Jukka Suomela
Aalto University

Locality

in online, dynamic, sequential, and distributed graph algorithms

Joint work with:

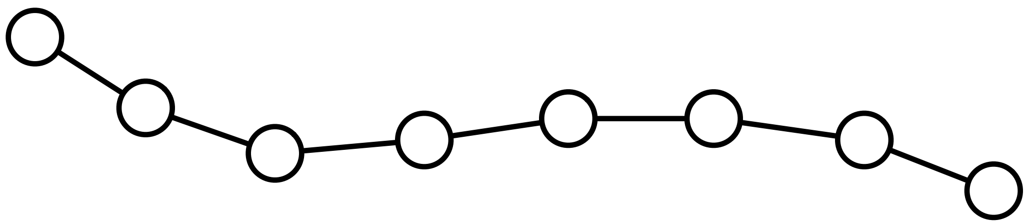
- Amirreza Akbari
- Navid Eslami
- Henrik Lievonen
- Darya Melnyk
- Joona Särkijärvi

arxiv.org/abs/2109.06593

Informal introduction to locality

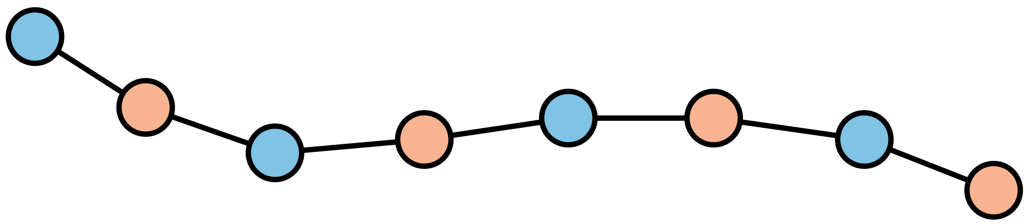
Coloring paths locally?

- **Given:** a path graph



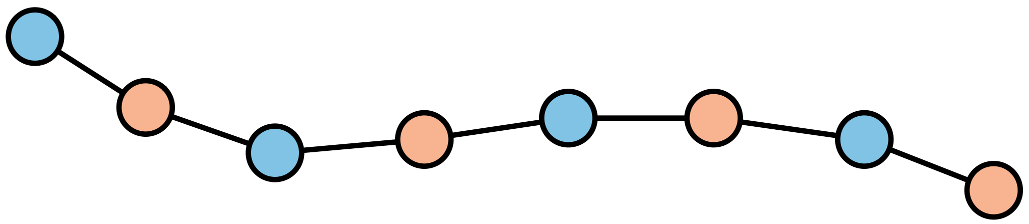
Coloring paths locally?

- **Given:** a path graph
- **Find:** a proper 2-coloring



Coloring paths locally?

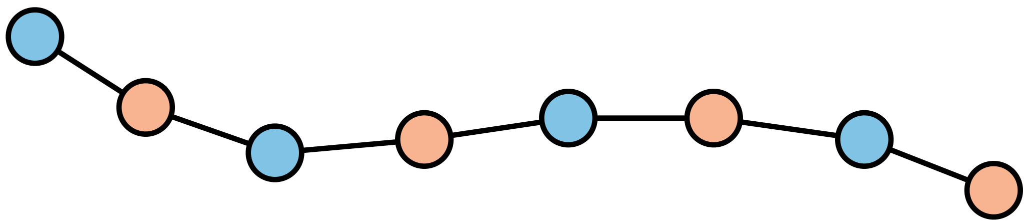
- **Given:** a path graph
- **Find:** a proper 2-coloring



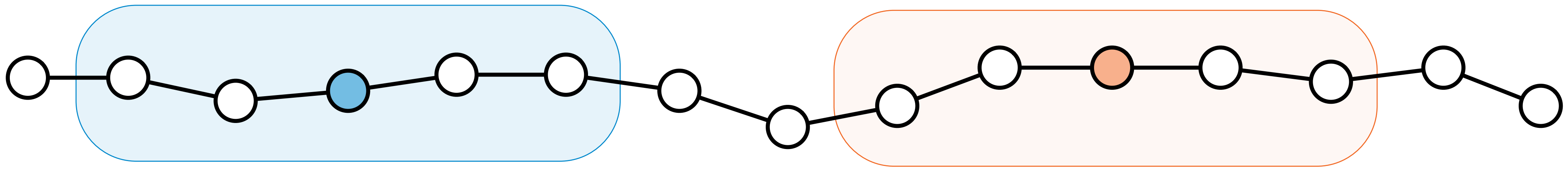
- **Restriction:** the color of each node ***only depends on its local neighborhood***
  - say, radius-$o(n)$ neighborhood

Coloring paths locally?

