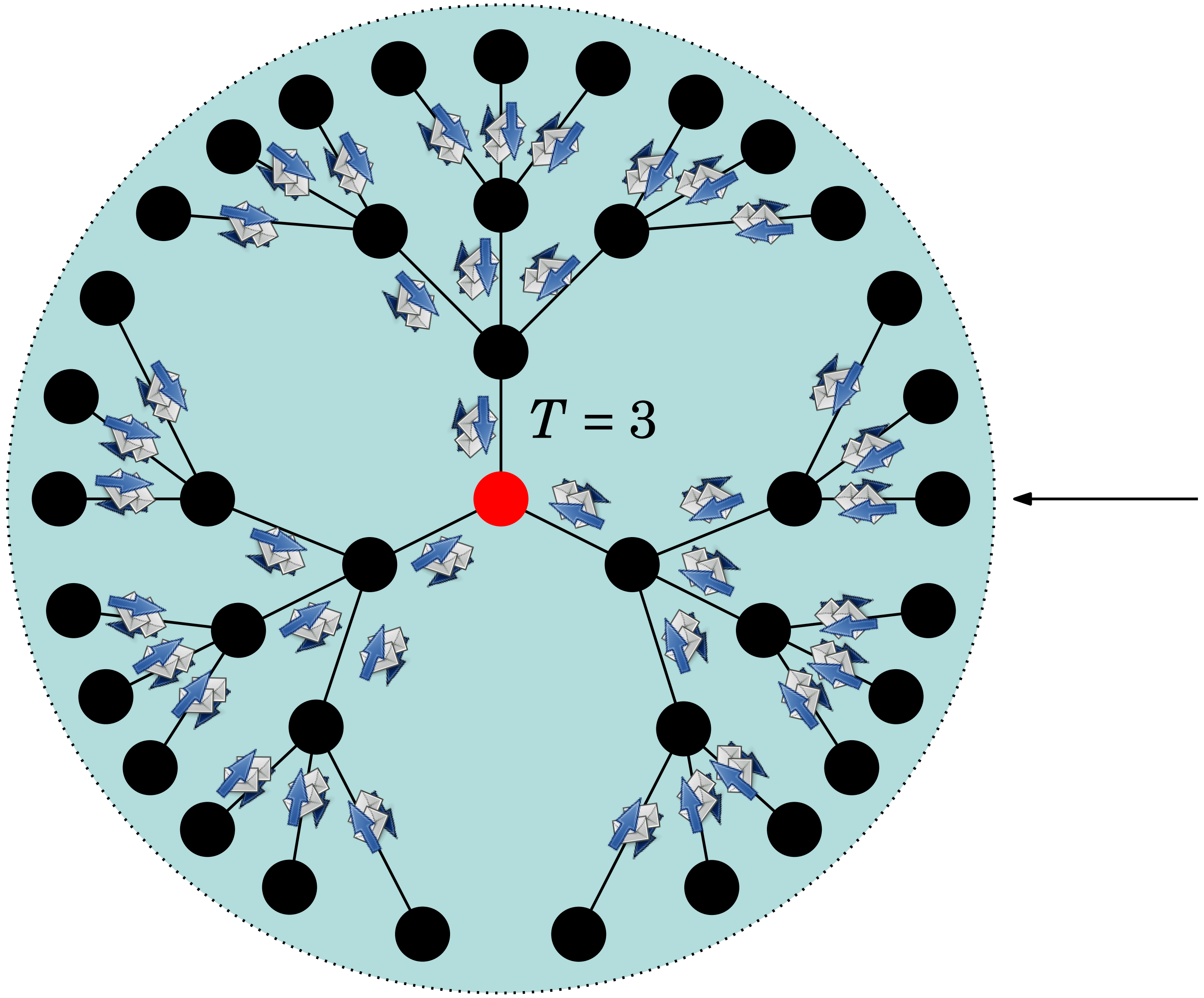


knowledge after T rounds of communication

Local view

Complexity measure: number of communication rounds

- What do we know after T rounds?

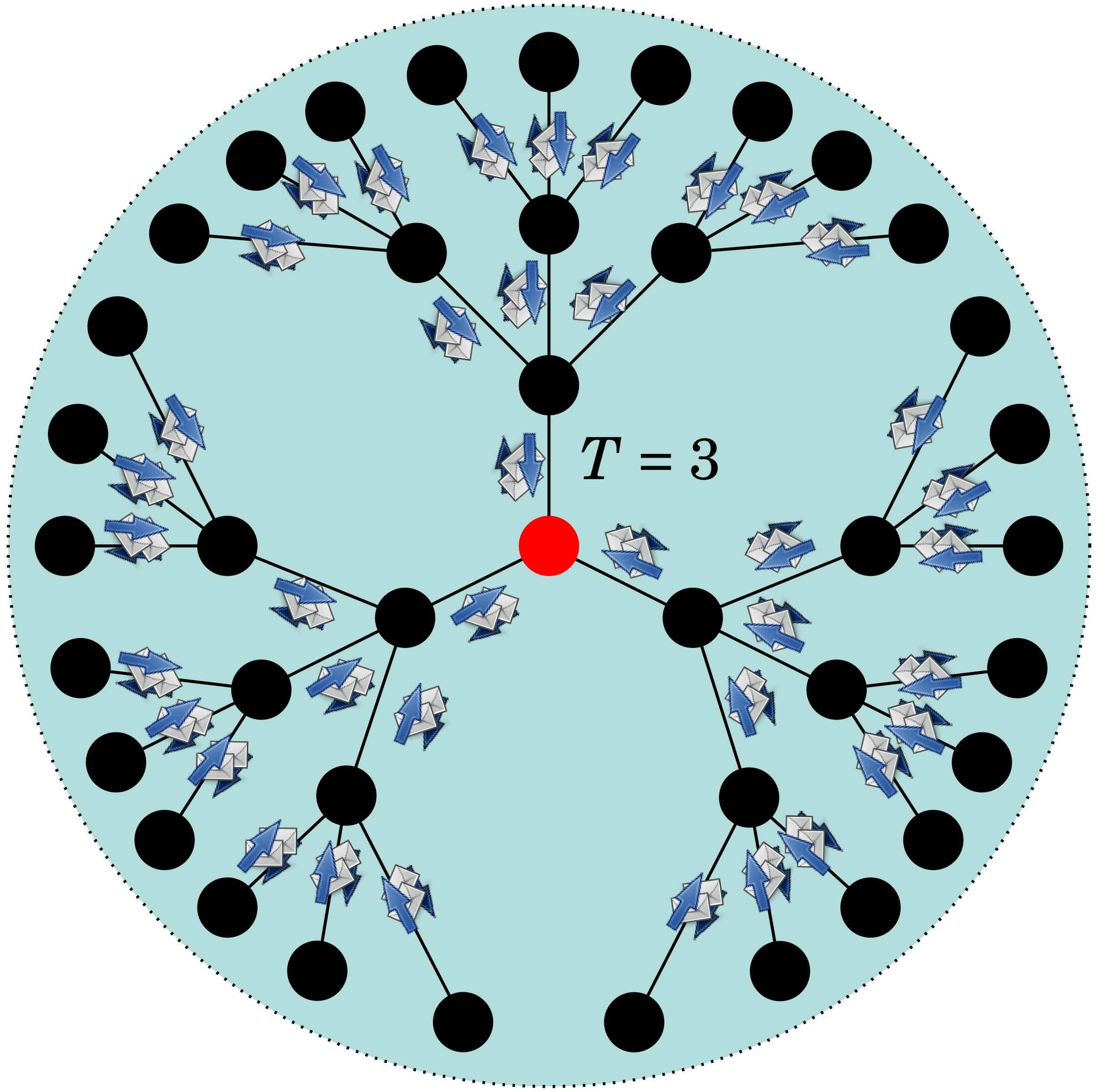


knowledge after T rounds of communication

Local view

Complexity measure: number of communication rounds

- What do we know after T rounds?



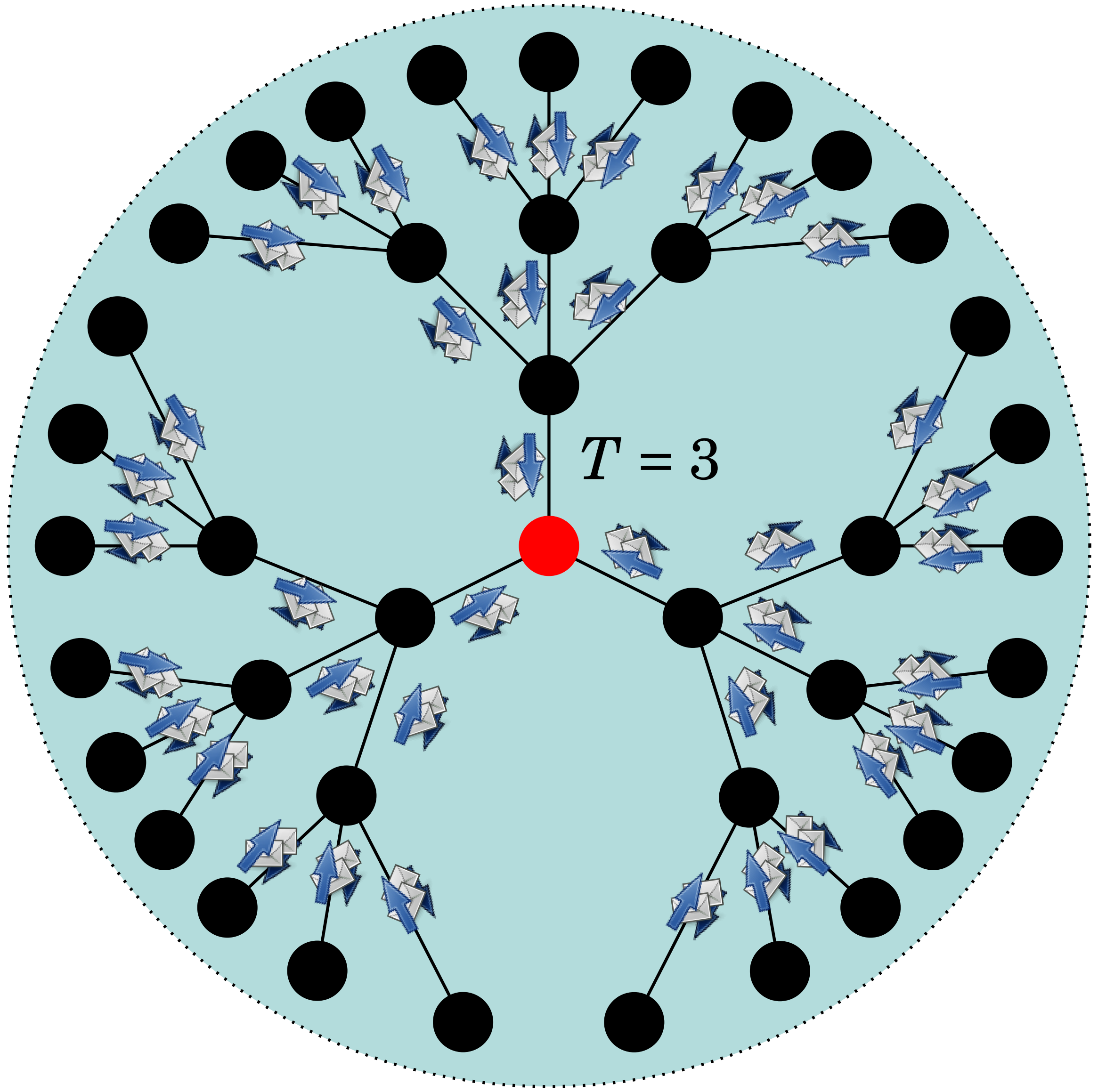
knowledge after T rounds of communication

- **Equivalence**: T-round algorithm ≈ funtion mapping radius-T neighborhoods to local outputs

Local view

Complexity measure: number of communication rounds

- What do we know after T rounds?



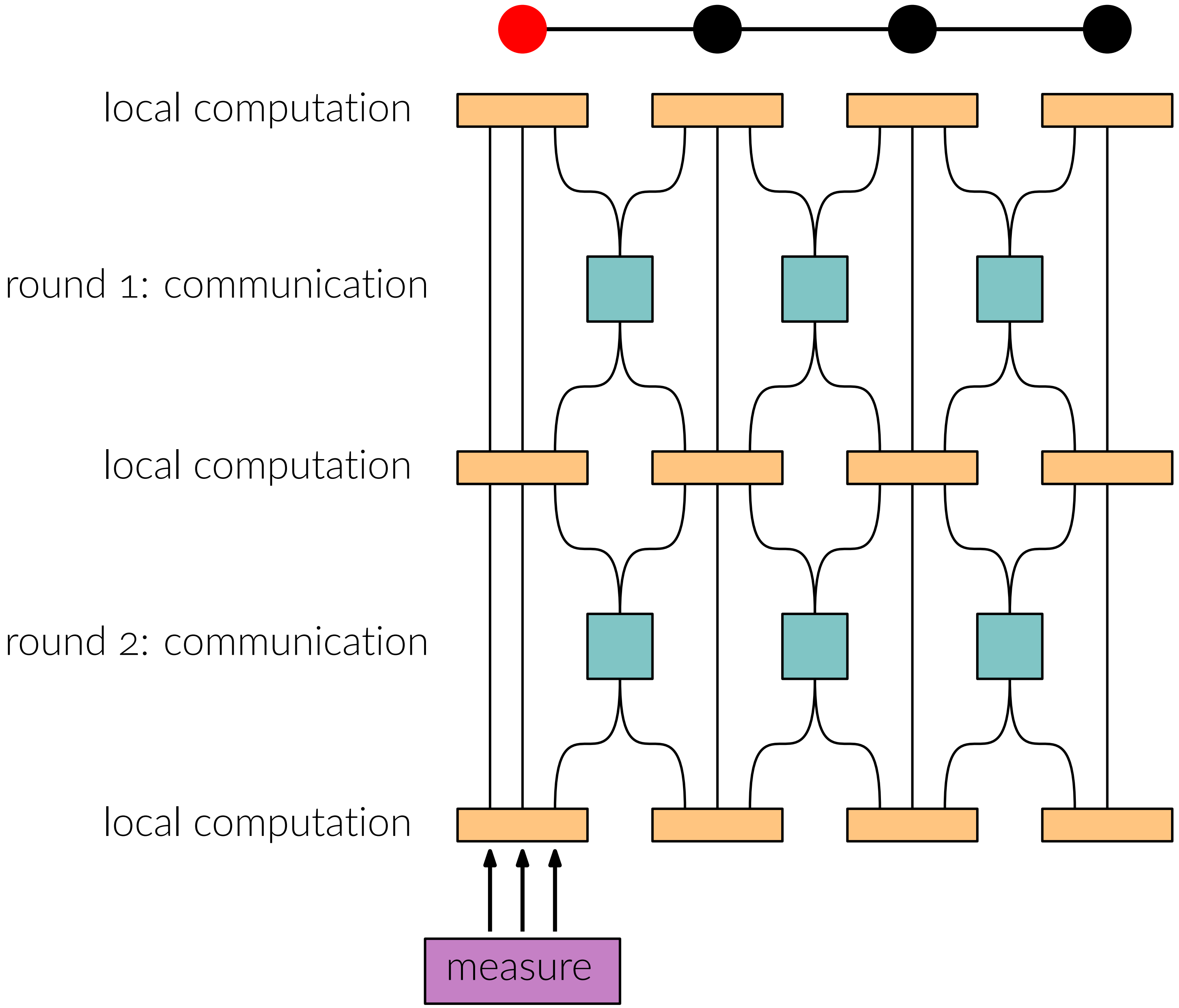
knowledge after T rounds of communication

- **Equivalence**: T-round algorithm ≈ funtion mapping radius-T neighborhoods to local outputs
- Locality $T = \mathrm{diam}(G) + 1$ is always sufficient to solve any problem: gathering algorithm

Quantum-LOCAL

[Gavoille et al., DISC '09]

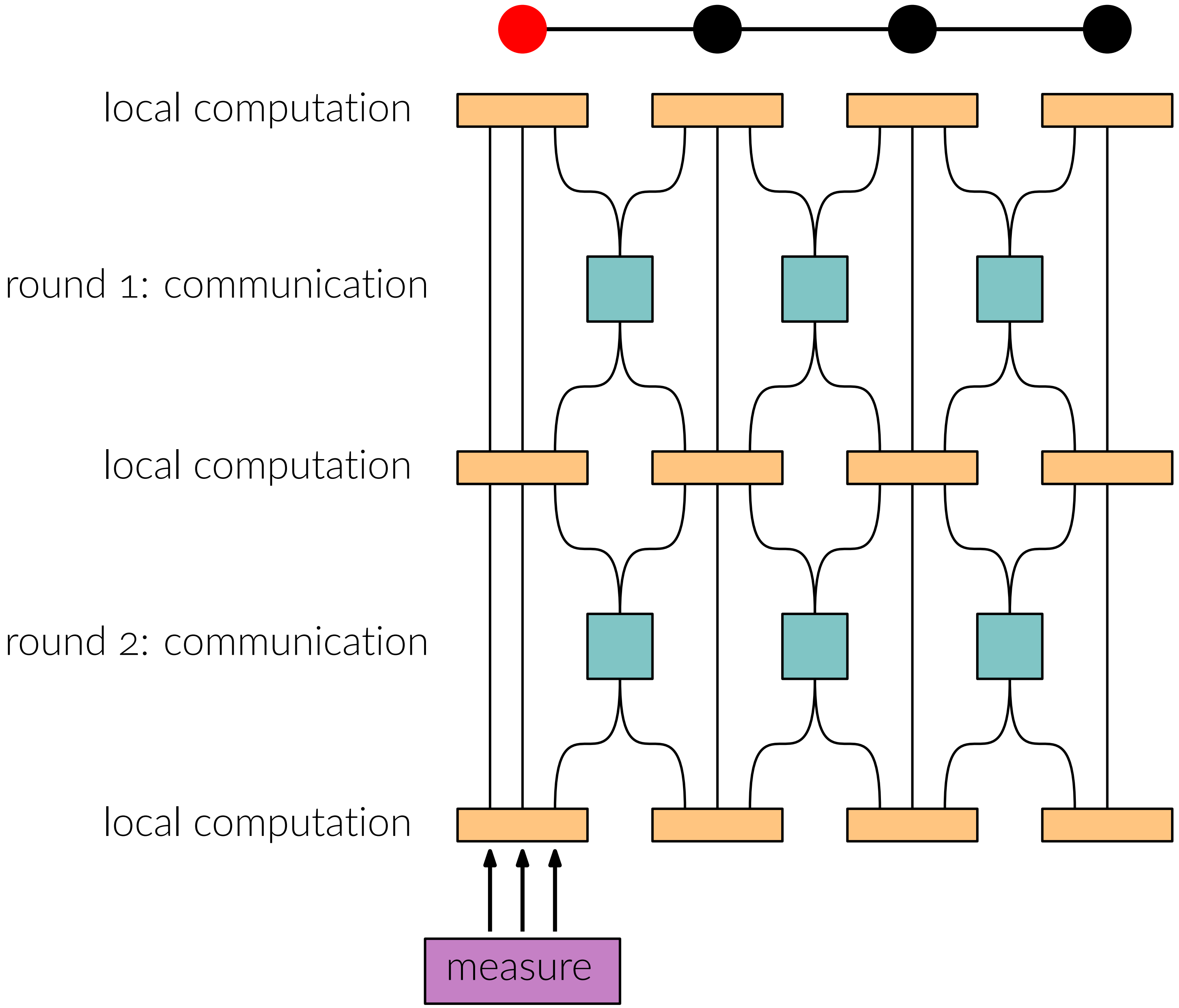
- **Distributed system** of n quantum processors/nodes
  - quantum computation
  - quantum communication (qubits)
  - output: measurement of qubits



Quantum-LOCAL

[Gavoille et al., DISC '09]

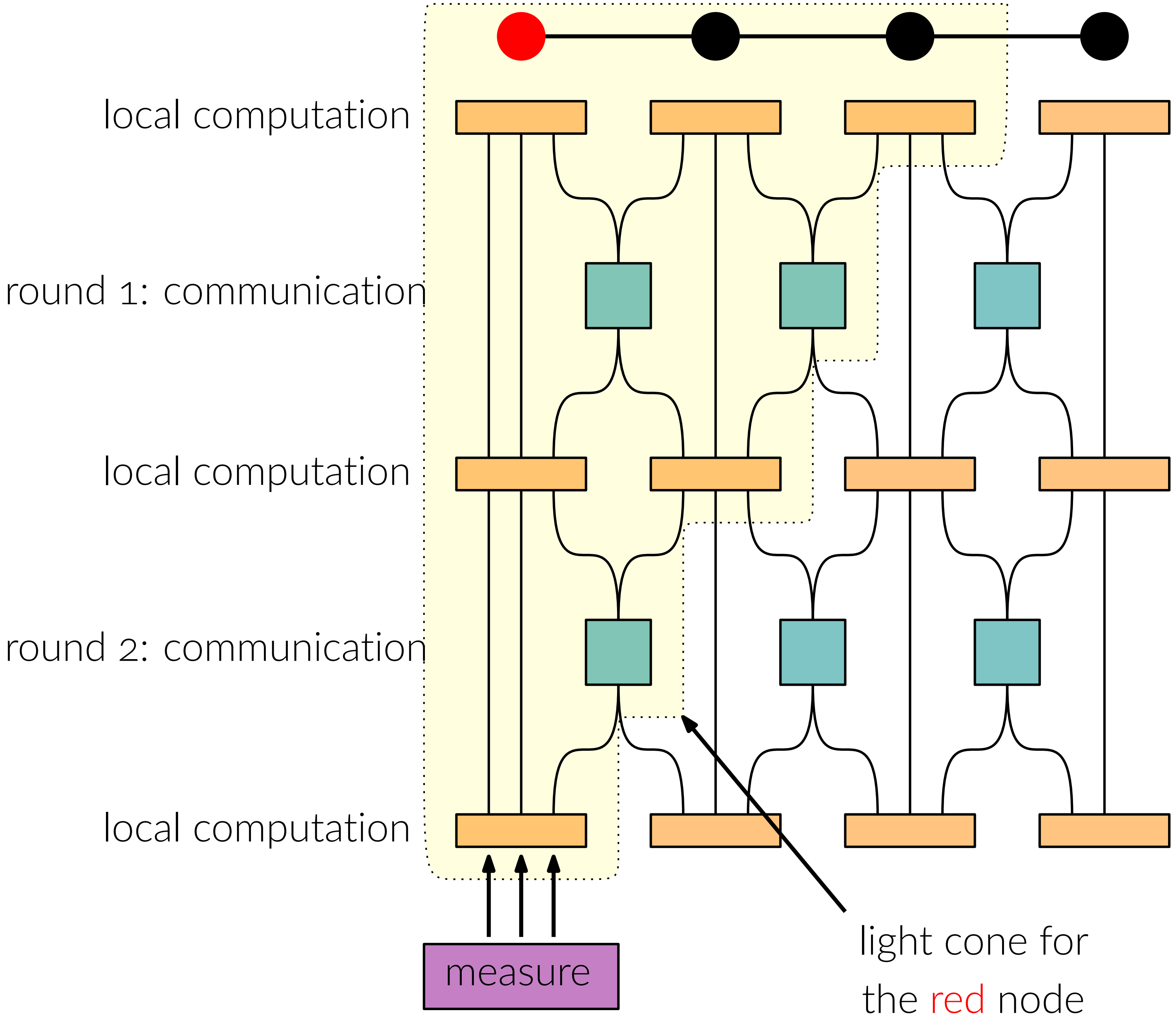
- **Distributed system** of n quantum processors/nodes
  - quantum computation
  - quantum communication (qubits)
  - output: measurement of qubits
- **Complexity measure**: number of communication rounds



Quantum-LOCAL

[Gavoille et al., DISC '09]

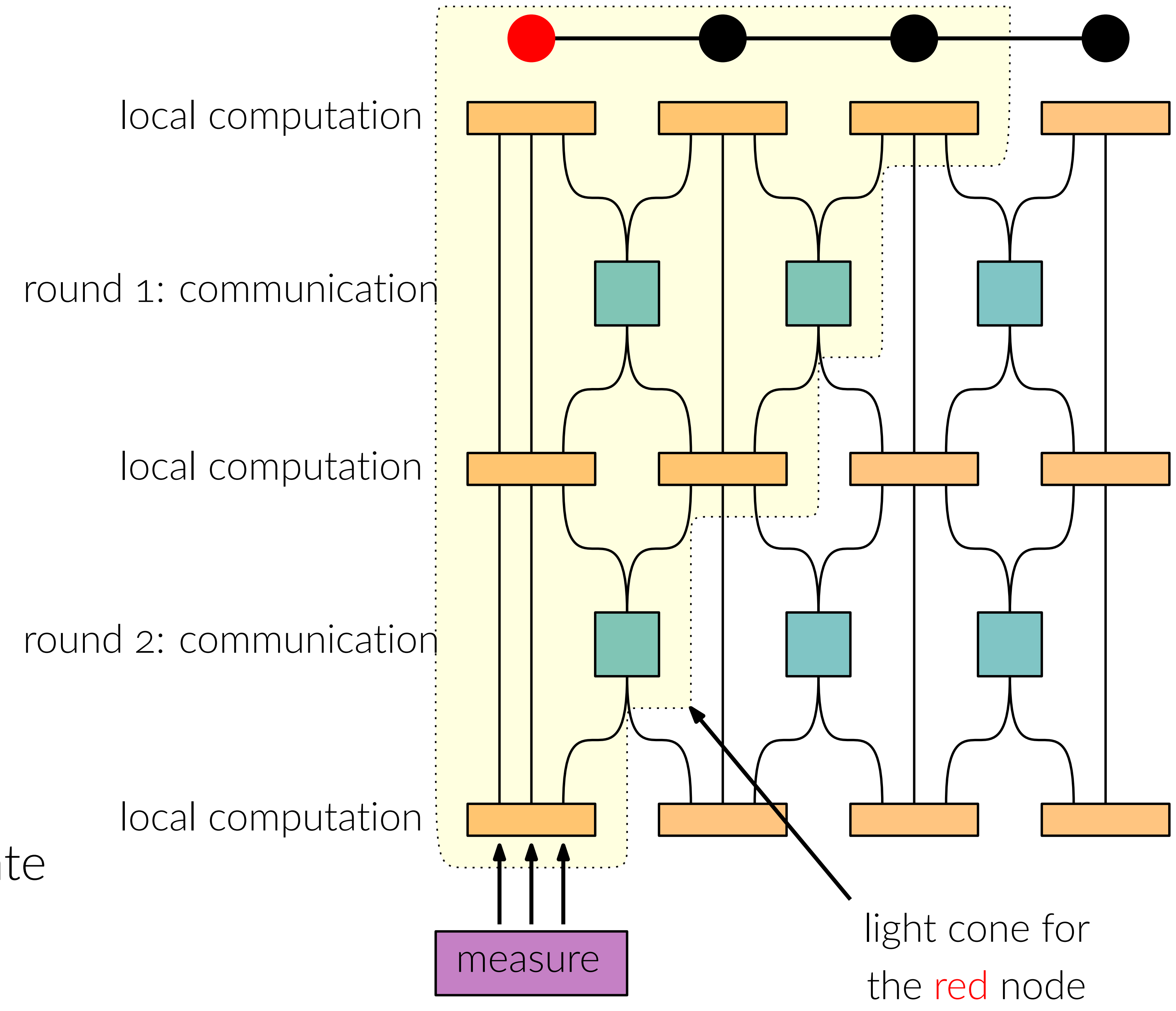
- **Distributed system** of n quantum processors/nodes
  - quantum computation
  - quantum communication (qubits)
  - output: measurement of qubits
- **Complexity measure**: number of communication rounds
- **Gathering algorithms are *weaker***! (More next slide)
  - measuring to clone "corrupts" the quantum state
  - quantum states cannot be *cloned* (no-cloning theorem)



Quantum-LOCAL

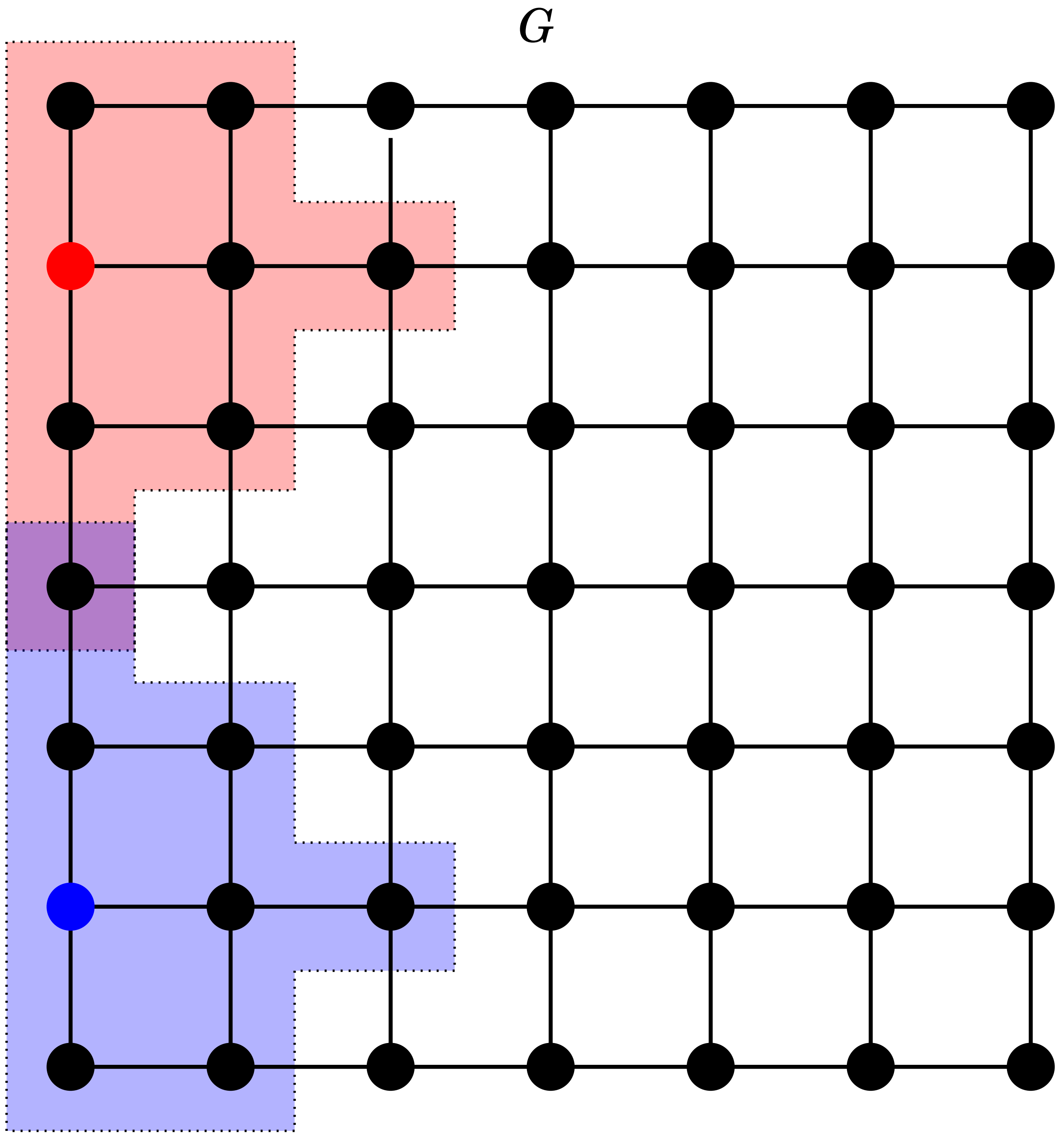
[Gavoille et al., DISC '09]

- **Distributed system** of n quantum processors/nodes
  - quantum computation
  - quantum communication (qubits)
  - output: measurement of qubits
- **Complexity measure**: number of communication rounds
- **Gathering algorithms are *weaker***! (More next slide)
  - measuring to clone "corrupts" the quantum state
  - quantum states cannot be *cloned* (no-cloning theorem)
- Still, locality identifies how *far* nodes need to communicate



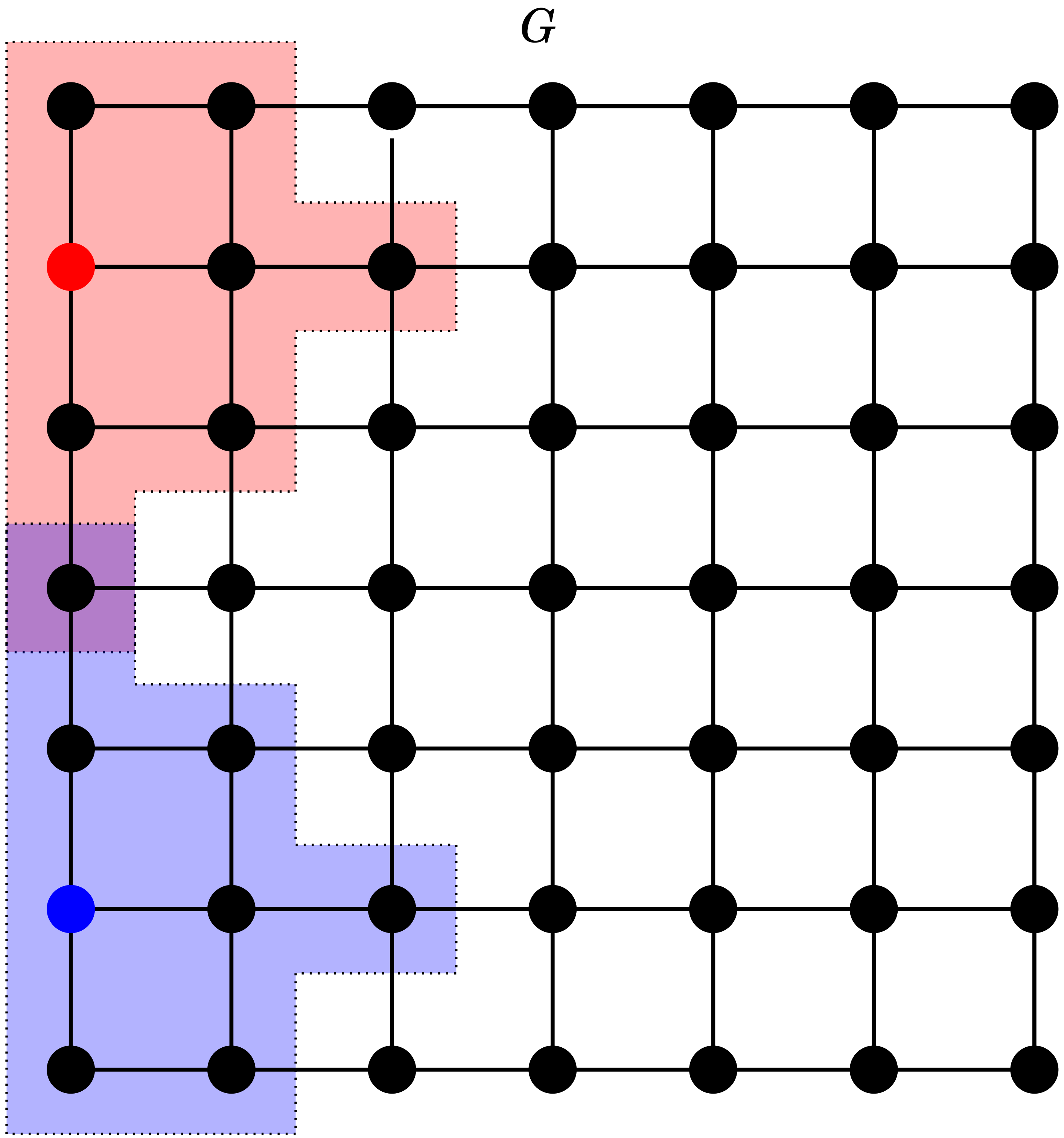
Quantum-LOCAL

- Consider a $\mathbf{2}$-round algorithm $\boldsymbol{A}$ in $\boldsymbol{G}$



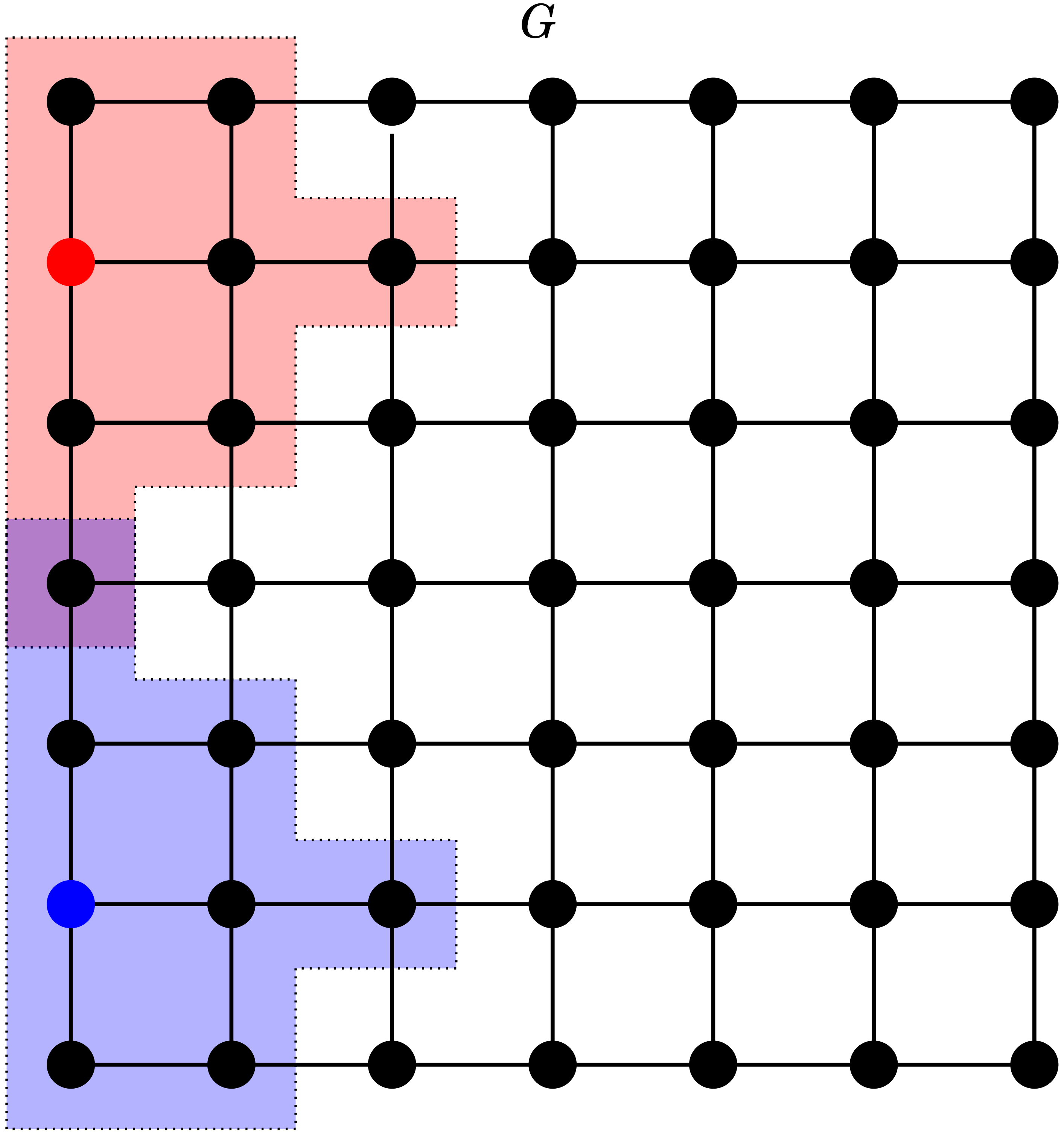
Quantum-LOCAL

- Consider a $\mathbf{2}$-round algorithm $\boldsymbol{A}$ in $\boldsymbol{G}$
  - *classical LOCAL*: output given by function of local views
    - given random bits, deterministic function



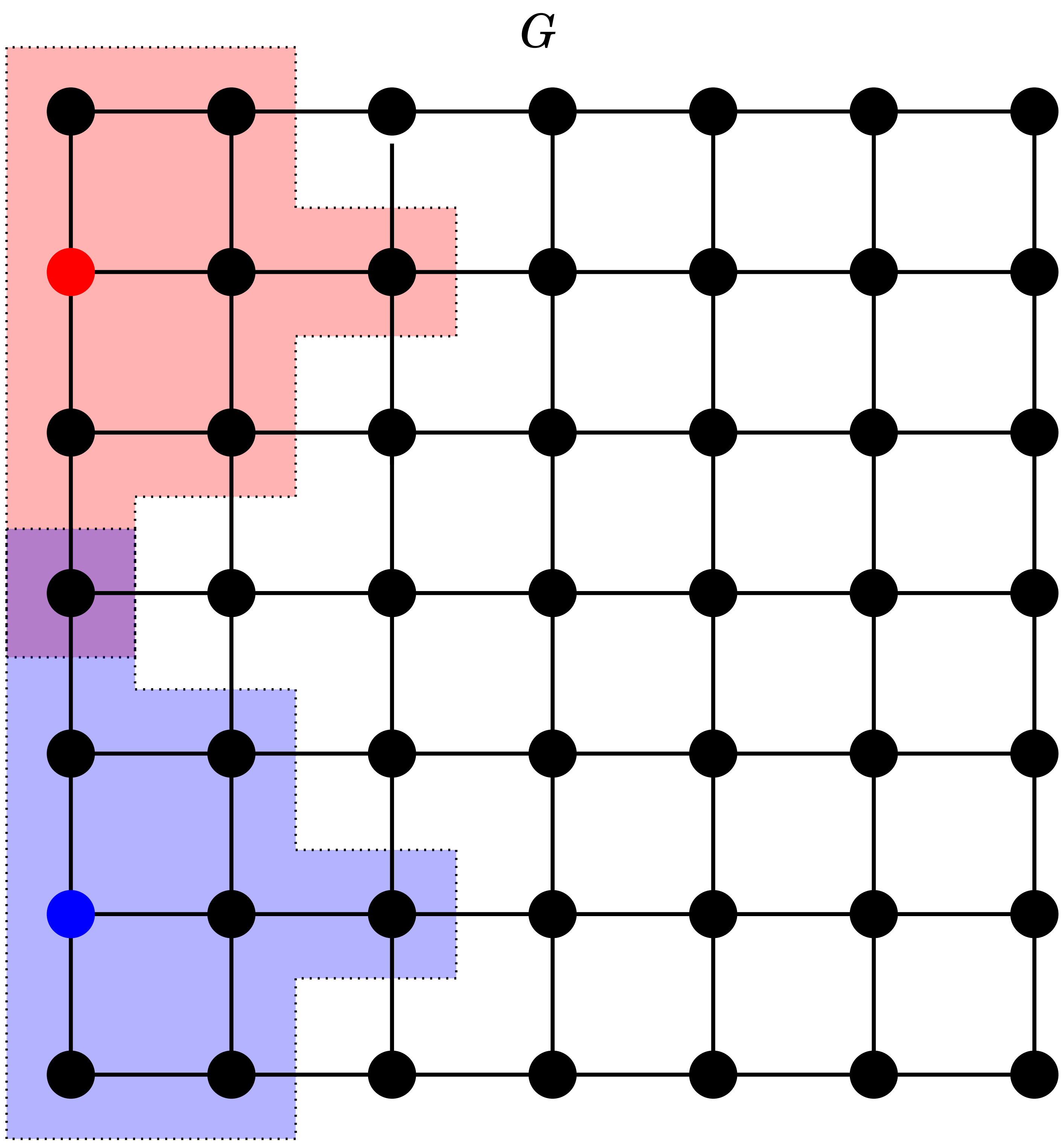
Quantum-LOCAL

- Consider a $\mathbf{2}$-round algorithm $\boldsymbol{A}$ in $\boldsymbol{G}$
  - *classical LOCAL*: output given by function of local views
    - given random bits, deterministic function
  - *quantum LOCAL: ??*
    - given quantum bits??



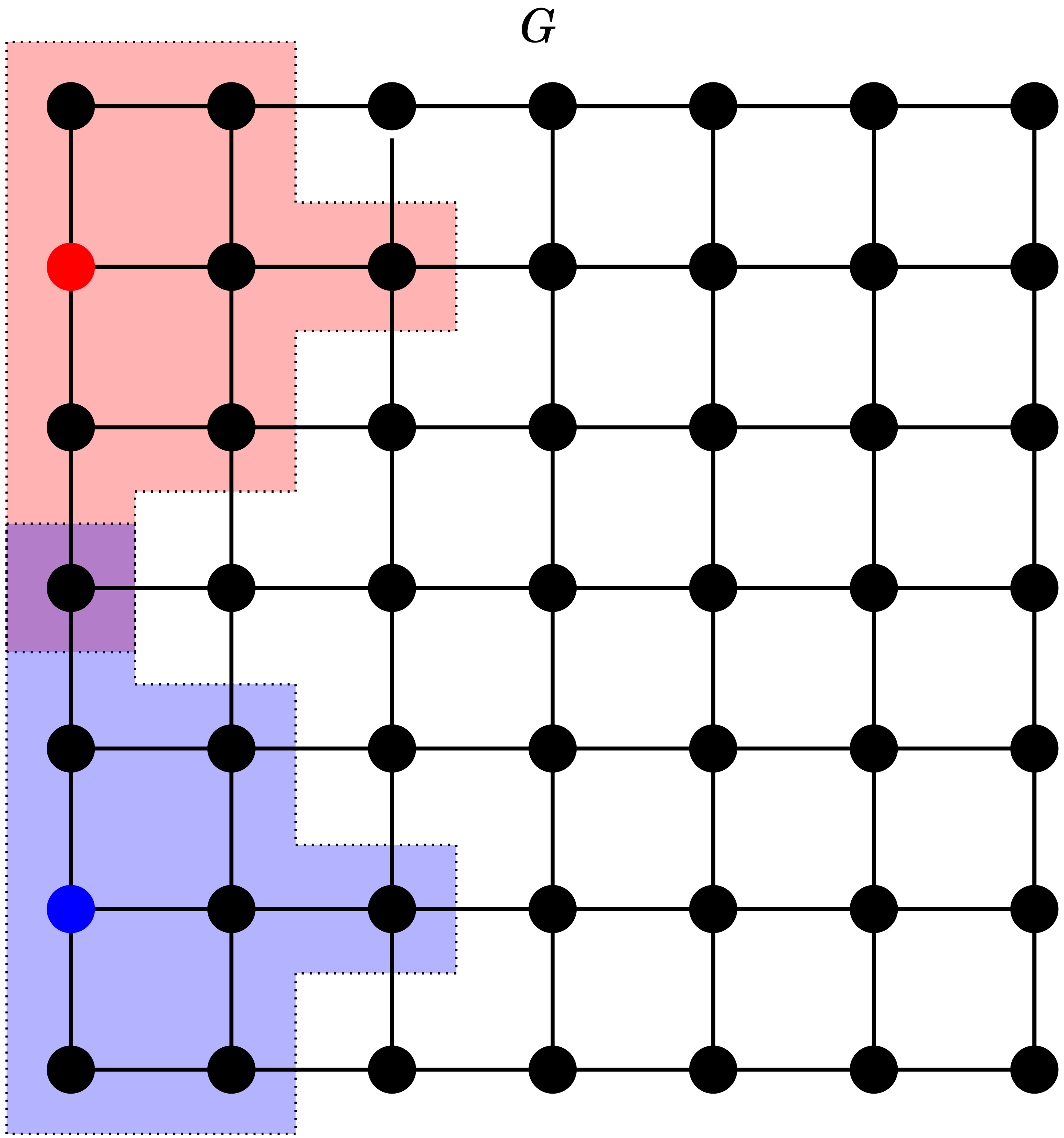
Quantum-LOCAL

- Consider a $\mathbf{2}$-round algorithm $\boldsymbol{A}$ in $\boldsymbol{G}$
  - *classical LOCAL*: output given by function of local views
    - given random bits, deterministic function
  - *quantum LOCAL: ??*
    - given quantum bits??
- **No-cloning**



Quantum-LOCAL

- Consider a $\mathbf{2}$-round algorithm $\boldsymbol{A}$ in $\boldsymbol{G}$
  - *classical LOCAL*: output given by function of local views
    - given random bits, deterministic function
  - *quantum LOCAL: ??*
    - given quantum bits??
- **No-cloning**
- Still, we can describe *output distributions*



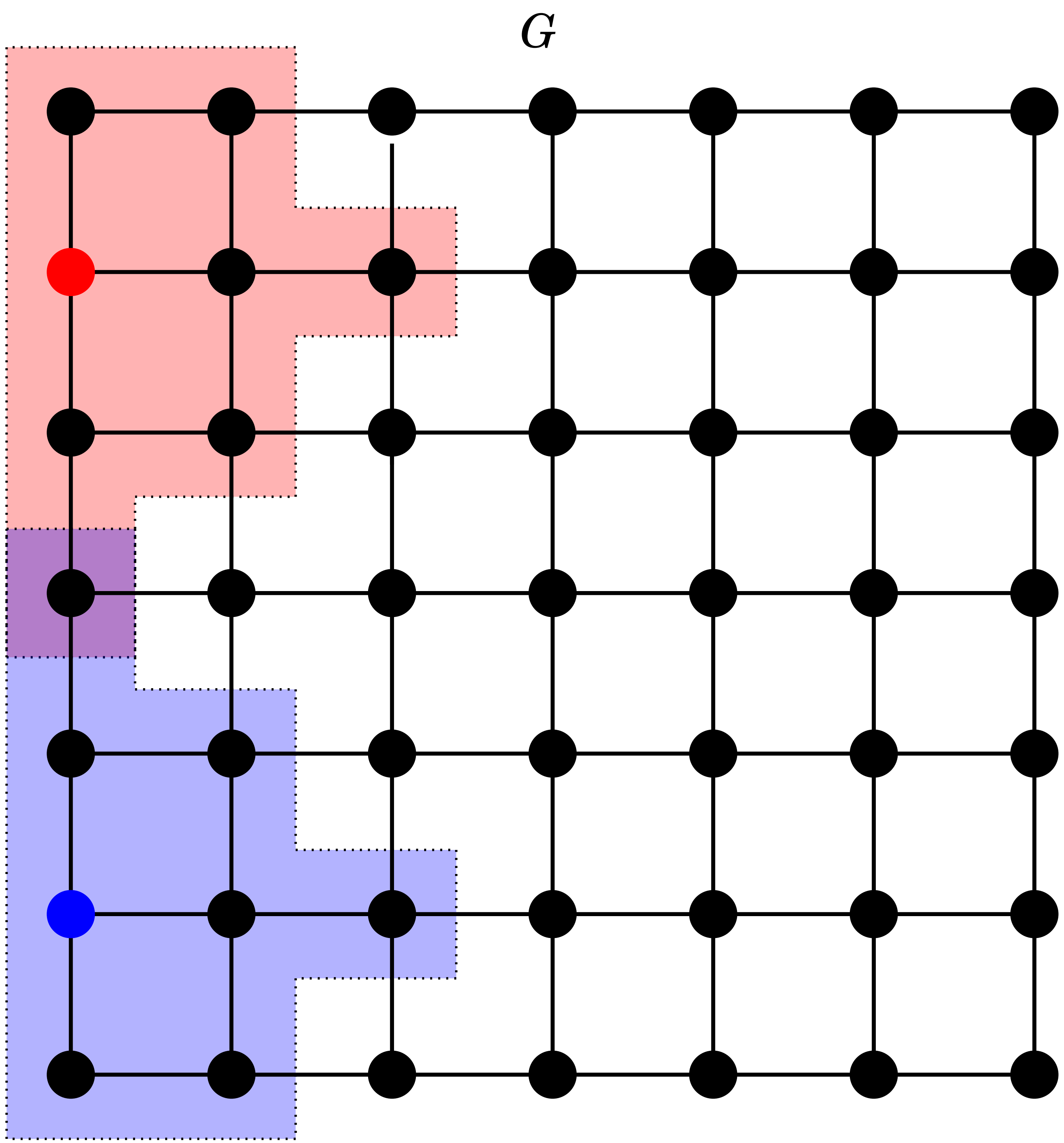
Quantum-LOCAL

- Consider a $\mathbf{2}$-round algorithm $\boldsymbol{A}$ in $\boldsymbol{G}$
  - *classical LOCAL*: output given by function of local views
    - given random bits, deterministic function
  - *quantum LOCAL: ??*
    - given quantum bits??

