A Lower Bound for the Distributed Lovász Local Lemma

Sebastian Brandt, Orr Fischer, Juho Hirvonen, Barbara Keller, Tuomo Lempiäinen, Joel Rybicki, Jukka Suomela, Jara Uitto

Aalto University, Comerge AG, ETH Zurich, Tel Aviv University

The Lovász Local Lemma

- «Bad» events $E_1, E_2, \ldots, E_n$ with $\Pr(E_i) < 1$

mutually independent

$$\Rightarrow \Pr(\neg E_1 \wedge \neg E_2 \wedge \cdots \wedge \neg E_n) > 0$$

Lovász Local Lemma

- Each event is independent of all but d other events
- $\Pr(E_i) < p$ where $ep(d+1) \leq 1$

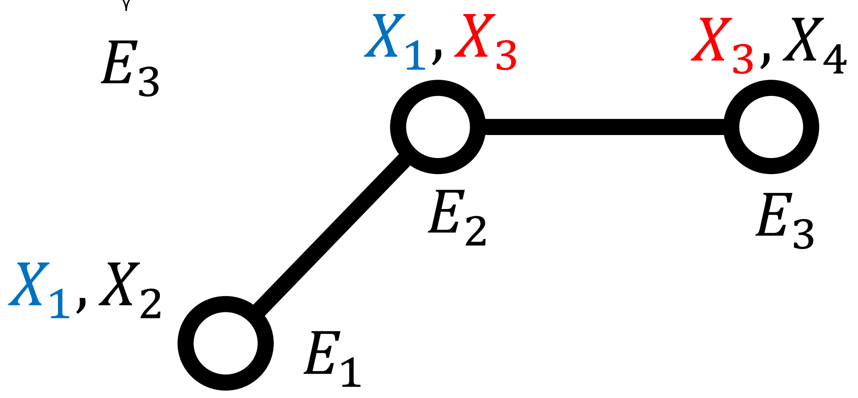
$$\Rightarrow \Pr(\neg E_1 \wedge \neg E_2 \wedge \cdots \wedge \neg E_n) > 0$$

The Constructive LLL

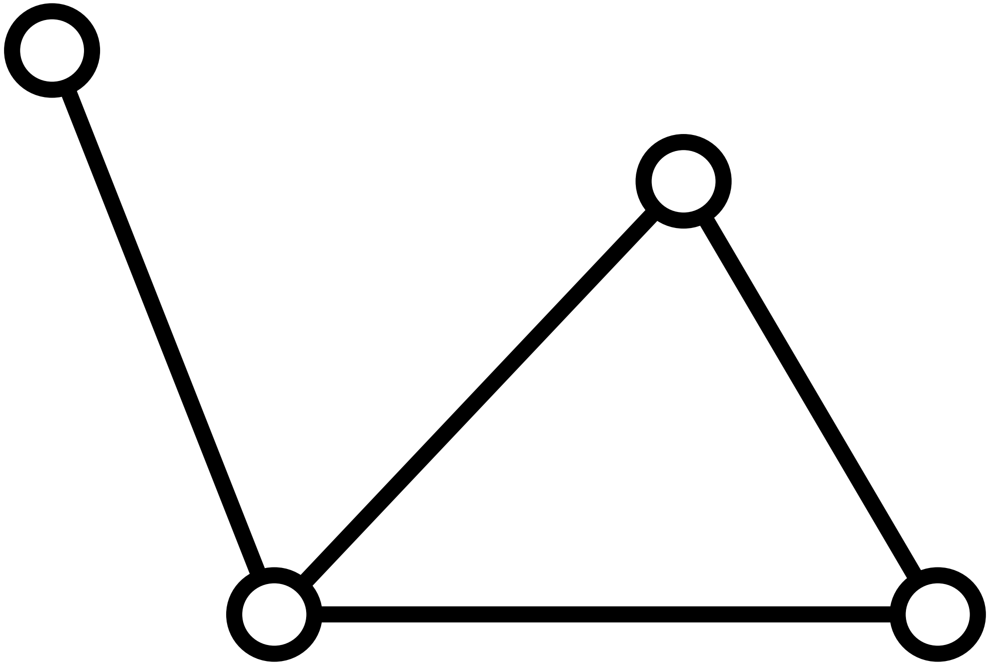
- Mutually independent random variables $X_1, X_2, \ldots, X_k$
- Bad events $E_1, E_2, \ldots, E_n$
- Each event is independent of all but d other events

$$\underbrace{(X_1 \lor \neg X_2)}_{E_1} \land \underbrace{(\neg X_1 \lor X_3)}_{E_2} \land \underbrace{(X_3 \lor X_4)}_{E_3}$$

