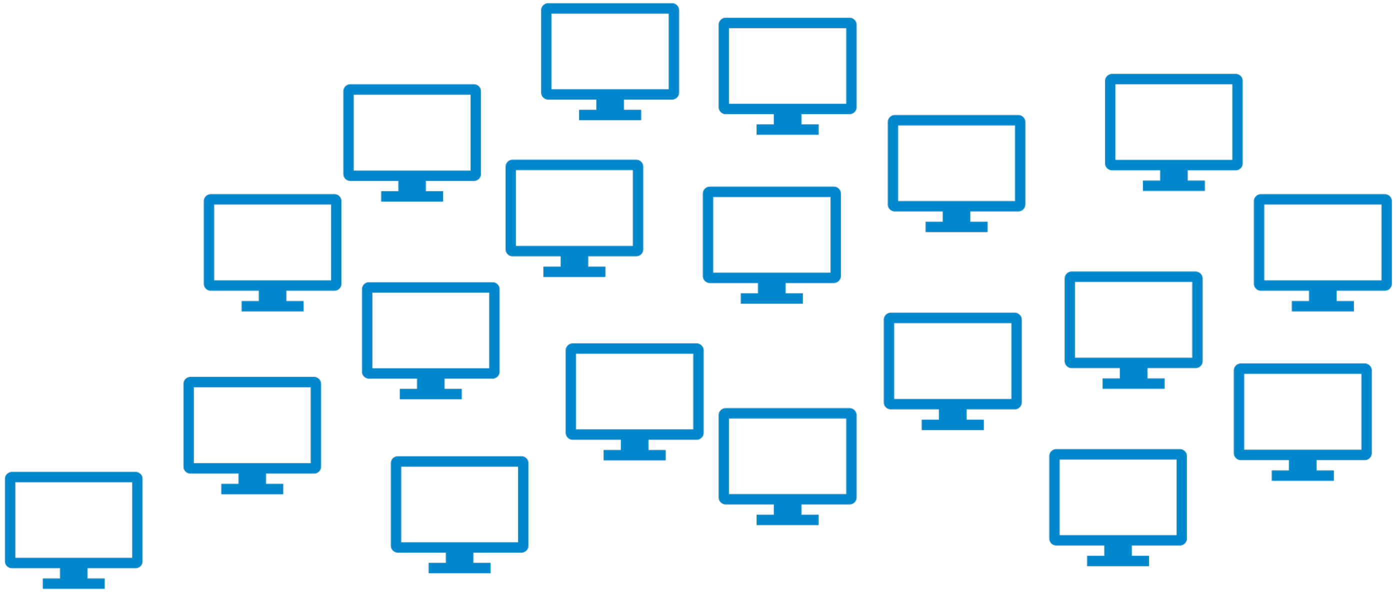
Foundations of Distributed Computing in the 2020s

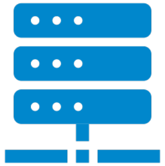
Jukka Suomela

Aalto University, Finland

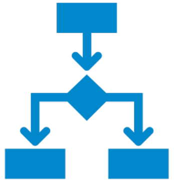


What are the theoretical foundations of the modern society?

- Modern world ≈ **large-scale communication networks**
- ***Physical side:***
  - **practice:** computers, network equipment, laser, fiber optics, radio …
  - **solid theoretical foundations:** electromagnetism, quantum mechanics …



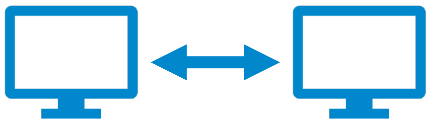
- ***Logical side:***
  - **practice:** communication protocols, networked applications …
  - **solid theoretical foundations:** ???



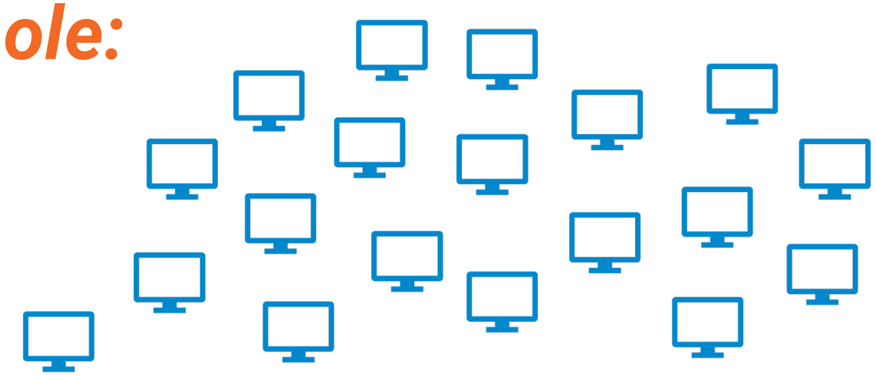
Logical foundations of large communication networks

- ***Computers:***
  - theory of computation, computability, computational complexity …

- ***Communication between computers:***
  - information theory, communication complexity theory …



- ***Computation in a network as a whole:***
  - theory of distributed computing



Our focus today

Logical foundations of computers vs. computer networks

- ***Theory of computation:***

Which tasks can be solved efficiently with a computer?

- ***Theory of distributed computing:***

Which tasks can be solved efficiently in a large computer network?

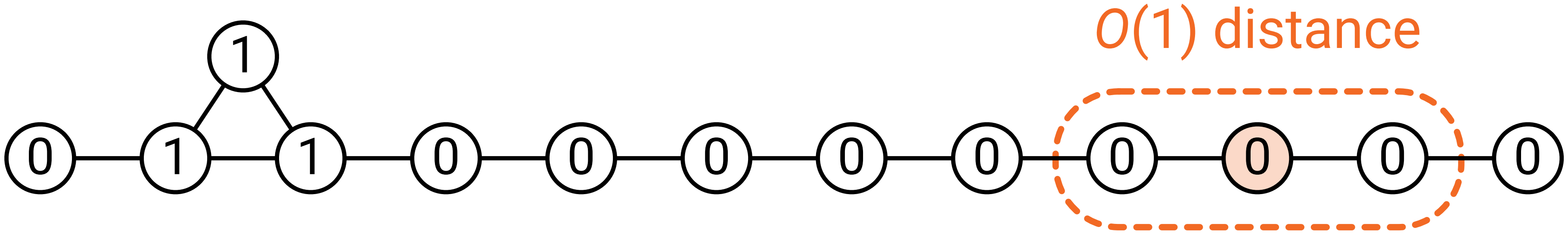
Logical foundations of computers vs. computer networks

- Example: **solving graph problems**
- ***Theory of computation:***

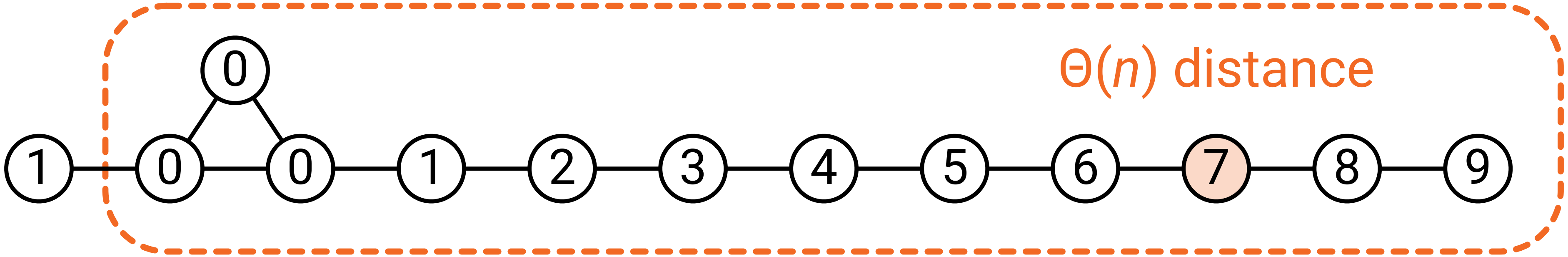
  - *"Here is a graph that is given as a string on a Turing machine tape"*
  - How many **steps** does a Turing machine need to solve this problem?
- ***Theory of distributed computing:***

  - *"I am a node in the middle of a very large graph"*
  - How **far** do I need to see to pick my own part of the solution?
  - How **much** of the graph do I need to see?
  - How many communication **rounds** are needed to solve the problem?

Local: am I part of a triangle?



Global: how far am I from the nearest triangle?

