OPTIMALITY IN DECENTRALIZED OPTIMIZATION UNDER BANDWIDTH CONSTRAINTS

Anonymous authors
Paper under double-blind review

ABSTRACT

We consider a realistic decentralized setup with *bandwidth-constrained communication* and derive *optimal time complexities* for non-convex stochastic parallel and asynchronous optimization (up to logarithmic factors). We develop the corresponding methods, Grace SGD and Leon SGD, for both homogeneous and heterogeneous settings. Unlike previous work, our optimal bounds are characterized in terms of *min-cut/max-flow* quantities and rely on tools from *Gomory–Hu trees* and *Steiner Tree Packing* problems, providing tighter and more practical complexities.

1 INTRODUCTION

We consider a decentralized distributed optimization setup with n workers, such as GPUs, CPUs, servers, or mobile devices (Kairouz et al., 2021), that aim to solve a common optimization problem by computing stochastic gradients and sharing this information with each other through a communication network. We study a smooth nonconvex minimization problem defined as

$$\min_{x\in\mathbb{R}^d} f(x), \tag{1}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ and d is the dimension of f. In this work, we assume that d is large, which is the case in large language models and modern machine learning training (Brown et al., 2020; Grattafiori et al., 2024). We begin with the *homogeneous* (i.i.d.) setting, where all workers compute stochastic gradients sampled from the same distribution. But we discuss our implications and also study *heterogeneous* (non-i.i.d.) settings in Sections 5 and 6.

Assumption 1.1. f is differentiable & L–smooth, i.e., $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \forall x, y \in \mathbb{R}^d$.

Assumption 1.2. There exist $f^* \in \mathbb{R}$ such that $f(x) \geq f^*$ for all $x \in \mathbb{R}^d$. We define $\Delta := f(x^0) - f^*$, where x^0 is a starting point.

Assumption 1.3 (Homogeneous setting)**.** For all $i \in [n]$, worker i can only calculate $\nabla f(x;\xi)$ and $\mathbb{E}_\xi[\nabla f(x;\xi)] = \nabla f(x)$ and $\mathbb{E}_\xi[\|\nabla f(x;\xi) - \nabla f(x)\|^2] \leq \sigma^2$ for all $x \in \mathbb{R}^d$, where $\sigma^2 \geq 0$.

In the nonconvex optimization, the goal is to find an ε–stationary point, a (random) point $\bar{x} \in \mathbb{R}^d$ such that $\mathbb{E}[\|\nabla f(\bar{x})\|^2] \leq \varepsilon$. It is well-known that the optimal *oracle complexity* in this setting is $\Theta(L\Delta/\varepsilon + \sigma^2 L\Delta/\varepsilon^2)$ (Arjevani et al., 2022) achieved by the classical SGD method (Lan, 2020).

Computation times. To present our time complexities and new algorithms, and to compare them with previous results in the distributed setup, we consider the following computation model:

> **Computation Model.** Worker i requires h_i seconds[1] to compute a stochastic gradient.

This is a standard and natural assumption in modern optimization (Mishchenko et al., 2022; Tyurin & Richtárik, 2023), as it allows us to compare parallel and asynchronous methods. Looking ahead, even when $h_i = h$ for all $i \in [n]$, our results remain new and relevant.

Communication times. Our main goal is to derive the optimal time complexity in the setup where communication times cannot be ignored. To the best of our knowledge, this work considers a new

[1] We can even assume that the computation times are not fixed to h_i and can vary in the interval $[\bar{c}_l h_i, \bar{c}_r h_i]$, where $\bar{c}_l$ and $\bar{c}_r$ are some constants (e.g., $\bar{c}_l = 0.9, \bar{c}_r = 1.1$), but this would not change the proved asymptotics.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

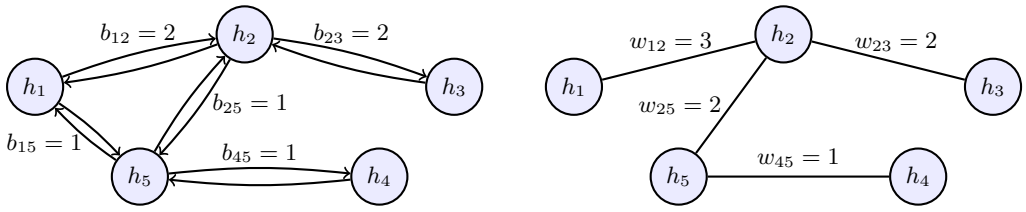


Figure 1: Left: communication graph G. Node i has computation time h_i; edges denote bidirectional links with bandwidth $b_{ij} = b_{ji}$. Right: a *Gomory–Hu tree* T of the undirected version of G (Definition E.6), *the central tool for the design of optimal decentralized optimization methods.*

communication assumption motivated by practical bottlenecks. For example, consider two identical GPUs with the same computation time h, connected by a communication link. The main bottleneck is the bandwidth b, i.e., the number of coordinates per second they can transmit to each other.

Graph-Bandwidth Communication Model. Motivated by the previous example, in general, we assume that there exists a directed connected weighted graph $G = (V, E, b)$ with $n = |V|$ vertices and $|E|$ edges. Each vertex $i \in V$ represents a worker, and each ordered edge $(i, j) \in E$ with weight $b_{ij} > 0$ represents a communication link between workers i and j. The weight/bandwidth b_{ij} represents the number of coordinates per second that worker i can send to worker j via edge (i, j). We assume that if $(i, j) \in E$, then $(j, i) \in E$ and $b_{ij} = b_{ji}$, meaning that the communication is bidirectional: workers can send and receive messages from each other at the same time. It is also convenient to define an *undirected* version of G, i.e., the undirected graph $\bar{G} = (V, \bar{E}, b)$ where $\{i, j\} \in \bar{E}$ with weight b_{ij} if and only if $(i, j) \in E$ and $(j, i) \in E$ with weight b_{ij}.

Without loss of generality (w.l.o.g.), we assume the graph is connected; if not, one should consider each connected component separately. We also assume that coordinates are infinitely divisible and can be transmitted continuously as a fluid, which is true in theory since one coordinate/number is an infinite sequence of bits. In Section L, we discuss this assumption, consider a model with communication latencies, and discuss how the model relates to previous work.

Assumption 1.4. The optimization environment satisfies Computation Model and Graph-Bandwidth Communication Model. All other operations, such as in-node aggregation, vector splitting, and other local computations (except for stochastic gradient computations), are assumed to take negligible time.

We emphasize that our setup allows freedom in how communication is performed and is more flexible than, for instance, the gossip protocol (Boyd et al., 2006). Workers can overlap communication and computation and are allowed to send and receive through all edges asynchronously. Examples:
Split vector: In Figure 1, worker 5 could send a vector of size d to worker 2 directly in d/b_{25} seconds, but it is faster to split the vector into two parts and route one part through worker 1, yielding time $\max\{d/\mathbf{2}b_{25}, d/\mathbf{2}\min\{b_{15}, b_{12}\}\}$, which is two times faster.
Online in-network aggregation: Assume that all workers want to *reduce* their vectors at worker 4. A naive approach that sends the vectors separately is bottlenecked by b_{45}, requiring $\mathbf{4}d/b_{45}$ seconds. A better strategy aggregates the vectors at worker 5 and forwards coordinates immediately as they arrive, yielding time d/b_{45}. See another example in Section H.
Interleaving: Worker 4 needs to send two vectors, one to worker 1 and one to worker 2. Instead of sending them sequentially, it may *interleave* transmissions: alternating coordinates of the two vectors. Worker 5 then forwards each coordinate to the appropriate destination.

1.1 Known methods

We now describe some known methods for decentralized distributed optimization. Related work under previous optimization setups are discussed in Section A.
Synchronous SGD. One of the most standard ways to solve the decentralized distributed optimization problem is to use Synchronous SGD: $x^{k+1} = x^k - \frac{\gamma}{n}\sum_{i=1}^n \nabla f(x^k; \xi_i^k)$, where ξ_i^k is a random variable sampled from the distribution $\mathcal{D}_\xi$ at worker i. In this method, all workers compute the gradients in parallel, and then the algorithm averages the gradients and updates the iterate. It is

well-known that this method finds an ε–stationary point after $\mathcal{O}\left(L\Delta/\varepsilon + \sigma^2 L\Delta/n\varepsilon^2\right)$ *iterations* (Lan, 2020). At the same time, under Assumption 1.4, the *time complexity* of this method is

$$\mathcal{O}\left(\left(\frac{d}{b_{\min}} + h_{\max}\right)\left(\frac{L\Delta}{\varepsilon} + \frac{\sigma^2 L\Delta}{n\varepsilon^2}\right)\right) \tag{2}$$

seconds, where $h_{\max} := \max_{i\in[n]} h_i$ comes from the fact that every worker computes one stochastic gradient, waiting for the slowest one. We define $b_{\min} := \min_{(i,j)\in E} b_{ij}$. The term $d/b_{\min}$ is a time to collect all stochastic gradients at a one worker, update the iterate, and broadcast x^{k+1} to all workers, using the *online in-network aggregation* discussed in Section 1 (see details in Section H)

Hero SGD. Another strategy is to simply run SGD on the fastest worker. In this case, the time complexity is $\mathcal{O}\left(h_{\min}\left(L\Delta/\varepsilon + \sigma^2 L\Delta/\varepsilon^2\right)\right)$, where $h_{\min} := \min_{i\in[n]} h_i$. As expected, this time complexity does not scale with n; nevertheless, it can be better than (2) if $d/b_{\min}$ or $h_{\max}$ are large.

Gossip methods. Another important family of decentralized methods is gossip-based methods (Boyd et al., 2006), where each worker communicates vectors with its neighbors. These methods are typically analyzed by deriving *iteration convergence rates* that depend on the spectral gap of a mixing matrix. As for Synchronous SGD and Hero SGD, it is possible to derive the time complexities of these methods; however, as we will show later, we present a *lower bound* (with a matching algorithm) that virtually none of these methods can break.

Compressed communication. Instead of sending the full vectors, it is possible to use lossy compression techniques (Seide et al., 2014; Alistarh et al., 2017), where the idea is to use different sparsification and quantization methods to reduce the number of coordinates/bits sent through the communication links (Beznosikov et al., 2020).

Main problem: There are many ways, from different centralized and decentralized optimization fields, to construct methods with time complexities that depend on the parameters in Assumptions 1.1, 1.2, 1.3, and 1.4. This naturally raises the central question of the paper: *what is the fastest possible method in this setting, and can we establish a matching lower bound?*

1.2 Contributions

♠ Our first main contribution is a new method, Grace SGD (Algorithm 1), that achieves an *optimal time complexity*, up to logarithmic factors. To design this method and obtain the discovered complexity, we developed two important new subroutines, Algorithms 2 and 3, which use tools from the *Gomory-Hu* trees (Gomory & Hu, 1961) and the *Steiner Tree Packing* problem (Lau, 2004) fields, and may be of independent interest. We now present the theorem explained in Section 2.

Theorem 1.5 (Upper Bound in the Homogeneous Setting). *Assume that Assumptions 1.1, 1.2, and 1.3 hold. Then Grace SGD (Algorithm 1) finds an ε-stationary point of (1) after $K = \lceil 4L\Delta/\varepsilon \rceil$ iterations, and the time complexity under Assumption 1.4 is*

$$\mathcal{O}\left(\min_{k\in[n]}\left(\frac{d}{\bar{w}_k} + \min_{p\in[k]}\min_{m\in[|S_{k,p}|]}\left[\left(\frac{1}{m}\sum_{i=1}^{m}\frac{1}{h_{\pi_i(S_{k,p})}}\right)^{-1}\left(1 + \frac{\sigma^2}{m\varepsilon}\right)\right]\right)\frac{L\Delta}{\varepsilon}\right) \tag{3}$$

up to a universal constant factor, where $\{S_{k,p}\}_{k\in[n],p\in[k]}$ and $\{\bar{w}_k\}_{k\in[n]}$ are the sequences generated by Algorithm 2, and $\pi(S)$ is a permutation that sorts $\{h_i\}_{i\in S} : h_{\pi_1(S)} \le \cdots \le h_{\pi_{|S|}(S)}$.

Corollary 1.6 (Equal computation times). *In view of Theorem 1.5, if $h_i = h$ for all $i \in [n]$, then the time complexity of Grace SGD is*

$$\mathcal{O}\left(\min_{k\in[n]}\left(\frac{d}{\bar{w}_k} + \min_{p\in[k]}\frac{h\sigma^2}{|S_{k,p}|\varepsilon}\right)\frac{L\Delta}{\varepsilon} + \frac{hL\Delta}{\varepsilon}\right). \tag{4}$$

The complexities depend on new quantities, $\bar{w}_k$ and $S_{k,p}$: $\bar{w}_k$ is the minimum s–t cut value in G for some $s, t \in V$, corresponding to an edge of the Gomory–Hu tree T, and $S_{k,p}$ is a subset of V whose communication bottleneck is at least $\bar{w}_k$. Intuitively, $\bar{w}_k$ reflects communication within $S_{k,p}$, and Grace SGD chooses k to balance $\bar{w}_k$ and the number of workers $\max_{p\in[k]} |S_{k,p}|$: the more workers compute stochastic gradients, the longer synchronization takes, and (3) and (4) capture the balance.

♣ While Theorem 1.5 may appear non-intuitive and semi-explicit, it is in fact *fundamental and optimal up to logarithmic factors*. Despite the flexibility and generality of the setup, it is possible to prove a matching lower bound using new techniques for a broad family of methods (zero-respecting),

Algorithm 1 Grace SGD

Require: Graph $G = (V, E, b)$, times $\{h_i\}_{i\in[n]}$, ratio σ^2/ε, smoothness constant L, point x^0

1: Find the fastest subset of workers S^* using Algorithm 2 **(crucial new step)**
2: Workers S^* locally initialize starting point with x^0 and step size $\gamma = 1/2L$
3: **for** $k = 0, 1, \ldots$ **do**
4: Workers in S^* compute $\sum_{j=1}^{B_i} \nabla f(x^k; \xi_{ij}^k)$ in parallel, increasing local batch sizes B_i over time, until $\sum_{i\in S^*} B_i = \max\{\lceil \sigma^2/\varepsilon \rceil, 1\}$
5: Run optimal-bandwidth AllReduce (e.g., Algorithm 3) so workers in S^* obtain $\sum_{i\in S^*} \sum_{j=1}^{B_i} \nabla f(x^k; \xi_{ij}^k)$ **(crucial new step)**
6: Every worker in S^* updates the iterate: $x^{k+1} = x^k - \frac{\gamma}{\sum_{i\in S^*} B_i} \sum_{i\in S^*} \sum_{j=1}^{B_i} \nabla f(x^k; \xi_{ij}^k)$
7: **end for**

including standard methods such as SGD, Asynchronous SGD (Recht et al., 2011), Local SGD (Zinkevich et al., 2010), and Adam (Kingma & Ba, 2015), as well as gossip methods and compressed methods via random sparsification. See Theorem 3.2 in Section 3.

♦ In Section 5, we also consider the *heterogeneous* setting, where workers have access to different distributions, and prove a new *optimal time complexity*, achieved by a new method called Leon SGD:

Theorem 1.7 (Upper Bound in the Heterogeneous Setting)**.** *Assume that Assumptions 1.1, 1.2, and 5.1 hold. Then Leon SGD (Algorithm 4) finds an ε-stationary point of (8) after $K = \Theta\left(L\Delta/\varepsilon\right)$ iterations, and the time complexity under Assumption 1.4 is*

$$\mathcal{O}\left(\max\left\{\frac{d}{\bar{w}_1}, \max_{i\in[n]} h_i, \frac{\sigma^2}{n\varepsilon}\left(\frac{1}{n}\sum_{i=1}^{n} h_i\right)\right\}\frac{L\Delta}{\varepsilon}\right) \tag{5}$$

seconds, where $\bar{w}_1 \equiv \min_{\{i,j\}\in F} w_{ij}$ is the minimum edge weight in the Gomory–Hu tree $T = (V, F, w)$ of $\bar{G}$. Equivalently, $\bar{w}_1$ is the global minimum cut value of G.

Unlike the homogeneous setting, Theorem 1.7 always depends on $\bar{w}_1$, the global minimum cut value, reflecting that the heterogeneous setting is inherently more challenging and is bottlenecked by the communication rate through the smallest cut[2]. This result is tight by our lower bound in Theorem G.12.

♥ In Section 6, we demonstrate an important corollary: for *sparse* graphs, Leon SGD and a naive mini-batch version of Synchronous SGD (Section 1.1) with complexity $\tilde{\Theta}(dL\Delta/b\varepsilon + h\sigma^2 L\Delta/n\varepsilon^2 + hL\Delta/\varepsilon)$ are optimal (up to logarithmic factors) when ε is small and n is large, even in the homogeneous setting. This means that a fully synchronized method that sends all stochastic vectors to one node is sufficient and *the first communication term does not scale with n* in sparse graphs. The main difficulty here was to show that this holds even in the homogeneous setting. In Corollaries 6.2 and F.3, we extend this observation and derive a *fundamental trade-off*. In Sections 4 and J, we discuss several other important examples and implications of our results for modern distributed optimization tasks.

2 Grace SGD: A New Method with Near-Optimal Time Complexity

We now present our new Algorithm 1, Grace SGD, designed for the practical Graph-Bandwidth Communication Model setup and achieving a near-optimal time complexity. Let us now explain how Grace SGD works. In a nutshell, Grace SGD is very simple: i) it finds the right subset of workers $S^* \subseteq [n]$ (Algorithm 2); ii) asks every worker $i \in S^*$ to calculate a mini-batch of stochastic gradients of size B_i such that the total batch size $\sum_{i\in S^*} B_i = \Theta(\sigma^2/\varepsilon)$; iii) runs an optimal-bandwidth AllReduce algorithm (e.g. Algorithm 3) so that all workers from S^* receive the whole batch $g^k = \sum_{i\in S^*} \sum_{j=1}^{B_i} \nabla f(x^k; \xi_{ij}^k) / \sum_{i\in S^*} B_i$; iv) every worker locally updates the iterate using the standard SGD step $x^{k+1} = x^k - \gamma g^k$. However, when it comes to the details, the main novelty lies in choosing S^*, implementing AllReduce in graph G in a right way, and proving that this whole scheme is indeed near-optimal. For clarity, assume that $h_i = h$ for all $i \in [n]$ (Corollary 1.6).

[2]For instance, in Figure 1, the smallest communication bottleneck $\bar{w}_1$ occurs between workers 5 and 4.

Algorithm 2 Find Fastest Subset of Workers (Preprocessing in Grace SGD)

Require: Graph $G = (V, E, b)$, times $\{h_i\}_{i\in[n]}$, ratio σ^2/ε

1: Construct Gomory-Hu tree $T = (V, F, w)$ of the undirected version $\bar{G}$
2: Sort $\{w_{ij}\}_{\{i,j\}\in F}$ and obtain $\bar{w}_1 \le \cdots \le \bar{w}_{n-1}$; also define $\bar{w}_n := \infty$ $\quad (|F| = n-1)$
3: Init $t^* = \infty$, $S^* =$ null, and $k^* =$ null
4: **for** $k = 1, \ldots, n$ **do**
5: Find connected components in T and denote them by $\mathcal{S}_k = (S_{k,1}, \ldots, S_{k,k})$
6: Find $\bar{S} = \arg\min_{S\in\mathcal{S}_k} \left\{ t_k(S) := \frac{d}{\bar{w}_k} + \min_{m\in[|S|]} \left[\left(\frac{1}{m}\sum_{i=1}^{m} \frac{1}{h_{\pi_i(S)}} \right)^{-1} \left(1 + \frac{\sigma^2}{\varepsilon m}\right) \right] \right\}$,
where $\pi(S)$ is a permutation that sorts $\{h_i\}_{i\in S} : h_{\pi_1(S)} \le \cdots \le h_{\pi_{|S|}(S)}$
7: If $t_k(\bar{S}) < t^*$, then $t^* = t_k(\bar{S})$, $S^* = \bar{S}$, and $k^* = k$
8: Remove the edge $\{i, j\}$ in T corresponding to $\bar{w}_k$ (except for $k = n$)
9: **end for**
10: **return** Subset of workers S^*

Choosing the best subset S^* (See visualization in Section C). Recall the discussion of Synchronous SGD and Hero SGD, two diametrically opposed methods, neither of which has universally best theoretical guarantees. The main advantage of Synchronous SGD is that it utilizes all workers; however, this also introduces its main drawback: communication among all workers can become a significant bottleneck. On the other hand, Hero SGD does not communicate but utilizes only one worker. Here comes our question: *How to choose a subset of workers that is large enough to cope with noises in gradients, but also does not require significant communication times? And, surprisingly, the answer lies in* Gomory–Hu *trees*, which naturally capture the *connectivity structure and bottlenecks* of G.

Using our *directed* graph G, we construct the *undirected* version $\bar{G} = (V, \bar{E}, b)$. For undirected graphs, Gomory & Hu (1961) proved that it is always possible to construct a tree $T = (V, F, w)$ (with the same set of vertices, but different sets of edges and weights). The Gomory–Hu tree T has one essential property. Consider any edge $e \in F$ of the tree with weight w_e. Then e separates the vertices of T into two sets, S_1 and S_2. It turns out that the value w_e is an upper bound on the *minimum s–t cut value* for all $s \in S_1$ and $t \in S_2$ (Theorem E.7). By the max-flow min-cut theorem, w_e is an upper bound on the number of coordinates per second (maximal flow) that S_1 and S_2 can transmit to each other. This immediately implies that if group S_1 or any worker from S_1 wants to send a vector of size ℓ to group S_2 or any worker from S_2, then it is necessary to wait at least ℓ/w_e seconds, *no matter what routing strategy one chooses.* Moreover, this communication rate is attained by at least one pair of workers $s \in S_1$ and $t \in S_2$, namely the endpoints of the edge e in T.