- **Given:** a path graph

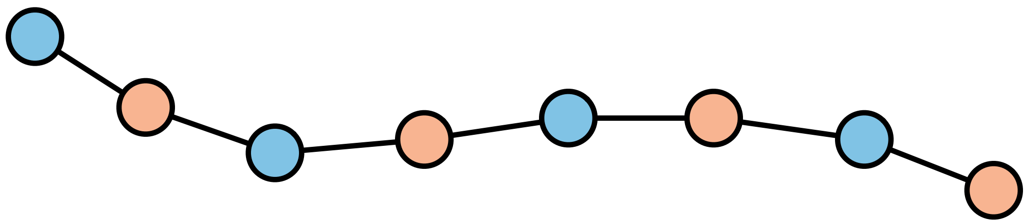


- **Find:** a proper 2-coloring
- **Restriction:** the color of each node *only depends on its local neighborhood*
  - say, radius-$o(n)$ neighborhood

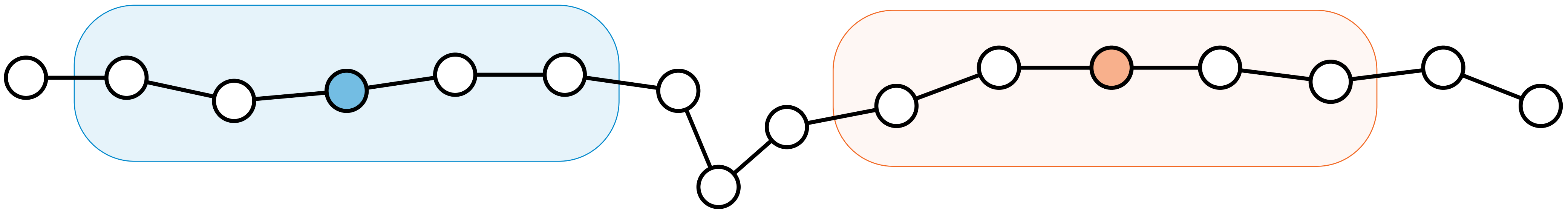


Coloring paths locally?

- **Given:** a path graph



- **Find:** a proper 2-coloring
- **Restriction:** the color of each node *only depends on its local neighborhood*
  - say, radius-$o(n)$ neighborhood

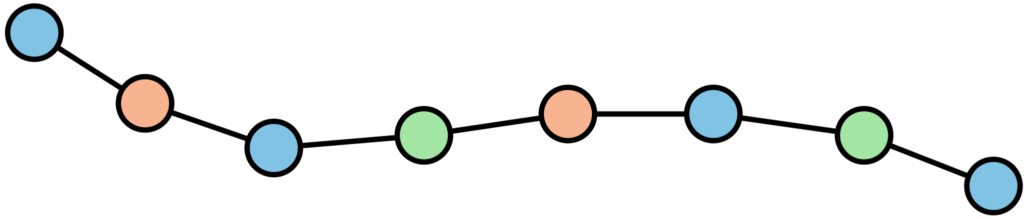


Conclusion:

Paths can’t be 2-colored with any local strategy

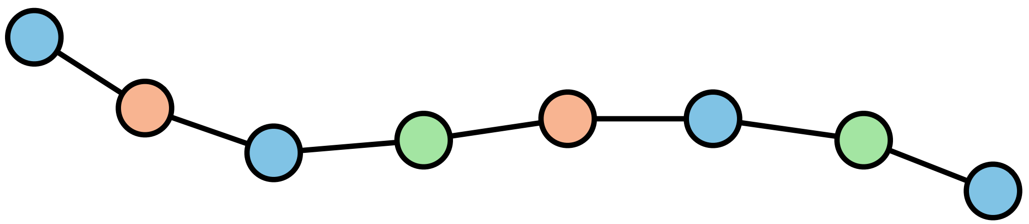
… and it doesn’t really depend on exactly how we define “local”

3-coloring paths locally?

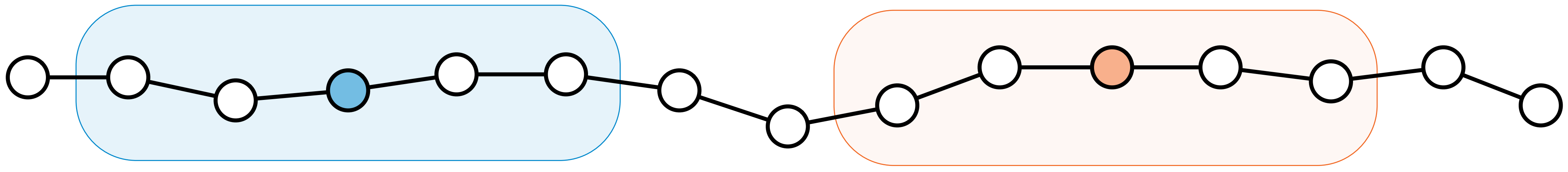


- **Given:** a path graph
- **Find:** a proper 3-coloring
- **Restriction:** the color of each node only depends on its local neighborhood