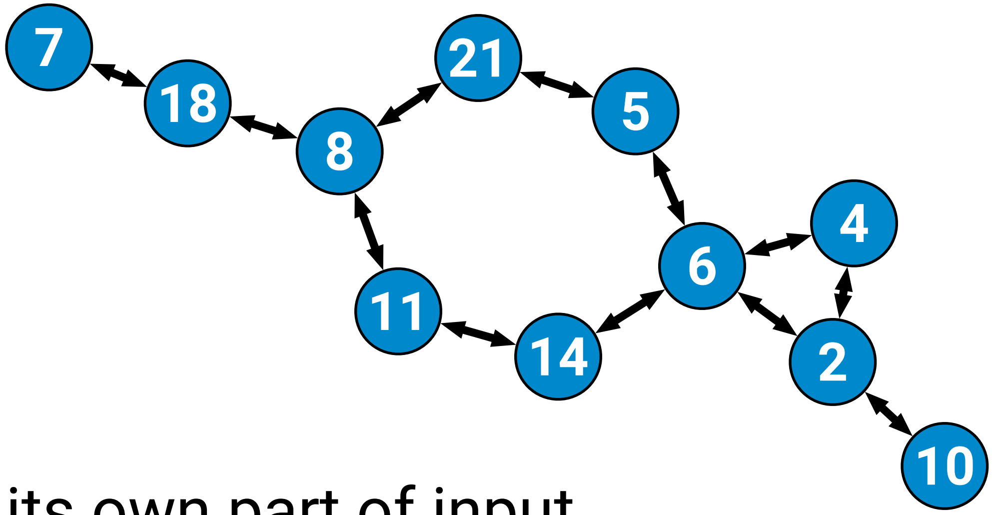


Logical foundations of computers vs. computer networks

- ***Theory of computation:***
  - e.g. hugely influential framework of **NP-completeness** (1970s)
- ***Theory of distributed computing:***
  - studied actively already since the 1980s
  - but we have only very recently started to really understand e.g. locality
  - solid theoretical foundations still largely missing
  - **lots of progress in the 2010s, tons of work left for the 2020s**

Distributed computing before the 2010s

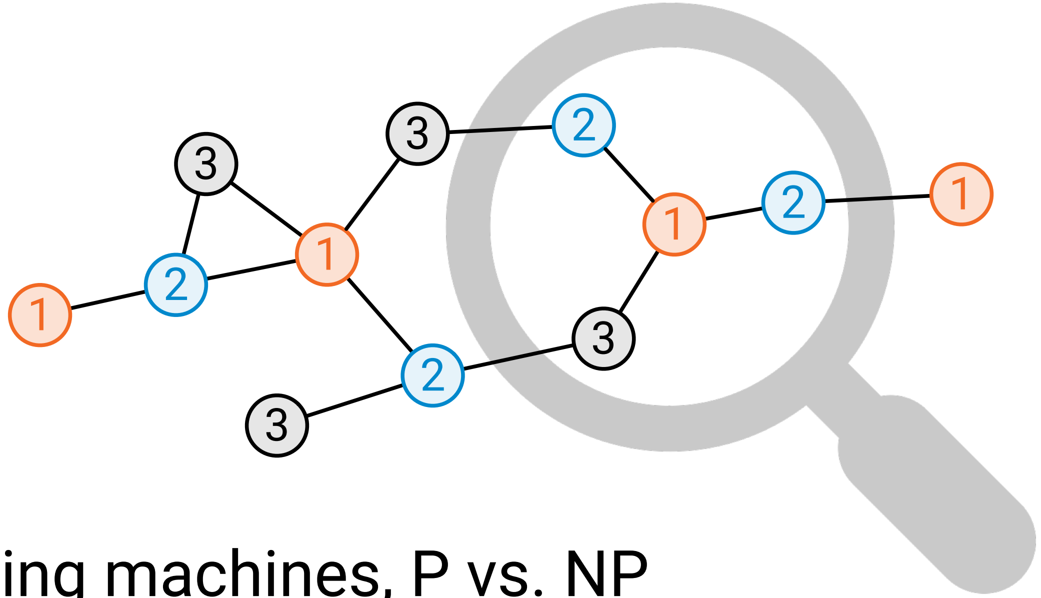
Standard models of computing



- **LOCAL model**
  - ***input graph = computer network***
  - **initially:** each node has a unique ID + its own part of input
  - **communication round:** ***each node sends a message to each neighbor***
  - **finally:** each nodes stops and outputs its own part of the solution
- **CONGEST model**
  - bounded-size messages
- **Port-numbering model**
  - no unique IDs

Number of rounds
= time = distance

Some important ideas and concepts



- ***Solving vs. checking***
  - finding a solution vs. verifying a solution
  - cf. deterministic vs. nondeterministic Turing machines, P vs. NP
- Problem family of "**locally checkable labelings**" (**LCL**s)
  - $O(1)$ input labels, $O(1)$ output labels, max degree $O(1)$
  - verification: check each radius-$O(1)$ neighborhood
  - Naor & Stockmeyer (1993, 1995)
- Proof labeling schemes
  - Korman, Kutten, Peleg (2005)

Example: vertex coloring with 3 colors

Some well-understood questions

- **What can be computed with deterministic algorithms in anonymous networks?**
  - e.g. Angluin (1980), Yamashita & Kameda (1996)
  - key technique: **covering maps**
- **Which LCL problems can be solved in constant time?**
  - e.g. Naor & Stockmeyer (1993, 1995)
  - key technique: **Ramsey theory**

Four key problems

maximal independent set	maximal matching
$(\Delta+1)$-vertex coloring	$(2\Delta-1)$-edge coloring

- ***Key primitives for symmetry breaking***
  - e.g. input is a symmetric cycle → output has to break symmetry
- ***Trivial linear-time centralized algorithms***
  - e.g. maximal matching: pick non-adjacent edges until stuck
- **Can we solve these efficiently in a distributed setting?**

Four key problems

maximal independent set	maximal matching
$(\Delta+1)$-vertex coloring	$(2\Delta-1)$-edge coloring

- ***Pioneering work on upper bounds:***
  - Cole & Vishkin (1986), Luby (1985, 1986), Alon, Babai, Itai (1986), Israeli & Itai (1986), Panconesi & Srinivasan (1996), Hanckowiak, Karonski, Panconesi (1998, 2001), Panconesi & Rizzi (2001) …
- ***Pioneering work on lower bounds:***
  - Linial (1987, 1992), Naor (1991), Kuhn, Moscibroda, Wattenhofer (2004)

Four key problems

maximal independent set	maximal matching
$(\Delta+1)$-vertex coloring	$(2\Delta-1)$-edge coloring

- Still wide gaps between upper and lower bounds
- Role of randomness poorly understood

It seems that before the 2010s…

- Lots of work focused on ***specific problems***
  - proving upper & lower bounds for problem *X*
  - connecting complexity of problem *X* through reductions to problem *Y*
- Not so much effort in understanding the ***overall landscape*** of distributed computational complexity
  - what are the meaningful **classes** of problems?
  - what can we prove about entire classes of problems?
- We were lacking ***general-purpose techniques*** for studying distributed computing

With hindsight…

- *Naor & Stockmeyer (1993, 1995)* introduced a very useful problem class (**LCL**s) and initiated the study of **decidability** of distributed complexity
  - but there was little follow-up work on these ideas until around ***2016***
- *Linial (1987, 1992)* already had the key idea behind **“round elimination”**
  - but it was not really recognized as a general-purpose proof technique until around ***2018***