Using this observation, we design the iterative Algorithm 2. In the first iteration, we take T and find all its connected components. Since T is a tree, there is only one connected component, $S_{1,1} = [n]$. The idea of the next steps of the loop is to measure the time complexity of the method when all workers in $S_{1,1} = [n]$ solve (1) and communicate to each other. When $h_i = h$ for all $i \in [n]$, the time complexity is $t_1(S_{1,1}) \times L\Delta/\varepsilon = \Theta\left(d/\bar{w}_1 + h\max\left\{\sigma^2/n\varepsilon, 1\right\}\right) L\Delta/\varepsilon$, where $h\max\left\{\sigma^2/n\varepsilon, 1\right\}$ is the "statistical" term, which improves with n, and $d/\bar{w}_1$ is the "communication" term. Notice that $\bar{w}_1$ is the smallest value among the weights in F of the tree T, which characterizes the slowest communication bottleneck and communication speed, and appears because all workers participate in the optimization. The term $d/\bar{w}_1$ can be huge and much larger than $h\sigma^2/n\varepsilon$.

The main idea of the last step in the loop is to remove the edge $\{i, j\}$ corresponding to $\bar{w}_1$ from T, consider the new graph $T - \{i, j\}$, which has two connected components, $S_{2,1}$ and $S_{2,2}$, and consider an optimization process in which only one of these subsets is used. Then, choose the best subset using the values $t_2(S_{2,1}) = d/\bar{w}_2 + h\max\left\{\sigma^2/|S_{2,1}|\varepsilon, 1\right\}$ vs. $t_2(S_{2,2}) = d/\bar{w}_2 + h\max\left\{\sigma^2/|S_{2,2}|\varepsilon, 1\right\}$. Basically, when $h_i = h$ for all $i \in [n]$, we have to choose the largest connected component. *Why is that a good strategy?* By removing the edge with weight $\bar{w}_1$, we disconnect the groups $S_{2,1}$ and $S_{2,2}$. However, due to the properties of the Gomory–Hu tree T, the communication bottleneck inside each group becomes greater than or equal to $\bar{w}_2$ (instead of $\bar{w}_1$). An important observation is that the "communication" term $d/\bar{w}_2 \le d/\bar{w}_1$; thus, the new time complexity $\min\{t_2(S_{2,1}), t_2(S_{2,2})\}$ can be smaller than $t_1(S_{1,1})$. We repeat this loop until there are n connected components (singletons) $\{1\}, \ldots, \{n\}$,

Algorithm 3 Optimal-Bandwidth AllReduce (AllReduce in Grace SGD and Leon SGD)

Require: Graph $G = (V, E, b)$, with $\{b_{ij}\}_{\{i,j\}\in\bar{E}}$ being integers (w.l.o.g.)[(a)], subset of workers $S \subseteq V$, vector a_i stored at worker i for all $i \in S$

(Preprocessing; performed once)

1: Find undirected unweighted version of $\bar{G}$: $\hat{G} = (V, \hat{E})$ where, if $\{i, j\} \in \bar{E}$, then $\{i, j\}$ is repeated b_{ij} times in $\hat{E}$

2: Solve *Steiner Tree Packing* problem for $S \subseteq V$ in $\hat{G}$ using (Lau, 2004, Theorem 1.2) and obtain collection of edge-disjoint trees $\mathcal{T} := (\hat{T}_1, \ldots, \hat{T}_p)$ that each connects S
(p equals the value of minimum S-cut in $\hat{G}$ up to a constant factor)

3: Fix any pivot worker $v \in S$

(End of preprocessing. Beginning of AllReduce)

4: Worker $i \in S$ divides a_i into p blocks of size $\lceil d/p \rceil$: $(a_{i,1}, \ldots, a_{i,p})$ (pad with zeros if needed)

5: For all $j \in [p]$ (in parallel), *reduce* blocks $\{a_{i,j}\}_{i\in S}$ to pivot worker v along tree $\hat{T}_j$ (use *interleaving* described in Sections 2 and 1). Worker v receives $(\sum_{i\in S} a_{i,1}, \ldots, \sum_{i\in S} a_{i,p})$

6: For all $j \in [p]$ (in parallel), *broadcast* $\sum_{i\in S} a_{i,j}$ from v to all other workers from S along $\hat{T}_j$

Ensure: Sum $\sum_{i\in S} a_i$ is stored at worker i for all $i \in S$

(a): otherwise, b_{ij} are rational numbers (or approximated with arbitrary precision) and can always be renormalized by the common denominator

which have the time complexity of Hero SGD: $t_n(S_{n,i}) \times L\Delta/\varepsilon = \Theta\left(h \max\left\{\sigma^2/\varepsilon, 1\right\}\right) L\Delta/\varepsilon$, where $d/\bar{w}_n = 0$ because $\bar{w}_n = \infty$, no communication is required. Using the described strategy, we can find the optimal subset of workers that obtains the time complexity

$$(4) = t_{k^*}(S^*) \times \frac{L\Delta}{\varepsilon} = \Theta\left(\frac{d}{\bar{w}_{k^*}} + h \max\left\{\frac{\sigma^2}{|S^*|\varepsilon}, 1\right\}\right) \frac{L\Delta}{\varepsilon}, \tag{6}$$

where, roughly speaking, the communication and statistical terms are almost equal. It remains to design a method that indeed achieves this complexity.

Optimal-bandwidth AllReduce (See visualization in Section D). We have selected the subset of workers S^*. Obtaining the statistical term in (6) can be easily achieved with Minibatch SGD and $|S^*|$ workers. The main difficulty is achieving the communication term $d/\bar{w}_{k^*}$. Intuitively, since the communication bottleneck in S^* is at least $\bar{w}_{k^*}$, formally, the minimum S^*-cut value is at least $\bar{w}_{k^*}$ (Definition E.5), there should be a way to develop an efficient synchronization algorithm with complexity $\mathcal{O}(d/\bar{w}_{k^*})$. We now implement an optimal-bandwidth AllReduce algorithm, Algorithm 3, that achieves this complexity.

W.l.o.g., assume that $\{b_{ij}\}$ are integers, and instead of the undirected graph $\bar{G}$, consider the unweighted multigraph $\hat{G}$ (Algorithm 3). That is, instead of considering one edge $\{i, j\}$ with weight b_{ij}, we consider an equivalent graph where $\{i, j\}$ is repeated b_{ij} times, each with unit bandwidth. In practice, the system still has a single edge with bandwidth b_{ij}; however, the behavior of b_{ij} parallel unit-bandwidth edges can be simulated by multiplexing transmissions using the *interleaving* strategy described in Section 1.

By construction, for all $i, j \in S^*$, if we consider the path of edges p from i to j in the tree T, then $\min_{e\in p} w_e \geq \bar{w}_{k^*}$. Using the Gomory–Hu tree properties, we can conclude that the value of a minimum S^*-cut (Definition E.5) is greater than or equal to $\bar{w}_{k^*}$. Using the result of Lau (2004, Theorem 1.2), there exists a polynomial-time algorithm that can find a collection of edge-disjoint trees $\mathcal{T} := (\hat{T}_1, \ldots, \hat{T}_p)$ with $p = \Theta(\bar{w}_{k^*})$, each connecting S^* in $\hat{G}$.

In terms of our problem, it means that there exist $\Theta(\bar{w}_{k^*})$ edge-disjoint communication pipes that can communicate coordinates between the workers in S^* with unit bandwidth. Now, we ask every worker to split its local vector a_i (local minibatch) into $\Theta(\bar{w}_{k^*})$ blocks. Using the edge-disjoint trees, the workers can first aggregate the blocks at one of the workers (the pivot worker), and then, using the same trees, the pivot worker can broadcast the blocks to all workers in S^*. Importantly, the key idea of this algorithm is to find edge-disjoint trees that enable independent, parallel reduce and broadcast operations, implemented via interleaving without congestion. Since every edge in every tree from $\mathcal{T}$ has unit bandwidth, we can conclude that the time required for reduce and broadcast is $\Theta\left(d/\bar{w}_{k^*}\right)$ seconds for blocks of size $\Theta\left(d/\bar{w}_{k^*}\right)$. See details in the proof of Theorem F.1.

This AllReduce works with an arbitrary graph G. In practice, graphs often have a particular structure (e.g., all-to-all graphs or torus graphs), where implementing an optimal-bandwidth AllReduce is more straightforward. See practical guidelines and numerical experiments in Section K. We also note that Grace SGD is *asynchronous-friendly and robust to heterogeneous fluctuations in computation*, since when communication times are negligible, it reduces to Rennala SGD, which was proved to be optimal under arbitrary computational dynamics (Tyurin, 2025).

Preprocessing complexity. Our upper and lower time-complexity bounds account only for the computation and communication times specified in Assumption 1.4. However, Algorithm 2 constructs one Gomory–Hu tree and performs n component/selection passes, while Algorithm 3 executes the polynomial-time Steiner-tree-packing algorithm once during preprocessing. Although these procedures also incur computational costs, both run in polynomial time. The selected worker subset and the resulting Steiner trees are then reused in every AllReduce operation. Therefore, the preprocessing cost is incurred only once and does not accumulate over the optimization iterations. Moreover, since communication graph structures are typically fixed, the same communication scheme can potentially be reused across multiple training runs.

3 Lower Bound in the Homogeneous Setting

In this section, we show that Grace SGD and the result in Theorem 1.5 are optimal up to logarithmic factors within a large family of optimization methods. The reader will see that we also use Algorithm 2 to state the lower bound, which is not a coincidence. The "communication side" of the lower bound proof is constructive and enables the design of Grace SGD.

Obtaining lower bounds in the homogeneous setting is significantly more technically challenging, since the standard trick of placing different blocks of a hard function on distant nodes cannot be used. We consider the standard class of zero-respecting algorithms (Nesterov, 2018; Arjevani et al., 2022), which includes SGD-like methods, Adam, gossip methods (e.g., (Nedic et al., 2017)), and even compressed decentralized methods (e.g., Choco-SGD (Koloskova et al., 2019), BEER (Zhao et al., 2022)). The optimization protocol is flexible and presented in Protocol 5. Every worker runs two parallel loops: i) in the first computation loop, worker i computes stochastic gradients locally, adds these vectors to local information I_i, and the algorithm is allowed to generate the next query point using I_i; ii) in the second communication loop, worker i, using I_i, prepares a set of coordinates, which can be random and not necessarily of size d, and the algorithm sends them using *any routing strategy* that obeys Assumption 1.4. This way, the protocol allows the use of local steps, minibatching, gossip communication, and compressed communication. The formal description of the allowed ways to prepare coordinates is presented in Assumption G.1, where we allow algorithms to use mappings that do not take into account local information and do not depend on $\{I_i\}_{i\in[n]}$ when choosing the indices in the sparsifiers. Assumption G.1 covers sending the full vector, sending a predefined block of the vector as is done in AllReduce algorithms, or even a random subset of coordinates, supporting Rand$K$ or Perm$K$ compressors (Szlendak et al., 2021).

Definition 3.1. We define $\mathcal{F}_{\Delta,L}$ as the set of functions $f:\mathbb{R}^p\to\mathbb{R}$ such that $p\geq 1$, f is L-smooth, i.e., $\|\nabla f(x)-\nabla f(y)\|\leq L\|x-y\|$ for all $x,y\in\mathbb{R}^p$, and $f(0)-\inf_{x\in\mathbb{R}^p} f(x)\leq\Delta$.

Theorem 3.2 (Lower Bound in the Homogeneous Setting)**.** *We consider any computational and communication environment, any graph G satisfying Assumption 1.4. Let $L,\Delta,\varepsilon,n,\sigma^2,d>0$ be any numbers such that $L\Delta/\varepsilon\geq\bar{c}_1\log^{10}(n+1)$ and dimension $d\geq\bar{c}_3 L\Delta/\log^3(n+1)\varepsilon$. Consider Protocol 5. For all $i\in[n]$ and $k\geq 0$, mappings $\mathcal{C}_i^k$ satisfy Assumption G.1. Then, for any algorithm A, there exists a function $f\in\mathcal{F}_{\Delta,L}$, exists an oracle that satisfies Assumption 1.3, and $\mathbb{E}\left[\inf_{y\in G_t}\|\nabla f(y)\|^2\right]>\varepsilon$ for all*

$$t\leq\frac{\bar{c}_2}{\log^{14}(n+1)}\min_{k\in[n]}\left(\frac{d}{\bar{w}_k}+\min_{p\in[k]}\min_{m\in[|S_{k,p}|]}\left[\left(\frac{1}{m}\sum_{i=1}^{m}\frac{1}{h_{\pi_i(S_{k,p})}}\right)^{-1}\left(1+\frac{\sigma^2}{m\varepsilon}\right)\right]\right)\frac{L\Delta}{\varepsilon},\quad(7)$$

where G_t is the set of all possible points that can be constructed by A up to time t based on $\{I_i\}_{i\in[n]}$. The quantities $\pi(S)$, $\{S_{k,p}\}_{k\in[n],p\in[k]}$, and $\{\bar{w}_k\}_{k\in[n]}$ are defined in Algorithm 2, and $\bar{c}_1$, $\bar{c}_2$, and $\bar{c}_3$ are universal constants.

Hence, Grace SGD matches (7) up to the multiplicative factor $\bar{c}_2/\log^{14}(n+1)$. Asymptotically, ignoring the polylogarithmic factor, Grace SGD is optimal. We believe that the $\log^{14}(n+1)$ term is an artifact of our proof techniques, and we conjecture that with a slightly more careful analysis around (35), it should be possible to improve this term to $\log^6(n+1)$ with more technical steps. Nevertheless, eliminating the logarithmic factor completely would likely require a different proof strategy. Moreover, following Tyurin (2026), Assumption G.1 covers only a subfamily of unbiased compressors, including Rand$K$ for all $K \in [d]$. As in (Huang et al., 2022; He et al., 2023), the lower bound remains valid for the full class of compressed methods if the adversary is allowed to choose a "worst-case" compressor. Extending this result to all compressors and non-zero-respecting algorithms is an important future direction. The proof sketch and the full proof are in Sections G.1 and G.2.

4 Examples with Different Graphs in the Homogeneous Setting

While Theorem 1.5 applies to any graph, we illustrate the time complexities for typical graphs and give explicit formulas. For clarity, we assume $h_i = h$ for all $i \in [n]$ (Corollary 1.6).

Example: One Worker. Assume that $n = 1$, i.e., G is a graph with one node. In this case, Gomory-Hu T is the same as G, $\bar{w}_1 = \infty$ and $S_{1,1} = \{1\}$ in Algorithm 2, and $(4) = \mathcal{O}\left(h\sigma^2 L\Delta/\varepsilon^2 + hL\Delta/\varepsilon\right)$, which restores the classical result of Ghadimi & Lan (2013); Arjevani et al. (2022).

Example: Star Graph (Centralized Setting). Consider another example where workers communicate through a server. It can equivalently be represented by the graph in Figure 5. One can show that a Gomory-Hu tree T of G is shown in Figure 5. In the tree, sorting the weights, we get $\bar{w}_1 = b, \ldots, \bar{w}_{n-1} = b, \bar{w}_n = \infty$. Using this, one can show that the total time complexity is $\mathcal{O}\left(\min\left\{\left(\frac{d}{b} + \frac{h\sigma^2}{n\varepsilon}\right)\frac{L\Delta}{\varepsilon} + \frac{hL\Delta}{\varepsilon}, \frac{h\sigma^2 L\Delta}{\varepsilon^2} + \frac{hL\Delta}{\varepsilon}\right\}\right)$ (see Section I.1). Notice that the first term corresponds to the complexity of Synchronous SGD with mini-batching (and Grace SGD), method that communicates the full vectors to and from the server, and the second term corresponds to Hero SGD. This optimality of this result was also proven in (Tyurin, 2026).

Example: p-Torus. We now assume that G is a p-Torus (Figure 7), a popular architecture in large-scale model training (Jouppi et al., 2020). A Gomory-Hu tree T presented in Figure 8. Sorting the weights, we get $\bar{w}_1 = 2pb, \ldots, \bar{w}_{n-1} = 2pb, \bar{w}_n = \infty$. Using the same steps as in the previous example, the time complexity of Grace SGD is $\mathcal{O}\left(\min\left\{\left(\frac{d}{pb} + \frac{h\sigma^2}{n\varepsilon}\right)\frac{L\Delta}{\varepsilon} + \frac{hL\Delta}{\varepsilon}, \frac{h\sigma^2 L\Delta}{\varepsilon^2} + \frac{hL\Delta}{\varepsilon}\right\}\right)$. Note that the first term improves with p, formalizing why a p-Torus is preferable in practice.

Example: all-to-all. In the all-to-all graph G, where all nodes are connected to each other, one can similarly derive the time complexity $\mathcal{O}\left(\min\left\{\left(\frac{d}{(n-1)b} + \frac{h\sigma^2}{n\varepsilon}\right)\frac{L\Delta}{\varepsilon} + \frac{hL\Delta}{\varepsilon}, \frac{h\sigma^2 L\Delta}{\varepsilon^2} + \frac{hL\Delta}{\varepsilon}\right\}\right)$, where the communication term has the best scaling among the previous examples, which is expected since the all-to-all graph has the best connectivity. Another important **example with K clusters** is considered in Section I.2.

Example: optimization with switches. In practice, workers are not directly connected; instead, there are intermediate nodes, switches, that do not compute stochastic gradients and only transmit data between the workers. Our theory even supports this setting by simply setting $h_i < \infty$ if node i is a worker, and $h_i = \infty$ if it is a switch. Then, one can apply Theorem 1.5 with any graph topology.

5 Heterogeneous Setting: Optimal Method and Lower Bound

We also consider and analyze the *heterogeneous* problem of minimizing

$$\min_{x \in \mathbb{R}^d} \left\{ f(x) := \frac{1}{n} \sum_{i=1}^{n} f_i(x) \right\}, \tag{8}$$

where $f_i : \mathbb{R}^d \to \mathbb{R}$ for all $i \in [n]$. Unlike the homogeneous setup, for all $i \in [n]$, worker i can only access stochastic gradients of f_i.

Assumption 5.1 (Heterogeneous setting)**.** For all $i \in [n]$, worker i can only calculate $\nabla f_i(x;\xi)$ and $\mathbb{E}_\xi[\nabla f_i(x;\xi)] = \nabla f_i(x)$ and $\mathbb{E}_\xi[\|\nabla f_i(x;\xi) - \nabla f_i(x)\|^2] \leq \sigma^2$ for all $x \in \mathbb{R}^d$, where $\sigma^2 \geq 0$.

Algorithm 4 Leon SGD

Require: Graph $G = (V, E, b)$, ratio σ^2/ε, smoothness constant L, point x^0

1: All workers $[n]$ locally initialize starting point with x^0 and step size $\gamma = 1/2L$
2: **for** $k = 0, 1, \ldots$ **do**
3: All workers compute $\frac{1}{B_i}\sum_{j=1}^{B_i} \nabla f_i(x^k; \xi_{ij}^k)$, increasing local batch sizes B_i over time, until $\left(\frac{1}{n}\sum_{i=1}^n \frac{1}{B_i}\right)^{-1} \geq \max\{\lceil \sigma^2/\varepsilon \rceil, n\}/n$
4: Run optimal-bandwidth AllReduce (e.g., Algorithm 3) so workers in $[n]$ obtain $\frac{1}{n}\sum_{i=1}^n \frac{1}{B_i}\sum_{j=1}^{B_i} \nabla f_i(x^k; \xi_{ij}^k)$ **(crucial new step)**
5: Every worker in $[n]$ updates the iterate: $x^{k+1} = x^k - \frac{\gamma}{n}\sum_{i=1}^n \frac{1}{B_i}\sum_{j=1}^{B_i} \nabla f_i(x^k; \xi_{ij}^k)$
6: **end for**

In Theorem 1.7, we show that Leon SGD (Algorithm 4) is an optimal method, where the first term (5) arises from performing AllReduce across all workers, and the second and third terms are the same as in Malenia SGD (Tyurin & Richtárik, 2023), which is not a coincidence, since the aggregation phase in Line 3 is the same as there.

Similarly to Section 3, we can provide a lower bound, Theorem G.12, for the heterogeneous setting. This result matches Theorem 1.7 up to a constant factor. Unlike the homogeneous setting, Theorems 1.7 and G.12 are much more pessimistic: i) the time complexity depends on the worst bottleneck $d/\bar{w}_1$, where $\bar{w}_1$ is the smallest weight in the Gomory–Hu tree T, or equivalently the minimum value over all cuts in the graph G; ii) similarly to (Tyurin & Richtárik, 2023), the dependence on $\{h_i\}_{i\in[n]}$ is arithmetic-like in (5) compared to the harmonic-like dependence in (7).

6 Fundamental Trade-offs in Optimization over Graphs

We now want to explain that the optimal time complexity (5), derived for the heterogeneous setting, cannot be significantly improved when the graph is *sparse*, even in the homogeneous setting. Sparse graphs include graphs with a ring structure and other low-connectivity topologies commonly used in decentralized optimization (see examples in (Koloskova et al., 2019)).

Corollary 6.1 (Sparse Graphs; Proof in Section F.3)**.** *In view of Theorems 3.2 and 1.7, if i) $h_i = h$ for all $i \in [n]$; ii) the degree of every node in G is bounded by a small number $p = \Theta(1)$; iii) and all edges have the same bandwidth b. When $\sigma^2/\varepsilon \geq \Omega(d/bh)$, the optimal time complexity for solving both (1) and (8) is $\tilde{\Theta}\left(dL\Delta/b\varepsilon + h\sigma^2 L\Delta/n\varepsilon^2 + hL\Delta/\varepsilon\right)$ and it is achieved by Leon SGD.*

Note that this complexity can also be achieved with a naive mini-batch version of Synchronous SGD: $x^{k+1} = x^k - \frac{\gamma}{nB}\sum_{i=1}^n \sum_{j=1}^B \nabla f_i(x^k; \xi_{ij}^k)$ with the simple communication approach from Section H. One of the main challenges was to show that this holds even in the homogeneous setting. The following corollary naturally generalizes Corollary 6.1 and provides additional insight:

Corollary 6.2 (Proof in Section F.4; see also dual Corollary F.3)**.** *In view of Theorems 3.2, if $h_i = h$ for all $i \in [n]$ and $b_{ij} = b$ for all $(i, j) \in E$, then the lower bound in all settings is $\tilde{\Omega}\left(\min\left\{\min_{m\in\{2,\ldots,n\}}\left[\frac{dL\Delta}{k(m)b\varepsilon} + \frac{h\sigma^2 L\Delta}{m\varepsilon^2}\right] + \frac{hL\Delta}{\varepsilon}, \frac{hL\Delta}{\varepsilon} + \frac{h\sigma^2 L\Delta}{\varepsilon^2}\right\}\right)$, where $k(m)$ is the m-th largest edge degree of a node in G.*

Corollaries 6.2 and F.3 are not necessarily tight (the tight bound is given in Theorem 3.2), but they provide important intuition about learning on graphs: *in both homogeneous and heterogeneous settings, the more workers that compute stochastic gradients, the better the scaling in the stochastic term $h\sigma^2 L\Delta/m\varepsilon^2$; however, the scaling in the communication term $dL\Delta/k(m)b\varepsilon$ may become worse.* Having more workers compute stochastic gradients may lead to slower overall communication due to the low connectivity of some workers, and there exists a golden mean that balances the number of active workers, depending on the structure of the graph and the d vs. σ^2/ε regime.