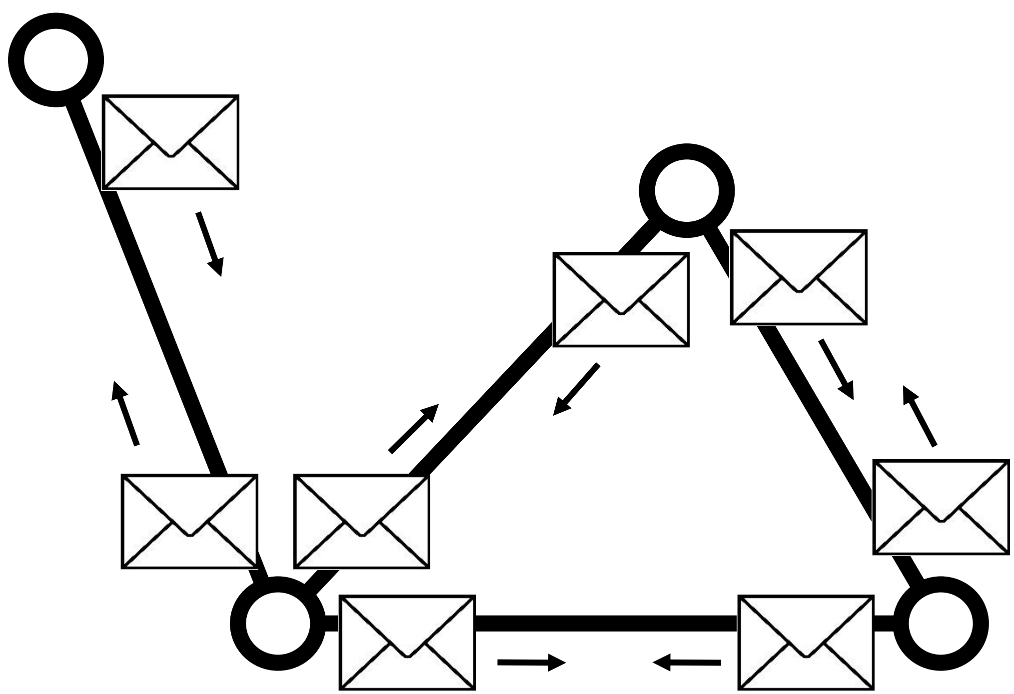
- Dependency graph



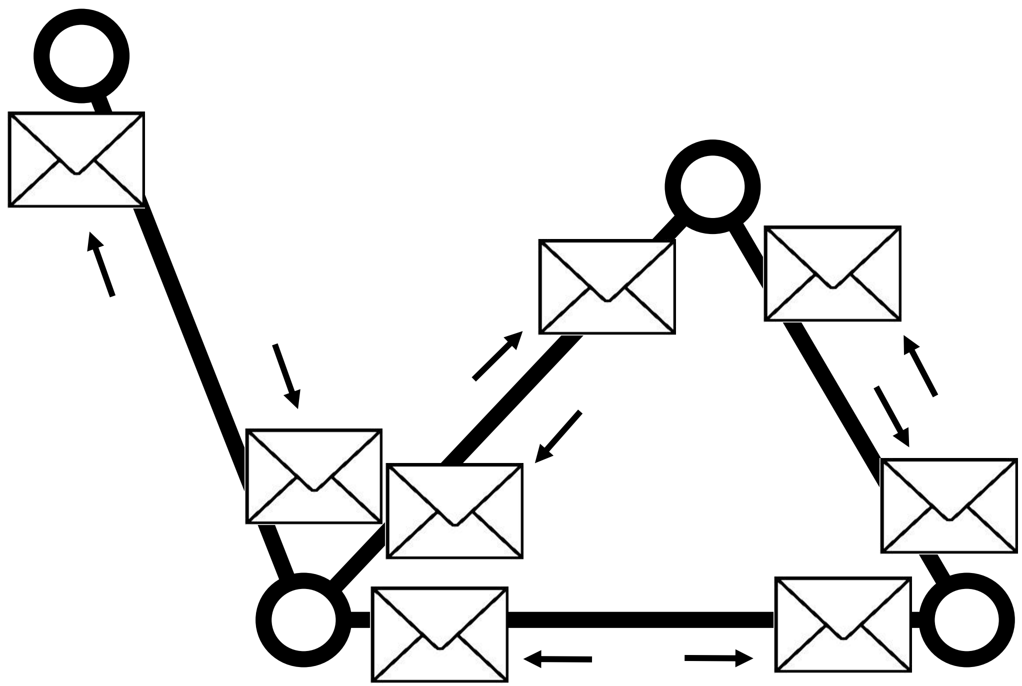
Distributed Computing



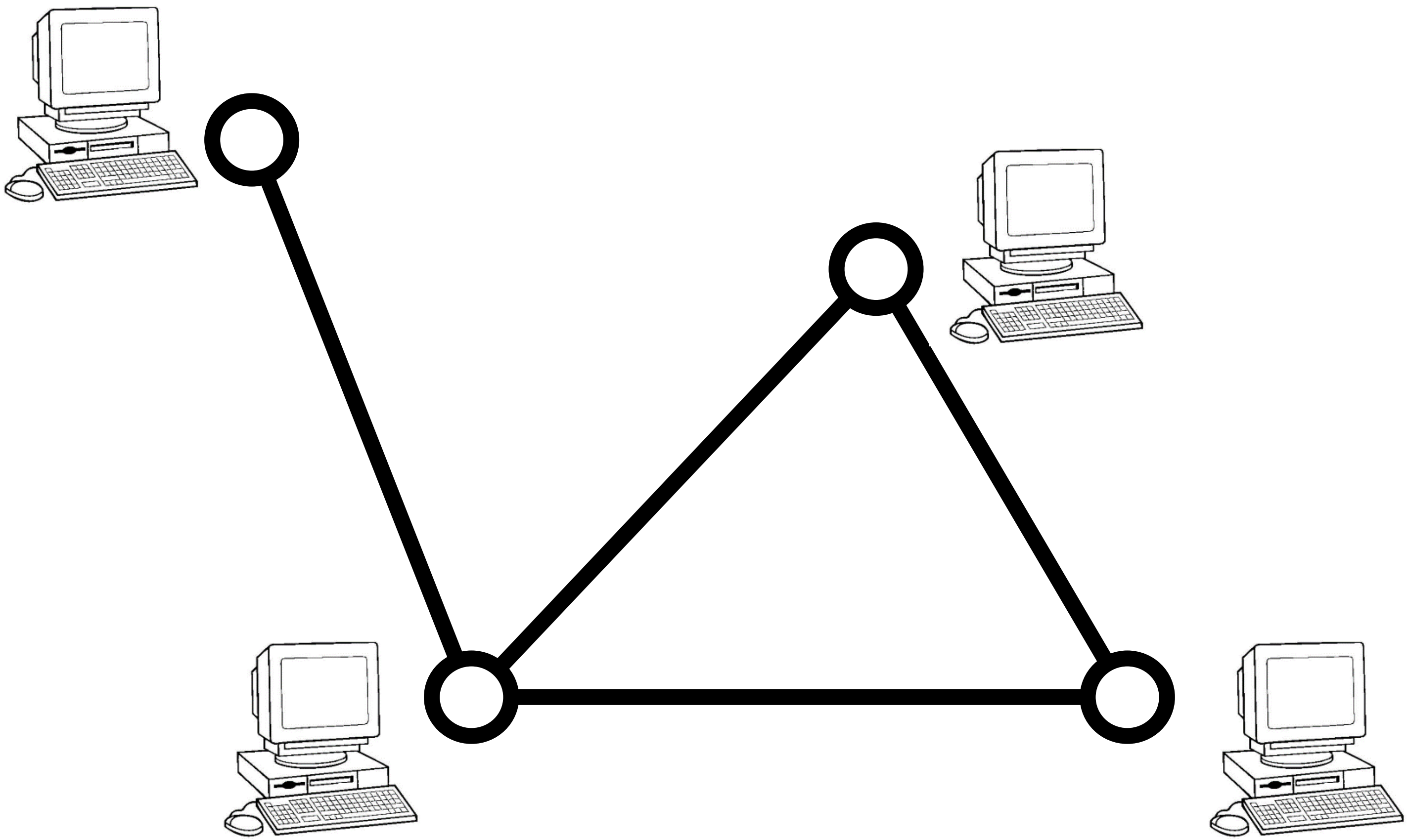
Distributed Computing



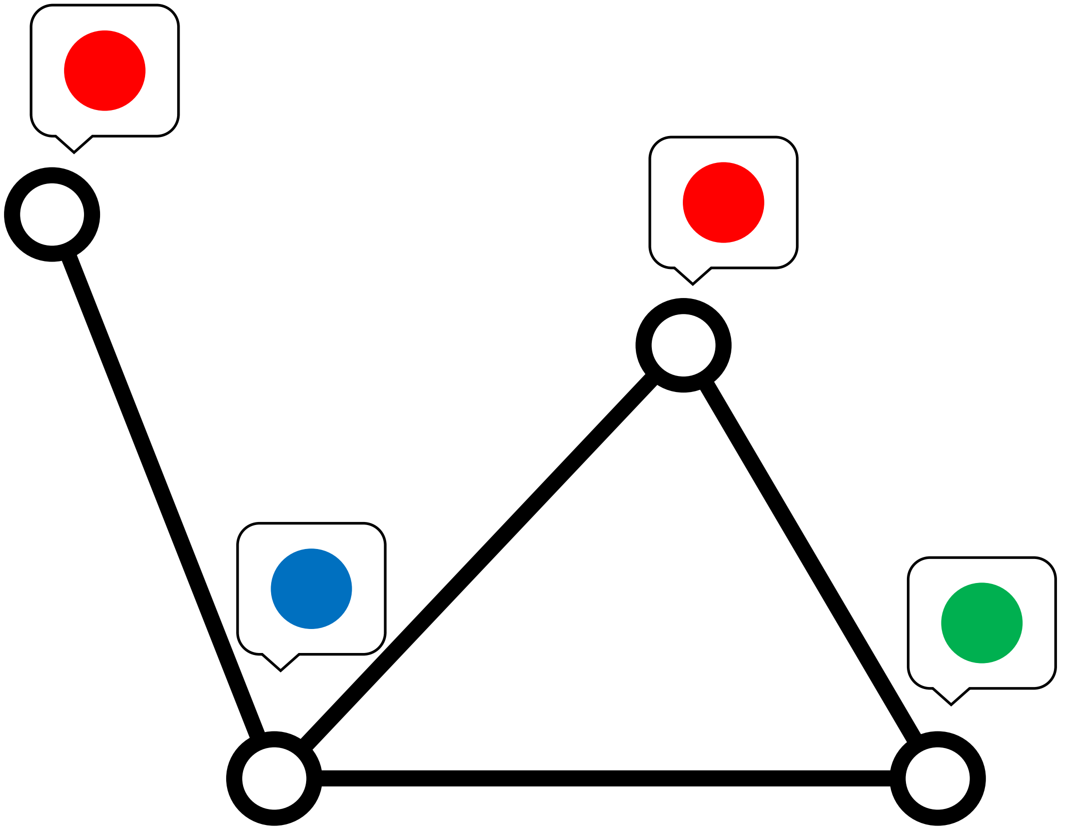
Distributed Computing



Distributed Computing



Distributed Computing



The Distributed LLL

- Input: dependency graph

- Additional input for each node E_i: the random variables that E_i depends on (and *how* E_i depends on them)

- Output of each node E_i: an assignment of the variables it depends on such that:
  1) it agrees with its neighbours
  2) the bad event E_i is avoided

Our result

- Moser and Tardos (2010): $O(\log^2 n)$

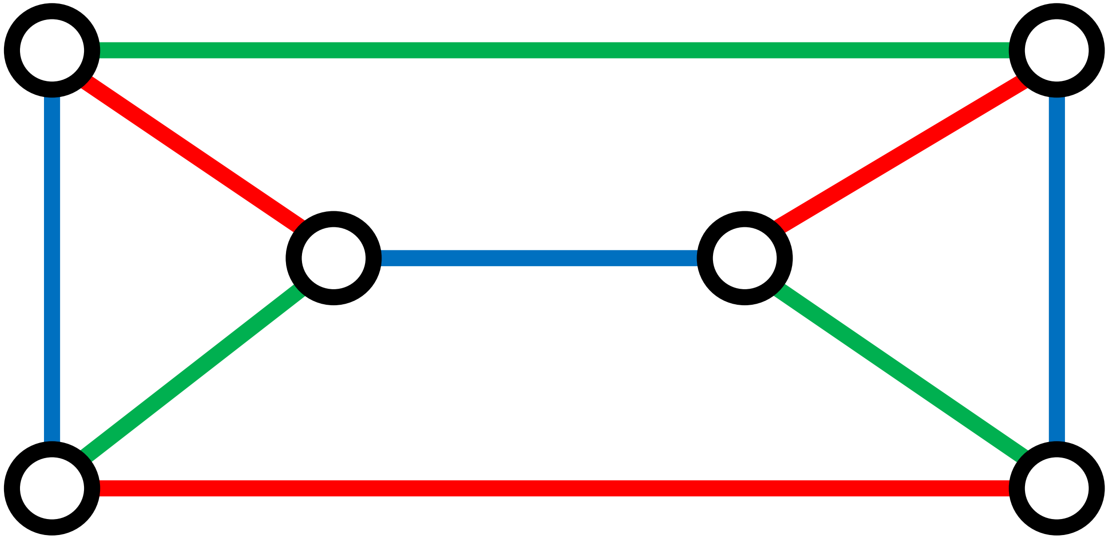
- Chung et al. (2014): $O(\log n)$ for bounded-degree graphs
  $\Omega(\log^* n)$

- $\Omega(\log\log n)$ (Monte-Carlo, w.h.p.)

Sinkless Orientation

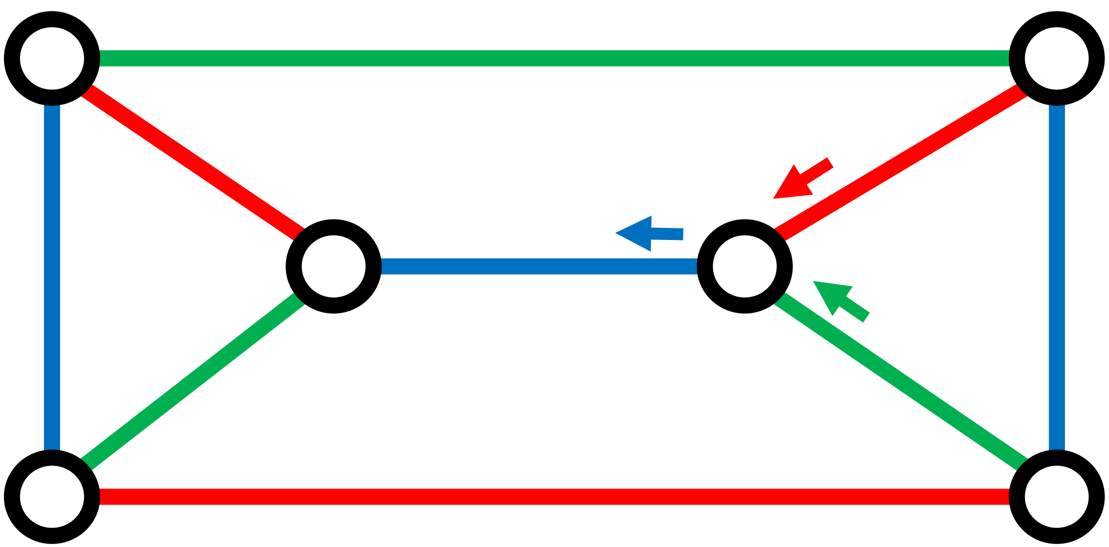
- Input: edge d-coloured, d-regular graph

- Output of each node: non-conflicting orientations of the incident edges such that the node itself is not a sink



Sinkless Orientation

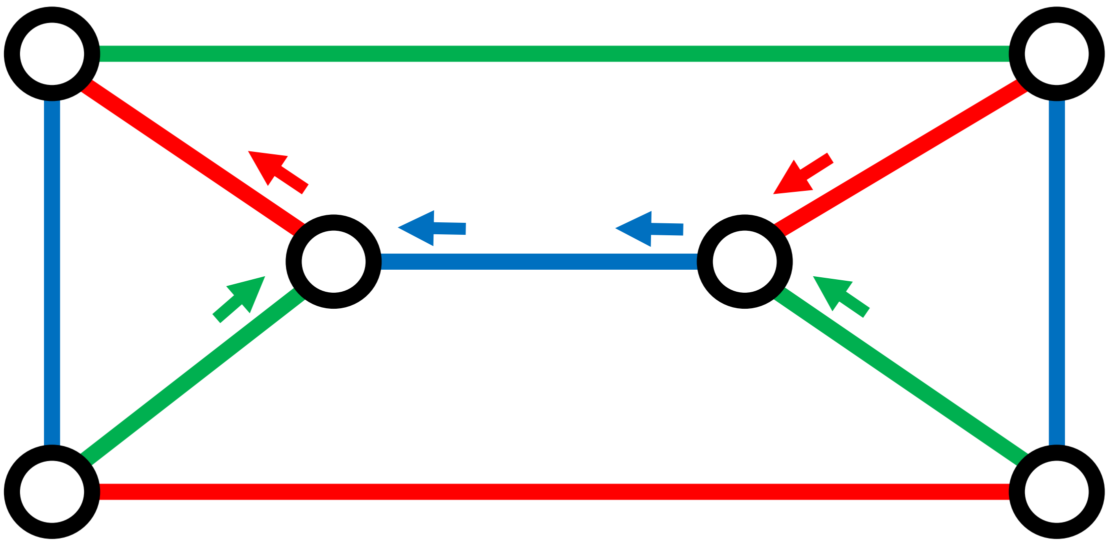
- Input: edge d-coloured, d-regular graph
- Output of each node: non-conflicting orientations of the incident edges such that the node itself is not a sink



Sinkless Orientation

- Input: edge d-coloured, d-regular graph

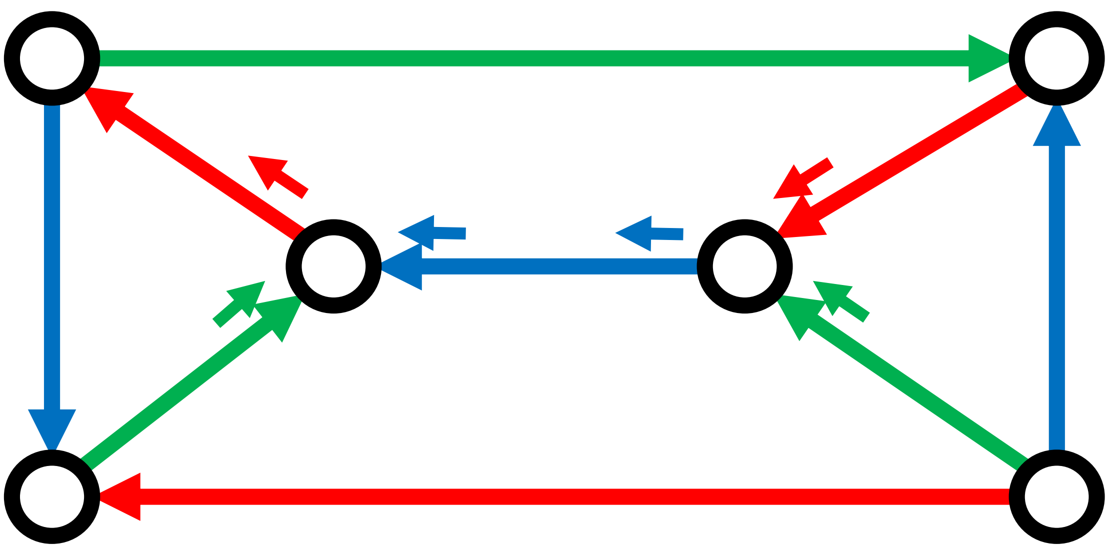
- Output of each node: non-conflicting orientations of the incident edges such that the node itself is not a sink