Some highlights of distributed computing in the 2010s

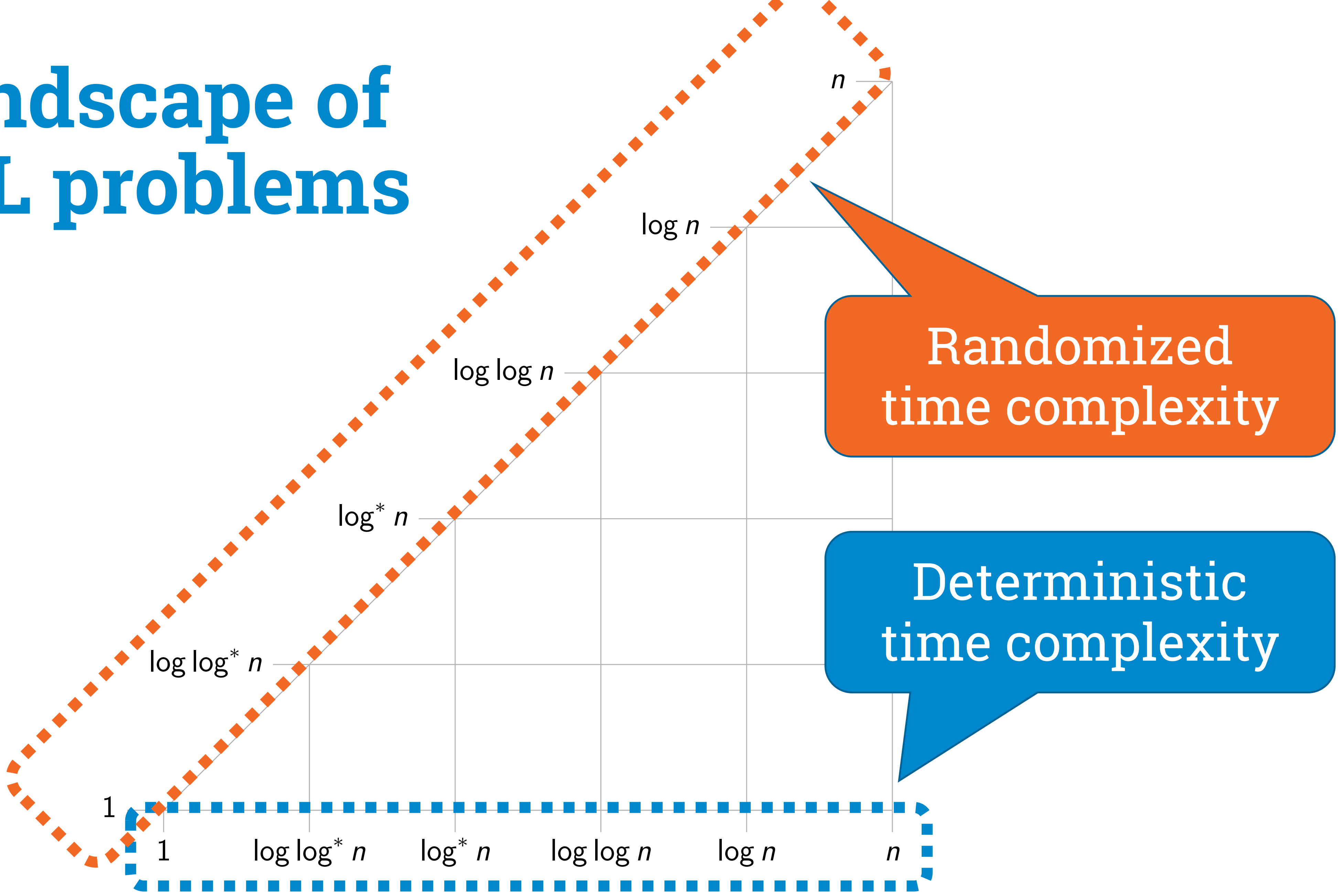
From the 2010s:
Classification of LCLs

LCL problems

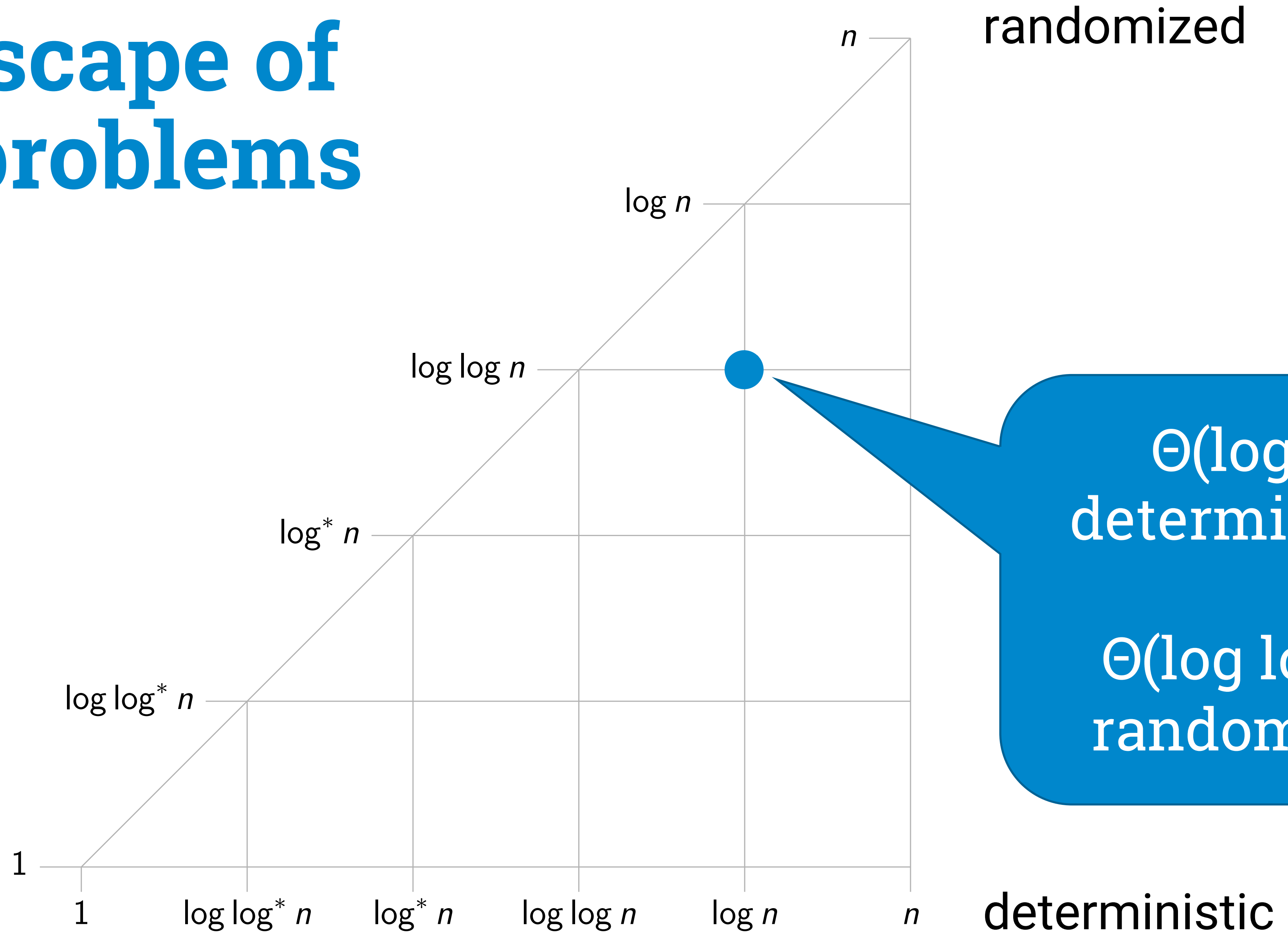
- Examples of LCL problems (in graphs of max degree $\Delta = O(1)$):
  - ***(Δ+1)-coloring***, ***Δ-coloring***, ***3-coloring*** …
  - ***maximal independent set***, ***maximal matching*** …
  - ***sinkless orientation***
    - orient all edges
    - all nodes of degree ≥ 3 have outdegree ≥ 1
  - ***locally optimal cut***
    - label nodes black/white
    - at least half of the neighbors have opposite color
  - ***SAT*** (when interpreted as a graph problem)
  - many other constraint satisfaction problems

Can we say something about all of these?