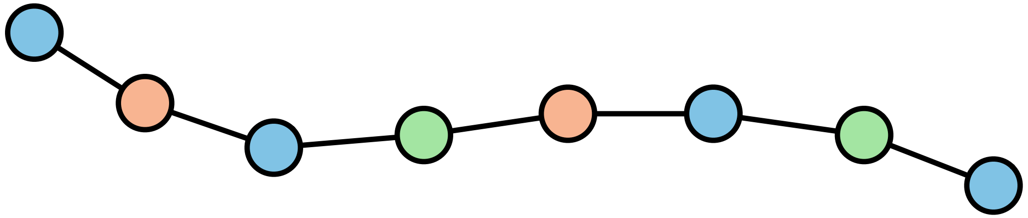
3-coloring paths locally?



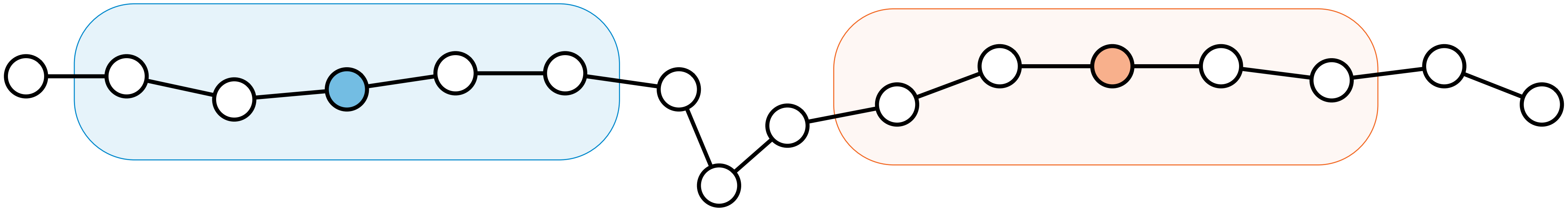
- **Given:** a path graph
- **Find:** a proper 3-coloring
- **Restriction:** the color of each node only depends on its local neighborhood



3-coloring paths locally?

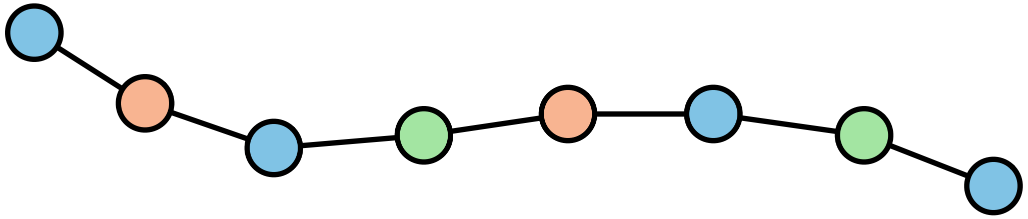


- **Given:** a path graph
- **Find:** a proper 3-coloring
- **Restriction:** the color of each node only depends on its local neighborhood

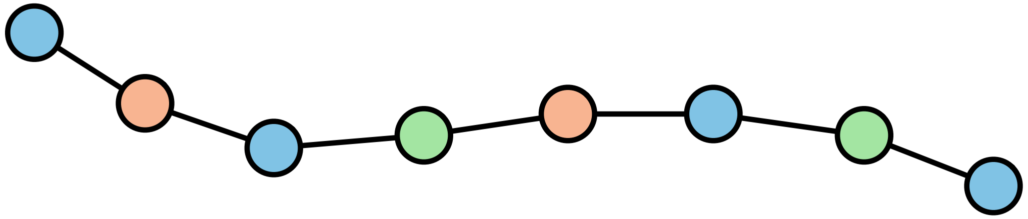


3-coloring paths locally?

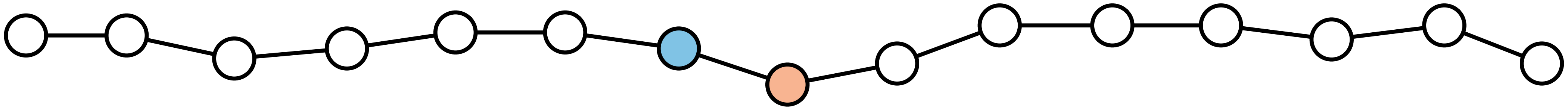
- **Given:** a path graph
- **Find:** a proper 3-coloring
- **Restriction:** the color of each node only depends on its local neighborhood



3-coloring paths locally?