AI USE STATEMENT

In this work, we used generative AI tools to check for and correct grammatical mistakes and check mathematical proofs. We have not used generative AI tools for other tasks that require disclosure. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Albert Alexandrov, Mihai F Ionescu, Klaus E Schauser, and Chris Scheiman. LogGP: Incorporating long messages into the logp model—one step closer towards a realistic model for parallel computation. In *Proceedings of the seventh annual ACM symposium on Parallel algorithms and architectures*, pp. 95–105, 1995.

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: Communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 1709–1720, 2017.

Yossi Arjevani, Yair Carmon, John C Duchi, Dylan J Foster, Nathan Srebro, and Blake Woodworth. Lower bounds for non-convex stochastic optimization. *Mathematical Programming*, pp. 1–50, 2022.

Grigory Begunov and Alexander Tyurin. Do we need Asynchronous SGD? On the near-optimality of synchronous methods. *arXiv preprint arXiv:2602.03802*, 2026.

Aleksandr Beznosikov, Samuel Horváth, Peter Richtárik, and Mher Safaryan. On biased compression for distributed learning. *arXiv preprint arXiv:2002.12410*, 2020.

Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE transactions on information theory*, 52(6):2508–2530, 2006.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Zixian Cai, Zhengyang Liu, Saeed Maleki, Madanlal Musuvathi, Todd Mytkowicz, Jacob Nelson, and Olli Saarikivi. Synthesizing optimal collective algorithms. In *Proceedings of the 26th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, pp. 62–75, 2021.

Yair Carmon, John C Duchi, Oliver Hinder, and Aaron Sidford. Lower bounds for finding stationary points i. *Mathematical Programming*, 184(1):71–120, 2020.

Alon Cohen, Amit Daniely, Yoel Drori, Tomer Koren, and Mariano Schain. Asynchronous stochastic optimization robust to arbitrary delays. *Advances in Neural Information Processing Systems*, 34: 9024–9035, 2021.

David Culler, Richard Karp, David Patterson, Abhijit Sahay, Klaus Erik Schauser, Eunice Santos, Ramesh Subramonian, and Thorsten Von Eicken. LogP: Towards a realistic model of parallel computation. In *Proceedings of the fourth ACM SIGPLAN symposium on Principles and practice of parallel programming*, pp. 1–12, 1993.

Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

Ralph E Gomory and Tien Chung Hu. Multi-terminal network flows. *Journal of the Society for Industrial and Applied Mathematics*, 9(4):551–570, 1961.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Aric Hagberg, Pieter J Swart, and Daniel A Schult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Laboratory (LANL), 2007.

Yutong He, Xinmeng Huang, and Kun Yuan. Unbiased compression saves communication in distributed optimization: When and how much? *Advances in Neural Information Processing Systems*, 36:47991–48020, 2023.

Xinmeng Huang, Yiming Chen, Wotao Yin, and Kun Yuan. Lower bounds and nearly optimal algorithms in distributed learning with communication compression. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Norman P Jouppi, Doe Hyun Yoon, George Kurian, Sheng Li, Nishant Patil, James Laudon, Cliff Young, and David Patterson. A domain-specific supercomputer for training deep neural networks. *Communications of the ACM*, 63(7):67–78, 2020.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*, 2015.

Anastasia Koloskova, Sebastian U Stich, and Martin Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *International Conference on Machine Learning*, 2019.

Anastasiia Koloskova, Sebastian U Stich, and Martin Jaggi. Sharper convergence guarantees for Asynchronous SGD for distributed and federated learning. *Advances in Neural Information Processing Systems*, 35:17202–17215, 2022.

Guanghui Lan. *First-order and stochastic optimization methods for machine learning*. Springer, 2020.

Lap Chi Lau. An approximate max-steiner-tree-packing min-steiner-cut theorem. In *45th Annual IEEE Symposium on Foundations of Computer Science*, pp. 61–70. IEEE, 2004.

Yann LeCun, Corinna Cortes, and CJ Burges. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Yucheng Lu and Christopher De Sa. Optimal complexity in decentralized training. In *International Conference on Machine Learning*, pp. 7111–7123. PMLR, 2021.

Artavazd Maranjyan, Alexander Tyurin, and Peter Richtárik. Ringmaster ASGD: The first Asynchronous SGD with optimal time complexity. In *International Conference on Machine Learning*, 2025.

Konstantin Mishchenko, Francis Bach, Mathieu Even, and Blake Woodworth. Asynchronous SGD beats minibatch SGD under arbitrary delays. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Angelia Nedic, Alex Olshevsky, and Wei Shi. Achieving geometric convergence for distributed optimization over time-varying graphs. *SIAM Journal on Optimization*, 27(4):2597–2633, 2017.

Arkadij Semenovič Nemirovskij and David Borisovich Yudin. Problem complexity and method efficiency in optimization. 1983.

Yurii Nesterov. A method for solving the convex programming problem with convergence rate o (1/k2). In *Dokl akad nauk Sssr*, volume 269, pp. 543, 1983.

Yurii Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018.

Pitch Patarasuk and Xin Yuan. Bandwidth optimal all-reduce algorithms for clusters of workstations. *Journal of Parallel and Distributed Computing*, 69(2):117–124, 2009.

Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. *Advances in Neural Information Processing Systems*, 24, 2011.

Amedeo Sapio, Marco Canini, Chen-Yu Ho, Jacob Nelson, Panos Kalnis, Changhoon Kim, Arvind Krishnamurthy, Masoud Moshref, Dan Ports, and Peter Richtárik. Scaling distributed machine learning with {In-Network} aggregation. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*, pp. 785–808, 2021.

Kevin Scaman, Francis Bach, Sébastien Bubeck, Yin Tat Lee, and Laurent Massoulié. Optimal algorithms for smooth and strongly convex distributed optimization in networks. In *International Conference on Machine Learning*, pp. 3027–3036. PMLR, 2017.

Alexander Schrijver et al. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer, 2003.

Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech DNNs. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

Rafał Szlendak, Alexander Tyurin, and Peter Richtárik. Permutation compressors for provably faster distributed nonconvex optimization. In *International Conference on Learning Representations*, 2021.

Rajeev Thakur, Rolf Rabenseifner, and William Gropp. Optimization of collective communication operations in MPICH. *The International Journal of High Performance Computing Applications*, 19(1):49–66, 2005.

Alexander Tyurin. Tight time complexities in parallel stochastic optimization with arbitrary computation dynamics. In *13th International Conference on Learning Representations*, 2025.

Alexander Tyurin. Proving the limited scalability of centralized distributed optimization via a new lower bound construction. In *International Conference on Learning Representations (ICLR)*, 2026.

Alexander Tyurin and Peter Richtárik. Optimal time complexities of parallel stochastic optimization methods under a fixed computation model. *Advances in Neural Information Processing Systems*, 2023.

Alexander Tyurin and Peter Richtárik. On the optimal time complexities in decentralized stochastic asynchronous optimization. *Advances in Neural Information Processing Systems*, 37, 2024.

Alexander Tyurin, Marta Pozzi, Ivan Ilin, and Peter Richtárik. Shadowheart SGD: Distributed asynchronous SGD with optimal time complexity under arbitrary computation and communication heterogeneity. *Advances in Neural Information Processing Systems*, 37, 2024.

Guanhua Wang, Shivaram Venkataraman, Amar Phanishayee, Nikhil Devanur, Jorgen Thelin, and Ion Stoica. Blink: Fast and generic collectives for distributed ml. *Proceedings of Machine Learning and Systems*, 2:172–186, 2020.

Haoyu Zhao, Boyue Li, Zhize Li, Peter Richtárik, and Yuejie Chi. BEER: Fast $O(1/T)$ rate for decentralized nonconvex optimization with communication compression. *Advances in Neural Information Processing Systems*, 35:31653–31667, 2022.

Liangyu Zhao and Arvind Krishnamurthy. Bandwidth optimal pipeline schedule for collective communication. *arXiv preprint arXiv:2305.18461*, 2023.

Martin Zinkevich, Markus Weimer, Lihong Li, and Alex Smola. Parallelized stochastic gradient descent. *Advances in Neural Information Processing Systems*, 23, 2010.

CONTENTS

1 Introduction **1**

1.1 Known methods 2

1.2 Contributions 3

2 Grace SGD: A New Method with Near-Optimal Time Complexity **4**

3 Lower Bound in the Homogeneous Setting **7**

4 Examples with Different Graphs in the Homogeneous Setting **8**

5 Heterogeneous Setting: Optimal Method and Lower Bound **8**

6 Fundamental Trade-offs in Optimization over Graphs **9**

A Additional Related Work **15**

B Notations **16**

C Visualization of Algorithm 2 on Gomory-Hu Tree from Figure 1 **17**

D Visualization of Algorithm 3 **18**

E Definitions and Results from Graph Theory **18**

F Upper Bounds **19**

F.1 Proof of Theorem 1.5 19

F.2 Proof of Theorem 1.7 23

F.3 Proof of Corollary 6.1 23

F.4 Proof of Corollaries F.3 and 6.2 24

G Lower Bounds **25**

G.1 Proof sketch 25

G.2 Full proof 27

G.3 Auxiliary lemmas 38

G.4 Proof of Lemma G.7 42

G.5 Proof of Theorem G.12 43

H One Step in Synchronous SGD **47**

I Examples of Graphs **48**

I.1 Star Graph 48

I.2 K clusters 49

I.3 p-Torus 50

J Examples with Different Graph Structures in the Heterogeneous Setting 51

K Practical Guidelines and Numerical Experiments 52

K.1 Numerical experiments 52

L Graph-Bandwidth Communication Model with Latencies and Fluid Assumption 55

L.1 Relation to Previous Communication Model Assumptions 55

A Additional Related Work

Classical theory. Starting with one of the seminal works (Nemirovskij & Yudin, 1983), the optimization field began investigating the optimality of methods. Initially, the focus was mainly on the oracle complexity, e.g., the number of function and gradient evaluations needed to achieve a given accuracy. One of the first results was obtained in the convex setting where it was shown that the optimal complexity for finding an ε-approximate solution is $\Theta(\sqrt{LR^2/\varepsilon})$ (Nesterov, 1983; 2018), where R is the distance between the initial point and an optimal point. For the nonconvex setting, the optimal complexity $\Theta(L\Delta/\varepsilon)$ was shown by Carmon et al. (2020) in the deterministic setting and $\Theta(L\Delta/\varepsilon + \sigma^2 L\Delta/\varepsilon^2)$ by Arjevani et al. (2022) in the stochastic setting.

Gossip protocol. One way to extend the classical results to the decentralized setting is to use the gossip protocol. As in Graph-Bandwidth Communication Model, workers are associated with nodes and communication links with edges of the graph. Then, a mixing matrix $W \in \mathbb{R}^{n\times n}$ is constructed such that $w_{ij} > 0$ if nodes i and j are connected, and $w_{ij} = 0$ otherwise. The upper and lower bounds are then constructed in terms of the number of communication rounds required to find an ε-stationary point (Boyd et al., 2006; Scaman et al., 2017; Koloskova et al., 2019; Lu & De Sa, 2021). However, this approach arguably has the following drawbacks: (i) this protocol allows methods to communicate only with neighbors and to use a single operation $y_i \leftarrow \sum_j w_{ij}x_j$, where x_j is the sent vector and y_i is the received vector. Such a protocol restricts many practical ways to disseminate information in the graph. For instance, the split vector and online in-network aggregation strategies, discussed in Section 1 and commonly used in practice, are ignored; (ii) the obtained complexities are typically defined in terms of *iteration complexity* and depend on the spectral gap of a mixing matrix, whereas our approach uses *time complexities* with explicit dependence on computation and communication time parameters $\{h_i\}$ and $\{b_{ij}\}$; (iii) the obtained results use "one-zero" encoding, assuming that all communication links are the same, whereas our approach uses bandwidths, a practical way of defining connectivity between two workers or switches.

Time complexities. The classical oracle complexity is a valid and intuitive metric for comparing methods with one worker/GPU/CPU/server. However, modern optimization requires hundreds, thousands, or even millions of workers, requiring the analysis of *parallel and asynchronous algorithms*, where a conceptually different metric is needed. Let us consider a recent paper by Mishchenko et al. (2022), where the authors compare Synchronous SGD and Asynchronous SGD. The *oracle complexity* of both methods is the same; moreover, the *iteration complexity* of Synchronous SGD is better. However, under the *time complexity*, a more appropriate metric for parallel methods, Asynchronous SGD is provably better. In particular, assuming the Computation Model and using the analysis by Cohen et al. (2021); Koloskova et al. (2022); Mishchenko et al. (2022), it is possible to show that Asynchronous SGD has provably better time complexity than Synchronous SGD. Further, Tyurin & Richtárik (2023) formalized the notion of time complexity and proved that the optimal time complexity is $\Theta(\min_{m\in[n]}[(1/m\sum_{i=1}^m 1/h_{\pi_i})^{-1}(1+\sigma^2/m\varepsilon)]L\Delta/\varepsilon)$, achieved[3] by a method called Rennala SGD, where π is a permutation that sorts $\{h_i\}_{i\in[n]}$.

Time complexities with communication times. A natural question is to determine the optimal time complexities when communication between workers cannot be ignored. One of the early works studying such complexities in the decentralized setting is Scaman et al. (2017), where the complexity was established for heterogeneous objectives, without stochastic gradients, and under the assumption that workers exchange full vectors with their neighbors at every step (gossip communication). At the same time, our Graph-Bandwidth Communication Model is much more flexible, while also supporting the homogeneous setup and stochastic gradients. The idea of using bandwidths on the communication links was considered in (Tyurin et al., 2024; Tyurin, 2026) (in fact, they define τs instead as the time to send one coordinate, which are the inverses of the bandwidths). However, an important limitation of these works is that they only consider the centralized setup (star graphs), while ours considers arbitrary graphs. The analysis of communication in star graphs is much easier due to the lack of congestion on the communication links and the uniqueness of the paths between the workers. A work by Tyurin & Richtárik (2024) is closest to our setting, since it also proves optimal time complexities in the decentralized setup with both homogeneous and heterogeneous functions

[3] It was also achieved by another method called Ringmaster ASGD (Maranjyan et al., 2025). Moreover, somewhat surprisingly, it was later shown that this complexity, up to a logarithmic factor, can also be achieved by a synchronous method (Begunov & Tyurin, 2026).

and stochastic gradients. However, it assumes that algorithms are only allowed to send full vectors to neighbors, an important limitation that does not capture, for instance, optimal-bandwidth AllReduce strategies.

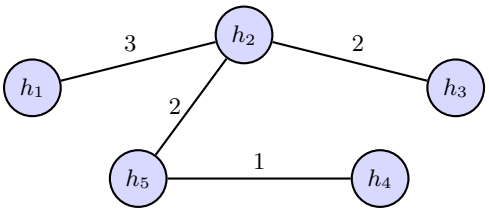
B NOTATIONS

Table 1: List of notations used throughout the paper.

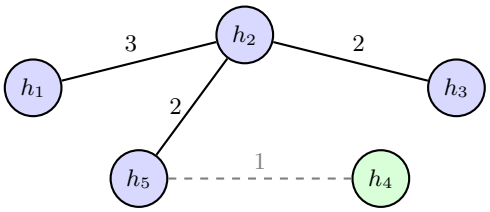
Notation	Meaning
$[n]$	Denotes the finite set $\{1, \dots, n\}$.
$\|x\|$	Euclidean norm of a vector x.
$g = \mathcal{O}(f)$	There exists $C > 0$ such that $g(z) \leq C\, f(z)$ for all $z \in \mathcal{Z}$.
$g = \Omega(f)$	There exists $C > 0$ such that $g(z) \geq C\, f(z)$ for all $z \in \mathcal{Z}$.
$g = \Theta(f)$	There exist $C_1, C_2 > 0$ such that $C_1 f(z) \leq g(z) \leq C_2 f(z)$ for all $z \in \mathcal{Z}$.
$\tilde{\mathcal{O}}, \tilde{\Omega}$, and $\tilde{\Theta}$	The same as $\mathcal{O}$, Ω, and Θ, but up to logarithmic factors.
Δ	Initial optimality gap, $\Delta := f(x^0) - f^*$.
$\mathbb{E}[\cdot]$	Full expectation.
$\mathbb{E}_\xi[\cdot]$	Expectation w.r.t. random variable ξ.
h_i	Time required by worker i to compute one stochastic gradient.
$G = (V, E, b)$	Directed weighted communication graph.
$\bar{G} = (V, \bar{E}, b)$	Undirected version of the communication graph G.
$T = (V, F, w)$	Gomory–Hu tree of the undirected graph $\bar{G}$.
$\bar{w}_1 \leq \cdots \leq \bar{w}_{n-1}$	Sorted edge weights of the Gomory–Hu tree.
$\bar{w}_n$	Auxiliary value defined as ∞.
$\pi(S)$	Permutation that sorts $\{h_i\}_{i \in S}$ increasingly.
$\hat{G} = (V, \hat{E})$	Unweighted multigraph obtained from $\bar{G}$ by repeating $\{i, j\}$ edge b_{ij} times.
$\mathcal{T} = (\hat{T}_1, \dots, \hat{T}_p)$	Collection of edge-disjoint Steiner trees connecting a subset $S \subseteq V$ in $\hat{G}$.

C VISUALIZATION OF ALGORITHM 2 ON GOMORY-HU TREE FROM FIGURE 1

Step 1: $k = 1$, $\bar{w}_1 = 1$
$\mathcal{S}_1 = \{S_{1,1}\}, \quad S_{1,1} = \{1, 2, 3, 4, 5\}$

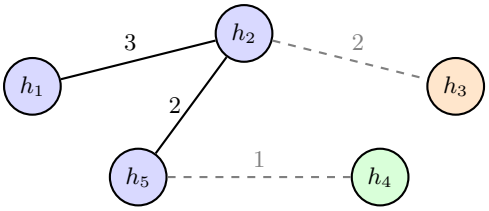


Step 2: remove edge of weight 1, $\bar{w}_2 = 2$
$\mathcal{S}_2 = \{S_{2,1}, S_{2,2}\}$



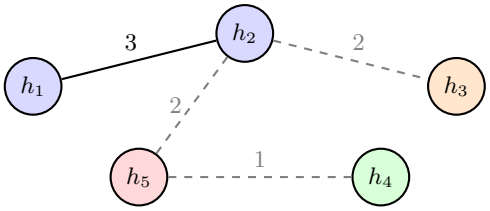
$S_{2,1} = \{1, 2, 3, 5\}, \quad S_{2,2} = \{4\}$

Step 3: remove one edge of weight 2, $\bar{w}_3 = 2$
$\mathcal{S}_3 = \{S_{3,1}, S_{3,2}, S_{3,3}\}$



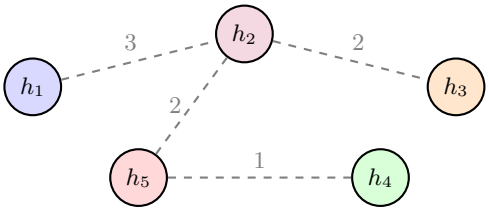
$S_{3,1} = \{1, 2, 5\}, \quad S_{3,2} = \{3\}, \quad S_{3,3} = \{4\}$

Step 4: remove second edge of weight 2, $\bar{w}_4 = 3$
$\mathcal{S}_4 = \{S_{4,1}, S_{4,2}, S_{4,3}, S_{4,4}\}$



$S_{4,1} = \{1, 2\}, \quad S_{4,2} = \{5\}, \quad S_{4,3} = \{3\}, \quad S_{4,4} = \{4\}$

Step 5: remove edge of weight 3, $\bar{w}_5 = \infty$
$\mathcal{S}_5 = \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}\}$



Hero-SGDs: no communication

Figure 2: Step-by-step visualization of Algorithm 2 on the Gomory–Hu tree T.

In Step 1, we assume that all workers communicate to each other. In this case, if we implement Grace SGD (with $S^* \to S_{1,1}$), then the guaranteed complexity would be

$$\mathcal{O}\left(\frac{d}{\bar{w}_1} \times \frac{L\Delta}{\varepsilon} + \min_{m \in [|S_{1,1}|]} \left[\left(\frac{1}{m}\sum_{i=1}^{m} \frac{1}{h_{\pi_i(S_{1,1})}}\right)^{-1} \left(1 + \frac{\sigma^2}{\varepsilon m}\right)\right] \frac{L\Delta}{\varepsilon}\right).$$

In Step 2, we split worker 4 from the other workers in T and evaluate the performance of Grace SGD (with $S^* \to S_{2,1}$ and $S^* \to S_{2,2}$), and record the best performance t^* found so far (Algorithm 2). We repeat these steps until all workers are split.

Important observation: Assume that $S^* = S_{4,1} = \{1, 2\}$ and Grace SGD decides to use only these workers in the optimization process. This does not mean that, for instance, node 5 would not participate. On the contrary, node 5 should participate, since part of the information would flow through it (see Figure 1, where node 5 is needed to achieve the total max flow from node 1 to node 2). However, node 5 would not perform any computations or produce any gradients. It would act as a switch. See also Section D.

D VISUALIZATION OF ALGORITHM 3

In this section, we consider the graph from Figure 3 and visualize the behavior of Algorithm 3.

