Decentralized Online Convex Optimization

joint work with Tim van Erven (UvA) and Dirk van der Hoeven (Leiden/Milan)

Hédi Hadiji (University of Amsterdam), à Nantes le 22/02/2022

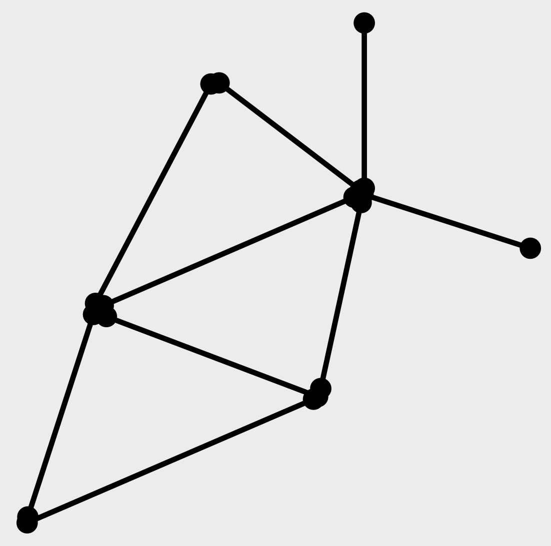
Introduction

Online learning/Online optimization

- Data/Objective coming in a stream, as the optimization is made

Federated learning

- Multiple agents collaborating to learn



Adaptive algorithms

- As little manual tuning as possible

(Unconstrained) Online Convex Optimization

Setting Zinkevich '03, McMahan Streeter '12, Orabona '19, Hazan '19

Adversary prepares a sequence of convex loss functions $\ell_t : \mathbb{R}^d \to \mathbb{R}$
At every time step:

- Player picks action $w_t \in \mathbb{R}^d$
- Adversary reveals loss ℓ_t

Minimize **regret** $$R_T(u) = \sum_{t=1}^{T} \ell_t(w_t) - \ell_t(u) \leqslant \sum_{t=1}^{T} \langle w_t - u, \nabla \ell_t(w_t) \rangle$$

Online Convex/Linear Optimization

Examples (see e.g. Cesa-Bianchi, Lugosi ’06; Hazan ’19)

- Prediction with expert advice. Actions: d-simplex, linear losses
- Online (supervised) learning: choose w_t to predict y_t, suffer loss $\ell(w_t, y_t)$
- Convex/Stochastic optimization $\ell_t = F(\cdot, \xi_t)$
- Portfolio selection, applications to boosting, learning equilibria in repeated games, etc.

- Generalizations: partial information, non-stationary regret, robustness, delays, …

Main algorithm: Online Gradient Descent

Fixed step-size analysis (Zinkevich ’03)

At time $t+1 \geqslant 2$,

- receive ℓ_t , compute $g_t = \nabla \ell_t(w_t)$
- play $w_{t+1} = w_t - \eta g_t$

Parameters:
- step-size $\eta > 0$
- $w_1 = 0$

$$R_T(u) \leqslant \frac{\|u\|^2}{2\eta} + \frac{\eta}{2} \sum_{s=1}^{T} \|g_t\|^2$$