Landscape of LCL problems



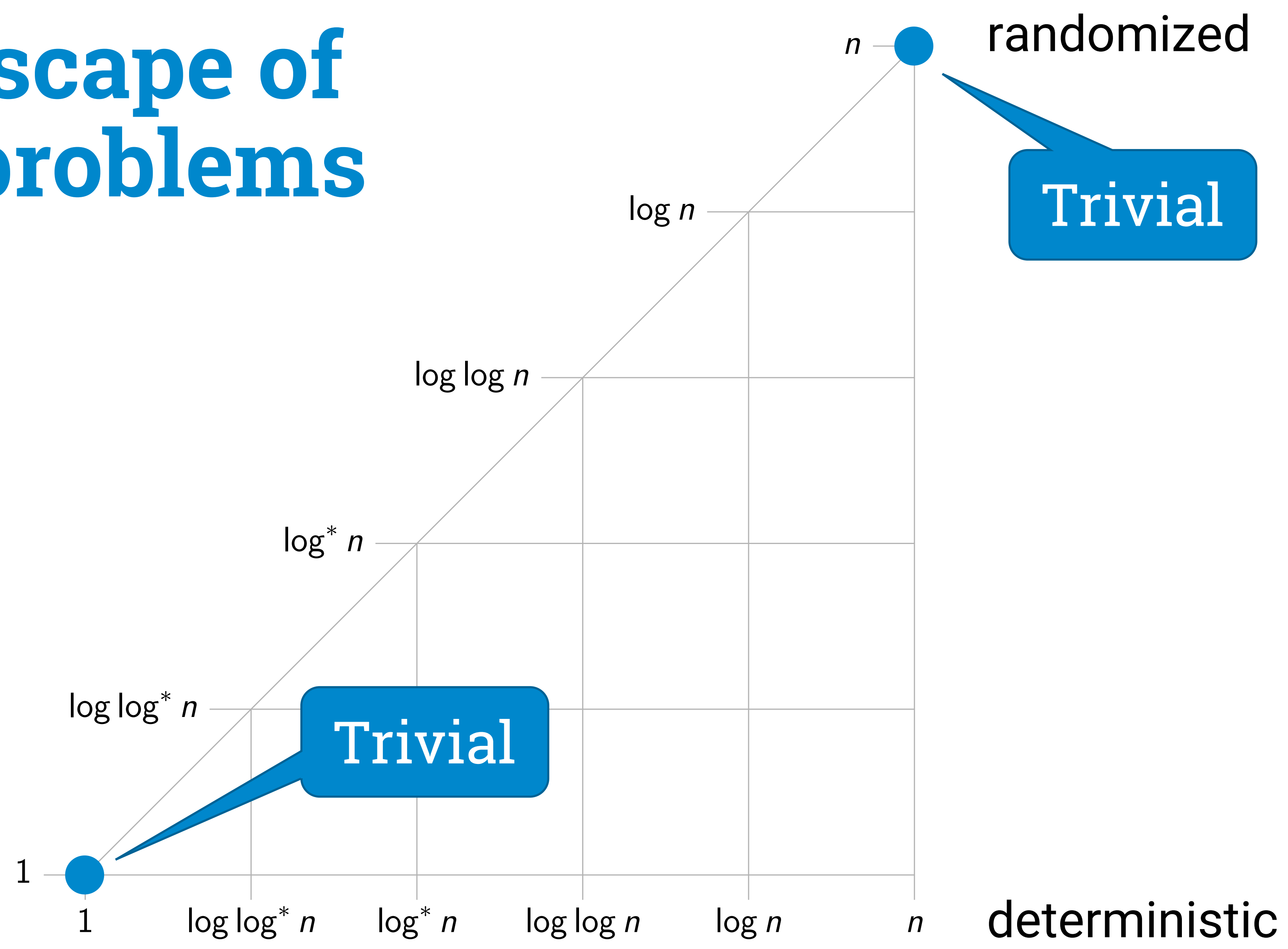
Landscape of LCL problems



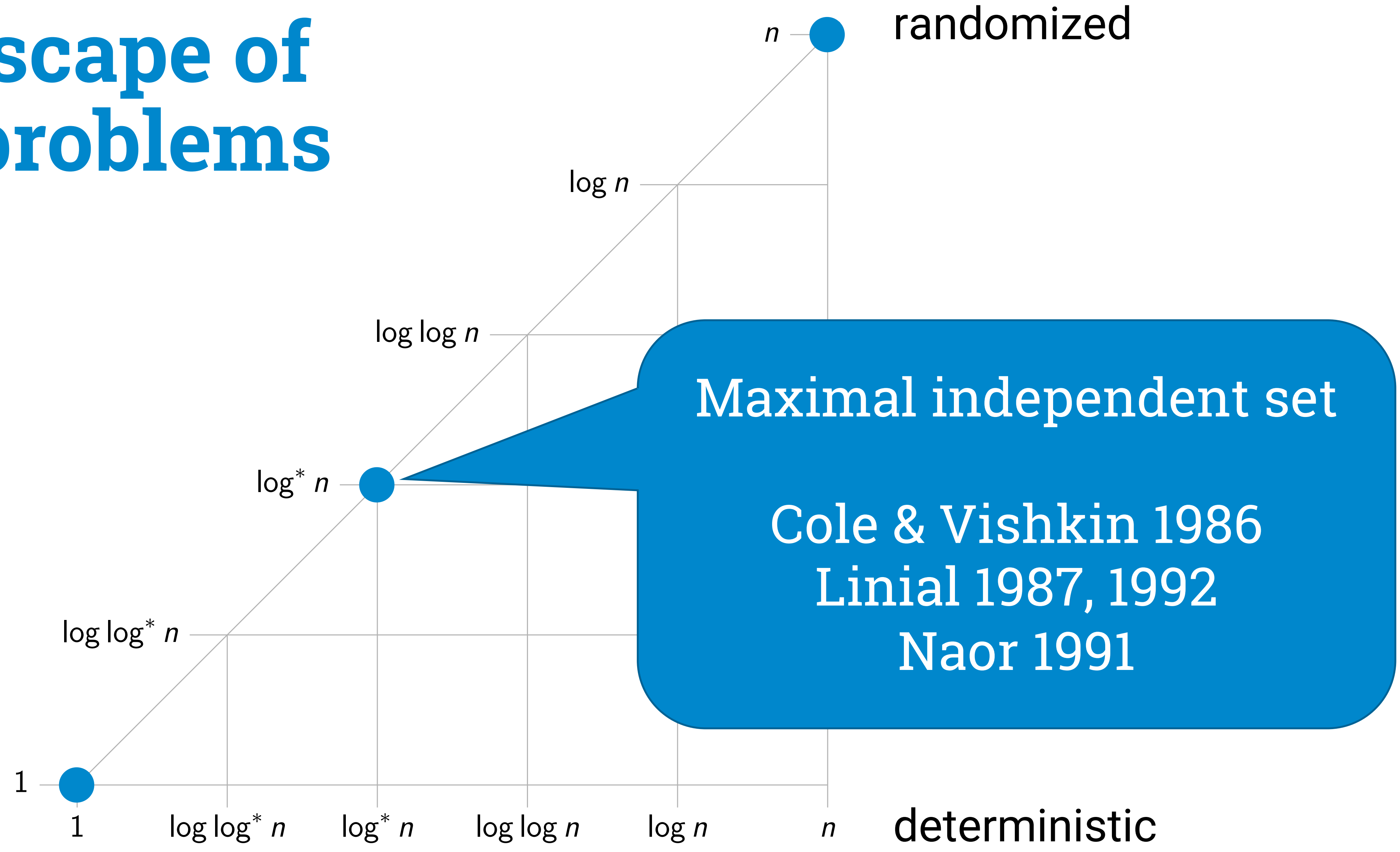
Θ(log n)
deterministic

Θ(log log n)
randomized

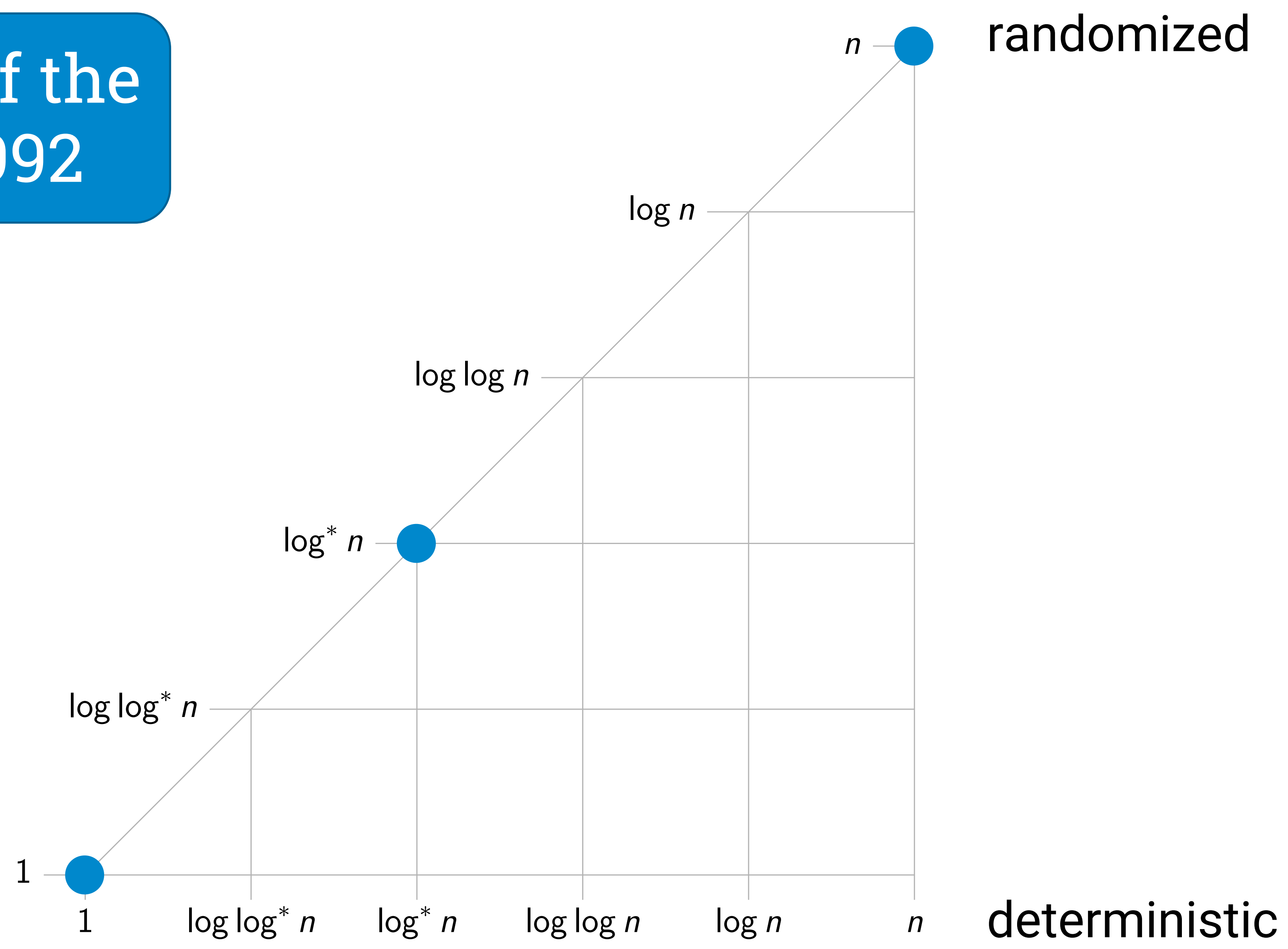
Landscape of LCL problems



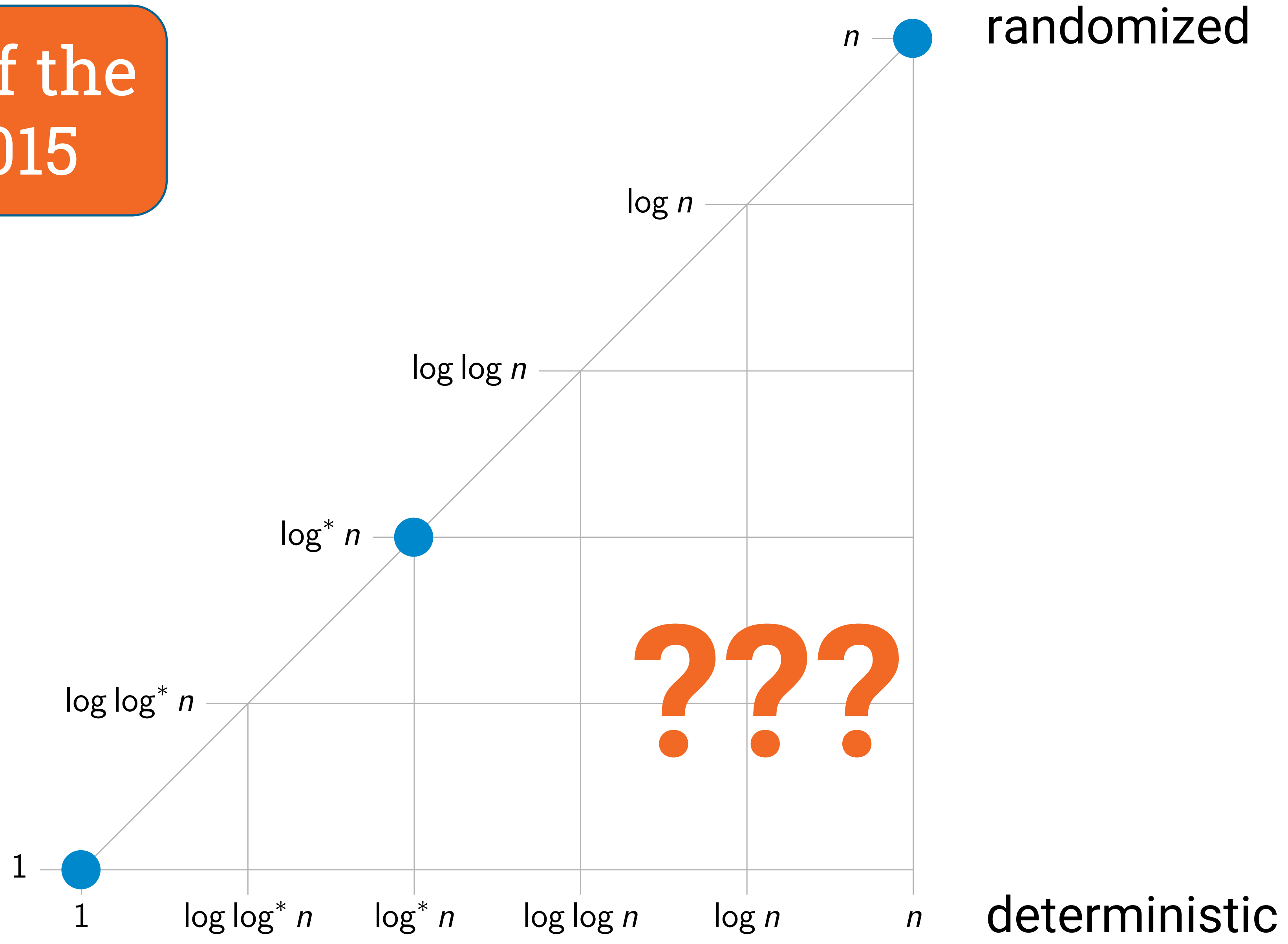
Landscape of LCL problems



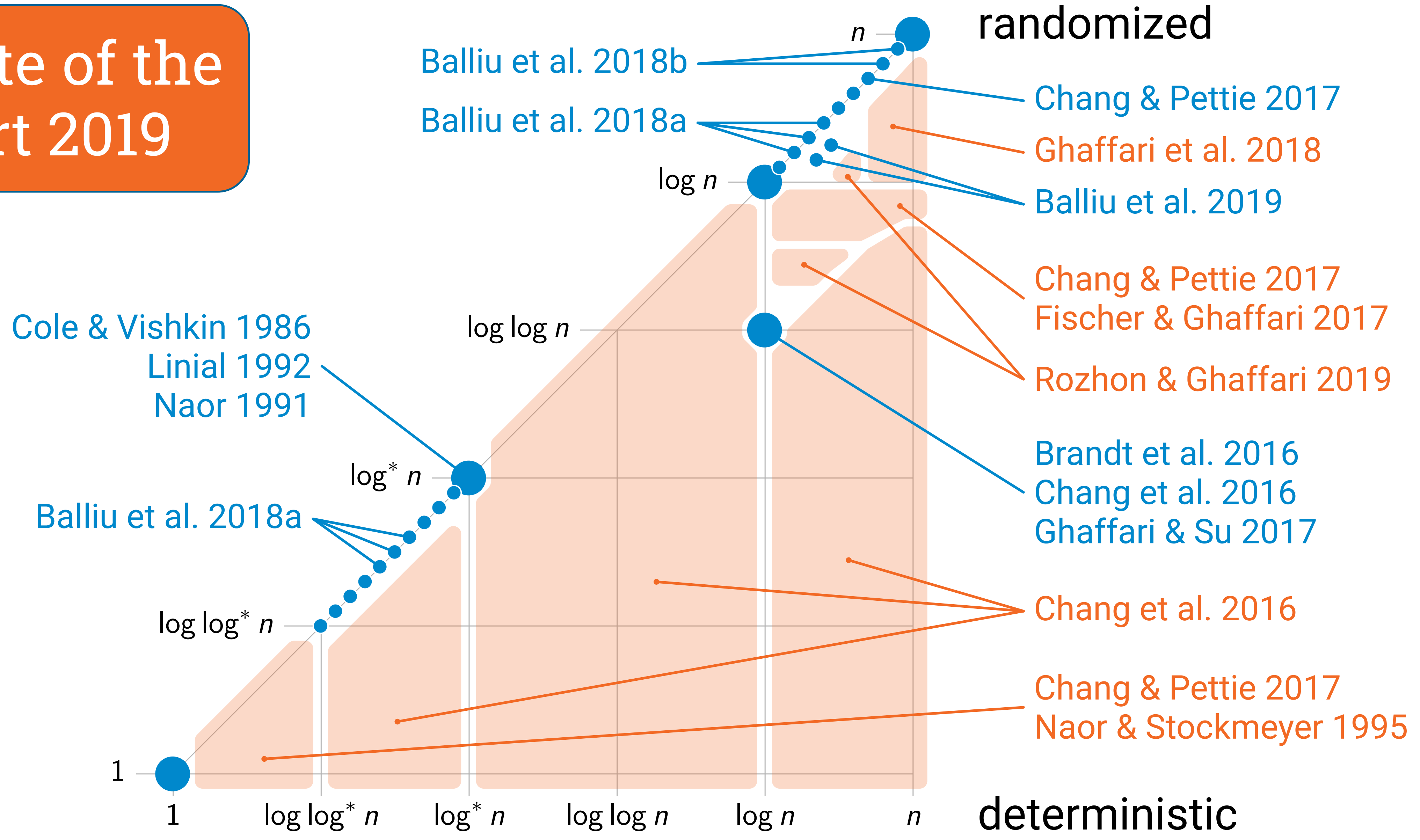
State of the art 1992



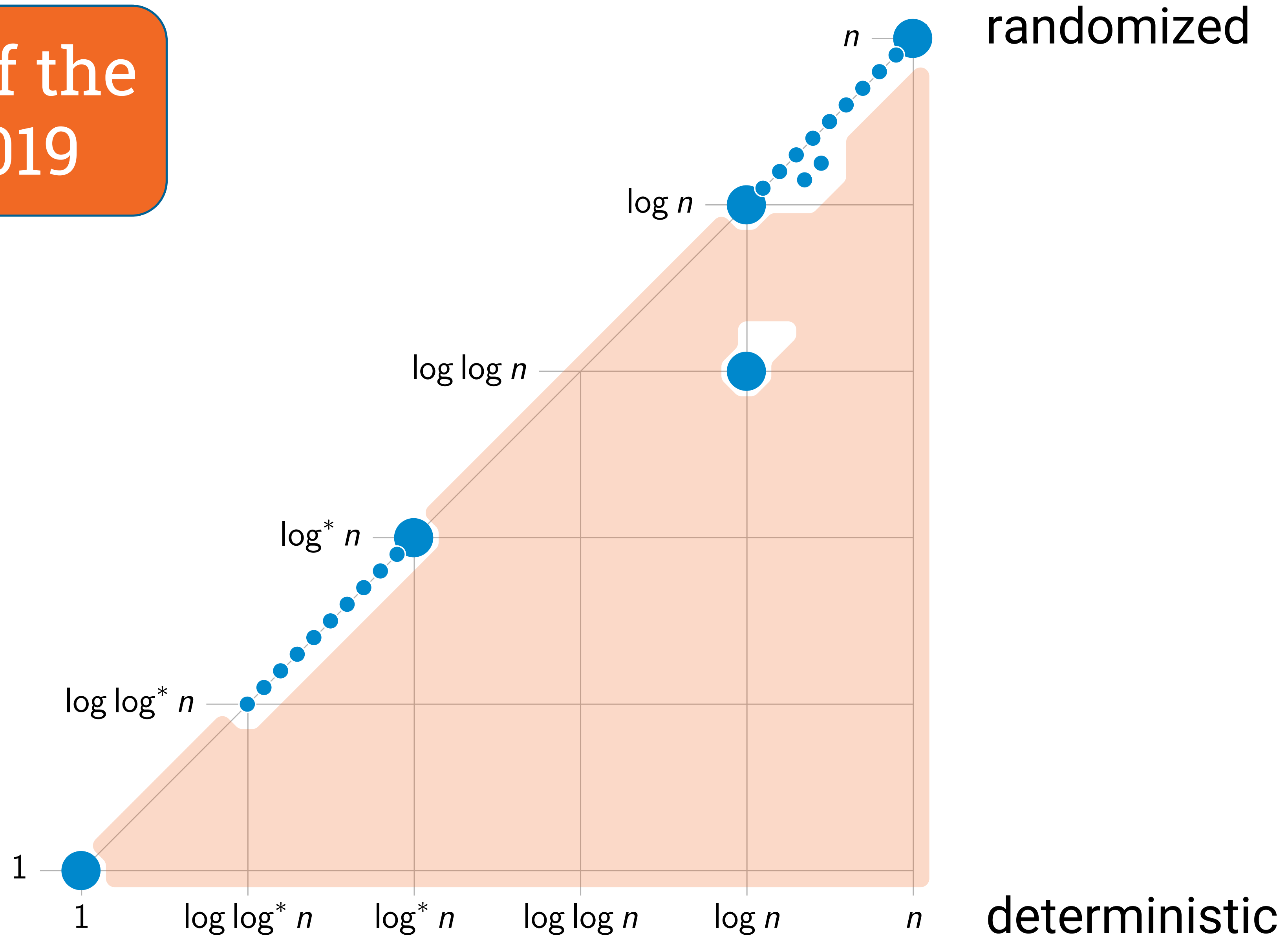
State of the art 2015



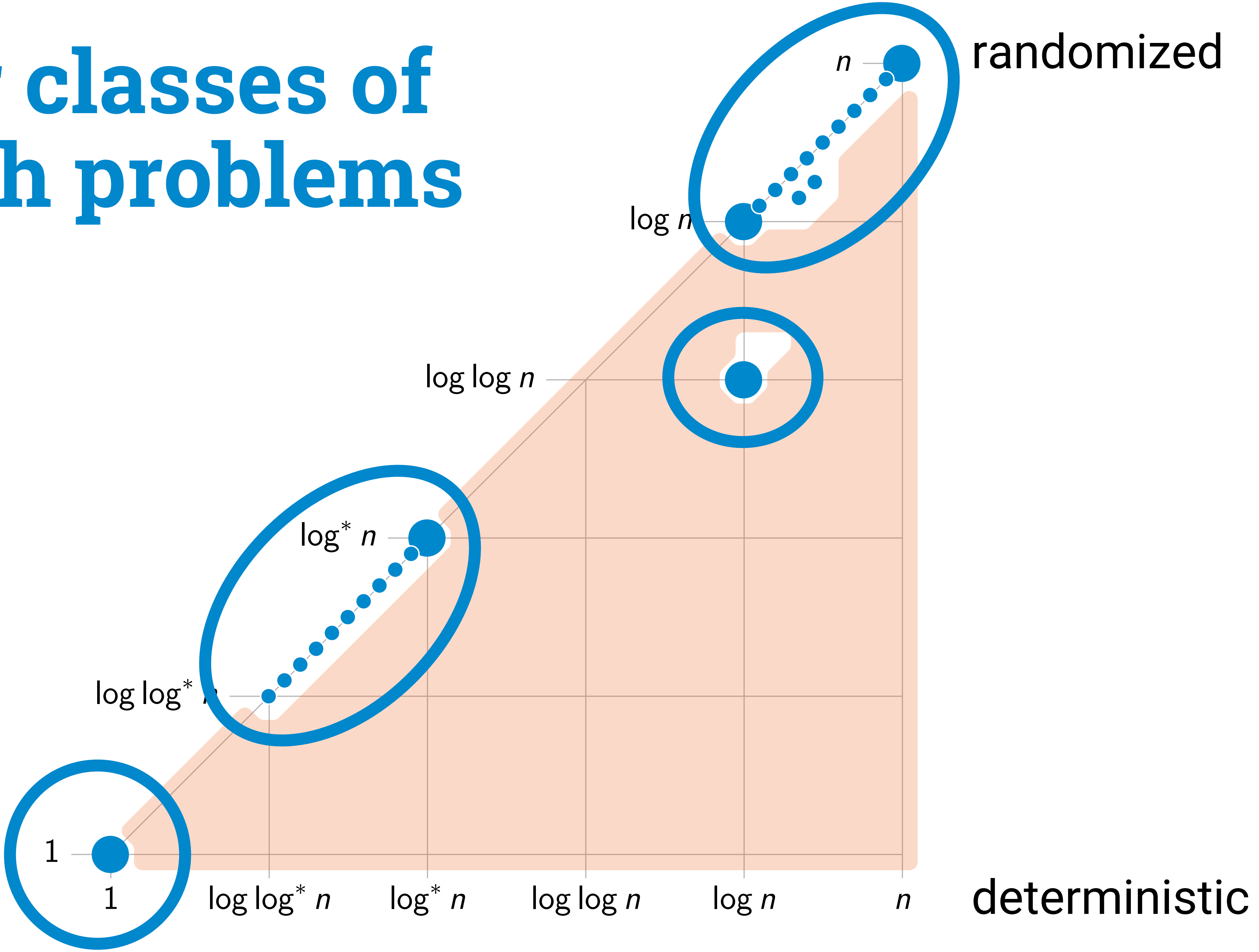
State of the art 2019



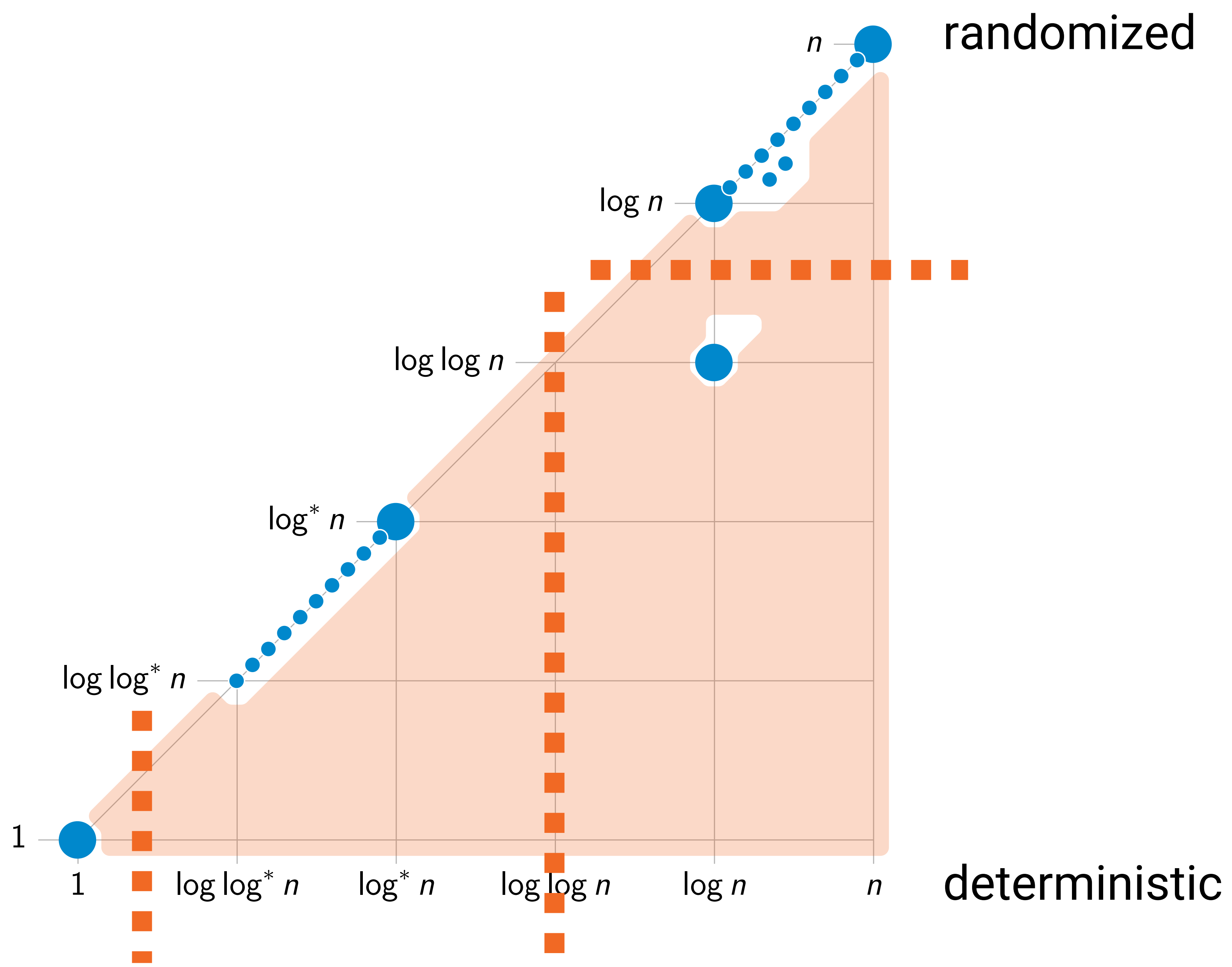
State of the art 2019



Four classes of graph problems



Gaps



randomized
n
log n
log log n
log* n
log log* n
1
1
log log* n
log* n
log log n
log n
n
deterministic

Gaps have direct algorithmic implications

If you can solve an LCL problem

- in $o(\log n)$ rounds with a ***deterministic*** algorithm ***or***
- in $o(\log \log n)$ rounds with a ***randomized*** algorithm

then you can also solve it

- in $O(\log^* n)$ rounds with a ***deterministic*** algorithms

Gaps have direct complexity-theoretic implications

If you can show that there is no $O(\log^* n)$-time deterministic algorithm then:

- *deterministic* complexity is at least $\Omega(\log n)$
- *randomized* complexity is at least $\Omega(\log \log n)$

From the 2010s:

Complexity of maximal independent set & maximal matching

2 of 4 key problems well understood

maximal independent set	maximal matching
$(\Delta+1)$-vertex coloring	$(2\Delta-1)$-edge coloring