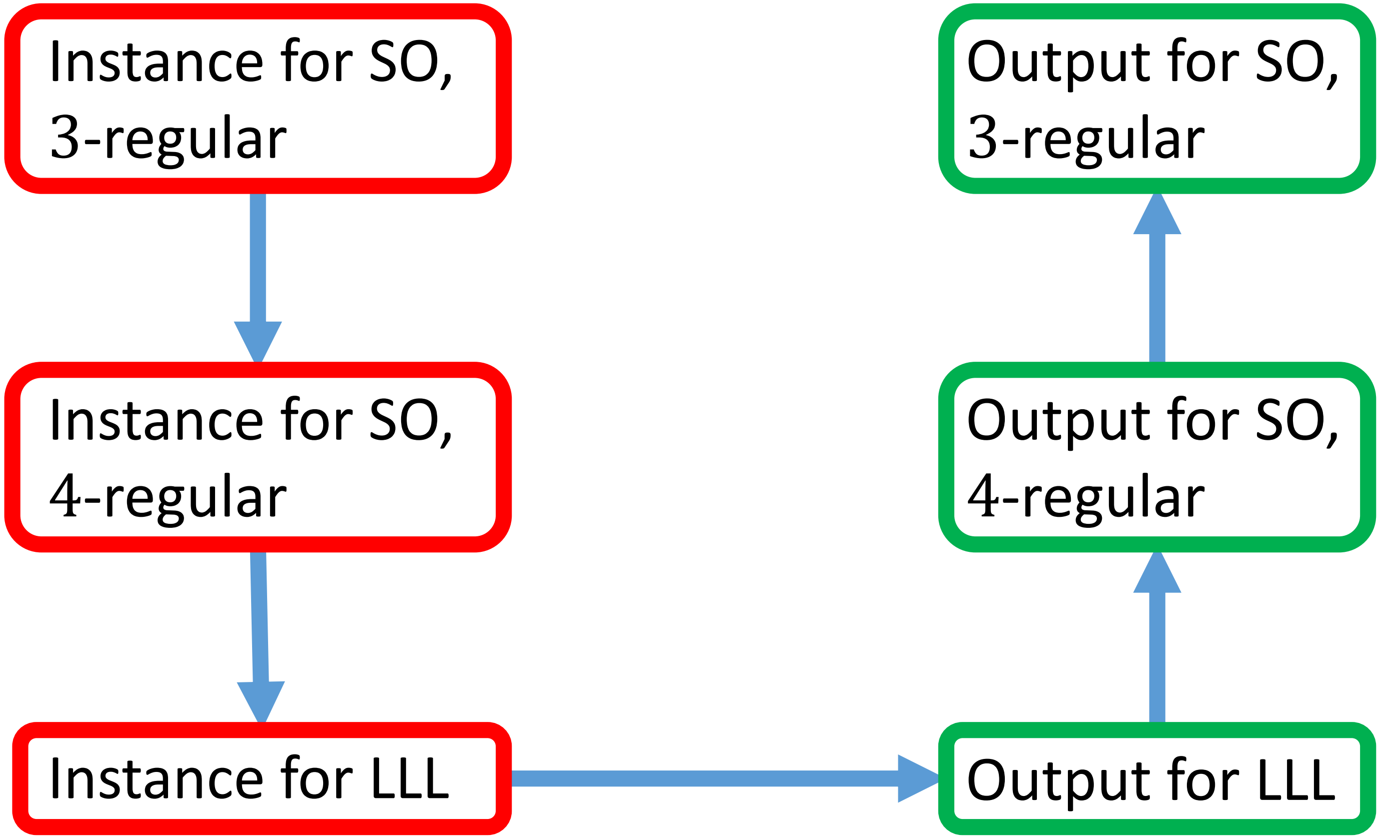
Sinkless Orientation

- Input: edge d-coloured, d-regular graph

- Output of each node: non-conflicting orientations of the incident edges such that the node itself is not a sink



Reduction from SO to LLL



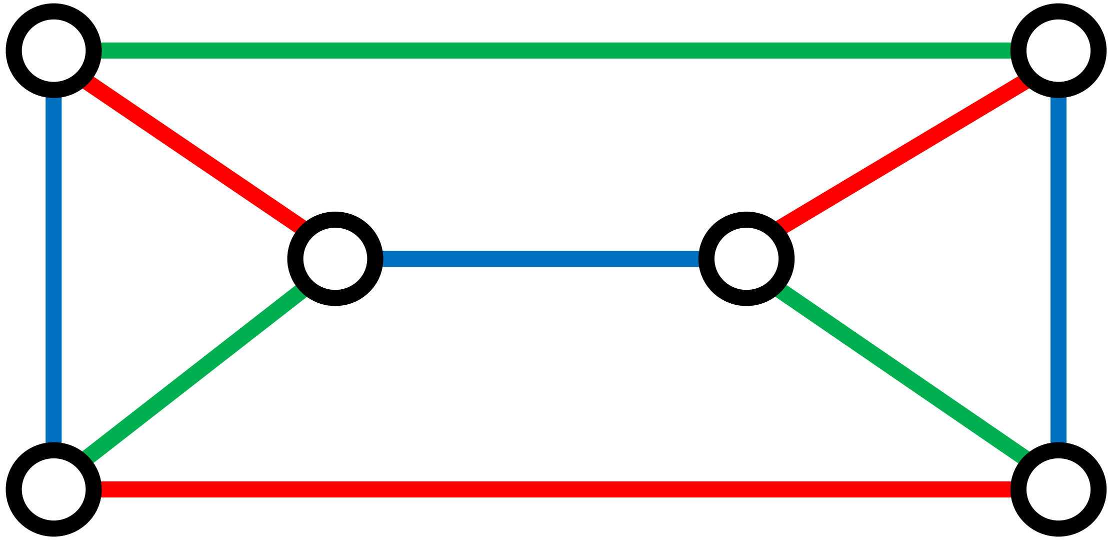
Sinkless Colouring

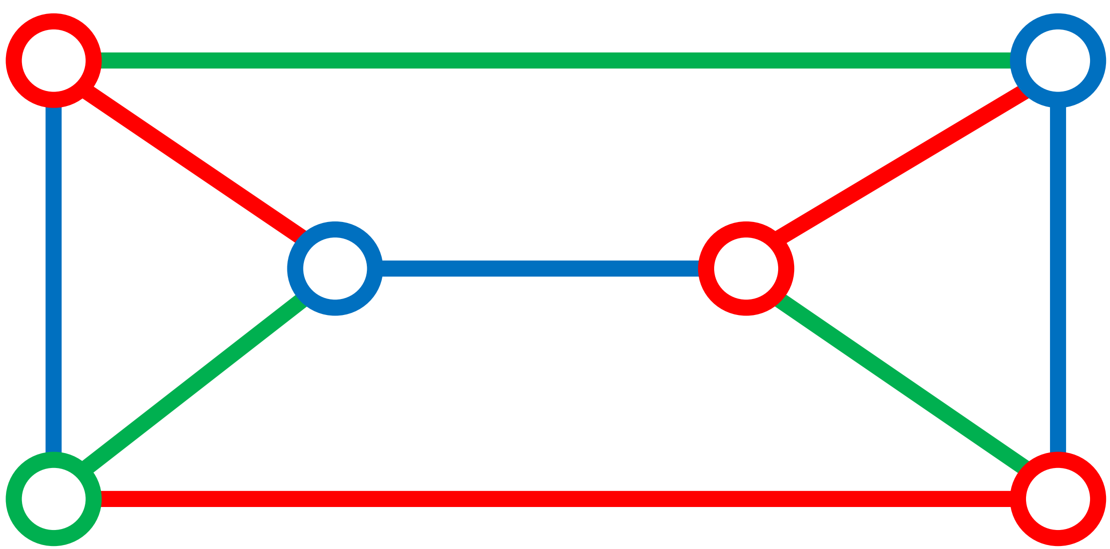
- Input: edge d-coloured, d-regular graph

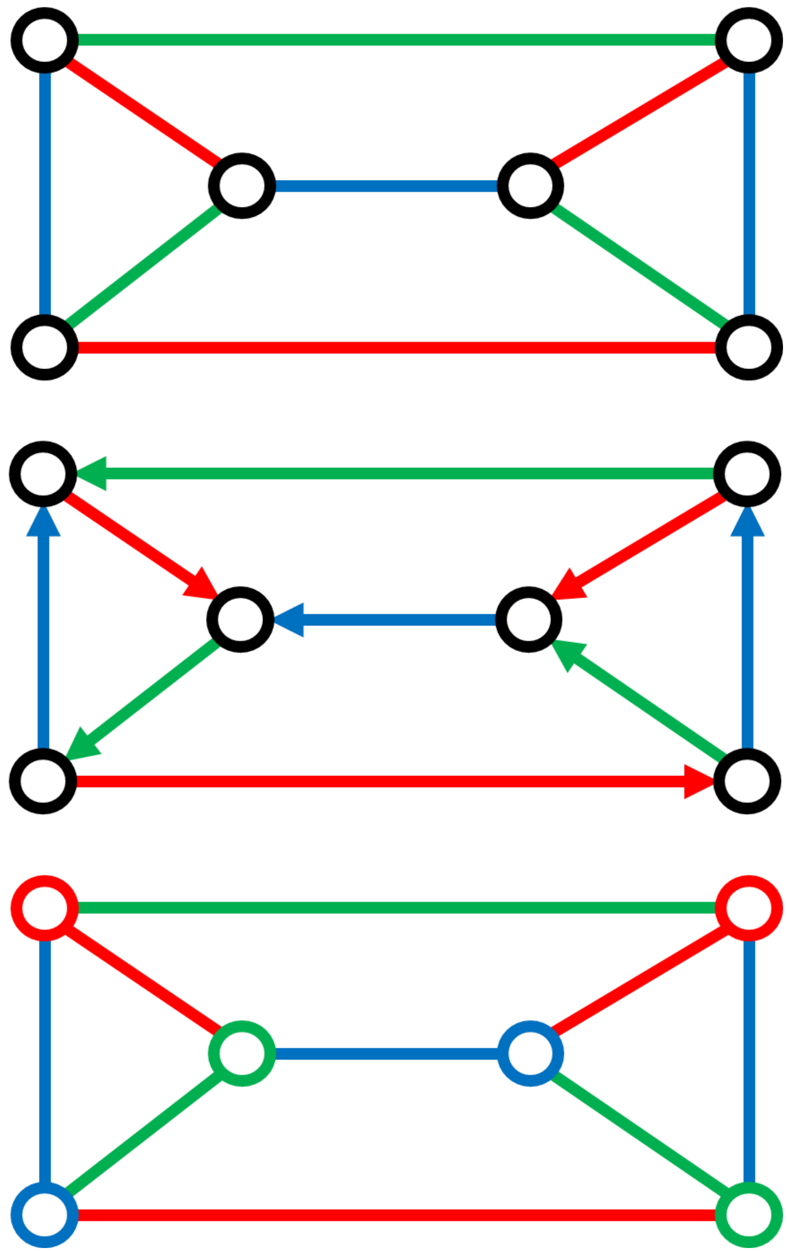
- Output of each node: one of the d colours such that no *forbidden configuration* occurs

Forbidden!