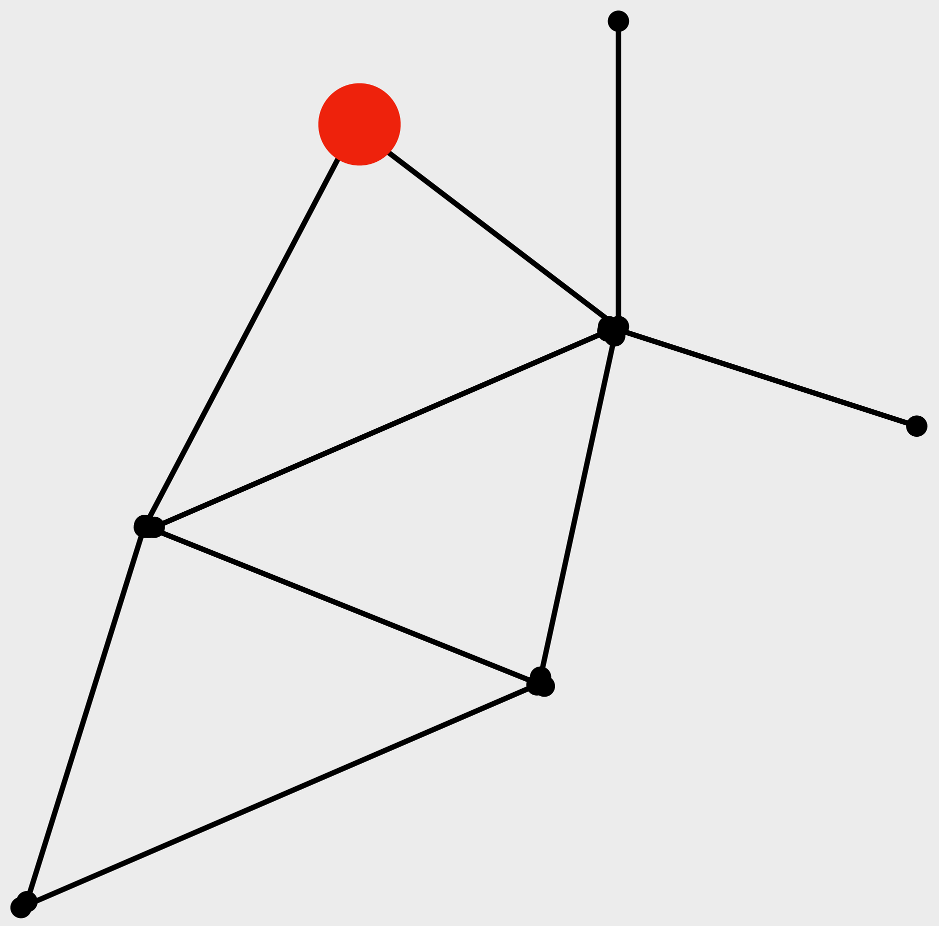
if $\|u\| \leqslant U$ and $\|g_t\| \leqslant G$, then setting $\eta = G/(U\sqrt{T})$

$R_T(u) \leqslant UG\sqrt{T}$ worst-case optimal

Decentralized OCO

Given graph $\mathcal{G}$, at every time step t,

- Adversary picks node n_t
- Node n_t picks action $w_t \in \mathbb{R}^d$
- Adversary reveals convex loss function $\ell_t : \mathbb{R}^d \to \mathbb{R}$
- All nodes communicate with neighbors



Minimize **joint regret**

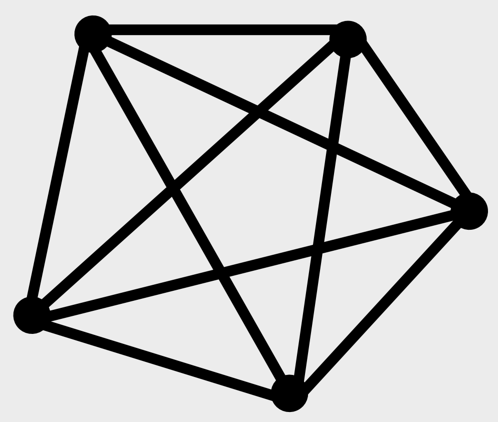
$$R_T(u) = \sum_{t=1}^{T} \ell_t(w_t) - \ell_t(u)$$

Related: Decentralized Optimization and Gossip

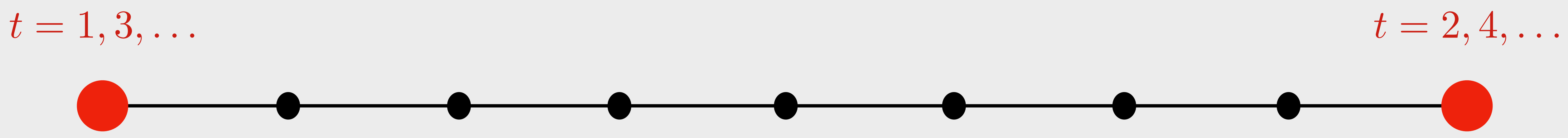
Hsieh et al. ’20; Cesa-Bianchi et al. ’20;

Special Cases

- Complete graph ⇔ One single player



- D-line with activation alternating at endpoints



~D/2 losses are missing at active node

What happens to Gradient Descent?