- **No-cloning**

- Still, we can describe *output distributions*

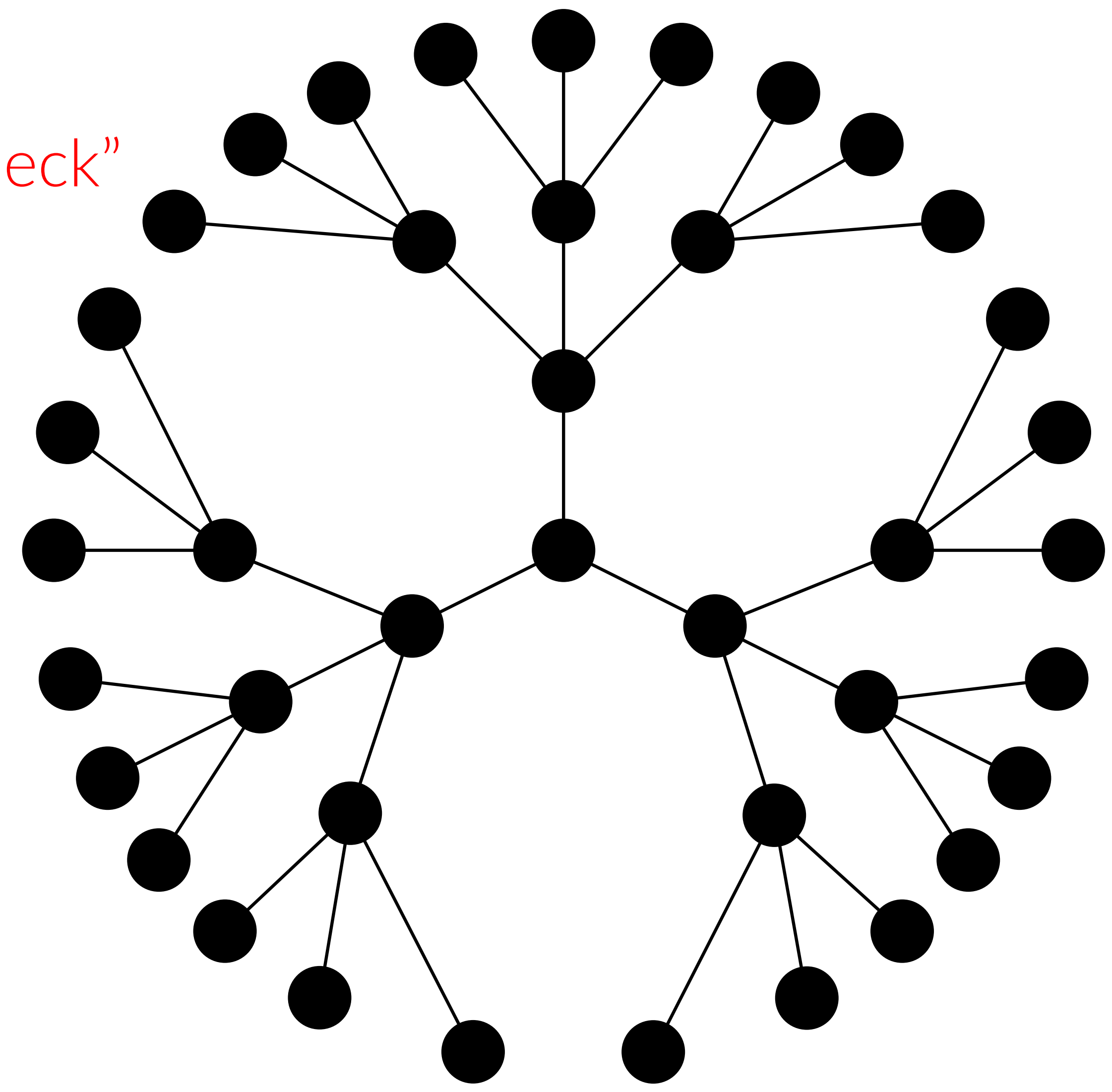
- **Question**: *is there any graph problem that admits quantum advantage?*



Locally checkable labeling (LCL) problems

[Naor and Stockmeyer, STOC '93 & SICOMP '95]

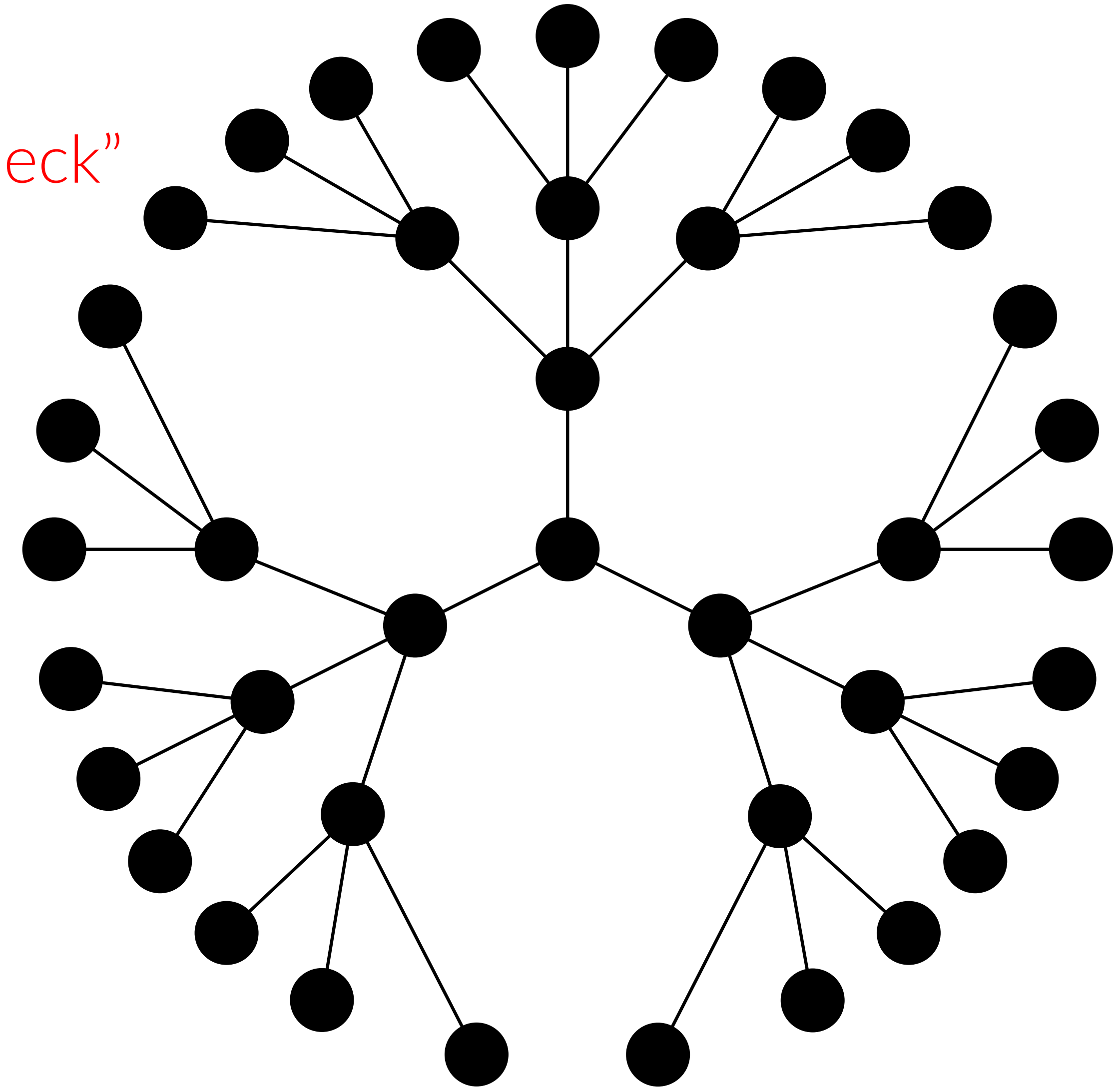
- **Problems** whose solutions might be "hard to find" but are "easy to check"
  - "analogue" of NP in the distributed setting
  - coloring, maximal independent set, maximal matching, etc.



Locally checkable labeling (LCL) problems

[Naor and Stockmeyer, STOC '93 & SICOMP '95]

- **Problems** whose solutions might be "hard to find" but are "easy to check"
  - "analogue" of NP in the distributed setting
  - coloring, maximal independent set, maximal matching, etc.



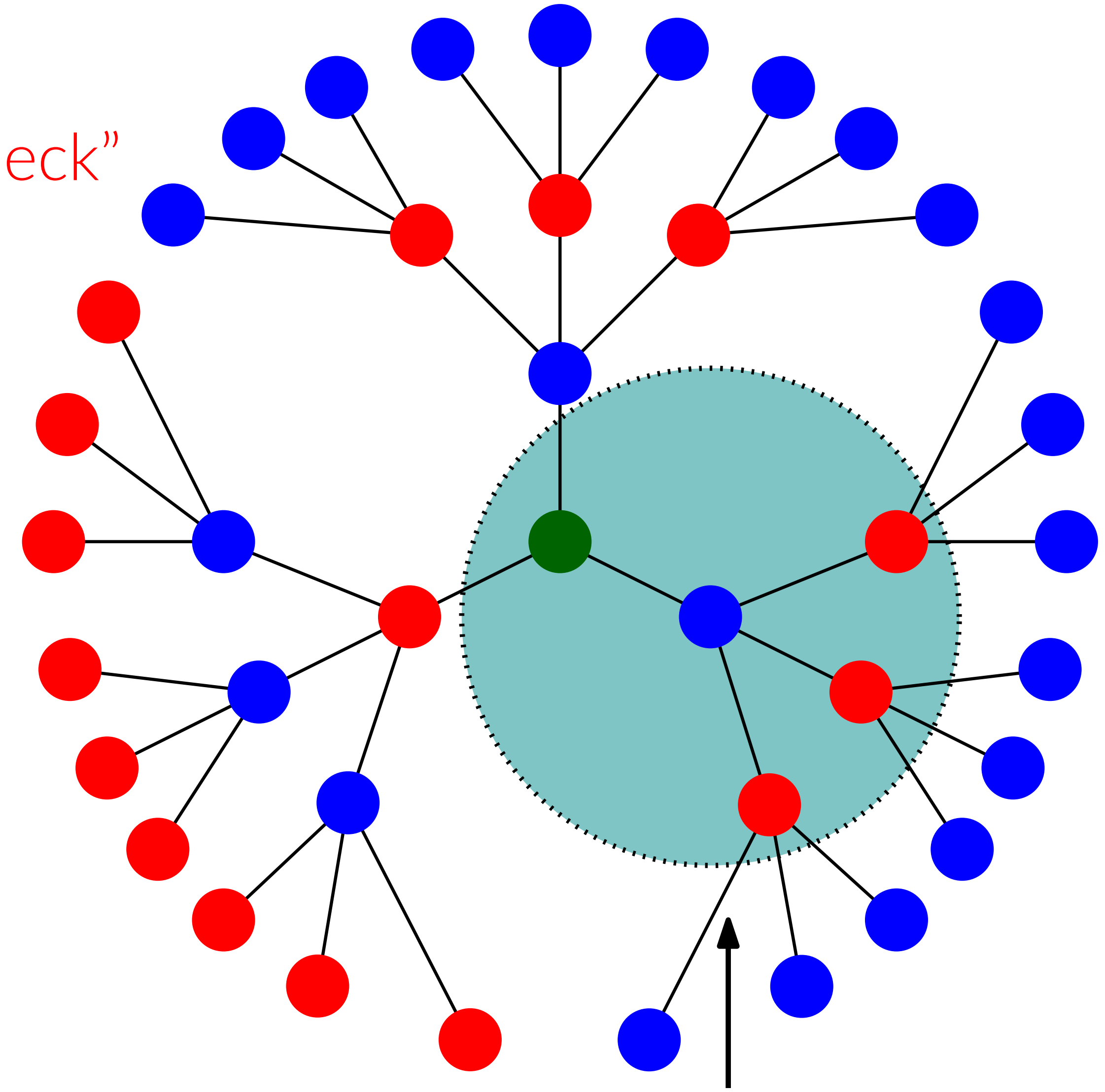
- **"Easy to check"**
  - radius $r = \Theta(1)$
  - each node can check its solution within its radius-r neighborhood
  - a globally valid iff each node is locally happy

Locally checkable labeling (LCL) problems

[Naor and Stockmeyer, STOC '93 & SICOMP '95]

- **Problems** whose solutions might be "hard to find" but are "easy to check"
  - "analogue" of NP in the distributed setting
  - coloring, maximal independent set, maximal matching, etc.

- **"Easy to check"**
  - radius $r = \Theta(1)$
  - each node can check its solution within its radius-r neighborhood
  - a globally valid iff each node is locally happy



3-**coloring**: the blue node checks if its color is different from those of its neighbors

valid LCL

Locally checkable labeling (LCL) problems

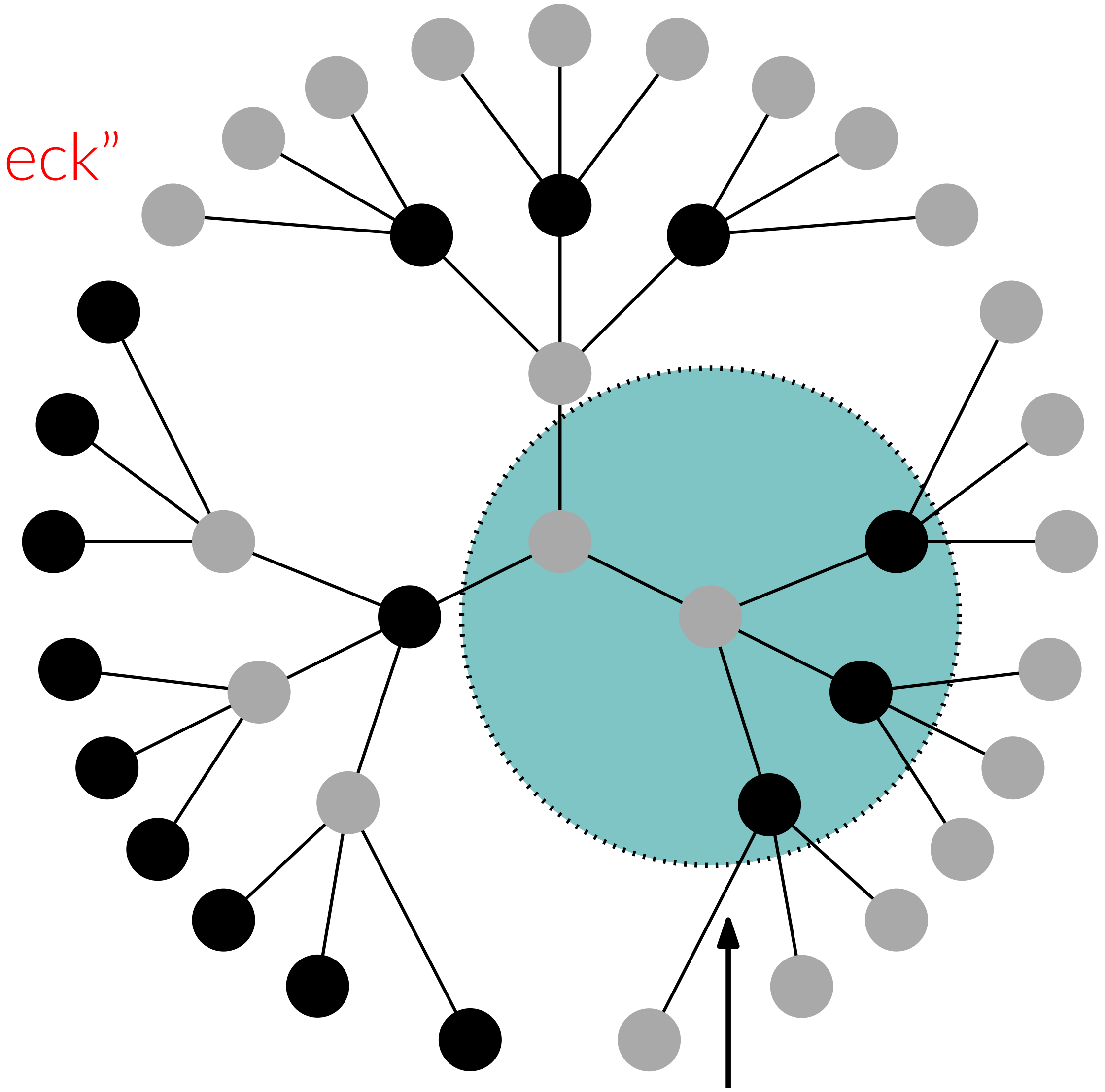
[Naor and Stockmeyer, STOC '93 & SICOMP '95]

- **Problems** whose solutions might be "hard to find" but are "easy to check"

  - "analogue" of NP in the distributed setting
  - coloring, maximal independent set, maximal matching, etc.

- **"Easy to check"**

  - radius $r = \Theta(1)$
  - each node can check its solution within its radius-r neighborhood
  - a globally valid iff each node is locally happy



MIS: each node checks if it is in the IS or if it has a neighbor in the IS

valid LCL

Locally checkable labeling (LCL) problems

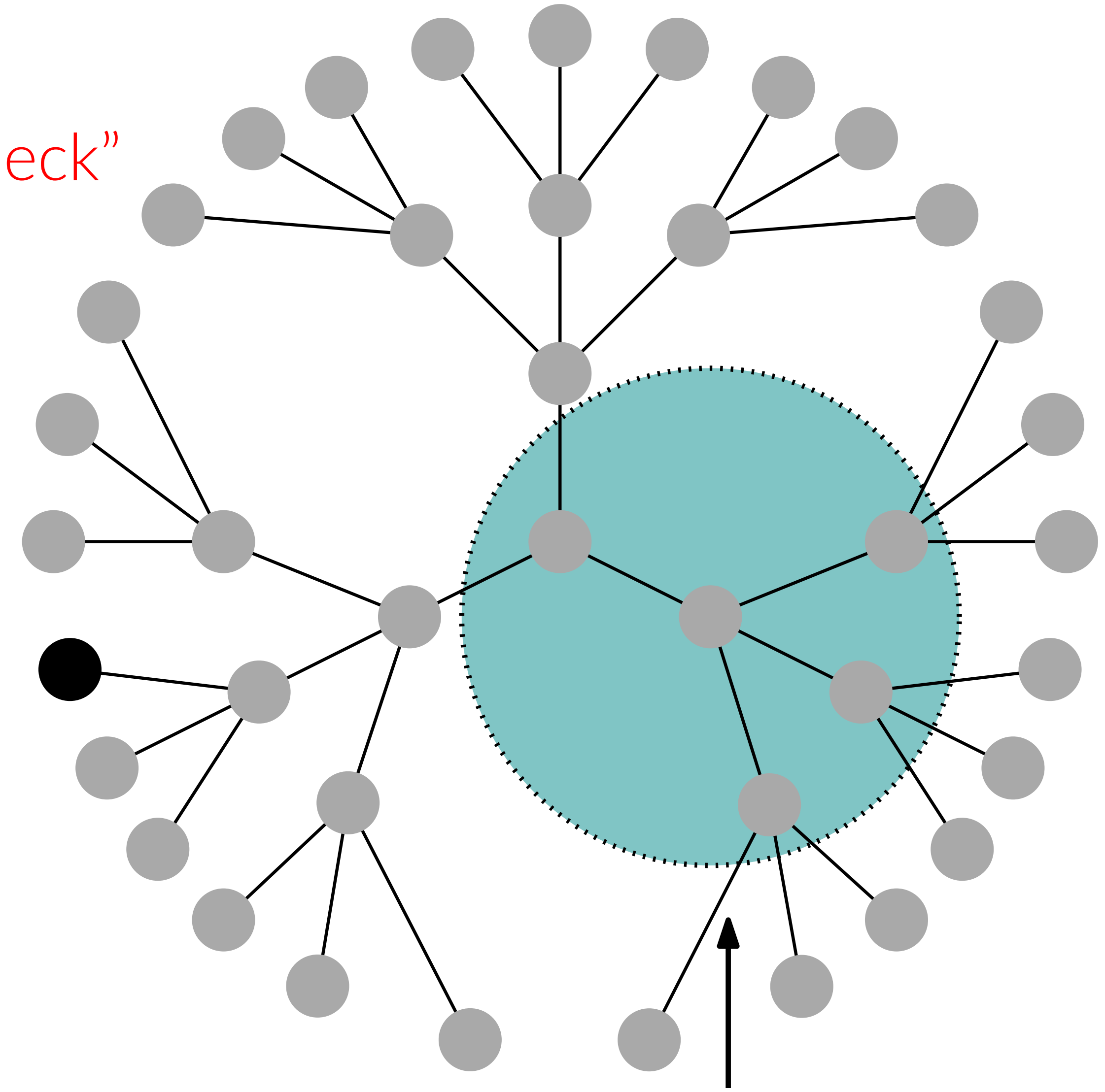
[Naor and Stockmeyer, STOC '93 & SICOMP '95]

- **Problems** whose solutions might be "hard to find" but are "easy to check"

  - "analogue" of NP in the distributed setting
  - coloring, maximal independent set, maximal matching, etc.

- **"Easy to check"**

  - radius $r = \Theta(1)$
  - each node can check its solution within its radius-r neighborhood
  - a globally valid iff each node is locally happy



Leader election: the checking radius should be $r = \mathrm{diam}(G)$

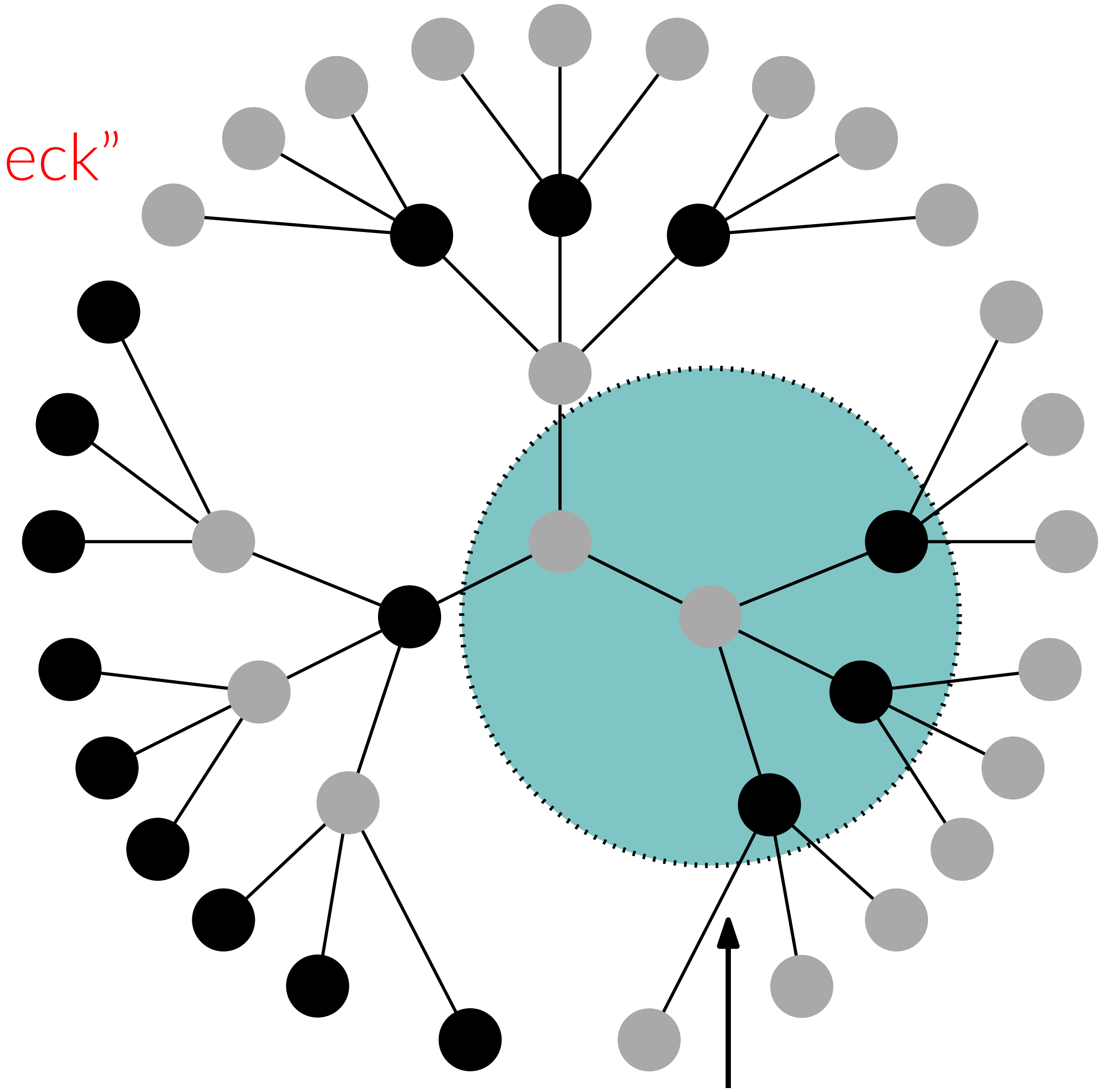
not an LCL

Locally checkable labeling (LCL) problems

[Naor and Stockmeyer, STOC '93 & SICOMP '95]

- **Problems** whose solutions might be "hard to find" but are "easy to check"
  - "analogue" of NP in the distributed setting
  - coloring, maximal independent set, maximal matching, etc.

- **"Easy to check"**
  - radius $r = \Theta(1)$
  - each node can check its solution within its radius-r neighborhood
  - a globally valid iff each node is locally happy



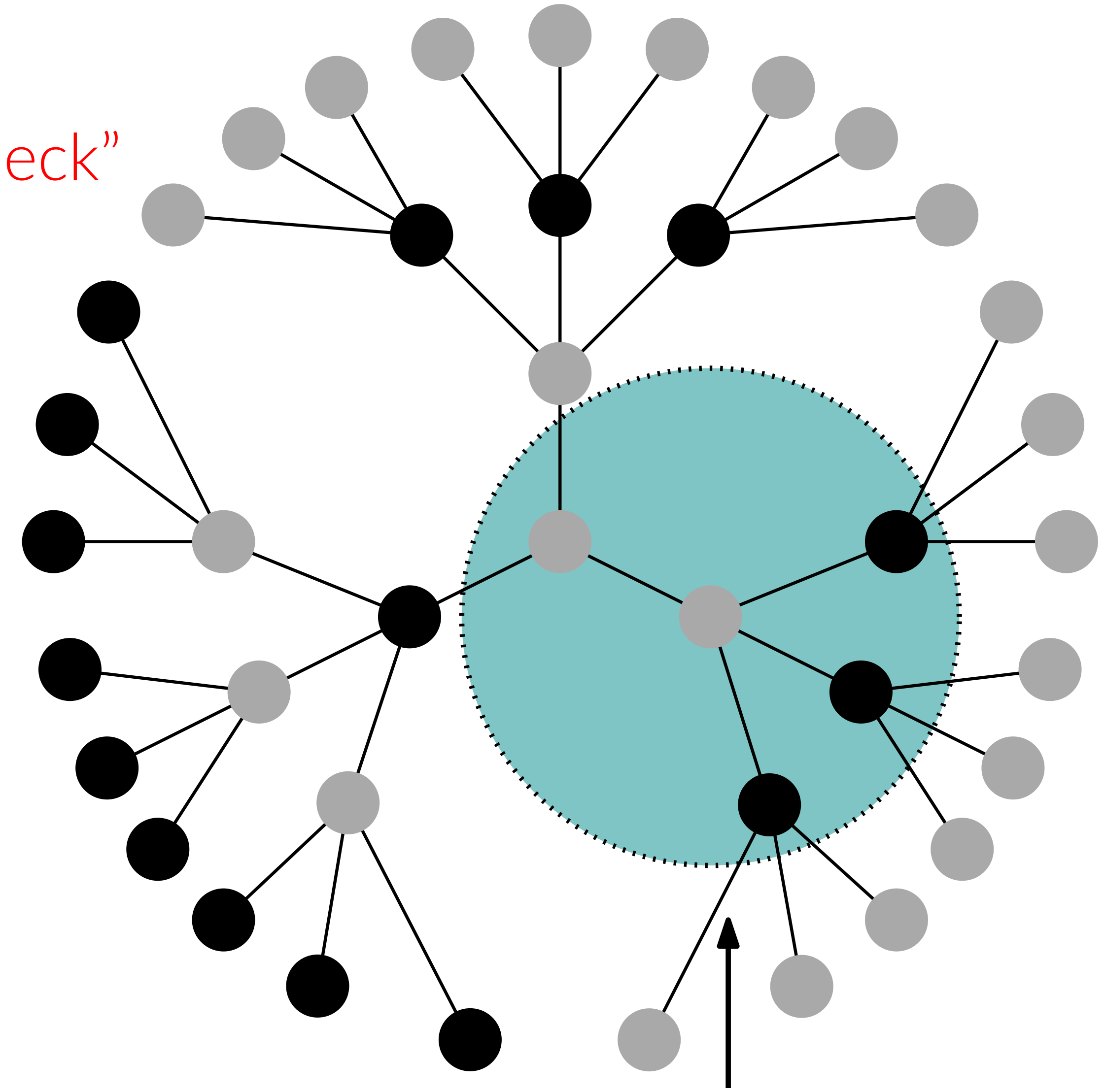
MIS: each node checks if it is in the IS or if it has a neighbor in the IS

Locally checkable labeling (LCL) problems

[Naor and Stockmeyer, STOC '93 & SICOMP '95]

- **Problems** whose solutions might be "hard to find" but are "easy to check"
  - "analogue" of NP in the distributed setting
  - coloring, maximal independent set, maximal matching, etc.

- **"Easy to check"**
  - radius $r = \Theta(1)$
  - each node can check its solution within its radius-r neighborhood
  - a globally valid iff each node is locally happy
  - max-degree Δ is bounded, i.e. $\Delta = O(1)$



MIS: each node checks if it is in the IS or if it has a neighbor in the IS

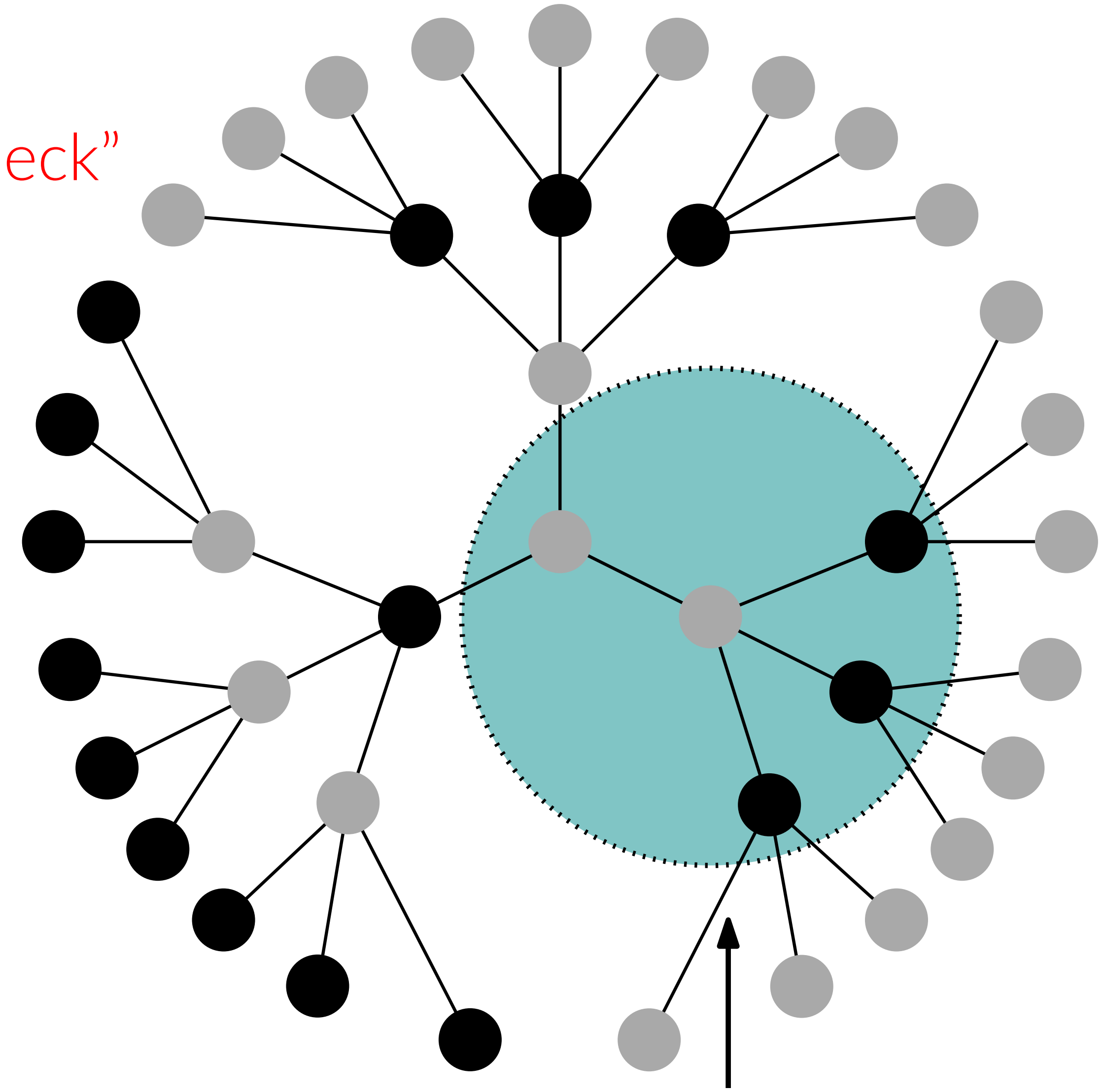
Locally checkable labeling (LCL) problems

[Naor and Stockmeyer, STOC '93 & SICOMP '95]

- **Problems** whose solutions might be "hard to find" but are "easy to check"
  - "analogue" of NP in the distributed setting
  - coloring, maximal independent set, maximal matching, etc.

- **"Easy to check"**
  - radius $r = \Theta(1)$
  - each node can check its solution within its radius-r neighborhood
  - a globally valid iff each node is locally happy
  - max-degree Δ is bounded, i.e. $\Delta = O(1)$

- A **lot of literature** studying LCLs:
  - classification of LCLs based on complexity (locality)
  - e.g.: complexity $T(n)$ in randomized-LOCAL $\implies O(T(2^{n^2}))$ in deterministic-LOCAL [Chang et al., SICOMP '19]



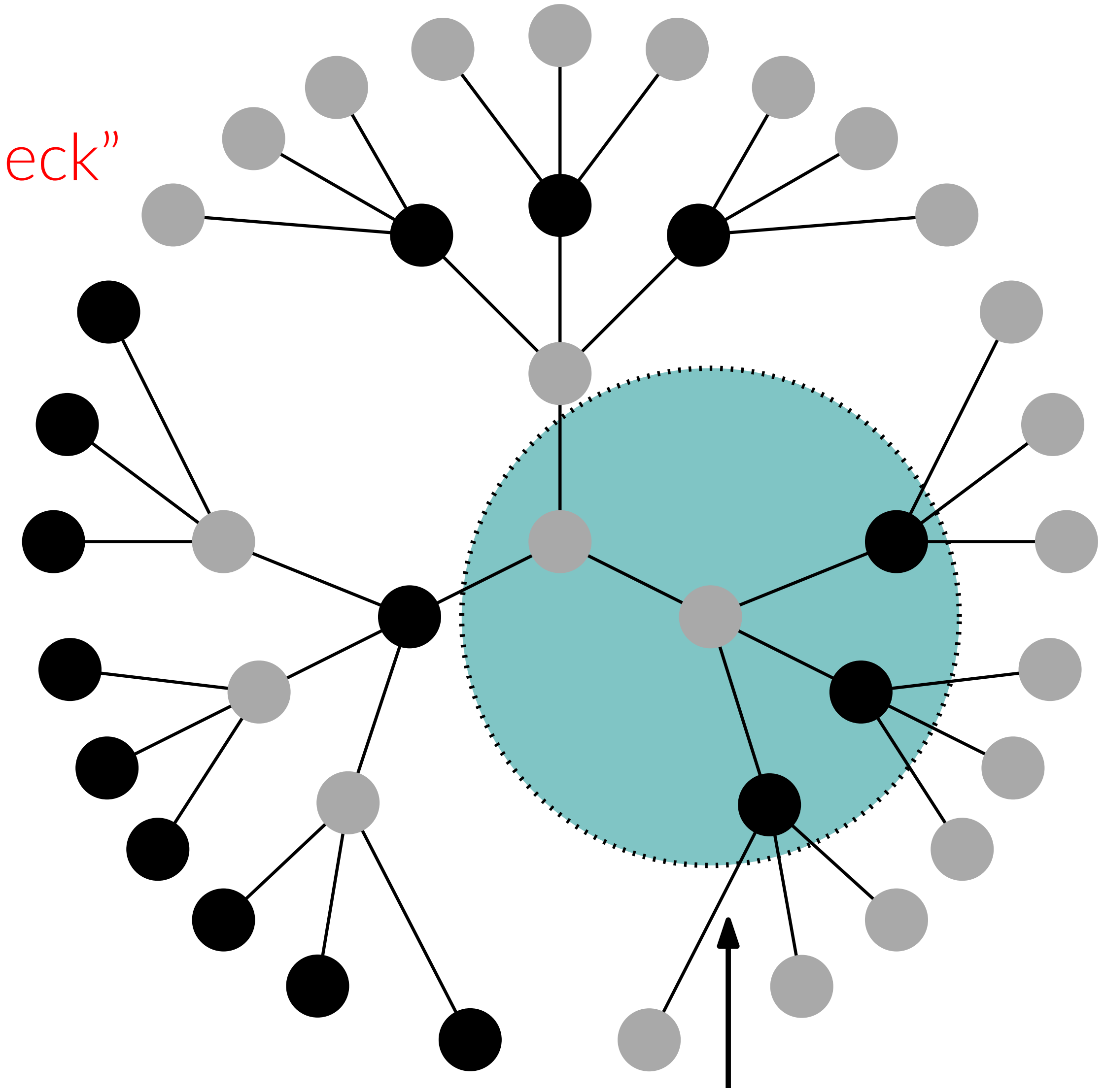
MIS: each node checks if it is in the IS or if it has a neighbor in the IS

Locally checkable labeling (LCL) problems

[Naor and Stockmeyer, STOC '93 & SICOMP '95]

- **Problems** whose solutions might be "hard to find" but are "easy to check"
  - "analogue" of NP in the distributed setting
  - coloring, maximal independent set, maximal matching, etc.

- **"Easy to check"**
  - radius $r = \Theta(1)$
  - each node can check its solution within its radius-r neighborhood
  - a globally valid iff each node is locally happy
  - max-degree Δ is bounded, i.e. $\Delta = O(1)$



MIS: each node checks if it is in the IS or if it has a neighbor in the IS

- A **lot of literature** studying LCLs:
  - classification of LCLs based on complexity (locality)
  - e.g.: complexity $T(n)$ in randomized-LOCAL $\implies O(T(2^{n^2}))$ in deterministic-LOCAL [Chang et al., SICOMP '19]
  - [BFHKLRSU STOC '16; BHKLOPRSU PODC'17; GKM STOC '17; GHK FOCS '18; CP SICOMP '19; BHKLOS STOC '18; BBCORS PODC '19; BBOS PODC '20; BBHORS JACM '21; BBCOSS DISC '22; AELMSS ICALP '23; etc.]

Complexity landscape of LCL problems

- **Paths and cycles**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n)$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n)$

Complexity landscape of LCL problems

- **Paths and cycles**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n)$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n)$

- **Balanced d-dimensional toroidal grids**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n^{1/d})$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n^{1/d})$

Complexity landscape of LCL problems

- **Paths and cycles**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n)$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n)$

- **Balanced d-dimensional toroidal grids**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n^{1/d})$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n^{1/d})$

→ randomness does not help

Complexity landscape of LCL problems

- **Paths and cycles**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n)$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n)$

- **Balanced d-dimensional toroidal grids**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n^{1/d})$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n^{1/d})$

randomness does not help

- **Bounded-degree trees**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$				$\Theta(n)$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$				$\Theta(n)$

Complexity landscape of LCL problems

- **Paths and cycles**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n)$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n)$

- **Balanced d-dimensional toroidal grids**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n^{1/d})$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n^{1/d})$

randomness does not help

- **Bounded-degree trees**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(\log n)$			$\Theta(n)$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(\log \log n)$	$\Theta(\log n)$		$\Theta(n)$

Complexity landscape of LCL problems

- **Paths and cycles**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n)$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n)$

- **Balanced d-dimensional toroidal grids**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n^{1/d})$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(n^{1/d})$

randomness does not help

- **Bounded-degree trees**

<table>
<tr><td>det-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log^\star n)$</td><td colspan="2">$\Theta(\log n)$</td><td>$\Theta(n^{1/k})$</td><td>$\Theta(n)$</td></tr>
<tr><td>rand-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log^\star n)$</td><td>$\Theta(\log\log n)$</td><td>$\Theta(\log n)$</td><td>$\Theta(n^{1/k})$</td><td>$\Theta(n)$</td></tr>
</table>

Complexity landscape of LCL problems

- **Bounded-degree trees**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(\log n)$		$\Theta(n^{1/k})$	$\Theta(n)$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(\log\log n)$	$\Theta(\log n)$	$\Theta(n^{1/k})$	$\Theta(n)$

- **General graphs**

det-LOCAL							
rand-LOCAL							

Complexity landscape of LCL problems

- **Bounded-degree trees**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(\log n)$		$\Theta(n^{1/k})$	$\Theta(n)$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(\log\log n)$	$\Theta(\log n)$	$\Theta(n^{1/k})$	$\Theta(n)$

- **General graphs**

det-LOCAL	$O(1)$						$\Theta(n)$
rand-LOCAL	$O(1)$						$\Theta(n)$

- **Bounded-degree trees**

det-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(\log n)$		$\Theta(n^{1/k})$	$\Theta(n)$
rand-LOCAL	$O(1)$	$\Theta(\log^\star n)$	$\Theta(\log\log n)$	$\Theta(\log n)$	$\Theta(n^{1/k})$	$\Theta(n)$

- **General graphs**

det-LOCAL	$O(1)$	$\Theta(\log\log^\star n)$					$\Theta(n)$
rand-LOCAL	$O(1)$	$\Theta(\log\log^\star n)$					$\Theta(n)$

Complexity landscape of LCL problems

- **Bounded-degree trees**

<table>
<tr><td>det-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log^\star n)$</td><td colspan="2">$\Theta(\log n)$</td><td>$\Theta(n^{1/k})$</td><td>$\Theta(n)$</td></tr>
<tr><td>rand-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log^\star n)$</td><td>$\Theta(\log\log n)$</td><td>$\Theta(\log n)$</td><td>$\Theta(n^{1/k})$</td><td>$\Theta(n)$</td></tr>
</table>

- **General graphs**

<table>
<tr><td>det-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log\log^\star n)$</td><td>$\Theta(\log^\star n)$</td><td colspan="2"></td><td></td><td>$\Theta(n)$</td></tr>
<tr><td>rand-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log\log^\star n)$</td><td>$\Theta(\log^\star n)$</td><td></td><td></td><td></td><td>$\Theta(n)$</td></tr>
</table>

Complexity landscape of LCL problems

- **Bounded-degree trees**

<table>
<tr><td>det-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log^\star n)$</td><td colspan="2">$\Theta(\log n)$</td><td>$\Theta(n^{1/k})$</td><td>$\Theta(n)$</td></tr>
<tr><td>rand-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log^\star n)$</td><td>$\Theta(\log\log n)$</td><td>$\Theta(\log n)$</td><td>$\Theta(n^{1/k})$</td><td>$\Theta(n)$</td></tr>
</table>

- **General graphs**

<table>
<tr><td>det-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log\log^\star n)$</td><td>$\Theta(\log^\star n)$</td><td colspan="2">$\Theta(\log n)$</td><td></td><td>$\Theta(n)$</td></tr>
<tr><td>rand-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log\log^\star n)$</td><td>$\Theta(\log^\star n)$</td><td>$\Theta(\log\log n)$</td><td>$\Theta(\log n)$</td><td></td><td>$\Theta(n)$</td></tr>
</table>

Complexity landscape of LCL problems

- **Bounded-degree trees**

<table>
<tr><td>det-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log^\star n)$</td><td colspan="2">$\Theta(\log n)$</td><td>$\Theta(n^{1/k})$</td><td>$\Theta(n)$</td></tr>
<tr><td>rand-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log^\star n)$</td><td>$\Theta(\log\log n)$</td><td>$\Theta(\log n)$</td><td>$\Theta(n^{1/k})$</td><td>$\Theta(n)$</td></tr>
</table>

- **General graphs**

<table>
<tr><td>det-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log\log^\star n)$</td><td>$\Theta(\log^\star n)$</td><td colspan="2">$\Theta(\log n)$</td><td>??</td><td>$\Theta(n)$</td></tr>
<tr><td>rand-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log\log^\star n)$</td><td>$\Theta(\log^\star n)$</td><td>$\Theta(\log\log n)$</td><td>$\Theta(\log n)$</td><td>??</td><td>$\Theta(n)$</td></tr>
</table>

Complexity landscape of LCL problems

- **Bounded-degree trees**

<table>
<tr><td>det-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log^\star n)$</td><td colspan="2">$\Theta(\log n)$</td><td>$\Theta(n^{1/k})$</td><td>$\Theta(n)$</td></tr>
<tr><td>rand-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log^\star n)$</td><td>$\Theta(\log\log n)$</td><td>$\Theta(\log n)$</td><td>$\Theta(n^{1/k})$</td><td>$\Theta(n)$</td></tr>
</table>

- **General graphs**

<table>
<tr><td>det-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log\log^\star n)$</td><td>$\Theta(\log^\star n)$</td><td colspan="2">$\Theta(\log n)$</td><td>??</td><td>$\Theta(n)$</td></tr>
<tr><td>rand-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log\log^\star n)$</td><td>$\Theta(\log^\star n)$</td><td>$\Theta(\log\log n)$</td><td>$\Theta(\log n)$</td><td>??</td><td>$\Theta(n)$</td></tr>
</table>

- **General graphs**: above $\Theta(\log n)$, every function $f(n)$ makes a complexity class

Complexity landscape of LCL problems

- **Bounded-degree trees**

<table>
<tr><td>det-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log^\star n)$</td><td colspan="2">$\Theta(\log n)$</td><td>$\Theta(n^{1/k})$</td><td>$\Theta(n)$</td></tr>
<tr><td>rand-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log^\star n)$</td><td>$\Theta(\log\log n)$</td><td>$\Theta(\log n)$</td><td>$\Theta(n^{1/k})$</td><td>$\Theta(n)$</td></tr>
</table>

- **General graphs**

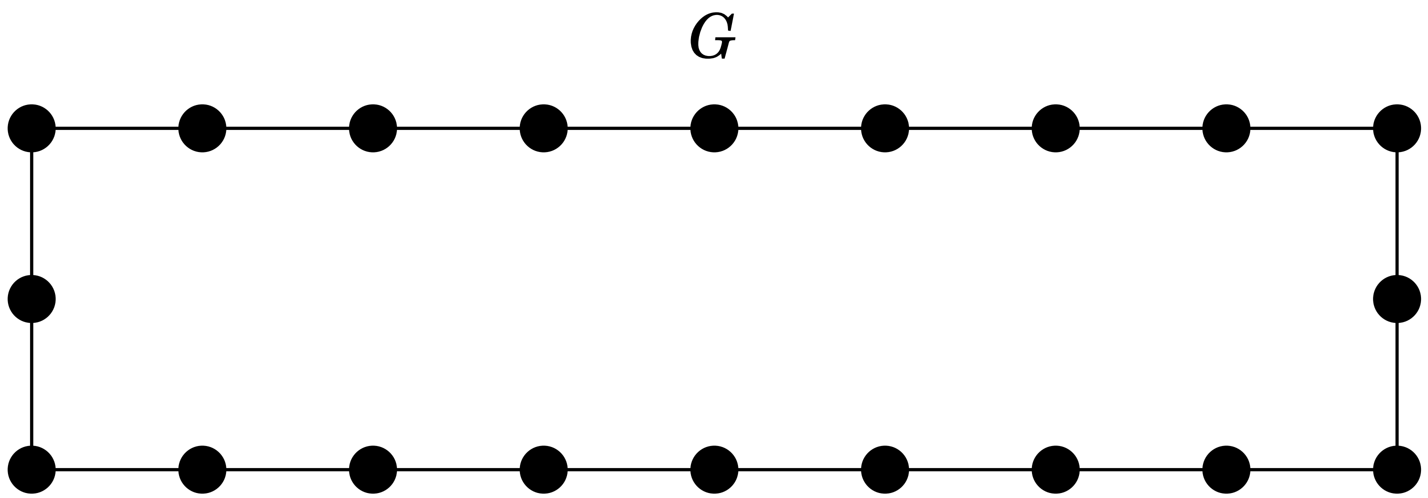
<table>
<tr><td>det-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log\log^\star n)$</td><td>$\Theta(\log^\star n)$</td><td colspan="2">$\Theta(\log n)$</td><td>??</td><td>$\Theta(n)$</td></tr>
<tr><td>rand-LOCAL</td><td>$O(1)$</td><td>$\Theta(\log\log^\star n)$</td><td>$\Theta(\log^\star n)$</td><td>$\Theta(\log\log n)$</td><td>$\Theta(\log n)$</td><td>??</td><td>$\Theta(n)$</td></tr>
</table>

- **General graphs**: above $\Theta(\log n)$, every function $f(n)$ makes a complexity class
    - role of quantum??

Table of content

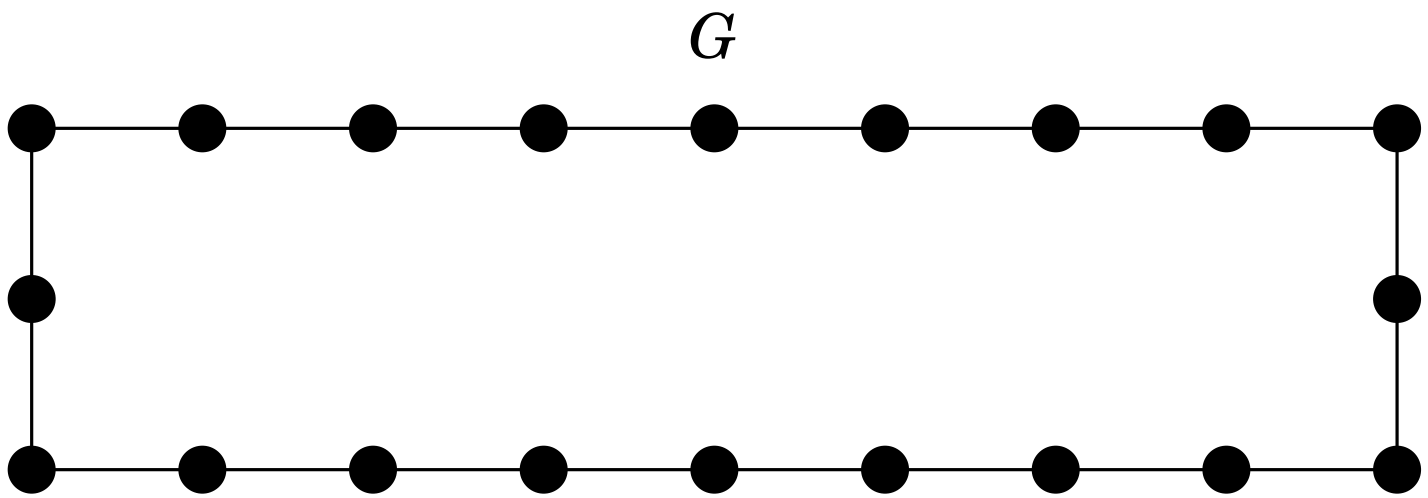
Indistinguishability argument

- **Problem**: 2-coloring paths & even cycles
- *Promise (for now)*: the graph constructed is either a path or an even cycle



Indistinguishability argument

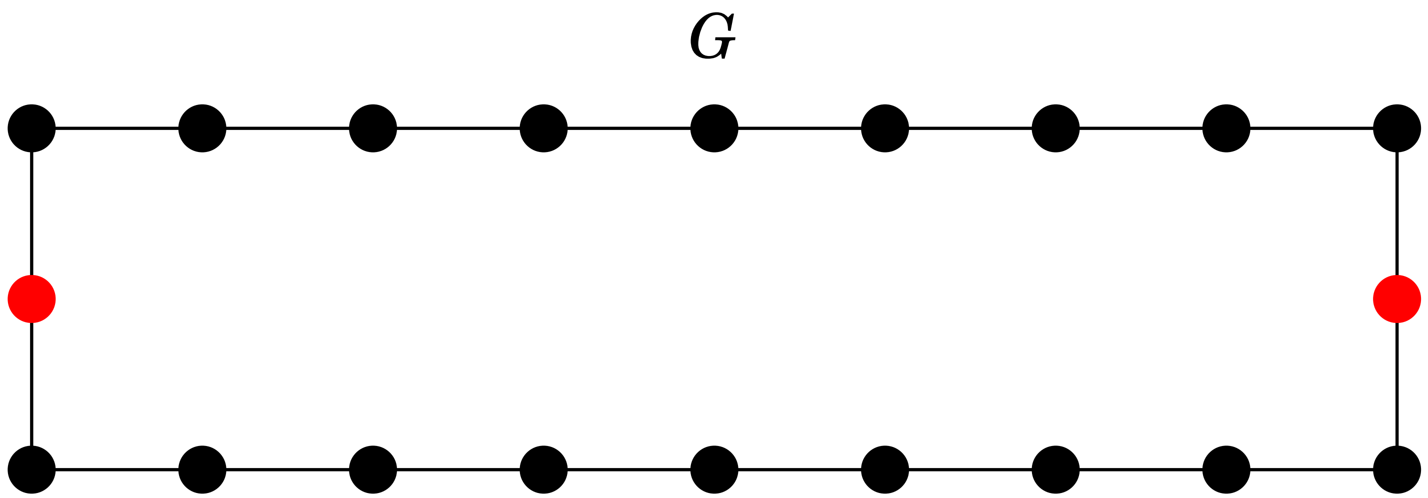
- **Problem**: 2-coloring paths & even cycles
- *Promise (for now)*: the graph constructed is either a path or an even cycle



- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5$

Indistinguishability argument

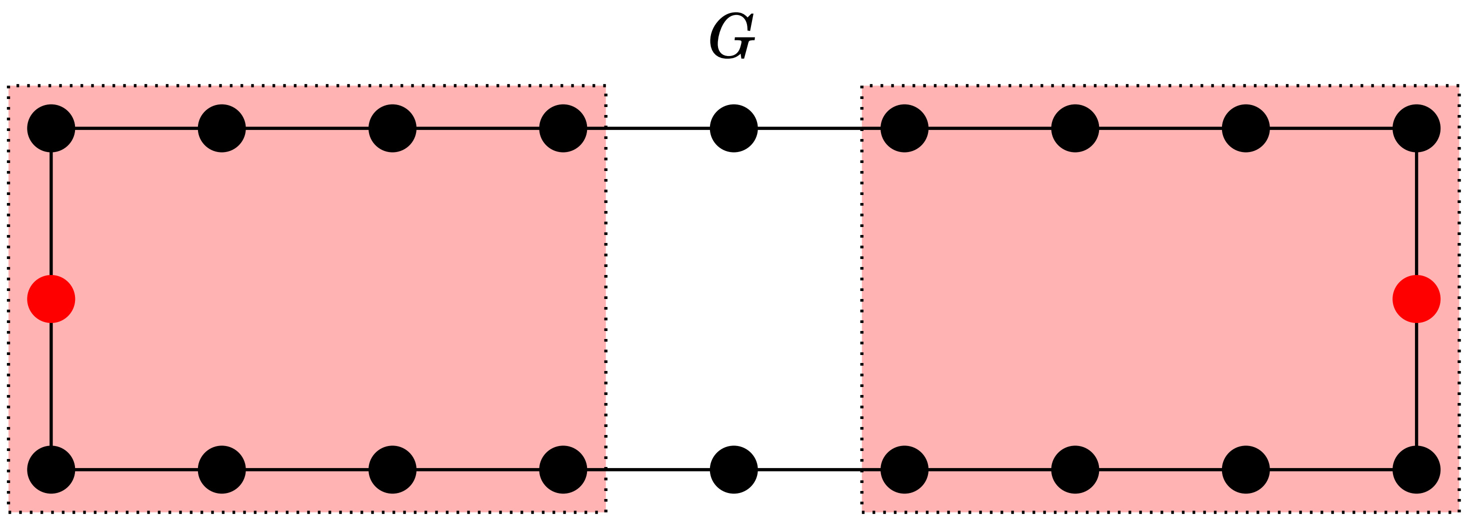
- **Problem**: 2-coloring paths & even cycles
- *Promise (for now)*: the graph constructed is either a path or an even cycle



- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5$

Indistinguishability argument

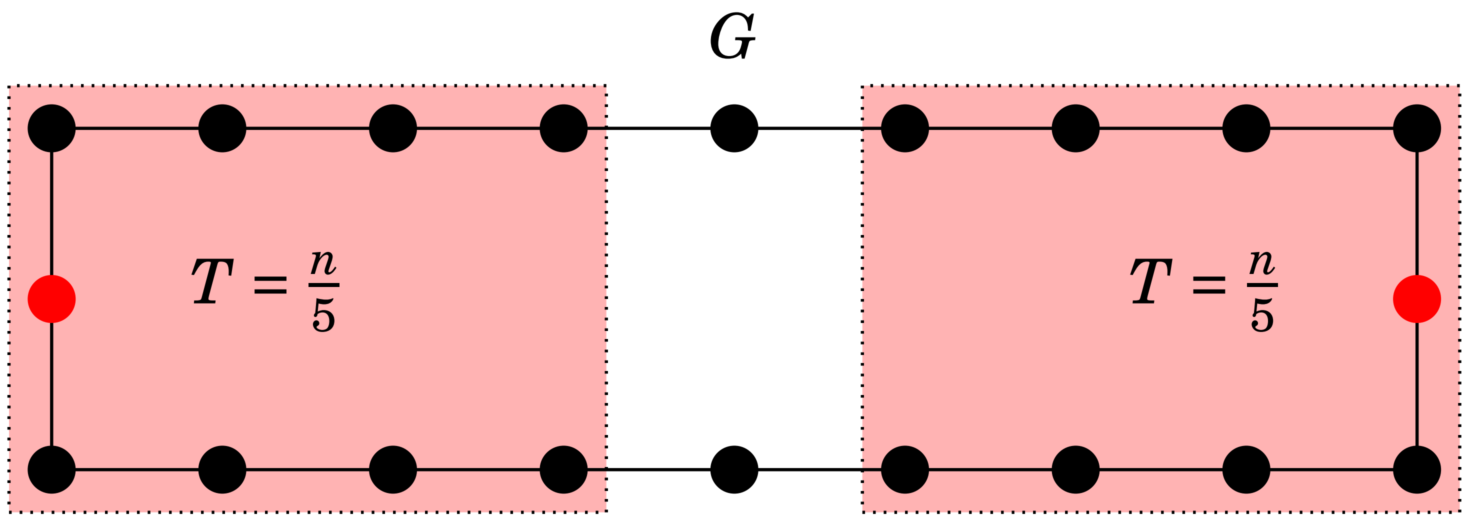
- **Problem**: 2-coloring paths & even cycles
- *Promise (for now)*: the graph constructed is either a path or an even cycle



  - Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5$

Indistinguishability argument

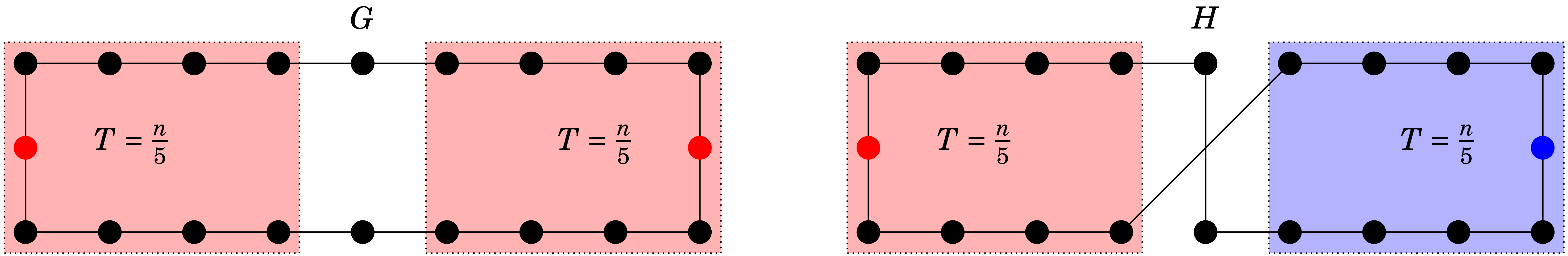
- **Problem**: 2-coloring paths & even cycles
- *Promise (for now)*: the graph constructed is either a path or an even cycle



- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5$

Indistinguishability argument

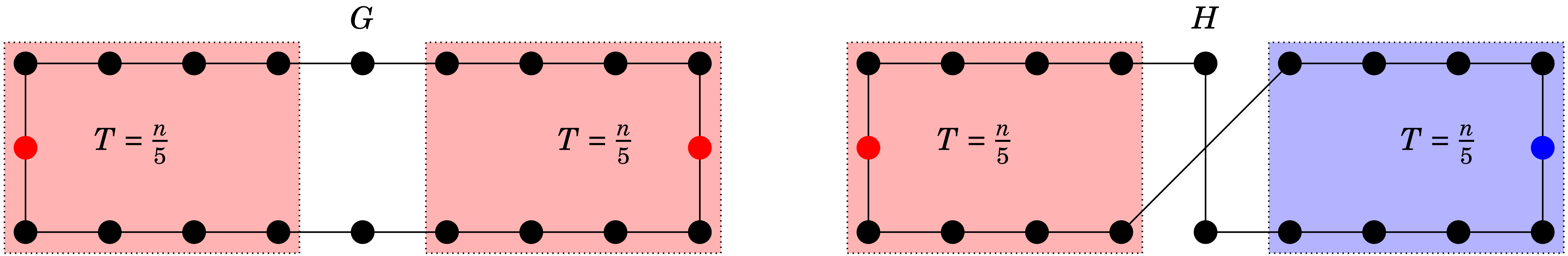
- **Problem**: 2-coloring paths & even cycles
- *Promise (for now)*: the graph constructed is either a path or an even cycle



- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5$

Indistinguishability argument

- **Problem**: 2-coloring paths & even cycles
- *Promise (for now)*: the graph constructed is either a path or an even cycle

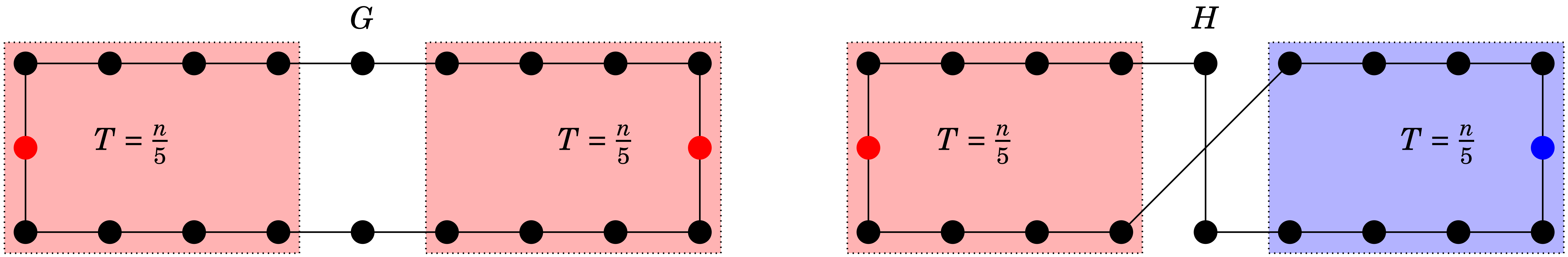


- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5$

- Impossible to distinguish between G and H

Indistinguishability argument

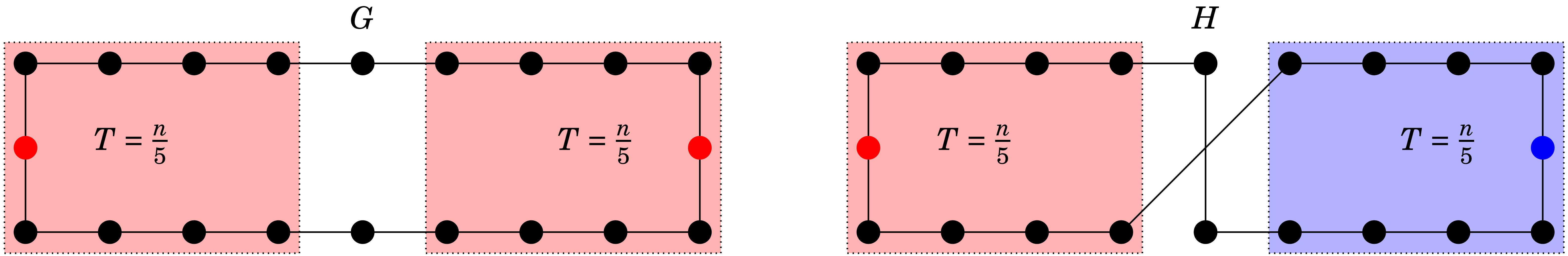
- **Problem**: 2-coloring paths & even cycles
- *Promise (for now)*: the graph constructed is either a path or an even cycle



- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5$
- Impossible to distinguish between G and H
- Deterministically: impossible

Indistinguishability argument

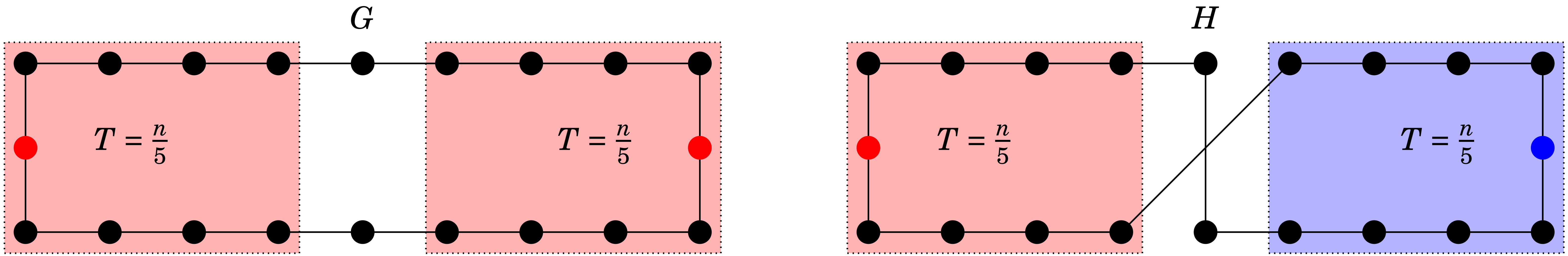
- **Problem**: 2-coloring paths & even cycles
- *Promise (for now)*: the graph constructed is either a path or an even cycle



- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5$
- Impossible to distinguish between G and H
- Deterministically: impossible
- With randomness: failure with prob. $\geq 1/2$

Indistinguishability argument

- **Problem**: 2-coloring paths & even cycles
- *Promise (for now)*: the graph constructed is either a path or an even cycle

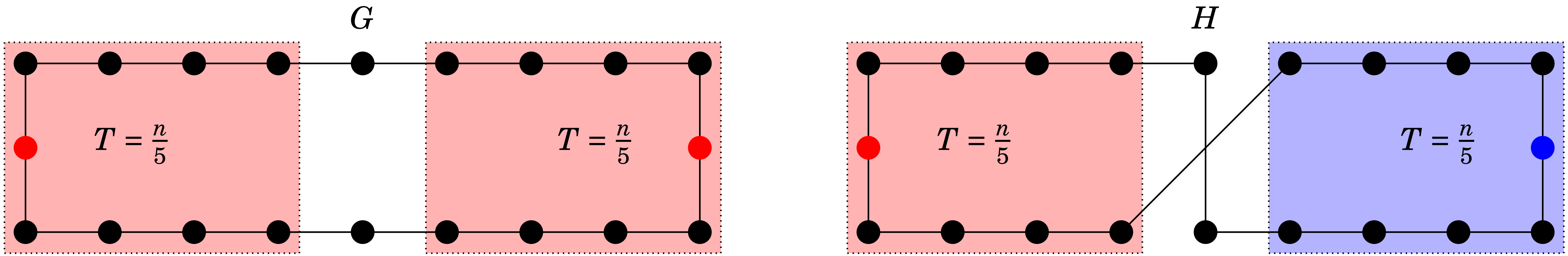


- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5$
- Impossible to distinguish between G and H
- Deterministically: impossible
- With randomness: failure with prob. $\geq 1/2$

- **Global problem**: complexity $\Theta(n)$

Indistinguishability argument

- **Problem**: 2-coloring paths & even cycles
- *Promise (for now)*: the graph constructed is either a path or an even cycle



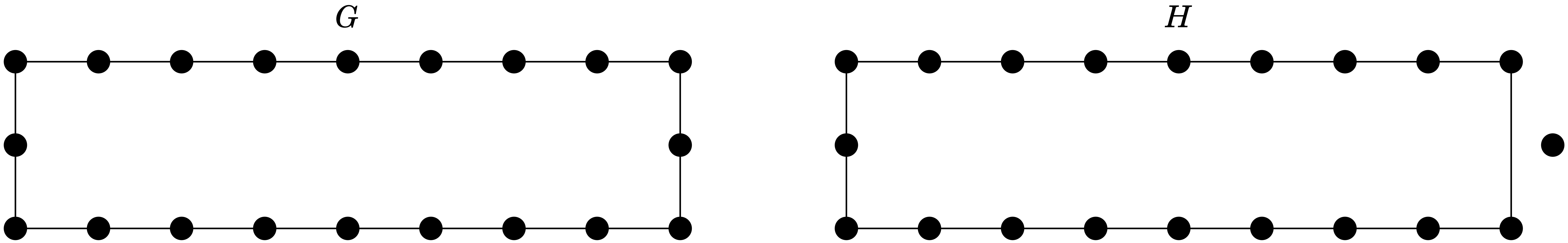
- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5$
- Impossible to distinguish between G and H
- Deterministically: impossible
- With randomness: failure with prob. $\geq 1/2$

indistinguishability argument

- **Global problem**: complexity $\Theta(n)$

Graph-existential indistinguishability (randomized)

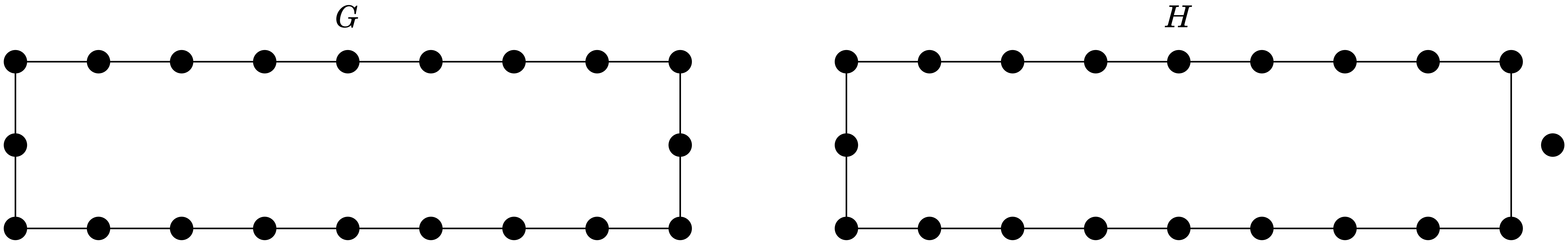
- **Problem**: 2-coloring paths & even cycles



- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$

Graph-existential indistinguishability (randomized)

- **Problem**: 2-coloring paths & even cycles

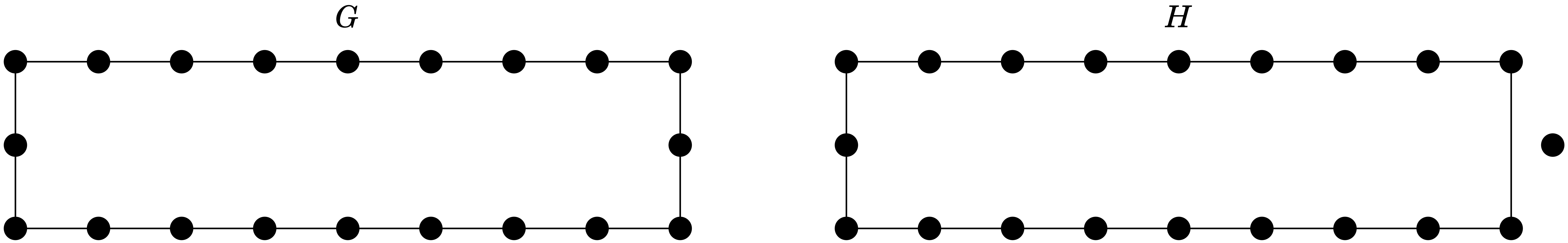


- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$

- Run $\mathcal{A}$ both in G and H

Graph-existential indistinguishability (randomized)

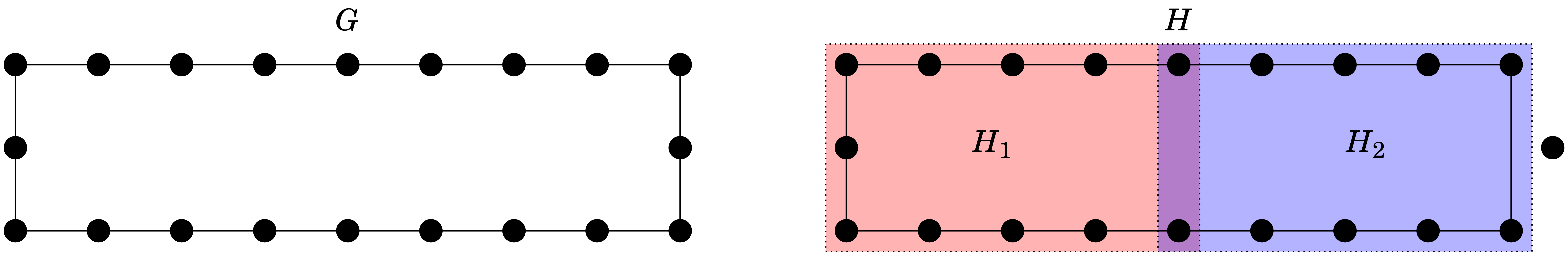
- **Problem**: 2-coloring paths & even cycles



- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$
- Run $\mathcal{A}$ both in G and H
- $\mathcal{A}$ fails in H with probability 1

Graph-existential indistinguishability (randomized)

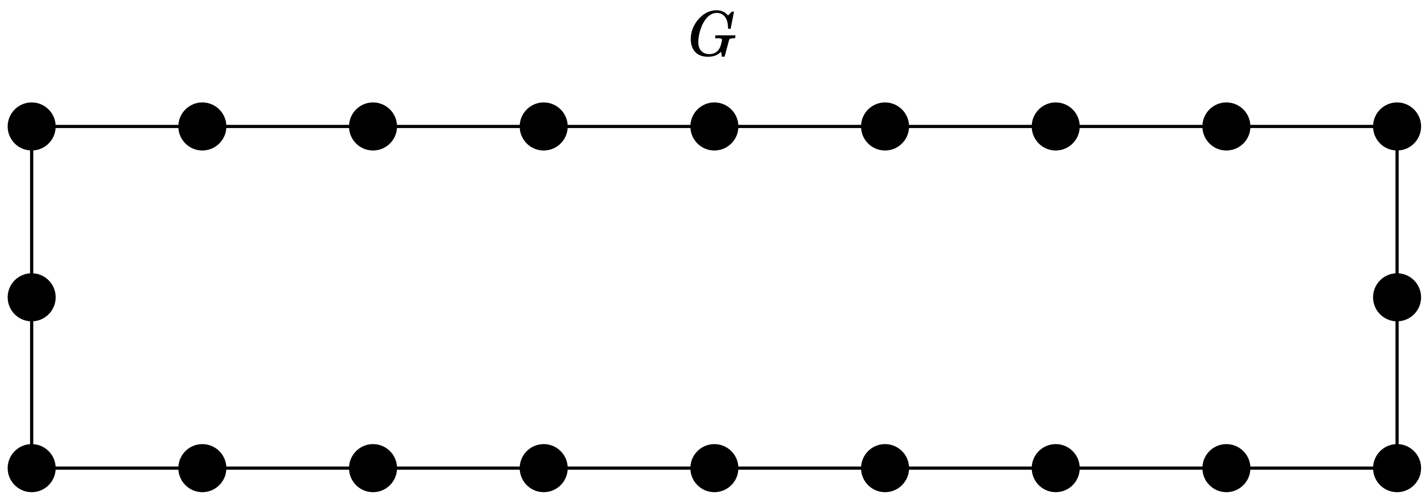
- **Problem**: 2-coloring paths & even cycles

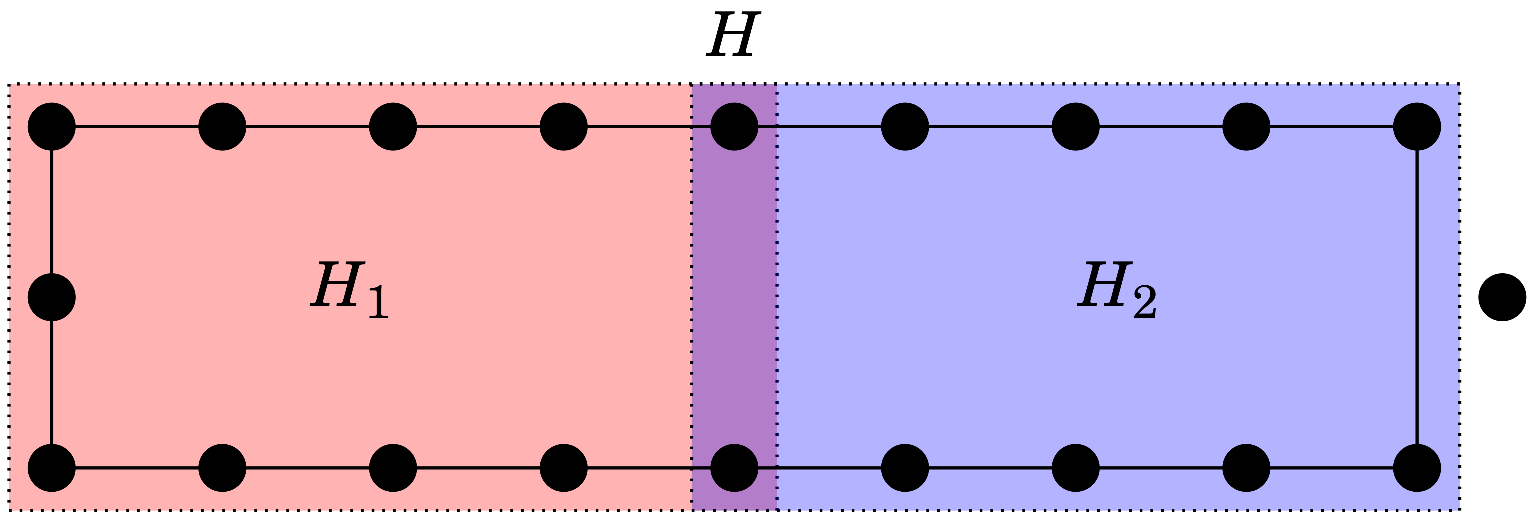


- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$
- Run $\mathcal{A}$ both in G and H
- $\mathcal{A}$ fails in H with probability 1

Graph-existential indistinguishability (randomized)

- **Problem**: 2-coloring paths & even cycles

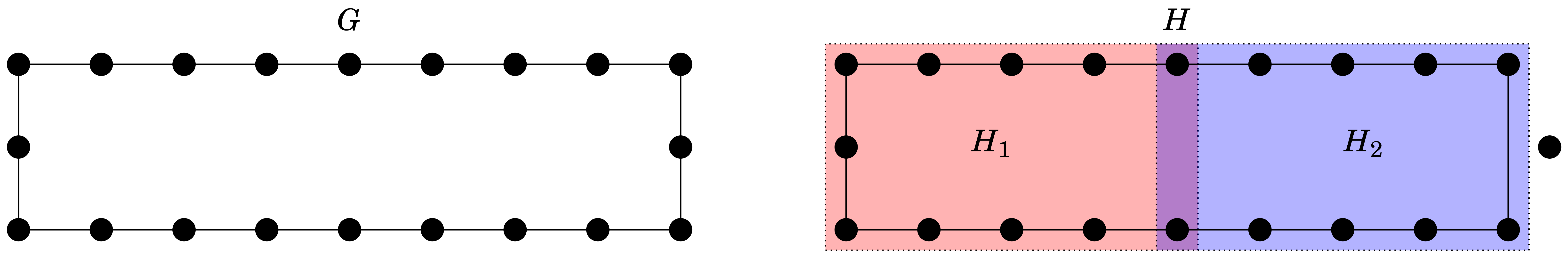




- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$
- Run $\mathcal{A}$ both in G and H
- $\mathcal{A}$ fails in H with probability 1
- $\mathcal{A}$ fails either in H_1 or in H_2 with probability $\geq 1/2$

Graph-existential indistinguishability (randomized)

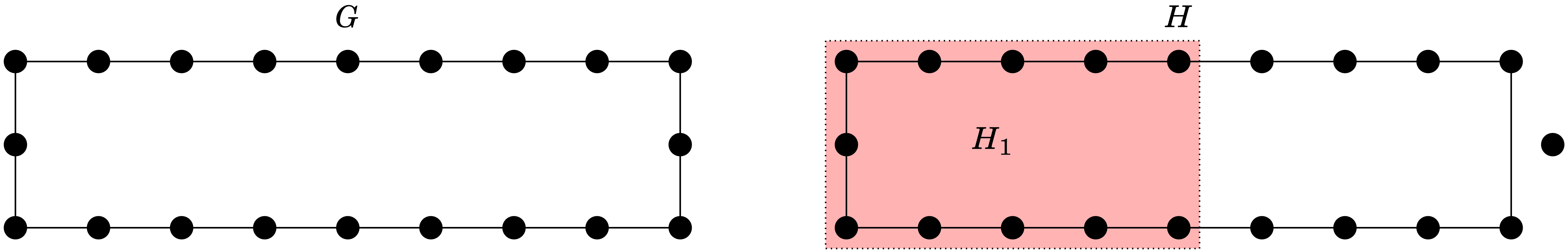
- **Problem**: 2-coloring paths & even cycles



- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$
- Run $\mathcal{A}$ both in G and H
- $\mathcal{A}$ fails in H with probability 1
- $\mathcal{A}$ fails either in H_1 or in H_2 with probability $\geq 1/2$. Wlog, in H_1

Graph-existential indistinguishability (randomized)

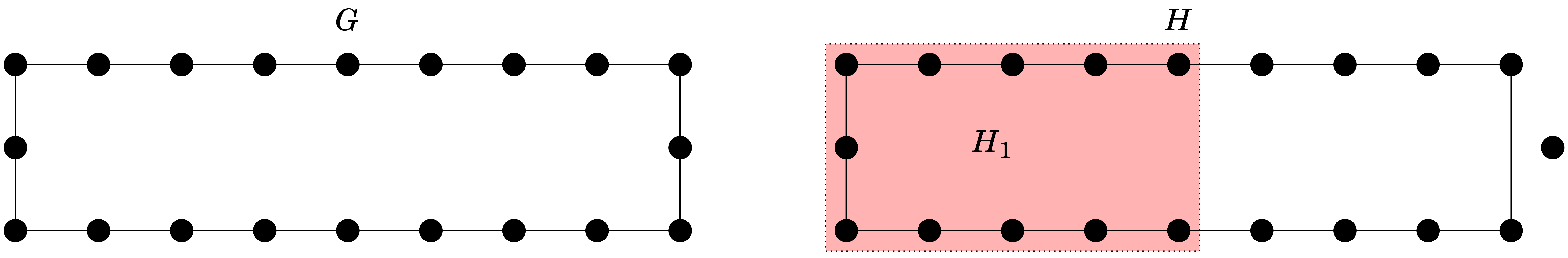
- **Problem**: 2-coloring paths & even cycles



- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$
- Run $\mathcal{A}$ both in G and H
- $\mathcal{A}$ fails in H with probability 1
- $\mathcal{A}$ fails either in H_1 or in H_2 with probability $\geq 1/2$. Wlog, in H_1

Graph-existential indistinguishability (randomized)

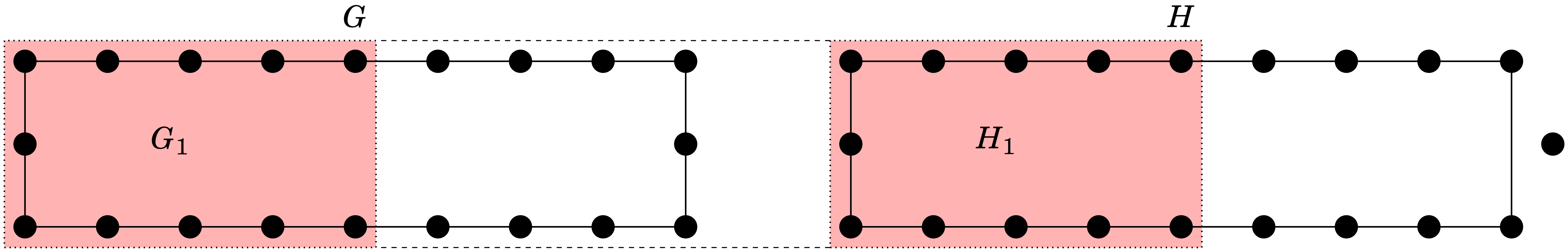
- **Problem**: 2-coloring paths & even cycles



  - Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$
  - Run $\mathcal{A}$ both in G and H
  - $\mathcal{A}$ fails in H with probability 1
  - $\mathcal{A}$ fails either in H_1 or in H_2 with probability $\geq 1/2$. Wlog, in H_1
  - Copy H_1 in G (*and* radius-T view)

Graph-existential indistinguishability (randomized)

- **Problem**: 2-coloring paths & even cycles



- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$

- Run $\mathcal{A}$ both in G and H

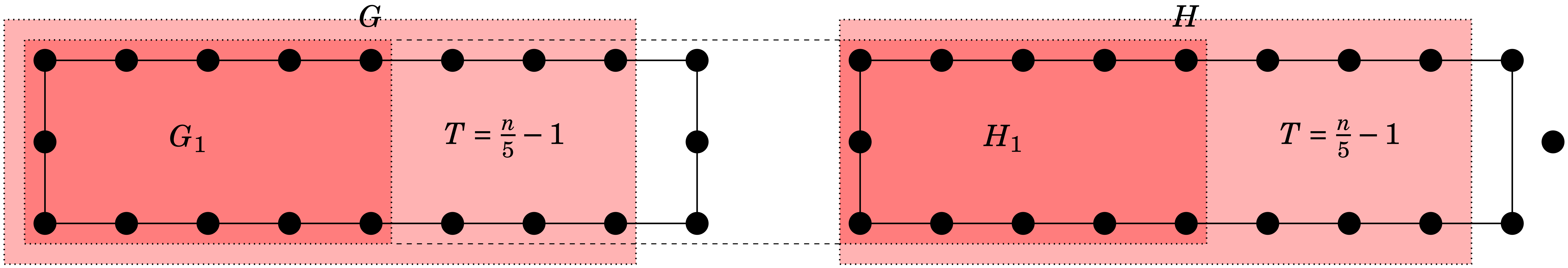
- $\mathcal{A}$ fails in H with probability 1

- $\mathcal{A}$ fails either in H_1 or in H_2 with probability $\geq 1/2$. Wlog, in H_1

- Copy H_1 in G (*and* radius-T view)

Graph-existential indistinguishability (randomized)

- **Problem**: 2-coloring paths & even cycles



- Suppose algorithm $\mathcal{A}$ with running time $T \le n/5 - 1$ succeeds in G with probability $\ge 1 - 1/n$

- Run $\mathcal{A}$ both in G and H

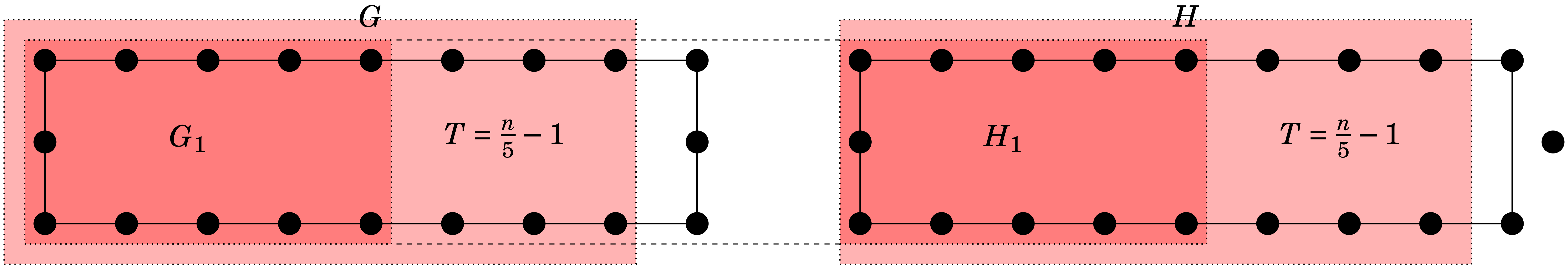
- $\mathcal{A}$ fails in H with probability 1

- $\mathcal{A}$ fails either in H_1 or in H_2 with probability $\ge 1/2$. Wlog, in H_1

- Copy H_1 in G (*and* radius-T view)

Graph-existential indistinguishability (randomized)

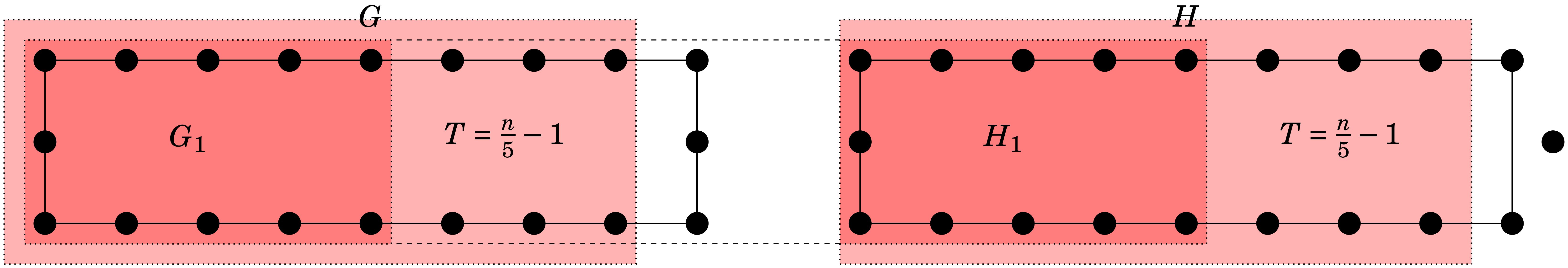
- **Problem**: 2-coloring paths & even cycles



- Suppose algorithm $\mathcal{A}$ with running time $T \le n/5 - 1$ succeeds in G with probability $\ge 1 - 1/n$
- Run $\mathcal{A}$ both in G and H
- $\mathcal{A}$ fails in H with probability 1
- $\mathcal{A}$ fails either in H_1 or in H_2 with probability $\ge 1/2$. Wlog, in H_1
- Copy H_1 in G (*and* radius-T view)
- Nodes in H_1/G_1 *cannot distinguish* between G and H

Graph-existential indistinguishability (randomized)

- **Problem**: 2-coloring paths & even cycles



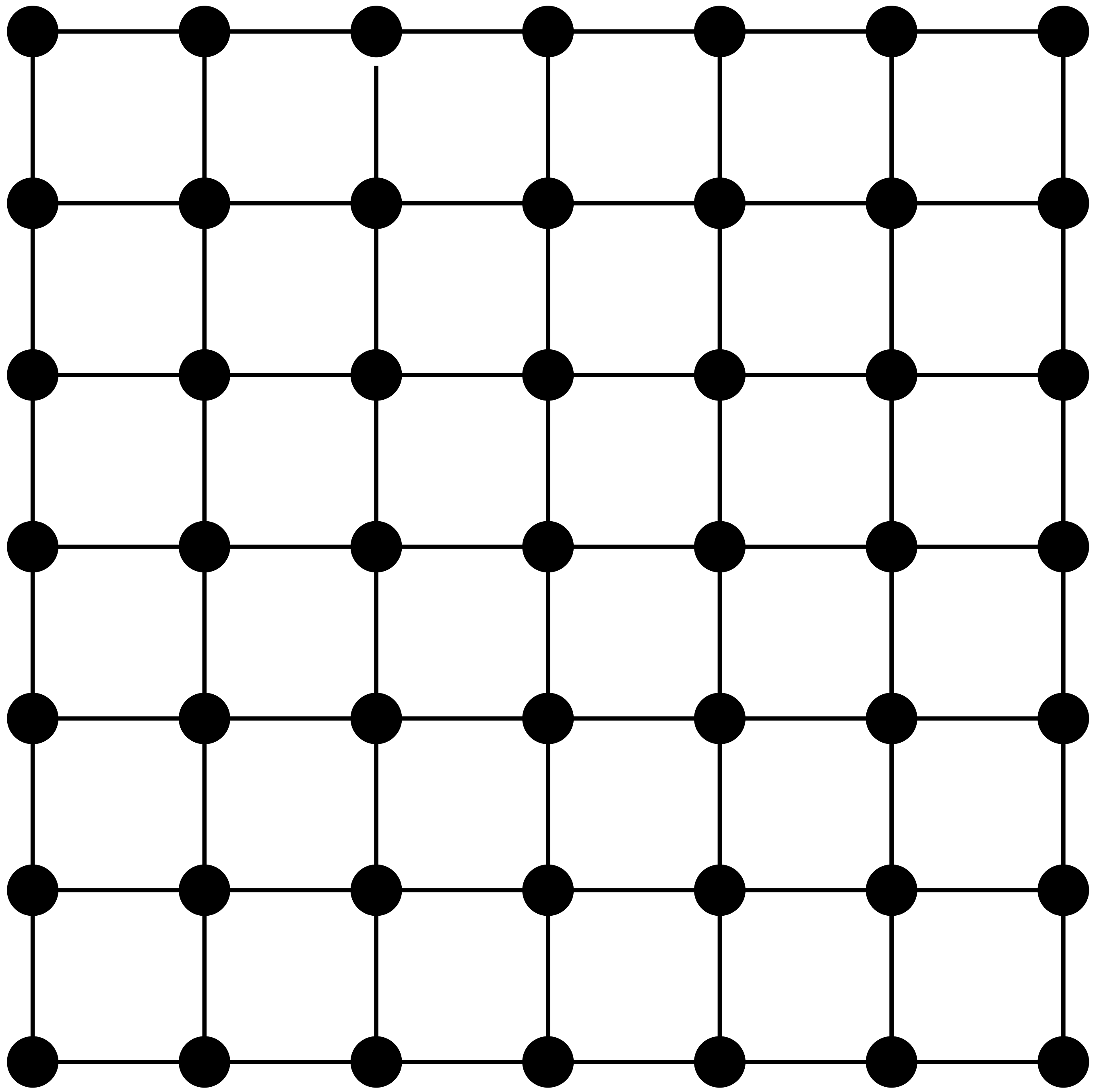
- Suppose algorithm $\mathcal{A}$ with running time $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$
- Run $\mathcal{A}$ both in G and H
- $\mathcal{A}$ fails in H with probability 1
- $\mathcal{A}$ fails either in H_1 or in H_2 with probability $\geq 1/2$. Wlog, in H_1
- Copy H_1 in G (*and* radius-T view)
- Nodes in H_1/G_1 *cannot distinguish* between G and H

$\mathcal{A}$ fails in G_1 with prob. $\geq 1/2$

Table of content

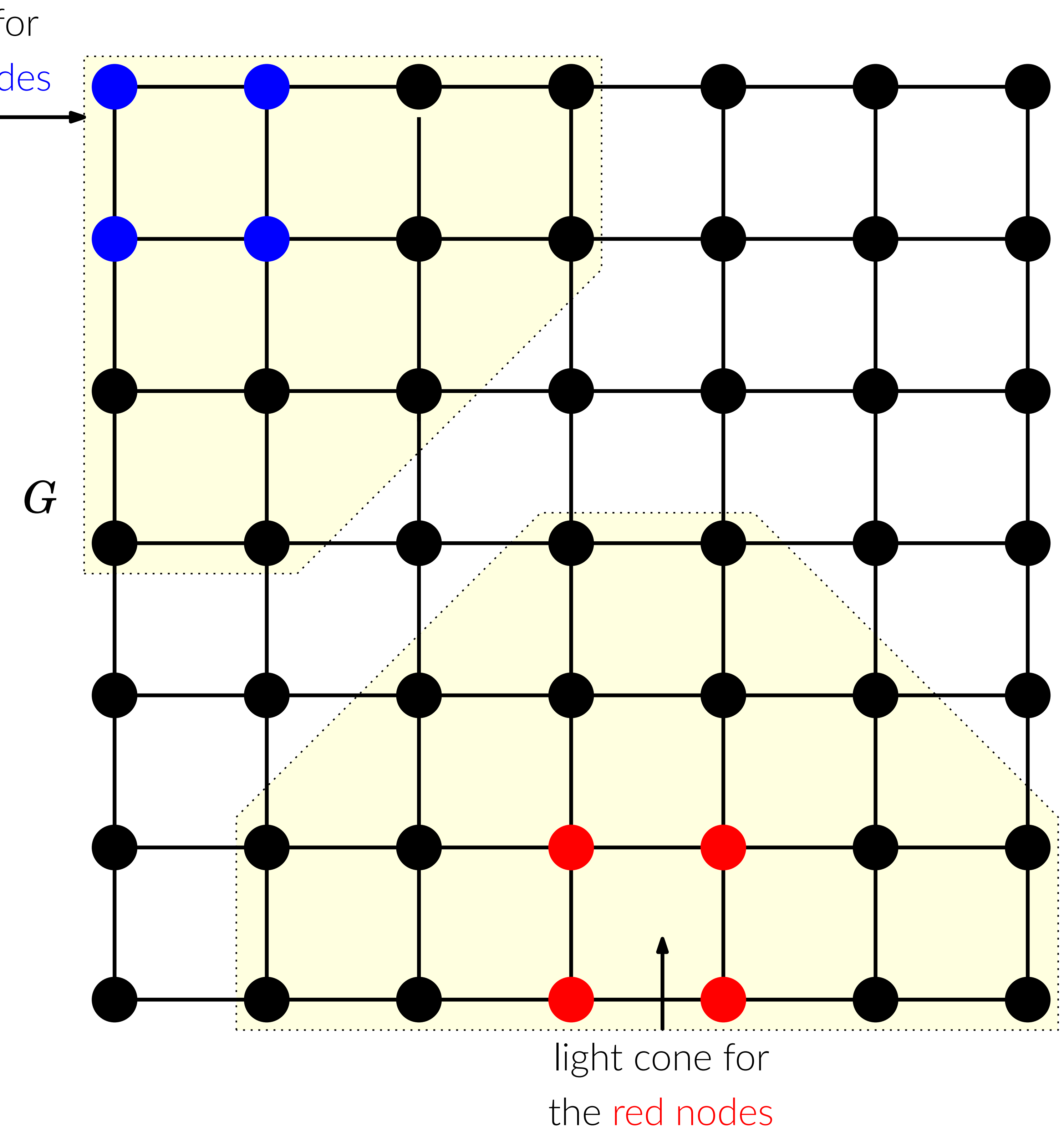
Properties of distributed algorithms

- **Run** a $\mathbf{2}$-round algorithm $\boldsymbol{A}$ in $\boldsymbol{G}$



Properties of distributed algorithms

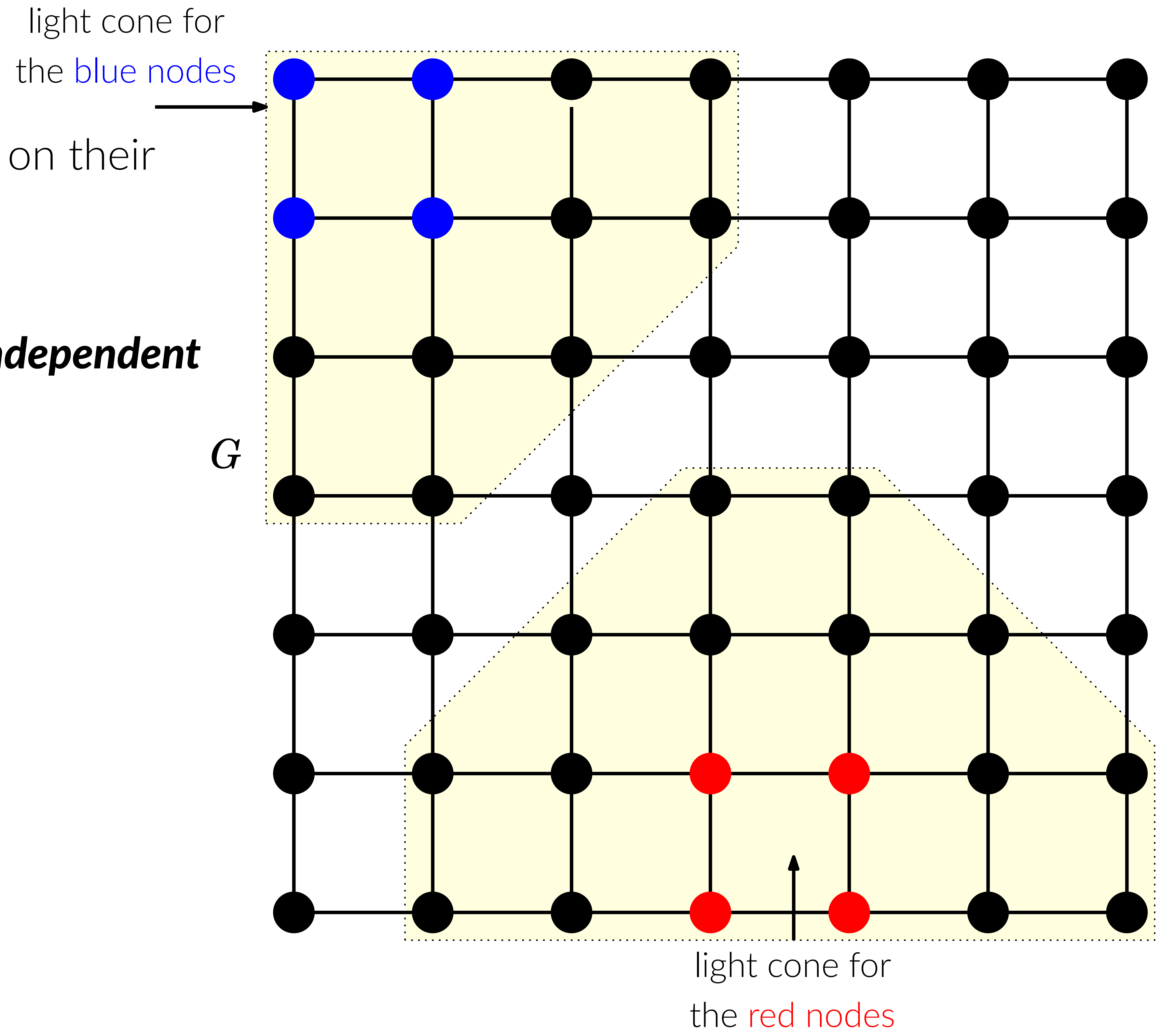
- **Run** a 2-round algorithm A in G
  - output for the red and blue nodes only depends on their respective light cones



Properties of distributed algorithms

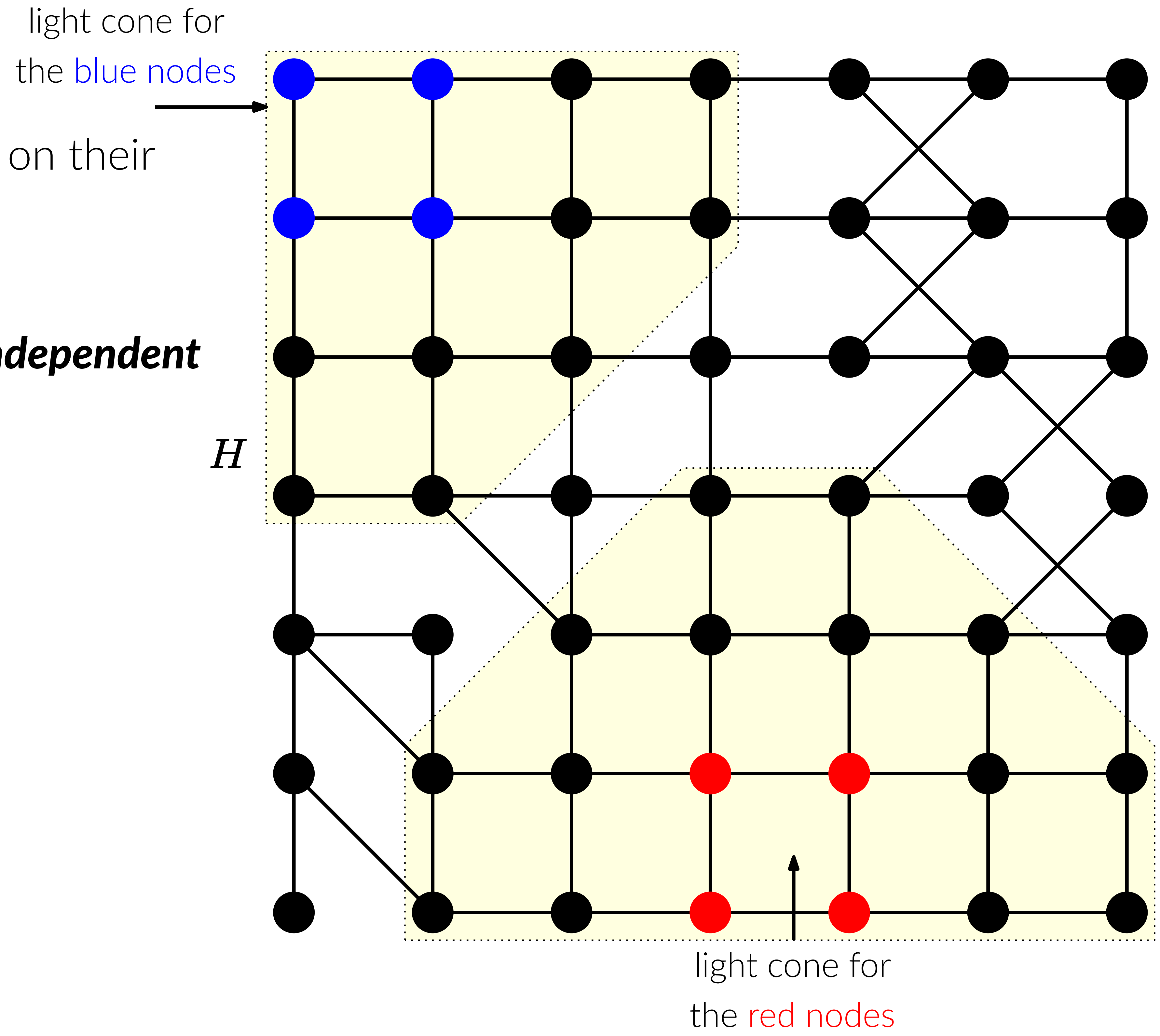
- **Run** a $\mathbf{2}$-round algorithm $\boldsymbol{A}$ in $\boldsymbol{G}$
  - output for the red and blue nodes only depends on their respective light cones

- **Output distributions** for red and blue nodes are ***independent***
  - as long as their distance is at least $\mathbf{5}$



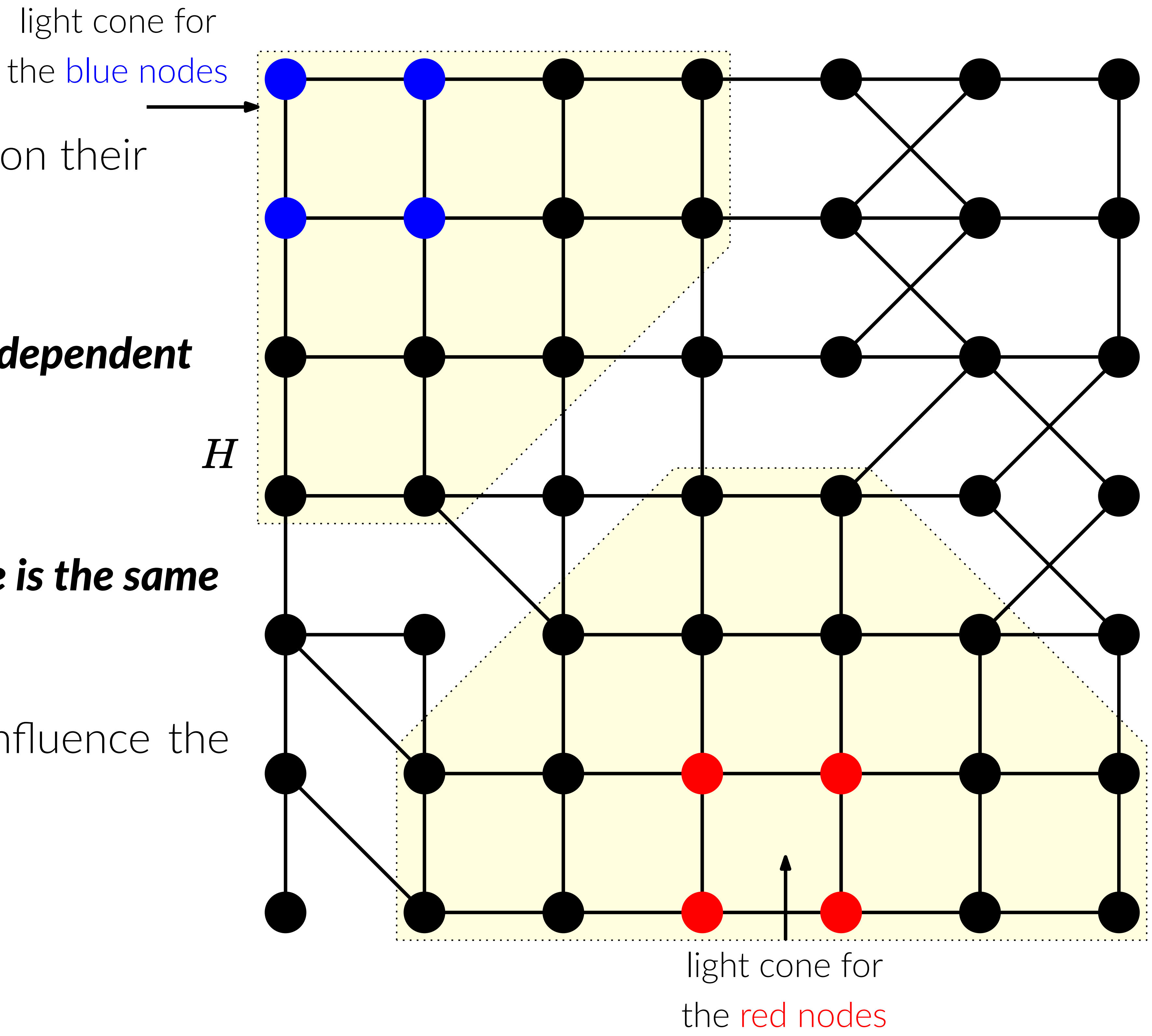
Properties of distributed algorithms

- **Run** a $\mathbf{2}$-round algorithm $\boldsymbol{A}$ in $\boldsymbol{G}$
  - output for the red and blue nodes only depends on their respective light cones
- **Output distributions** for red and blue nodes are ***independent***
  - as long as their distance is at least $\mathbf{5}$



Properties of distributed algorithms

- **Run** a 2-round algorithm A in G
  - output for the red and blue nodes only depends on their respective light cones
- **Output distributions** for red and blue nodes are ***independent***
  - as long as their distance is at least 5
- **Output distributions** remains ***the same*** if ***light cone is the same***
  - non-signaling property
  - changes that are beyond 2-hops away do not influence the output distribution
  - also known as causality



Abstracting output distributions

- A T-round distributed algorithm yields an **output distribution** with the following **properties**:
  - outputs of subsets of nodes at distance more than $2T$ are independent
  - non-signaling beyond distance T

Abstracting output distributions

- A T-round distributed algorithm yields an **output distribution** with the following **properties**:
  - outputs of subsets of nodes at distance more than $2T$ are independent
  - non-signaling beyond distance T
- Then we can ***just think about output distributions***!
  - computational models that produce directly distributions with the aforementioned properties

Abstracting output distributions

- A T-round distributed algorithm yields an **output distribution** with the following **properties**:
  - outputs of subsets of nodes at distance more than $2T$ are independent
  - non-signaling beyond distance T
- Then we can ***just think about output distributions***!
  - computational models that produce directly distributions with the aforementioned properties

$$\boxed{\text{bounded-dependence model}} \longrightarrow \boxed{\text{non-signaling LOCAL}}$$

locality T =
independence at distance $2T + 1$ plus
non-signaling beyond distance T

locality T =
non-signaling beyond distance T

[Holroyd and Liggett, Forum of Mathematics, Pi '14]
[STOC '25a]

* finitely-dependent distributions if $T = O(1)$

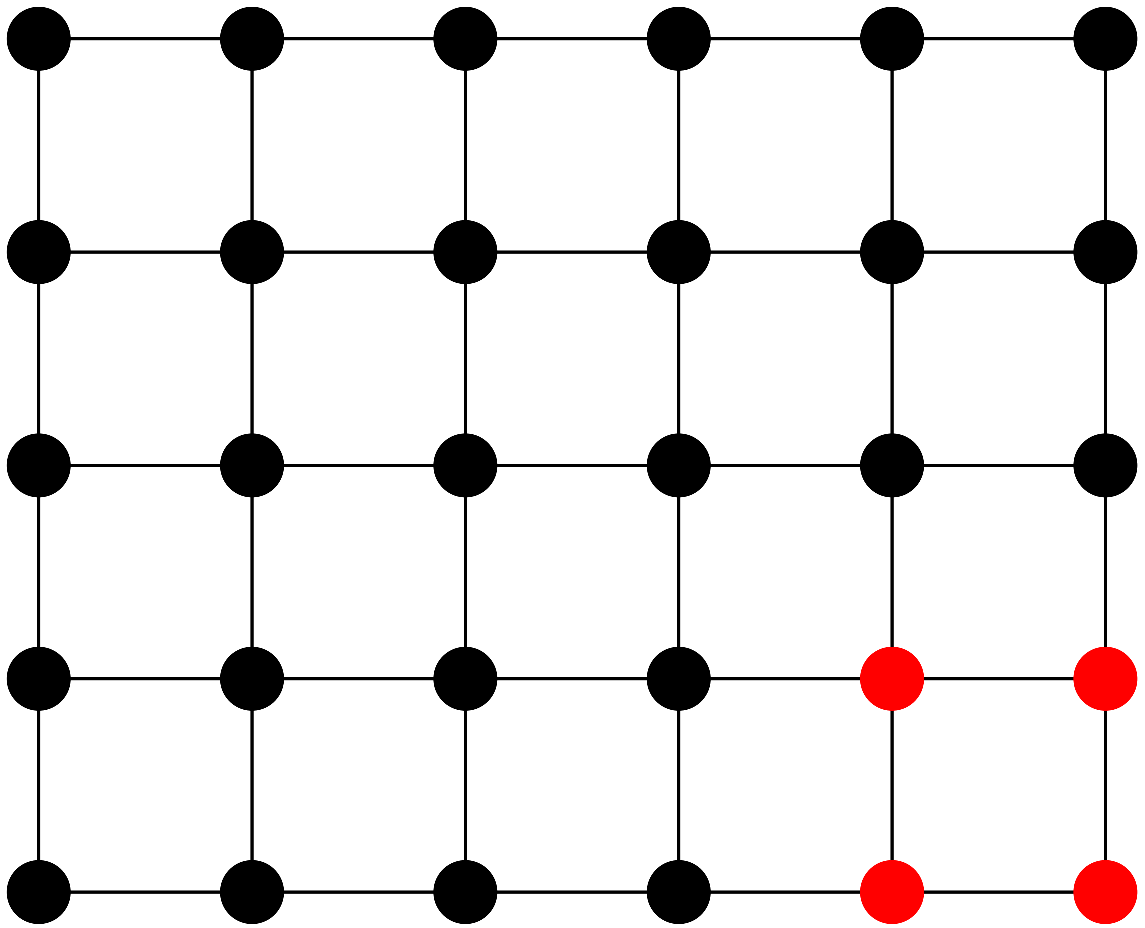
[Gavoille, Kosowski, and Markiewicz, DISC '09]
[Arfaoui and Fraigniaud, PODC '12 & SIGACT News '14]