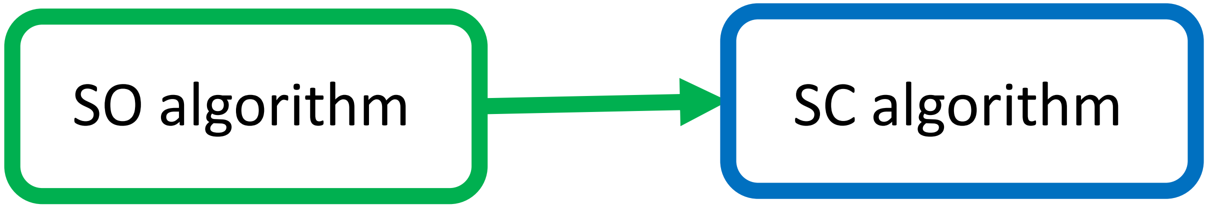
Fine!

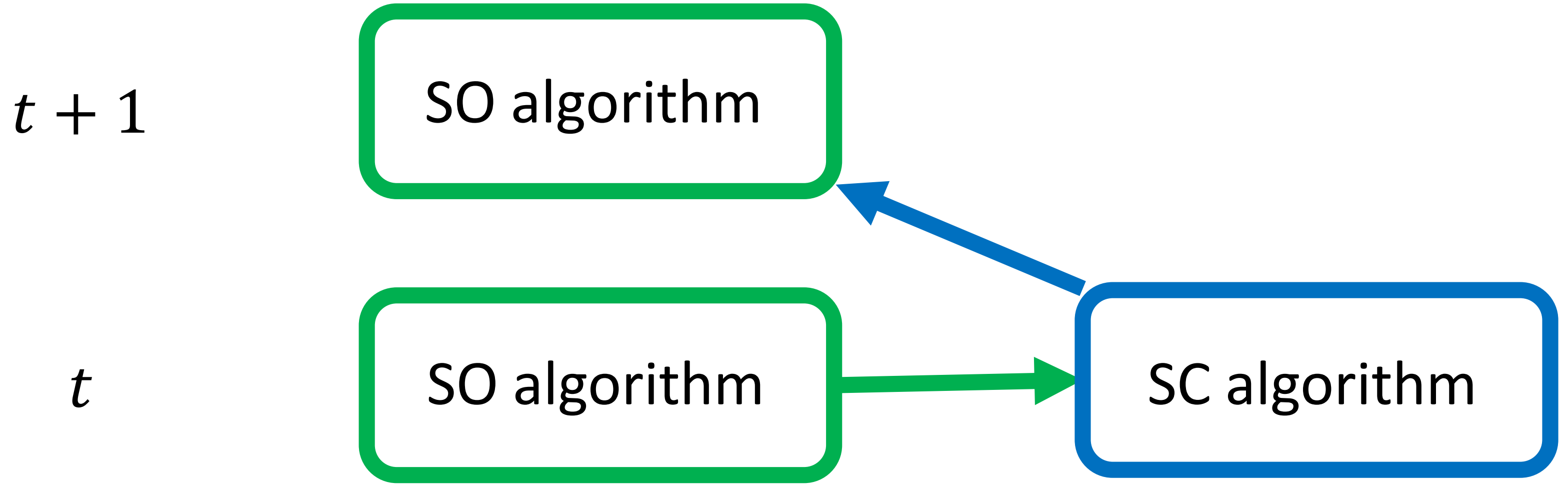


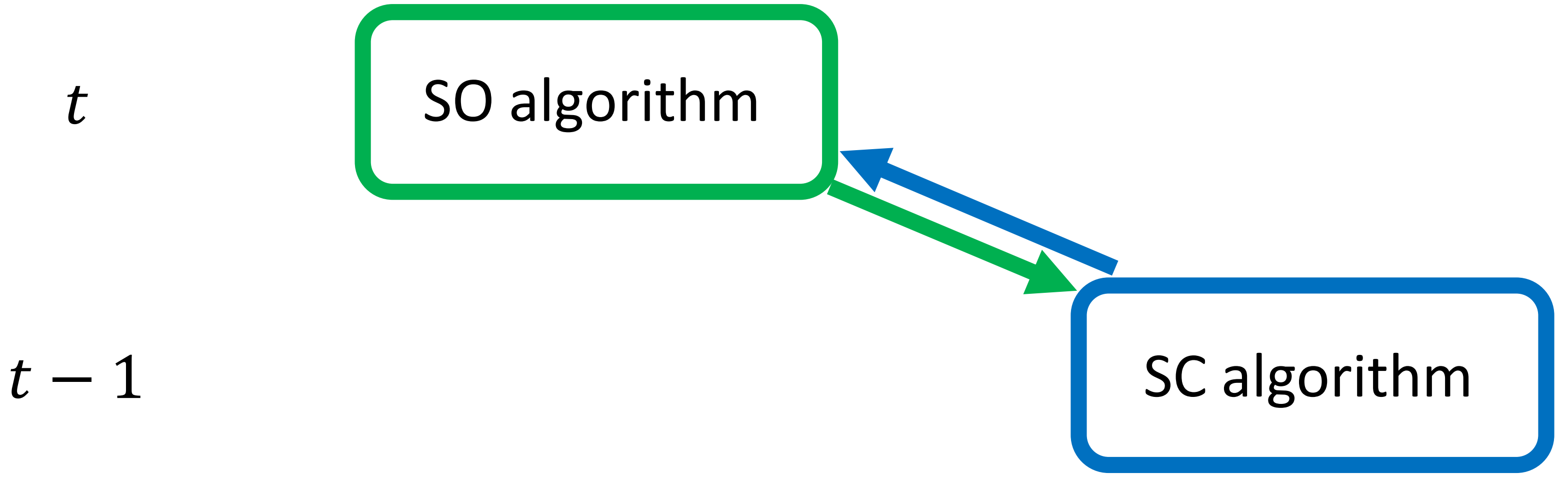




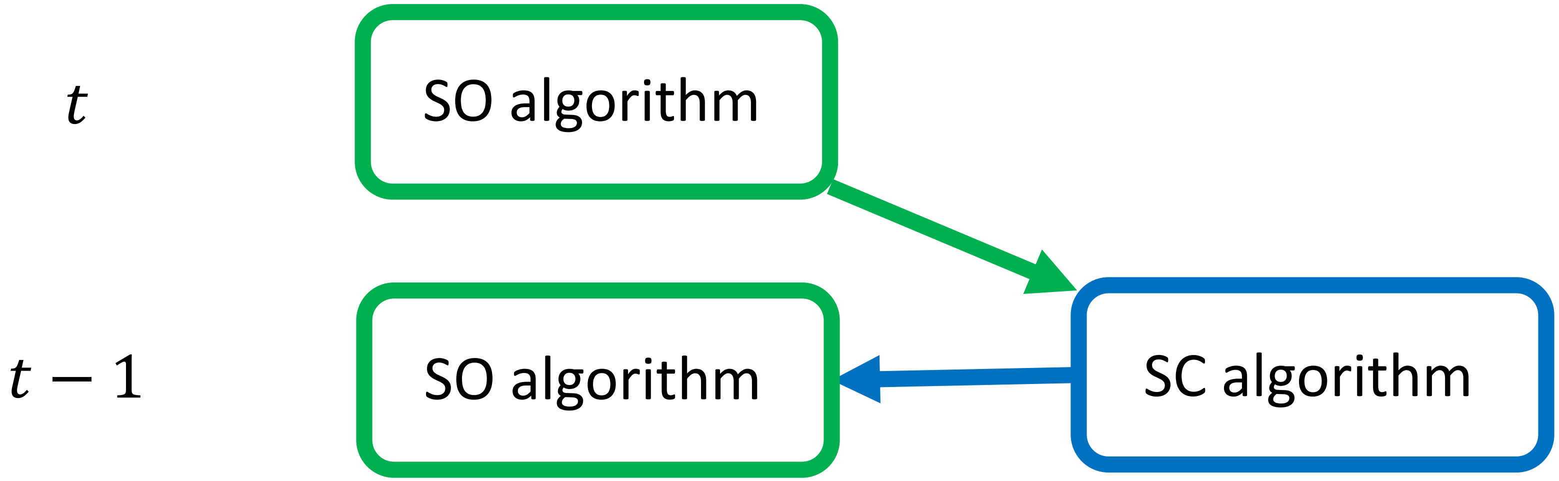
t

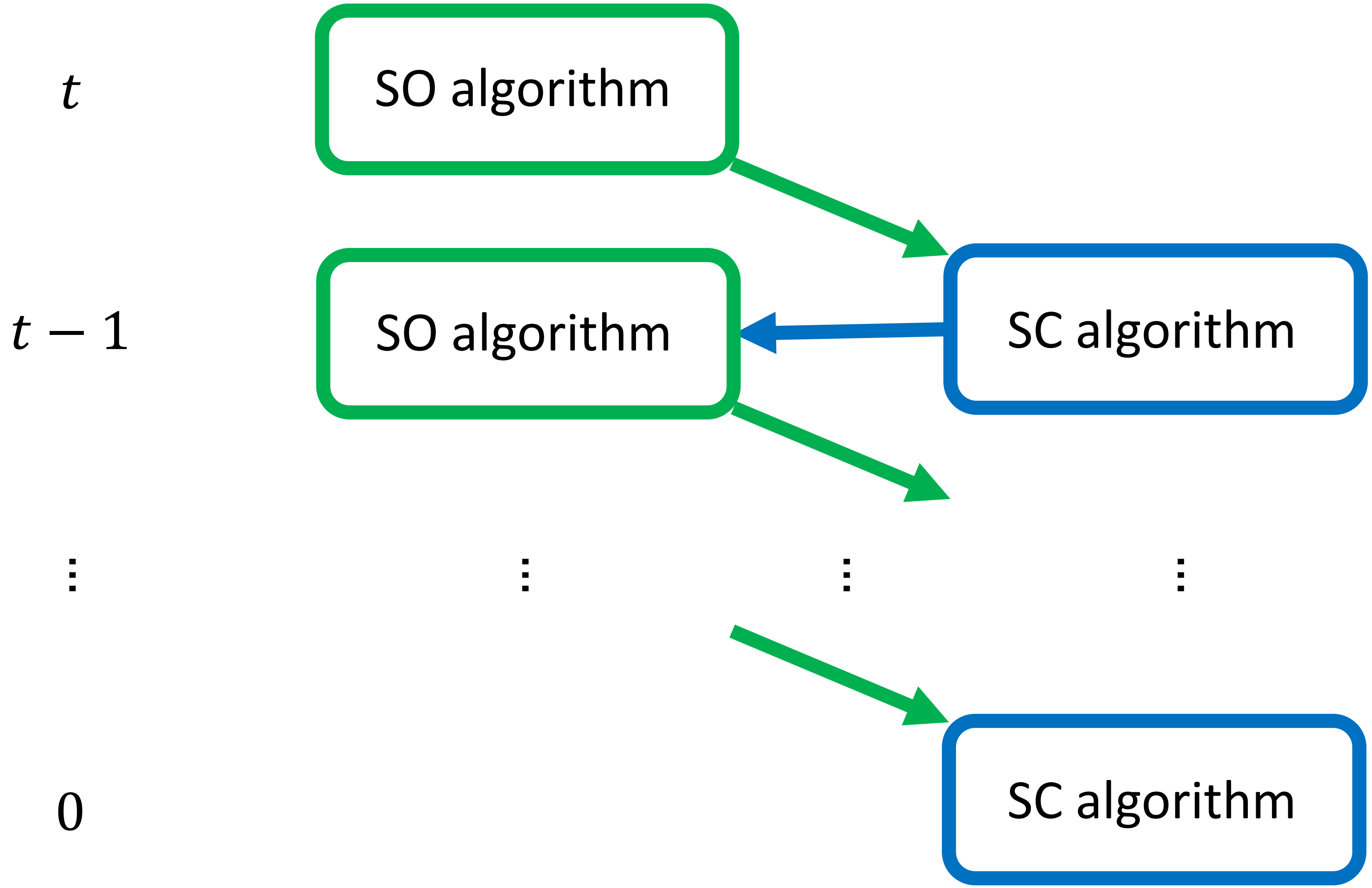



$t+1$
SO algorithm
t
SO algorithm
SC algorithm

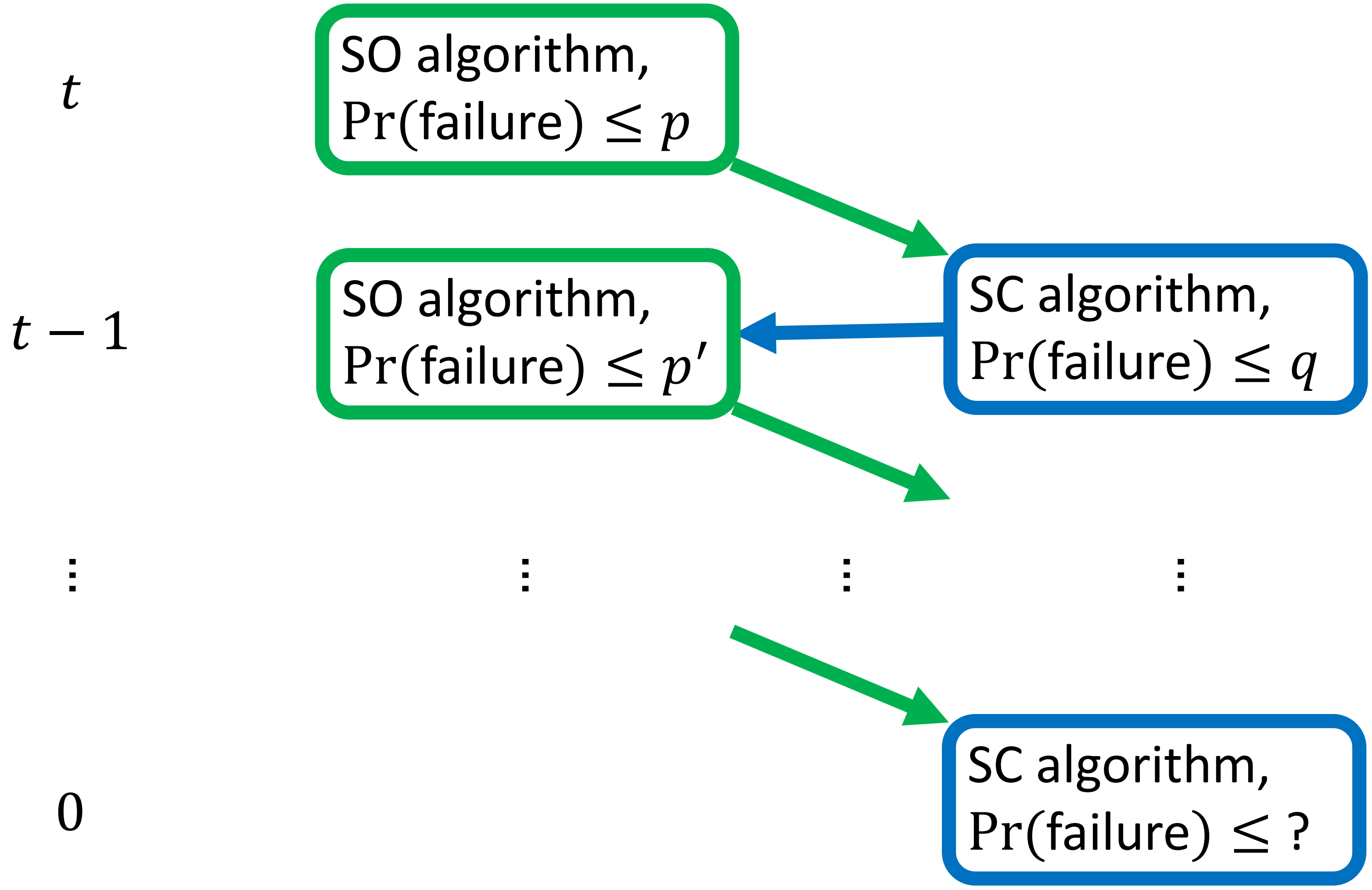


t
SO algorithm
$t-1$
SC algorithm

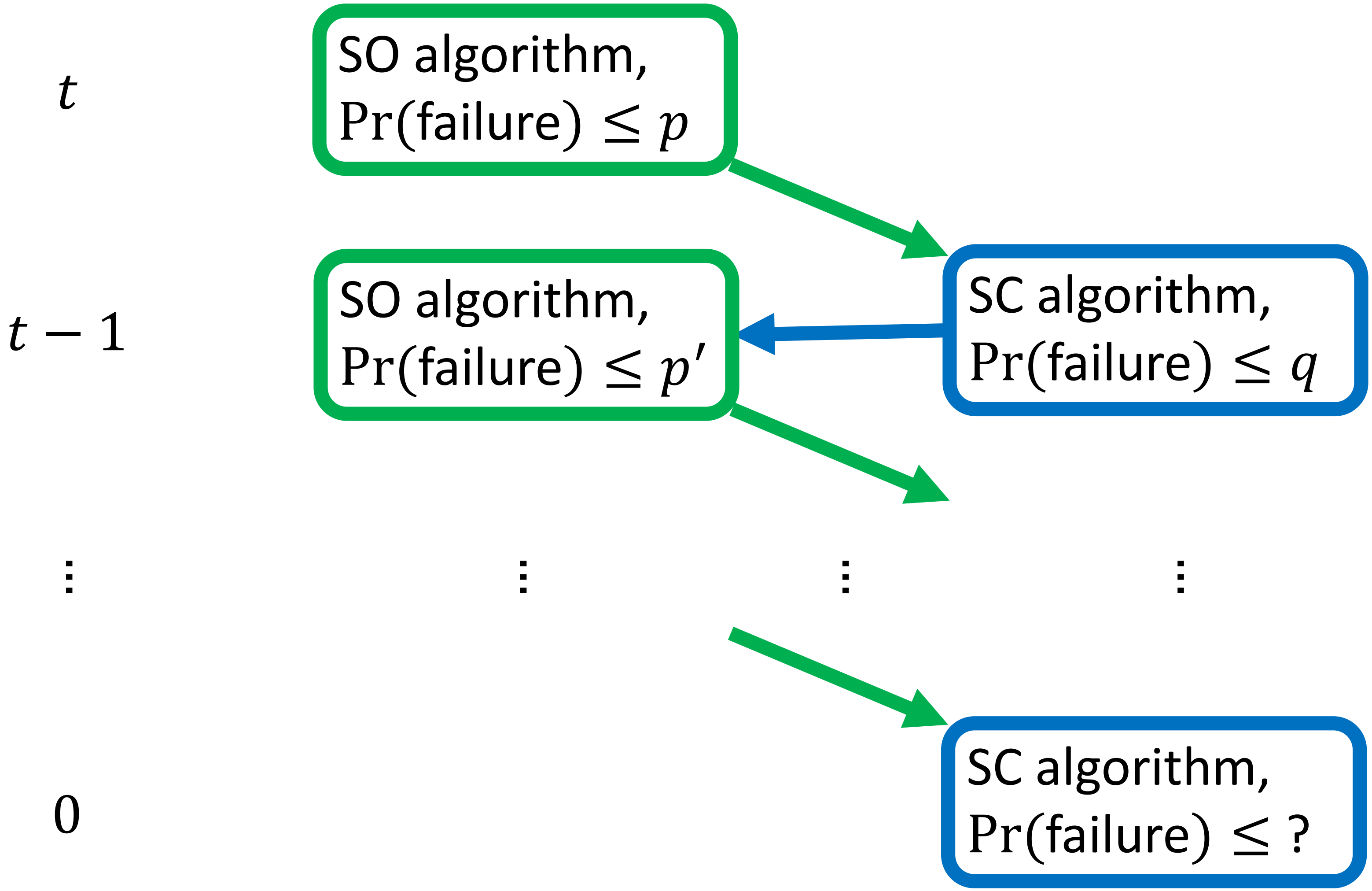

t
$t-1$
SO algorithm
SO algorithm
SC algorithm

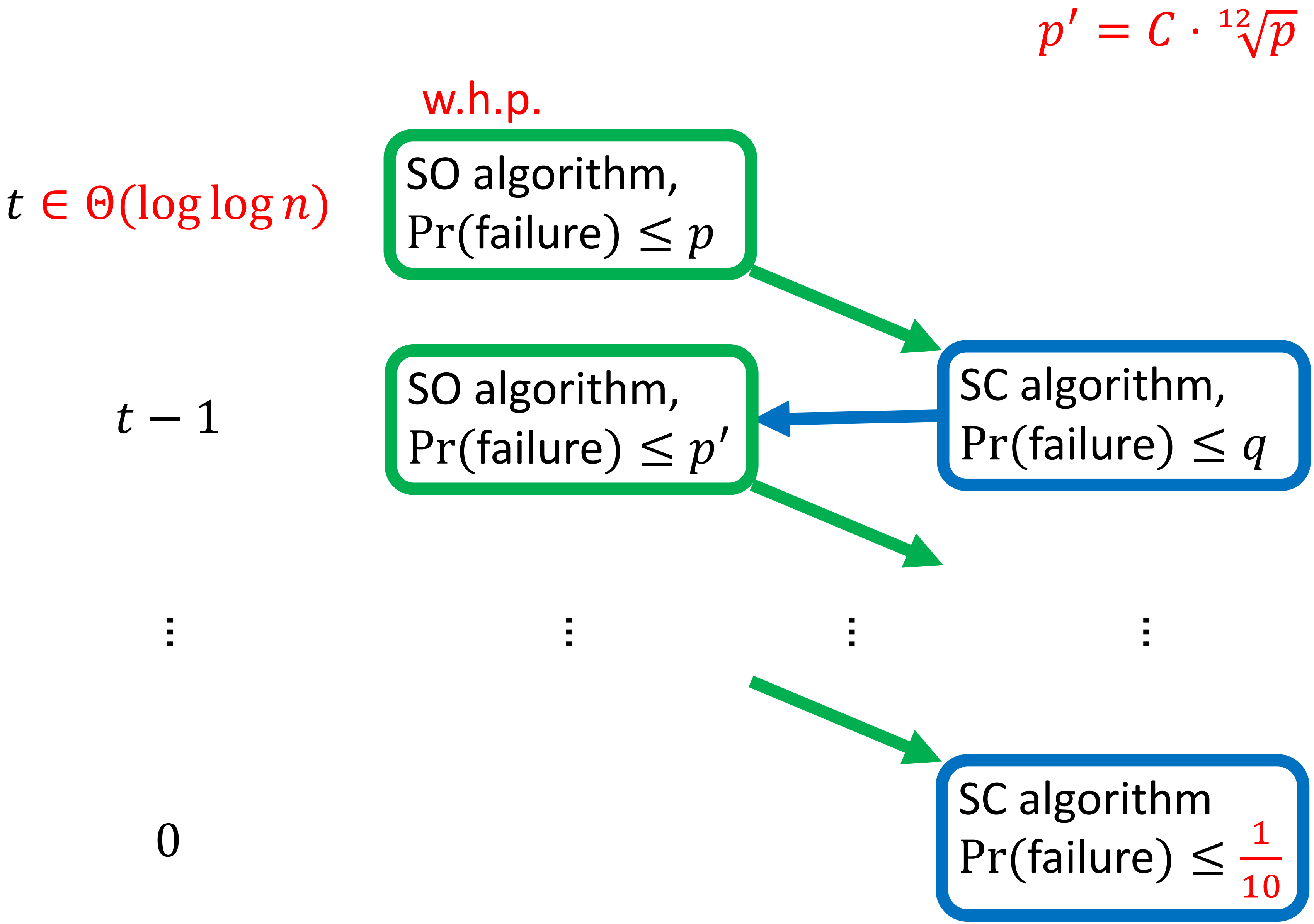


t
SO algorithm
$t-1$
SO algorithm
SC algorithm
$\vdots$
0
SC algorithm


t
SO algorithm,
Pr(failure) $\leq p$
$t-1$
SO algorithm,
Pr(failure) $\leq p'$
SC algorithm,
Pr(failure) $\leq q$
$\vdots$
0
SC algorithm,
Pr(failure) $\leq$?

$$p' = C \cdot \sqrt[12]{p}$$




$p' = C \cdot \sqrt[12]{p}$
w.h.p.
$t \in \Theta(\log \log n)$
SO algorithm,
Pr(failure) $\leq p$
$t - 1$
SO algorithm,
Pr(failure) $\leq p'$
SC algorithm,
Pr(failure) $\leq q$
$\vdots$
0
SC algorithm
Pr(failure) $\leq \frac{1}{10}$

Any Monte-Carlo algorithm for the distributed LLL that gives a correct output w.h.p. needs $\Omega(\log \log n)$ rounds.