- **Given:** a path graph
- **Find:** a proper 3-coloring
- **Restriction:** the color of each node only depends on its local neighborhood



Conclusion:

We need some way to break local symmetry

Conclusion:

We need some way to break local symmetry

- randomness
- unique node identifiers
- sequential ordering …

3-coloring paths locally

Nodes labeled with (small) unique identifiers:

locality ≈ ½ log* *n*

3-coloring paths locally

Nodes labeled with (small) unique identifiers:

locality ≈ ½ log* *n*

Nodes labeled with random bit strings:

locality ≈ ½ log* *n*

3-coloring paths locally

Nodes labeled with (small) unique identifiers:

locality ≈ ½ log* *n*

Nodes labeled with random bit strings:

locality ≈ ½ log* *n*

[Cole & Vishkin 1986, Linial 1992, Naor 1991]

Four models of computing

SLOCAL
distributed,
sequential

LOCAL
distributed,
parallel

online LOCAL
centralized

dynamic LOCAL
centralized

SLOCAL
distributed,
sequential

LOCAL
distributed,
parallel

online LOCAL
centralized

dynamic LOCAL
centralized

LOCAL

distributed,
parallel

Each node **in parallel**:

- looks at its radius-T neighborhood
- picks its output based on this information

(nodes have unique identifiers)

SLOCAL
distributed,
sequential

LOCAL
distributed,
parallel

online LOCAL
centralized

dynamic LOCAL
centralized

SLOCAL
distributed,
sequential

LOCAL
distributed,
parallel

online LOCAL
centralized

dynamic LOCAL
centralized

SLOCAL

distributed,
sequential

Each node in a **sequential, adversarial order**:

- looks at its radius-T neighborhood
- picks its output & state based on this information

SLOCAL
distributed,
sequential

LOCAL
distributed,
parallel

online LOCAL
centralized

dynamic LOCAL
centralized

SLOCAL

distributed,
sequential

LOCAL

distributed,
parallel

online LOCAL

centralized

dynamic LOCAL

centralized

Graph **constructed** by an adversary that adds nodes and edges one by one

We can **see everything**

We can **change** our output only within distance T from a point of change

dynamic

LOCAL

centralized

SLOCAL
distributed,
sequential

LOCAL
distributed,
parallel

online LOCAL
centralized

dynamic LOCAL
centralized

SLOCAL

distributed,
sequential

LOCAL

distributed,
parallel

online LOCAL

centralized

dynamic LOCAL

centralized

Some unknown input graph is **revealed** piece by piece:

- adversary points at a node *v*
- we can see the radius-*T* neighborhood of *v*
- we have to choose the label for *v*