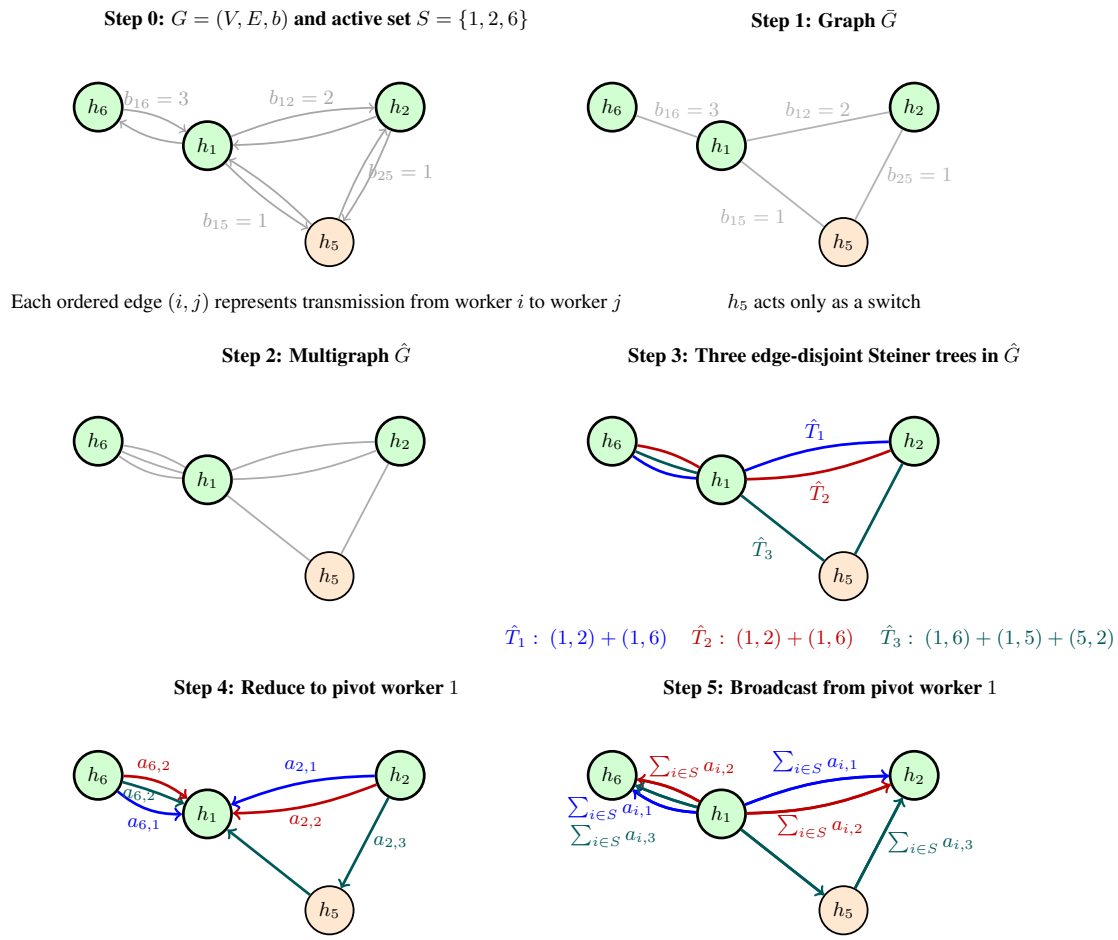


Figure 3: Step-by-step visualization of the optimal-bandwidth AllReduce for $S = \{1, 2, 6\}$. Step 0 shows the initial directed graph $G = (V, E, b)$, where each ordered edge (i, j) represents a communication link from worker i to worker j with bandwidth b_{ij}. Step 1 shows the corresponding undirected graph $\bar{G}$. Step 2 shows the multigraph $\hat{G}$, where each edge of bandwidth b_{ij} is replaced by b_{ij} parallel unit-bandwidth edges. Step 3 illustrates three edge-disjoint Steiner trees in $\hat{G}$. Step 4 shows the reduce phase: each worker sends one block through each tree to the pivot worker 1. Step 5 shows the broadcast phase: the pivot sends the aggregated blocks back through the same trees. **Important observation:** Node 5 is still required in the optimization process, but only as a switch.

E DEFINITIONS AND RESULTS FROM GRAPH THEORY

In this section, we list the standard definition and results from graph theory.

Definition E.1. Given an undirected weighted graph $\bar{G} = (V, \bar{E}, b)$ and a vertex set $U \subseteq V$, we define

$$\delta(U) \coloneqq \{\, e = \{x, y\} \in \bar{E} \;:\; x \in U,\; y \in V \setminus U \,\},$$

that is, the set of edges with exactly one endpoint in U.

Definition E.2. Given an undirected weighted graph $\bar{G} = (V, \bar{E}, b)$ and two vertices $u, v \in V$, a *u–v cut* is a partition $(U, V \setminus U)$ of V such that $u \in U$ and $v \in V \setminus U$. The value of the cut is defined as

$$\sum_{e \in \delta(U)} b_e.$$

Definition E.3. Given an undirected weighted graph $\bar{G} = (V, \bar{E}, b)$ and two vertices $u, v \in V$, we define $\alpha_{\bar{G}}(u, v)$ as the minimum value of a u–v cut in $\bar{G}$. Moreover, any u–v cut that attains this minimum value is called a *minimum u–v cut*.

Definition E.4. Given an undirected weighted graph $\bar{G} = (V, \bar{E}, b)$ and a vertex set $S \subseteq V$, an *S-cut* is a partition $(U, V \setminus U)$ of V such that

$$U \cap S \neq \emptyset \quad \text{and} \quad (V \setminus U) \cap S \neq \emptyset.$$

The value of the cut is defined as

$$\sum_{e \in \delta(U)} b_e.$$

Definition E.5. Given an undirected weighted graph $\bar{G} = (V, \bar{E}, b)$ and a vertex set $S \subseteq V$, we define $\alpha_{\bar{G}}(S)$ as the minimum value of an S-cut in $\bar{G}$. Moreover, any S-cut that attains this minimum value is called a *minimum S-cut*.

Definition E.6 (Gomory & Hu (1961))**.** A Gomory–Hu tree $T = (V, F, w)$ of an undirected weighted graph $\bar{G} = (V, \bar{E}, b)$ is a tree $T = (V, F, w)$, which consists of a tree edge set F and capacities w such that, for every edge $\{i, j\} \in F$, $\delta(W)$ is a minimum $i - j$ cut in $\bar{G}$ and w_{ij} is the value of this cut, where W is one component[4] of $T - \{i, j\}$.

Notice that, by definition, we have $w_{ij} = \alpha_{\bar{G}}(i, j)$ for all $\{i, j\} \in F$. Let us also recall the following important theorem.

Theorem E.7 (e.g. (Schrijver et al., 2003))**.** *Let $T = (V, F, w)$ be a Gomory–Hu tree of $\bar{G}$. For any two vertices $s, t \in V$, consider the unique s–t path in T, and let $\{u, v\} \in F$ be an edge on this path minimizing the weight w_{uv}. Then,*

$$\alpha_{\bar{G}}(s, t) = \alpha_{\bar{G}}(u, v),$$

and for any component K of $T - \{u, v\}$, the cut $\delta(K)$ is a minimum s–t cut in $\bar{G}$.

It turns out that the Gomory–Hu tree naturally captures the *connectivity structure and bottlenecks* of both graphs $\bar{G}$ and G. Indeed, consider any tree edge $\{u, v\} \in F$ and the corresponding components W_1 and W_2 obtained by removing the edge $\{u, v\}$ from T. Then, $\alpha_{\bar{G}}(s, t) \leq \alpha_{\bar{G}}(u, v) = w_{uv}$ for all $s \in W_1$ and $t \in W_2$ by Theorem E.7. By the max-flow min-cut theorem, the *maximum s–t flow* in $\bar{G}$ is upper bounded by w_{uv}.

F Upper Bounds

F.1 Proof of Theorem 1.5

Theorem 1.5 (Upper Bound in the Homogeneous Setting)**.** *Assume that Assumptions 1.1, 1.2, and 1.3 hold. Then Grace SGD (Algorithm 1) finds an ε-stationary point of (1) after $K = \lceil 4L\Delta/\varepsilon \rceil$ iterations, and the time complexity under Assumption 1.4 is*

$$\mathcal{O}\left(\min_{k \in [n]}\left(\frac{d}{\bar{w}_k} + \min_{p \in [k]} \min_{m \in [|S_{k,p}|]}\left[\left(\frac{1}{m}\sum_{i=1}^{m} \frac{1}{h_{\pi_i(S_{k,p})}}\right)^{-1}\left(1 + \frac{\sigma^2}{m\varepsilon}\right)\right]\right)\frac{L\Delta}{\varepsilon}\right) \tag{3}$$

up to a universal constant factor, where $\{S_{k,p}\}_{k \in [n], p \in [k]}$ and $\{\bar{w}_k\}_{k \in [n]}$ are the sequences generated by Algorithm 2, and $\pi(S)$ is a permutation that sorts $\{h_i\}_{i \in S}$: $h_{\pi_1(S)} \leq \cdots \leq h_{\pi_{|S|}(S)}$.

[4] Removing the edge $\{i, j\}$ from the tree T yields two sets of nodes W_1 and W_2, and W can be either of them, without loss of generality, since $\delta(W_1) = \delta(W_2)$.

Proof. (**Iteration rate**). The proof of the iteration is standard and we prove it for completeness. Fix any subset of workers $S \subseteq [n]$, and let

$$B = \max\left\{\left\lceil \frac{\sigma^2}{\varepsilon} \right\rceil, 1\right\}.$$

At iteration k, Grace SGD forms the mini-batch estimator

$$g^k = \frac{1}{B} \sum_{i \in S} \sum_{j=1}^{B_i^k} \nabla f(x^k; \xi_{i,j}^k), \qquad \sum_{i \in S} B_i^k = B.$$

By Assumption 1.3, $\mathbb{E}\left[g^k \middle| x^k\right] = \nabla f(x^k)$ and $\mathbb{E}\left[\|g^k - \nabla f(x^k)\|^2 \middle| x^k\right] \leq \frac{\sigma^2}{B} \leq \varepsilon$.

Using L-smoothness of f and the update

$$x^{k+1} = x^k - \gamma g^k, \qquad \gamma = \frac{1}{2L},$$

we get

$$\mathbb{E}\left[f(x^{k+1}) \middle| x^k\right] \leq f(x^k) - \gamma \|\nabla f(x^k)\|^2 + \frac{L\gamma^2}{2} \mathbb{E}\left[\|g^k\|^2 \middle| x^k\right].$$

Since

$$\mathbb{E}\left[\|g^k\|^2 \middle| x^k\right] = \|\nabla f(x^k)\|^2 + \mathbb{E}\left[\|g^k - \nabla f(x^k)\|^2 \middle| x^k\right] \leq \|\nabla f(x^k)\|^2 + \varepsilon,$$

it follows that

$$\begin{aligned}\mathbb{E}\left[f(x^{k+1}) \middle| x^k\right] &\leq f(x^k) - \left(\gamma - \frac{L\gamma^2}{2}\right) \|\nabla f(x^k)\|^2 + \frac{L\gamma^2}{2}\varepsilon \\ &= f(x^k) - \frac{3}{8L} \|\nabla f(x^k)\|^2 + \frac{\varepsilon}{8L}.\end{aligned}$$

Rearranging and taking full expectation,

$$\frac{3}{8L} \mathbb{E}\left[\left\|\nabla f(x^k)\right\|^2\right] \leq \mathbb{E}\left[f(x^k)\right] - \mathbb{E}\left[f(x^{k+1})\right] + \frac{\varepsilon}{8L}.$$

Summing over $k = 0, \ldots, K-1$ gives

$$\frac{3}{8L} \sum_{k=0}^{K-1} \mathbb{E}\left[\|\nabla f(x^k)\|^2\right] \leq f(x^0) - \inf_{x \in \mathbb{R}^d} f(x) + \frac{K\varepsilon}{8L} = \Delta + \frac{K\varepsilon}{8L}.$$

Hence

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E}\left[\left\|\nabla f(x^k)\right\|^2\right] \leq \frac{8L\Delta}{3K} + \frac{\varepsilon}{3}.$$

So it is enough to choose K such that the right-hand side is at most ε. In particular,

$$K = \left\lceil \frac{4L\Delta}{\varepsilon} \right\rceil$$

is sufficient, which proves the iteration complexity.

(**Time complexity**). It remains to upper bound the time complexity of one iteration. Consider the chosen subset of workers $S^* \subseteq [n]$ in Algorithm 2. Under Assumption 1.4 (Computation Model), the time required to collect a batch of size $B = \max\left\{\left\lceil \frac{\sigma^2}{\varepsilon} \right\rceil, 1\right\}$ by the workers from S^* is at most

$$\min_{m \in [|S^*|]} \left[\left(\frac{1}{m} \sum_{i=1}^{m} \frac{1}{h_{\pi_i(S^*)}}\right)^{-1}\right] \left(1 + \frac{\sigma^2}{m\varepsilon}\right)$$

seconds (Lemma F.2). Moreover, by our construction in Algorithm 2, we know that the value of a minimum S^*-cut (Definition E.5) is greater than or equal to $\bar{w}_{k^*}$. Thus, there exists an optimal-bandwidth AllReduce algorithm (Algorithm 3, Theorem F.1) such that the allreduce operation is bounded by $\mathcal{O}\left(d/\bar{w}_{k^*}\right)$ seconds. Hence one iteration on this subset costs at most

$$\mathcal{O}\left(\frac{d}{\bar{w}_{k^*}} + \min_{m\in[|S^*|]}\left[\left(\frac{1}{m}\sum_{i=1}^{m}\frac{1}{h_{\pi_i(S^*)}}\right)^{-1}\right]\left(1+\frac{\sigma^2}{m\varepsilon}\right)\right).$$

Algorithm 2 chooses the subset S^* minimizing this quantity over all candidates $\{S_{k,p}\}$. Therefore, the per-iteration time is at most

$$\min_{k\in[n]}\left(\frac{d}{\bar{w}_k} + \min_{p\in[k]}\min_{m\in[|S_{k,p}|]}\left[\left(\frac{1}{m}\sum_{i=1}^{m}\frac{1}{h_{\pi_i(S_{k,p})}}\right)^{-1}\left(1+\frac{\sigma^2}{m\varepsilon}\right)\right]\right).$$

Multiplying by the number of iterations $K = \left\lceil \frac{4L\Delta}{\varepsilon} \right\rceil$, we get (3). □

Theorem F.1. *Let $\bar{G} = (V, \bar{E}, b)$ be the undirected version of $G : \{i, j\} \in \bar{E}$ with weight b_{ij} iff $(i, j) \in E$ with weight b_{ij}. Under Assumption 1.4 (Graph-Bandwidth Communication Model), the time complexity of the AllReduce algorithm in Algorithm 3 is*

$$\mathcal{O}\left(\frac{d}{\alpha_{\bar{G}}(S)}\right),$$

where $\alpha_{\bar{G}}(S)$ denotes the value of a minimum S-cut in the graph $\bar{G}$ (Definition E.5).

Proof. Consider the unweighted multigraph $\hat{G} = (V, \hat{E})$ obtained from $\bar{G} = (V, \bar{E}, b)$ by replacing every edge $\{i, j\} \in \bar{E}$ of weight b_{ij} with b_{ij} parallel edges of unit bandwidth. Clearly, any cut in $\bar{G}$ and the corresponding cut in $\hat{G}$ have the same value. Therefore,

$$\alpha_{\bar{G}}(S) = \alpha_{\hat{G}}(S).$$

We explain in the main part that in practice, the system still has a single edge with bandwidth b_{ij}; however, the behavior of b_{ij} parallel unit-bandwidth edges can be simulated by multiplexing transmissions using the *interleaving* strategy described in Section 1.

By the Steiner tree packing result of Lau (2004, Theorem 1.2), there exists a polynomial-time algorithm that finds a collection of

$$p = \Theta\big(\alpha_{\hat{G}}(S)\big) = \Theta\big(\alpha_{\bar{G}}(S)\big)$$

edge-disjoint trees

$$\mathcal{T} = (\hat{T}_1, \ldots, \hat{T}_p)$$

in $\hat{G}$, each of which connects all vertices of S.

We use these trees as parallel communication pipes. Split every local vector $a_i \in \mathbb{R}^d$ stored at worker $i \in S$ into p disjoint blocks,

$$a_i = \big(a_{i,1}, \ldots, a_{i,p}\big),$$

where each block has size $\mathcal{O}(d/p)$ coordinates (w.l.o.g., we assume that d is divisible by p; otherwise, we can pad with zero values). Assign block ℓ to tree $\hat{T}_\ell$.

Fix any pivot worker $r \in S$. We first perform a reduce operation. For every $\ell \in [p]$ and any worker $i \in S$, find the unique path from i to the root r. Each worker $i \in S$ starts streaming the coordinates of its block $a_{i,\ell}$ along the unique path in $\hat{T}_\ell$ toward r. Every intermediate node waits until it receives the next coordinate from all of its children, i.e., the neighbors of the intermediate node that send coordinates to the root through this intermediate node. Importantly, the intermediate node does not wait to receive the entire block of size $\Theta(d/p)$. As soon as the first coordinate is received from all children, the node aggregates these values (adding its own coordinate if it belongs to S) and immediately forwards the result to its parent, i.e., the next node on the path towards the root r. The same procedure is then applied to subsequent coordinates.

Since the trees are edge-disjoint, these communications do not interfere with one another. Moreover, every edge in $\hat{G}$ has unit bandwidth, so the time needed to transmit one block through one tree is proportional to the block size, that is,

$$\mathcal{O}(d/p).$$

Thus, after $\mathcal{O}(d/p)$ seconds, the pivot worker r has obtained the sum of block ℓ for every $\ell \in [p]$, and hence the full sum $\sum_{i\in S} a_i$. In practice, of course, in addition to sending the values of the coordinates, the workers might also transfer the indices of these coordinates and other meta-information; nevertheless, this would increase the cost by at most a multiplicative constant factor $\Theta(1)$.

Next, we perform a broadcast operation. Using the same collection of trees, the pivot worker sends block ℓ of the aggregated vector through $\hat{T}_\ell$, and intermediate nodes forward the received information further. Importantly, the intermediate node does not wait to receive the entire block and immediately broadcasts a new coordinate upon receiving it, without waiting for the next ones. Again, because the trees are edge-disjoint, all broadcasts proceed in parallel without congestion, and the required time is

$$\mathcal{O}(d/p).$$

Combining the reduce and broadcast phases, the total communication time is

$$\mathcal{O}(d/p) + \mathcal{O}(d/p) = \mathcal{O}(d/p).$$

Finally, since

$$p = \Theta\big(\alpha_{\bar{G}}(S)\big),$$

we obtain that the AllReduce time complexity is

$$\mathcal{O}\left(\frac{d}{\alpha_{\bar{G}}(S)}\right).$$

□

Lemma F.2. *Let $S \subseteq [n]$ be a subset of workers. The time required to collect a batch of size B using workers in S under the Computation Model is at most*

$$\min_{m\in[|S|]} \left[\left(\frac{1}{m}\sum_{i=1}^{m}\frac{1}{h_{\pi_i(S)}}\right)^{-1}\left(1+\frac{B}{m}\right)\right] \tag{9}$$

seconds, $\pi(S)$ is a permutation that sorts $\{h_i\}_{i\in S} : h_{\pi_1(S)} \leq \cdots \leq h_{\pi_{|S|}(S)}$.

Proof. Let

$$t = \min_{m\in[|S|]} \left(\left(\sum_{i=1}^{m}\frac{1}{h_{\pi_i(S)}}\right)^{-1}(B+m)\right).$$

As soon as a worker finishes computing a stochastic gradient, it immediately starts computing the next one. Hence, by time t, worker i will have computed at least

$$\left\lfloor \frac{t}{h_i} \right\rfloor$$

stochastic gradients. Therefore the total number of gradients computed by workers in S by time t is at least

$$\sum_{i\in S}\left\lfloor \frac{t}{h_i}\right\rfloor \geq \sum_{i=1}^{m^*}\left\lfloor \frac{t}{h_{\pi_i(S)}}\right\rfloor,$$

where

$$m^* = \arg\min_{m\in[|S|]} \left(\left(\sum_{i=1}^{m}\frac{1}{h_{\pi_i(S)}}\right)^{-1}(B+m)\right).$$

Since $\lfloor x \rfloor \geq x - 1$ for all $x \geq 0$, we obtain

$$\sum_{i=1}^{m^*} \left\lfloor \frac{t}{h_{\pi_i(S)}} \right\rfloor \geq \sum_{i=1}^{m^*} \left(\frac{t}{h_{\pi_i(S)}} - 1 \right)$$
$$= t \sum_{i=1}^{m^*} \frac{1}{h_{\pi_i(S)}} - m^* = \left(\sum_{i=1}^{m^*} \frac{1}{h_{\pi_i(S)}} \right) \left(\left(\sum_{i=1}^{m^*} \frac{1}{h_{\pi_i(S)}} \right)^{-1} (B + m^*) \right) - m^* = B.$$

Thus, by time t, at least B stochastic gradients have been computed. □

F.2 Proof of Theorem 1.7

Theorem 1.7 (Upper Bound in the Heterogeneous Setting)**.** *Assume that Assumptions 1.1, 1.2, and 5.1 hold. Then Leon SGD (Algorithm 4) finds an ε-stationary point of (8) after $K = \Theta\left(L\Delta/\varepsilon\right)$ iterations, and the time complexity under Assumption 1.4 is*

$$\mathcal{O}\left(\max\left\{ \frac{d}{\bar{w}_1}, \max_{i \in [n]} h_i, \frac{\sigma^2}{n\varepsilon} \left(\frac{1}{n} \sum_{i=1}^{n} h_i \right) \right\} \frac{L\Delta}{\varepsilon} \right) \tag{5}$$

seconds, where $\bar{w}_1 \equiv \min_{\{i,j\} \in F} w_{ij}$ is the minimum edge weight in the Gomory–Hu tree $T = (V, F, w)$ of $\bar{G}$. Equivalently, $\bar{w}_1$ is the global minimum cut value of G.