Any Monte-Carlo algorithm for the distributed LLL that gives a correct output w.h.p. needs $\Omega(\log \log n)$ rounds.

Any Monte-Carlo algorithm for finding a node d-colouring in d-regular, bipartite, $\Omega(\log n)$-girth graphs that gives a correct output w.h.p. needs $\Omega(\log \log n)$ rounds.

Chang et al. (2016)

The randomised time complexity of finding a node d-colouring in trees with maximum degree d is $\Theta(\log_{\mathrm{d}} \log n)$, the deterministic complexity is $\Theta(\log_{\mathrm{d}} n)$.

Backup Slides

The Constructive LLL

- Each E_i shares variables with at most d other events
- $\Pr(E_i) < p$ where $ep(d+1) \leq 1$

$\Rightarrow$ An assignment of the random variables that avoids all bad events can be found efficiently

Moser and Tardos, 2010

- Example: X_i binary

$$(X_1 \vee \neg X_2) \wedge (\neg X_1 \vee X_3 \vee \neg X_4) \wedge (X_2 \vee X_4) \wedge (\neg X_3 \vee X_4)$$

vbl$(E_1) = \{X_1, X_2\}$, vbl$(E_2) = \{X_1, X_3, X_4\}$,
vbl$(E_3) = \{X_2, X_4\}$, vbl$(E_4) = \{X_3, X_4\}$

$d = 3,\ p = \frac{1}{4}$

The Dependency Graph

- Nodes: events
- Edges: the events share a variable

- Example:

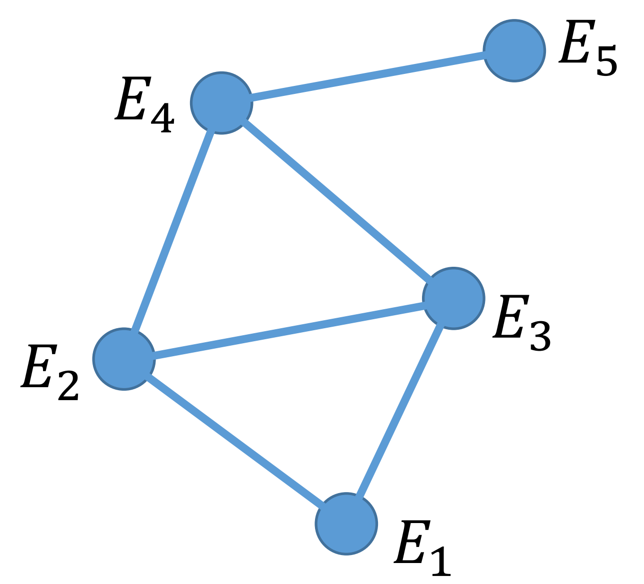
  $\mathrm{vbl}(E_1) = \{X_1\}$

  $\mathrm{vbl}(E_2) = \{X_1, X_2\}$

  $\mathrm{vbl}(E_3) = \{X_1, X_3\}$

  $\mathrm{vbl}(E_4) = \{X_2, X_3, X_4\}$

  $\mathrm{vbl}(E_5) = \{X_4\}$



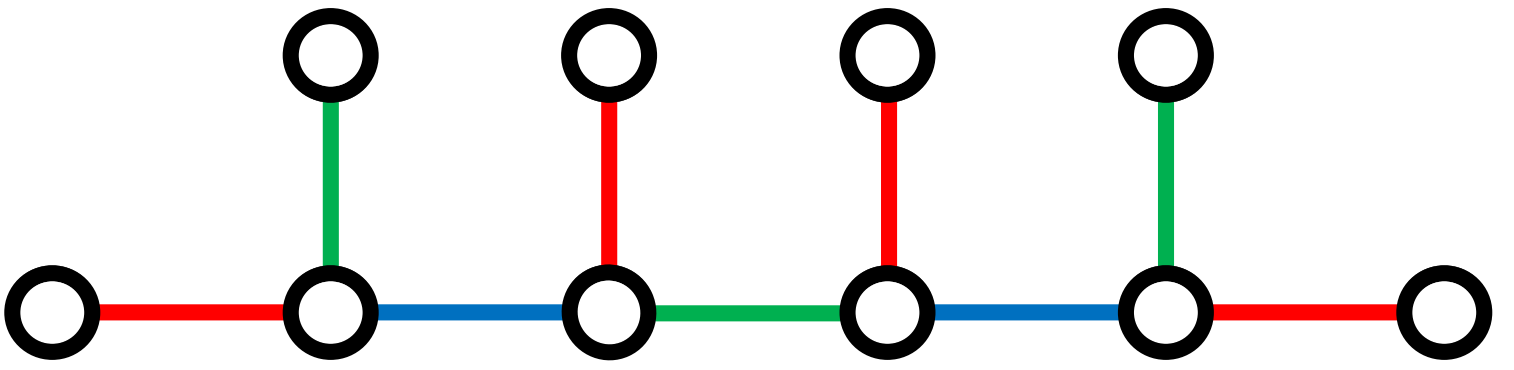
- Maximum degree d

Distributed Computing

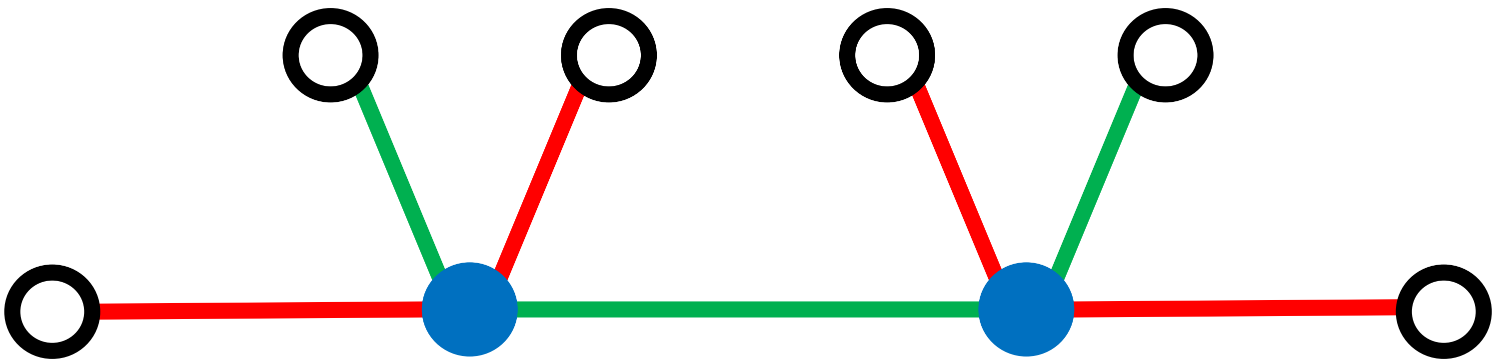
- Input: simple undirected graph (+ some task-specific input)
- Nodes: computational entities
- Edges: communication channels

- Synchronous rounds
- In each round, each node ...
    1) sends an arbitrarily large message to each neighbour
    2) receives sent messages
    3) performs local computations
- Each node has to output a correct answer
- Time complexity: number of rounds (worst-case input)