Table of content

The non-signaling model

- Σ finite set of labels
  - always contains garbage output $\perp$

[STOC '24]

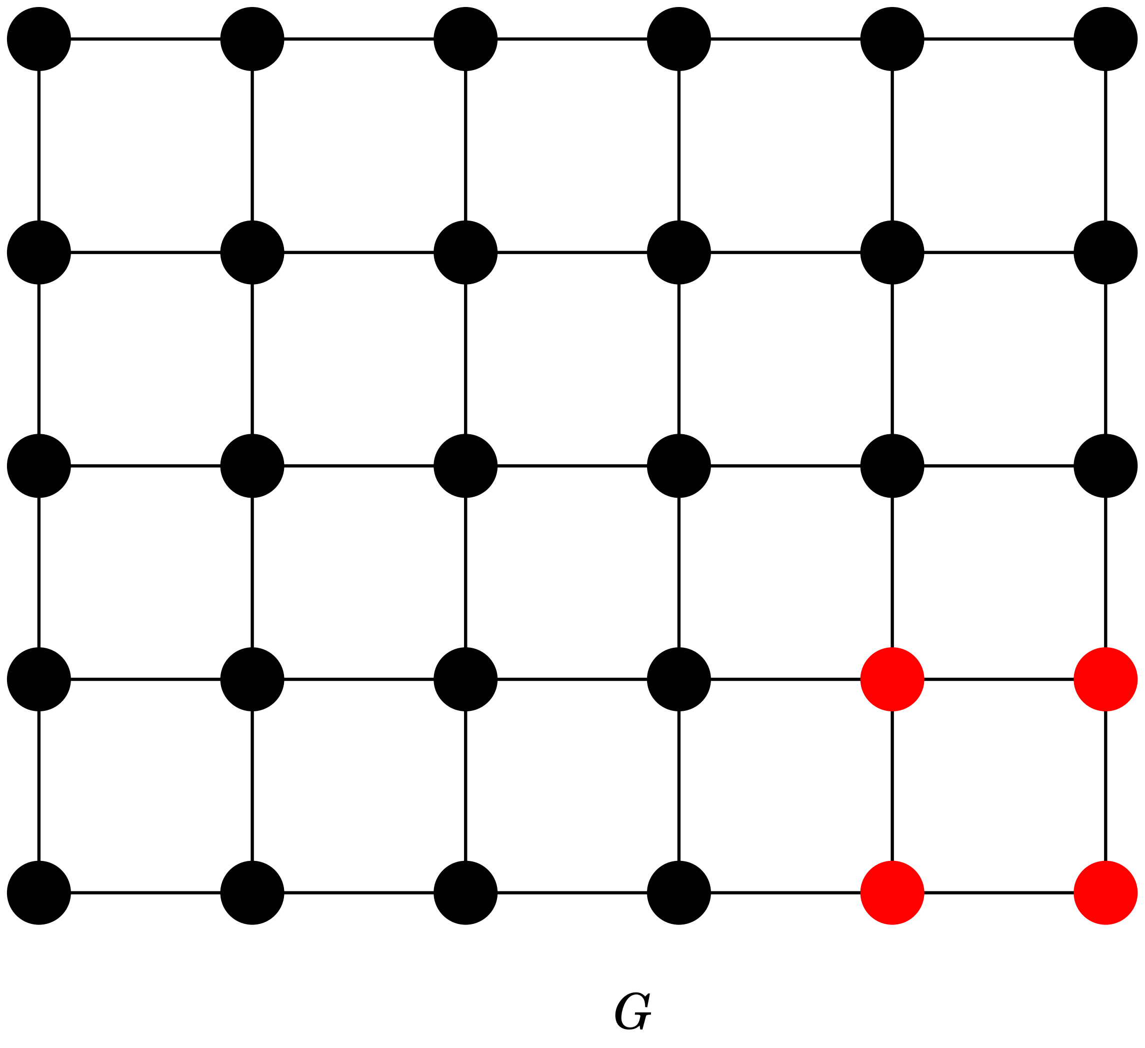


G

The non-signaling model

- Σ finite set of labels
  - always contains garbage output $\perp$
- **Outcome**: function $\mathbf{O}\colon (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$

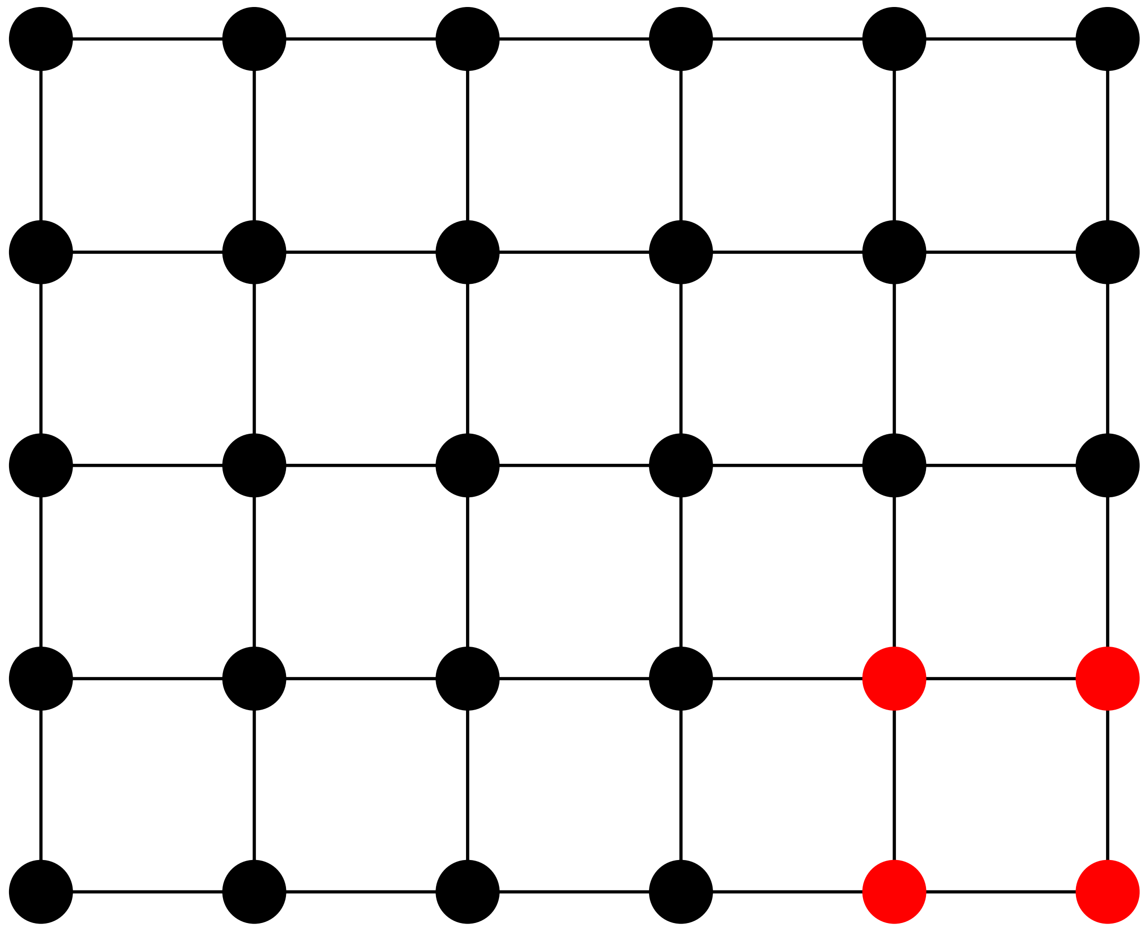
[STOC '24]



The non-signaling model

- Σ finite set of labels
  - always contains garbage output $\perp$
- **Outcome**: function $\mathbf{O}: (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
  - $y_i : V(G) \to \Sigma$ node labeling

[STOC '24]

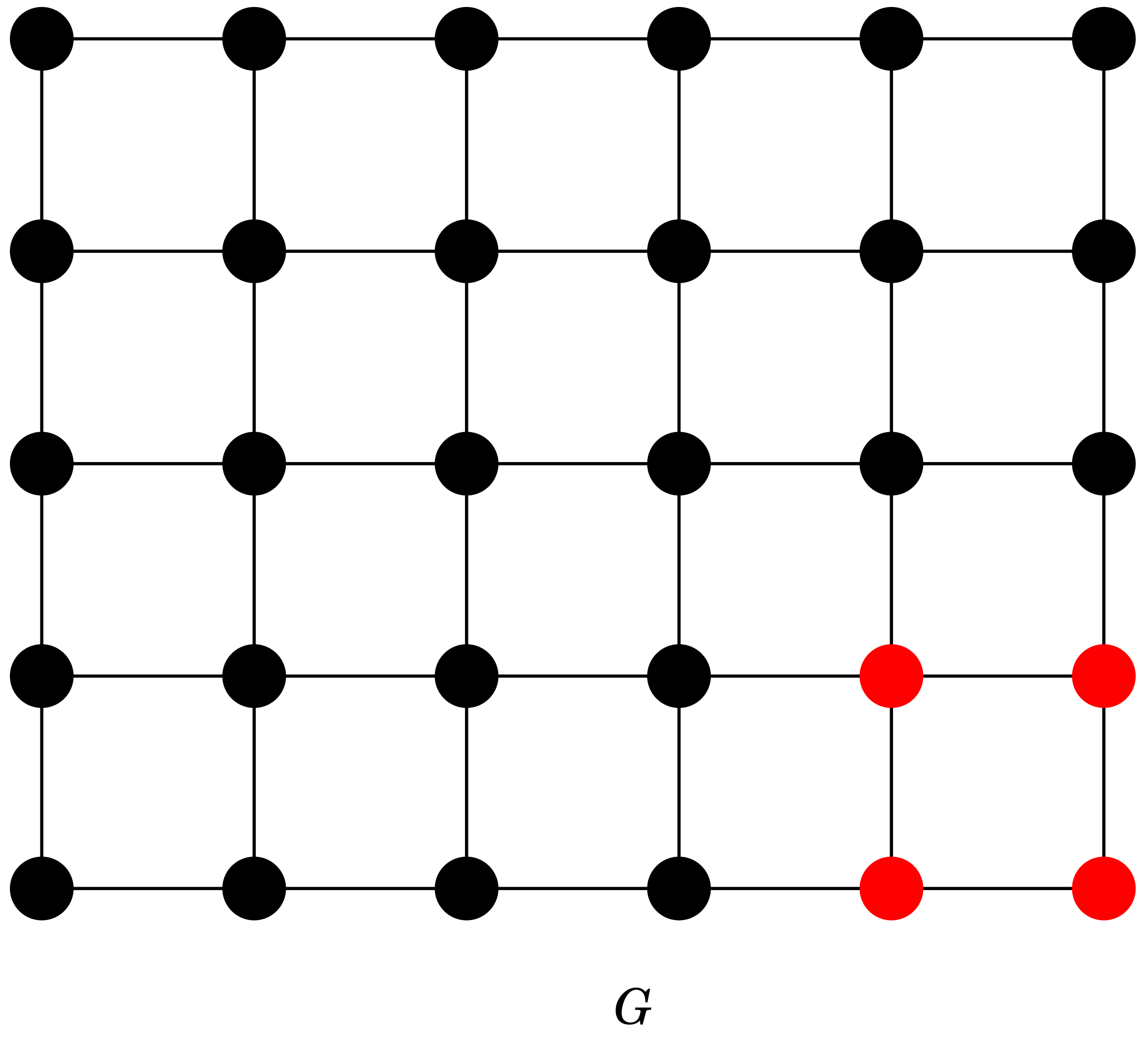


G

The non-signaling model

[STOC '24]

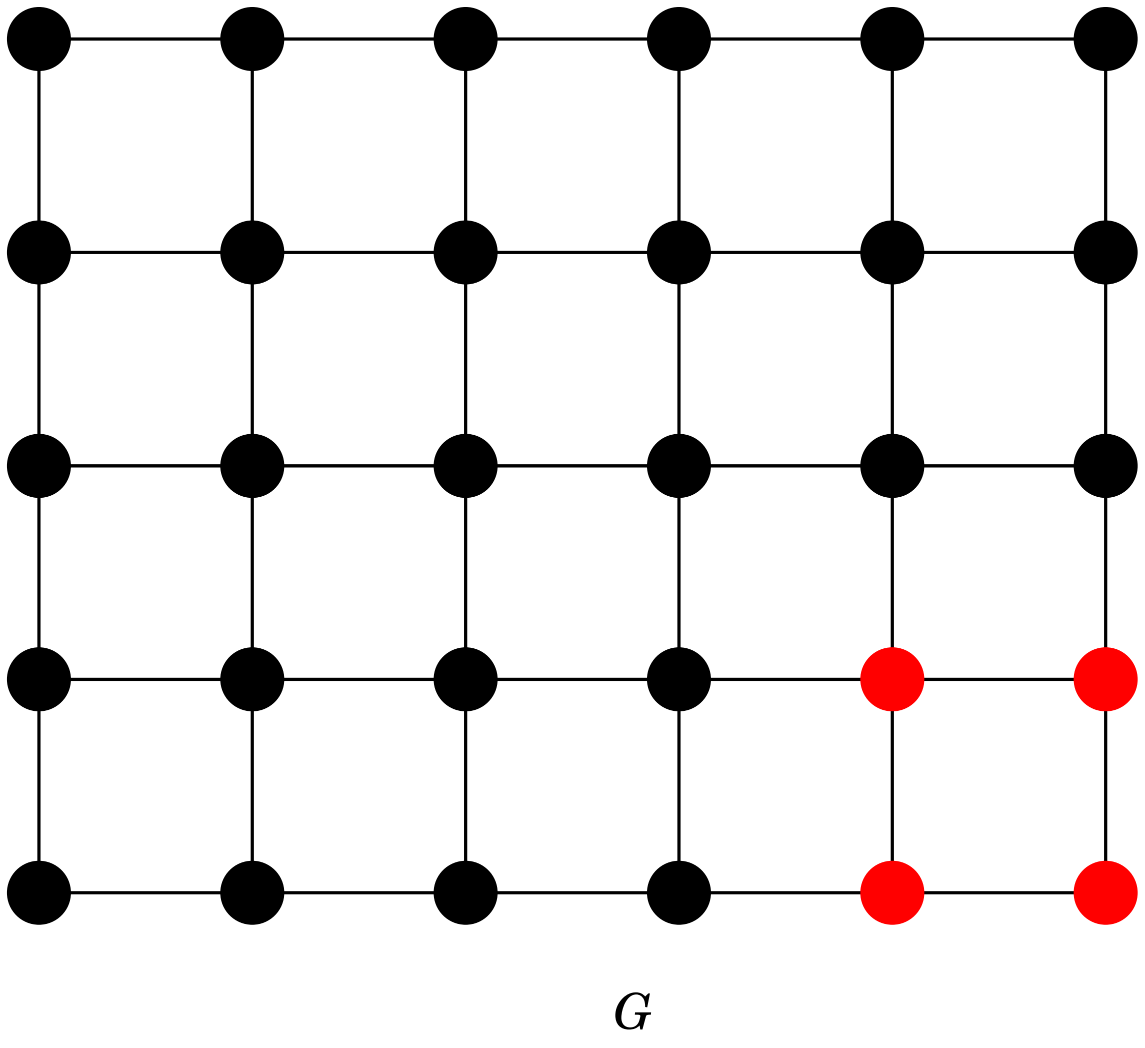
- Σ finite set of labels
  - always contains garbage output $\perp$
- **Outcome**: function $\mathbf{O}: (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
  - $y_i : V(G) \to \Sigma$ node labeling
  - $p_i \geq 0$
  - $\sum_{i \in I} p_i = 1$



The non-signaling model

[STOC '24]

- Σ finite set of labels
  - always contains garbage output $\perp$
- **Outcome**: function $\mathrm{O}: (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
  - $y_i : V(G) \to \Sigma$ node labeling
  - $p_i \geq 0$
  - $\sum_{i \in I} p_i = 1$
  - maps input graphs to probability distributions over output labelings



The non-signaling model

- Σ finite set of labels

  - always contains garbage output $\perp$

[STOC '24]

- **Outcome**: function $\mathbf{O}: (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$

  - $y_i : V(G) \to \Sigma$ node labeling

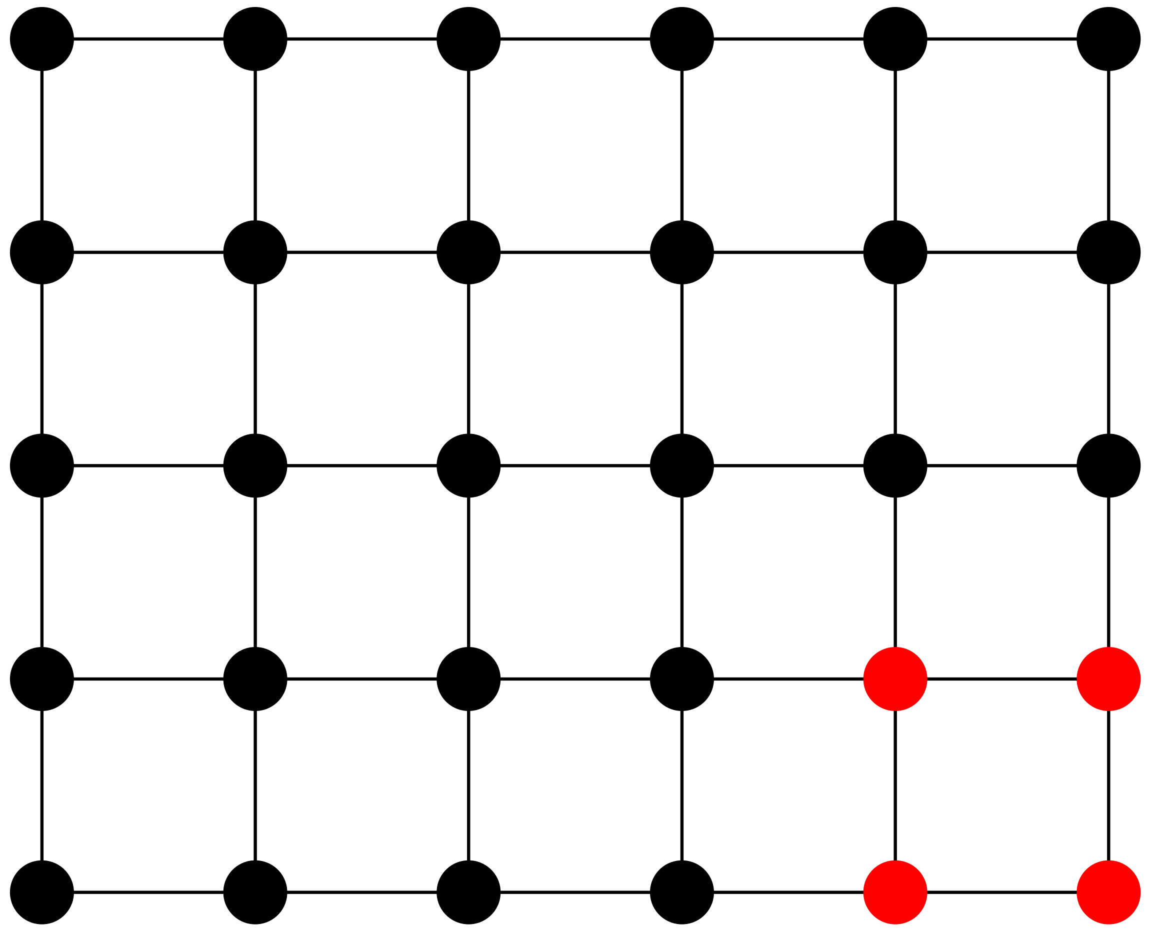
  - $p_i \geq 0$

  - $\sum_{i \in I} p_i = 1$

  - maps input graphs to probability distributions over output labelings

- **Non-signaling** beyond distance T

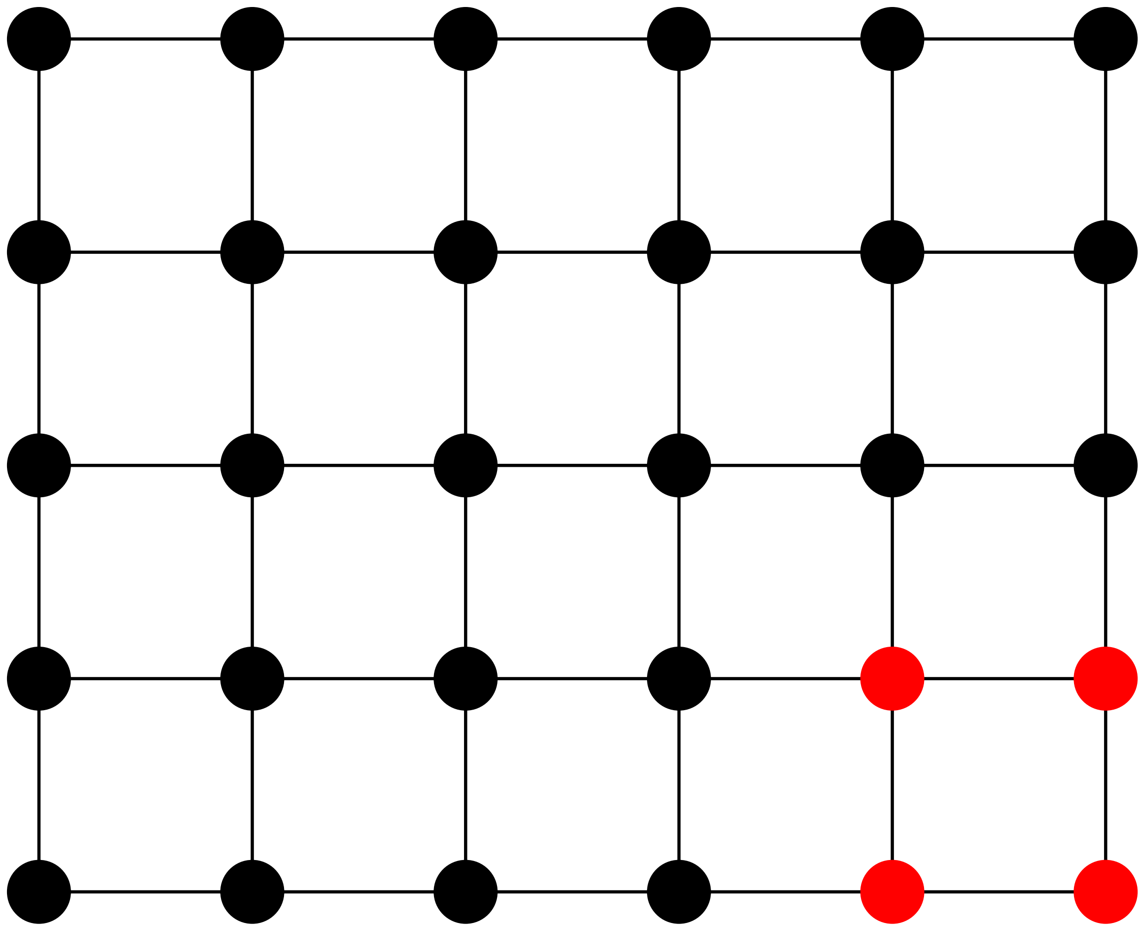
  - outcome $\mathbf{O}: (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$



The non-signaling model

[STOC '24]

- Σ finite set of labels
  - always contains garbage output $\perp$
- **Outcome**: function $\mathbf{O}: (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
  - $y_i : V(G) \to \Sigma$ node labeling
  - $p_i \geq 0$
  - $\sum_{i \in I} p_i = 1$
  - maps input graphs to probability distributions over output labelings
- **Non-signaling** beyond distance T
  - outcome $\mathbf{O}: (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
  - two inputs (G, x), (H, y)

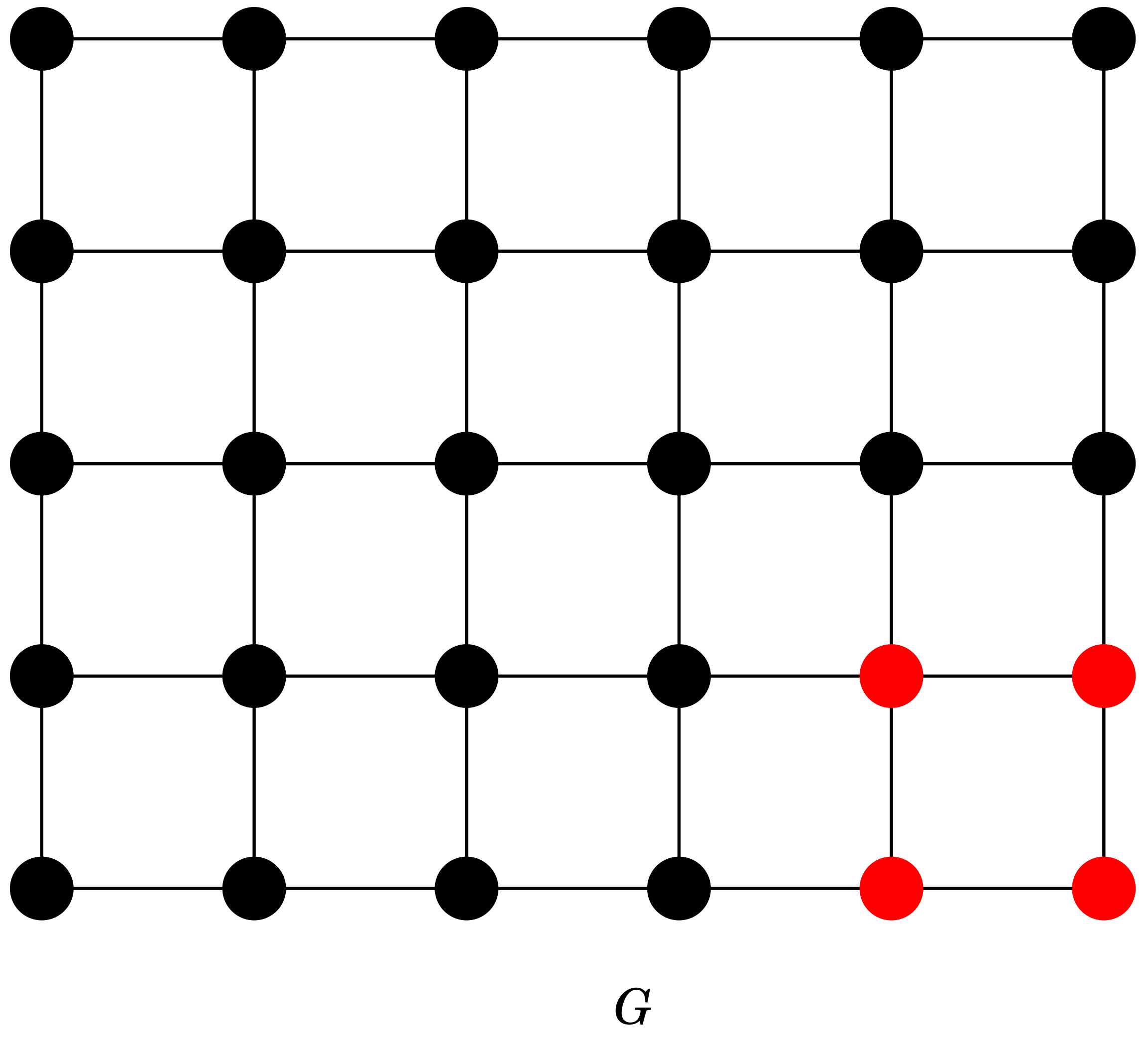


G

The non-signaling model

[STOC '24]

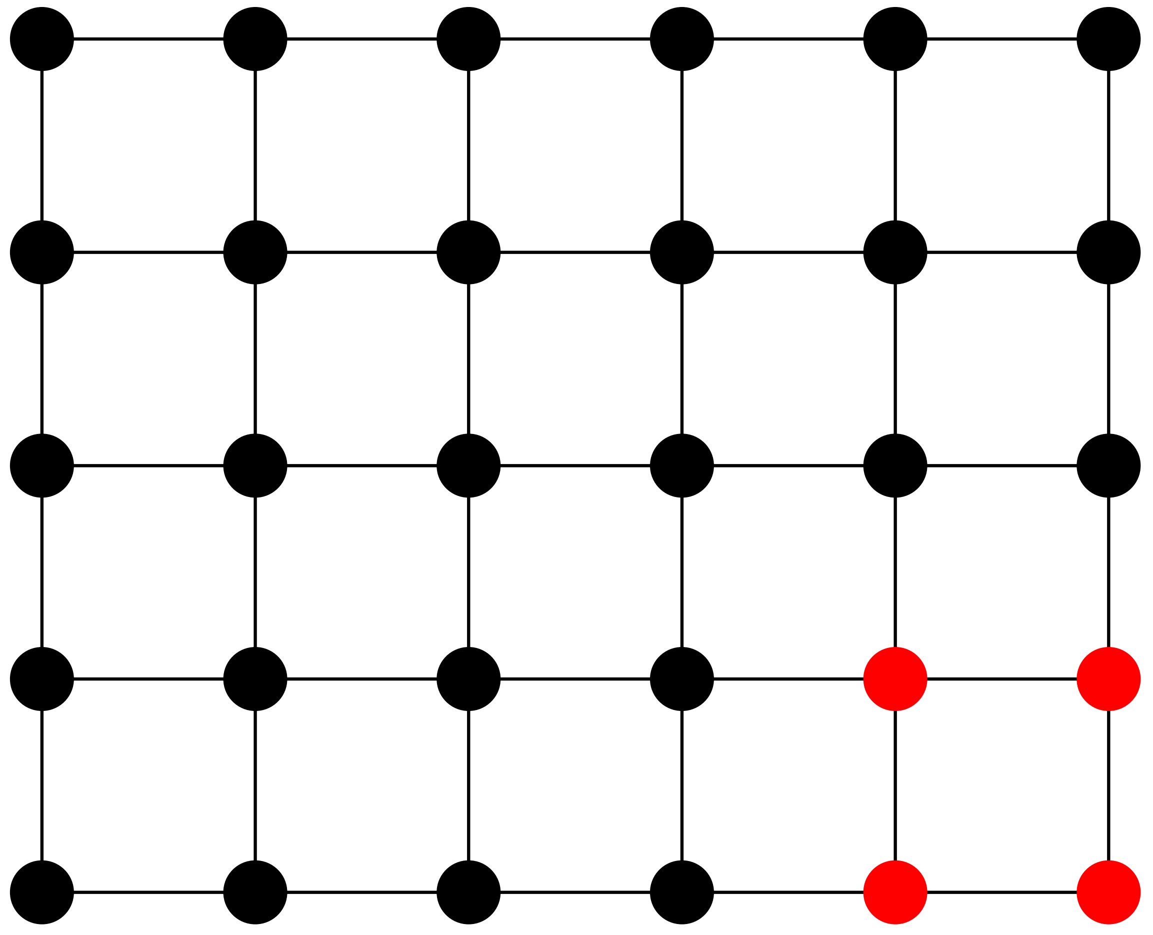
- Σ finite set of labels
  - always contains garbage output $\perp$
- **Outcome**: function $\mathbf{O}: (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
  - $y_i : V(G) \to \Sigma$ node labeling
  - $p_i \geq 0$
  - $\sum_{i \in I} p_i = 1$
  - maps input graphs to probability distributions over output labelings
- **Non-signaling** beyond distance T
  - outcome $\mathbf{O}: (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
  - two inputs (G, x), (H, y)
  - $S \subseteq V(G)$, $S' \subseteq V'(H)$ so that $G[S] \approx H[S']$ preserving x, y



The non-signaling model

[STOC '24]

- Σ finite set of labels
  - always contains garbage output $\perp$
- **Outcome**: function $\mathbf{O}: (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
  - $y_i : V(G) \to \Sigma$ node labeling
  - $p_i \geq 0$
  - $\sum_{i \in I} p_i = 1$
  - maps input graphs to probability distributions over output labelings
- **Non-signaling** beyond distance T
  - outcome $\mathbf{O}: (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
  - two inputs (G, x), (H, y)
  - $S \subseteq V(G)$, $S' \subseteq V'(H)$ so that $G[S] \approx H[S']$ preserving x, y
  - $\mathcal{V}_T(G[S], x) \approx \mathcal{V}_T(H[S'], y)$

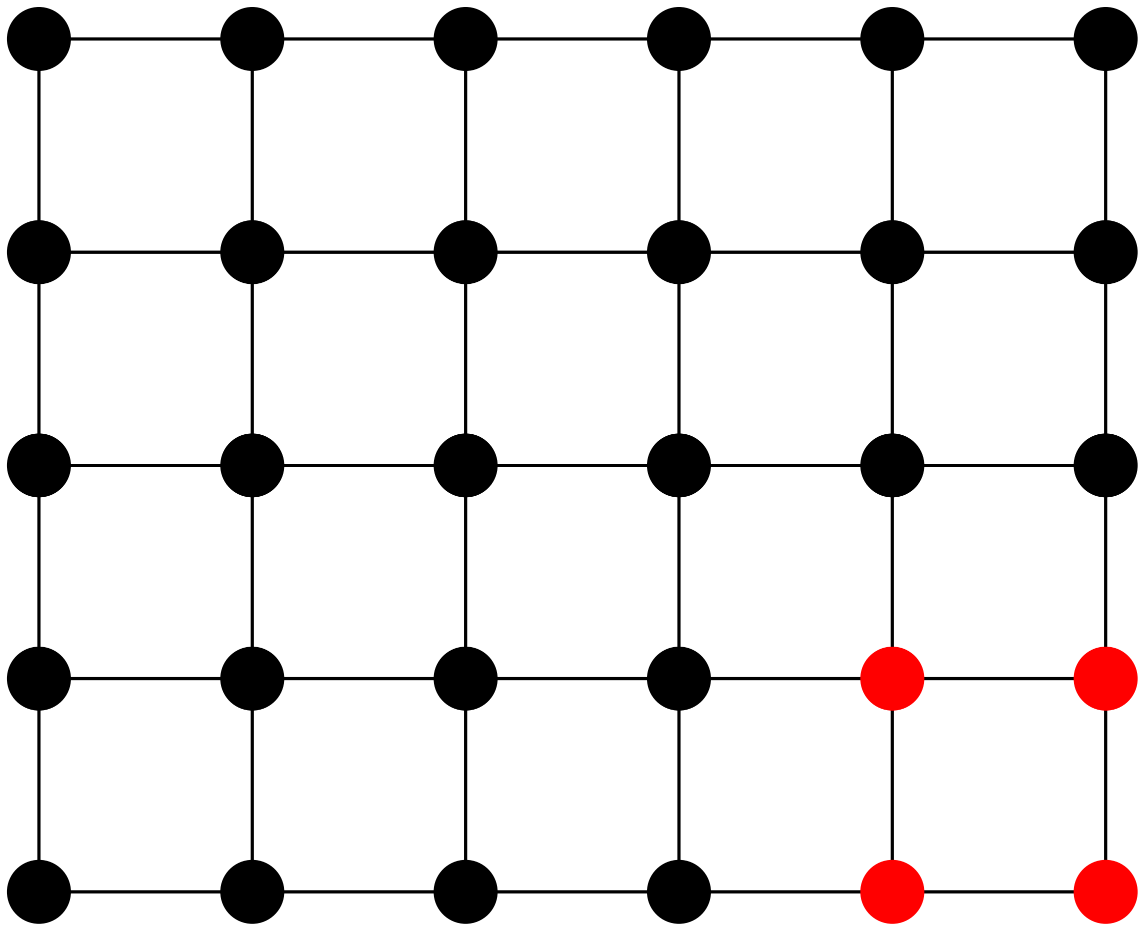


The non-signaling model

[STOC '24]

- Σ finite set of labels
  - always contains garbage output $\perp$
- **Outcome**: function $\mathbf{O}: (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
  - $y_i : V(G) \to \Sigma$ node labeling
  - $p_i \geq 0$
  - $\sum_{i \in I} p_i = 1$
  - maps input graphs to probability distributions over output labelings
- **Non-signaling** beyond distance T
  - outcome $\mathbf{O}: (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
  - two inputs (G, x), (H, y)
  - $S \subseteq V(G)$, $S' \subseteq V'(H)$ so that $G[S] \approx H[S']$ preserving x, y
  - $\mathcal{V}_T(G[S], x) \approx \mathcal{V}_T(H[S'], y)$

$\rightarrow$ - $\mathrm{O}(G, x) \restriction_S \approx \mathrm{O}(H, y) \restriction_{S'}$

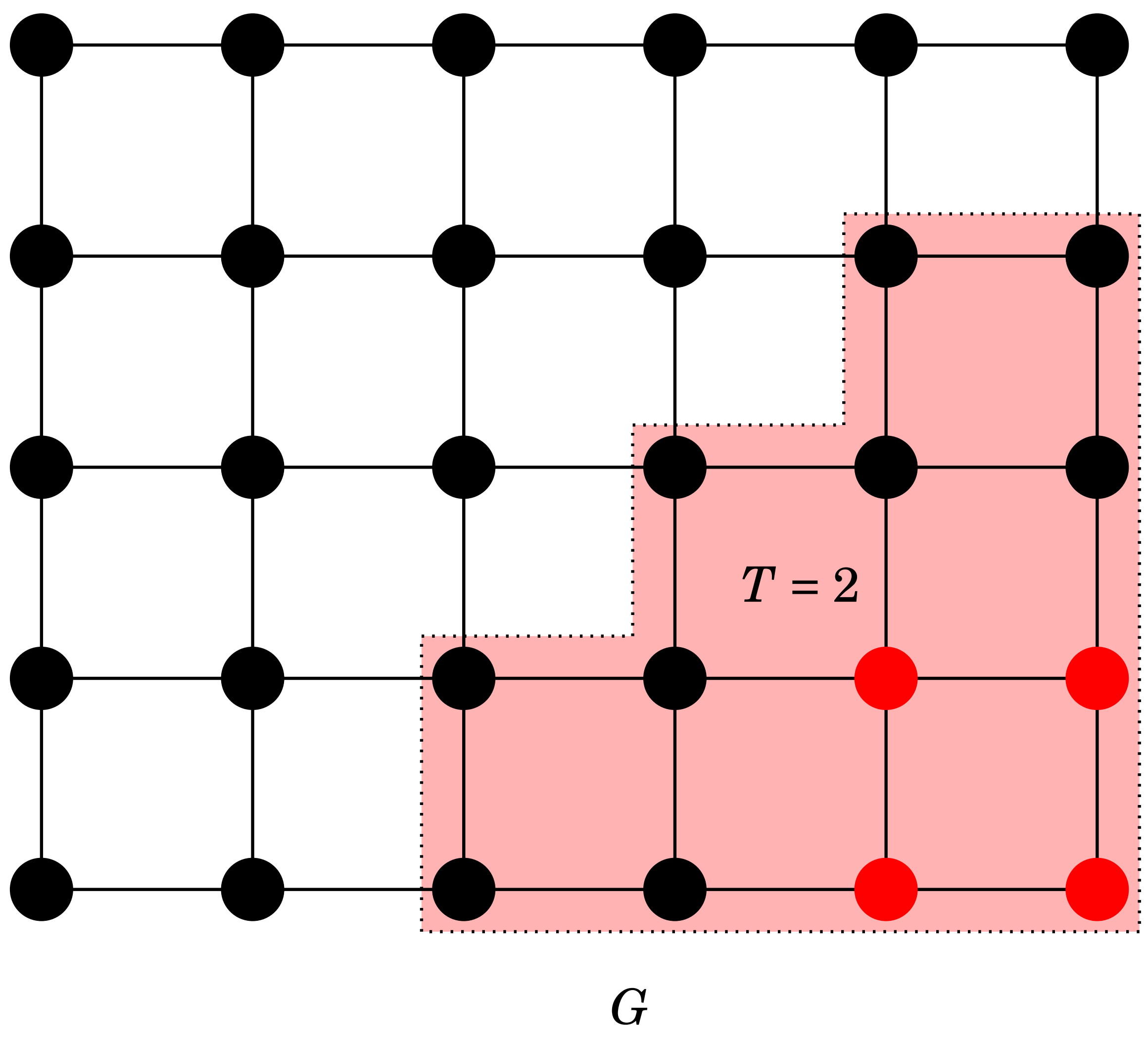


The non-signaling model

[STOC '24]

- Σ finite set of labels
  - always contains garbage output $\perp$
- **Outcome**: function $\mathbf{O}\colon (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
  - $y_i \colon V(G) \to \Sigma$ node labeling
  - $p_i \geq 0$
  - $\sum_{i \in I} p_i = 1$
  - maps input graphs to probability distributions over output labelings
- **Non-signaling** beyond distance T
  - outcome $\mathbf{O}\colon (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
  - two inputs (G, x), (H, y)
  - $S \subseteq V(G)$, $S' \subseteq V'(H)$ so that $G[S] \approx H[S']$ preserving x, y
  - $\mathcal{V}_T(G[S], x) \approx \mathcal{V}_T(H[S'], y)$

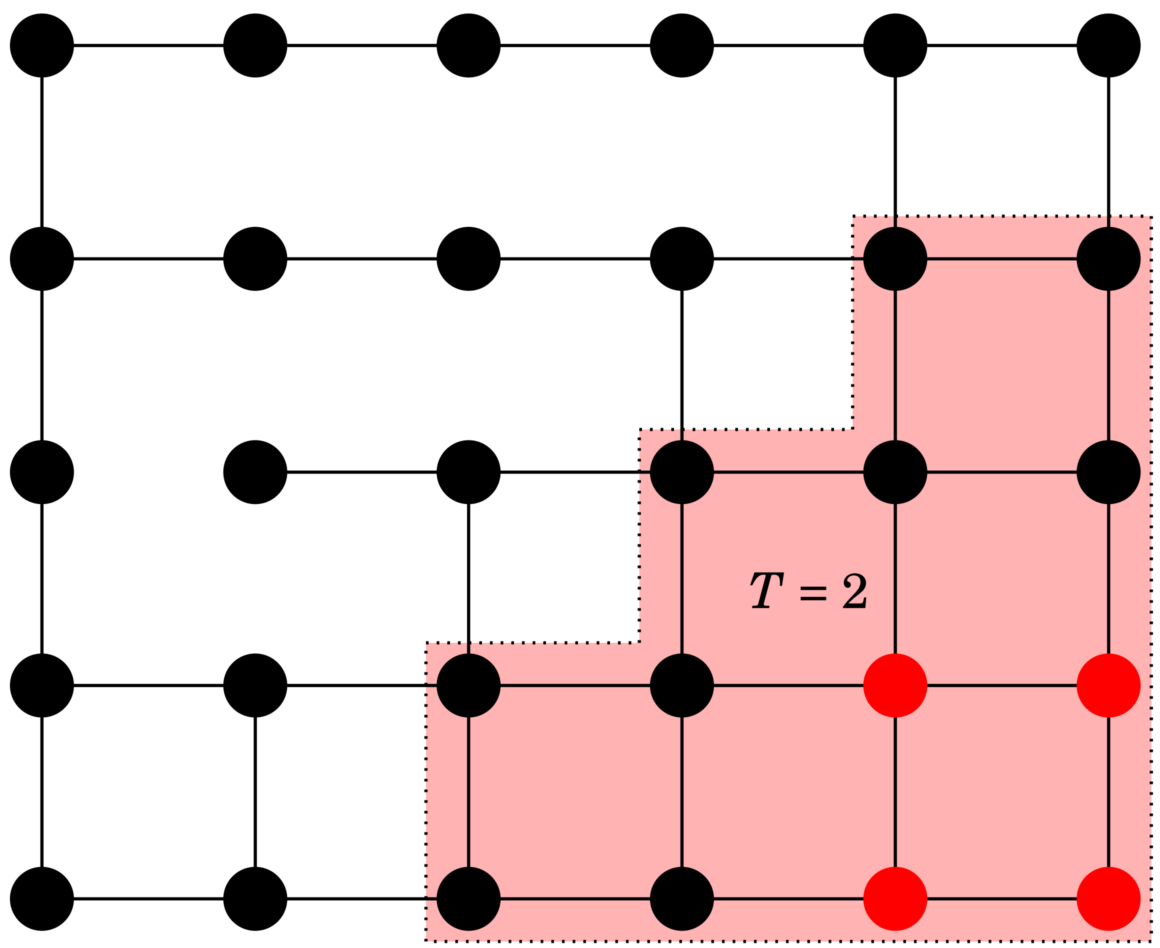
$\rightarrow$ - $\mathrm{O}(G, x) \restriction_S \approx \mathrm{O}(H, y) \restriction_{S'}$



The non-signaling model

[STOC '24]

- Σ finite set of labels
  - always contains garbage output $\perp$
- **Outcome**: function $\mathbf{O}\colon (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
  - $y_i \colon V(G) \to \Sigma$ node labeling
  - $p_i \geq 0$
  - $\sum_{i \in I} p_i = 1$
  - maps input graphs to probability distributions over output labelings



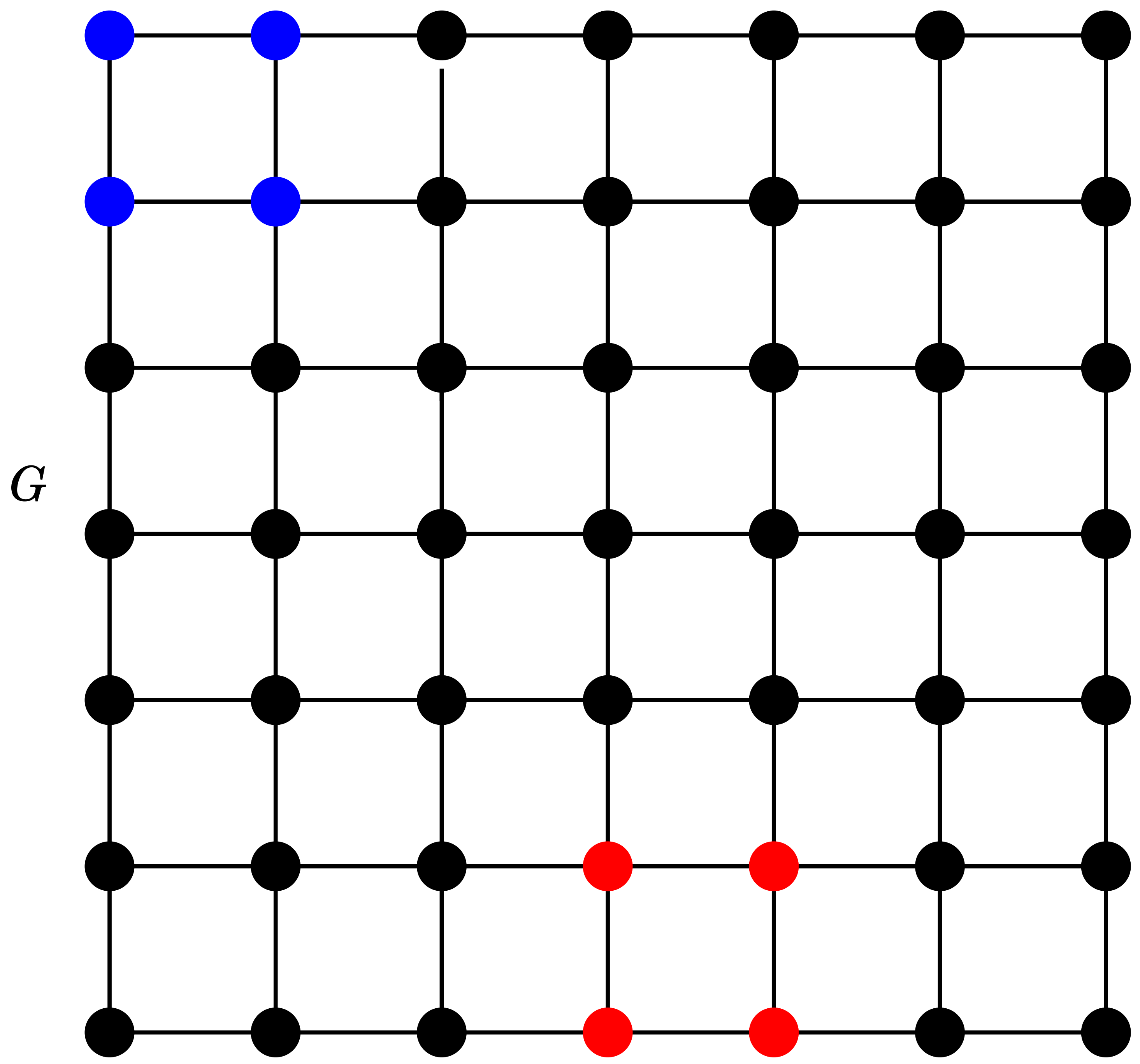
- **Non-signaling** beyond distance T
  - outcome $\mathbf{O}\colon (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
  - two inputs (G, x), (H, y)
  - $S \subseteq V(G)$, $S' \subseteq V'(H)$ so that $G[S] \approx H[S']$ preserving x, y
  - $\mathcal{V}_T(G[S], x) \approx \mathcal{V}_T(H[S'], y)$

$\longrightarrow$ - $\mathrm{O}(G, x) \restriction_S \approx \mathrm{O}(H, y) \restriction_{S'}$

The bounded-dependence model

[STOC '25a]

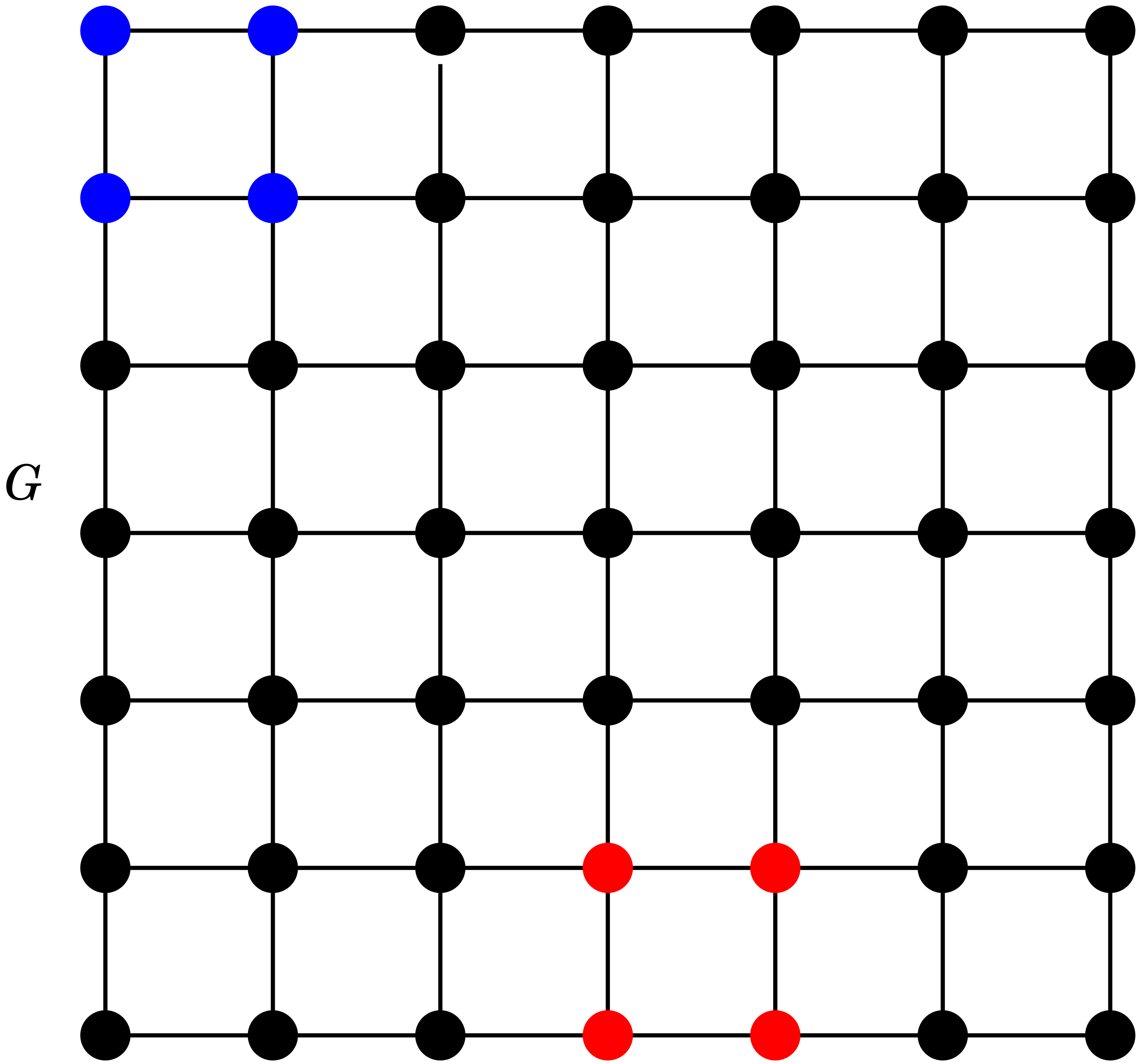
- **Outcome**: function $\mathbf{O}\colon (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
- **Non-signaling** beyond distance T



The bounded-dependence model

[STOC '25a]

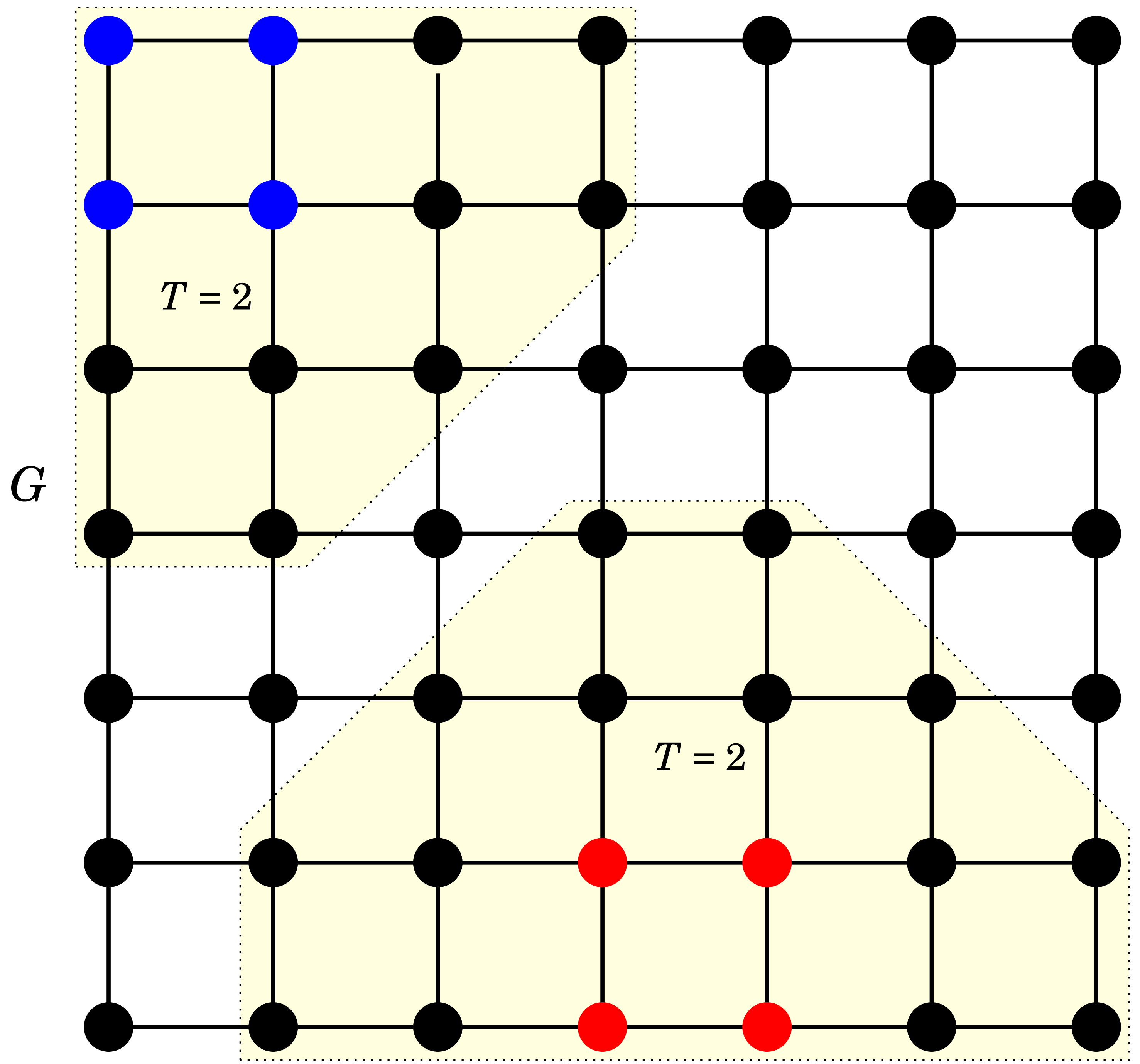
- **Outcome**: function $\mathbf{O}\colon (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
- **Non-signaling** beyond distance T
- **Independence** at distance $\geq 2T + 1$