- ***Maximal independent set & matching:***
  - deterministic $\boldsymbol{O(\Delta + \log^* n)}$
  - deterministic $\mathbf{poly}(\boldsymbol{\log n})$
  - randomized $\boldsymbol{O(\log \Delta)} + \mathbf{poly}(\boldsymbol{\log\log n})$
  - ***cannot improve any of these much***
- Upper bound: Rozhon & Ghaffari (2019) + many others
  - a new algorithm for **deterministic network decomposition**
- Lower bound: Balliu et al. (2019)
  - based on the **"round elimination"** technique

From the 2010s:
Round elimination technique

Round elimination technique

- **Given:**
  - algorithm $\boldsymbol{A_0}$ solves problem $\boldsymbol{P_0}$ in T rounds
- **We construct:**
  - algorithm $\boldsymbol{A_1}$ solves problem $\boldsymbol{P_1}$ in $T - 1$ rounds
  - algorithm $\boldsymbol{A_2}$ solves problem $\boldsymbol{P_2}$ in $T - 2$ rounds
  - algorithm $\boldsymbol{A_3}$ solves problem $\boldsymbol{P_3}$ in $T - 3$ rounds

    …
  - algorithm $\boldsymbol{A_T}$ solves problem $\boldsymbol{P_T}$ in **0** rounds
- But $\boldsymbol{P_T}$ is nontrivial, so $\boldsymbol{A_0}$ cannot exist

Linial (1987, 1992): coloring cycles

- **Given:**
  - algorithm A_0 solves 3-coloring in $T = o(\log^* n)$ rounds
- **We construct:**
  - algorithm A_1 solves 2^3-coloring in $T - 1$ rounds
  - algorithm A_2 solves 2^{2^3}-coloring in $T - 2$ rounds
  - algorithm A_3 solves $2^{2^{2^3}}$-coloring in $T - 3$ rounds

  …
  - algorithm A_T solves $o(n)$-coloring in 0 rounds