Natural idea: every node subtracts $-\eta g$ for every new gradient g observed

Let $w_t^\star$ be the updates of oracle GD that knows all gradients

$$\sum_{t=1}^{T}\langle w_t - u, g_t\rangle = \sum_{t=1}^{T}\langle w_t^\star - u, g_t\rangle + \sum_{t=1}^{T}\langle w_t - w_t^\star, g_t\rangle$$

Regret of oracle GD

$$w_t - w_t^\star = \eta \sum_{s\in\gamma(t)} g_s$$

$$R_T \leqslant \frac{\|u\|^2}{2\eta} + \frac{\eta}{2}\sum_{t=1}^{T}\left(\|g_t\|^2 + 2\|g_t\| \sum_{s\in\gamma(t)} \|g_s\|\right)$$

Decentralized GD II

$$R_T \leqslant \frac{\|u\|^2}{2\eta} + \frac{\eta}{2}\sum_{t=1}^{T}\left(\|g_t\|^2 + 2\|g_t\|\sum_{s\in\gamma(t)}\|g_s\|\right)$$

At most $D(\mathcal{G}) - 1$ gradients are missing $|\gamma(t)| \leqslant D(\mathcal{G}) - 1$

$$R_T(u) \leqslant \frac{\|u\|^2}{2\eta} + \frac{\eta}{2}G^2\sum_{t=1}^{T}(1 + 2|\gamma(t)|) \leqslant \frac{\|u\|^2}{2\eta} + \frac{\eta}{2}G^2(2D(\mathcal{G}) - 1)T$$

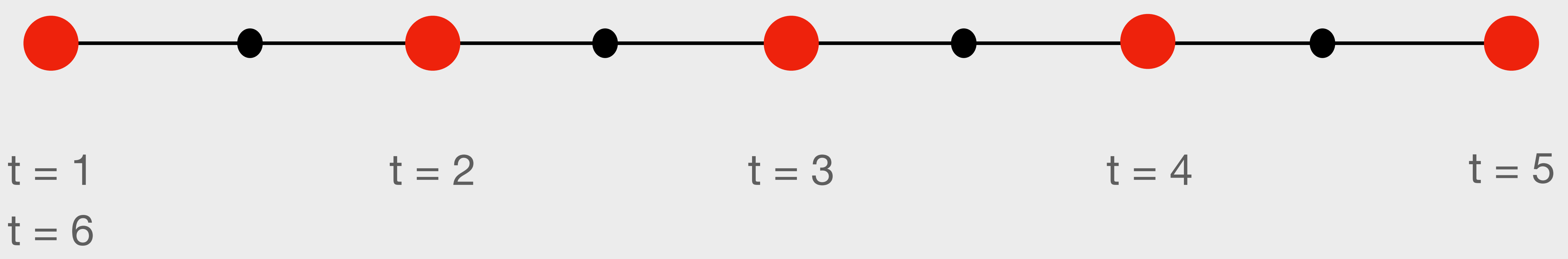
$R_T \lesssim UG\sqrt{D(\mathcal{G})T}$ worst-case optimal

Worst-case Activation Sequence

Theorem: For any graph, for any algorithm, there exists an activation sequence and losses such that

$$\max_{\|u\|\leqslant U} R_T \geqslant c\, UG\sqrt{TD(\mathcal{G})}$$

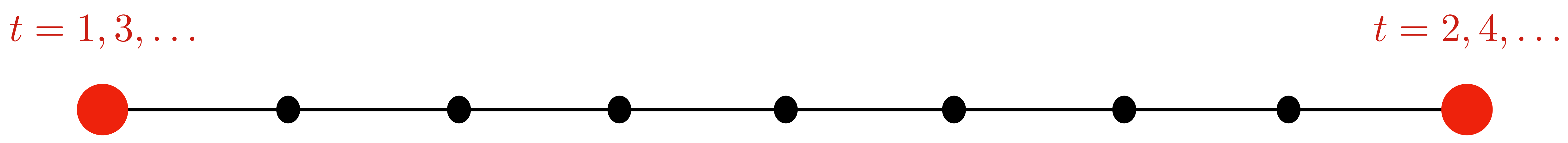
Proof: Pick a maximal-length path



Adversary can play the same gradients D / 2 times

But might be suboptimal for specific cases

- Recall the line



- **Ignoring missing gradients:** $R_T \leqslant UG\sqrt{2T} \ll UG\sqrt{D(\mathcal{G})T}$
- But ignoring missing gradients is bad in general (up to $UG\sqrt{|\mathcal{N}|T}$)

How to adapt to the activation sequence?