Proof. Notice that Leon SGD (Algorithm 4) is a Minibatch SGD method with steps

$$x^{k+1} = x^k - \frac{\gamma}{n} \sum_{i=1}^{n} \frac{1}{B_i} \sum_{j=1}^{B_i} \nabla f_i(x^k; \xi_{ij}^k). \tag{10}$$

Similarly to (Tyurin & Richtárik, 2023; 2024), using the standard SGD analysis (Lan, 2020), one can show that this method converges after $\mathcal{O}\left(\frac{L\Delta}{\varepsilon}\right)$ iterations due to the fact that $\left(\frac{1}{n}\sum_{i=1}^{n}\frac{1}{B_i}\right)^{-1} \geq \max\{\lceil \sigma^2/\varepsilon \rceil, n\}/n$. It is left to bound the time of one iteration under Assumption 1.4. The time to collect the minibatch in (10) is the same as in (Tyurin & Richtárik, 2023; 2024) and can be bounded by

$$\mathcal{O}\left(\max\left\{ \max_{i \in [n]} h_i, \frac{\sigma^2}{n\varepsilon} \left(\frac{1}{n} \sum_{i=1}^{n} h_i \right) \right\} \right) \tag{11}$$

(e.g., see Theorem A.4 in (Tyurin & Richtárik, 2023)). The time to run AllReduce can be upper bounded by

$$\mathcal{O}\left(\frac{d}{\min\limits_{\{i,j\} \in F} w_{ij}} \right) \tag{12}$$

due to Theorem F.1, since $\bar{w}_1 := \min\limits_{\{i,j\} \in F} w_{ij}$ is the smallest value of a min-cut in the graph $\bar{G}$, which is the same as the value of a minimum $[n]$-cut in the graph $\bar{G}$. It remains to sum (11) and (12) and multiply them by $\mathcal{O}\left(\frac{L\Delta}{\varepsilon}\right)$. □

F.3 Proof of Corollary 6.1

Corollary 6.1 (Sparse Graphs; Proof in Section F.3)**.** *In view of Theorems 3.2 and 1.7, if i) $h_i = h$ for all $i \in [n]$; ii) the degree of every node in G is bounded by a small number $p = \Theta(1)$; iii) and all edges have the same bandwidth b. When $\sigma^2/\varepsilon \geq \Omega(d/bh)$, the optimal time complexity for solving both (1) and (8) is $\tilde{\Theta}\left(dL\Delta/b\varepsilon + h\sigma^2 L\Delta/n\varepsilon^2 + hL\Delta/\varepsilon\right)$ and it is achieved by Leon SGD.*

Proof. Under the new assumptions, the lower bound in Theorem 3.2, proved for both the *homogeneous* and *heterogeneous* settings, is greater than or equal to

$$\tilde{\Omega}\left(\min\left\{\underbrace{\frac{dL\Delta}{pb\varepsilon}+\frac{hL\Delta}{\varepsilon}+\frac{h\sigma^2L\Delta}{n\varepsilon^2}}_{L_1:=},\underbrace{\frac{hL\Delta}{\varepsilon}+\frac{h\sigma^2L\Delta}{\varepsilon^2}}_{L_2:=}\right\}\right) \tag{13}$$

since the weights in the corresponding Gomory-Hu tree are less than or equal to pb. On the other hand, the upper bound by Leon SGD in Theorem 1.7 is less than or equal $U := \mathcal{O}\left({}^{dL\Delta}/_{b\varepsilon} + {}^{hL\Delta}/_{\varepsilon} + {}^{h\sigma^2L\Delta}/_{n\varepsilon^2}\right)$ since all edges have the same bandwidth b. Hero SGD with complexity L_2 can potentially improve U when ε is not too large and n is small. However, in the regime when ε is small and σ is large (${}^{\sigma^2}/_{\varepsilon} \geq \Omega({}^{d}/_{bh})$), the term L_1 can be smaller than or equal to L_2, and in this case, comparing L_1 and U, one can see that L_1 can improve the communication term by at most p times. If $p = \Theta(1)$, i.e., the graph is sparse, then it is infeasible to improve the time complexity U even in the homogeneous setting. Note that U can also be achieved with a naive mini-batch version of Synchronous SGD: $x^{k+1} = x^k - \frac{\gamma}{nB}\sum_{i=1}^n\sum_{j=1}^B \nabla f_i(x^k;\xi_{ij}^k)$ with the simple communication approach from Section H. □

F.4 Proof of Corollaries F.3 and 6.2

Corollary F.3 (Proof in Section F.4). *In view of Theorems 3.2, if $h_i = h$ for all $i \in [n]$ and $b_{ij} = b$ for all $(i,j) \in E$, then the lower bound is $\tilde{\Omega}\left(\min\left\{\min_{m\in[n-1]}\left[\frac{dL\Delta}{mb\varepsilon}+\frac{h\sigma^2L\Delta}{n(m)\varepsilon^2}\right]+\frac{hL\Delta}{\varepsilon},\frac{hL\Delta}{\varepsilon}+\frac{h\sigma^2L\Delta}{\varepsilon^2}\right\}\right)$, where $n(m)$ is the number of nodes having the number of incident edges greater than or equal m.*

Proof. Using (7), the lower bound is

$$LB := \tilde{\Omega}\left(\min_{k\in[n]}\left(\frac{d}{\bar{w}_k}+\min_{p\in[k]}\frac{h\sigma^2}{|S_{k,p}|\,\varepsilon}\right)\frac{L\Delta}{\varepsilon}+\frac{hL\Delta}{\varepsilon}\right)$$

in both homogeneous and heterogeneous settings when $h_i = h$ for all $i \in [n]$. Clearly,

$$LB = \tilde{\Omega}\left(\left(\frac{d}{\bar{w}_{k^*}}+\frac{h\sigma^2}{|S_{k^*,p^*}|\,\varepsilon}\right)\frac{L\Delta}{\varepsilon}+\frac{hL\Delta}{\varepsilon}\right)$$

for some $k^* \in [n]$ and $p^* \in [k^*]$. If $|S_{k^*,p^*}| = 1$, then Corollary F.3 is true. Let $|S_{k^*,p^*}| \geq 2$, then necessarily $k^* \in [n-1]$. Notice that $\bar{w}_{k^*} = i_{k^*}b$ for some $i_{k^*} \in [n-1]$ since all bandwidths equal to b. Thus

$$LB = \tilde{\Omega}\left(\left(\frac{d}{bi_{k^*}}+\frac{h\sigma^2}{|S_{k^*,p^*}|\,\varepsilon}\right)\frac{L\Delta}{\varepsilon}+\frac{hL\Delta}{\varepsilon}\right).$$

It is left to show that $|S_{k^*,p^*}| \leq n(i_{k^*})$. By construction, S_{k^*,p^*} is a set of nodes in which each node has at least i_{k^*} incident edges (if one of them had fewer than i_{k^*} incident edges, then the nodes in S_{k^*,p^*} would have been disconnected in Algorithm 2 at some iteration $j \in [k^*-1]$, because there would exist an edge e in T with $w_e < bi_{k^*}$ that separates two nodes in S_{k^*,p^*}; this leads to a contradiction). Thus, $|S_{k^*,p^*}| \leq n(i_{k^*})$ and

$$LB \geq \tilde{\Omega}\left(\left(\frac{d}{bi_{k^*}}+\frac{h\sigma^2}{n(i_{k^*})\varepsilon}\right)\frac{L\Delta}{\varepsilon}+\frac{hL\Delta}{\varepsilon}\right) \geq \tilde{\Omega}\left(\min_{m\in[n-1]}\left(\frac{d}{bm}+\frac{h\sigma^2}{n(m)\varepsilon}\right)\frac{L\Delta}{\varepsilon}+\frac{hL\Delta}{\varepsilon}\right).$$

□

Corollary 6.2 (Proof in Section F.4; see also dual Corollary F.3). *In view of Theorems 3.2, if $h_i = h$ for all $i \in [n]$ and $b_{ij} = b$ for all $(i,j) \in E$, then the lower bound in all settings is $\tilde{\Omega}\left(\min\left\{\min_{m\in\{2,\dots,n\}}\left[\frac{dL\Delta}{k(m)b\varepsilon}+\frac{h\sigma^2L\Delta}{m\varepsilon^2}\right]+\frac{hL\Delta}{\varepsilon},\frac{hL\Delta}{\varepsilon}+\frac{h\sigma^2L\Delta}{\varepsilon^2}\right\}\right)$, where $k(m)$ is the m-th largest edge degree of a node in G.*

Proof. Similarly to the previous proof, a lower bound is

$$LB = \tilde{\Omega}\left(\left(\frac{d}{\bar{w}_{k^*}} + \frac{h\sigma^2}{|S_{k^*,p^*}|\,\varepsilon}\right)\frac{L\Delta}{\varepsilon} + \frac{hL\Delta}{\varepsilon}\right)$$

for some $k^* \in [n]$ and $p^* \in [k^*]$. If $|S_{k^*,p^*}| = 1$, then the lower bound is true. Let $|S_{k^*,p^*}| \geq 2$ and $\bar{d}_1 \geq \cdots \geq \bar{d}_{|S_{k^*,p^*}|}$ be the node degrees of the nodes in set S_{k^*,p^*}. Notice that $\bar{w}_{k^*} \leq b \times \bar{d}_{|S_{k^*,p^*}|}$ (the case $\bar{w}_{k^*} > b \times \bar{d}_{|S_{k^*,p^*}|}$ is impossible since there exists a node in S_{k^*,p^*} with degree $\bar{d}_{|S_{k^*,p^*}|}$, which would be separated by Algorithm 2 in some iteration $j \in [k^* - 1]$ from another node in S_{k^*,p^*}). Thus,

$$\begin{aligned}
LB &\geq \tilde{\Omega}\left(\left(\frac{d}{\bar{d}_{|S_{k^*,p^*}|}b} + \frac{h\sigma^2}{|S_{k^*,p^*}|\,\varepsilon}\right)\frac{L\Delta}{\varepsilon} + \frac{hL\Delta}{\varepsilon}\right) \\
&\geq \tilde{\Omega}\left(\left(\frac{d}{k(|S_{k^*,p^*}|)b} + \frac{h\sigma^2}{|S_{k^*,p^*}|\,\varepsilon}\right)\frac{L\Delta}{\varepsilon} + \frac{hL\Delta}{\varepsilon}\right) \\
&\geq \tilde{\Omega}\left(\min_{m\in\{2,\ldots,n\}}\left(\frac{d}{k(m)b} + \frac{h\sigma^2}{m\varepsilon}\right)\frac{L\Delta}{\varepsilon} + \frac{hL\Delta}{\varepsilon}\right)
\end{aligned}$$

where we use that $\bar{d}_{|S_{k^*,p^*}|} \leq k(|S_{k^*,p^*}|)$. □

G Lower Bounds

G.1 Proof sketch

In this section, we give a proof sketch of Theorem 1.5.

("Worst-case" function and stochastic oracle). In the first step of the proof, we follow (Carmon et al., 2020; Arjevani et al., 2022; Huang et al., 2022; Tyurin & Richtárik, 2023) and construct a "worst-case" function. However, one crucial detail is that we use the function $F_{T,K,a}$ from (Tyurin, 2026) instead of the function from (Carmon et al., 2020). Recall the function F_T from (Carmon et al., 2020), which has two important properties: i) if an algorithm wants to find an ε–stationary point, then it is necessary to discover[5] the last T^{th} coordinate; ii) an algorithm can discover the new coordinate $j+1$ only if the j^{th} coordinate is discovered. The new construction $F_{T,K,a}$ by Tyurin (2026) generalizes the last property: an algorithm can discover the new coordinate $j+1$ only if the j^{th}, $(j-1)^{\text{th}}$, …, $(j-K+1)^{\text{th}}$ coordinates are discovered. The stochastic oracle is the same as in (Arjevani et al., 2022): it simply zeros out the newly discovered coordinate, the one with the largest index among all discovered coordinates, with probability $1 - p_\sigma$, where $p_\sigma = \Theta\left(\varepsilon/\sigma^2\right)$.

(Time complexity). Recall Algorithm 2 that generates the sequences $\{\bar{w}_k\}_{k\in[n]}$ and $\{S_{k,p}\}_{k\in[n],p\in[k]}$ (see the description in Section 2). Now, consider the lower bound (7), which is equivalent to

$$t^* := \min_{k\in[n]} \bar{t}(k). \tag{14}$$

where

$$\bar{t}(k) := \frac{1}{c_1 \log^{c_2}(n+1)} \times \max\left\{\frac{d}{\bar{w}_k}, \min_{p\in[k]} B_h(\sigma^2/\varepsilon, S_{k,p})\right\}\frac{L\Delta}{\varepsilon}, \tag{15}$$

$c_1 := 2^{43}3^7 5^{14}\pi^2 e^{10}$, $c_2 := 14$, $B_h(\sigma^2/\varepsilon, S) := \min_{m\in[|S|]}\left[\left(\frac{1}{m}\sum_{i=1}^m \frac{1}{h_{\pi_i(S)}}\right)^{-1}\left(1 + \frac{\sigma^2}{\varepsilon m}\right)\right]$, and $\pi(S)$ is a permutation that sorts $\{h_i\}_{i\in S} : h_{\pi_1(S)} \leq \cdots \leq h_{\pi_{|S|}(S)}$.

Our goal now is to show that (14) is a valid lower bound. Notice that $\bar{w}_1 \leq \cdots \leq \bar{w}_n$, and $\{S_{k+1,p}\}_{p\in[k+1]}$ is constructed from $\{S_{k,p}\}_{p\in[k]}$ by splitting one of the sets.

High-level intuition of why (15) is valid lower bound: Assume that an algorithm decides to use all workers ($k = 1$). In this case, all workers are allowed to work together to obtain the best possible

[5] In the paper, when we say that a worker i *discovers* a coordinate with index j, it means that it adds a vector to I_i in which the corresponding value of that coordinate is non-zero.

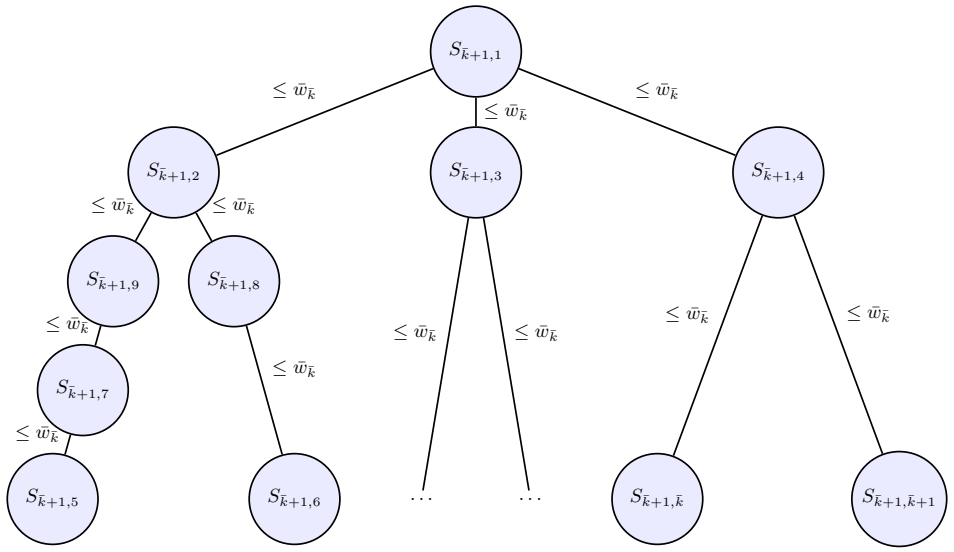


Figure 4: Tree of metanodes corresponding to the partition $\{S_{\bar{k}+1,p}\}_{p\in[\bar{k}+1]}$. Each node represents a subset of workers, and every edge corresponds to a communication link whose bandwidth is at most $\bar{w}_{\bar{k}}$.

computation term $B_h(\sigma^2/\varepsilon, S_{1,1}) = B_h(\sigma^2/\varepsilon, [n])$. However, intuitively, the algorithm must pay the communication price $\ell/\bar{w}_1$ because $\bar{w}_1$ is the smallest possible value of a min-cut in the graph G, and there exist two workers such that the maximum flow between them is bounded by $\bar{w}_1$. **No matter what routing strategy the algorithm chooses, if it wants to transfer a vector of size ℓ between these two workers, it is necessarily required to wait $\ell/\bar{w}_1$ seconds. Therefore, the only way to remove the dependence on $\ell/\bar{w}_1$ is for the algorithm to "disconnect" the workers separated by the bottleneck min-cut $\bar{w}_1$.** This way, we get the sets $S_{2,1}$ and $S_{2,2}$. Notice that $\min\{B_h(\sigma^2/\varepsilon, S_{2,1}), B_h(\sigma^2/\varepsilon, S_{2,2})\} \leq B_h(\sigma^2/\varepsilon, S_{1,1})$.

Recall that $\{\bar{w}_k\}_{k\in[n]}$ and $\{\min_{p\in[k]} B_h(\sigma^2/\varepsilon, S_{k,p})\}_{k\in[n]}$ are non-decreasing. Thus, expect one corner case when $\bar{w}_1$ is large, it means that there exists $\bar{k}$ such that $d/\bar{w}_{\bar{k}} \approx \min_{p\in[\bar{k}]} B_h(\sigma^2/\varepsilon, S_{\bar{k},p})$. In particular, we can show that there exists $\bar{k}$ such that

$$t^* = \frac{1}{c_1 \log^{c_2}(n+1)} \times \min\left\{\frac{d}{\bar{w}_{\bar{k}}}, \min_{p\in[\bar{k}+1]} B_h(\sigma^2/\varepsilon, S_{\bar{k}+1,p})\right\} \frac{L\Delta}{\varepsilon}. \tag{16}$$

In other words, either $\bar{k}$ or $\bar{k}+1$ is the optimal index that balances the two sequences, and the partition $\{S_{\bar{k}+1,p}\}_{p\in[\bar{k}+1]}$ is one that an optimal algorithm would choose to identify the best subset of workers (see Figure 4). Thus, it remains to use this partition of workers and show that, with this partition, the workers would require (16) seconds with high probability.

(**Analysis of the partition**.) In Figure 4, consider the (meta) node $S_{\bar{k}+1,5}$. For simplicity, assume that $h_i = h$ for all $i \in [n]$. There are $\left|S_{\bar{k}+1,5}\right|$ workers in $S_{\bar{k}+1,5}$. If $S_{\bar{k}+1,5}$ were on its own and isolated, then the required time to find an ε–stationary point would be

$$\Theta\left(B_h(\sigma^2/\varepsilon, S_{\bar{k}+1,p})\right) \frac{L\Delta}{\varepsilon} = h \times \left(\frac{L\Delta}{\varepsilon} + \frac{\sigma^2 L\Delta}{\left|S_{\bar{k}+1,p}\right| \varepsilon^2}\right)$$

(Arjevani et al., 2022; Tyurin & Richtárik, 2023). However, in the tree, $S_{\bar{k}+1,5}$ is connected to other workers through *one* edge with weight $\leq \bar{w}_k$. It means that the maximal number of coordinates per second received from $[n] \setminus S_{\bar{k}+1,5}$ is bounded by $\bar{w}_k$.

In the construction of the "worst-case" function, we randomly permute the coordinates, meaning that when workers $[n] \setminus S_{\bar{k}+1,5}$ send a sequence of coordinates to $S_{\bar{k}+1,5}$, the probability of sending the "right" coordinate is less than or equal to $\tilde{\mathcal{O}}\left(1/d\right)$. Thus, on average, they have to send d coordinates to provide a "useful" coordinate, which would take at least $d/\bar{w}_{\bar{k}}$ seconds. In this way, we can show

that it would require at least $\approx \min\left\{\frac{d}{\bar{w}_{\bar{k}}}, B_h(\sigma^2/\varepsilon, S_{\bar{k}+1,5})\right\}\frac{L\Delta}{\varepsilon}$ seconds to solve the problem by one of the workers from $S_{\bar{k}+1,5}$.

(**Recursive analysis**.) The final main challenge was to extend this idea not only to the leaves ($\bar{\mathcal{L}}_1 = \{S_{\bar{k}+1,5}, S_{\bar{k}+1,6}, S_{\bar{k}+1,\bar{k}}, S_{\bar{k}+1,\bar{k}+1}, \dots\}$), but to all nodes in Figure 4. Moreover, it should hold for all nodes at the same time. In the full proof, we discovered a recursive technique and the *leaf-branch peeling* procedure that, starting from the leaves, recursively proves a similar result for all nodes. Roughly speaking, using the Chernoff's method, we show

$$\approx \min\left\{\frac{d}{\bar{w}_{\bar{k}}}, B_h(\sigma^2/\varepsilon, S)\right\}\frac{L\Delta}{\varepsilon}$$

is a lower bound for all leaves $S \in \bar{\mathcal{L}}_1$, including $S_{\bar{k}+1,5}$. Then, we show that this is true for all $S \in \bar{\mathcal{B}}_1$, where $\bar{\mathcal{B}}_1$ is the set of "line nodes" incident to $\bar{\mathcal{L}}_1$. In Figure 4, $\bar{\mathcal{B}}_1 = \{S_{\bar{k}+1,7}, S_{\bar{k}+1,9}, S_{\bar{k}+1,8}\}$. Then, we "remove" $\bar{\mathcal{L}}_1 \cup \bar{\mathcal{B}}_1$ and repeat the steps again. In the next step, the new "leaves" $\bar{\mathcal{L}}_2$ are $S_{\bar{k}+1,2}$ and $S_{\bar{k}+1,4}$. Notice that $S_{\bar{k}+1,2}$ has *one edge* connected to $S_{\bar{k}+1,1}$ and two edges connected to $\bar{\mathcal{L}}_1 \cup \bar{\mathcal{B}}_1$. Thus, it has many more sources from which to discover a new coordinate. Nevertheless, recursively, we have already shown that $\bar{\mathcal{L}}_1 \cup \bar{\mathcal{B}}_1$ cannot provide much useful information to $S_{\bar{k}+1,2}$; thus, the only possible direction is the edge corresponding to $S_{\bar{k}+1,1}$. Controlling the "information leak" from only one edge is possible and easier.