- But $o(n)$-coloring is nontrivial, so A_0 cannot exist

Brandt et al. (2016): sinkless orientation

- **Given:**
  - algorithm A_0 solves **sinkless orientation** in $T = o(\log n)$ rounds
- **We construct:**
  - algorithm A_1 solves **sinkless coloring** in $T - 1$ rounds
  - algorithm A_2 solves **sinkless orientation** in $T - 2$ rounds
  - algorithm A_3 solves **sinkless coloring** in $T - 3$ rounds

    …
  - algorithm A_T solves **sinkless orientation** in 0 rounds
- But **sinkless orientation** is nontrivial, so A_0 cannot exist

Round elimination can be automated

Brandt 2019

- Always possible for **any graph problem P_0** that is locally checkable
- If problem P_0 has complexity T, we can always find in a mechanical manner problem P_1 that has complexity $T - 1$
- Holds for tree-like neighborhoods (e.g. high-girth graphs)
- Can be used to derive **lower bounds** and to **design algorithms**

From the 2010s:
Using computers to study distributed computing

Using computers to do study distributed computing

- Many questions related to distributed computational complexity have turned out to be ***decidable*** or semi-decidable
  - at least in principle, and often also in practice
  - we can start to ***automate our own work*** and outsource algorithm design & lower bound construction to computers
- Automatic round elimination implemented, available online: *github.com/olidennis/round-eliminator* (Olivetti 2019)
  - in 2016 a lower bound for “sinkless orientation” was a STOC paper