Comparator-Adaptive Algorithms

also called parameter-free, or model selection type-bounds

Theorem : There is an algorithm for Decentralized-OCO s.t. for user-specified $B > 0$

$$R_T(u) \leqslant \|u\| G \sqrt{D(\mathcal{G}) T \log\left(1 + \frac{TG\|u\|}{B}\right)} + B$$ for any $u \in \mathbb{R}^d$, $T > 0$ and $\mathcal{G}$

The simpler the comparator is, the smaller the regret bound

In particular, $R_T(0) \leqslant B$

In OCO: McMahan Streeter '12; Orabona; Cutkosky; Koolen, Mhammedi and van Erven '19; Foster et al. '18;

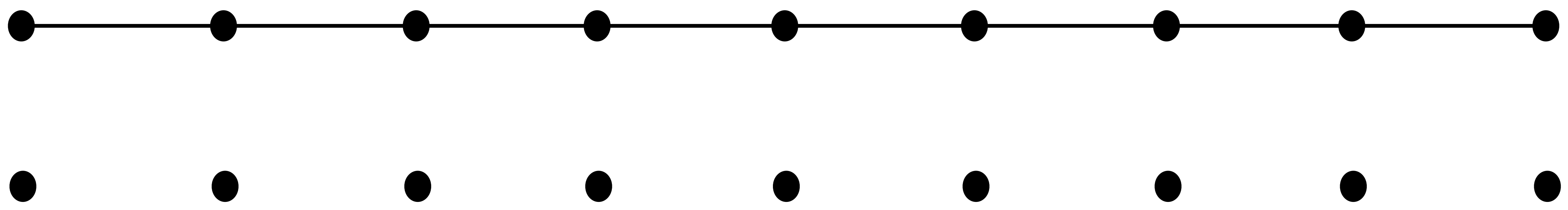
Iterate Addition

Cutkosky '19

Back to the line example

- Each node keeps two algorithms:

  $w_t^{(G)}$: iterates of $\mathcal{A}(\mathcal{G})$
  $w_t^{(n)}$: iterates of $\mathcal{A}(\{n\})$



- and active node n_t plays $\boxed{w_t^{(n_t)} + w_t^{(\mathcal{G})}}$

Iterate Addition II

Adding iterates guarantees both

$$\sum_{t=1}^{T}\langle w_t^{(n_t)}, g_t\rangle + \sum_{t=1}^{T}\langle w_t^{(\mathcal{G})} - u, g_t\rangle$$

$$\sum_{n\in\mathcal{N}} R|_{\{n\}}(0) + R_T(u)$$

$$|\mathcal{N}|B + \|u\|G\sqrt{D(\mathcal{G})T\log\left(1+\frac{T\|u\|G}{B}\right)} + B$$

(almost) worst-case optimal

$$\sum_{t=1}^{T}\langle w_t^{(n_t)} - u, g_t\rangle + \sum_{t=1}^{T}\langle w_t^{(\mathcal{G})}, g_t\rangle$$

$$\sum_{n\in\mathcal{N}} R|_{\{n\}}(u) + R_T(0)$$

$$\sum_{n\in\mathcal{N}} \|u\|G\sqrt{T^{(n)}\log\left(1+\frac{T^{(n)}\|u\|G}{B}\right)} + |\mathcal{N}|B + B$$

better when only one node is selected

More generally

Learning as well as the best $\mathcal{Q}$-partition

- Given a collection of subgraphs $\mathcal{Q}$ play

$$w_t = \sum_{\mathcal{H} \in Q} w_t^{\mathcal{H}} \mathbf{1}\{n_t \in \mathcal{H}\}$$