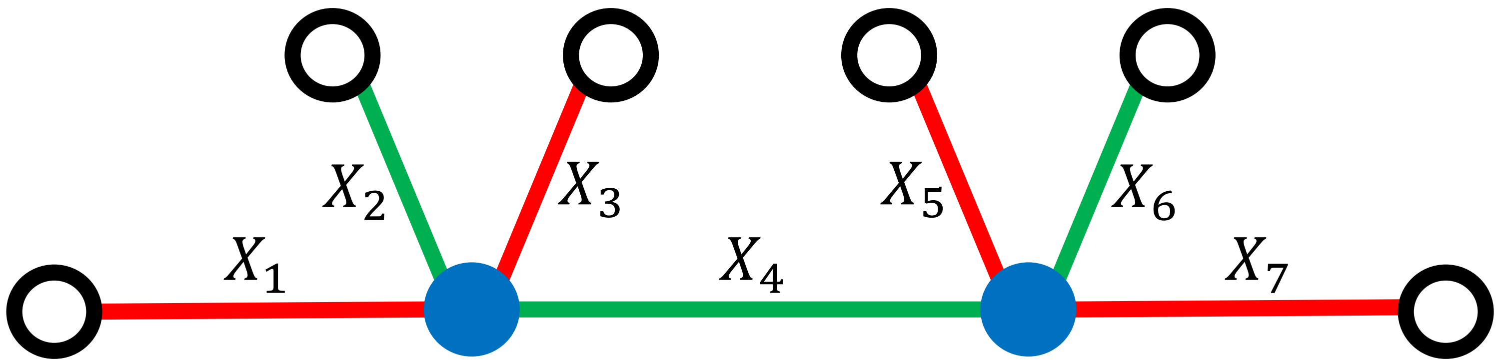
Reduction from SO to LLL



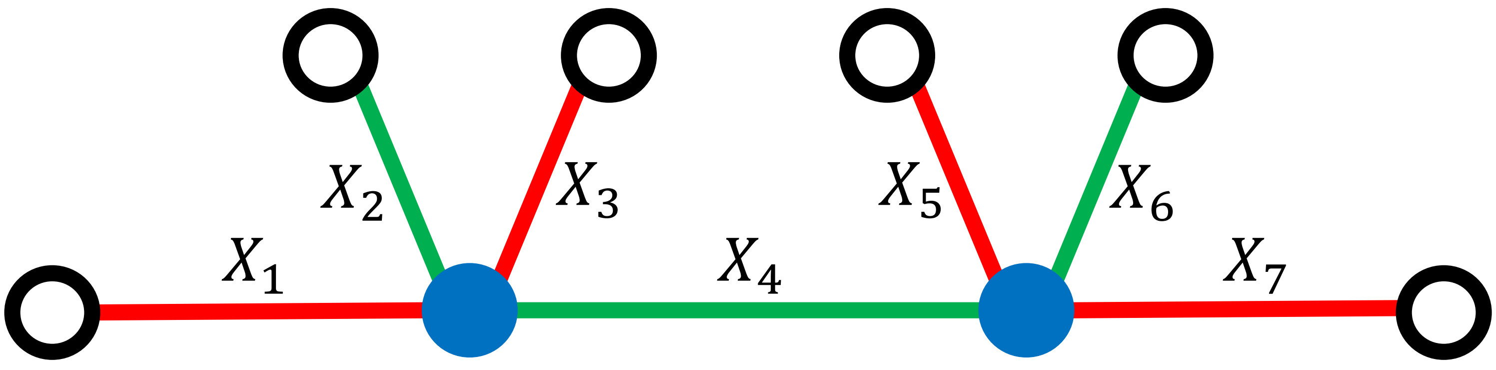
Reduction from SO to LLL



Reduction from SO to LLL

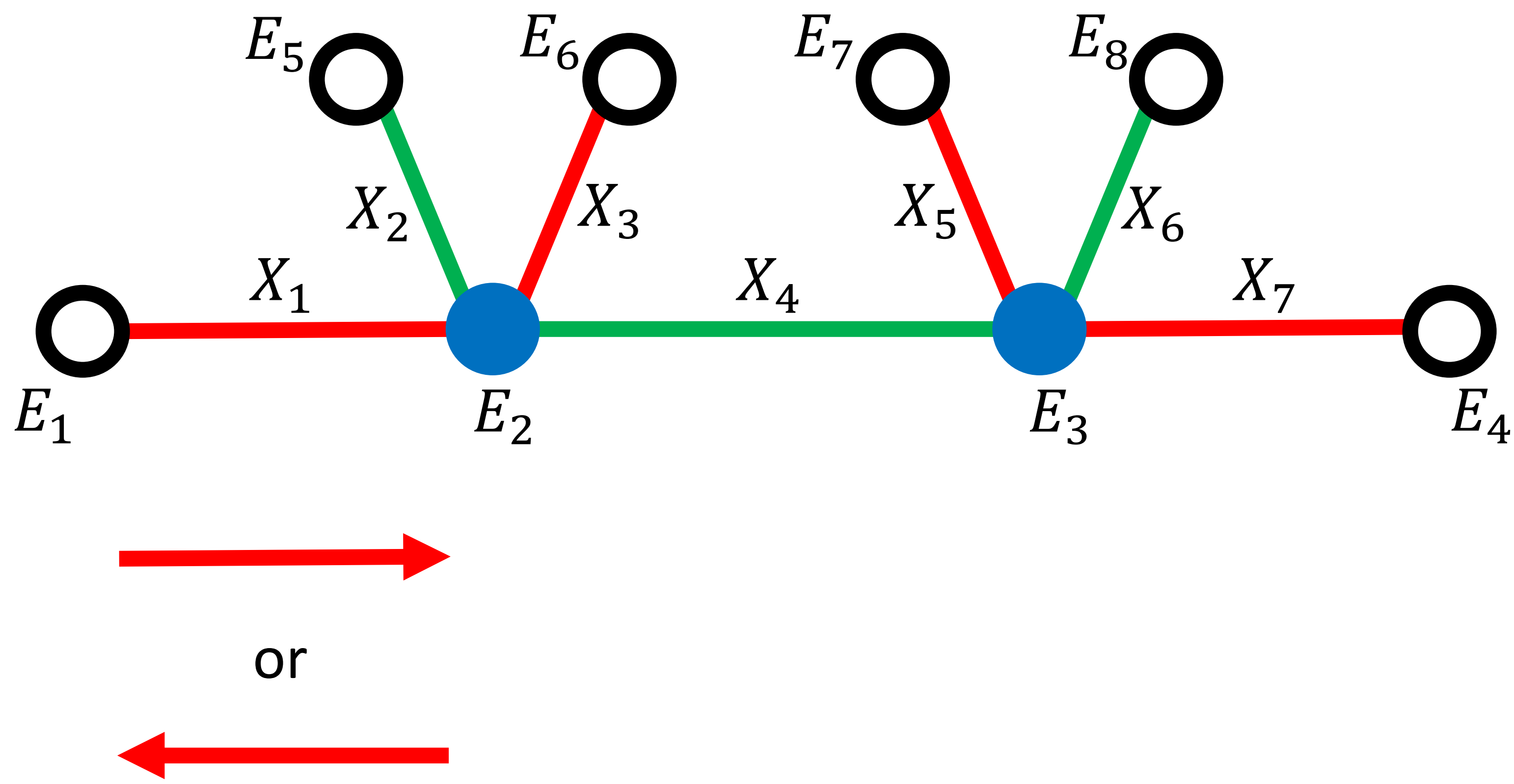


Reduction from SO to LLL

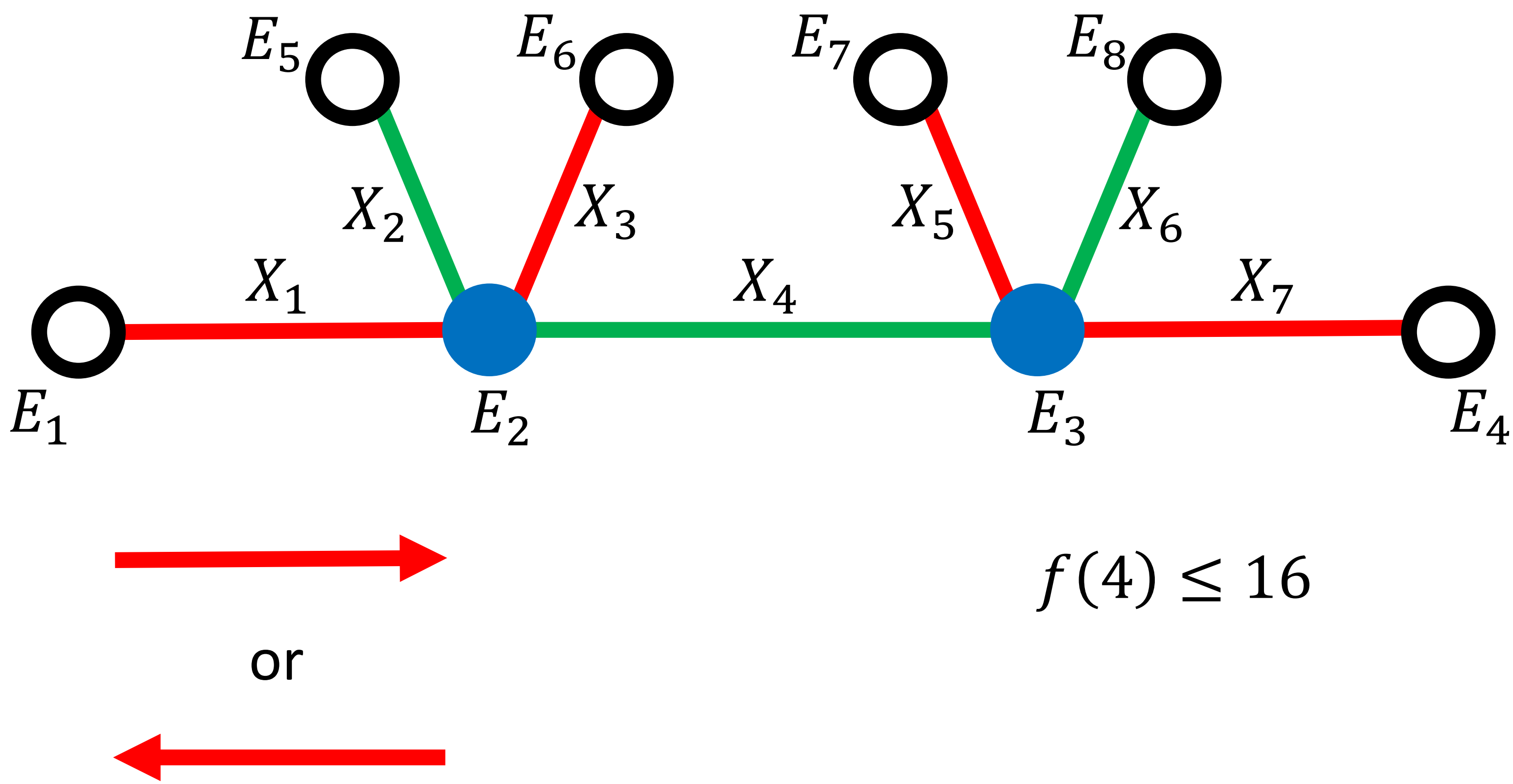


or

Reduction from SO to LLL

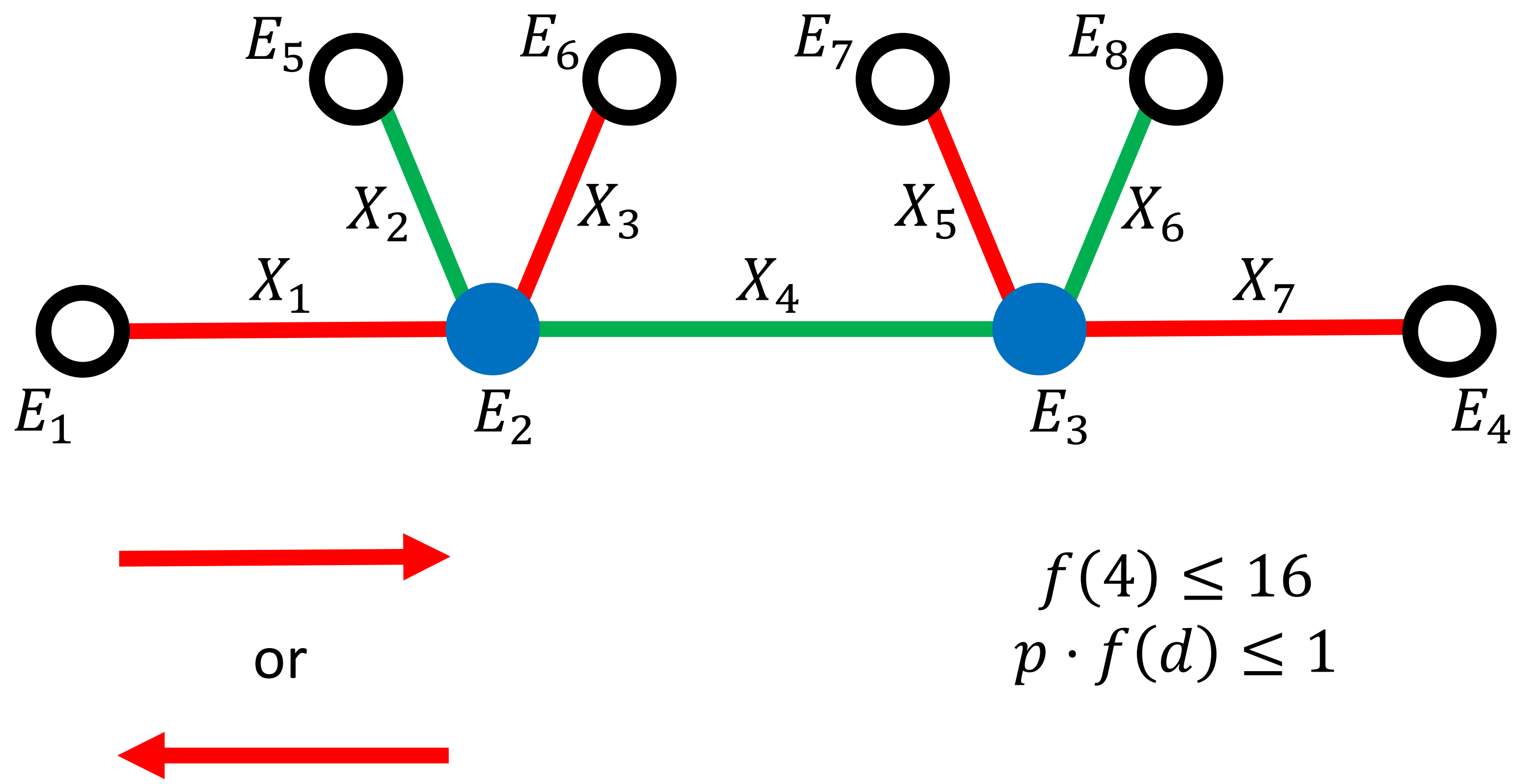


Reduction from SO to LLL



$f(4) \leq 16$

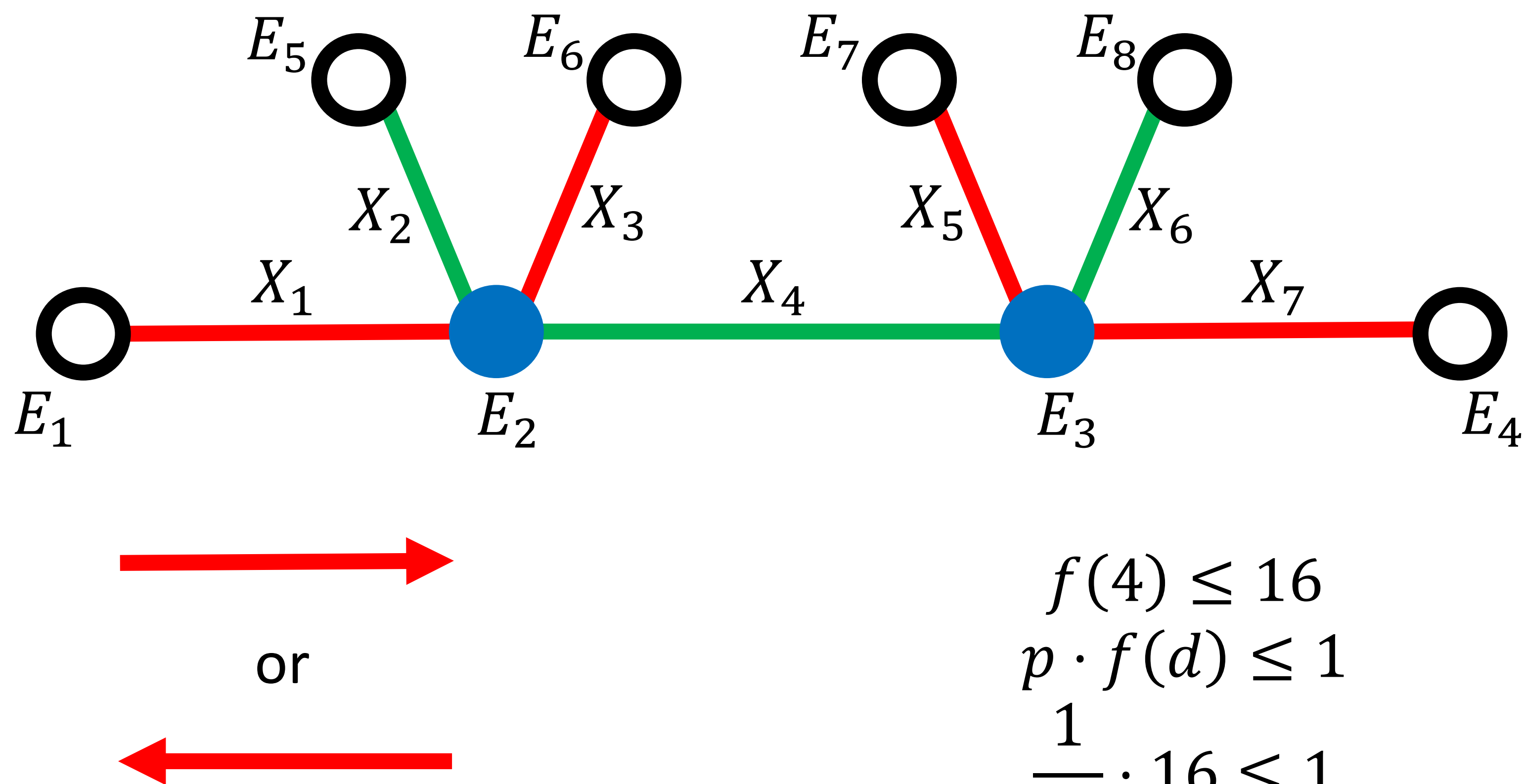
Reduction from SO to LLL



$f(4) \leq 16$

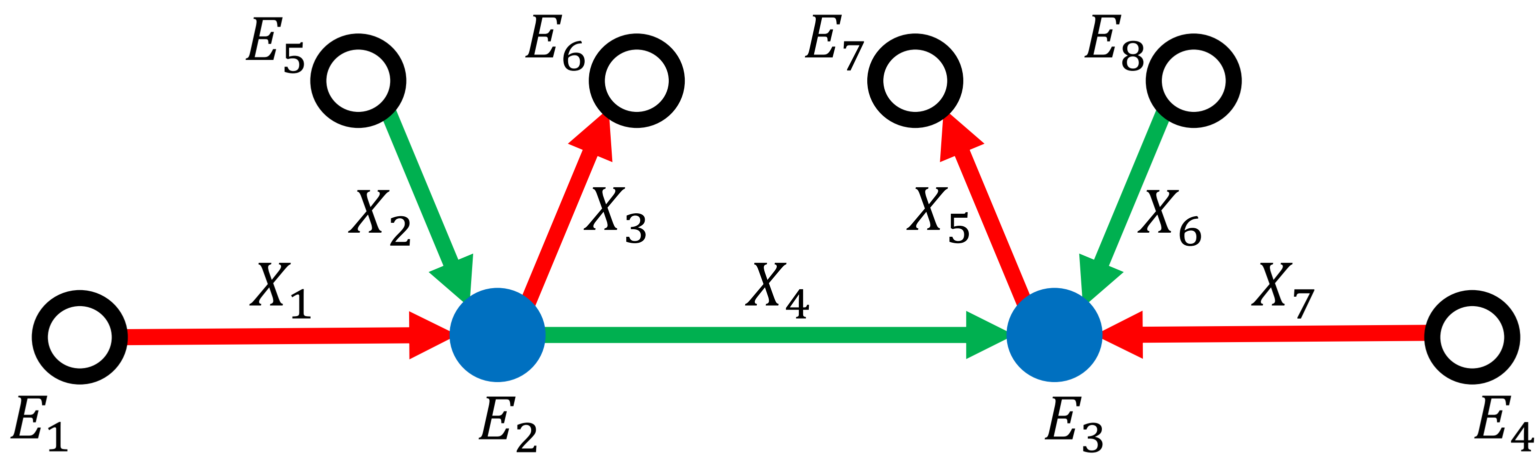
$p \cdot f(d) \leq 1$