  - in 2019 you can reproduce it in your web browser

Distributed computing in the 2020s

Distributed complexity theory beyond LCLs

- We can nowadays say a lot about LCL problems:
  - near-complete classification of distributed complexity
  - systematic studies, powerful proof techniques, automatic tools
- ***How could we extend all this to non-LCLs?***
- Small first steps for the coming years:
  - locally checkable problems with **unbounded degrees**?
  - locally checkable problems with **countably many labels**?
  - locally checkable problems with **real numbers** and linear constraints?
  - **optimization problems** with locally checkable constraints?

Four key problems

maximal independent set	maximal matching
(Δ+1)-vertex coloring	(2Δ−1)-edge coloring

- Independent sets & matchings: now well understood
- ***Coloring:*** distributed complexity still wide open
- “Small” first step for the coming years:
    - show that (Δ+1)-vertex coloring cannot be solved in $o(\log \Delta) + O(\log^* n)$ rounds

Two perspectives of distributed computing

Network algorithms

- Solving problems related to the network structure
- Example: network protocols
- Key limitation: **long distances**
- No centralized control
- Local perspective

Big data

- Solving large computational tasks with many computers
- Example: MapReduce
- Key limitation: **bandwidth**
- Fully centralized control
- Global perspective

Two perspectives of distributed computing

Unifying models?