We can **remember** everything

online
LOCAL
centralized

SLOCAL
distributed,
sequential

LOCAL
distributed,
parallel

online LOCAL
centralized

dynamic LOCAL
centralized

Genuinely different models

SLOCAL
distributed,
sequential

LOCAL
distributed,
parallel

online LOCAL
centralized

dynamic LOCAL
centralized

coloring

SLOCAL

distributed,
sequential

LOCAL

distributed,
parallel

online LOCAL

centralized

dynamic LOCAL

centralized

coloring

SLOCAL
distributed,
sequential

LOCAL
distributed,
parallel

online LOCAL
centralized

cycle
detection

dynamic LOCAL
centralized

SLOCAL
distributed,
sequential

coloring

LOCAL
distributed,
parallel

online LOCAL
centralized

dynamic LOCAL
centralized

cycle detection

leader election

Closely related models

SLOCAL

distributed,
sequential

LOCAL

distributed,
parallel

online LOCAL

centralized

dynamic LOCAL

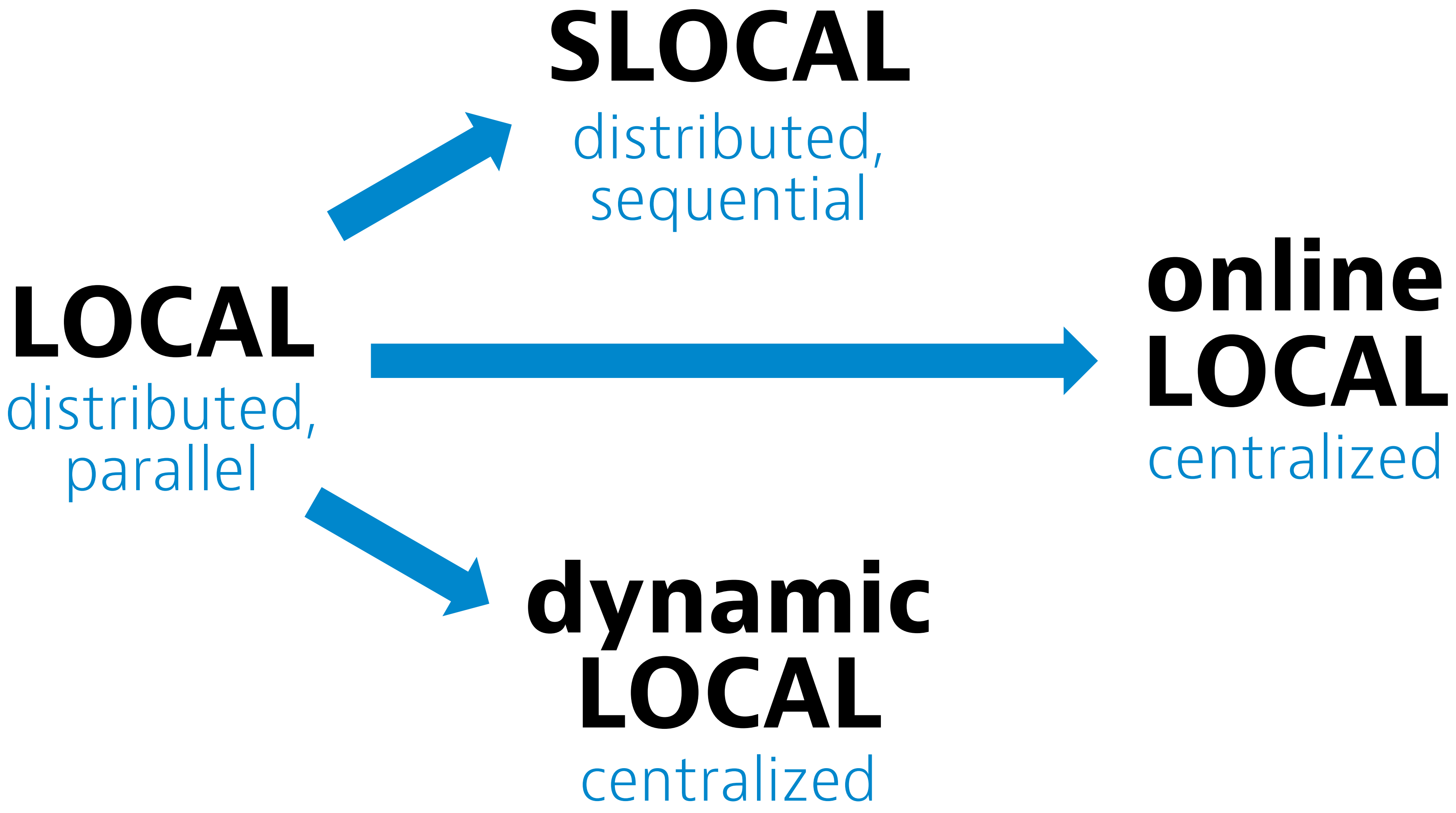
centralized

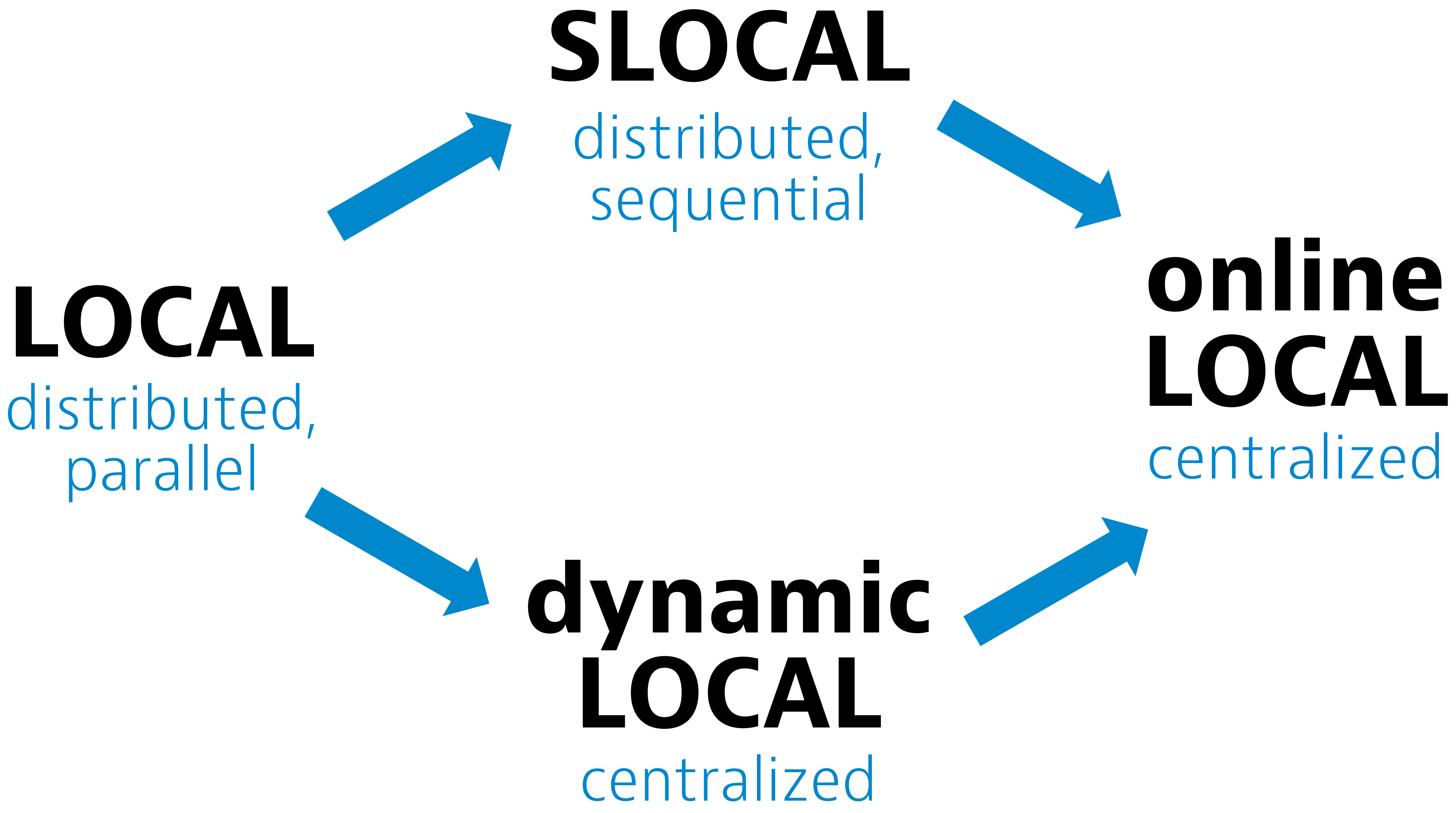
LOCAL
distributed,
parallel

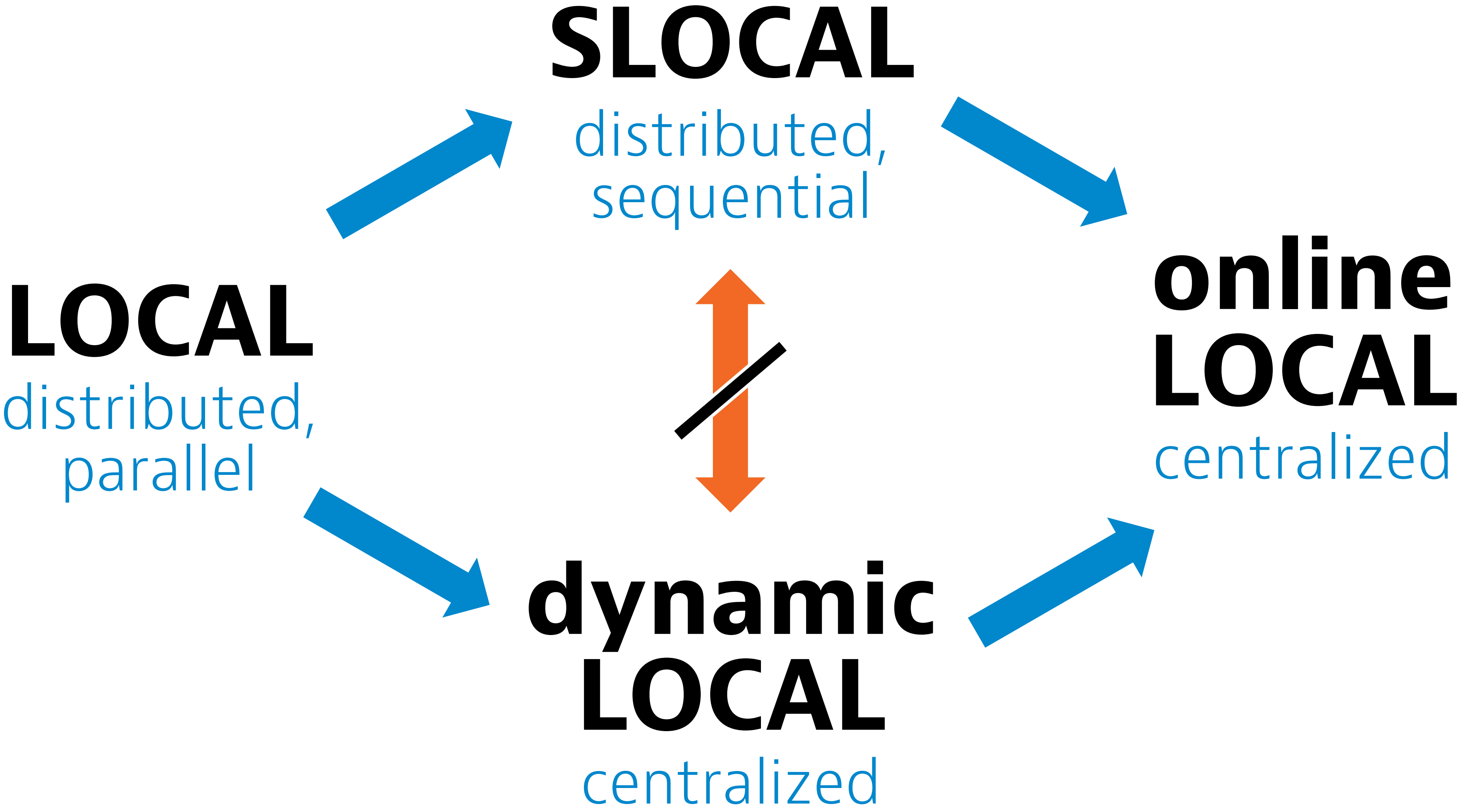
SLOCAL
distributed,
sequential

dynamic LOCAL
centralized

online LOCAL
centralized


SLOCAL
distributed,
sequential
LOCAL
distributed,
parallel
online
LOCAL
centralized
dynamic
LOCAL
centralized


SLOCAL
distributed, sequential
LOCAL
distributed, parallel
online LOCAL
centralized
dynamic LOCAL
centralized


SLOCAL
distributed, sequential
LOCAL
distributed, parallel
online LOCAL
centralized
dynamic LOCAL
centralized

Collapse in rooted trees

LCLs in rooted trees