In the proof, due to the union bounds, Chernoff's method "accumulates probabilities," and we obtain an exponential dependence on the number of steps $\bar{d}$ in the *leaf-branch peeling* procedure. Luckily, we can show that $\bar{d} = \mathcal{O}(\log n)$, which is sufficient to obtain an optimal lower bound up to logarithmic factors.

The proof of Theorem 1.7 is arguably more straightforward and self-contained, and should be clear once the proof of Theorem 1.5 is understood.

G.2 Full proof

In this section, we provide the full proof of the lower bound in the homogeneous setting. We consider Protocol 5 and Assumption G.1.

Assumption G.1 (Predefined and Random Sparsifiers)**.** Assume that the optimized function $f : \mathbb{R}^d \to \mathbb{R}$ is a randomly generated function such that $f(x) = \hat{f}(Px)$, where $\hat{f}$ is a deterministic function and P is a random permutation drawn from a distribution $\mathcal{D}_P$. For all $i \in [n]$ and $k \geq 0$, an algorithm A is allowed to use any mapping $\mathcal{C}_i^k : \mathbb{R}^d \times \mathbb{S}_\zeta \to [d]^{p_i^k} \times \mathbb{R}^{p_i^k}$ with any $p_i^k \geq 1$ such that $[\mathcal{C}_i^k(s;\zeta)]_j = (\nu_{i,j}^k, c_{i,j}^k \cdot [s]_{\nu_{i,j}^k})$ for all $s \in \mathbb{R}^d$, $\zeta \in \mathbb{S}_\zeta$, and $j \in [p_i^k]$, where $c_{i,j}^k \equiv c_{i,j}^k(\zeta) \in \mathbb{R}$ is an arbitrary random value, and $\nu_{i,j}^k = \nu_{i,j}^k(\zeta) \in [d]$ is an arbitrary random coordinate such that the distribution of $\{\nu_{i,j}^k\}_{k\geq 0, i\in[n], j\in[p_i^k]}$ is independent of $P \sim \mathcal{D}_P$.

In other words, if we change the optimized function or permute the coordinates in f, then the generated indices do not change. Assumption G.1 is one way of expressing that the workers and the algorithm *do not* take into account the local information and do not depend on $\{I_i\}_{i\in[n]}$ or s when choosing the indices in the sparsifiers. It is general enough to support sending the full vector, sending a predefined block of the vector as is done in AllReduce algorithms, or even a random subset of coordinates, supporting Rand$K$ or Perm$K$ compressed communication. For instance, for all $K \in [d]$, Rand$K$, which sends a random subset of coordinates (scaled by d/K) satisfies this assumption with $p = K$ because it samples subsets independently.

Theorem 3.2 (Lower Bound in the Homogeneous Setting)**.** *We consider any computational and communication environment, any graph G satisfying Assumption 1.4. Let $L, \Delta, \varepsilon, n, \sigma^2, d > 0$ be any numbers such that $L\Delta/\varepsilon \geq \bar{c}_1 \log^{10}(n+1)$ and dimension $d \geq \bar{c}_3 L\Delta/\log^3(n+1)\varepsilon$. Consider Protocol 5. For all $i \in [n]$ and $k \geq 0$, mappings $\mathcal{C}_i^k$ satisfy Assumption G.1. Then, for any algorithm A, there exists a function $f \in \mathcal{F}_{\Delta,L}$, exists an oracle that satisfies Assumption 1.3, and*

[6] Calculates $\nabla f_i(x;\xi)$ in the heterogeneous setting.

Protocol 5

Input: Algorithm A
Init $I_i = \emptyset$ (all available information) on worker i for all $i \in [n]$
Run the following two loops in parallel on the workers.
for $i = 1, \ldots, n$ (in parallel on the workers) **do**
  while true **do**
    Algorithm A calculates a new point x based on local information I_i: (takes 0 seconds)
    any vector $x \in \mathbb{R}^d$ such that $\text{supp}(x) \subseteq \cup_{y \in I_i} \text{supp}(y)$ ($\text{supp}(v) := \{ i \in [d] : v_i \neq 0 \}$)
    Calculate one stochastic gradient[6] $\nabla f(x; \xi)$, $\xi \sim \mathcal{D}_\xi$ (ξ are i.i.d.) (takes h_i seconds)
    Add $\nabla f(x; \xi)$ to I_i (takes 0 seconds)
    Optionally wait before starting the next calculation
  end while
end for
for $i = 1, \ldots, n$ (in parallel on the workers) **do**
  for $k = 0, 1, \ldots$ **do**
    Algorithm A calculates a new point s based on local information I_i: (takes 0 seconds)
    any vector $s \in \mathbb{R}^d$ such that $\text{supp}(s) \subseteq \cup_{y \in I_i} \text{supp}(y)$
    Algorithm A chooses any destination worker j (takes 0 seconds)
    Compute the transformation $\mathcal{C}_i^k(s; \zeta)$, $\zeta \sim \mathcal{D}_\zeta$ (takes 0 seconds; Assumption G.1)
    Asynchronously send $\mathcal{C}_i^k(s; \zeta)$ to worker j through G using any feasible routing strategy (the protocol does not wait for the send to complete); add[(a)] $\mathcal{C}_i^k(s; \zeta)$ to I_j once it arrives (the time depends on G, the chosen routing strategy, and the current network load)
    Optionally wait before starting the next send
  end for
end for
(a vector may be added to I_j while the algorithm computes a new point; in this case, the protocol first adds the vector, incurring no delay since the operation takes 0 seconds)
(a): The worker that receives the compressed vector decompresses using the operation $\sum_{(\nu, v) \in \mathcal{C}_i^k(s;\zeta)} v \cdot e_\nu$, where e_ν is the ν^{th} vector of the standard basis.

$$\mathbb{E}\left[\inf_{y \in G_t} \|\nabla f(y)\|^2\right] > \varepsilon \text{ for all}$$

$$t \leq \frac{\bar{c}_2}{\log^{14}(n+1)} \min_{k \in [n]} \left(\frac{d}{\bar{w}_k} + \min_{p \in [k]} \min_{m \in [|S_{k,p}|]} \left[\left(\frac{1}{m} \sum_{i=1}^{m} \frac{1}{h_{\pi_i(S_{k,p})}} \right)^{-1} \left(1 + \frac{\sigma^2}{m\varepsilon} \right) \right] \right) \frac{L\Delta}{\varepsilon}, \tag{7}$$

where G_t is the set of all possible points that can be constructed by A up to time t based on $\{I_i\}_{i \in [n]}$. The quantities $\pi(S)$, $\{S_{k,p}\}_{k \in [n], p \in [k]}$, and $\{\bar{w}_k\}_{k \in [n]}$ are defined in Algorithm 2, and $\bar{c}_1$, $\bar{c}_2$, and $\bar{c}_3$ are universal constants.

Proof.
(Step 1: "Worst-case" function). In our proof, we use a slightly modified function by Tyurin (2026), which is based on the "worst-case" function by Carmon et al. (2020). For any $T, K \in \mathbb{N}$, and $e \geq a > 1$, Tyurin (2026) defined the function $F_{T,K,a} : \mathbb{R}^T \to \mathbb{R}$ such that

$$F_{T,K,a}(x) = -\sum_{i=1}^{T} \Psi_a(x_{i-K}) \ldots \Psi_a(x_{i-2}) \Psi_a(x_{i-1}) \Phi(x_i) + \sum_{i=1}^{T} \Gamma(x_i), \tag{17}$$

where x_i is the i^{th} coordinate of a vector $x \in \mathbb{R}^T$ and

$$\Psi_a(x) = \begin{cases} 0, & x \leq 1/2, \\ \exp\left(\log a \cdot \left(1 - \frac{1}{(2x-1)^2} \right) \right), & x > 1/2, \end{cases} \qquad \Phi(x) = \sqrt{e} \int_{-\infty}^{x} e^{-\frac{1}{2}t^2} dt,$$

and

$$\Gamma(x) = \begin{cases} -xe^{1/x+1}, & x < 0, \\ 0, & x \geq 0. \end{cases}$$

We assume that $x_0 = \cdots = x_{-K+1} \equiv 1$. Throughout the lower bound analysis, we assume that $e \geq a > 1$ in Ψ_a. We also define

$$\text{supp}(v) := \{\, i \in [d] : v_i \neq 0 \,\} \qquad \forall v \in \mathbb{R}^d$$

and

$$\begin{aligned}&\text{prog}^K(x) := \max\{i \geq 0 \,|\, x_i \neq 0, x_{i-1} \neq 0, \ldots, x_{i-K+1} \neq 0\}\\ &(x_0 = \cdots = x_{-K+1} \equiv 1),\end{aligned}$$

which extends the standard progress operator $\text{prog}(x) \equiv \text{prog}^1(x) := \max\{i \geq 0 \,|\, x_i \neq 0\}$ ($x_0 \equiv 1$). This function has the following properties proved in (Tyurin, 2026):

Lemma G.2. *For all $x \in \mathbb{R}^T$, $\text{supp}(\nabla F_{T,K,a}(x)) \subseteq \{1, \ldots, \text{prog}^K(x) + 1\} \cup \text{supp}(x)$, where $\text{supp}(v) := \{\, i \in [d] : v_i \neq 0 \,\}$ for all $d \geq 1$ and $v \in \mathbb{R}^d$*

Lemma G.3. *For all $x \in \mathbb{R}^T$, if $\text{prog}^K(x) < T$, then $\|\nabla F_{T,K,a}(x)\| > 1$.*

Lemma G.4. *Function $F_{T,K,a}$ satisfies*

$$F_{T,K,a}(0) - \inf_{x \in \mathbb{R}^T} F_{T,K,a}(x) \leq \Delta^0(K,a) \cdot T,$$

where $\Delta^0(K,a) := \sqrt{2\pi e} \cdot a^K$.

Lemma G.5. *For all $x \in \mathbb{R}^T$, $\|\nabla F_{T,K,a}(x)\|_\infty \leq \gamma_\infty(K,a)$, where $\gamma_\infty(K,a) := 6\sqrt{2\pi}e^{3/2} \cdot \frac{Ka^K}{\sqrt{\log a}}$.*

Lemma G.6. *The function $F_{T,K,a}$ is $\ell_1(K,a)$–smooth, i.e., $\left\|\nabla^2 F_{T,K,a}(x)\right\| \leq \ell_1(K,a)$ for all $x \in \mathbb{R}^T$, where $\ell_1(K,a) := 12\sqrt{2\pi}e^{5/2} \cdot \frac{K^2 a^K}{\log a}$.*

Using this construction, we define a scaled version with random coordinates. We first sample a uniformly random subset $R_T = [R_{T,1}, \ldots, R_{T,T}]$ without repetitions from set $[d]$ ($R_{T,1} < \cdots < R_{T,T}$). Let us take any $\lambda > 0$, $d, T \in \mathbb{N}$, $d \geq T$, and take the function $f : \mathbb{R}^d \to \mathbb{R}$ such that

$$f(x) := \frac{L\lambda^2}{\ell_1(K,a)} F_{T,K,a}\left(\frac{x_{[R_T]}}{\lambda}\right), \tag{18}$$

where $\ell_1(K,a)$ is defined in Lemma G.6 and $x_{[R_T]} \in \mathbb{R}^T$ is the subvector of size T of vector $x \in \mathbb{R}^d$ such that $[x_{[R_T]}]_i = [x]_{R_{T,i}}$ for all $i \in [T]$. Notice that the $d - T$ coordinates are artificial. We have to ensure that f is L-smooth and $f(0) - \inf_{x \in \mathbb{R}^d} f(x) \leq \Delta$, Using Lemma G.6,

$$\begin{aligned}\|\nabla f(x) - \nabla f(y)\| &= \frac{L\lambda}{\ell_1(K,a)} \left\|\nabla F_{T,K,a}\left(\frac{x_{[R_T]}}{\lambda}\right) - \nabla F_{T,K,a}\left(\frac{y_{[R_T]}}{\lambda}\right)\right\| \leq L\lambda \left\|\frac{x_{[R_T]}}{\lambda} - \frac{y_{[R_T]}}{\lambda}\right\|\\ &= L\left\|x_{[R_T]} - y_{[R_T]}\right\| \leq L\|x - y\| \quad \forall x, y \in \mathbb{R}^d.\end{aligned}$$

We choose

$$T = \left\lfloor \frac{\Delta \cdot \ell_1(K,a)}{L\lambda^2 \cdot \Delta^0(K,a)} \right\rfloor.$$

Due to Lemma G.4,

$$f(0) - \inf_{x \in \mathbb{R}^d} f(x) = \frac{L\lambda^2}{\ell_1(K,a)}(F_{T,K,a}(0) - \inf_{x \in \mathbb{R}^T} F_{T,K,a}(x)) \leq \frac{L\lambda^2 \Delta^0(K,a) T}{\ell_1(K,a)} \leq \Delta,$$

where $\Delta^0(K,a)$ is defined in Lemma G.4. We also take

$$\lambda = \frac{\sqrt{2\varepsilon}\ell_1(K,a)}{L} \tag{19}$$

to get

$$\|\nabla f(x)\|^2 = \frac{L^2\lambda^2}{\ell_1^2(K,a)} \left\|\nabla F_{T,K,a}\left(\frac{x_{[R_T]}}{\lambda}\right)\right\|^2 = 2\varepsilon \left\|\nabla F_{T,K,a}\left(\frac{x_{[R_T]}}{\lambda}\right)\right\|^2 > 2\varepsilon \cdot \mathbb{1}\left[\text{prog}^K(x_{[R_T]}) < T\right], \tag{20}$$

where the last inequality due to Lemma G.3. Note that

$$T = \left\lfloor \frac{L\Delta}{2\Delta^0(K,a)\cdot \ell_1(K,a)\cdot \varepsilon} \right\rfloor. \tag{21}$$

(Step 2: Stochastic Oracle). We consider a stochastic oracle similar to (Arjevani et al., 2022). We define

$$[\nabla f(x;\xi)]_j := \begin{cases} [\nabla f(x)]_j \times \left(\frac{\xi}{p_\sigma}\right), & \text{for } j = R_{T,i} \text{ if } i \in [T], \text{ where } i = \text{prog}^K\left(x_{[R_T]}\right) + 1, \\ [\nabla f(x)]_j, & \text{for all other } j \in [d], \end{cases} \tag{22}$$

for all $\forall x \in \mathbb{R}^d$ and take $\mathcal{D}_\xi = \text{Bernouilli}(p_\sigma)$, where $p_\sigma \in (0,1]$. For all $x \in \mathbb{R}^d$, $[x]_j$ is the j^{th} coordinate of x. Similarly to (Arjevani et al., 2022), we now show this oracle is unbiased and σ^2-variance-bounded. Clearly, for $j \neq R_{T,i}$, $\mathbb{E}_\xi\left[[\nabla f(x,\xi)]_j\right] = [\nabla f(x)]_j$. Otherwise,

$$\mathbb{E}_\xi\left[[\nabla f(x,\xi)]_j\right] = [\nabla f(x)]_j \frac{\mathbb{E}[\xi]}{p_\sigma} = [\nabla f(x)]_j,$$

and

$$\mathbb{E}_\xi\left[\|\nabla f(x;\xi) - \nabla f(x)\|^2\right] \le \max_{j\in[d]} |[\nabla f(x)]_j|^2\, \mathbb{E}\left[\left(\frac{\xi}{p_\sigma} - 1\right)^2\right]$$

because the difference is non-zero only in one coordinate. Thus

$$\begin{aligned}\mathbb{E}_\xi\left[\|\nabla f(x,\xi) - \nabla f(x)\|^2\right] &\le \frac{\|\nabla f(x)\|_\infty^2 (1-p_\sigma)}{p_\sigma} = \frac{L^2\lambda^2 \left\|\nabla F_{T,K,a}\left(\frac{x_{[R_T]}}{\lambda}\right)\right\|_\infty^2 (1-p_\sigma)}{\ell_1^2(K,a)p_\sigma} \\ &\le \frac{L^2\lambda^2\gamma_\infty^2(K,a)(1-p_\sigma)}{\ell_1^2(K,a)p_\sigma},\end{aligned}$$

where we apply Lemma G.5. Taking

$$p_\sigma = \min\left\{\frac{L^2\lambda^2\gamma_\infty^2(K,a)}{\sigma^2\ell_1^2(K,a)}, 1\right\} \overset{(19)}{=} \min\left\{\frac{2\varepsilon\gamma_\infty^2(K,a)}{\sigma^2}, 1\right\}, \tag{23}$$

we get $\mathbb{E}_\xi\left[\|\nabla f(x,\xi) - \nabla f(x)\|^2\right] \le \sigma^2$.

(Step 3: Graph Analysis). We now consider the defined graph $G = (V, E, b)$. The first step is to construct an undirected version $\bar{G}$ of the graph G, where we keep the same set of nodes and retain only one edge from $\{(i,j),(j,i)\}$ for each $(i,j) \in E$ with the same weight b_{ij}, making this edge unordered. Thus, $\bar{G} = (V, \bar{E}, b)$, where $\{i,j\} \in \bar{E}$ with weight $b_{ij} > 0$ if and only if $(i,j) \in E$ with weight $b_{ij} > 0$.

The second substep is to construct a Gomory–Hu tree $T = (V, F, w)$ of $\bar{G}$, which in general is not unique but always exists. Notice that, by definition, we have $w_{ij} = \alpha_{\bar{G}}(i,j)$ for all $\{i,j\} \in F$. Recall the important Theorem E.7. By the max-flow min-cut theorem, the *maximum s–t flow* in $\bar{G}$ is upper bounded by w_{uv}. The *maximum s–t flow* in G is equal to that in $\bar{G}$ and is therefore also upper bounded by w_{uv}, since G is obtained from $\bar{G}$ by replacing each undirected edge $\{i,j\}$ of capacity b_{ij} with two directed edges (i,j) and (j,i), each of capacity $b_{ij} = b_{ji}$. In terms of our problem, the maximum number of coordinates per second that nodes s and t can send to each other is bounded by w_{uv}.

Next, we take the values $\{w_{ij}\}_{ij\in F} \cup \{\infty\}$, sort them in ascending order, and define this list as $\bar{w} := (\bar{w}_1, \ldots, \bar{w}_n)$. The idea is to take all bandwidth thresholds in T; we also include ∞ to capture the case when the nodes do not communicate.

Consider the Gomory–Hu tree T and Algorithm 2. At the beginning, we define the triple $(T_1, (S_{1,1}), \bar{w}_1)$, where $T_1 = T$ and $S_{1,1} = [n]$ is the only connected component of T_1 (since G is a connected graph; see Section 1). Now, inductively, given a triple $(T_k, (S_{k,1}, \ldots, S_{k,k}), \bar{w}_k)$, we remove an edge with value $\bar{w}_k$ in T_k and define the corresponding graph as T_{k+1}. Then, we define the triple $(T_{k+1}, (S_{k+1,1}, \ldots, S_{k+1,k+1}), \bar{w}_{k+1})$, where $(S_{k+1,1}, \ldots, S_{k+1,k+1})$ are the connected components of T_{k+1}. We repeat this procedure until $k = |F| + 1 = n$, when the corresponding triple is $(T_k, (S_{k,1}, \ldots, S_{k,k}), \bar{w}_k)$ with $\bar{w}_k = \infty$ and $(S_{k,1}, \ldots, S_{k,k}) = (\{1\}, \ldots, \{n\})$, which is a collection of singletons.

The idea of this procedure is to take the initial Gomory-Hu tree, and one by one remove the edges, from the smallest to the largest value. This way, we can construct a sequence of connected components/partitions $(S_{k,1}, \ldots, S_{k,k})$, where the number of partitions increases by one since we remove only one edge.

(Step 4: Proposed lower bound). Once the necessary constructions are defined, we now consider

$$t^* := \min_{k \in [n]} \bar{t}(k). \tag{24}$$

where

$$\bar{t}(k) := \frac{1}{c_1 \log^{c_2}(n+1)} \times \max\left\{ \frac{d}{\bar{w}_k}, \min_{p \in [k]} B_h(\sigma^2/\varepsilon, S_{k,p}) \right\} \frac{L\Delta}{\varepsilon},$$

$c_1 := 2^{43} 3^7 5^{14} \pi^2 e^{10}$, $c_2 := 14$,

$$B_h(\sigma^2/\varepsilon, S) := \min_{m \in [|S|]} \left[\left(\frac{1}{m} \sum_{i=1}^{m} \frac{1}{h_{\pi_i(S)}} \right)^{-1} \left(1 + \frac{\sigma^2}{\varepsilon m} \right) \right], \tag{25}$$

and $\pi(S)$ is a permutation that sorts $\{h_i\}_{i \in S} : h_{\pi_1(S)} \leq \cdots \leq h_{\pi_{|S|}(S)}$. We will show that (24) is a valid lower bound.

(Corner case). If $\min_{p \in [k]} B_h(\sigma^2/\varepsilon, S_{k,p}) \geq \frac{d}{\bar{w}_k}$ for all $k \in [n]$, then $t^* = \bar{t}(k)$ with $k = 1$ because $B_h(\sigma^2/\varepsilon, S_1) \geq B_h(\sigma^2/\varepsilon, S_2)$ if $S_1 \subseteq S_2$. In this case,

$$\begin{aligned} t^* &= \frac{1}{c_1 \log^{c_2}(n+1)} \times \min_{p \in [1]} B_h(\sigma^2/\varepsilon, S_{1,p}) \frac{L\Delta}{\varepsilon} = \frac{1}{c_1 \log^{c_2}(n+1)} \times B_h(\sigma^2/\varepsilon, S_{1,1}) \frac{L\Delta}{\varepsilon} \\ &= \frac{1}{c_1 \log^{c_2}(n+1)} \times \min_{m \in [n]} \left[\left(\frac{1}{m} \sum_{i=1}^{m} \frac{1}{h_{\pi_i}} \right)^{-1} \left(1 + \frac{\sigma^2}{\varepsilon m} \right) \right] \frac{L\Delta}{\varepsilon}, \end{aligned} \tag{26}$$

since $S_{1,1} = [n]$, where π is a permutation that sorts $\{h_i\}_{i \in [n]}$. In this corner case, (26) is a lower bound due to (Tyurin & Richtárik, 2023), where the authors consider the setting without communication times. If one starts taking communication times into account, the time complexity of algorithms can only increase.