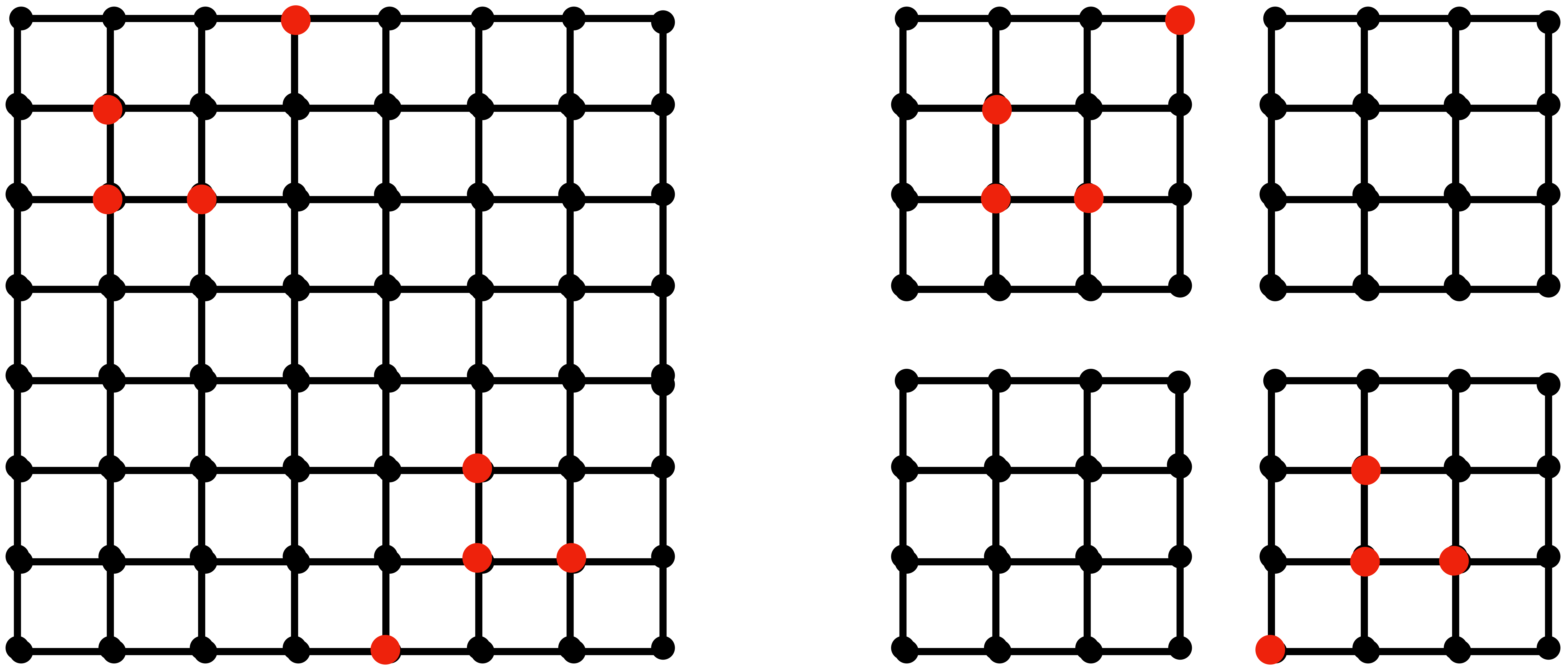
- **For any partition** $\{\mathcal{F}\}$ of the graph made of elements in $\mathcal{Q}$

$$R_T(u) \leqslant \sum_{\mathcal{F}} R|_{\mathcal{F}}(u) + \big(|\mathcal{Q}| - |\{\mathcal{F}\}|\big) B$$

$$\leqslant \|u\| G \sum_{\mathcal{F}} \sqrt{D(\mathcal{F}) T^{(\mathcal{F})} \log\left(1 + \frac{T^{(\mathcal{F})} \|u\| G}{B}\right)} + |Q| B$$

More generally

Learning as well as the best $\mathcal{Q}$-partition



What's more

$$R_T(u) \lesssim \|u\|\left(\sqrt{\Lambda_T \ln\left(1 + \frac{\|u\|\Lambda_T}{B}\right)} + D(\mathcal{G})G\right) + B$$

where $\Lambda_T = \sum_{t=1}^{T} \|g_t\|^2 + 2\|g_t\| \sum_{s \in \gamma(t)} \|g_s\|$

In the paper

- Adapt to small gradients
- Limited communication bandwidth: nodes can send k-bit messages

In the future

- Relax the synchronisation assumptions
- Study more in depth more efficient ways to communicate gradients
- Computational complexity? Reducing the number of algorithms maintained?