- **Rooted regular trees**
- **Locally checkable labelings** (LCLs)
  - solution valid if it “looks good everywhere”
  - example: 3-coloring
- ***In this setting all models equally strong!***

LCLs in rooted trees

SLOCAL

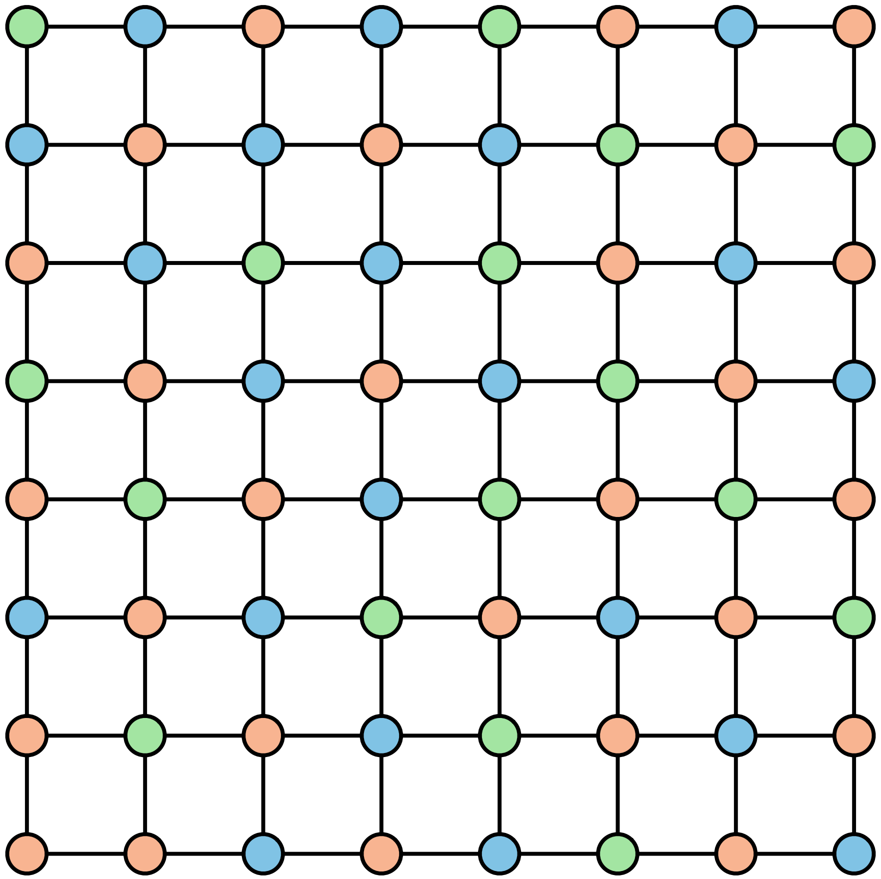
LOCAL ≈ SLOCAL ≈ online LOCAL

LOCAL ≈ dynamic LOCAL ≈ online LOCAL

dynamic LOCAL

Case study: grids

Coloring grids



Coloring grids

- **5-coloring:** local in all models (easy to see)

Coloring grids

- **5-coloring:** local in all models (easy to see)
- **4-coloring:** local in all models (hard to see)

Coloring grids

- **5-coloring:** local in all models (easy to see)
- **4-coloring:** local in all models (hard to see)
- **3-coloring:**
  - LOCAL, SLOCAL: global