The bounded-dependence model

[STOC '25a]

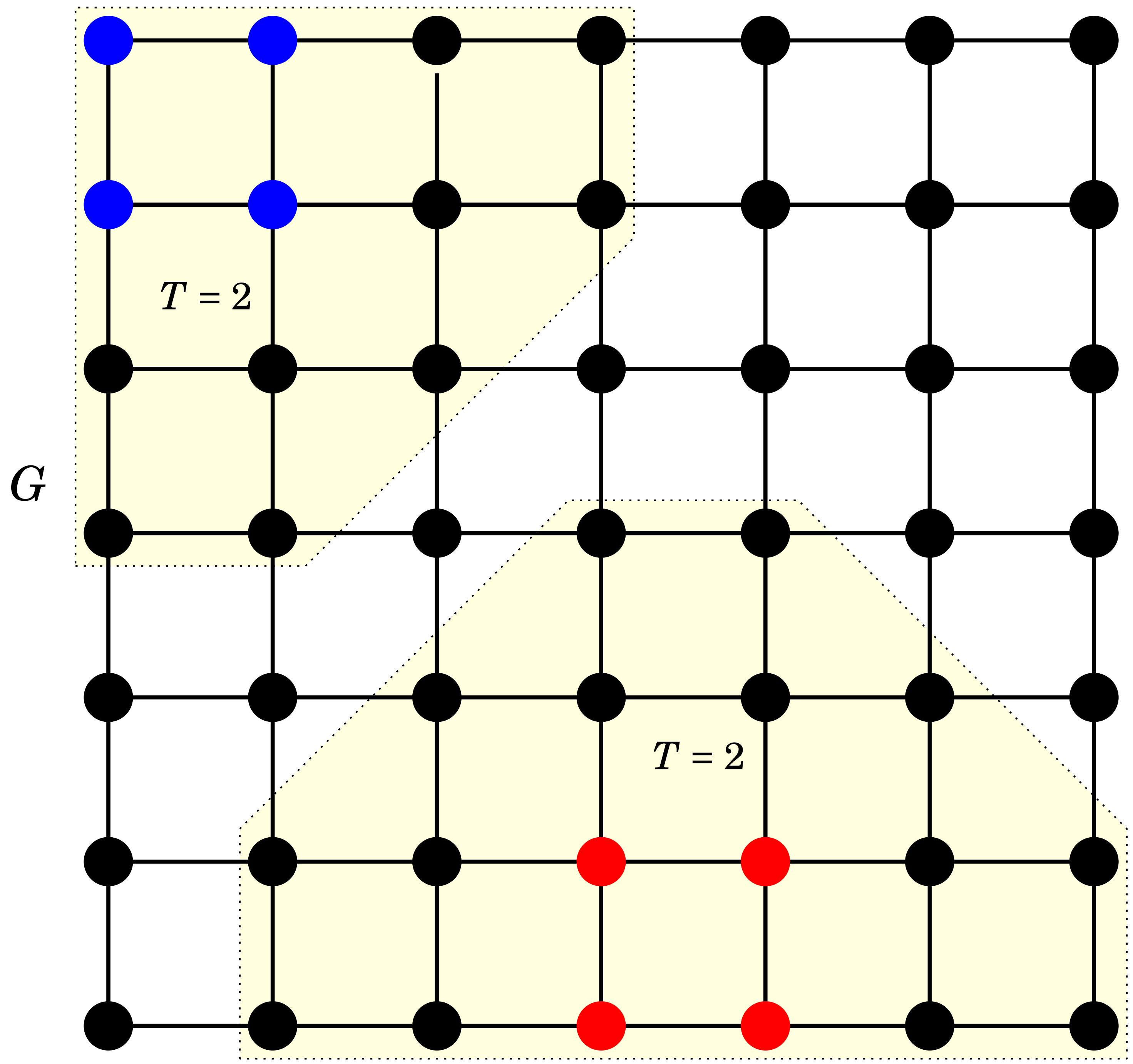
- **Outcome**: function $\mathbf{O}: (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
- **Non-signaling** beyond distance T
- **Independence** at distance $\geq 2T + 1$



The bounded-dependence model

[STOC '25a]

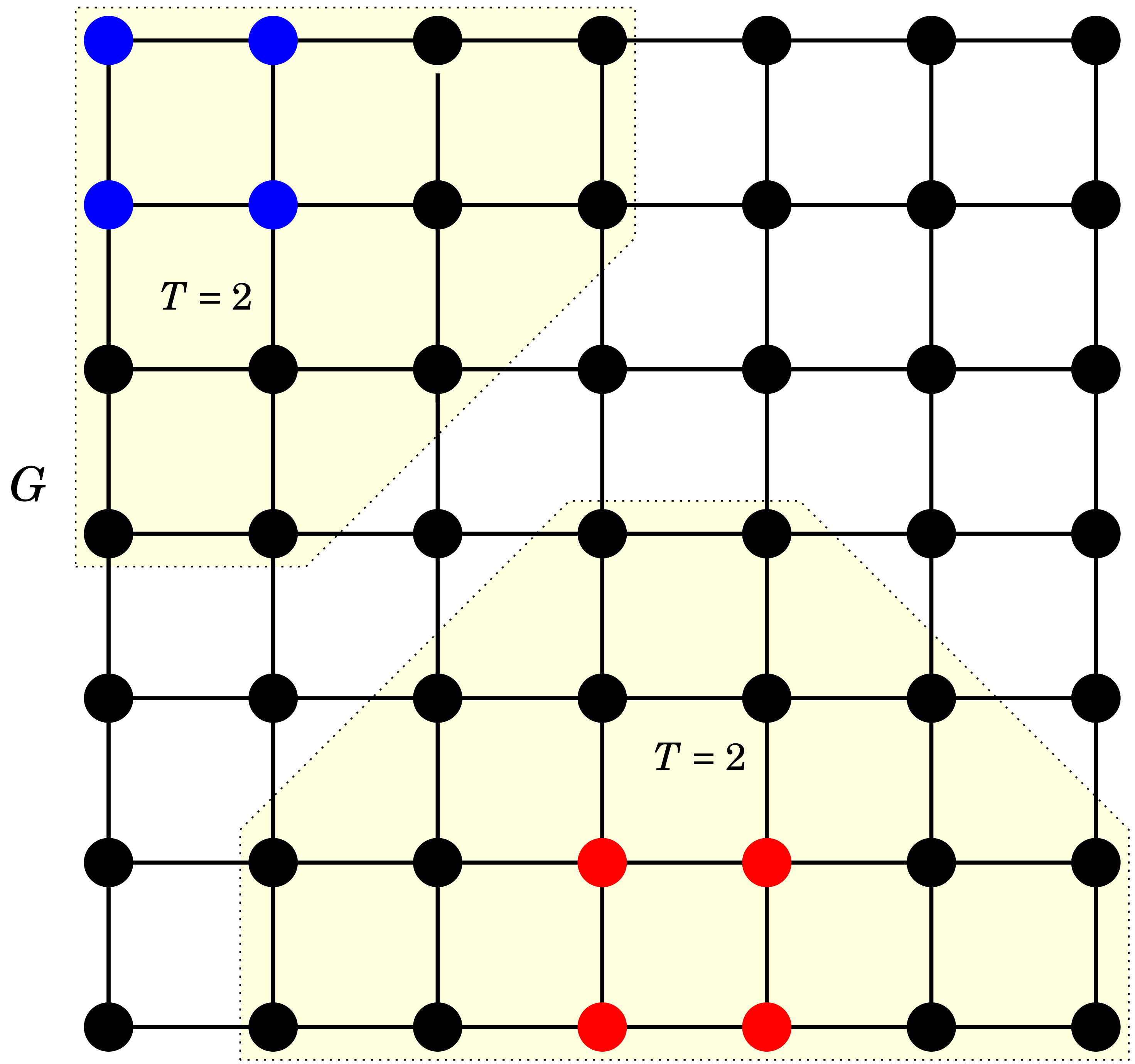
- **Outcome**: function $\mathbf{O}\colon (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
- **Non-signaling** beyond distance T
- **Independence** at distance $\geq 2T + 1$
- Bounded-dependent distribution with locality T



The bounded-dependence model

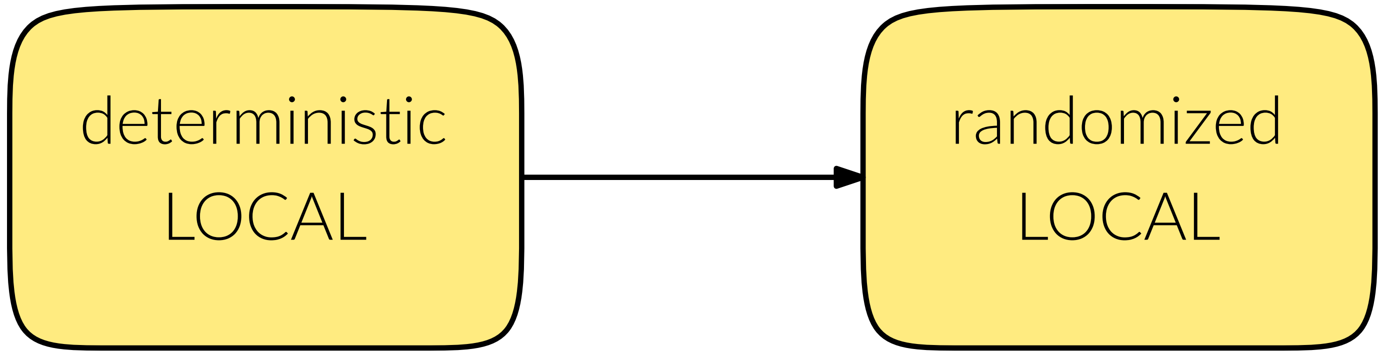
[STOC '25a]

- **Outcome**: function $\mathbf{O}: (G, x) \mapsto \{(y_i, p_i)\}_{i \in I}$
- **Non-signaling** beyond distance T
- **Independence** at distance $\geq 2T + 1$
- Bounded-dependent distribution with locality T
  - if $T = O(1)$, *finitely-dependent* distribution



Relations among models

- $X \to Y$ means that locality T in X becomes locality $O(T)$ in Y

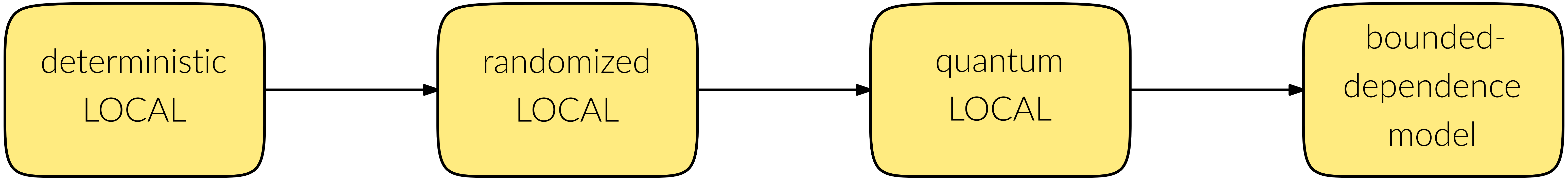


Relations among models

- $X \to Y$ means that locality T in X becomes locality $O(T)$ in Y

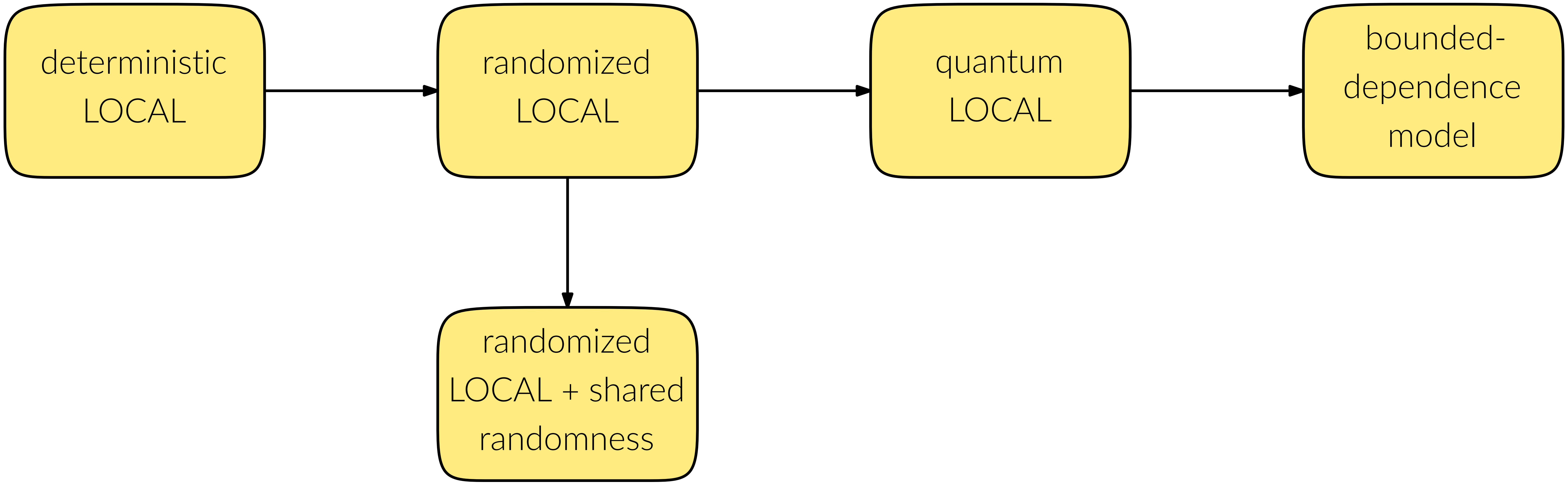
Relations among models

- $X \to Y$ means that locality T in X becomes locality $O(T)$ in Y



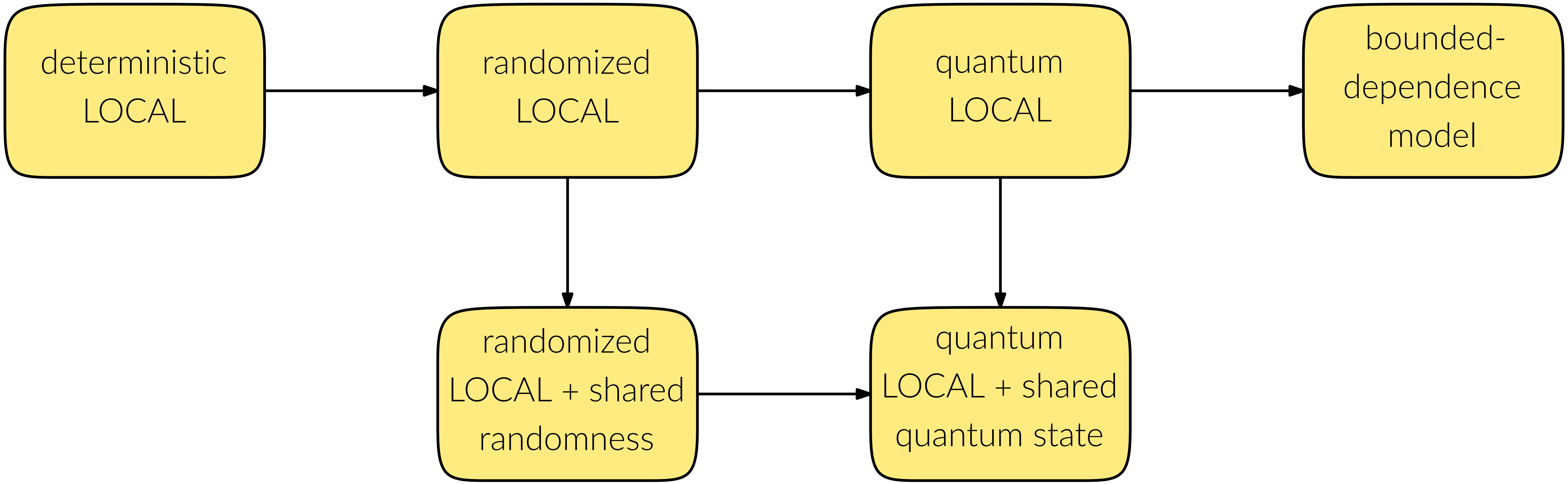
Relations among models

- $X \to Y$ means that locality T in X becomes locality $O(T)$ in Y



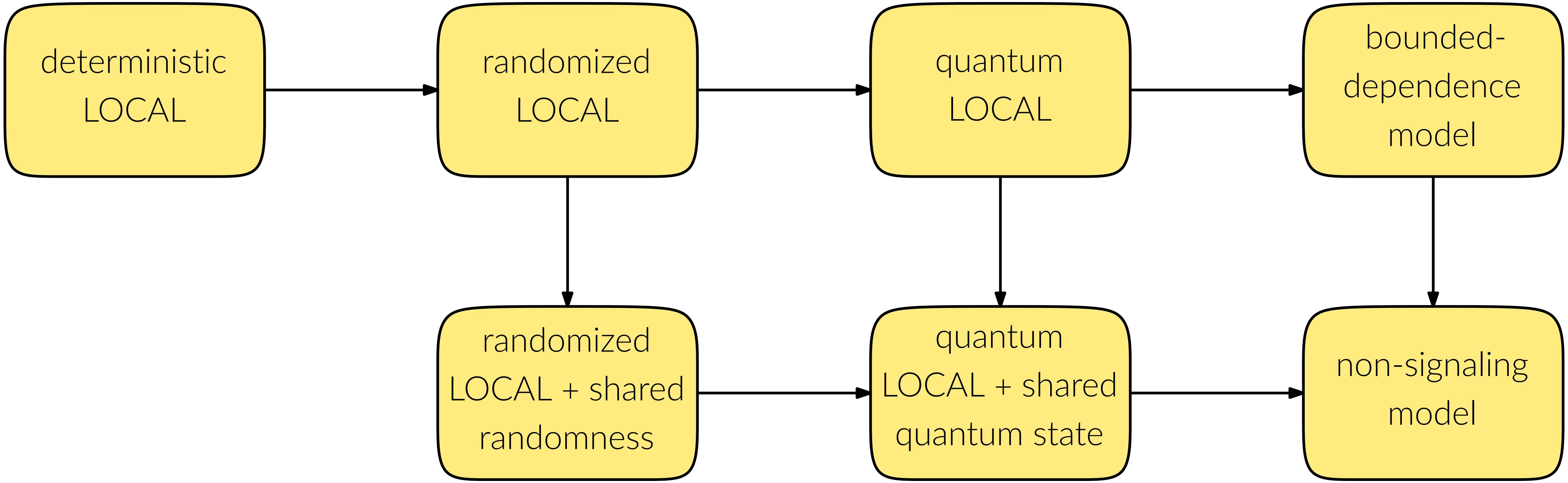
Relations among models

- $X \to Y$ means that locality T in X becomes locality $O(T)$ in Y



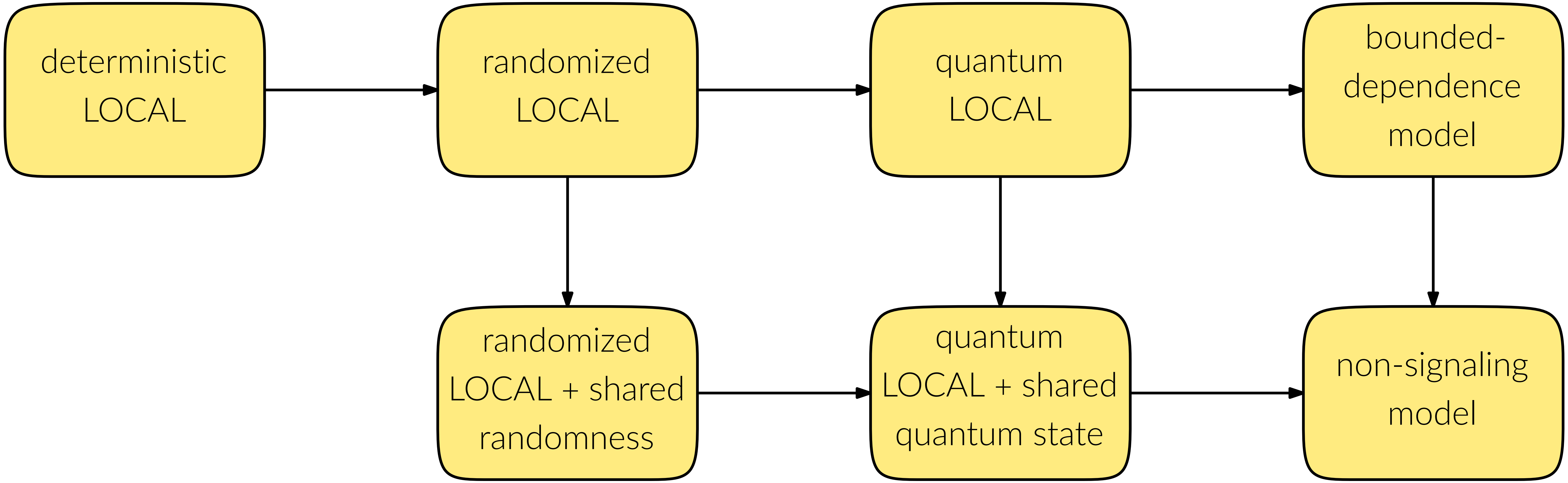
Relations among models

- $X \to Y$ means that locality T in X becomes locality $O(T)$ in Y



Relations among models

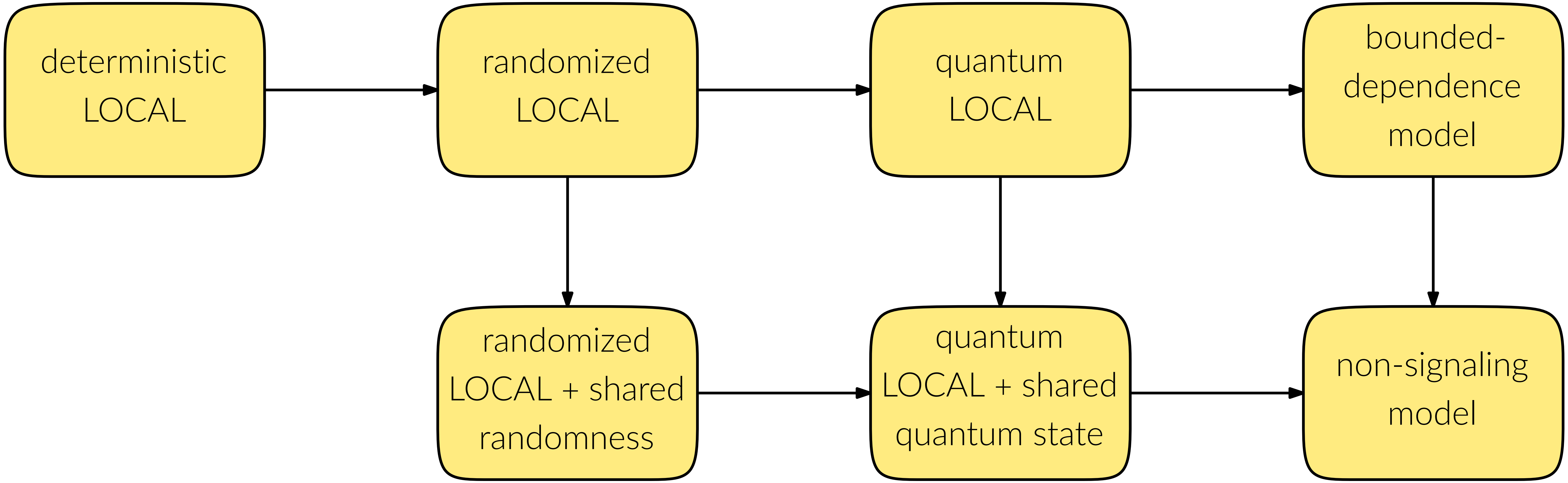
- $X \to Y$ means that locality T in X becomes locality $O(T)$ in Y



- Is it possible to **“sandwich” quantum-LOCAL** between weaker and stronger models?

Relations among models

- $X \to Y$ means that locality T in X becomes locality $O(T)$ in Y

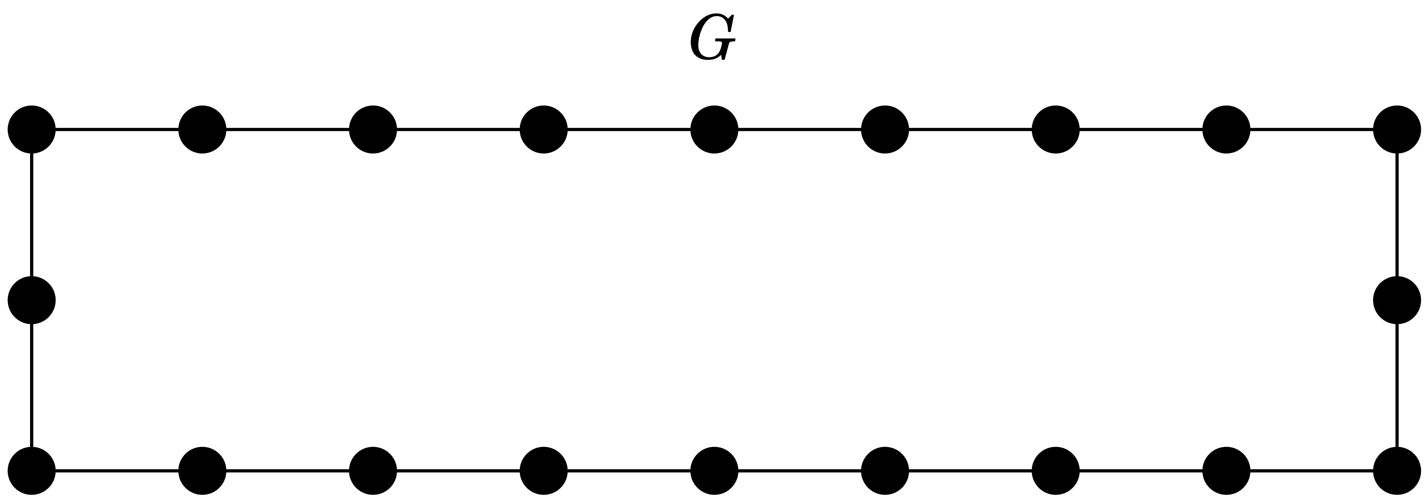


- Is it possible to **"sandwich" quantum-LOCAL** between weaker and stronger models?

  - yes!

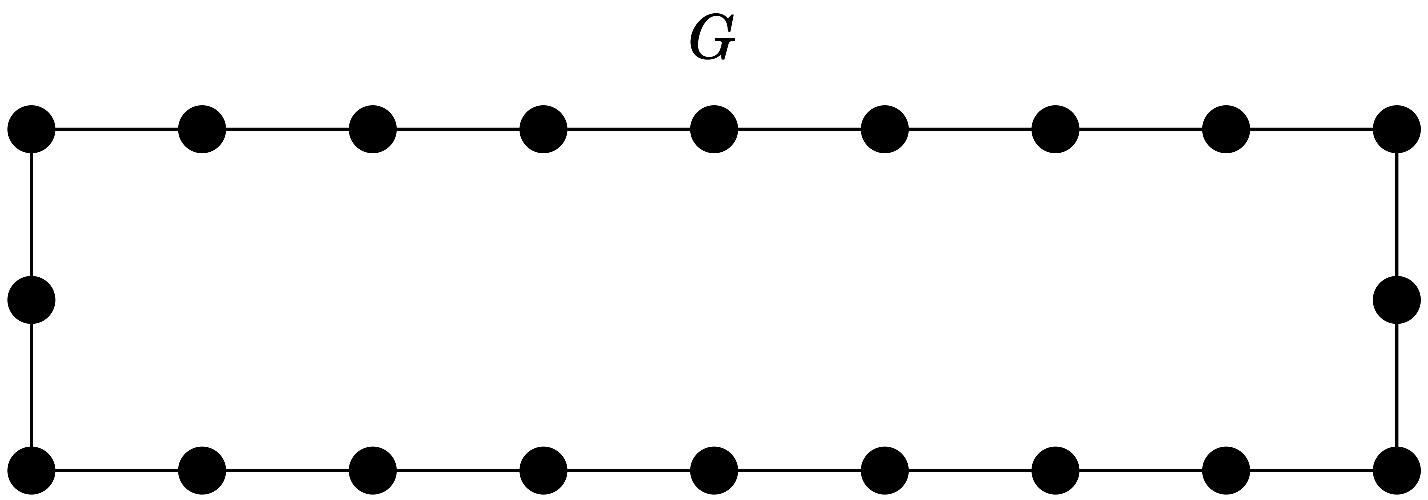
Indistinguishability argument

- **Problem**: 2-coloring paths & even cycles in the *non-signaling* model



Indistinguishability argument

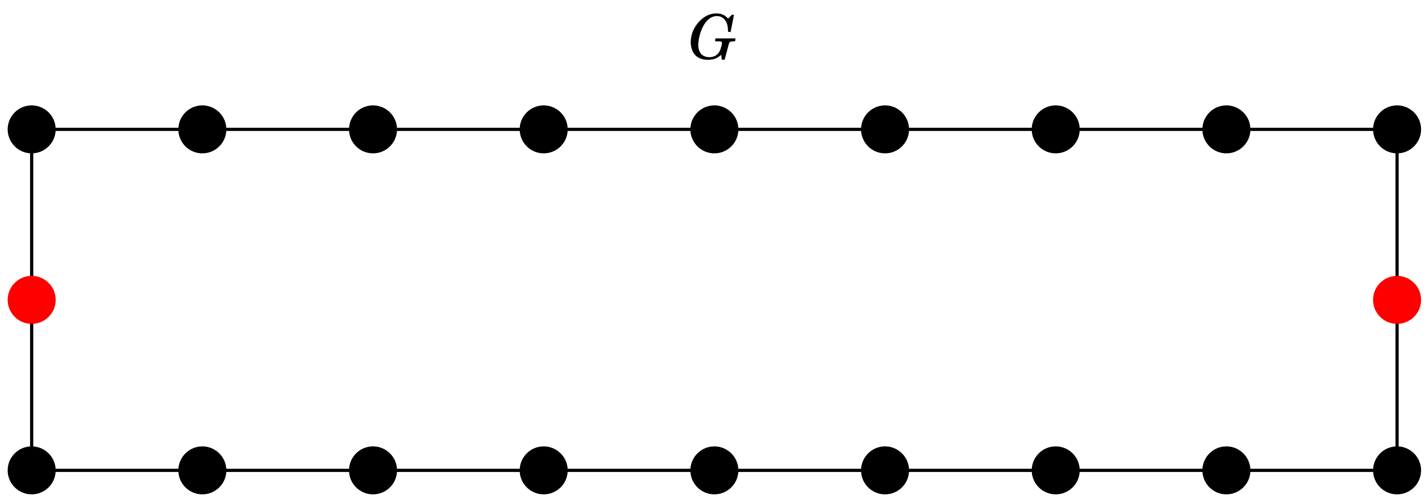
- **Problem**: $\mathbf{2}$-coloring paths & even cycles in the *non-signaling* model



- Suppose outcome $\mathbf{O}$ with locality $T \leq n/5$

Indistinguishability argument

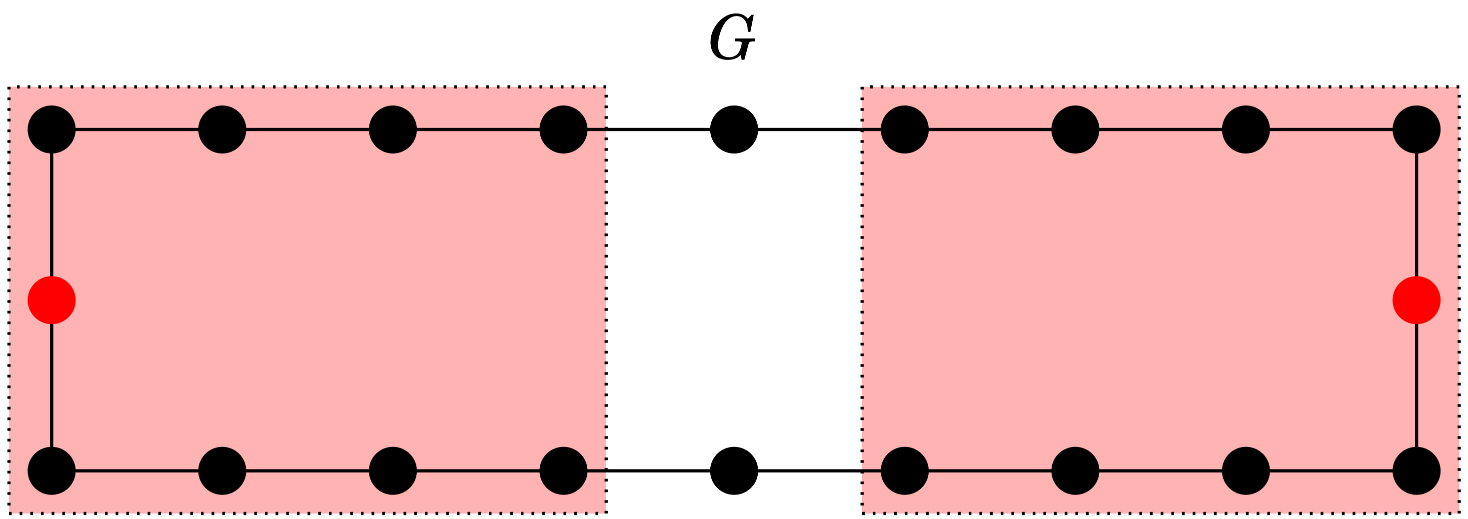
- **Problem**: 2-coloring paths & even cycles in the *non-signaling* model



- Suppose outcome $\mathbf{O}$ with locality $T \leq n/5$

Indistinguishability argument

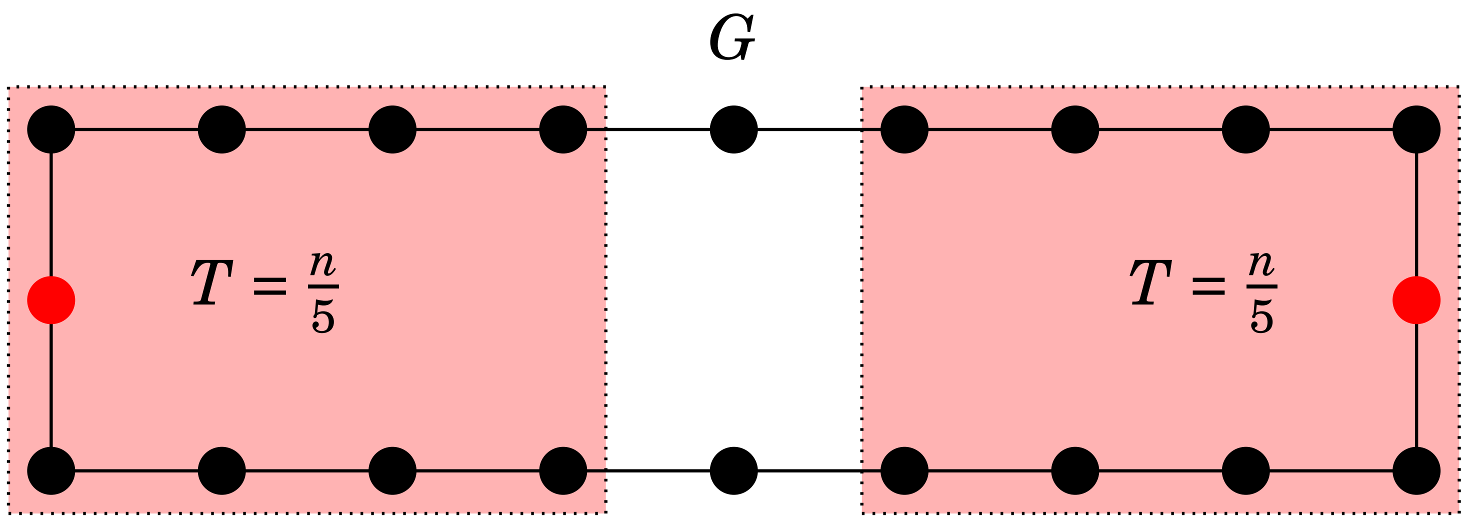
- **Problem**: $\mathbf{2}$-coloring paths & even cycles in the *non-signaling* model



- Suppose outcome $\mathbf{O}$ with locality $T \leq n/5$

Indistinguishability argument

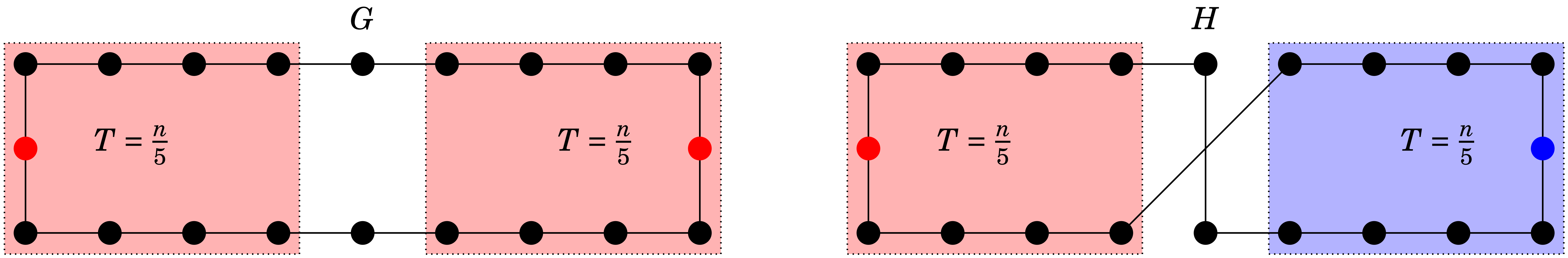
- **Problem**: 2-coloring paths & even cycles in the *non-signaling* model



- Suppose outcome $\mathbf{O}$ with locality $T \leq n/5$

Indistinguishability argument

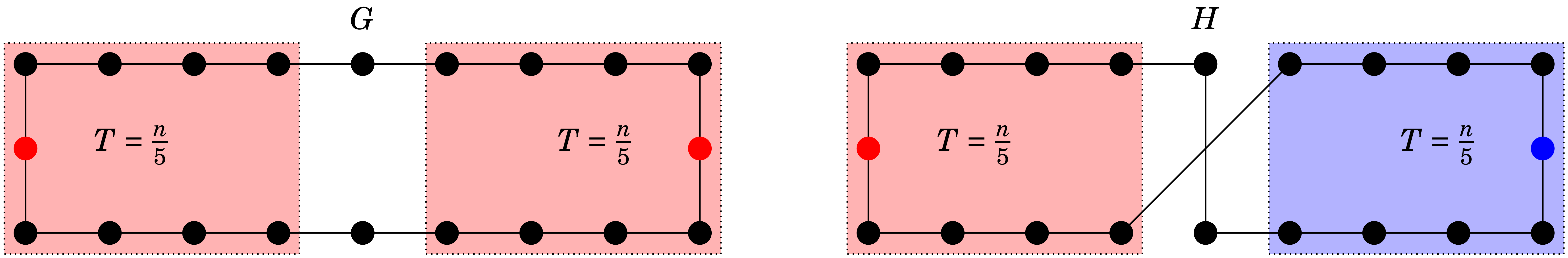
- **Problem**: 2-coloring paths & even cycles in the *non-signaling* model



- Suppose outcome $\mathbf{O}$ with locality $T \leq n/5$

Indistinguishability argument

- **Problem**: 2-coloring paths & even cycles in the *non-signaling* model

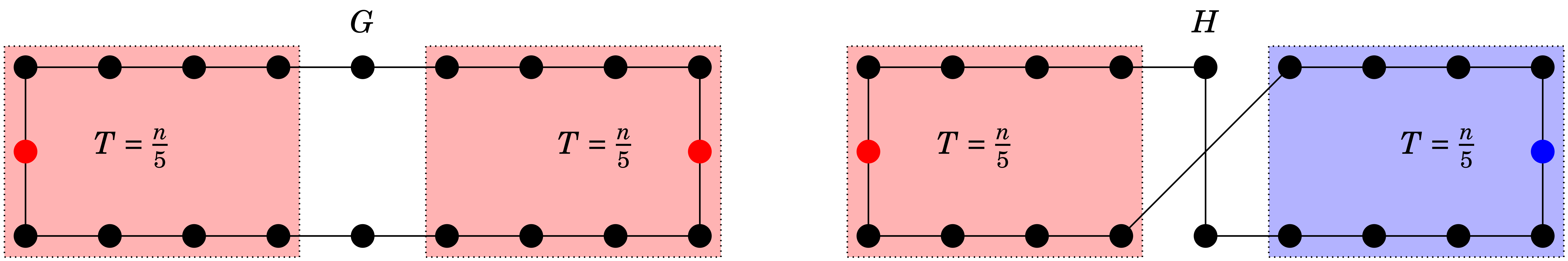


- Suppose outcome $\mathbf{O}$ with locality $T \leq n/5$

- Impossible to distinguish between G and H

Indistinguishability argument

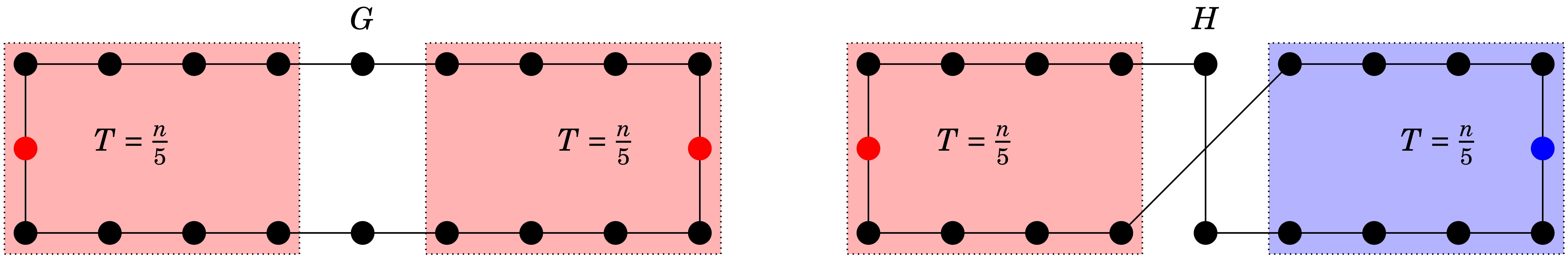
- **Problem**: 2-coloring paths & even cycles in the *non-signaling* model



- Suppose outcome $\mathbf{O}$ with locality $T \leq n/5$
- Impossible to distinguish between G and H
- Failure with prob. $\geq 1/2$

Indistinguishability argument

- **Problem**: 2-coloring paths & even cycles in the *non-signaling* model

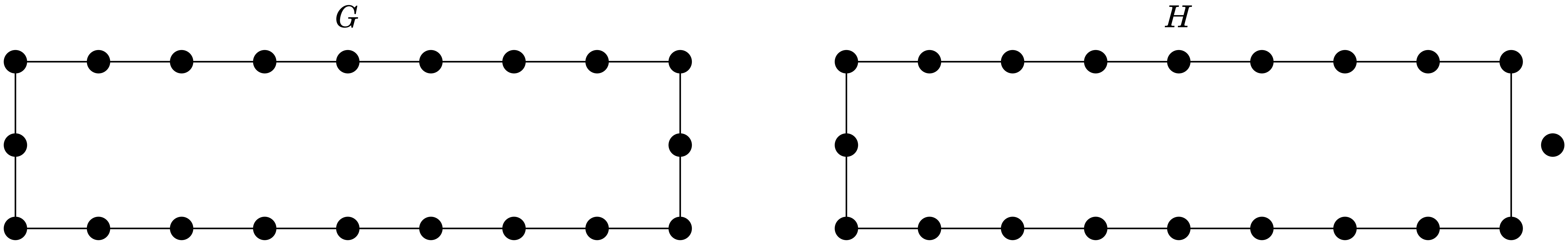


  - Suppose outcome $\mathbf{O}$ with locality $T \leq n/5$
  - Impossible to distinguish between G and H
  - Failure with prob. $\geq 1/2$

- **Global problem**: complexity $\Theta(n)$

Graph-existential indistinguishability

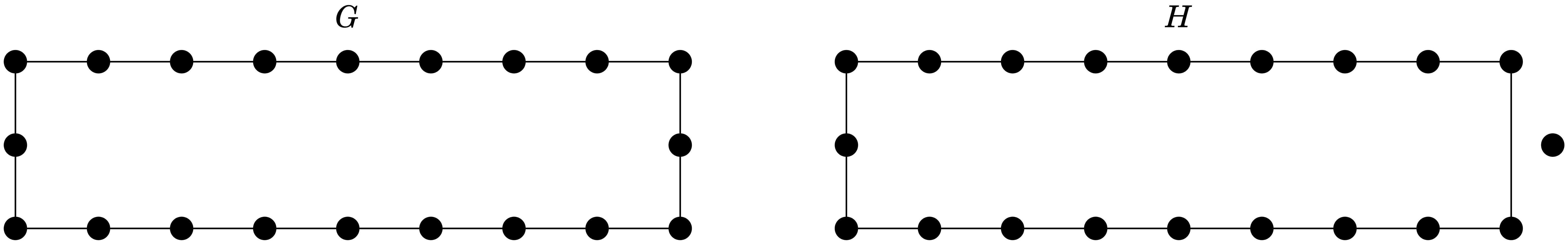
- **Problem**: 2-coloring paths & even cycles



  - Suppose outcome $\mathbf{O}$ with locality $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$

Graph-existential indistinguishability

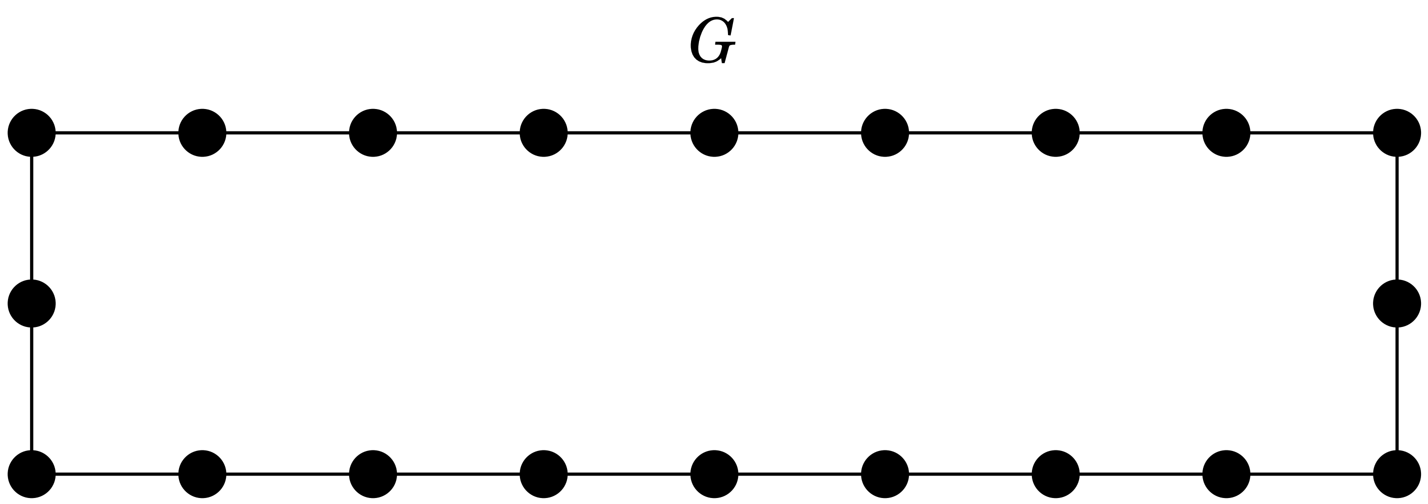
- **Problem**: 2-coloring paths & even cycles

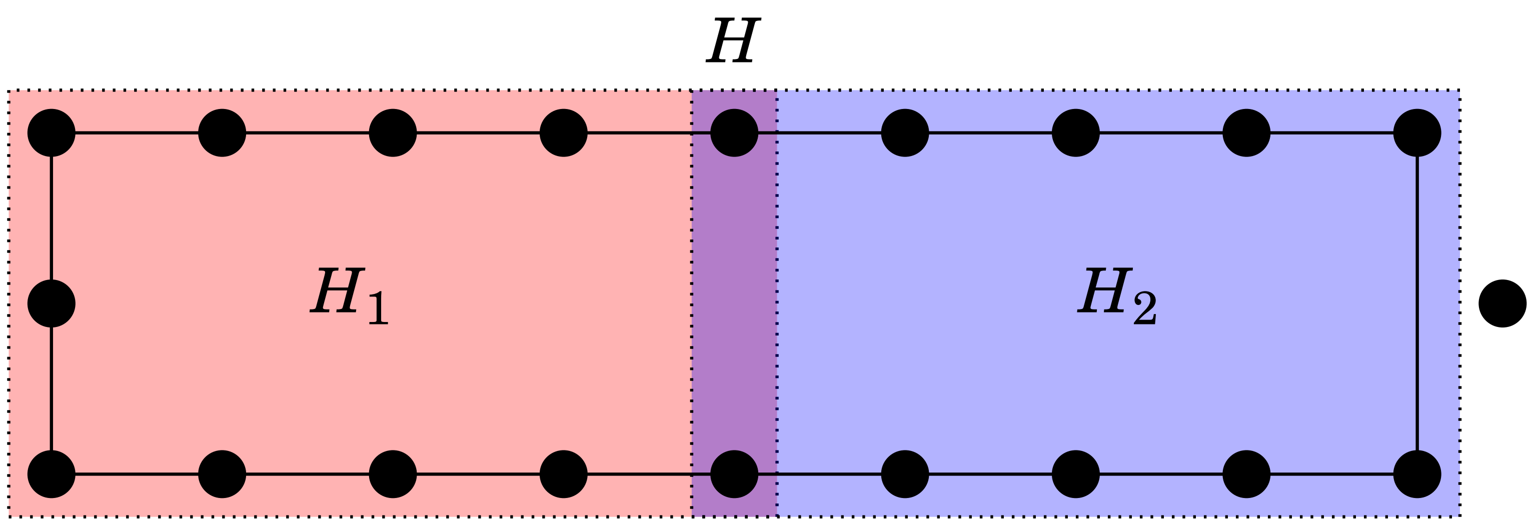


- Suppose outcome $\mathbf{O}$ with locality $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$
- $\mathbf{O}$ fails in H with probability 1

Graph-existential indistinguishability

- **Problem**: 2-coloring paths & even cycles

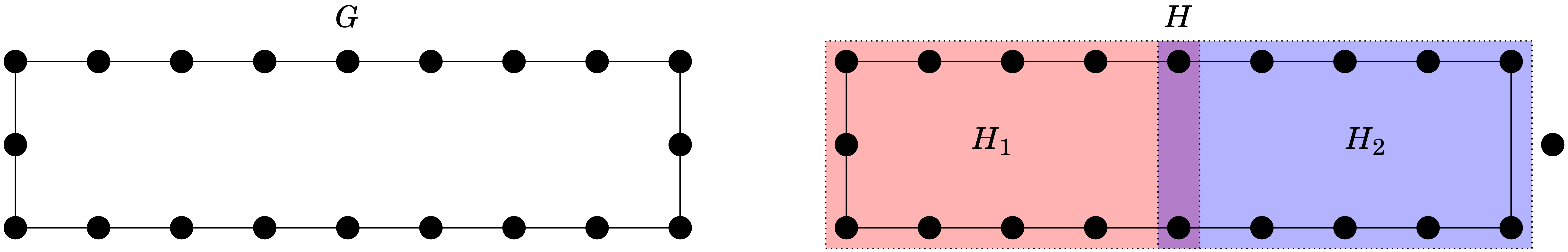




- Suppose outcome $\mathbf{O}$ with locality $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$
- $\mathbf{O}$ fails in H with probability 1

Graph-existential indistinguishability

- **Problem**: 2-coloring paths & even cycles

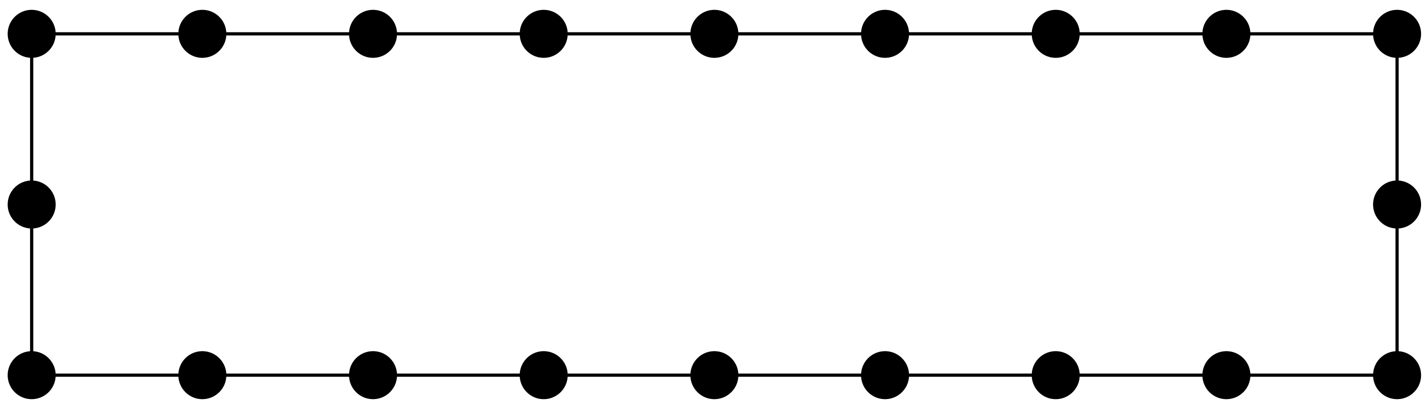


- Suppose outcome O with locality $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$
- O fails in H with probability 1
- O fails either in H_1 or in H_2 with probability $\geq 1/2$. Wlog, in H_1