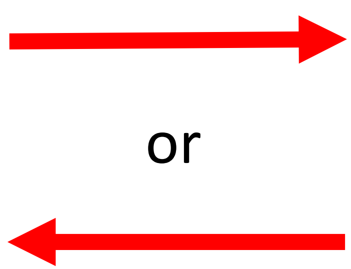
Reduction from SO to LLL



$$f(4) \leq 16$$
$$p \cdot f(d) \leq 1$$
$$\frac{1}{16} \cdot 16 \leq 1$$

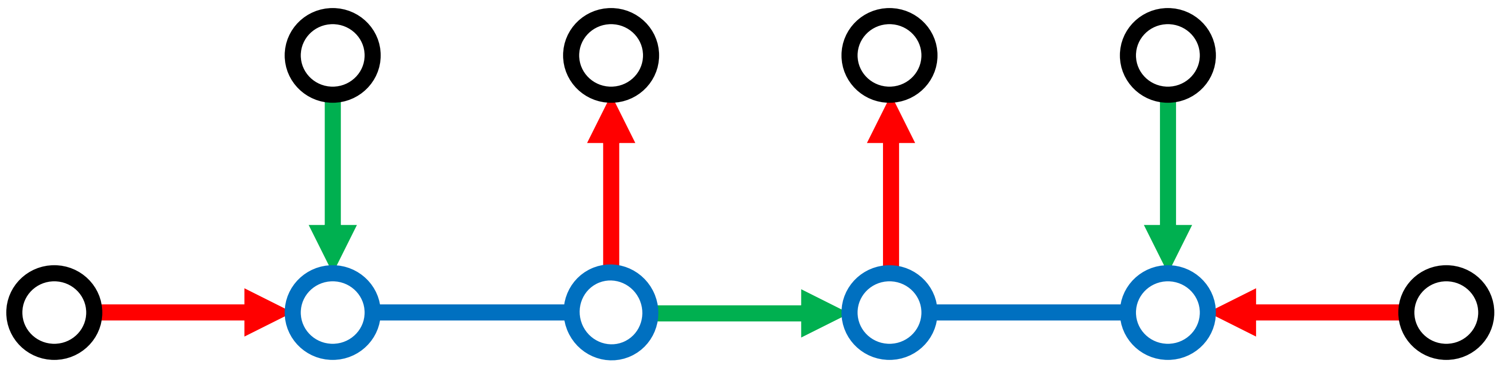
Reduction from SO to LLL



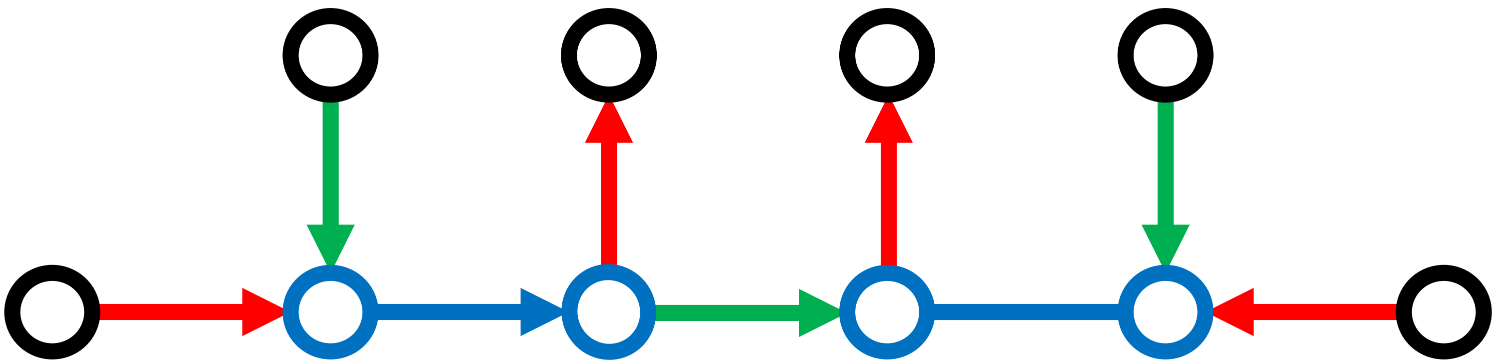


$$f(4) \leq 16$$
$$p \cdot f(d) \leq 1$$
$$\frac{1}{16} \cdot 16 \leq 1$$

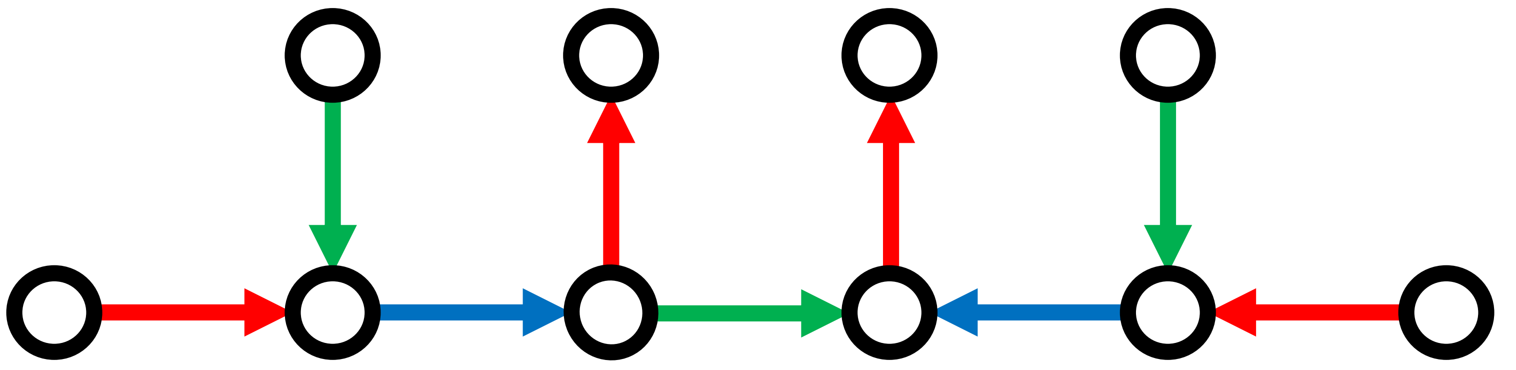
Reduction from SO to LLL



Reduction from SO to LLL



Reduction from SO to LLL



Technicalities

- Monte-Carlo algorithms, w.h.p.
- Girth $\geq 2t + 1$
- $d = 3$
- Failure probability $p_f(v)$ resp. $p_f(e)$