Coloring grids

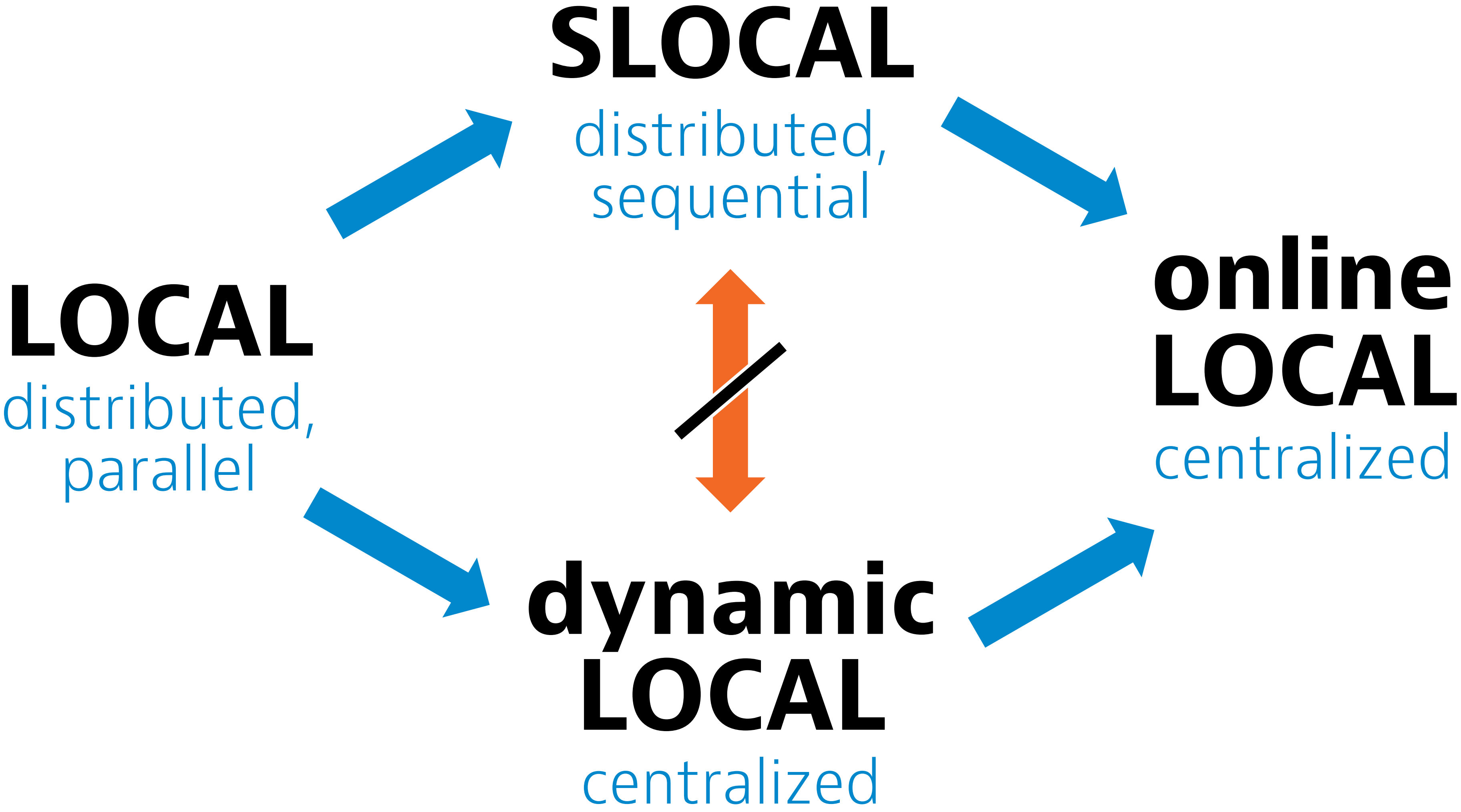
- **5-coloring:** local in all models (easy to see)
- **4-coloring:** local in all models (hard to see)
- **3-coloring:**
  - LOCAL, SLOCAL: global
  - **online-LOCAL: $O(\log n)$**

Coloring grids

- **5-coloring:** local in all models (easy to see)
- **4-coloring:** local in all models (hard to see)
- **3-coloring:**
  - LOCAL, SLOCAL: global
  - online-LOCAL: $O(\log n)$ — **is this tight?**

Coloring grids

- **5-coloring:** local in all models (easy to see)
- **4-coloring:** local in all models (hard to see)
- **3-coloring:**
  - LOCAL, SLOCAL: global
  - online-LOCAL: $O(\log n)$ — **is this tight?**
  - dynamic-LOCAL: **open**


SLOCAL
distributed, sequential
LOCAL
distributed, parallel
online LOCAL
centralized
dynamic LOCAL
centralized