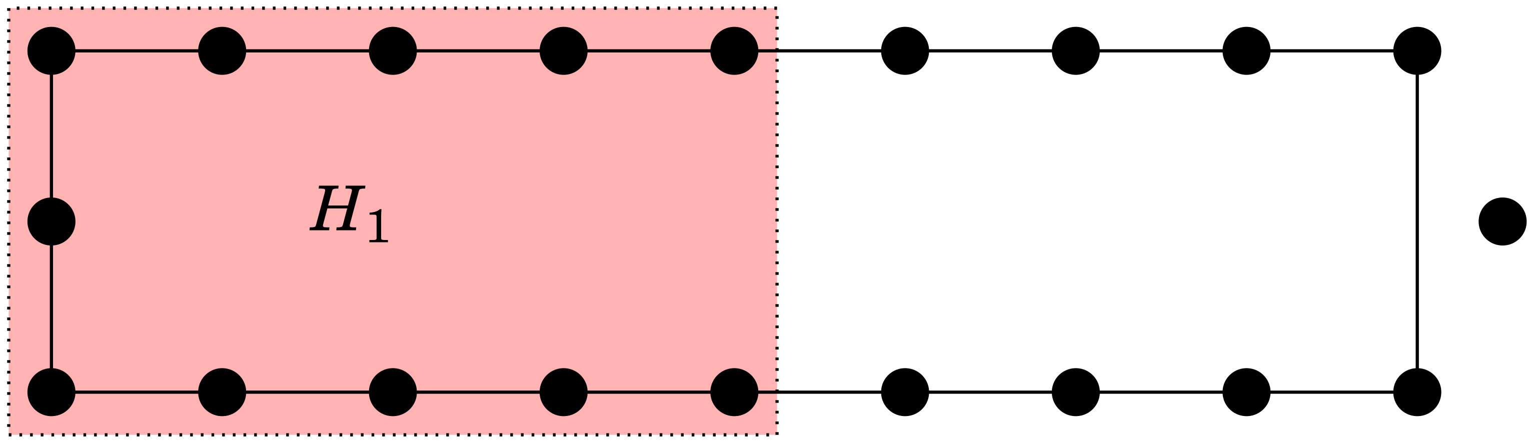
Graph-existential indistinguishability

- **Problem**: 2-coloring paths & even cycles

G



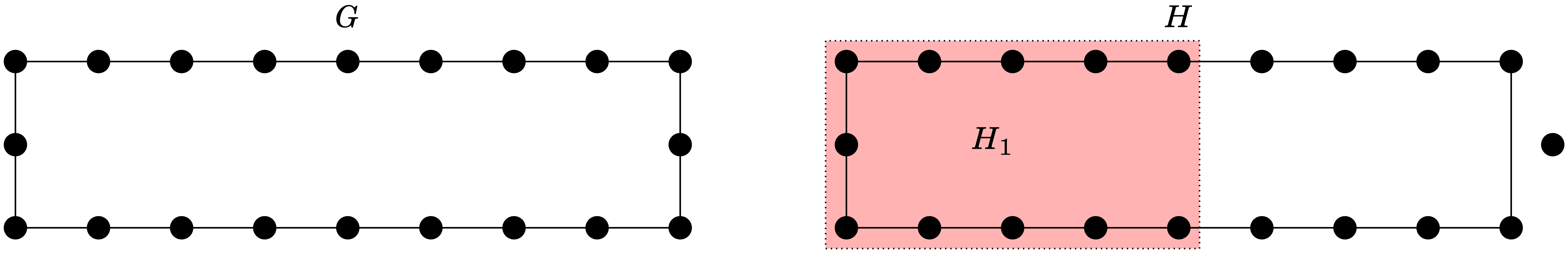
H



- Suppose outcome $\mathbf{O}$ with locality $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$
- $\mathbf{O}$ fails in H with probability 1
- $\mathbf{O}$ fails either in H_1 or in H_2 with probability $\geq 1/2$. Wlog, in H_1

Graph-existential indistinguishability

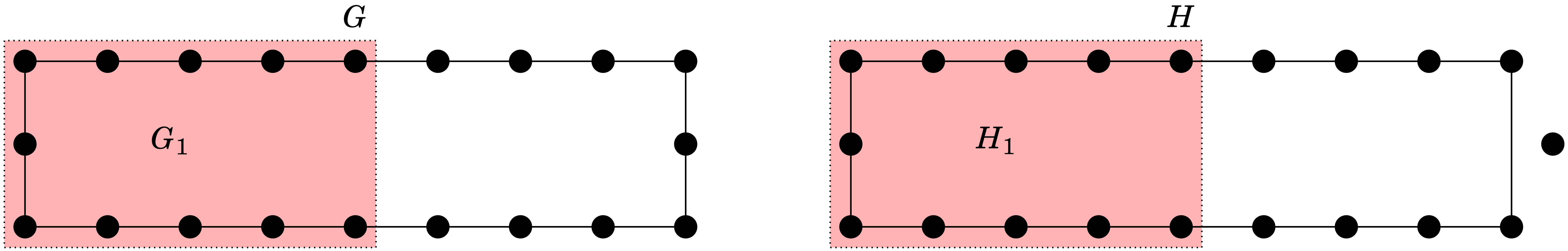
- **Problem**: 2-coloring paths & even cycles



- Suppose outcome $\mathbf{O}$ with locality $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$
- $\mathbf{O}$ fails in H with probability 1
- $\mathbf{O}$ fails either in H_1 or in H_2 with probability $\geq 1/2$. Wlog, in H_1
- Copy H_1 in G

Graph-existential indistinguishability

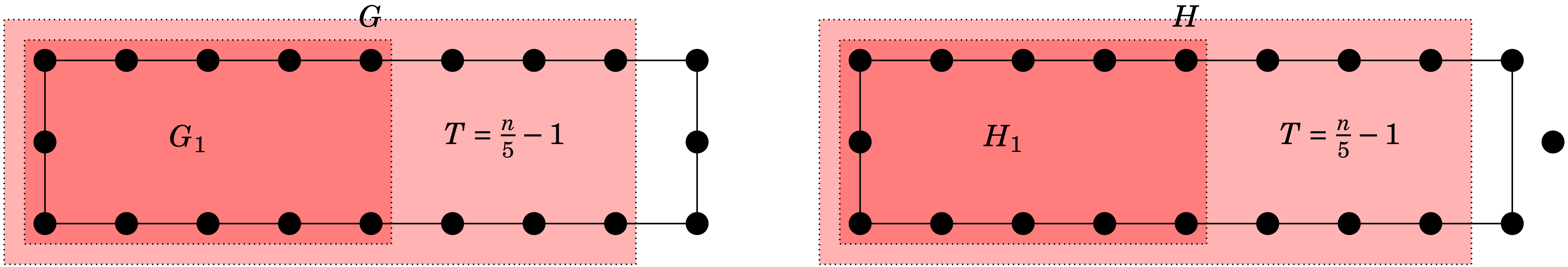
- **Problem**: 2-coloring paths & even cycles



- Suppose outcome O with locality $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$
- O fails in H with probability 1
- O fails either in H_1 or in H_2 with probability $\geq 1/2$. Wlog, in H_1
- Copy H_1 in G

Graph-existential indistinguishability

- **Problem**: 2-coloring paths & even cycles



- Suppose outcome $\mathbf{O}$ with locality $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$

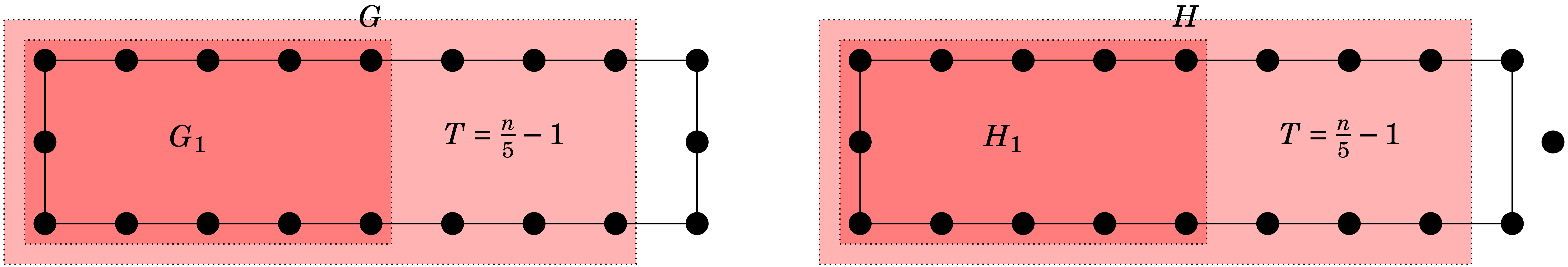
- $\mathbf{O}$ fails in H with probability 1

- $\mathbf{O}$ fails either in H_1 or in H_2 with probability $\geq 1/2$. Wlog, in H_1

- Copy H_1 in G

Graph-existential indistinguishability

- **Problem**: 2-coloring paths & even cycles

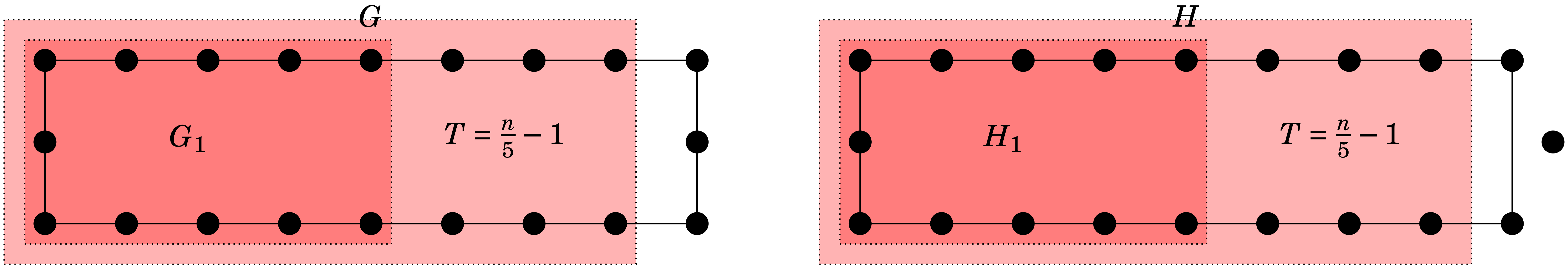


  - Suppose outcome $\mathbf{O}$ with locality $T \le n/5 - 1$ succeeds in G with probability $\ge 1 - 1/n$
  - $\mathbf{O}$ fails in H with probability 1
  - $\mathbf{O}$ fails either in H_1 or in H_2 with probability $\ge 1/2$. Wlog, in H_1
  - Copy H_1 in G

- **Non-signaling property**: Distributions of nodes in H_1/G_1 must be the same

Graph-existential indistinguishability

- **Problem**: 2-coloring paths & even cycles



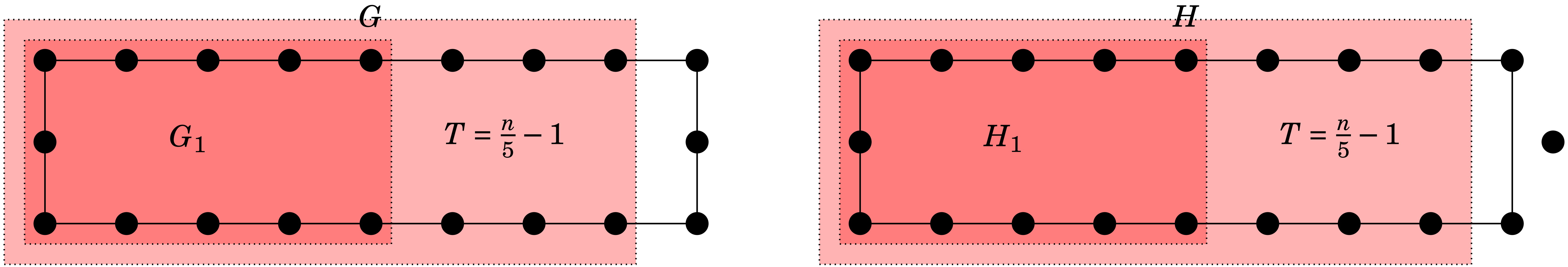
  - Suppose outcome $\mathbf{O}$ with locality $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$
  - $\mathbf{O}$ fails in H with probability 1
  - $\mathbf{O}$ fails either in H_1 or in H_2 with probability $\geq 1/2$. Wlog, in H_1
  - Copy H_1 in G

$\mathbf{O}$ fails in G_1 with prob. $\geq 1/2$

- **Non-signaling property**: Distributions of nodes in H_1/G_1 must be the same

Graph-existential indistinguishability

- **Problem**: 2-coloring paths & even cycles



  - Suppose outcome $\mathbf{O}$ with locality $T \leq n/5 - 1$ succeeds in G with probability $\geq 1 - 1/n$
  - $\mathbf{O}$ fails in H with probability 1
  - $\mathbf{O}$ fails either in H_1 or in H_2 with probability $\geq 1/2$. Wlog, in H_1
  - Copy H_1 in G

$\mathbf{O}$ fails in G_1 with prob. $\geq 1/2$

- **Non-signaling property**: Distributions of nodes in H_1/G_1 must be the same
- **Boosting failure probability**: possibility to boos failure prob. to *any* constant (non-trivial)

Table of content

Complexity of approximate graph coloring

Problem: c-coloring χ-chromatic graphs [STOC '24]

		upper bound		lower bound		
χ	c	old	new	old	new	ref
2	2	$O(n)$	$O(n)$	$\Omega(n)$	$\Omega(n)$	trivial
2	3					
2	4					
2	5					
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
χ	χ					
χ	$c > \chi$					

Complexity of approximate graph coloring

Problem: c-coloring χ-chromatic graphs [STOC '24]

		upper bound		lower bound		
χ	c	old	new	old	new	ref
2	2	$O(n)$	$O(n)$	$\Omega(n)$	$\Omega(n)$	trivial
2	3	$O(n)$		$\Omega(\sqrt{n})$		[Brandt et al. '20]
2	4					
2	5					
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
χ	χ					
χ	$c > \chi$					

Complexity of approximate graph coloring

Problem: c-coloring χ-chromatic graphs [STOC '24]

		upper bound		lower bound		
χ	c	old	new	old	new	ref
2	2	$O(n)$	$O(n)$	$\Omega(n)$	$\Omega(n)$	trivial
2	3	$O(n)$		$\Omega(\sqrt{n})$		[Brandt et al. '20]
2	4	$O(n)$		$\Omega(\log n)$		[Linial '92]
2	5	$O(n)$		$\Omega(\log n)$		[Linial '92]
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
χ	χ					
χ	$c > \chi$	$O(n)$		$\Omega(\log n)$		[Linial '92]

Complexity of approximate graph coloring

Problem: c-coloring χ-chromatic graphs [STOC '24]

		upper bound		lower bound		
χ	c	old	new	old	new	ref
2	2	$O(n)$	$O(n)$	$\Omega(n)$	$\Omega(n)$	trivial
2	3	$O(n)$		$\Omega(\sqrt{n})$		[Brandt et al. '20]
2	4	$O(n)$		$\Omega(\log n)$		[Linial '92]
2	5	$O(n)$		$\Omega(\log n)$		[Linial '92]
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
χ	χ	$O(n)$	$O(n)$	$\Omega(n)$	$\Omega(n)$	trivial
χ	$c > \chi$	$O(n)$		$\Omega(\log n)$		[Linial '92]

Complexity of approximate graph coloring

Problem: c-coloring χ-chromatic graphs [STOC '24]

		upper bound		lower bound		
χ	c	old	new	old	new	ref
2	2	$O(n)$	$O(n)$	$\Omega(n)$	$\Omega(n)$	trivial
2	3	$O(n)$	$\tilde{O}(\sqrt{n})$	$\Omega(\sqrt{n})$	$\Omega(\sqrt{n})$	[Brandt et al. '20]
2	4	$O(n)$		$\Omega(\log n)$		[Linial '92]
2	5	$O(n)$		$\Omega(\log n)$		[Linial '92]
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
χ	χ	$O(n)$	$O(n)$	$\Omega(n)$	$\Omega(n)$	trivial
χ	$c > \chi$	$O(n)$		$\Omega(\log n)$		[Linial '92]

Complexity of approximate graph coloring

Problem: c-coloring χ-chromatic graphs [STOC '24]

		upper bound		lower bound		
χ	c	old	new	old	new	ref
2	2	$O(n)$	$O(n)$	$\Omega(n)$	$\Omega(n)$	trivial
2	3	$O(n)$	$\tilde{O}(\sqrt{n})$	$\Omega(\sqrt{n})$	$\Omega(\sqrt{n})$	[Brandt et al. '20]
2	4	$O(n)$	$\tilde{O}(n^{\frac{1}{3}})$	$\Omega(\log n)$	$\Omega(n^{\frac{1}{3}})$	[Linial '92]
2	5	$O(n)$		$\Omega(\log n)$		[Linial '92]
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
χ	χ	$O(n)$	$O(n)$	$\Omega(n)$	$\Omega(n)$	trivial
χ	$c > \chi$	$O(n)$		$\Omega(\log n)$		[Linial '92]

Complexity of approximate graph coloring

Problem: c-coloring χ-chromatic graphs [STOC '24]

		upper bound		lower bound		
χ	c	old	new	old	new	ref
2	2	$O(n)$	$O(n)$	$\Omega(n)$	$\Omega(n)$	trivial
2	3	$O(n)$	$\tilde{O}(\sqrt{n})$	$\Omega(\sqrt{n})$	$\Omega(\sqrt{n})$	[Brandt et al. '20]
2	4	$O(n)$	$\tilde{O}(n^{\frac{1}{3}})$	$\Omega(\log n)$	$\Omega(n^{\frac{1}{3}})$	[Linial '92]
2	5	$O(n)$	$\tilde{O}(n^{\frac{1}{4}})$	$\Omega(\log n)$	$\Omega(n^{\frac{1}{4}})$	[Linial '92]
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
χ	χ	$O(n)$	$O(n)$	$\Omega(n)$	$\Omega(n)$	trivial
χ	$c > \chi$	$O(n)$		$\Omega(\log n)$		[Linial '92]

Complexity of approximate graph coloring

Problem: c-coloring χ-chromatic graphs [STOC '24]

		upper bound		lower bound		
χ	c	old	new	old	new	ref
2	2	$O(n)$	$O(n)$	$\Omega(n)$	$\Omega(n)$	trivial
2	3	$O(n)$	$\tilde{O}(\sqrt{n})$	$\Omega(\sqrt{n})$	$\Omega(\sqrt{n})$	[Brandt et al. '20]
2	4	$O(n)$	$\tilde{O}(n^{\frac{1}{3}})$	$\Omega(\log n)$	$\Omega(n^{\frac{1}{3}})$	[Linial '92]
2	5	$O(n)$	$\tilde{O}(n^{\frac{1}{4}})$	$\Omega(\log n)$	$\Omega(n^{\frac{1}{4}})$	[Linial '92]
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
χ	χ	$O(n)$	$O(n)$	$\Omega(n)$	$\Omega(n)$	trivial
χ	$c > \chi$	$O(n)$	$\tilde{O}(n^{1/\lfloor\frac{c-1}{\chi-1}\rfloor})$	$\Omega(\log n)$	$\Omega(n^{1/\lfloor\frac{c-1}{\chi-1}\rfloor})$	[Linial '92]

- $\alpha = \lfloor\frac{c-1}{\chi-1}\rfloor$ approximation ratio

Some results: non-signaling model

Graph-existential lower bound arguments based on indistinguishability

- **Graph coloring**: c-coloring χ-chromatic graphs has complexity $\tilde{\Theta}(n^{1/\lfloor \frac{c-1}{\chi-1} \rfloor})$ [STOC '24]
  - makes use of a "cheating graph" from [Bogdanov, '13]
  - upper bound in deterministic LOCAL, lower bound in non-signaling LOCAL
  - no quantum advantage

Some results: non-signaling model

Graph-existential lower bound arguments based on indistinguishability

- **Graph coloring**: c-coloring χ-chromatic graphs has complexity $\tilde{\Theta}(n^{1/\lfloor \frac{c-1}{\chi-1} \rfloor})$ [STOC '24]

  - makes use of a "cheating graph" from [Bogdanov, '13]
  - upper bound in deterministic LOCAL, lower bound in non-signaling LOCAL
  - no quantum advantage

- **Tree coloring**: c-coloring trees has complexity $\Omega(\log_c n)$ [STOC '24]

  - revisitation of [Linial, FOCS '87]'s lower bound
  - no quantum advantage if high degree

Some results: non-signaling model

Graph-existential lower bound arguments based on indistinguishability

- **Graph coloring**: c-coloring χ-chromatic graphs has complexity $\tilde{\Theta}(n^{1/\lfloor \frac{c-1}{\chi-1} \rfloor})$ [STOC '24]
  - makes use of a "cheating graph" from [Bogdanov, '13]
  - upper bound in deterministic LOCAL, lower bound in non-signaling LOCAL
  - no quantum advantage

- **Tree coloring**: c-coloring trees has complexity $\Omega(\log_c n)$ [STOC '24]
  - revisitation of [Linial, FOCS '87]'s lower bound
  - no quantum advantage if high degree

- **Grid coloring**: 3-coloring grids of size $n_1 \times n_2$ has complexity $\Omega(\min\{n_1, n_2\})$ [STOC '24]
  - makes use of odd quadrangulations of Klein-bottles [MST, Combinatorica '13]
  - no quantum advantage

Some results: non-signaling model

Graph-existential lower bound arguments based on indistinguishability

- **Graph coloring**: c-coloring χ-chromatic graphs has complexity $\tilde{\Theta}(n^{1/\lfloor \frac{c-1}{\chi-1} \rfloor})$ [STOC '24]

  - makes use of a "cheating graph" from [Bogdanov, '13]
  - upper bound in deterministic LOCAL, lower bound in non-signaling LOCAL
  - no quantum advantage

- **Tree coloring**: c-coloring trees has complexity $\Omega(\log_c n)$ [STOC '24]

  - revisitation of [Linial, FOCS '87]'s lower bound
  - no quantum advantage if high degree

- **Grid coloring**: 3-coloring grids of size $n_1 \times n_2$ has complexity $\Omega(\min\{n_1, n_2\})$ [STOC '24]

  - makes use of odd quadrangulations of Klein-bottles [MST, Combinatorica '13]
  - no quantum advantage

What about other known lower bounds? E.g., 3-coloring cycles has complexity $\Theta(\log^\star n)$ [Linial, FOCS '87]

Some results: bounded-dependence model

- Can we similarly **rule out quantum advantage for 3-coloring cycles**? Classical complexity $T = \Theta(\log^\star n)$

Some results: bounded-dependence model

- Can we similarly **rule out quantum advantage for 3-coloring cycles**? Classical complexity $T = \Theta(\log^\star n)$
  - no! *There is a finitely-dependent distribution that 3-colors paths and cycles* ($T = O(1)$)
  - [Holroyd and Liggett, Forum of Mathematics, Pi '14]
  - [Holroyd, Hutchcroft, and Levy, Electronic Communications in Probability '18]

Some results: bounded-dependence model

- Can we similarly **rule out quantum advantage for 3-coloring cycles**? Classical complexity $T = \Theta(\log^\star n)$
  - no! *There is a finitely-dependent distribution that $\mathbf{3}$-colors paths and cycles* ($\boldsymbol{T} = \boldsymbol{O}(\mathbf{1})$)
  - [Holroyd and Liggett, Forum of Mathematics, Pi ’14]
  - [Holroyd, Hutchcroft, and Levy, Electronic Communications in Probability ’18]

- Is there **any quantum-LOCAL algorithm that 3-colors paths and cycles with locality** $T = o(\log^\star n)$?
  - major open question

Some results: bounded-dependence model

- Can we similarly **rule out quantum advantage for 3-coloring cycles**? Classical complexity $T = \Theta(\log^\star n)$
  - no! *There is a finitely-dependent distribution that* $\mathbf{3}$*-colors paths and cycles* ($\boldsymbol{T} = \boldsymbol{O}(\mathbf{1})$)
  - [Holroyd and Liggett, Forum of Mathematics, Pi '14]
  - [Holroyd, Hutchcroft, and Levy, Electronic Communications in Probability '18]

- Is there **any quantum-LOCAL algorithm that 3-colors paths and cycles with locality** $T = o(\log^\star n)$?
  - major open question

- Is there any hope to **rule out quantum advantage for LCLs of complexity $\Theta(\log^\star n)$ in classical LOCAL**?
  * using stronger models

Some results: bounded-dependence model

- Can we similarly **rule out quantum advantage for 3-coloring cycles**? Classical complexity $T = \Theta(\log^\star n)$
  - no! *There is a finitely-dependent distribution that* 3*-colors paths and cycles* ($T = O(1)$)
  - [Holroyd and Liggett, Forum of Mathematics, Pi '14]
  - [Holroyd, Hutchcroft, and Levy, Electronic Communications in Probability '18]

- Is there **any quantum-LOCAL algorithm that 3-colors paths and cycles with locality** $T = o(\log^\star n)$?
  - major open question

- Is there any hope to **rule out quantum advantage for LCLs of complexity $\Theta(\log^\star n)$ in classical LOCAL**?
  * using stronger models
  - no!
  - For any LCL Π on bounded degree graphs, *there is a finitely-dependent distribution* ($T = O(1)$) *solving* Π
  - [STOC '25a]

Bounding general quantum advantage

- **Theorem** [SODA '26]: given any LCL Π

Bounding general quantum advantage

- **Theorem** [SODA '26]: given any LCL Π
  - T-dependent distribution $\implies O(\sqrt{Tn}\,\mathrm{poly}\log(n))$-time randomized LOCAL algorithm

Bounding general quantum advantage

- **Theorem** [SODA '26]: given any LCL Π
  - T-dependent distribution $\implies O(\sqrt{Tn}\ \text{poly}\log(n))$-time randomized LOCAL algorithm

- **Observations**:
  - $O(1)$-dependent distribution $\implies O(\sqrt{n}\ \text{poly}\log(n))$-time randomized LOCAL algorithm

Bounding general quantum advantage

- **Theorem** [SODA '26]: given any LCL Π
  - T-dependent distribution $\implies O(\sqrt{Tn}\text{ poly}\log(n))$-time randomized LOCAL algorithm

- **Observations**:
  - $O(1)$-dependent distribution $\implies O(\sqrt{n}\text{ poly}\log(n))$-time randomized LOCAL algorithm
  - $\Omega(n)$-time randomized LOCAL algorithm $\implies \Omega(n/\text{ poly}\log(n))$-dependent distribution

Bounding general quantum advantage

- **Theorem** [SODA '26]: given any LCL Π
  - T-dependent distribution $\implies O(\sqrt{Tn} \text{ poly} \log(n))$-time randomized LOCAL algorithm

- **Observations**:
  - $O(1)$-dependent distribution $\implies O(\sqrt{n} \text{ poly} \log(n))$-time randomized LOCAL algorithm
  - $\Omega(n)$-time randomized LOCAL algorithm $\implies \Omega(n/ \text{ poly} \log(n))$-dependent distribution
  - $\Omega(T)$-time randomized LOCAL algorithm $\implies \Omega(T^2/(n \text{ poly} \log(n)))$-dependent distribution

$$T \gg \sqrt{n} \text{ poly} \log(n)$$

Bounding general quantum advantage

- **Theorem** [SODA '26]: given any LCL Π
  - T-dependent distribution $\implies O(\sqrt{Tn}\text{ poly}\log(n))$-time randomized LOCAL algorithm

- **Observations**:
  - $O(1)$-dependent distribution $\implies O(\sqrt{n}\text{ poly}\log(n))$-time randomized LOCAL algorithm
  - $\Omega(n)$-time randomized LOCAL algorithm $\implies \Omega(n/\text{ poly}\log(n))$-dependent distribution
  - $\Omega(T)$-time randomized LOCAL algorithm $\implies \Omega(T^2/(n\text{ poly}\log(n)))$-dependent distribution

$$T \gg \sqrt{n}\text{ poly}\log(n)$$

  - We can derandomize at poly $\log(n)$ cost

Bounding general quantum advantage

- **Theorem** [SODA '26]: given any LCL Π

  - T-dependent distribution $\implies O(\sqrt{Tn}\text{ poly}\log(n))$-time randomized LOCAL algorithm

- **Observations**:

  - $O(1)$-dependent distribution $\implies O(\sqrt{n}\text{ poly}\log(n))$-time randomized LOCAL algorithm
  - $\Omega(n)$-time randomized LOCAL algorithm $\implies \Omega(n/\text{ poly}\log(n))$-dependent distribution
  - $\Omega(T)$-time randomized LOCAL algorithm $\implies \Omega(T^2/(n\text{ poly}\log(n)))$-dependent distribution

$$T \gg \sqrt{n}\text{ poly}\log(n)$$

  - We can derandomize at poly $\log(n)$ cost
  - T-dependent distribution $\implies O(\sqrt{Tn}\text{ poly}\log(n))$-time deterministic LOCAL algorithm

Relations among models

- $X \to Y$ means that locality T in X becomes locality $O(T)$ in Y

T-dependent distribution $\implies O(\sqrt{Tn}\text{ poly}\log(n))$-time deterministic LOCAL algorithm

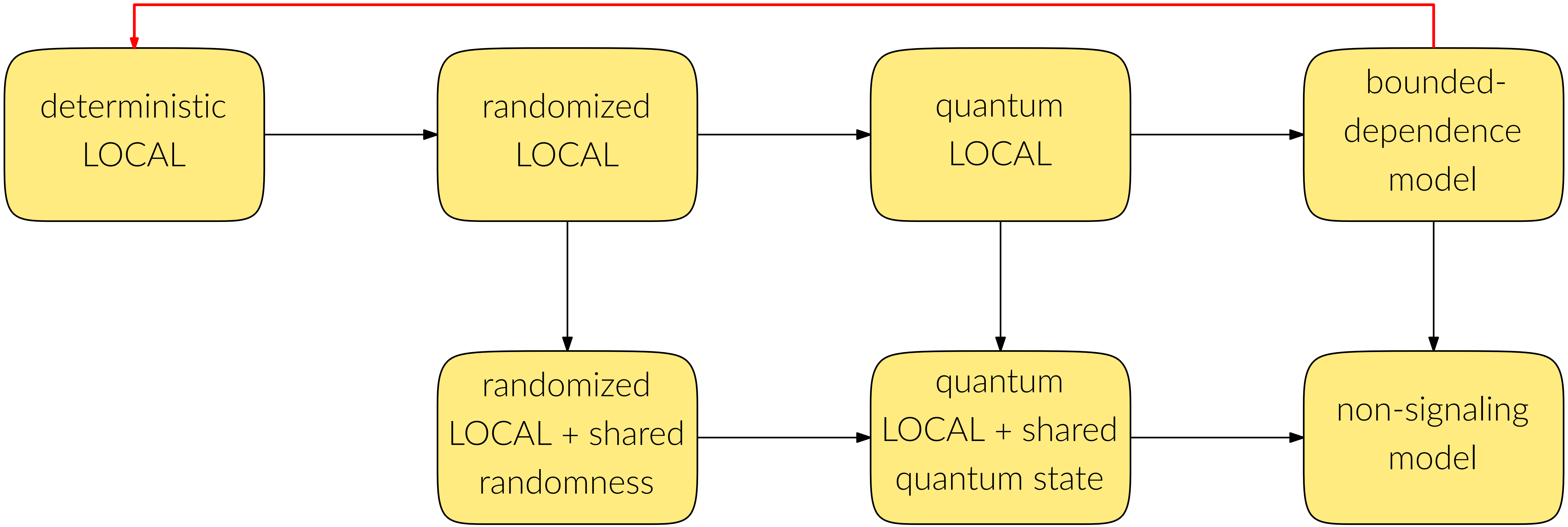


Table of content

Distributed quantum computing

- **Quantum-CONGEST**: computing the diameter of a network [Le Gall, Magniez, PODC '18]
  - **Classically**: $\Theta(n)$
  - **Quantum**: $\tilde{\Theta}(\sqrt{n})$

Distributed quantum computing

- **Quantum-CONGEST**: computing the diameter of a network [Le Gall, Magniez, PODC '18]
  - **Classically**: $\Theta(n)$
  - **Quantum**: $\tilde{\Theta}(\sqrt{n})$
- **Quantum-LOCAL**: there is a problem with quantum advantage [Le Gall, Nishimura, Rosmanis, STACS '19]
  - **Classically**: $\Theta(n)$
  - **Quantum**: 2 rounds

Distributed quantum computing

- **Quantum-CONGEST**: computing the diameter of a network [Le Gall, Magniez, PODC '18]
  - **Classically**: $\Theta(n)$
  - **Quantum**: $\tilde{\Theta}(\sqrt{n})$

- **Quantum-LOCAL**: there is a problem with quantum advantage [Le Gall, Nishimura, Rosmanis, STACS '19]
  - **Classically**: $\Theta(n)$
  - **Quantum**: 2 rounds
  - **Weakness 1**: useless computational task

Distributed quantum computing

- **Quantum-CONGEST**: computing the diameter of a network [Le Gall, Magniez, PODC '18]
  - **Classically**: $\Theta(n)$
  - **Quantum**: $\tilde{\Theta}(\sqrt{n})$

- **Quantum-LOCAL**: there is a problem with quantum advantage [Le Gall, Nishimura, Rosmanis, STACS '19]
  - **Classically**: $\Theta(n)$
  - **Quantum**: 2 rounds
  - **Weakness** 1: useless computational task
  - **Weakness** 2: not locally checkable

Distributed quantum computing

- **Quantum-CONGEST**: computing the diameter of a network [Le Gall, Magniez, PODC '18]
  - **Classically**: $\Theta(n)$
  - **Quantum**: $\tilde{\Theta}(\sqrt{n})$

- **Quantum-LOCAL**: there is a problem with quantum advantage [Le Gall, Nishimura, Rosmanis, STACS '19]
  - **Classically**: $\Theta(n)$
  - **Quantum**: 2 rounds
  - **Weakness** 1: useless computational task
  - **Weakness** 2: not locally checkable
  - **Problem**: *sampling from the output distribution of a quantum circuit that measures a graph state in a random basis*

Distributed quantum computing

- **Quantum-CONGEST**: computing the diameter of a network [Le Gall, Magniez, PODC '18]
  - **Classically**: $\Theta(n)$
  - **Quantum**: $\tilde{\Theta}(\sqrt{n})$

- **Quantum-LOCAL**: there is a problem with quantum advantage [Le Gall, Nishimura, Rosmanis, STACS '19]
  - **Classically**: $\Theta(n)$
  - **Quantum**: 2 rounds
  - **Weakness 1**: useless computational task
  - **Weakness 2**: not locally checkable
  - **Problem**: *sampling from the output distribution of a quantum circuit that measures a graph state in a random basis*

- **Quantum-LOCAL**: can we do something better? [STOC '25b, SODA '26]

Actual quantum advantage for LCLs

- Iterated GHZ (locally checkable)
  - **Classical**: $\Omega(\Delta)$ by round elimination [STOC '25b]
  - **Quantum**: 1 round

Actual quantum advantage for LCLs

- Iterated GHZ (locally checkable)
  - **Classical**: $\Omega(\Delta)$ by round elimination [STOC '25b]
  - **Quantum**: 1 round
- In [SODA '26] we lift the problem so that we construct an LCL s.t.

Actual quantum advantage for LCLs

- Iterated GHZ (locally checkable)
  - **Classical**: $\Omega(\Delta)$ by round elimination [STOC '25b]
  - **Quantum**: 1 round
- In [SODA '26] we lift the problem so that we construct an LCL s.t.
  - **Classical**: $\Omega(\log n \frac{\log\log n}{\log\log\log n})$

Actual quantum advantage for LCLs

- Iterated GHZ (locally checkable)
  - **Classical**: $\Omega(\Delta)$ by round elimination [STOC '25b]
  - **Quantum**: 1 round
- In [SODA '26] we lift the problem so that we construct an LCL s.t.
  - **Classical**: $\Omega(\log n \, \frac{\log\log n}{\log\log\log n})$
  - **Quantum**: $O(\log n)$

Actual quantum advantage for LCLs

- Iterated GHZ (locally checkable)
  - **Classical**: $\Omega(\Delta)$ by round elimination [STOC '25b]
  - **Quantum**: 1 round
- In [SODA '26] we lift the problem so that we construct an LCL s.t.
  - **Classical**: $\Omega(\log n \frac{\log\log n}{\log\log\log n})$
  - **Quantum**: $O(\log n)$
- **Found** an LCL problem that admits quantum advantage

Actual quantum advantage for LCLs

- Iterated GHZ (locally checkable)
  - **Classical**: $\Omega(\Delta)$ by round elimination [STOC '25b]
  - **Quantum**: 1 round
- In [SODA '26] we lift the problem so that we construct an LCL s.t.
  - **Classical**: $\Omega(\log n \, \frac{\log\log n}{\log\log\log n})$
  - **Quantum**: $O(\log n)$
- **Found** an LCL problem that admits quantum advantage
  - **useless** problem
  - what about problems that interest the community?

Actual quantum advantage for LCLs

- Iterated GHZ (locally checkable)
  - **Classical**: $\Omega(\Delta)$ by round elimination [STOC '25b]
  - **Quantum**: 1 round
- In [SODA '26] we lift the problem so that we construct an LCL s.t.
  - **Classical**: $\Omega(\log n \, \frac{\log \log n}{\log \log \log n})$
  - **Quantum**: $O(\log n)$
- **Found** an LCL problem that admits quantum advantage
  - **useless** problem
  - what about problems that interest the community?
- At the moment, we have **no single example**
  - major open question

Actual quantum advantage for LCLs

THANKS! Questions?

- Iterated GHZ (locally checkable)
  - **Classical**: $\Omega(\Delta)$ by round elimination [STOC '25b]
  - **Quantum**: 1 round
- In [SODA '26] we lift the problem so that we construct an LCL s.t.
  - **Classical**: $\Omega(\log n \frac{\log\log n}{\log\log\log n})$
  - **Quantum**: $O(\log n)$
- **Found** an LCL problem that admits quantum advantage
  - **useless** problem
  - what about problems that interest the community?
- At the moment, we have **no single example**
  - major open question

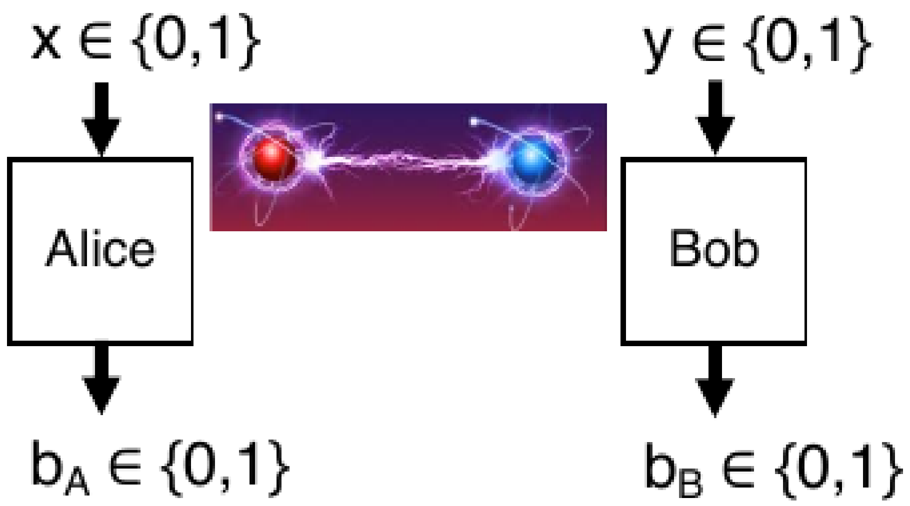
Non-signaling & quantum games

Alice

Bob

- Both Alice and Bob receive an input bit x and y in $\{0,1\}$
- They must output a bit each a and b according to some rule

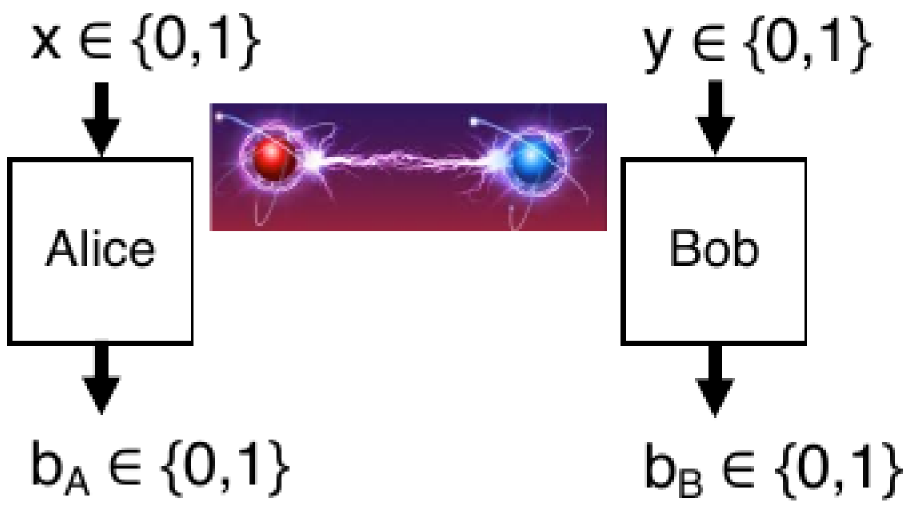
CHSH Game [Clauser, Horne, Shimony, Holt 1969]



winning condition: $b_A \oplus b_B = xy$

CHSH game

CHSH Game [Clauser, Horne, Shimony, Holt 1969]

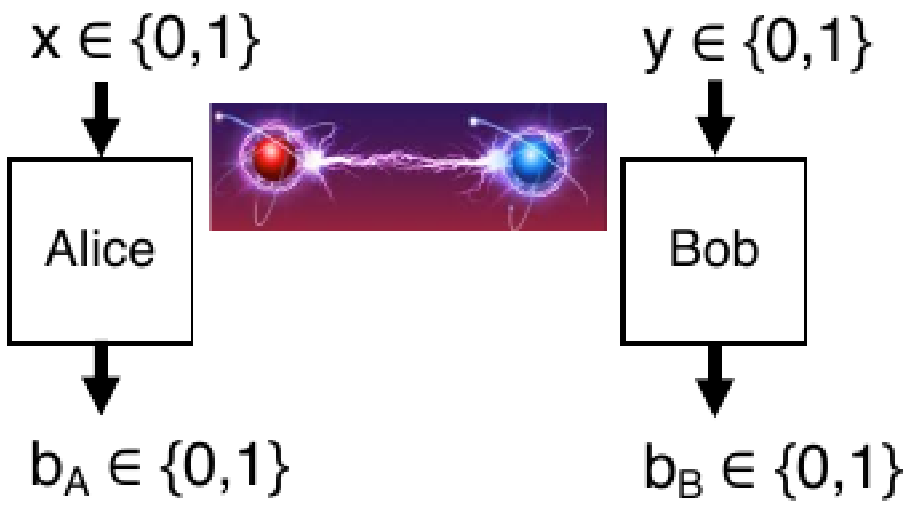


winning condition: $b_A \oplus b_B = xy$

- The **XOR** of the **outputs** is the **AND** of the **inputs**, **without communication**

CHSH game

CHSH Game [Clauser, Horne, Shimony, Holt 1969]



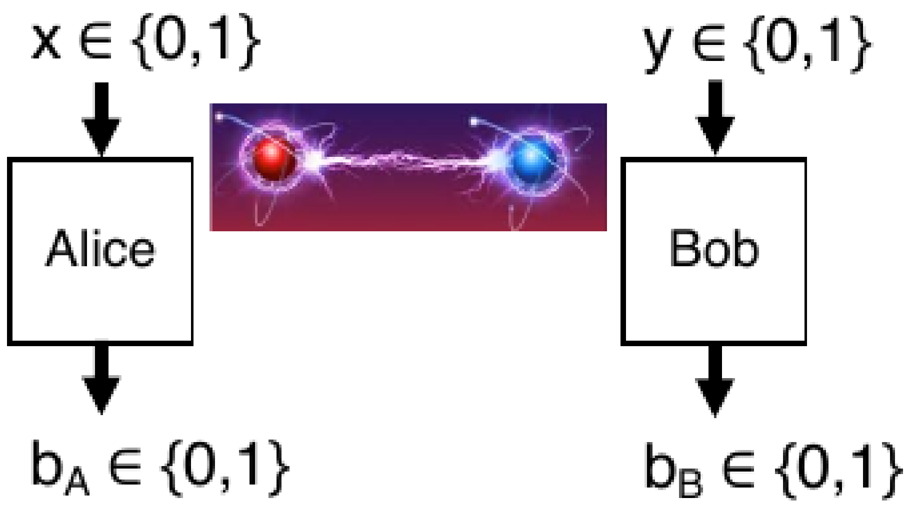
winning condition: $b_A \oplus b_B = xy$

Inputs	0, 0	0, 1	1, 0	1, 1
Outputs	0, 0	0, 0	0, 0	0, 1
	1, 1	1, 1	1, 1	1, 0

- The **XOR** of the **outputs** is the **AND** of the **inputs**, **without communication**

CHSH game

CHSH Game [Clauser, Horne, Shimony, Holt 1969]



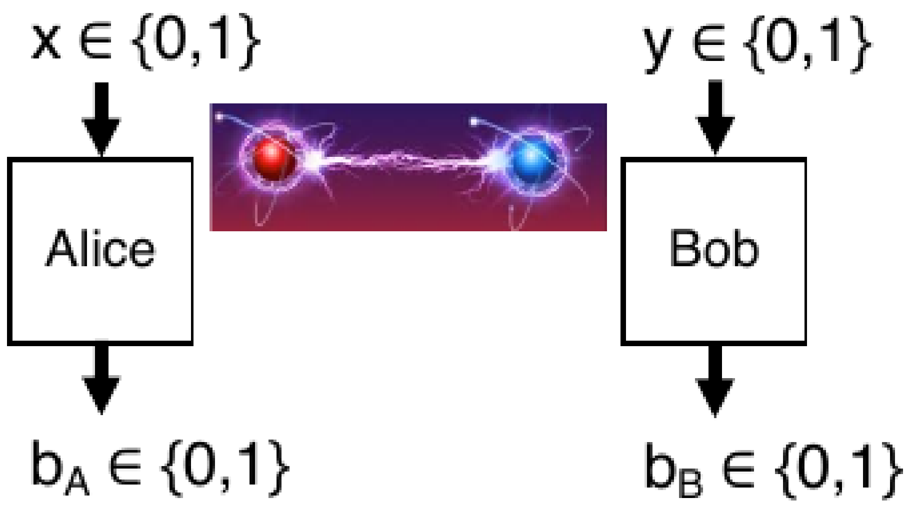
winning condition: $b_A \oplus b_B = xy$

Inputs	$0,0$	$0,1$	$1,0$	$1,1$
Outputs	$0,0$	$0,0$	$0,0$	$0,1$
	$1,1$	$1,1$	$1,1$	$1,0$

- The **XOR** of the **outputs** is the **AND** of the **inputs**, **without communication**
- **Classically**: winning probability $\leq 75\%$, even with shared randomness

CHSH game

CHSH Game [Clauser, Horne, Shimony, Holt 1969]



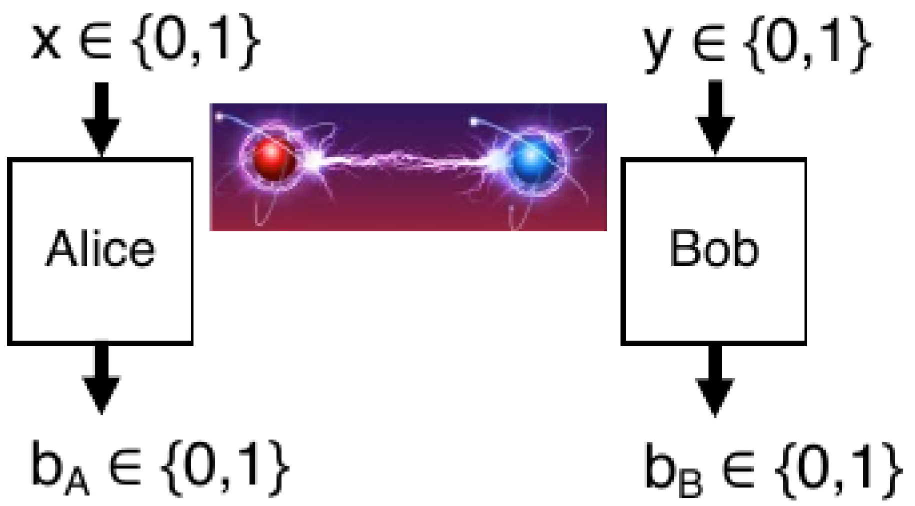
winning condition: $b_A \oplus b_B = xy$

Inputs	0,0	0,1	1,0	1,1
Outputs	0,0	0,0	0,0	0,1
	1,1	1,1	1,1	1,0

- The **XOR** of the **outputs** is the **AND** of the **inputs**, **without communication**
- **Classically**: winning probability $\leq 75\%$, even with shared randomness
- **Quantum**: winning probability $\approx 83\%$ with shared quantum state (entanglement)

CHSH game

CHSH Game [Clauser, Horne, Shimony, Holt 1969]

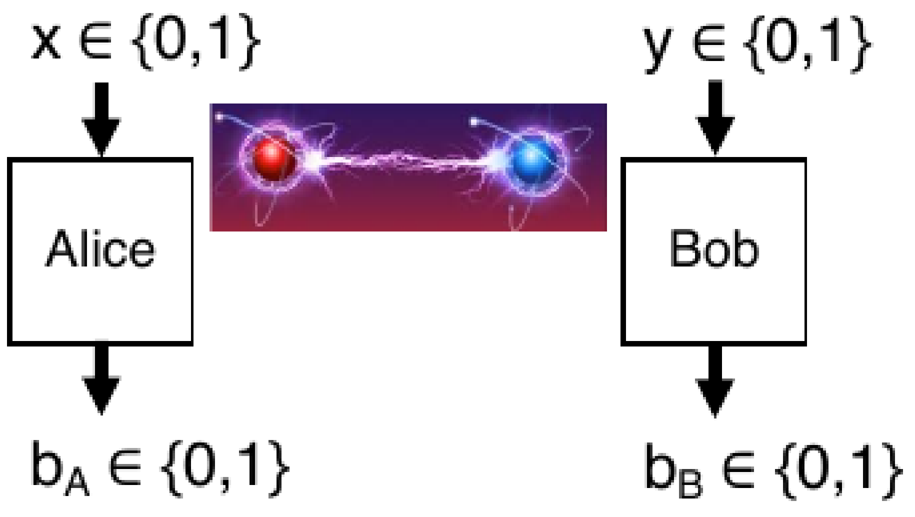


winning condition: $b_A \oplus b_B = xy$

Inputs	0,0	0,1	1,0	1,1
Outputs	0,0	0,0	0,0	0,1
	1,1	1,1	1,1	1,0

- The **XOR** of the **outputs** is the **AND** of the **inputs**, **without communication**
- **Classically**: winning probability $\leq 75\%$, even with shared randomness
- **Quantum**: winning probability $\approx 83\%$ with shared quantum state (entanglement)
- **Non-signaling**: winning probability 1

CHSH Game [Clauser, Horne, Shimony, Holt 1969]



winning condition: $b_A \oplus b_B = xy$

Inputs	$0,0$	$0,1$	$1,0$	$1,1$
Outputs	$0,0$	$0,0$	$0,0$	$0,1$
	$1,1$	$1,1$	$1,1$	$1,0$

- The **XOR** of the **outputs** is the **AND** of the **inputs**, **without communication**
- **Classically**: winning probability $\leq 75\%$, even with shared randomness
- **Quantum**: winning probability $\approx 83\%$ with shared quantum state (entanglement)
- **Non-signaling**: winning probability 1
  - *strategy*: sample u.a.r. among the correct solutions