(General cases). Starting from this point, we only consider general cases when there exists the largest index $\bar{k}$ such that $\min_{p \in [\bar{k}]} B_h(\sigma^2/\varepsilon, S_{\bar{k},p}) < \frac{d}{\bar{w}_{\bar{k}}}$. Notice that $\bar{k} < |F| + 1$, since $\bar{w}_{|F|+1} = \infty$, and $\min_{p \in [\bar{k}+1]} B_h(\sigma^2/\varepsilon, S_{\bar{k}+1,p}) \geq \frac{d}{\bar{w}_{\bar{k}+1}}$. Moreover, the sequence $\left\{ \frac{d}{\bar{w}_{\bar{k}}} \right\}_k$ is non-increasing and $\left\{ \min_{p \in [k]} B_h(\sigma^2/\varepsilon, S_{k,p}) \right\}_k$ is non-decreasing, where the latter follows from the definition of the partitions $\{(S_{k,1}, \ldots, S_{k,k})\}_k$ and the fact that $B_h(\sigma^2/\varepsilon, S_1) \geq B_h(\sigma^2/\varepsilon, S_2)$ if $S_1 \subseteq S_2$. In total, we can conclude that

$$t^* = \min\left\{ \bar{t}(\bar{k}), \bar{t}(\bar{k}+1) \right\} = \frac{1}{c_1 \log^{c_2}(n+1)} \times \min\left\{ \frac{d}{\bar{w}_{\bar{k}}}, \min_{p \in [\bar{k}+1]} B_h(\sigma^2/\varepsilon, S_{\bar{k}+1,p}) \right\} \frac{L\Delta}{\varepsilon}. \tag{27}$$

(Step 5: Concentration analysis). Recall the construction (18), where the function depends only on the coordinates indexed by R_T. We now split R_T into $B = \lfloor T/K \rfloor$ blocks of size K with a possible residue of size $T - BK$. The coordinates $B_1 := [R_{T,1}, \ldots, R_{T,K}]$ belong to block 1, $B_2 := [R_{T,K+1}, \ldots, R_{T,2K}]$ belong to block 2, and so forth. Let us define y_i^k as the first time moment

when worker i can start discovering[7] a new coordinate in the $k+1^{\text{th}}$ block, and $y^k := \min_{i\in[n]} y_i^k$ as the time when any of them. Notice that $y_i^0 = 0$ for all $i \in [n]$.

Consider the partition $(S_{\bar{k}+1,1}, \ldots, S_{\bar{k}+1,\bar{k}+1})$ of the workers. Recall that this partition is constructed from the tree T by removing $\bar{k}$ edges such that all removed edges $\{u, v\}$ satisfy $w_{uv} \leq \bar{w}_{\bar{k}}$. Notice that this partition decomposes the tree T into subtrees $\bar{T}_1, \ldots, \bar{T}_{\bar{k}+1}$, which are connected to each other by the removed edges. Thus, we can consider a new meta tree $\bar{T}$ whose nodes are $\bar{V} := (S_{\bar{k}+1,1}, \ldots, S_{\bar{k}+1,\bar{k}+1})$ and whose edges are the removed edges. For all $S \in \bar{V}$, we define $\bar{E}(S)$ as the set of edges incident to S.

In this meta tree $\bar{T}$, we run the following *leaf-branch peeling* procedure. For all $p \geq 1$, let $\bar{\mathcal{L}}_p$ be the (non-empty) set of leaves of $\bar{T}_p := \bar{T} \setminus \bigcup_{i<p} \left(\bar{\mathcal{L}}_i \bigcup \bar{\mathcal{B}}_i\right)$ of size $\bar{n}_p > 0$. Define $\bar{\mathcal{B}}_p = \emptyset$, and recursively add to $\bar{\mathcal{B}}_p$ all nodes from $\bar{T}_p$ with 2 edges that have a neighbor in $\bar{\mathcal{L}}_p$ or $\bar{\mathcal{B}}_p$: first, add all nodes with 2 edges that have a neighboring node in $\bar{\mathcal{L}}_p$; then add all nodes with 2 edges that have a neighboring node in $\bar{\mathcal{B}}_p$, and repeat the last step until no candidate nodes remain.

Stop the procedure when $\bar{T}_{\bar{d}+1}$ is empty, where $\bar{d}$ is the number of steps in the procedure. It is possible to prove the following logarithmic bound on the number of steps $\bar{d}$:

Lemma G.7 (Proof in Section G.4)**.** *The number of steps $\bar{d}$ in the leaf-branch peeling procedure satisfies*

$$\bar{d} \leq \lfloor \log_2(n+2) \rfloor .$$

In the meta graph $\bar{T}$, consider any node $\bar{S}$ in $\bar{V}$. Also, consider any $i \in \bar{S}$. There are **two ways** to discover one of the K coordinates in block k: (1) either one of the workers (potentially worker i itself, but not necessarily) from the group $\bar{S}$ discovers a coordinate by computing a stochastic gradient with $\xi = 1$, or (2) group $\bar{S}$ discovers a coordinate through one of the communication channels from one of other groups in $\bar{T}$.

First way to discover coordinates. Before time y^k, for all $i \in [n]$, worker i has discovered at most coordinates $[R_{T,1}, \ldots, R_{T,(k-1)K}] \cup [j_1, \ldots, j_p]$, where $[j_1, \ldots, j_p] \subset B_k := [R_{T,1+(k-1)K}, \ldots, R_{T,kK}]$ and $p \geq 0$. Due to Lemma G.2, worker i can discover at most one coordinate in B_k when it calculates a stochastic gradient, at position $\bar{j}$ such that $\bar{j}$ is the smallest index in $B_k \setminus (j_1, \ldots, j_p)$. However, due to the construction (22), worker i can discover $\bar{j}$ only if it receives a "lucky" random Bernoulli variable with value 1. The time required to calculate one stochastic gradient is h_i. Therefore, worker i requires at least $h_i \eta_{k,i,1}$ seconds to discover a new coordinate in B_k, where $\eta_{k,i,1} \sim \text{Geom}(p_\sigma)$.

Similarly, group $\bar{S}$ can discover the first coordinate, that was not discovered by any other node from this group and by computing stochastic gradients, after at least $\min_{i\in\bar{S}} h_i \eta_{k,i,1}$ seconds, where $\{\eta_{k,i,1}\}_{k\geq 1, i\in[n]}$ are i.i.d random variables from $\text{Geom}(p_\sigma)$, and the m^{th} coordinate after at least $\sum_{j=1}^{m} \min_{i\in\bar{S}} h_i \eta_{k,i,j}$ seconds, where $\{\eta_{k,i,j}\}_{k\geq 1, i\in[n], j\geq 1}$ are i.i.d random variables from $\text{Geom}(p_\sigma)$. The min comes from the fact that the workers can calculate in parallel.

Second way to discover coordinates (see Figure 4). At the same time, workers from $\bar{V} \setminus \bar{S}$ can share a new coordinate with the workers from $\bar{S}$ via communication through the graph G. Consider any $\hat{S} \in \bar{V} \setminus \bar{S}$. Recall that the workers in $\bar{S}$ are separated from the workers in $\hat{S}$ in the Gomory–Hu tree T by the $\bar{S}$'edge of weight w_{uv} such that $w_{uv} \leq \bar{w}_{\bar{k}}$, meaning that worker $i \in \bar{S}$ is separated from the workers in $\hat{S}$ by a cut of value less than $\bar{w}_{\bar{k}}$ (Theorem E.7). Thus, the maximal flow (number of coordinates per second) that worker $i \in \bar{S}$ can receive from $\hat{S}$ is less than or equal $\bar{w}_{\bar{k}}$. After at least time y^k, workers $\hat{S}$ can start sending a sequence of coordinates with indices $(\nu_1, \nu_2, \ldots)$, where worker $i \in \bar{S}$ can discover a new coordinate. In order to discover a new coordinate, worker $i \in \bar{S}$ should receive some ν_j such that $\nu_j \in B_k := [R_{T,1+(k-1)K}, \ldots, R_{T,kK}]$.

Every edge e of $\bar{S}$ in $\bar{T}$ separates $\bar{S}$ from the set of groups $\mathcal{S}$ on the other side of the edge. For all $k \geq 1$, let $\{\nu_{k,\bar{S},e,j}\}_{j\geq 1}$ be the sequence of coordinates sent by $\mathcal{S}$ to $\bar{S}$ after the moment when one

[7] In the paper, when we say that a worker i *discovers* a coordinate with index j, it means that it adds a vector to I_i in which the corresponding value of that coordinate is non-zero.

of the workers from $\mathcal{S}$ can start discovering k^{th} block. We define $\mu_{k,\bar{S},e,1}$ as the number of received coordinates $\{\nu_{k,\bar{S},e,j}\}_{j\geq 1}$ from $\mathcal{S}$ until the moment when a received coordinate belongs to block k in any worker i from $\bar{S}$. Similarly, let $\mu_{k,\bar{S},e,p}$ be the number of received coordinates until the moment when a received coordinate belongs to block k, after the $(p-1)^{\text{th}}$ time this has happened in any worker i from $\bar{S}$, and this coordinate does not equal to the previous $(p-1)$ coordinates belonging to block k. In total, starting from y^k, group $\bar{S}$ can discover m coordinates from $\mathcal{S}$ in block k after at least

$$\sum_{j=1}^{m} \frac{\mu_{k,\bar{S},e,j}}{\bar{w}_{\bar{k}}}$$

seconds since it takes $\frac{\ell}{\bar{w}_{\bar{k}}}$ seconds to send ℓ coordinates through the communication channel.

Properties of the random variables. For all $k \geq 0$, we define $\mathcal{G}_k$ as the sigma-algebra generated by $\{\eta_{k',i,j}\}_{1\leq k'\leq k, i\in[n], j\geq 1}$, $\{\mu_{k',S,e,p}\}_{p\geq 1, 1\leq k'\leq k, S\in\bar{V}, e\in\bar{E}(S)}$, and $\{\nu_{k,S,e,j}\}_{k\geq 1, j\geq 1, S\in\bar{V}, e\in\bar{E}(S)}$.

In Section G.3, we prove the following bounds on the probabilities.

Lemma G.8 (Proof in Section G.3)**.** *For all $i \in [n]$,*

$$\mathbb{P}\left(\eta_{k,i,j} \leq t \middle| \{\eta_{k,i,j'}\}_{i\in[n], 1\leq j'<j}, \mathcal{G}_{k-1}\right) \leq \lfloor t \rfloor \, p_\sigma$$

for all $t \geq 0$, $k \geq 1$ and $j \geq 1$. Moreover, consider any $\bar{S} \in \bar{V}$ and $e \in \bar{E}(\bar{S})$, then

$$\mathbb{P}\left(\mu_{k,\bar{S},e,m} \leq t \middle| \mu_{k,\bar{S},e,m-1}, \ldots, \mu_{k,\bar{S},e,1}, \mathcal{G}_{k-1}\right) \leq \frac{Kt}{\max\{\frac{d}{2} - \sum_{p=1}^{m-1} \mu_{k,\bar{S},e,p}, 0\}}$$

for all $t \geq 0$, $k \geq 1$, and $m \geq 1$.

Auxiliary bounds. Before continuing with the proof of Theorem 3.2, we now consider two important lemmas, which follow from Lemma G.8.

Lemma G.9. *For all $S \in \bar{V}$, $k \geq 1$,$K \geq 1$, and $s \geq \frac{128}{B_h(1/p_\sigma, S)}$,*

$$\mathbb{E}\left[\exp\left(-s\sum_{j=1}^{K} \min_{i\in S} h_i \eta_{k,i,j}\right) \middle| \mathcal{G}_{k-1}\right] \leq \frac{1}{8^K}. \tag{28}$$

Lemma G.10. *For all $S \in \bar{V}$, $e \in \bar{E}(S)$, $k \geq 1$,$K \geq 1$, $w > 0$, and $s \geq \frac{1024Kw}{d}$,*

$$\mathbb{E}\left[\exp\left(-s\sum_{j=1}^{K} \frac{\mu_{k,S,e,j}}{w}\right) \middle| \mathcal{G}_{k-1}\right] \leq \frac{1}{8^K}. \tag{29}$$

We now continue with the proof of Theorem 3.2.

(Step 6: Concentration bound). Recall that $\bar{\mathcal{L}}_1$ is the set of leaves in $\bar{V}$, where each leave has at most one neighbor. Moreover, recall that we have two ways to discover K coordinates in block k. Since one of the two discovery methods must discover at least $\frac{K}{3}$ coordinates, we can conclude that

$$\min_{i\in S} y_i^{k+1} \geq \min\left\{ y^k + \sum_{j=1}^{\frac{K}{3}} \min_{i\in S} h_i \eta_{k,i,j}, y^k + \sum_{j=1}^{\frac{K}{3}} \frac{\mu_{k,S,e,j}}{\bar{w}_{\bar{k}}} \right\}$$

for all $S \in \bar{\mathcal{L}}_1$, where e is the only edge of S that separates S from all other groups (there may be no other groups, then the second term under the min does not appear). Similarly, for all $p \in [\bar{d}]$ and $S \in \bar{\mathcal{B}}_p$, using the same reasoning,

$$\min_{i\in S} y_i^{k+1} \geq \min\left\{ y^k + \sum_{j=1}^{\frac{K}{3}} \min_{i\in S} h_i \eta_{k,i,j}, y^k + \sum_{j=1}^{\frac{K}{3}} \frac{\mu_{k,S,e_1,j}}{\bar{w}_{\bar{k}}}, y^k + \sum_{j=1}^{\frac{K}{3}} \frac{\mu_{k,S,e_2,j}}{\bar{w}_{\bar{k}}} \right\}$$

for all $S \in \bar{\mathcal{B}}_p$ and $p \in [\bar{d}]$, where e_1 and e_2 are the only edges of S. It is left to get a bound for $\bar{\mathcal{L}}_{p+1}$ with $p \geq 1$. Unlike $\bar{\mathcal{L}}_1$, $\bar{\mathcal{L}}_{p+1}$ with $p \geq 1$ is not a set of leaves and each $S \in \bar{\mathcal{L}}_{p+1}$ might have potentially a large number of edges. Consider any $S \in \bar{\mathcal{L}}_{p+1}$. It has at most one outgoing edge e that connects S with the nodes from $\bar{V} \setminus \bigcup_{j=1}^{p} \left(\bar{\mathcal{L}}_j \cup \bar{\mathcal{B}}_j\right)$, and all other edges connect it directly to $\bigcup_{j=1}^{p} \left(\bar{\mathcal{L}}_j \cup \bar{\mathcal{B}}_j\right)$ by the construction. Thus, for all $S \in \bar{\mathcal{L}}_{p+1}$,

$$\min_{i \in S} y_i^{k+1} \geq \min \left\{ y^k + \sum_{j=1}^{\frac{K}{3}} \min_{i \in S} h_i \eta_{k,i,j}, y^k + \sum_{j=1}^{\frac{K}{3}} \frac{\mu_{k,S,e,j}}{\bar{w}_{\bar{k}}}, \min_{S \in \cup_{j=1}^{p} \left(\bar{\mathcal{L}}_j \cup \bar{\mathcal{B}}_j\right)} \min_{i \in S} y_i^k \right\},$$

where the last term in $\min$ comes from the fact that $\min_{S \in \cup_{j=1}^{p} \left(\bar{\mathcal{L}}_j \cup \bar{\mathcal{B}}_j\right)} \min_{i \in S} y_i^k$ is the earliest time when nodes from $\cup_{j=1}^{p} \left(\bar{\mathcal{L}}_j \cup \bar{\mathcal{B}}_j\right)$ can start sharing coordinates from block k.

Since

$$y^k = \min_{S \in \cup_{j=1}^{\bar{d}} \left(\bar{\mathcal{L}}_j \cup \bar{\mathcal{B}}_j\right)} \min_{i \in S} y_i^k$$

for all $k \geq 0$, it is sufficient to analyze the sequences

$$\bar{y}_S^{k+1} := \min \left\{ \bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \min_{i \in S} h_i \eta_{k,i,j}, \bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \frac{\mu_{k,S,e,j}}{\bar{w}_{\bar{k}}} \right\} \tag{30}$$

for all $S \in \bar{\mathcal{L}}_1$,

$$\bar{y}_S^{k+1} := \min \left\{ \bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \min_{i \in S} h_i \eta_{k,i,j}, \bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \frac{\mu_{k,S,e_1,j}}{\bar{w}_{\bar{k}}}, \bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \frac{\mu_{k,S,e_2,j}}{\bar{w}_{\bar{k}}} \right\} \tag{31}$$

for all $S \in \bar{\mathcal{B}}_p$ and $p \in [\bar{d}]$,

$$\bar{y}_S^{k+1} := \min \left\{ \bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \min_{i \in S} h_i \eta_{k,i,j}, \bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \frac{\mu_{k,S,e,j}}{\bar{w}_{\bar{k}}}, \min_{S \in \cup_{j=1}^{p} \left(\bar{\mathcal{L}}_j \cup \bar{\mathcal{B}}_j\right)} \bar{y}_S^k \right\}, \tag{32}$$

for all $p \in \{1, \ldots, \bar{d} - 1\}$ and $S \in \bar{\mathcal{L}}_{p+1}$, and

$$\bar{y}^{k+1} := \min_{S \in \cup_{j=1}^{\bar{d}} \left(\bar{\mathcal{L}}_j \cup \bar{\mathcal{B}}_j\right)} \bar{y}_S^{k+1}$$

with $\bar{y}^k = 0$ and $\bar{y}_S^k = 0$ for $k = 0$ and $S \in \cup_{j=1}^{\bar{d}} \left(\bar{\mathcal{L}}_j \cup \bar{\mathcal{B}}_j\right)$. Inductively, one can easily show that $y^k \geq \bar{y}^k$ for all $k \geq 0$.

Using mathematical induction, taking

$$s = 1024 \max \left\{ \frac{1}{\min_{S \in \bar{V}} B_h(1/p_\sigma, S)}, \frac{K \bar{w}_{\bar{k}}}{d} \right\}, \tag{33}$$

we now prove that

$$\mathbb{E}\left[\exp\left(-s\bar{y}^k\right)\right] \leq e^{-(k+1-K)} \tag{34}$$

and

$$\mathbb{E}\left[\exp\left(-s \min_{S \in \bar{\mathcal{L}}_p \cup \bar{\mathcal{B}}_p} \bar{y}_S^k\right)\right] \leq \frac{3 \cdot 2^{p-1} e^{p-1} n^p p^p}{2^K} e^{-(k-K)} \tag{35}$$

for all $k \geq 0$ and $p \in [\bar{d}]$. Notice that it is true for $k = 0$. Consider the inequalities

$$\begin{aligned} \mathbb{E}\left[\exp\left(-s \min_{S \in \bar{\mathcal{L}}_1 \cup \bar{\mathcal{B}}_1} \bar{y}_S^{k+1}\right)\right] &\leq \mathbb{E}\left[\exp\left(-s \min_{S \in \bar{\mathcal{L}}_1} \bar{y}_S^{k+1}\right)\right] + \mathbb{E}\left[\exp\left(-s \min_{S \in \bar{\mathcal{B}}_1} \bar{y}_S^{k+1}\right)\right] \\ &\leq \sum_{S \in \bar{\mathcal{L}}_1} \mathbb{E}\left[\exp\left(-s\bar{y}_S^{k+1}\right)\right] + \sum_{S \in \bar{\mathcal{B}}_1} \mathbb{E}\left[\exp\left(-s\bar{y}_S^{k+1}\right)\right]. \end{aligned} \tag{36}$$

Using (30),

$$\begin{aligned}
&\mathbb{E}\left[\exp\left(-s\bar{y}_S^{k+1}\right)\right] \\
&\leq \mathbb{E}\left[\exp\left(-s\min\left\{\bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \min_{i\in S} h_i\eta_{k,i,j}, \bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \frac{\mu_{k,S,e,j}}{\bar{w}_{\bar{k}}}\right\}\right)\right] \\
&\leq \mathbb{E}\left[\exp\left(-s\left(\bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \min_{i\in S} h_i\eta_{k,i,j}\right)\right)\right] + \mathbb{E}\left[\exp\left(-s\left(\bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \frac{\mu_{k,S,e,j}}{\bar{w}_{\bar{k}}}\right)\right)\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[\exp\left(-s\left(\sum_{j=1}^{\frac{K}{3}} \min_{i\in S} h_i\eta_{k,i,j}\right)\right)\middle| \mathcal{G}_{k-1}\right]\exp\left(-s\bar{y}^k\right)\right] \\
&\quad + \mathbb{E}\left[\mathbb{E}\left[\exp\left(-s\left(\sum_{j=1}^{\frac{K}{3}} \frac{\mu_{k,S,e,j}}{\bar{w}_{\bar{k}}}\right)\right)\middle| \mathcal{G}_{k-1}\right]\exp\left(-s\bar{y}^k\right)\right]
\end{aligned}$$

for all $S \in \bar{\mathcal{L}}_1$. Using (28), (29), and (34),

$$\mathbb{E}\left[\exp\left(-s\bar{y}_S^{k+1}\right)\right] \leq \frac{2}{2^K}e^{-(k+1-K)} = \frac{2}{2^K}e^{-(k+1-K)}$$

for all $S \in \bar{\mathcal{L}}_1$ and our choice of s in (33). For all $S \in \bar{\mathcal{B}}_1$, using the same derivations, we can show that

$$\begin{aligned}
&\mathbb{E}\left[\exp\left(-s\bar{y}_S^{k+1}\right)\right] \\
&\leq \mathbb{E}\left[\exp\left(-s\left(\bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \min_{i\in S} h_i\eta_{k,i,j}\right)\right)\right] + \mathbb{E}\left[\exp\left(-s\left(\bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \frac{\mu_{k,S,e_1,j}}{\bar{w}_{\bar{k}}}\right)\right)\right] \\
&\quad + \mathbb{E}\left[\exp\left(-s\left(\bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \frac{\mu_{k,S,e_2,j}}{\bar{w}_{\bar{k}}}\right)\right)\right] \leq \frac{3}{2^K}e^{-(k+1-K)} = \frac{3}{2^K}e^{-(k+1-K)}.
\end{aligned}$$