Network algorithms

- LOCAL
- CONGEST

Big data

- PRAM
- MPC = Massively Parallel Computation
- BSP = Bulk-Synchronous Parallel
- Congested clique

Two perspectives of distributed computing

Technology transfer?

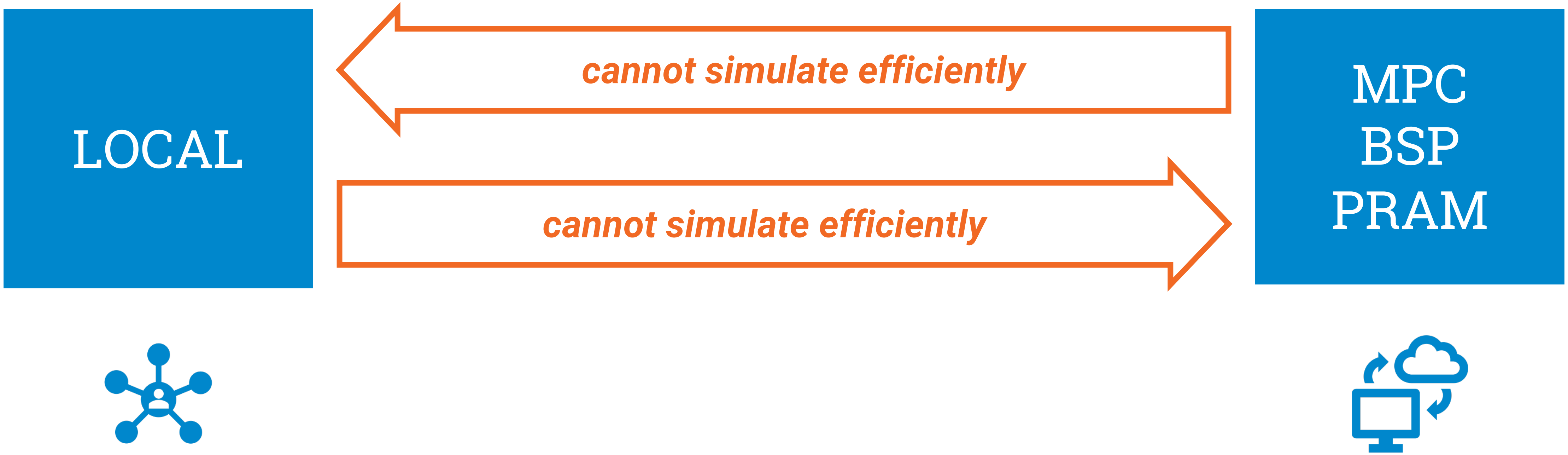
Network algorithms

- tight unconditional lower bounds for many problems

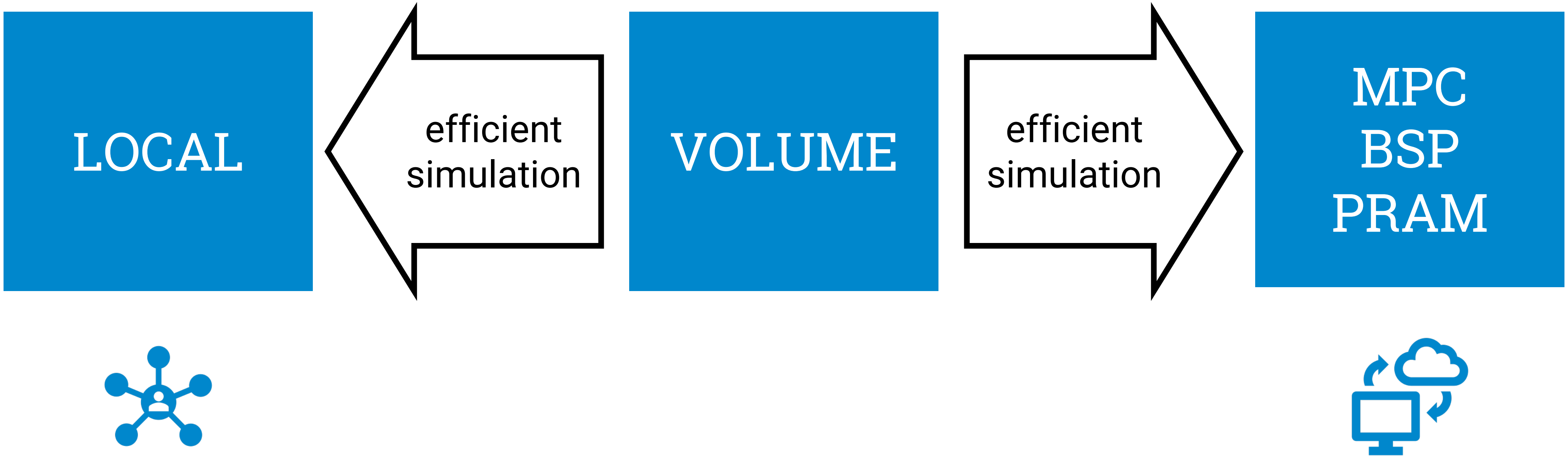
Big data

- typically at best conditional lower bounds

Two perspectives of distributed computing

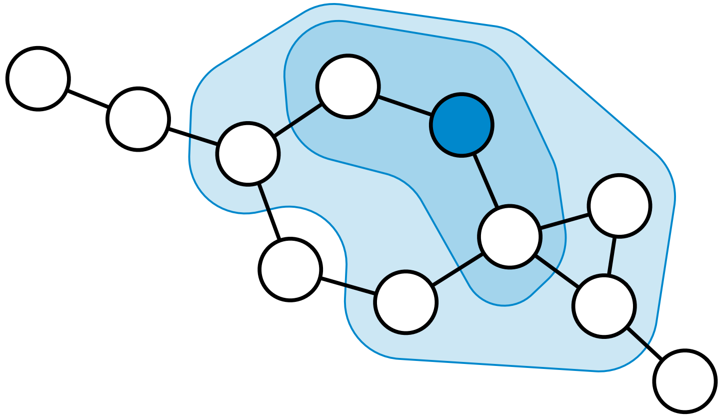


Two perspectives of distributed computing

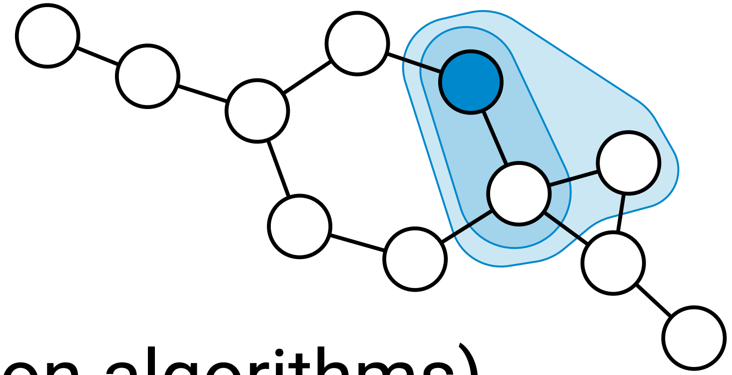


Volume model

- **Time *T* in LOCAL model:**
  - each node can explore a subgraph of **radius *T*** around it and then choose its output



- **Time *T* in VOLUME model:**
  - each node can **adaptively** explore a subgraph of **size *T*** around it and then choose its output



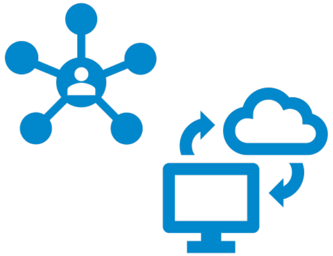
- Closely related model: ***LCA*** (local computation algorithms), a.k.a. centralized LOCAL algorithms or CentLOCAL

Volume model

- Bridge between two flavors of distributed computing
- Close enough to LOCAL so that it is possible to prove unconditional lower bounds
- Yet ***poorly understood***: typically exponential gaps between upper and lower bounds
- Not-so-small first steps:
  - charting the **landscape of LCL problems** in the volume model
  - tight bounds for e.g. **sinkless orientation, maximal matching** …
  - volume analogue of **round elimination**

Summary

- **2010s:**
  - systematic study of LCL problems in the LOCAL model
  - new techniques and automatic tools
- **2020s:**
  - extending theory ***beyond LCLs***
  - technology transfer ***LOCAL*** → ***VOLUME*** → MPC, PRAM, …



- **Small puzzles to solve:**
  - show that **$O(\Delta)$ volume** is not enough for bipartite maximal matching
  - construct an LCL problem with deterministic volume **$\omega(\log^* n)$ … $o(n)$**