Iterated CHSH problem

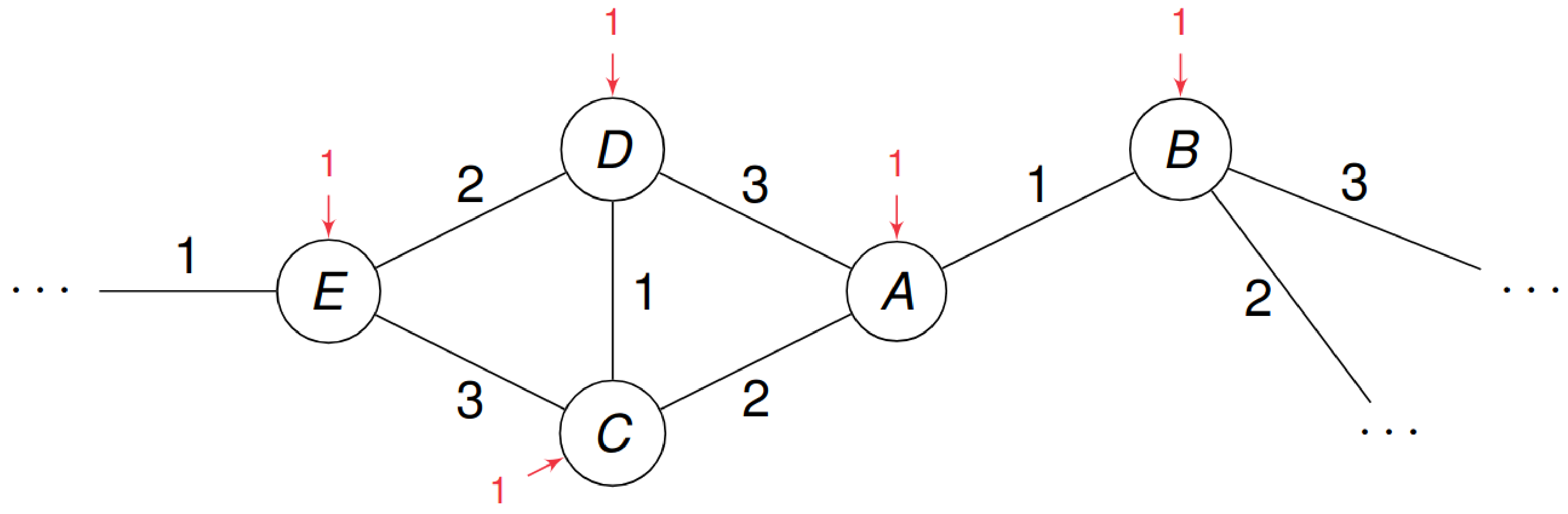
- **Network of CHSH!**

Iterated CHSH problem

- **Network of CHSH!**
- **How?**

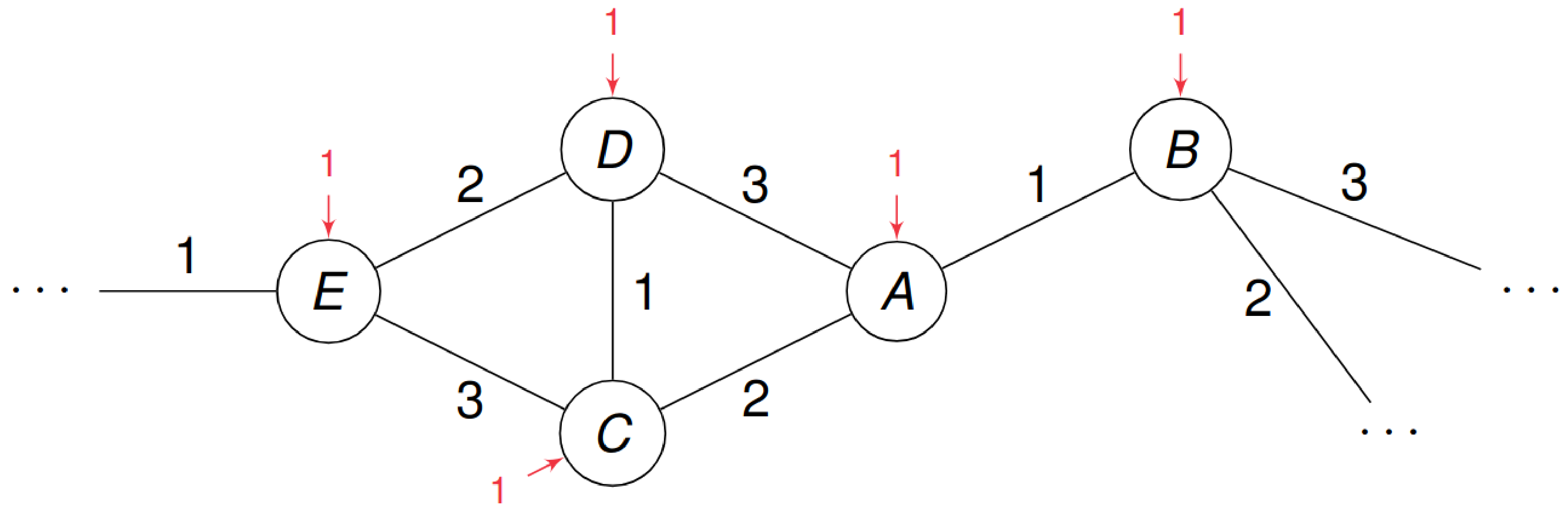
Iterated CHSH problem

- **Network of CHSH!**
- **How?**



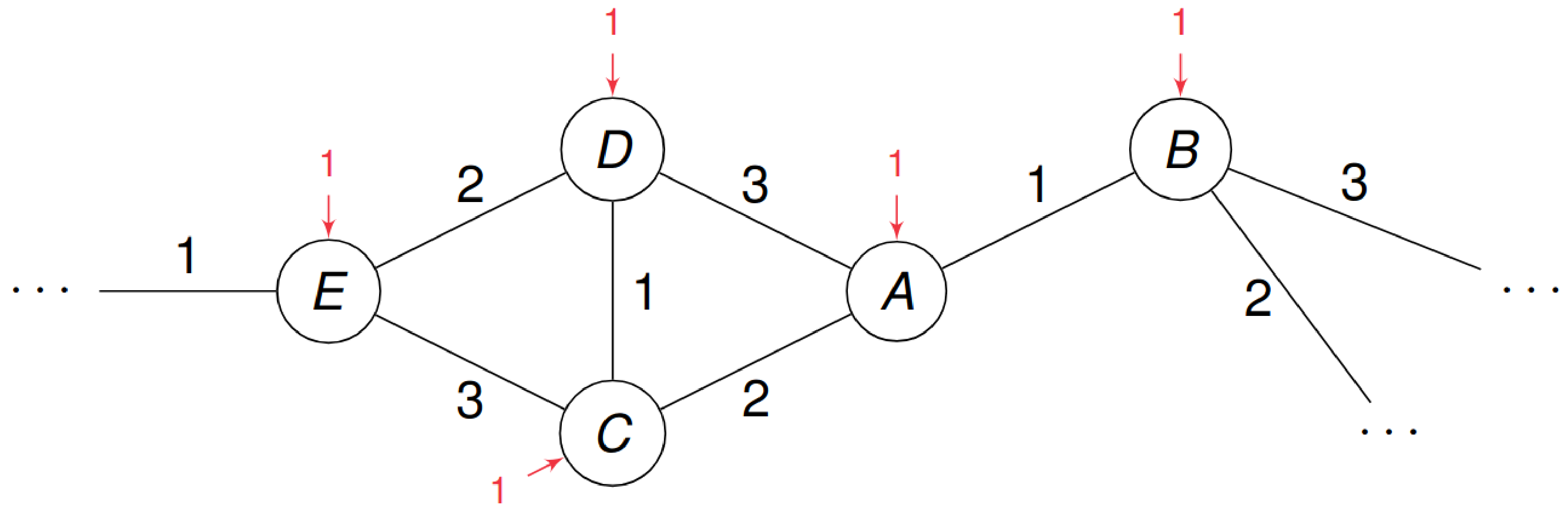
Iterated CHSH problem

- **Network of CHSH!**
- **How?**
- Δ-regular graph



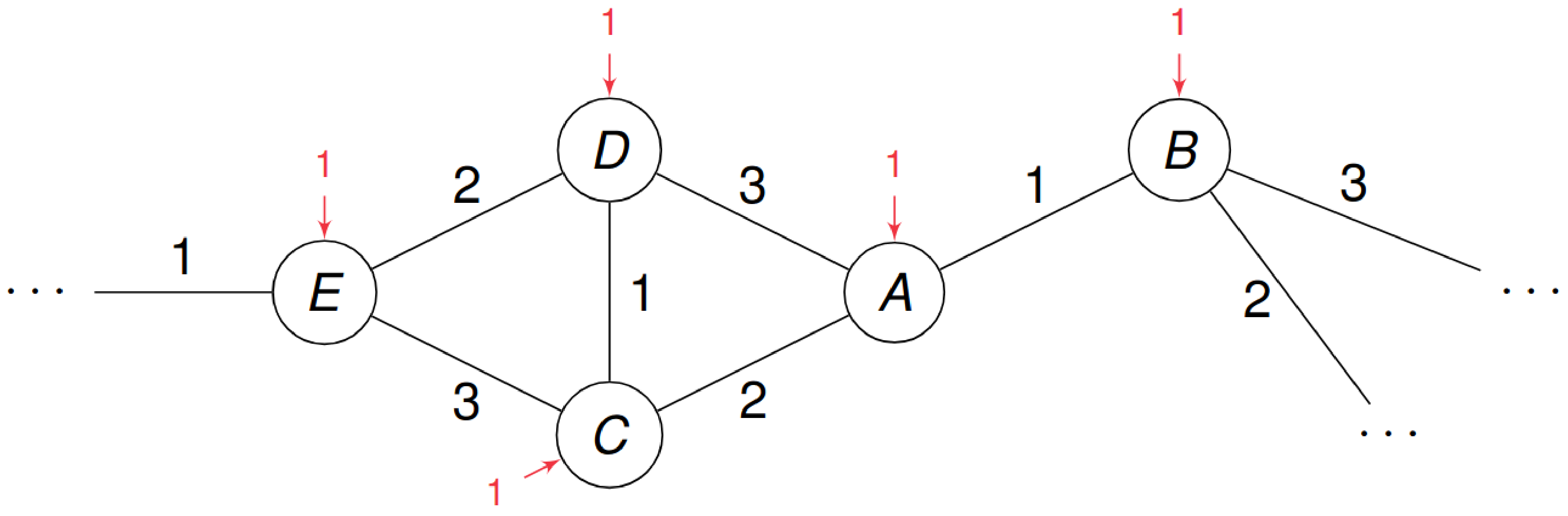
Iterated CHSH problem

- **Network of CHSH!**
- **How?**
- Δ-regular graph
- Input Δ-edge coloring



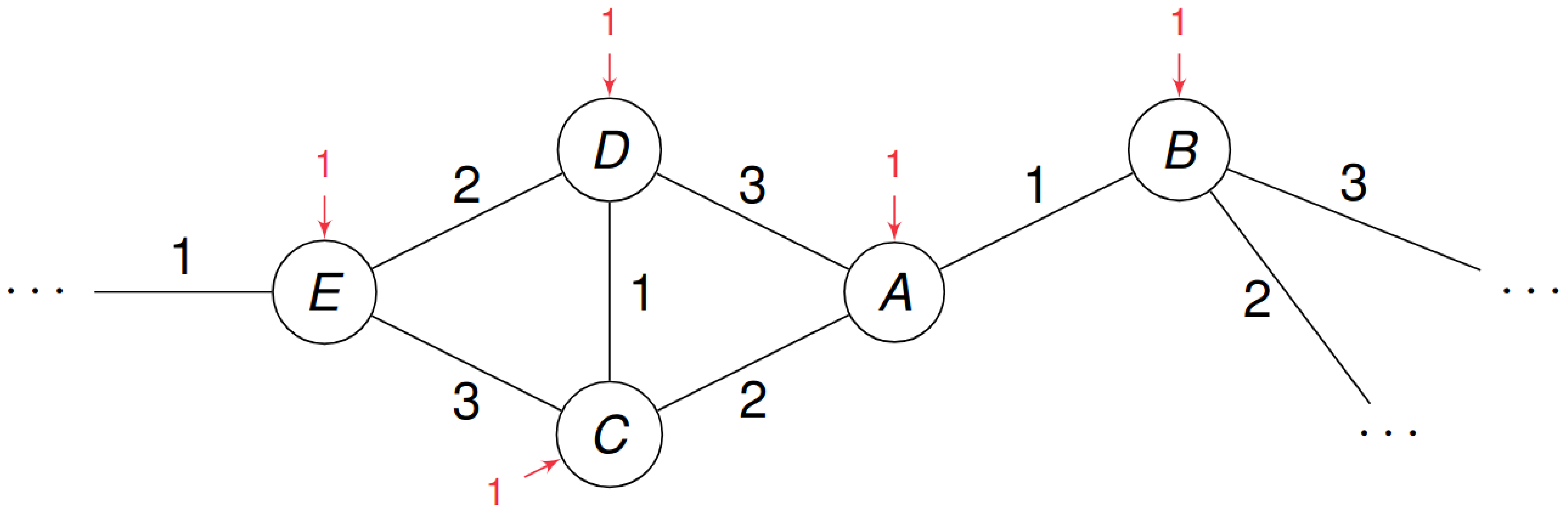
Iterated CHSH problem

- **Network of CHSH!**
- **How?**
- Δ-regular graph
- Input Δ-edge coloring
    - each edge is a CHSH game



Iterated CHSH problem

- **Network of CHSH!**
- **How?**
- Δ-regular graph
- Input Δ-edge coloring
  - each edge is a CHSH game
  - gives *ordering* of games



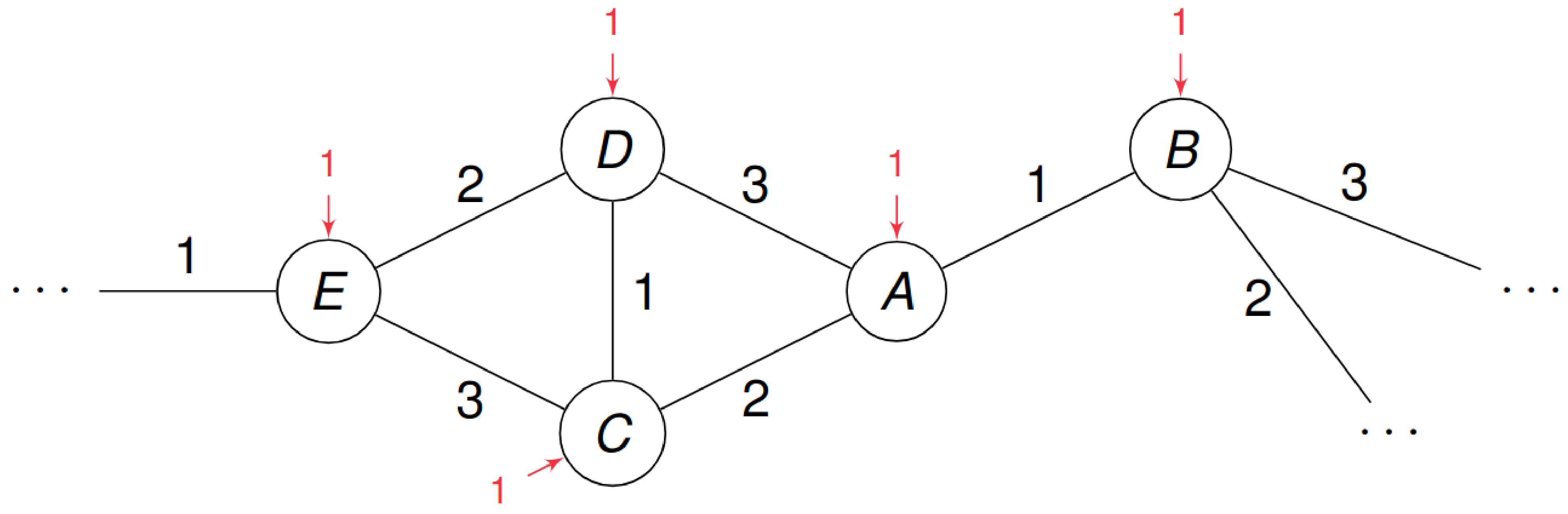
Iterated CHSH problem

- **Network of CHSH!**
- **How?**
- Δ-regular graph
- Input Δ-edge coloring

  - each edge is a CHSH game

  - gives *ordering* of games

- Input of game i is output of game $i-1$

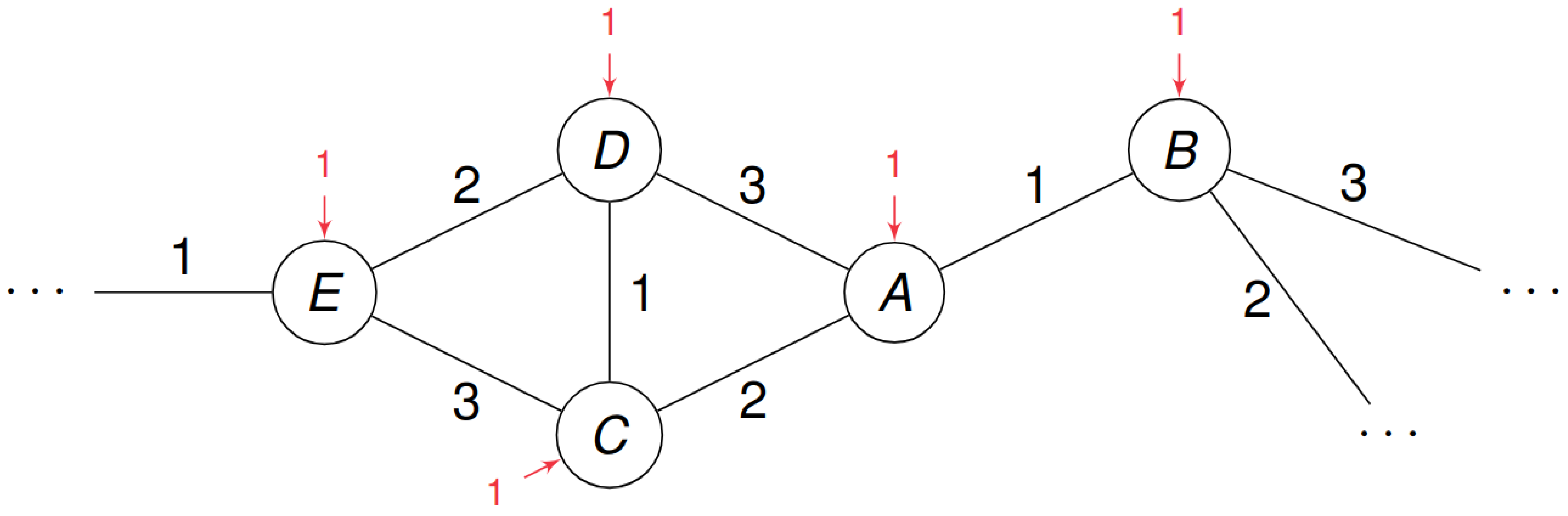


Iterated CHSH problem

- **Network of CHSH!**
- **How?**
- Δ-regular graph
- Input Δ-edge coloring

    - each edge is a CHSH game

    - gives *ordering* of games



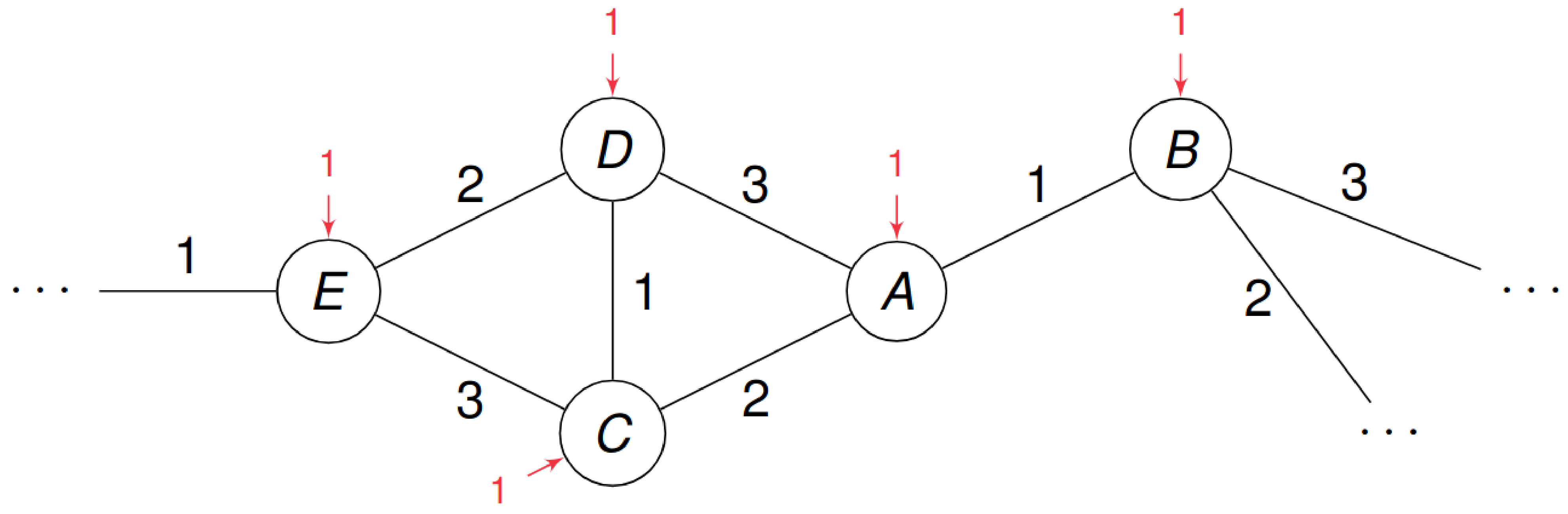
- Input of game i is output of game $i - 1$
- Input of game 1 is always 1

Iterated CHSH problem

- **Network of CHSH!**
- **How?**
- Δ-regular graph
- Input Δ-edge coloring

  - each edge is a CHSH game

  - gives *ordering* of games

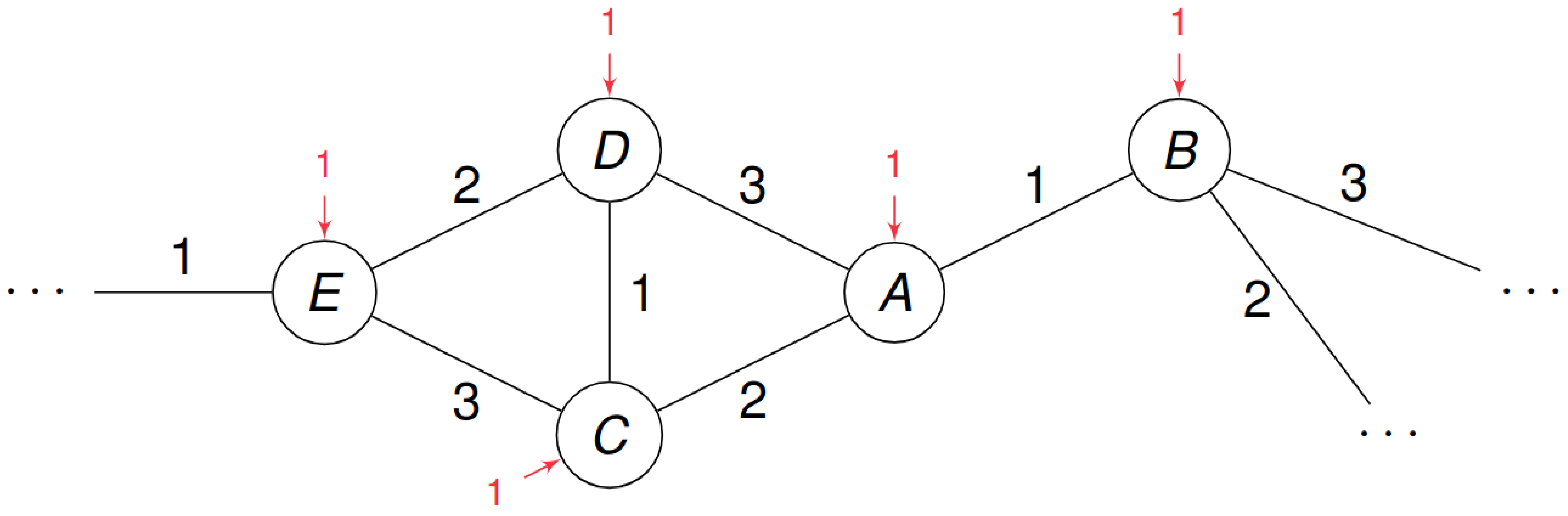


- Input of game i is output of game $i-1$
- Input of game $\mathbf{1}$ is always $\mathbf{1}$

  - otherwise we can “cheat” by collapsing quickly to output $\mathbf{0}$

Iterated CHSH problem

- **Network of CHSH!**
- **How?**
- Δ-regular graph
- Input Δ-edge coloring
  - each edge is a CHSH game
  - gives *ordering* of games

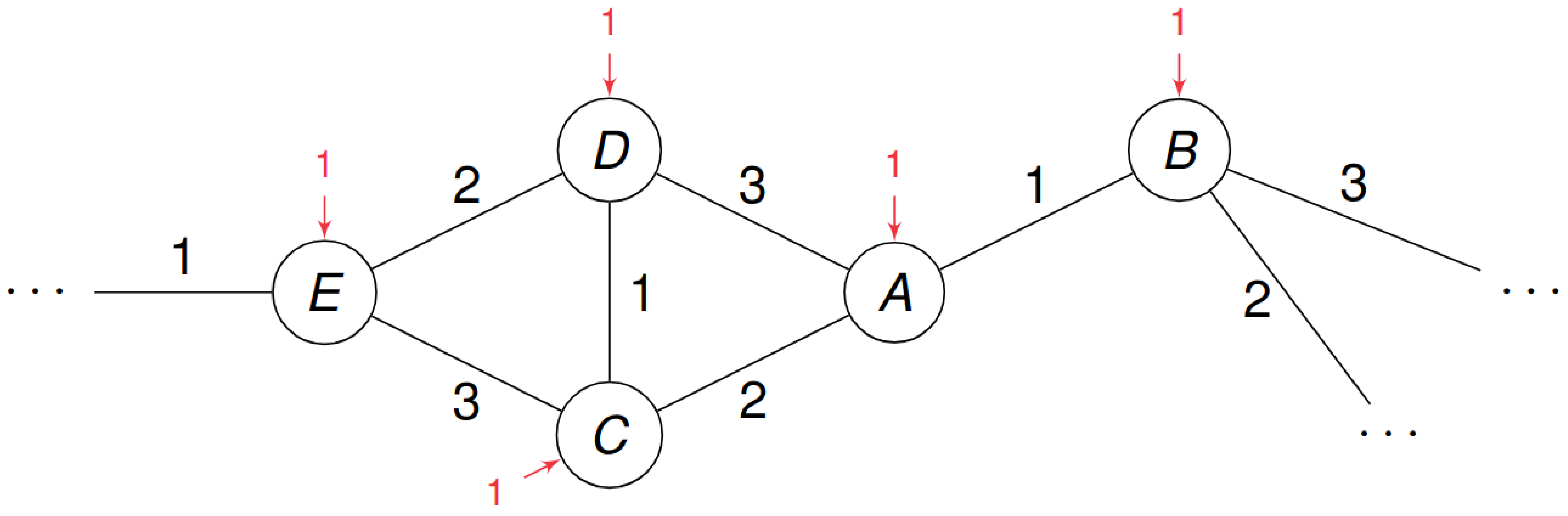


- Input of game i is output of game $i-1$
- Input of game 1 is always 1
  - otherwise we can “cheat” by collapsing quickly to output 0

- **Classically**: trivial $O(\Delta)$-time algorithm (solve one by one)

Iterated CHSH problem

- **Network of CHSH!**
- **How?**
- Δ-regular graph
- Input Δ-edge coloring
  - each edge is a CHSH game
  - gives *ordering* of games



- Input of game i is output of game $i-1$
- Input of game 1 is always 1
  - otherwise we can “cheat” by collapsing quickly to output 0

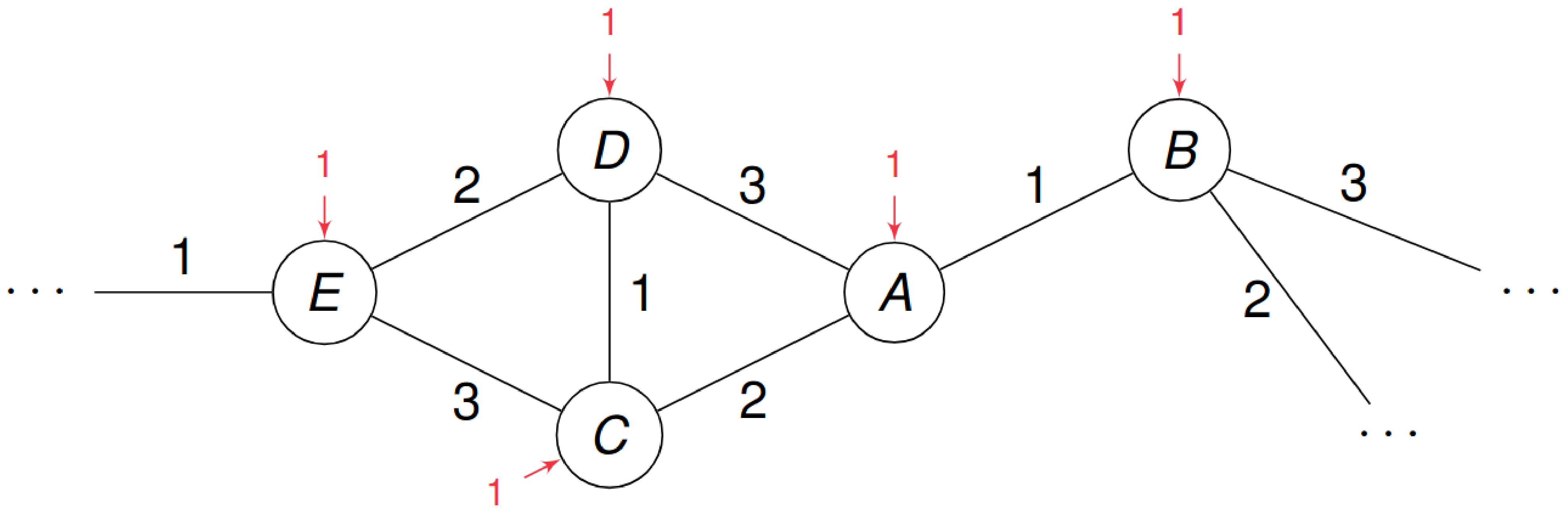
- **Classically**: trivial $O(\Delta)$-time algorithm (solve one by one)
- **Non-signaling**: non-signaling distribution with locality 0

Iterated CHSH problem

- **Network of CHSH!**
- **How?**
- Δ-regular graph
- Input Δ-edge coloring

  - each edge is a CHSH game

  - gives *ordering* of games



- Input of game i is output of game $i-1$
- Input of game 1 is always 1

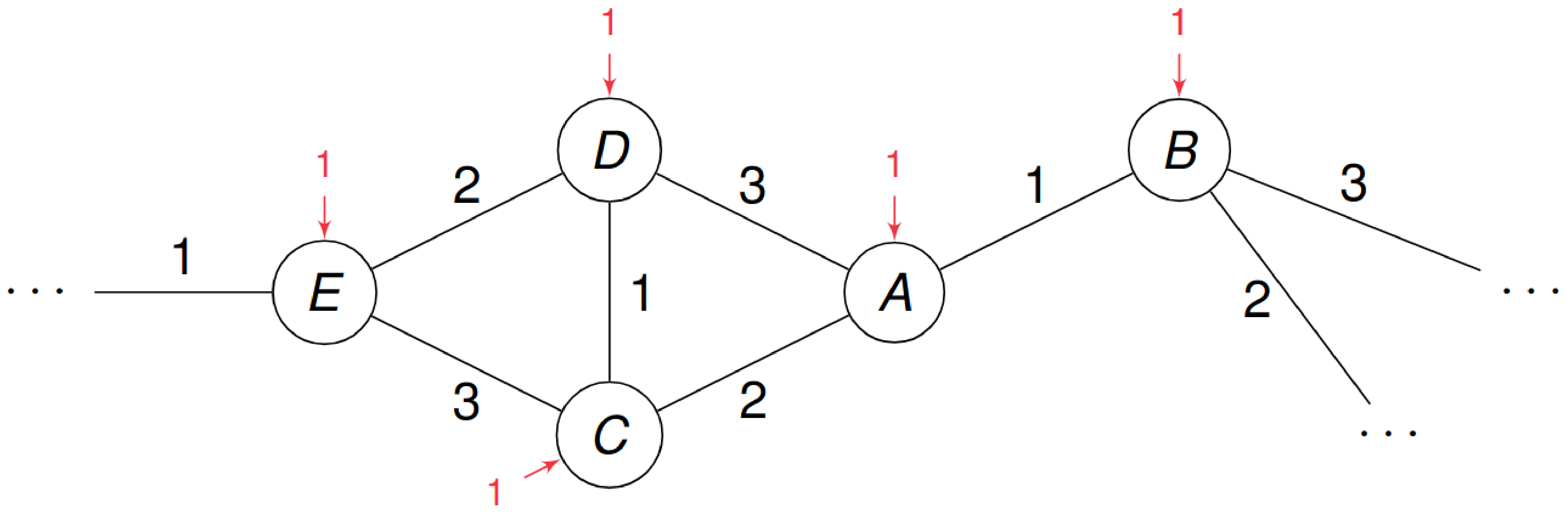
  - otherwise we can "cheat" by collapsing quickly to output 0

- **Classical lower bound?**

- **Classically**: trivial $O(\Delta)$-time algorithm (solve one by one)
- **Non-signaling**: non-signaling distribution with locality 0

Iterated CHSH problem

- **Network of CHSH!**
- **How?**
- Δ-regular graph
- Input Δ-edge coloring
  - each edge is a CHSH game
  - gives *ordering* of games

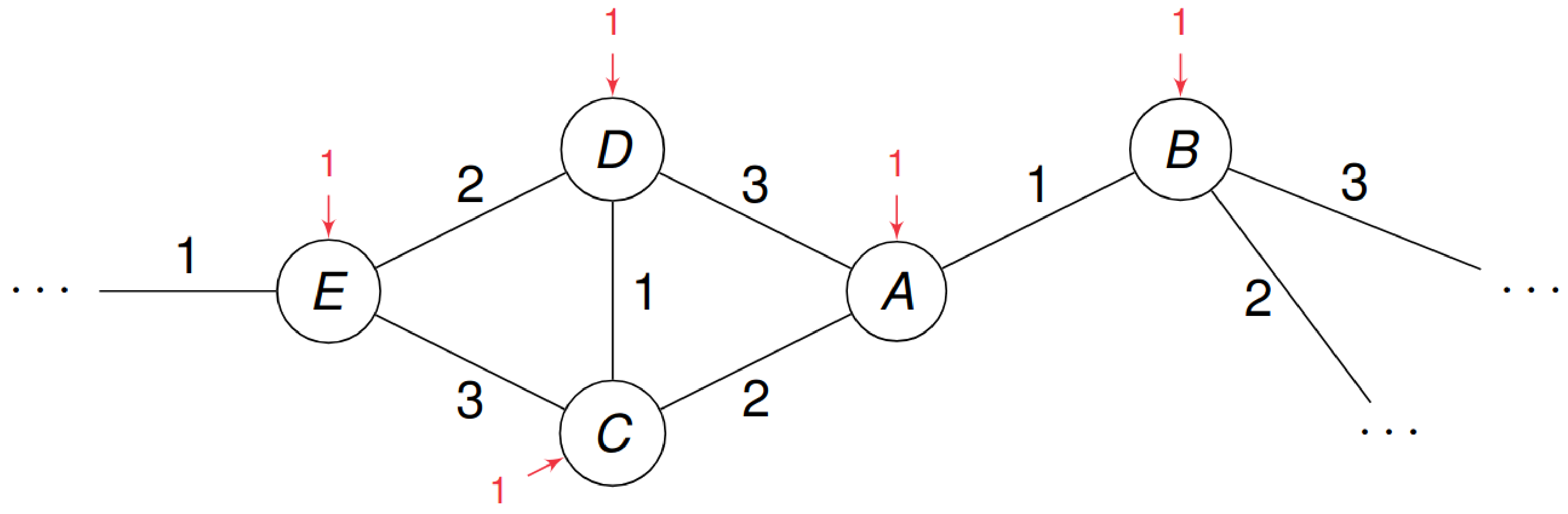


- Input of game i is output of game $i-1$
- Input of game 1 is always 1
  - otherwise we can "cheat" by collapsing quickly to output 0
- **Classically**: trivial $O(\Delta)$-time algorithm (solve one by one)
- **Non-signaling**: non-signaling distribution with locality 0

- **Classical lower bound?**
  - $\Omega(\Delta)$ by round elimination [STOC '25b]

Iterated CHSH problem

- **Network of CHSH!**
- **How?**
- Δ-regular graph
- Input Δ-edge coloring
    - each edge is a CHSH game
    - gives *ordering* of games

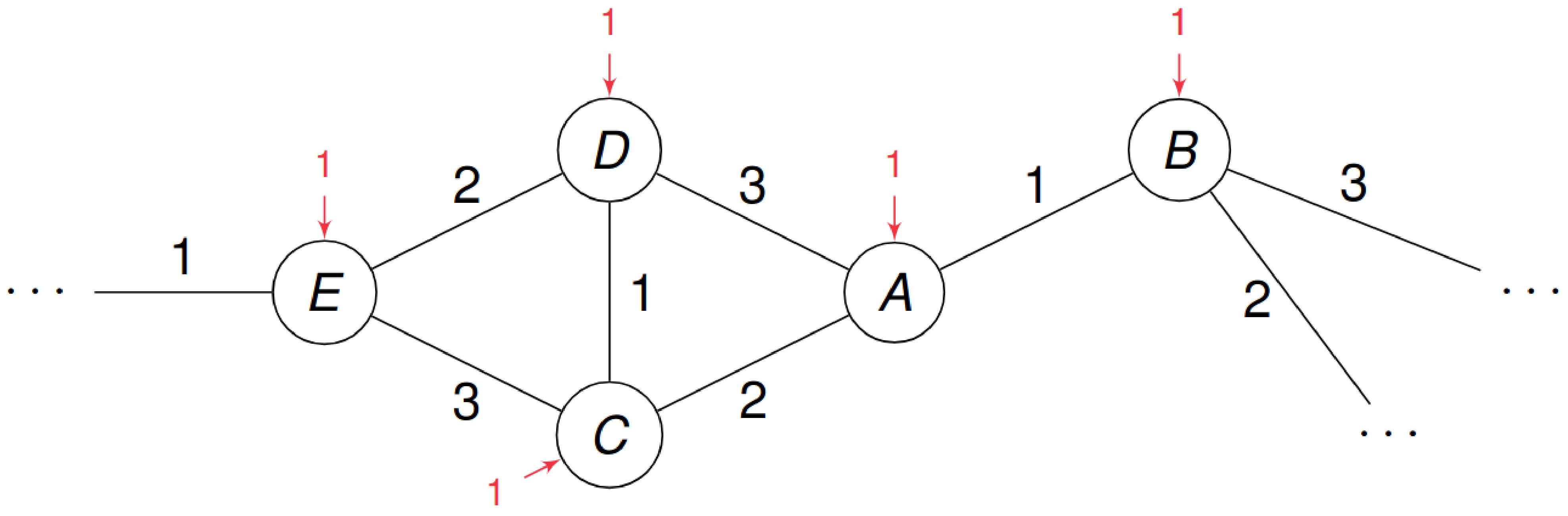


- Input of game i is output of game $i-1$
- Input of game 1 is always 1
    - otherwise we can "cheat" by collapsing quickly to output 0
- **Classically**: trivial $O(\Delta)$-time algorithm (solve one by one)
- **Non-signaling**: non-signaling distribution with locality 0

- **Classical lower bound?**
    - $\Omega(\Delta)$ by round elimination [STOC '25b]
- **Quantum upper bound?**

Iterated CHSH problem

- **Network of CHSH!**
- **How?**
- Δ-regular graph
- Input Δ-edge coloring
    - each edge is a CHSH game
    - gives *ordering* of games

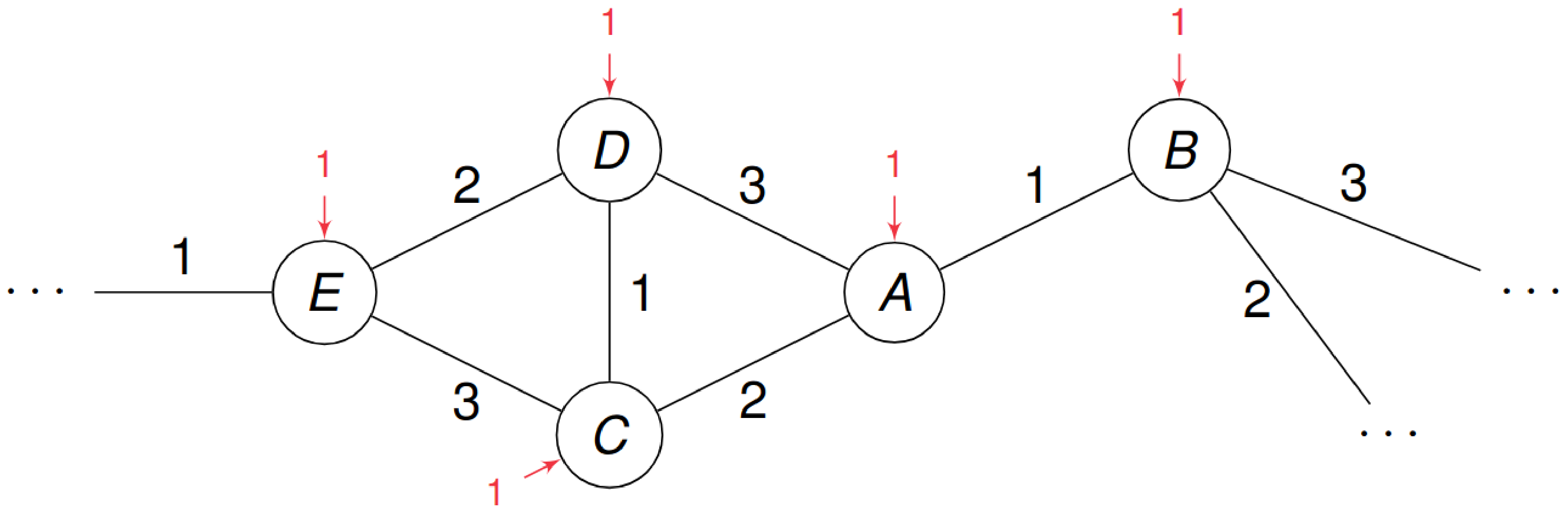


- Input of game i is output of game $i-1$
- Input of game 1 is always 1
    - otherwise we can "cheat" by collapsing quickly to output 0
- **Classically**: trivial $O(\Delta)$-time algorithm (solve one by one)
- **Non-signaling**: non-signaling distribution with locality 0

- **Classical lower bound?**
    - $\Omega(\Delta)$ by round elimination [STOC '25b]
- **Quantum upper bound?**
    - winning prob. of single game is $\approx 83\%$

Iterated CHSH problem

- **Network of CHSH!**
- **How?**
- Δ-regular graph
- Input Δ-edge coloring
  - each edge is a CHSH game
  - gives *ordering* of games



- Input of game i is output of game $i-1$
- Input of game 1 is always 1
  - otherwise we can "cheat" by collapsing quickly to output 0
- **Classically**: trivial $O(\Delta)$-time algorithm (solve one by one)
- **Non-signaling**: non-signaling distribution with locality 0

- **Classical lower bound?**
  - $\Omega(\Delta)$ by round elimination [STOC '25b]
- **Quantum upper bound?**
  - winning prob. of single game is $\approx 83\%$
  - we need a better game (win 100%)

GHZ game

- **Greenberger-Horne-Zeilinger game**

Alice

Bob

Charlie

Inputs	$0,0,0$	$0,1,1$	$1,0,1$	$1,1,0$
Outputs	$0,0,0$	$1,0,0$	$1,0,0$	$1,0,0$
	$0,1,1$	$0,1,0$	$0,1,0$	$0,1,0$
	$1,0,1$	$0,0,1$	$0,0,1$	$0,0,1$
	$1,1,0$	$1,1,1$	$1,1,1$	$1,1,1$

GHZ game

- **Greenberger-Horne-Zeilinger game**

Alice

Bob

Charlie

- Inputs always even number of 1

Inputs	$0,0,0$	$0,1,1$	$1,0,1$	$1,1,0$
Outputs	$0,0,0$	$1,0,0$	$1,0,0$	$1,0,0$
	$0,1,1$	$0,1,0$	$0,1,0$	$0,1,0$
	$1,0,1$	$0,0,1$	$0,0,1$	$0,0,1$
	$1,1,0$	$1,1,1$	$1,1,1$	$1,1,1$

GHZ game

- **Greenberger-Horne-Zeilinger game**

Alice

Bob

Charlie

Inputs	$0,0,0$	$0,1,1$	$1,0,1$	$1,1,0$
Outputs	$0,0,0$	$1,0,0$	$1,0,0$	$1,0,0$
	$0,1,1$	$0,1,0$	$0,1,0$	$0,1,0$
	$1,0,1$	$0,0,1$	$0,0,1$	$0,0,1$
	$1,1,0$	$1,1,1$	$1,1,1$	$1,1,1$

- Inputs always even number of 1
- **XOR** of **outputs** is 0 iff inputs are all 0, otherwise 1

GHZ game

- **Greenberger-Horne-Zeilinger game**

Alice

Bob

Charlie

Inputs	$0,0,0$	$0,1,1$	$1,0,1$	$1,1,0$
Outputs	$0,0,0$	$1,0,0$	$1,0,0$	$1,0,0$
	$0,1,1$	$0,1,0$	$0,1,0$	$0,1,0$
	$1,0,1$	$0,0,1$	$0,0,1$	$0,0,1$
	$1,1,0$	$1,1,1$	$1,1,1$	$1,1,1$

- Inputs always even number of 1
- **XOR** of **outputs** is 0 iff inputs are all 0, otherwise 1
- **Classically**: winning probability $\leq 75\%$, even with shared randomness

GHZ game

- **Greenberger-Horne-Zeilinger game**

Alice **Bob** **Charlie**

Inputs	$0,0,0$	$0,1,1$	$1,0,1$	$1,1,0$
Outputs	$0,0,0$	$1,0,0$	$1,0,0$	$1,0,0$
	$0,1,1$	$0,1,0$	$0,1,0$	$0,1,0$
	$1,0,1$	$0,0,1$	$0,0,1$	$0,0,1$
	$1,1,0$	$1,1,1$	$1,1,1$	$1,1,1$

- Inputs always even number of 1
- **XOR** of **outputs** is 0 iff inputs are all 0, otherwise 1
- **Classically**: winning probability $\leq 75\%$, even with shared randomness
- **Quantum**: winning probability 1 with shared quantum state (entanglement)

GHZ game

- **Greenberger-Horne-Zeilinger game**

Alice

Bob

Charlie

Inputs	$0,0,0$	$0,1,1$	$1,0,1$	$1,1,0$
Outputs	$0,0,0$	$1,0,0$	$1,0,0$	$1,0,0$
	$0,1,1$	$0,1,0$	$0,1,0$	$0,1,0$
	$1,0,1$	$0,0,1$	$0,0,1$	$0,0,1$
	$1,1,0$	$1,1,1$	$1,1,1$	$1,1,1$

- Inputs always even number of 1
- **XOR** of **outputs** is 0 iff inputs are all 0, otherwise 1
- **Classically**: winning probability $\leq 75\%$, even with shared randomness
- **Quantum**: winning probability 1 with shared quantum state (entanglement)
- Construct a network of **iterated GHZ** like before: 3 players $\implies$ *hypergraph!*

GHZ game

- **Greenberger-Horne-Zeilinger game**

Alice

Bob

Charlie

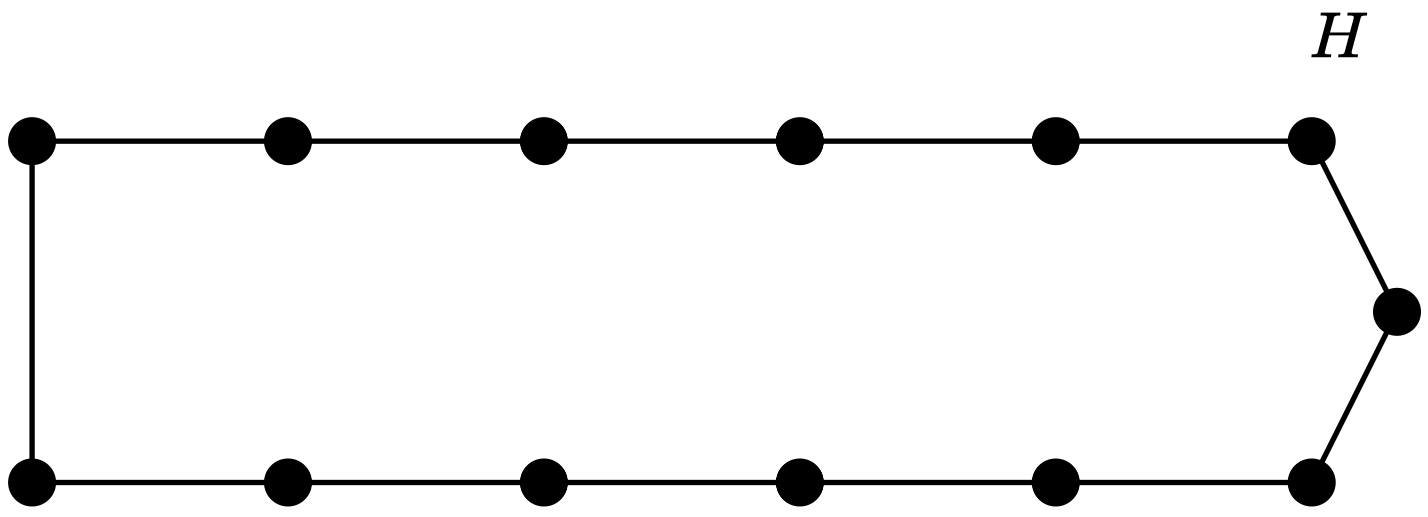
Inputs	$0,0,0$	$0,1,1$	$1,0,1$	$1,1,0$
Outputs	$0,0,0$	$1,0,0$	$1,0,0$	$1,0,0$
	$0,1,1$	$0,1,0$	$0,1,0$	$0,1,0$
	$1,0,1$	$0,0,1$	$0,0,1$	$0,0,1$
	$1,1,0$	$1,1,1$	$1,1,1$	$1,1,1$

- Inputs always even number of 1
- **XOR** of **outputs** is 0 iff inputs are all 0, otherwise 1
- **Classically**: winning probability $\leq 75\%$, even with shared randomness
- **Quantum**: winning probability 1 with shared quantum state (entanglement)
- Construct a network of **iterated GHZ** like before: 3 players $\implies$ *hypergraph!*
- Classical complexity $\Theta(\Delta)$, quantum complexity 1 round (just to share the quantum state)

Boosting failure probability: randomized-LOCAL

Problem: $\mathbf{2}$-coloring $\mathbf{2}$-chromatic graphs

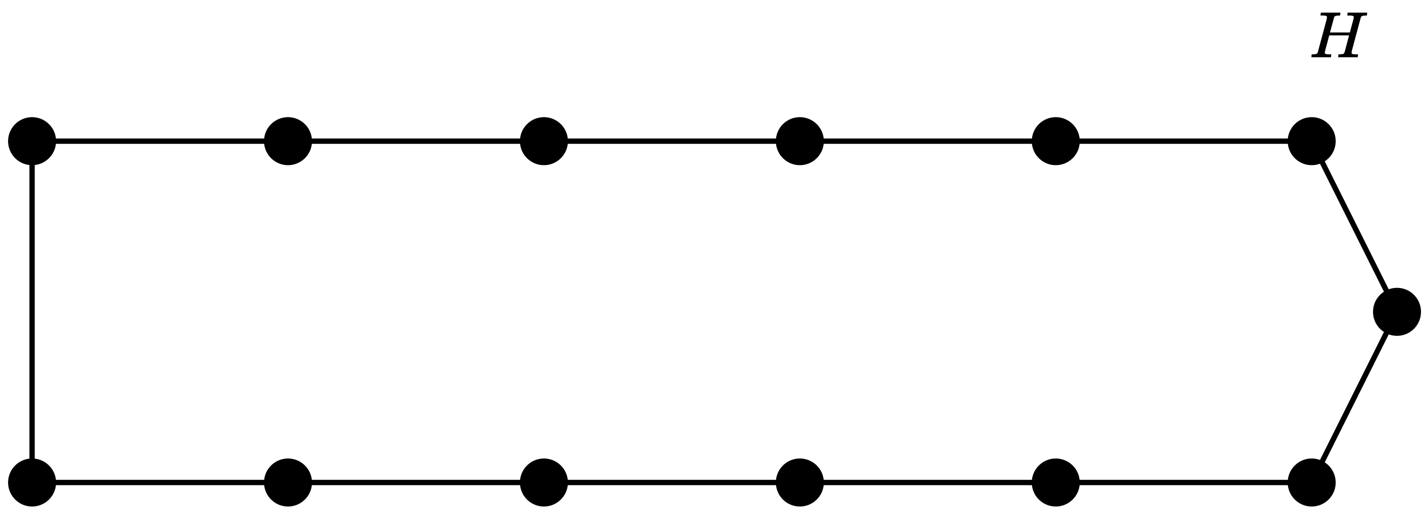
- randomized LOCAL



Boosting failure probability: randomized-LOCAL

Problem: $\mathbf{2}$-coloring $\mathbf{2}$-chromatic graphs

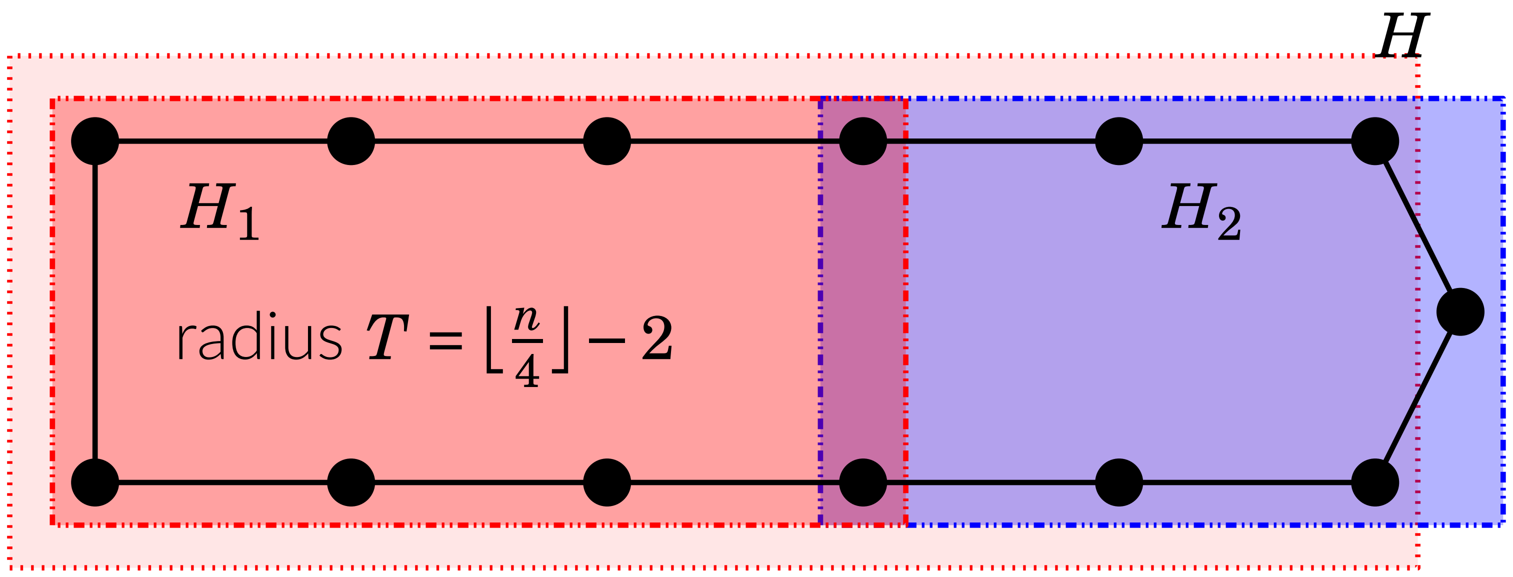
- randomized LOCAL
- "subdivide" graph $\boldsymbol{H}$ in regions $\boldsymbol{H_1}, \boldsymbol{H_2}$ such that
  - $\boldsymbol{H}[\mathcal{N}_{\boldsymbol{T}}(\boldsymbol{H_i})]$ is 2-colorable



Boosting failure probability: randomized-LOCAL

Problem: $\mathbf{2}$-coloring $\mathbf{2}$-chromatic graphs

- randomized LOCAL
- "subdivide" graph $\boldsymbol{H}$ in regions $\boldsymbol{H_1}, \boldsymbol{H_2}$ such that
  - $\boldsymbol{H}[\mathcal{N}_{\boldsymbol{T}}(\boldsymbol{H_i})]$ is 2-colorable

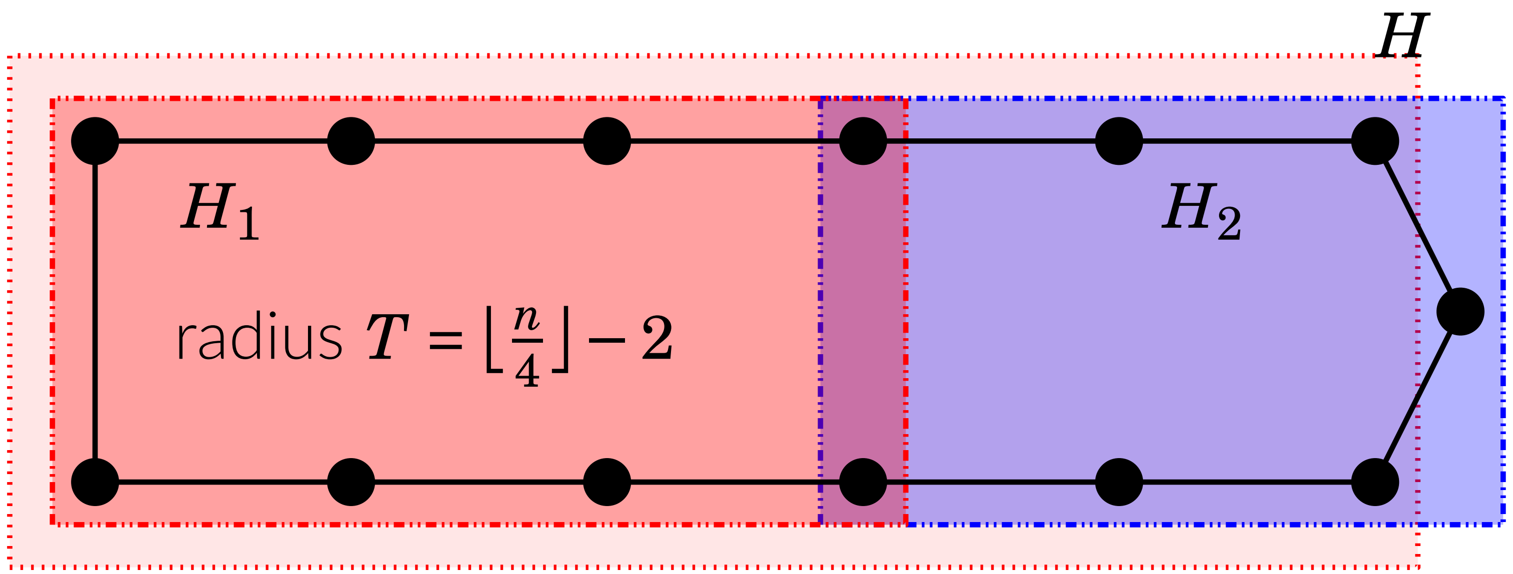


$$\max_{i=1,2}\{\Pr[\text{failure on } H_i]\} \geq \frac{1}{2}$$

Boosting failure probability: randomized-LOCAL

Problem: $\mathbf{2}$-coloring $\mathbf{2}$-chromatic graphs

- randomized LOCAL
- “subdivide” graph $\boldsymbol{H}$ in regions $\boldsymbol{H_1}, \boldsymbol{H_2}$ such that
  - $\boldsymbol{H}[\mathcal{N}_{\boldsymbol{T}}(\boldsymbol{H_i})]$ is 2-colorable
  - $\boldsymbol{N}$ copies of $\boldsymbol{H}[\mathcal{N}_{\boldsymbol{T}}(\boldsymbol{H_i})]$ can be glued together to form a 2-colorable graph



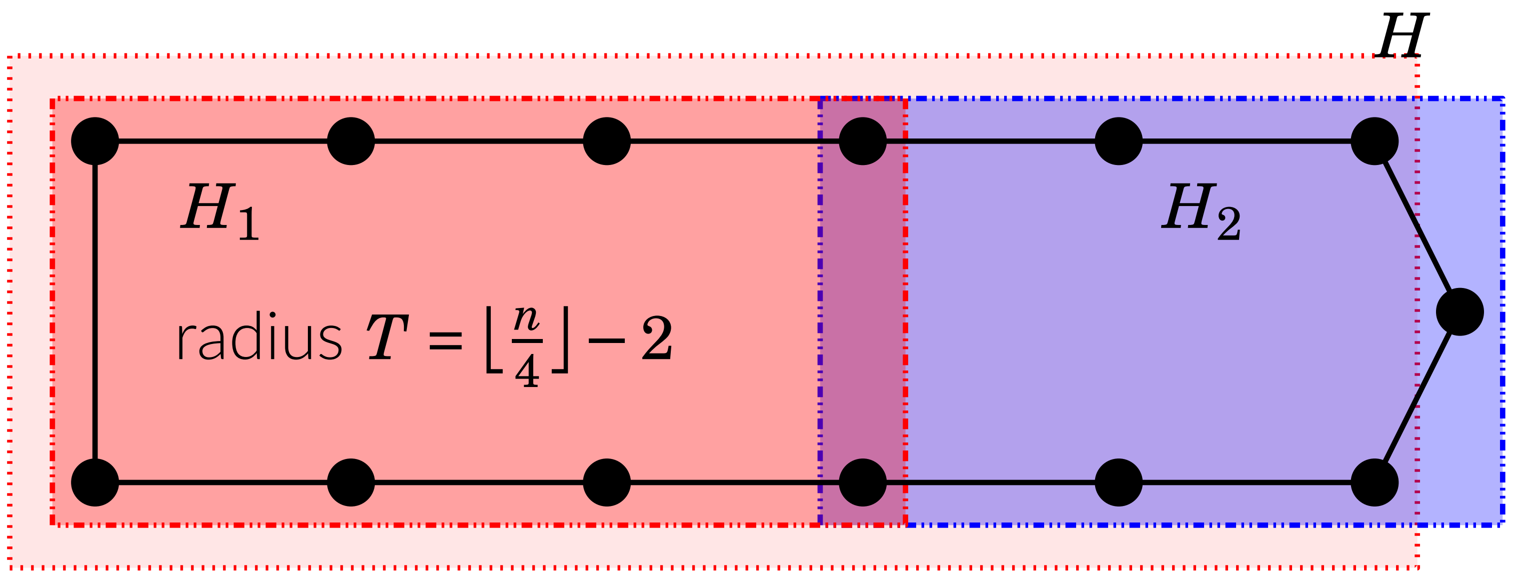
$$\max_{i=1,2}\{\Pr[\text{failure on } H_i]\} \geq \frac{1}{2}$$

Boosting failure probability: randomized-LOCAL

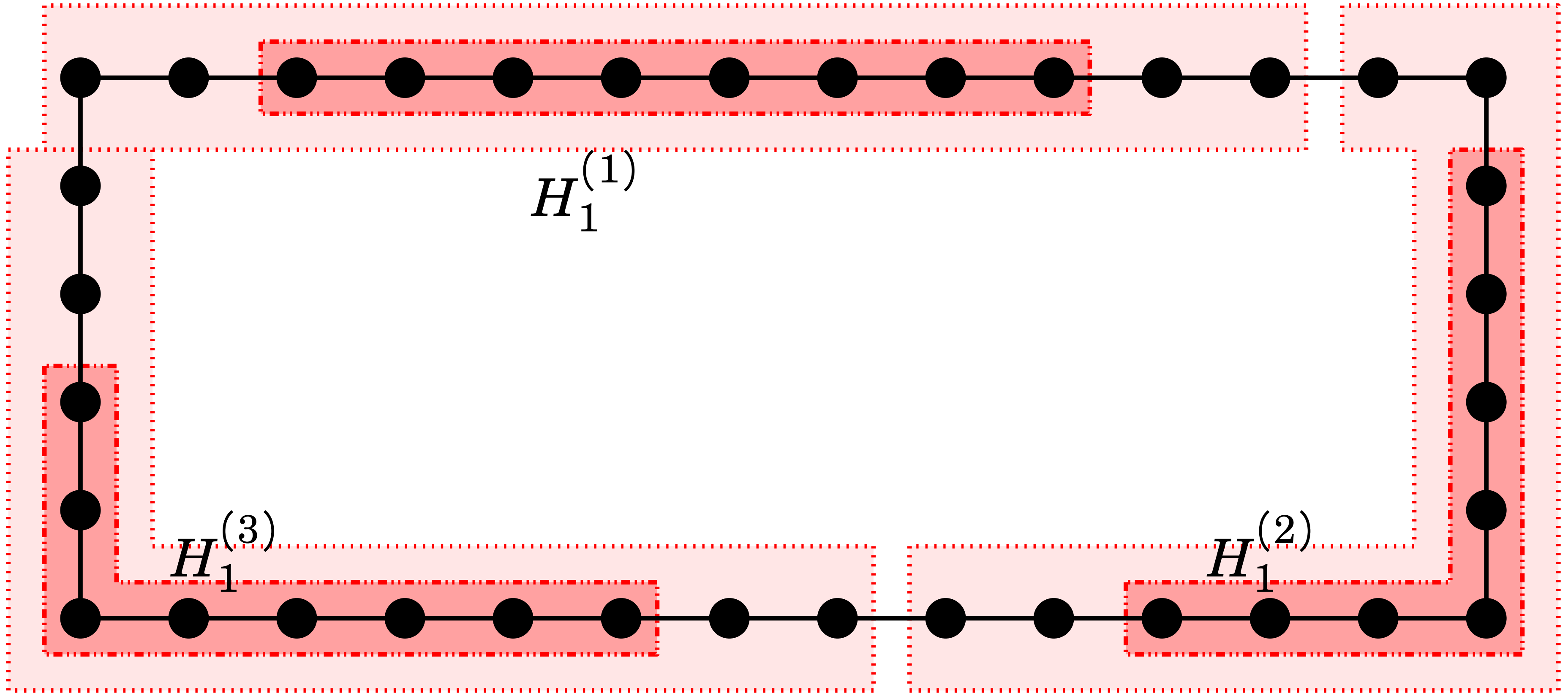
Problem: 2-coloring 2-chromatic graphs

- randomized LOCAL

- "subdivide" graph H in regions H_1, H_2 such that
  - $H[\mathcal{N}_T(H_i)]$ is 2-colorable
  - N copies of $H[\mathcal{N}_T(H_i)]$ can be glued together to form a 2-colorable graph



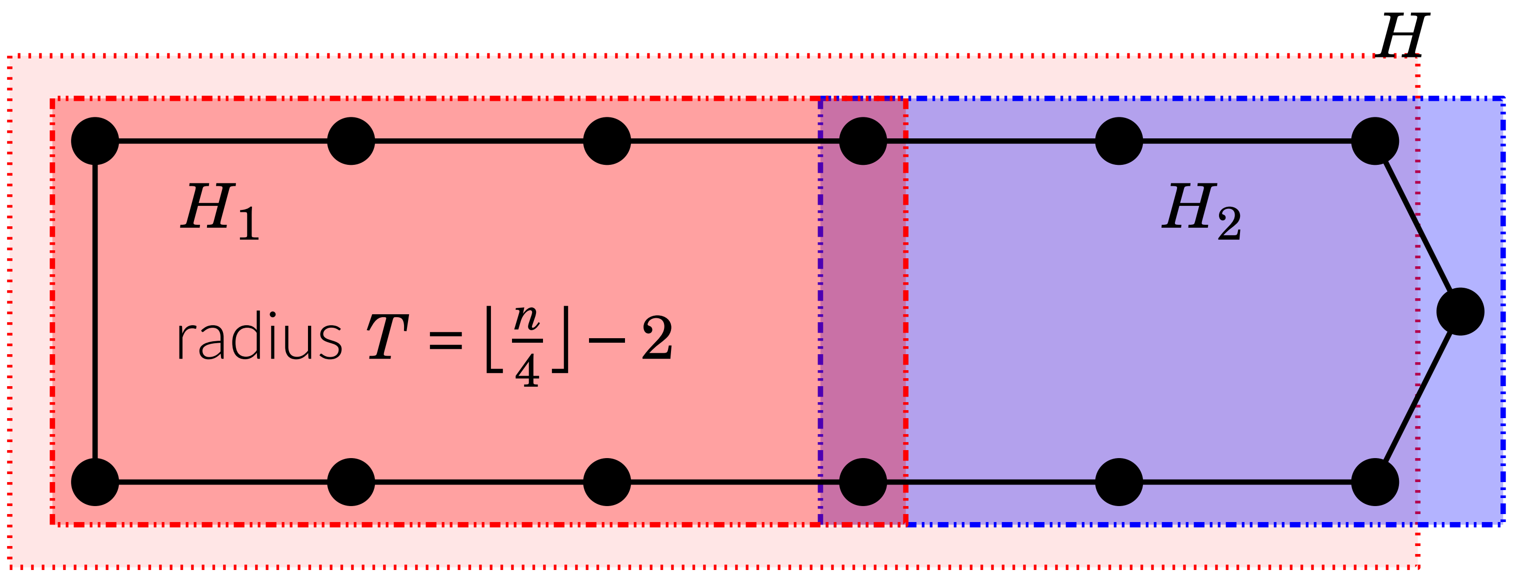
$$\max_{i=1,2}\{\Pr[\text{failure on } H_i]\} \geq \frac{1}{2}$$



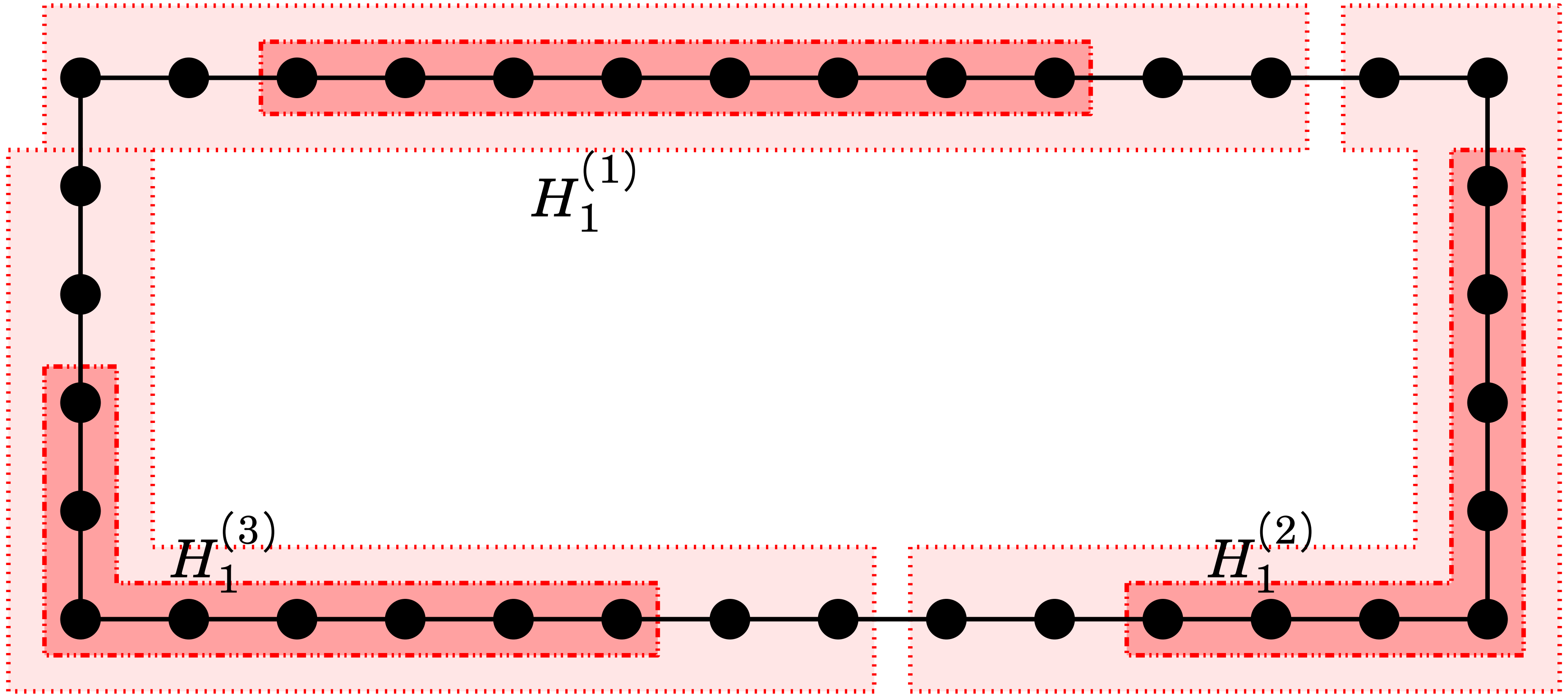
Boosting failure probability: randomized-LOCAL

Problem: 2-coloring 2-chromatic graphs

- randomized LOCAL
- "subdivide" graph H in regions H_1, H_2 such that
  - $H[\mathcal{N}_T(H_i)]$ is 2-colorable
  - N copies of $H[\mathcal{N}_T(H_i)]$ can be glued together to form a 2-colorable graph
- probability of failure $\geq 1 - (1 - \frac{1}{2})^N$
  - independence + union bound
  - cloning principle



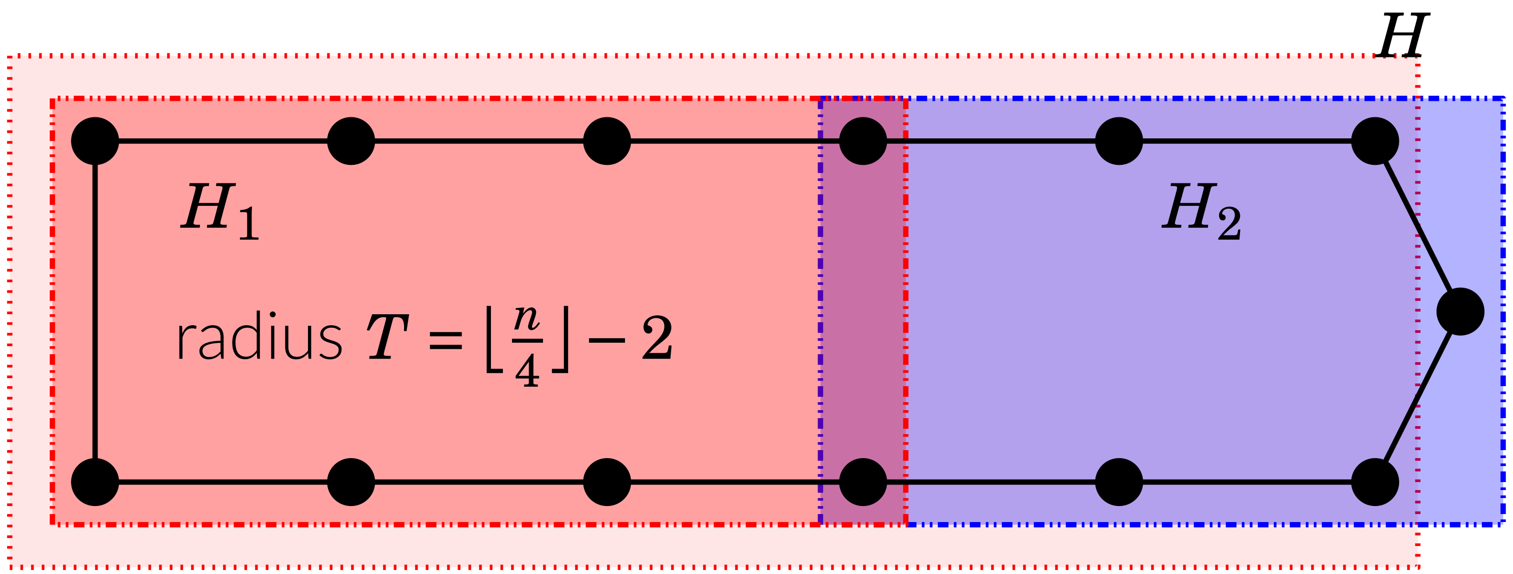
$\max_{i=1,2}\{\Pr[\text{failure on } H_i]\} \geq \frac{1}{2}$



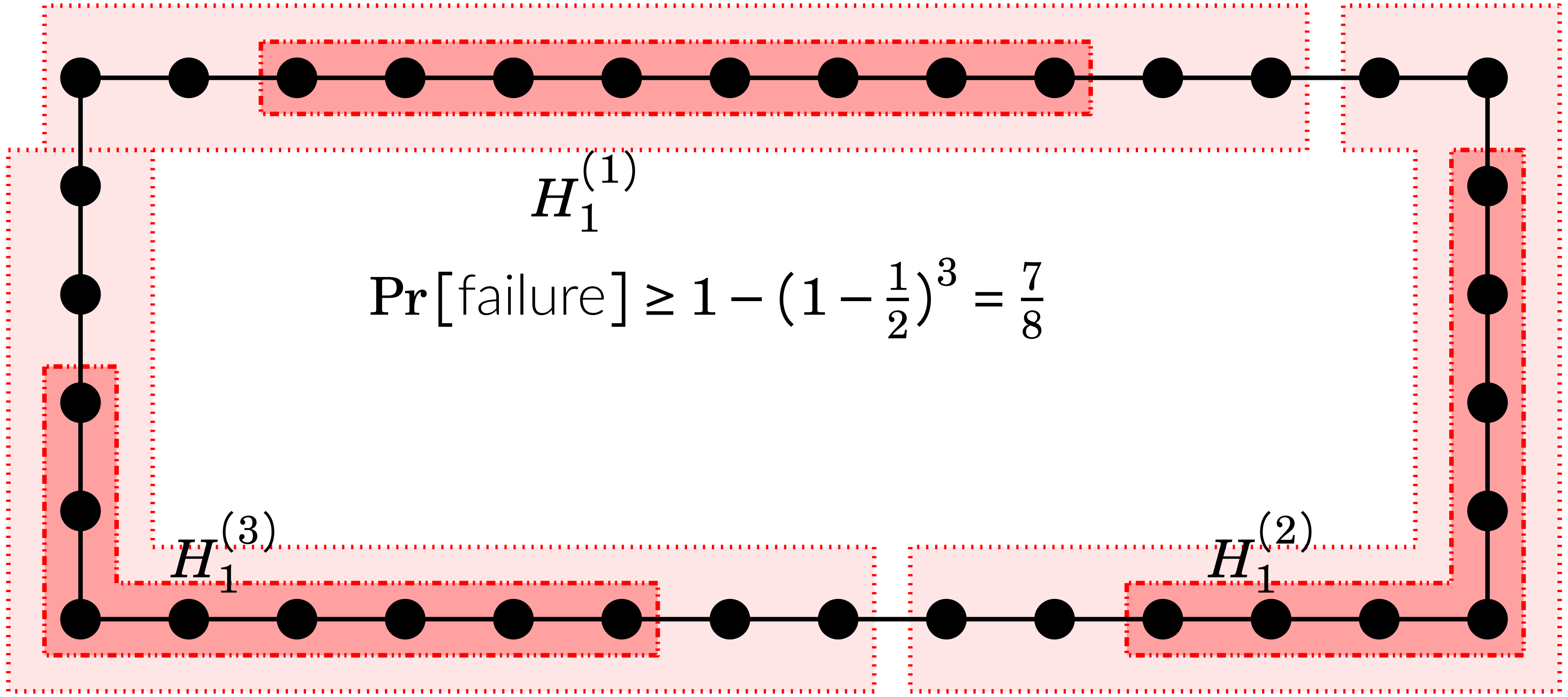
Boosting failure probability: randomized-LOCAL

Problem: 2-coloring 2-chromatic graphs

- randomized LOCAL
- "subdivide" graph H in regions H_1, H_2 such that
  - $H[\mathcal{N}_T(H_i)]$ is 2-colorable
  - N copies of $H[\mathcal{N}_T(H_i)]$ can be glued together to form a 2-colorable graph
- probability of failure $\geq 1 - (1 - \frac{1}{2})^N$
  - independence + union bound
  - cloning principle



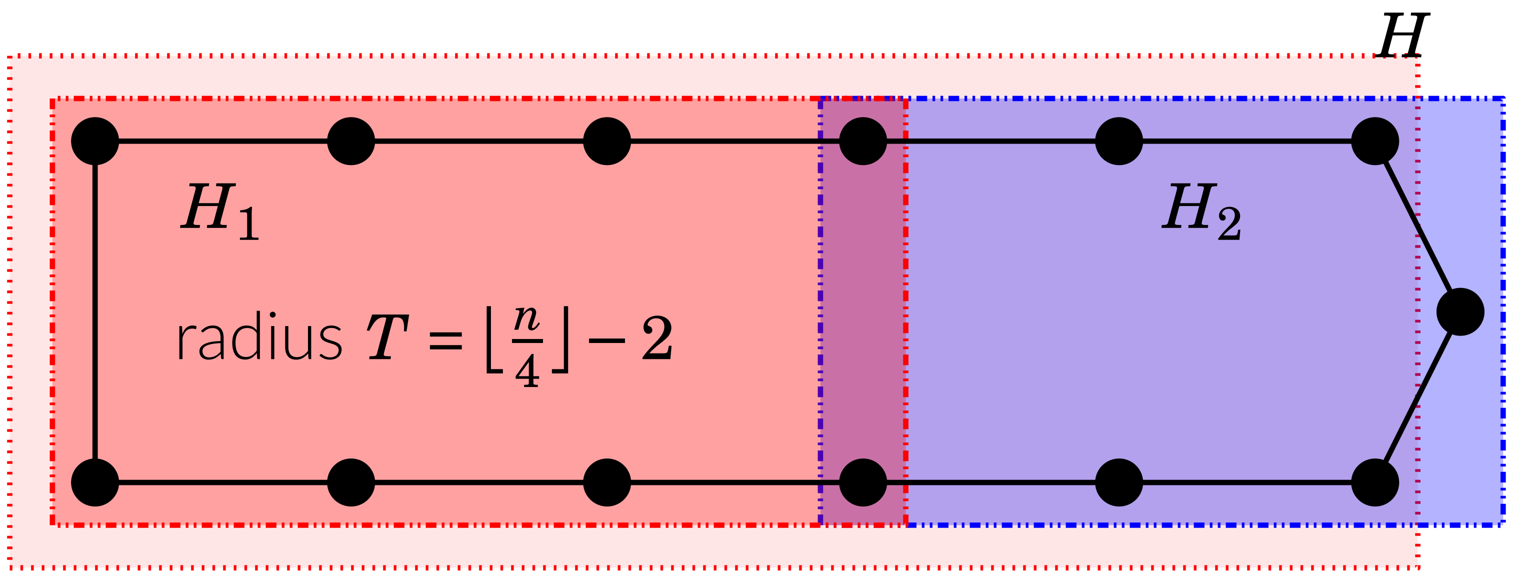
$\mathbf{max}_{i=1,2}\{\mathbf{Pr}[\text{failure on } H_i]\} \geq \frac{1}{2}$



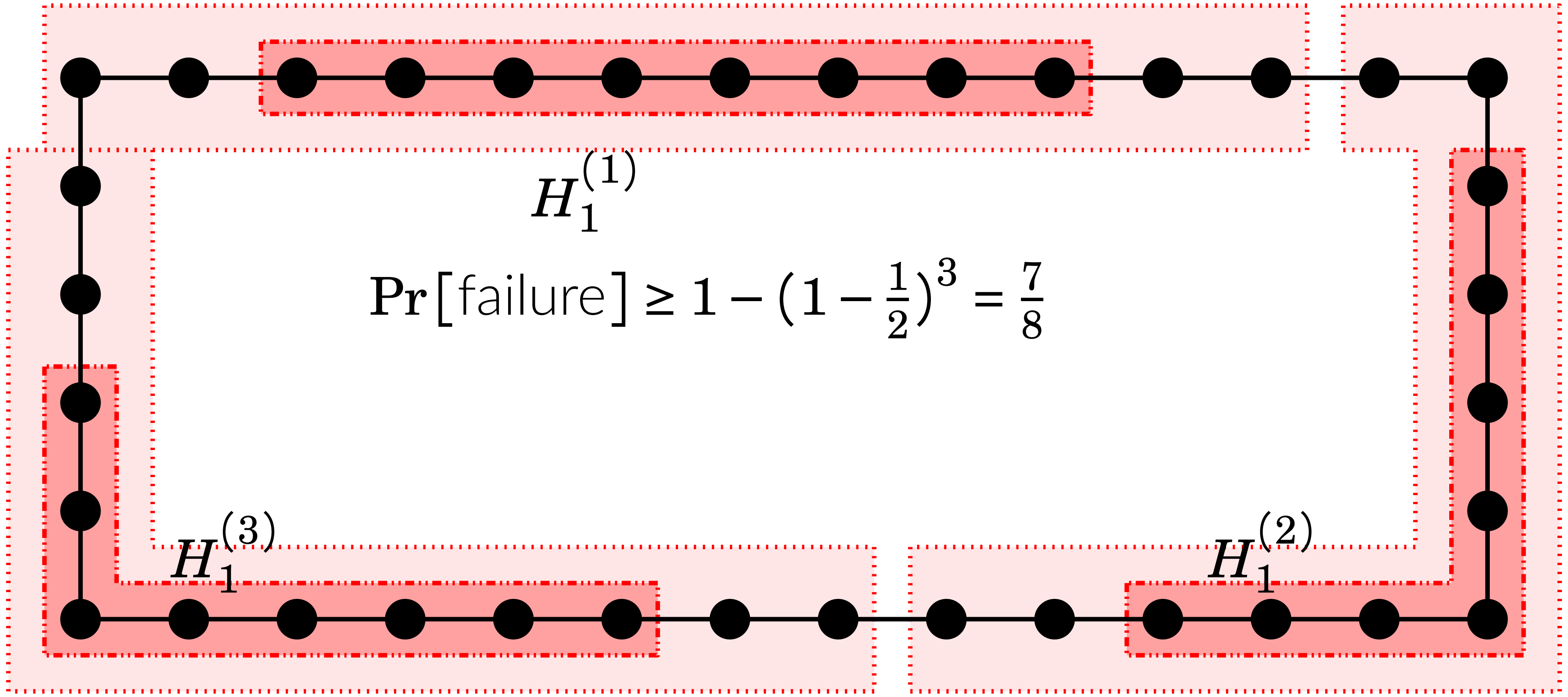
Boosting failure probability: randomized-LOCAL

Problem: 2-coloring 2-chromatic graphs

- randomized LOCAL
- "subdivide" graph H in regions H_1, H_2 such that
  - $H[\mathcal{N}_T(H_i)]$ is 2-colorable
  - N copies of $H[\mathcal{N}_T(H_i)]$ can be glued together to form a 2-colorable graph
- probability of failure $\geq 1-(1-\frac{1}{2})^N$
  - independence + union bound
  - cloning principle
- lower bound of $T(\frac{n}{N})$



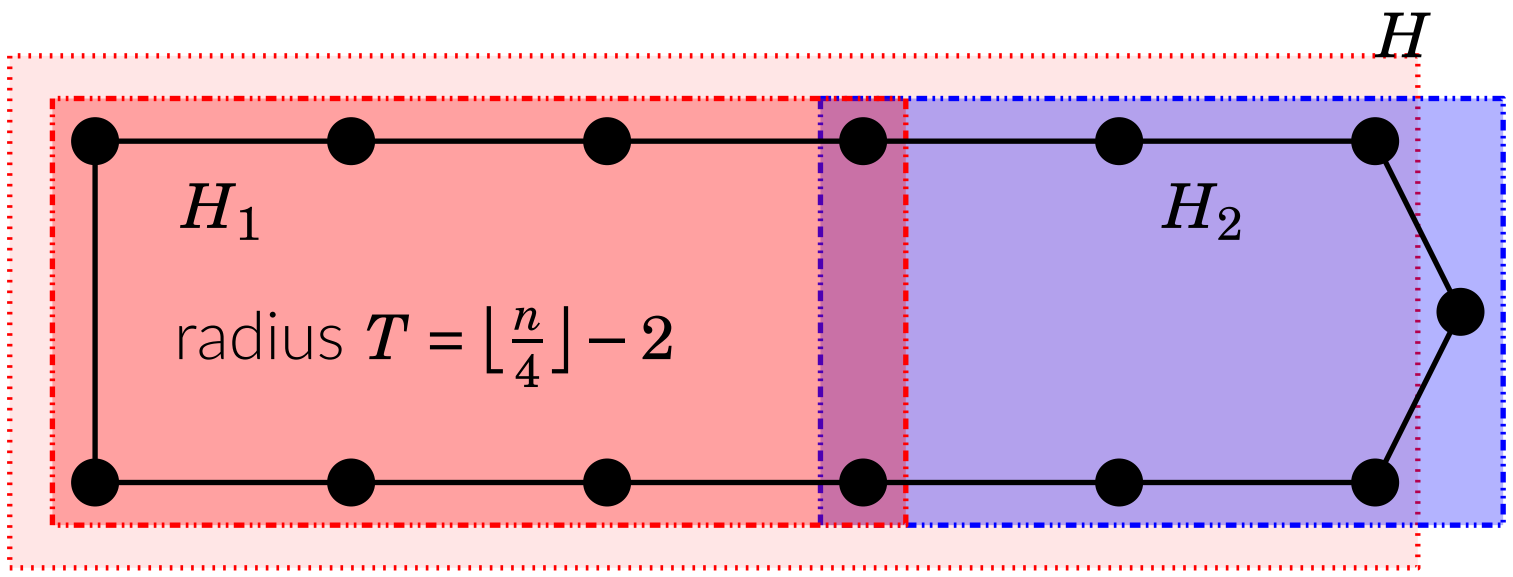
$\max_{i=1,2}\{\mathbf{Pr}[\text{failure on } H_i]\} \geq \frac{1}{2}$



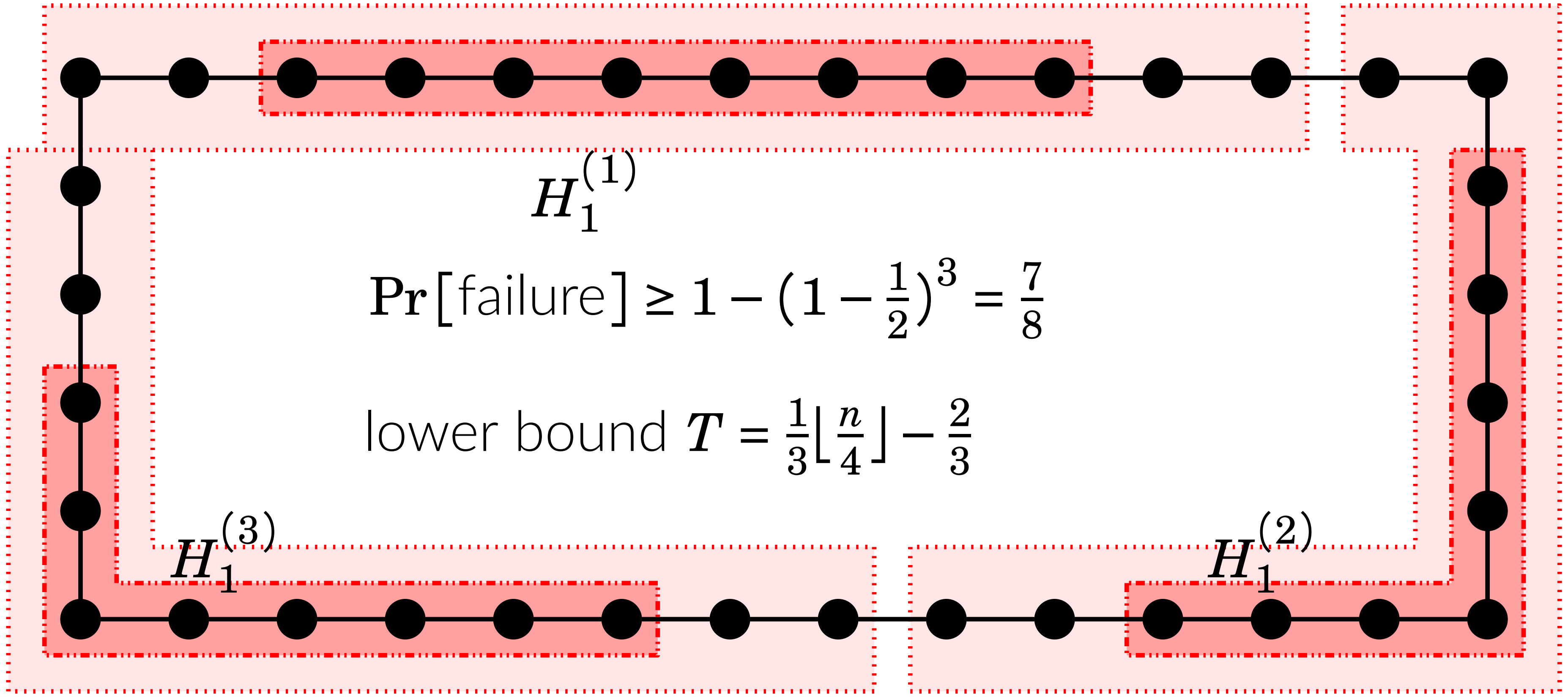
Boosting failure probability: randomized-LOCAL

Problem: 2-coloring 2-chromatic graphs

- randomized LOCAL
- "subdivide" graph H in regions H_1, H_2 such that
  - $H[\mathcal{N}_T(H_i)]$ is 2-colorable
  - N copies of $H[\mathcal{N}_T(H_i)]$ can be glued together to form a 2-colorable graph
- probability of failure $\geq 1 - (1 - \frac{1}{2})^N$
  - independence + union bound
  - cloning principle
- lower bound of $T(\frac{n}{N})$



$\max_{i=1,2}\{\mathbf{Pr}[\text{failure on } H_i]\} \geq \frac{1}{2}$



Problems in non-signaling: cloning and dependency

The **non-signaling** model:

- produces non-signaling outcomes

- outcome: function assigning to inputs (G, x) a distribution over labelings $\{(y_i, p_i)\}_{i \in I}$
- non-signaling beyond distance T

Problems in non-signaling: cloning and dependency

The **non-signaling** model:

- produces non-signaling outcomes

  - outcome: function assigning to inputs (G, x) a distribution over labelings $\{(y_i, p_i)\}_{i \in I}$
  - non-signaling beyond distance T

- **No-cloning principle**: no guarantees if we change the number of nodes

Problems in non-signaling: cloning and dependency

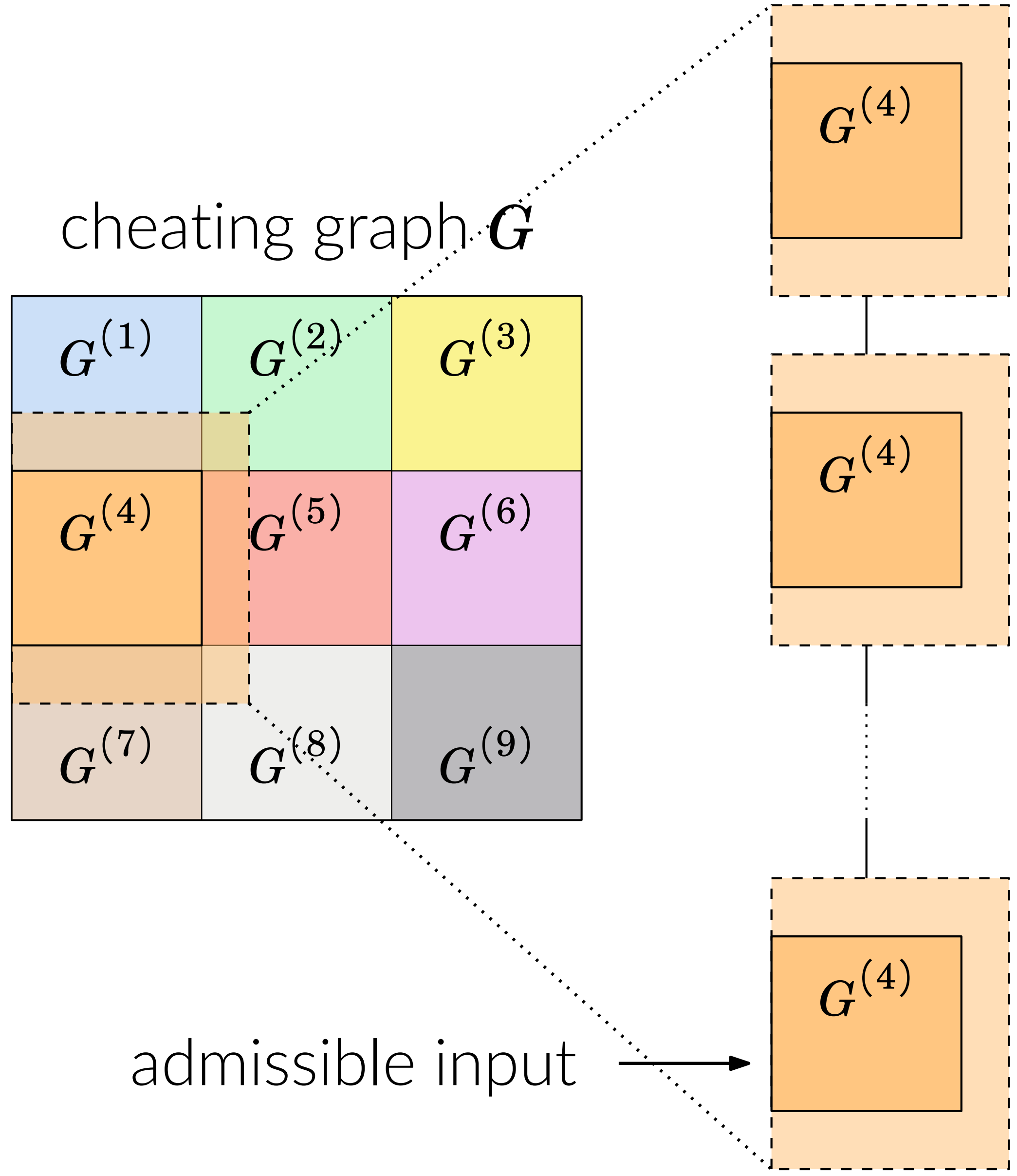
The **non-signaling** model:

- produces non-signaling outcomes
  - outcome: function assigning to inputs (G, x) a distribution over labelings $\{(y_i, p_i)\}_{i \in I}$
  - non-signaling beyond distance T
- **No-cloning principle**: no guarantees if we change the number of nodes
- **General dependencies**: no guarantee of independence between far apart regions of the graph

Problems in non-signaling: cloning and dependency

The **non-signaling** model:

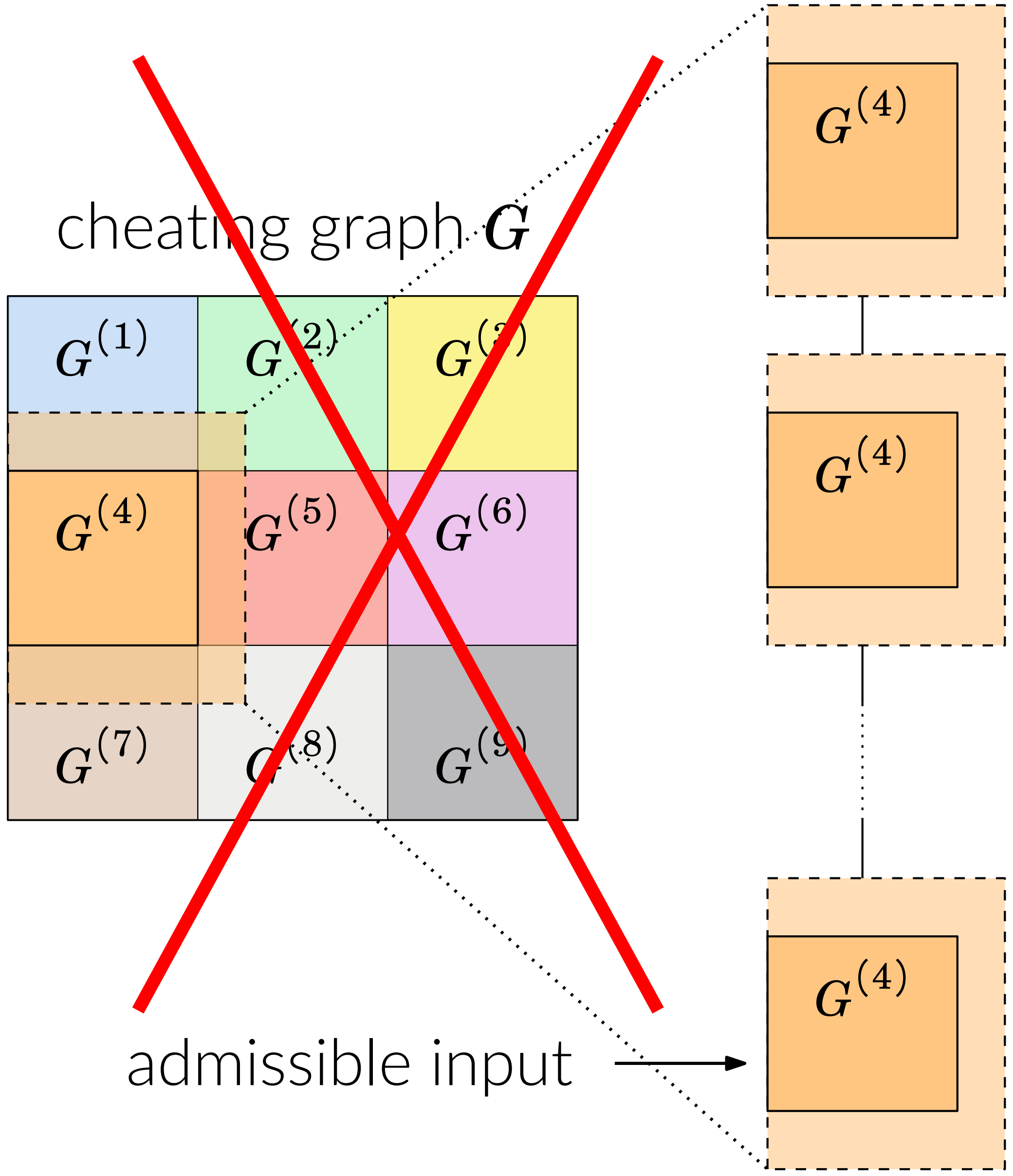
- produces non-signaling outcomes
  - outcome: function assigning to inputs (G, x) a distribution over labelings $\{(y_i, p_i)\}_{i \in I}$
  - non-signaling beyond distance T
- **No-cloning principle**: no guarantees if we change the number of nodes
- **General dependencies**: no guarantee of independence between far apart regions of the graph



Problems in non-signaling: cloning and dependency

The **non-signaling** model:

- produces non-signaling outcomes
  - outcome: function assigning to inputs (G, x) a distribution over labelings $\{(y_i, p_i)\}_{i \in I}$
  - non-signaling beyond distance T
- **No-cloning principle**: no guarantees if we change the number of nodes
- **General dependencies**: no guarantee of independence between far apart regions of the graph



Boosting failure probability in non-signaling

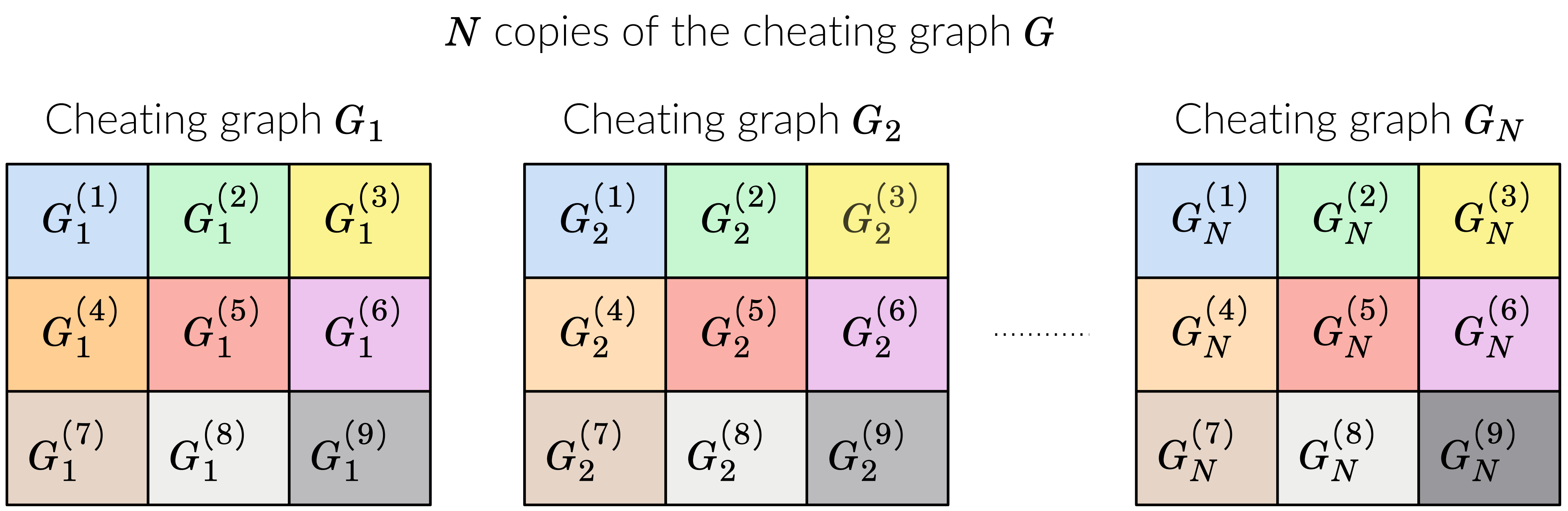
- Addressing no-cloning

- many copies of the cheating graph

Boosting failure probability in non-signaling

- Addressing no-cloning

  \- many copies of the cheating graph

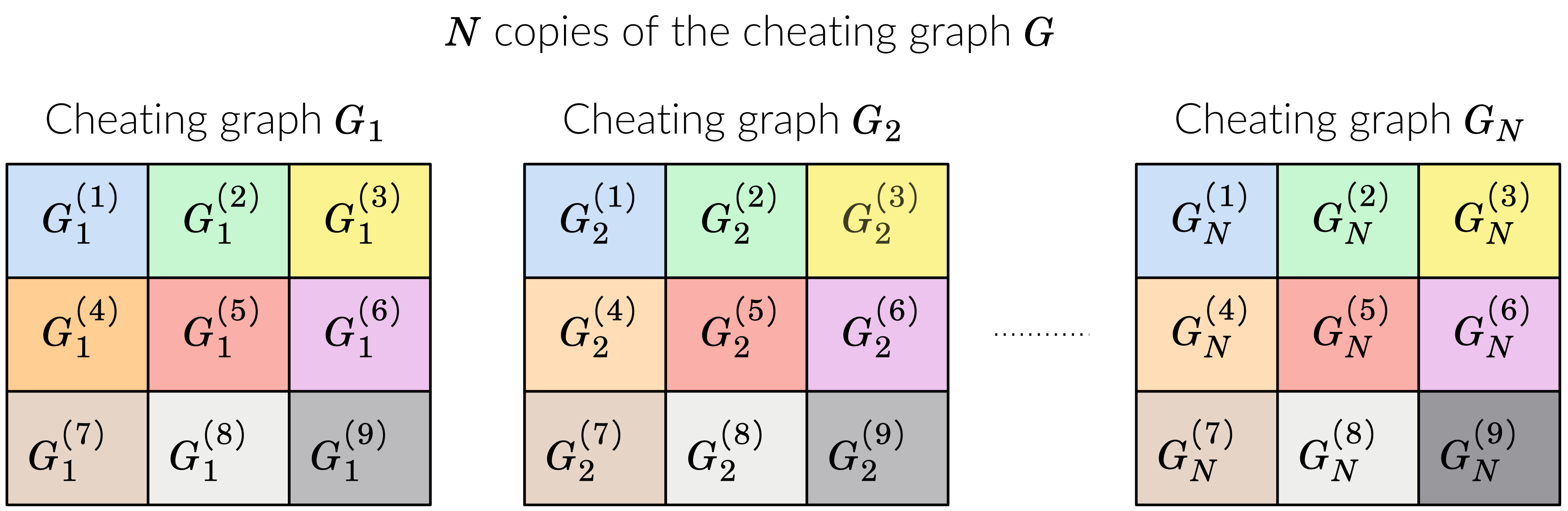


Boosting failure probability in non-signaling

- Addressing no-cloning

 - many copies of the cheating graph

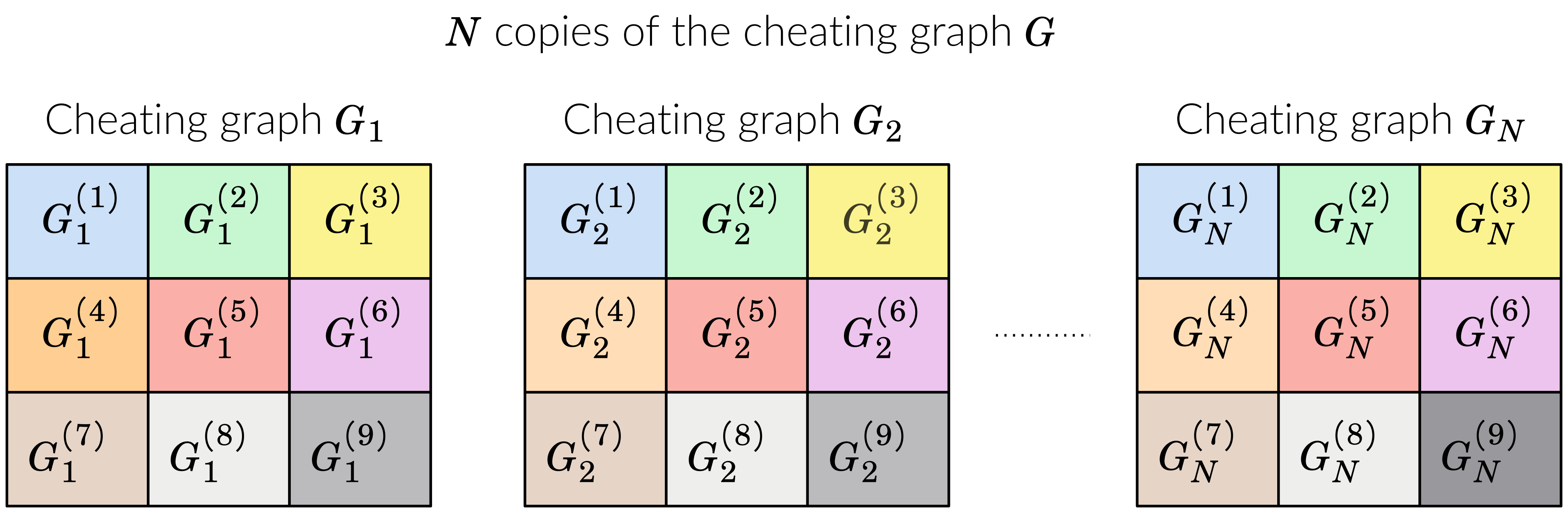
 - consider all nodes, “rearrange” edges



Boosting failure probability in non-signaling

- Addressing no-cloning
  - many copies of the cheating graph
  - consider all nodes, "rearrange" edges

- Addressing dependencies
  - consider joint probabilities



Boosting failure probability in non-signaling

- Addressing no-cloning

  - many copies of the cheating graph

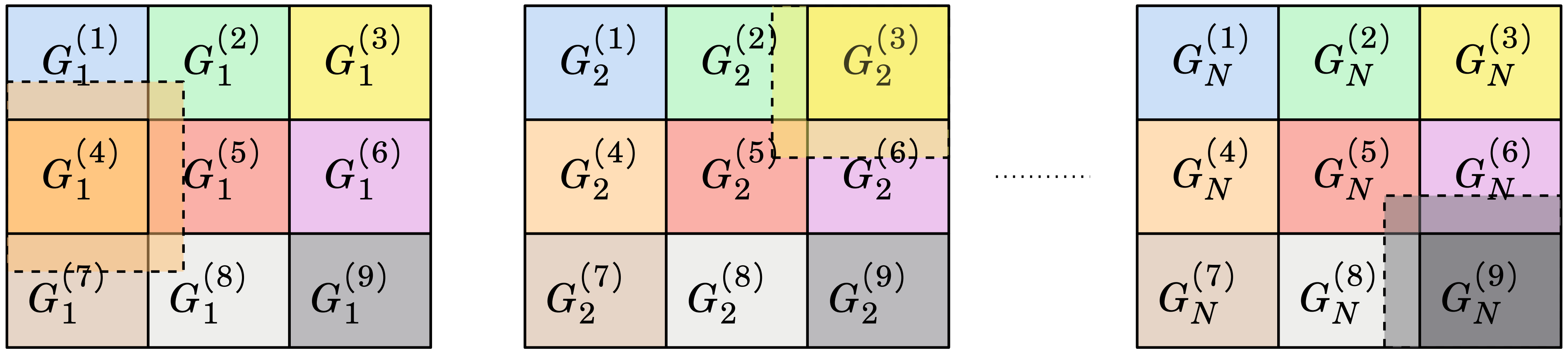
  - consider all nodes, “rearrange” edges

- Addressing dependencies

  - consider joint probabilities

N copies of the cheating graph G

Cheating graph G_1 Cheating graph G_2 Cheating graph G_N



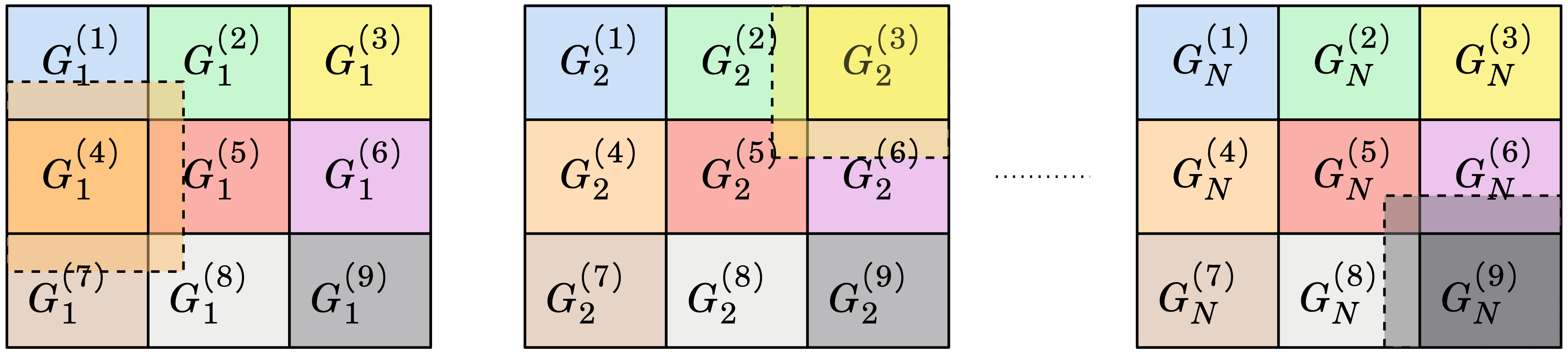
$\Pr(\mathcal{A}$ fails on $\cup_{j\in[N]} G_j^{(x_j)}) \geq 1-(1-1/k)^N$ for $\mathbf{x} = (4,3,\ldots,9)$

Boosting failure probability in non-signaling

- Addressing no-cloning
  - many copies of the cheating graph
  - consider all nodes, “rearrange” edges
- Addressing dependencies
  - consider joint probabilities
- Failure probability as in rand-LOCAL
  - different “bad event”

$\boldsymbol{N}$ copies of the cheating graph $\boldsymbol{G}$

Cheating graph $\boldsymbol{G_1}$ Cheating graph $\boldsymbol{G_2}$ Cheating graph $\boldsymbol{G_N}$



Pr($\mathcal{A}$ fails on $\cup_{j\in[N]} G_j^{(x_j)}$) $\geq 1-(1-1/k)^N$ for $\mathbf{x} = (4, 3, \ldots, 9)$

Boosting failure probability in non-signaling

- Addressing no-cloning
  - many copies of the cheating graph
  - consider all nodes, "rearrange" edges
- Addressing dependencies
  - consider joint probabilities
- Failure probability as in rand-LOCAL
  - different "bad event"