Substituting to (36),

$$\mathbb{E}\left[\exp\left(-s\min_{S\in\bar{\mathcal{L}}_1\cup\bar{\mathcal{B}}_1}\bar{y}_S^{k+1}\right)\right] \leq \frac{3\left(\left|\bar{\mathcal{L}}_1\right| + \left|\bar{\mathcal{B}}_1\right|\right)}{2^K}e^{-(k+1-K)} \leq \frac{3\cdot n}{2^K}e^{-(k+1-K)}.$$

We have proved (35) for $p = 1$ and $k \to k+1$. For $p \geq 1$, let us consider

$$\mathbb{E}\left[\exp\left(-s\min_{S\in\bar{\mathcal{L}}_{p+1}\cup\bar{\mathcal{B}}_{p+1}}\bar{y}_S^{k+1}\right)\right] \leq \sum_{S\in\bar{\mathcal{L}}_{p+1}}\mathbb{E}\left[\exp\left(-s\bar{y}_S^{k+1}\right)\right] + \sum_{S\in\bar{\mathcal{B}}_{p+1}}\mathbb{E}\left[\exp\left(-s\bar{y}_S^{k+1}\right)\right].$$

Similarly, to the base case, $\bar{\mathcal{B}}_{p+1}$ is a set of nodes with 2 edges. Thus, using (31), (28), (29), and (34),

$$\mathbb{E}\left[\exp\left(-s\bar{y}_S^{k+1}\right)\right] \leq \frac{3}{2^K}e^{-(k+1-K)}.$$

for all $S \in \bar{\mathcal{B}}_{p+1}$. For all $S \in \bar{\mathcal{L}}_{p+1}$, using (32),

$$\begin{aligned}
&\mathbb{E}\left[\exp\left(-s\bar{y}_S^{k+1}\right)\right] \\
&\leq \mathbb{E}\left[\exp\left(-s\min\left\{\bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \min_{i\in S} h_i\eta_{k,i,j}, \bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \frac{\mu_{k,S,e,j}}{\bar{w}_{\bar{k}}}, \min_{S\in\cup_{j=1}^{p}\left(\bar{\mathcal{L}}_j\cup\bar{\mathcal{B}}_j\right)}\bar{y}_S^k\right\}\right)\right] \\
&\leq \mathbb{E}\left[\exp\left(-s\left(\bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \min_{i\in S} h_i\eta_{k,i,j}\right)\right)\right] + \mathbb{E}\left[\exp\left(-s\left(\bar{y}^k + \sum_{j=1}^{\frac{K}{3}} \frac{\mu_{k,S,e,j}}{\bar{w}_{\bar{k}}}\right)\right)\right] \\
&\quad + \mathbb{E}\left[\exp\left(-s\min\left(\min_{S\in\cup_{j=1}^{p}\left(\bar{\mathcal{L}}_j\cup\bar{\mathcal{B}}_j\right)}\bar{y}_S^k\right)\right)\right].
\end{aligned}$$

Using (28), (29), and (34),

$$\mathbb{E}\left[\exp\left(-s\bar{y}_S^{k+1}\right)\right] \leq \frac{2}{2^K}e^{-(k+1-K)} + \mathbb{E}\left[\exp\left(-s\min_{S\in\cup_{j=1}^{p}\left(\bar{\mathcal{L}}_j\cup\bar{\mathcal{B}}_j\right)}\bar{y}_S^k\right)\right],$$

where we apply (29) with $K \to \frac{K}{3}$. Notice that

$$\mathbb{E}\left[\exp\left(-s\min_{S\in\cup_{j=1}^{p}\left(\bar{\mathcal{L}}_j\cup\bar{\mathcal{B}}_j\right)}\bar{y}_S^k\right)\right] \leq p\times\frac{3\cdot 2^{p-1}e^{p-1}n^p p^p}{2^K}e^{-(k-K)} = \frac{3\cdot 2^{p-1}e^{p-1}n^p p^{p+1}}{2^K}e^{-(k-K)}. \tag{37}$$

due to (35). Using (37),

$$\begin{aligned}\sum_{S\in\bar{\mathcal{L}}_{p+1}}\mathbb{E}\left[\exp\left(-s\bar{y}_S^{k+1}\right)\right] &\leq \frac{2\left|\bar{\mathcal{L}}_{p+1}\right|}{2^K}e^{-(k+1-K)} + \frac{3\cdot 2^{p-1}e^{p-1}n^p p^{p+1}}{2^K}e^{-(k-K)}\left|\bar{\mathcal{L}}_{p+1}\right| \\ &\leq \frac{2\left|\bar{\mathcal{L}}_{p+1}\right|}{2^K}e^{-(k+1-K)} + \frac{3\cdot 2^{p-1}e^{p-1}n^{p+1}p^{p+1}}{2^K}e^{-(k-K)}.\end{aligned}$$

since $\left|\bar{\mathcal{L}}_{p+1}\right| \leq n$. Therefore,

$$\begin{aligned}\mathbb{E}\left[\exp\left(-s\min_{S\in\bar{\mathcal{L}}_{p+1}\cup\bar{\mathcal{B}}_{p+1}}\bar{y}_S^{k+1}\right)\right] &\leq \frac{3\cdot 2^{p-1}e^{p-1}n^{p+1}p^{p+1}}{2^K}e^{-(k-K)} + \frac{2\left|\bar{\mathcal{L}}_{p+1}\right|}{2^K}e^{-(k+1-K)} + \frac{3\left|\bar{\mathcal{B}}_{p+1}\right|}{2^K}e^{-(k+1-K)} \\ &\leq \frac{3\cdot 2^{p-1}e^{p}n^{p+1}p^{p+1}}{2^K}e^{-(k+1-K)} + \frac{3n}{2^K}e^{-(k+1-K)} \\ &\leq \frac{3\cdot 2^{p}e^{p}n^{p+1}(p+1)^{p+1}}{2^K}e^{-(k+1-K)}.\end{aligned} \tag{38}$$

for all $p \geq 1$. We have proved the next step of (35) for $k \to k+1$. It left to prove (34) for $k \to k+1$. Since $\bar{V} = \cup_{j=1}^{\bar{d}}\left(\bar{\mathcal{L}}_j \cup \bar{\mathcal{B}}_j\right)$,

$$\mathbb{E}\left[\exp\left(-s\bar{y}^{k+1}\right)\right] \overset{(38)}{\leq} \bar{d}\times\frac{3\cdot 2^{\bar{d}-1}e^{\bar{d}-1}n^{\bar{d}}\bar{d}^{\bar{d}}}{2^K}e^{-(k+1-K)} = \frac{3\cdot 2^{\bar{d}-1}e^{\bar{d}-1}n^{\bar{d}}\bar{d}^{\bar{d}+1}}{2^K}e^{-(k+1-K)}. \tag{39}$$

Notice that

$$\log_2\left(3\cdot 2^{\bar{d}-1}e^{\bar{d}-1}n^{\bar{d}}\bar{d}^{\bar{d}+1}\right) = \log_2(3) + (1+\log_2 e)(\bar{d}-1) + \bar{d}\log_2(n) + (\bar{d}+1)\log_2(\bar{d})$$

Due to Lemma G.7, $\bar{d} \leq \log_2(n+2)$. Thus,

$$\log_2\left(3\cdot 2^{\bar{d}-1}n^{\bar{d}}\bar{d}^{\bar{d}+1}\right) \leq 8\log_2^2(n+2).$$

It is sufficient to take

$$K = 300\left\lceil\log^2(n+1)\right\rceil \geq 24\log_2^2(n+2) \tag{40}$$

in (39) to ensure that (34) holds for $k \to k+1$:

$$\mathbb{E}\left[\exp\left(-s\bar{y}^{k+1}\right)\right] \leq e^{-(k+2-K)}.$$

(Step 7: Endgame). Using Chernoff's method, (34), and $y^B \geq \bar{y}^B$,

$$\begin{aligned}\mathbb{P}\left(y^B \leq \bar{t}\right) &\leq \mathbb{P}\left(\bar{y}^B \leq \bar{t}\right) \leq \mathbb{P}\left(\exp\left(-s\bar{y}^B\right) \geq \exp\left(-s\bar{t}\right)\right) \leq \exp\left(s\bar{t}\right)\mathbb{E}\left[\exp\left(-s\bar{y}^B\right)\right] \\ &\leq \exp\left(s\bar{t} - (B+1-K)\right) = \delta\end{aligned}$$

for all $\delta \in (0,1]$ and $k \geq 0$, and fixing

$$\bar{t} = \frac{1}{s}\left(\log(\delta) + (B+1-K)\right). \tag{41}$$

Using (20), for all $t \geq 0$,

$$\inf_{x \in G_t} \|\nabla f(x)\|^2 > 2\varepsilon \inf_{x \in G_t} \mathbb{1}\left[\text{prog}^K(x_{[R_T]}) < T\right],$$

where G_t is the set of points computed by the algorithm A up to time t. By construction, if $y^B > t$, then $\inf_{x \in G_t} \mathbb{1}\left[\text{prog}^K(x_{[R_T]}) < T\right] = 1$; thus,

$$\mathbb{E}\left[\inf_{y \in G_t} \|\nabla f(y)\|^2\right] > 2\varepsilon \times \mathbb{P}\left(y^B > t\right) \geq 2\varepsilon(1-\delta).$$

Choosing $\delta = \frac{1}{2}$ and $t = \bar{t}$,

$$\mathbb{E}\left[\inf_{y \in G_t} \|\nabla f(y)\|^2\right] > \varepsilon.$$

By construction, notice that f is random. Nevertheless,

$$\mathbb{E}\left[\mathbb{E}\left[\inf_{y \in G_t} \|\nabla f(y)\|^2 \middle| f\right]\right] > \varepsilon.$$

Thus, there exists a deterministic $\bar{f}$ such that

$$\mathbb{E}\left[\inf_{y \in G_t(\bar{f})} \left\|\nabla \bar{f}(y)\right\|^2\right] = \mathbb{E}\left[\inf_{y \in G_t} \|\nabla f(y)\|^2 \middle| f = \bar{f}\right] > \varepsilon,$$

where $G_t(\bar{f})$ are outputs of the algorithm given $\bar{f}$ (in the statement of the theorem, we rename $\bar{f}$ to f).

It is left to find the asymptotic of $t = \bar{t}$ using (41):

$$\bar{t} \geq \frac{1}{s}(B-K) \overset{(33)}{=} \frac{1}{2^{10}} \min\left\{\min_{S \in \bar{V}} B_h(1/p_\sigma, S), \frac{d}{K\bar{w}_{\bar{k}}}\right\}\left(\left\lfloor \frac{T}{K} \right\rfloor - K\right). \tag{42}$$

Consider (21):

$$T = \left\lfloor \frac{L\Delta}{2\Delta^0(K,a) \cdot \ell_1(K,a) \cdot \varepsilon} \right\rfloor.$$

Using the definitions of $\Delta^0(K,a)$ and $\ell_1(K,a)$,

$$T = \left\lfloor \frac{L\Delta \log a}{48\pi e^3 K^2 a^{2K} \varepsilon} \right\rfloor.$$

Choosing $a = 1 + \frac{1}{K}$,

$$T \geq \left\lfloor \frac{L\Delta}{96\pi e^5 K^3 \varepsilon} \right\rfloor$$

since $\log\left(1 + \frac{1}{K}\right) \geq \frac{1}{2K}$ and $\left(1 + \frac{1}{K}\right)^{2K} \leq e^2$ for all $K \geq 1$. Recall the definition of K in (40). Since in the theorem, we assume that $\frac{L\Delta}{\varepsilon} \geq \bar{c}_1 \log^{10}(n+1)$ for some universal constant $\bar{c}_1$,

$$T \geq \frac{L\Delta}{2 \cdot 96\pi e^5 K^3 \varepsilon},$$

$$\left\lfloor \frac{T}{K} \right\rfloor \geq \frac{L\Delta}{4 \cdot 96\pi e^5 K^4 \varepsilon},$$

and

$$\left\lfloor \frac{T}{K} \right\rfloor - K \geq \frac{L\Delta}{4 \cdot 96\pi e^5 K^4 \varepsilon} - K \geq \frac{L\Delta}{8 \cdot 96\pi e^5 K^4 \varepsilon}.$$

Substituting to (42),

$$\bar{t} \geq \frac{1}{s}(B-K) \geq \frac{1}{2^{20}\pi e^5 K^4} \times \min\left\{\min_{S \in \bar{V}} B_h\left(\frac{1}{p_\sigma}, S\right), \frac{d}{K\bar{w}_{\bar{k}}}\right\} \frac{L\Delta}{\varepsilon}. \tag{43}$$

Using the definition of $\gamma_\infty(K,a)$,

$$\frac{\sigma^2}{2\varepsilon\gamma_\infty^2(K,a)} = \frac{\sigma^2 \log a}{144\pi e^3 K^2 a^{2K}\varepsilon} \geq \frac{\sigma^2}{288\pi e^5 K^3 \varepsilon}$$

since $\log\left(1+\frac{1}{K}\right) \geq \frac{1}{2K}$ and $\left(1+\frac{1}{K}\right)^{2K} \leq e^2$ for all $K \geq 1$. Due to (25),

$$\begin{aligned}
B_h(1/p_\sigma, S) &= B_h\left(\max\left\{\frac{\sigma^2}{2\varepsilon\gamma_\infty^2(K,a)}, 1\right\}, S\right) \\
&= \min_{m\in[|S|]}\left[\left(\frac{1}{m}\sum_{i=1}^{m}\frac{1}{h_{\pi_i(S)}}\right)^{-1}\left(1+\max\left\{\frac{\sigma^2}{2\varepsilon\gamma_\infty^2(K,a)}, 1\right\}\times\frac{1}{m}\right)\right] \\
&\geq \frac{1}{288\pi e^5 K^3} \times \min_{m\in[|S|]}\left[\left(\frac{1}{m}\sum_{i=1}^{m}\frac{1}{h_{\pi_i(S)}}\right)^{-1}\left(1+\frac{\sigma^2}{\varepsilon m}\right)\right] = \frac{1}{288\pi e^5 K^3} \times B_h\left(\sigma^2/\varepsilon, S\right).
\end{aligned}$$

Substituting to (43),

$$\begin{aligned}
\bar{t} &\geq \frac{1}{2^{20}\pi e^5 K^4} \times \min\left\{\frac{1}{288\pi e^5 K^3} \times \min_{S\in\bar{V}} B_h\left(\sigma^2/\varepsilon, S\right), \frac{d}{K\bar{w}_{\bar{k}}}\right\}\frac{L\Delta}{\varepsilon} \\
&\geq \frac{1}{2^{29}\pi^2 e^{10} K^7} \times \min\left\{\min_{S\in\bar{V}} B_h\left(\sigma^2/\varepsilon, S\right), \frac{d}{\bar{w}_{\bar{k}}}\right\}\frac{L\Delta}{\varepsilon} \\
&\geq \frac{1}{2^{43}3^7 5^{14}\pi^2 e^{10}\cdot\log^{14}(n+1)} \times \min\left\{\min_{S\in\bar{V}} B_h\left(\sigma^2/\varepsilon, S\right), \frac{d}{\bar{w}_{\bar{k}}}\right\}\frac{L\Delta}{\varepsilon},
\end{aligned}$$

which matches (27) because $\bar{V} := (S_{\bar{k}+1,1}, \ldots, S_{\bar{k}+1,\bar{k}+1})$.

□

G.3 Auxiliary lemmas

Lemma G.8 (Proof in Section G.3). *For all $i \in [n]$,*

$$\mathbb{P}\left(\eta_{k,i,j} \leq t \middle| \{\eta_{k,i,j'}\}_{i\in[n],1\leq j'<j}, \mathcal{G}_{k-1}\right) \leq \lfloor t \rfloor p_\sigma$$

for all $t \geq 0$, $k \geq 1$ and $j \geq 1$. Moreover, consider any $\bar{S} \in \bar{V}$ and $e \in \bar{E}(\bar{S})$, then

$$\mathbb{P}\left(\mu_{k,\bar{S},e,m} \leq t \middle| \mu_{k,\bar{S},e,m-1}, \ldots, \mu_{k,\bar{S},e,1}, \mathcal{G}_{k-1}\right) \leq \frac{Kt}{\max\{\frac{d}{2} - \sum_{p=1}^{m-1}\mu_{k,\bar{S},e,p}, 0\}}$$

for all $t \geq 0$, $k \geq 1$, and $m \geq 1$.

Proof. Since the oracle draws $\{\xi\}$ are i.i.d in (22) and the generated $\{\xi\}$ are independent of $\{\mu_{k',S,e,p}\}_{p\geq 1,1\leq k'\leq k,S\in\bar{V},e\in\bar{E}(S)}$ and $\{\nu_{k,S,e,j}\}_{k\geq 1,j\geq 1,S\in\bar{V},e\in\bar{E}(S)}$, we can conclude that

$$\begin{aligned}
&\mathbb{P}\left(\eta_{k,i,j} \leq t \middle| \{\eta_{k,i,j'}\}_{i\in[n],1\leq j'<j}, \mathcal{G}_{k-1}\right) \\
&= \sum_{p=1}^{\lfloor t \rfloor}\mathbb{P}\left(\xi_{k,i,j,p}=1, \xi_{k,i,j,p-1}=0, \ldots, \xi_{k,i,j,1}=0\right) \leq \sum_{p=1}^{\lfloor t \rfloor}\mathbb{P}\left(\xi_{k,i,j,p}=1\right) \leq \lfloor t \rfloor p_\sigma
\end{aligned} \tag{44}$$

for all $k \geq 1$, $i \in [n]$, and $j \geq 1$, where $\{\xi_{k,i,j,p}\}_{k,i,j,p\geq 1}$ are i.i.d. Bernoulli random variables.

Let us fix any $\bar{S} \in \bar{V}$ and $e \in \bar{E}(\bar{S})$. Recall that R_T is a uniformly random subset without repetitions; thus,

$$\begin{aligned}
\mathbb{P}\left(\mu_{k,\bar{S},e,1} = j \middle| \mathcal{G}_{k-1}\right) &= \mathbb{P}\left(\nu_{k,\bar{S},e,j} \in B_k, \nu_{k,\bar{S},e,j-1} \notin B_k, \ldots, \nu_{k,\bar{S},e,1} \notin B_k \middle| \mathcal{G}_{k-1}\right) \\
&\leq \mathbb{P}\left(\nu_{k,\bar{S},e,j} \in B_k \middle| \mathcal{G}_{k-1}\right) = \mathbb{E}\left[\mathbf{1}[\nu_{k,\bar{S},e,j} \in B_k] \middle| \mathcal{G}_{k-1}\right].
\end{aligned}$$

Let us define $\mathcal{B}_{k-1}$ as the sigma-algebra generated by $\{\eta_{k',i,j}\}_{1\le k'\le k-1, i\in[n], j\ge 1}$, $\{B_{k'}\}_{1\le k'\le k-1}$, and $\{\nu_{k,\bar{S},e,j}\}_{j\ge 1}$. Notice that $\mathcal{G}_{k-1}\subseteq\mathcal{B}_{k-1}$ since $\{\mu_{k',S,e,p}\}_{p\ge 1, 1\le k'\le k-1, S\in\bar{V}, e\in\bar{E}(S)}$ are deterministic, knowing $\mathcal{B}_{k-1}$. Thus,

$$\begin{aligned}\mathbb{P}\left(\mu_{k,\bar{S},e,1}=j\middle|\mathcal{G}_{k-1}\right) &= \mathbb{E}\left[\mathbb{E}\left[\mathbf{1}[\nu_{k,\bar{S},e,j}\in B_k]\middle|\mathcal{B}_{k-1}\right]\middle|\mathcal{G}_{k-1}\right]\\ &= \mathbb{E}\left[\mathbb{E}\left[\mathbf{1}[\nu_{k,\bar{S},e,j}\in B_k]\middle|\{B_{k'}\}_{1\le k'\le k-1},\{\nu_{k,\bar{S},e,j}\}_{j\ge 1}\right]\middle|\mathcal{G}_{k-1}\right]\\ &= \mathbb{E}\left[\mathbb{P}\left(\nu_{k,\bar{S},e,j}\in B_k\middle|\{B_{k'}\}_{1\le k'\le k-1},\{\nu_{k,\bar{S},e,j}\}_{j\ge 1}\right)\middle|\mathcal{G}_{k-1}\right]\\ &\le \frac{K}{d-(k-1)K}.\end{aligned}$$

The last inequality follows from the fact that $\{B_k\}_{k\in[B]}$ is independent of $\{\nu_{k,\bar{S},e,j}\}$, and from evaluating the probability that a deterministic value belongs to a random subset B_k, given $\{B_{k'}\}_{1\le k'\le k-1}$. The fact that $\{B_k\}_{k\in[B]}$ is independent of $\{\nu_{k,\bar{S},e,j}\}$ follows from Assumption G.1.

Since $T\le\frac{d}{2}$,

$$\mathbb{P}\left(\mu_{k,\bar{S},e,1}=j\middle|\mathcal{G}_{k-1}\right)\le\frac{K}{d-T}\le\frac{K}{d/2}.$$

Let us define $u_m:=\sum_{p=1}^{m}\mu_{k,\bar{S},e,p}$. Similarly,

$$\begin{aligned}&\mathbb{P}\left(\mu_{k,\bar{S},e,m}=j\middle|\mu_{k,\bar{S},e,m-1},\dots,\mu_{k,\bar{S},e,1},\mathcal{G}_{k-1}\right)\\ &=\mathbb{P}\Big(\nu_{k,\bar{S},e,u_{m-1}+j}\in B_k,\cap_{p=1}^{j-1}\{\nu_{k,\bar{S},e,u_{m-1}+p}\notin B_k\},\cap_{p=1}^{m-1}\{\nu_{k,\bar{S},e,u_{m-1}+j}\neq\nu_{k,\bar{S},e,u_p}\}\Big|\\ &\qquad\nu_{k,\bar{S},e,u_{m-1}}\in B_k,\dots,\nu_{k,\bar{S},e,u_{m-2}+1}\notin B_k,\nu_{k,\bar{S},e,u_{m-2}}\in B_k,\dots,\nu_{k,\bar{S},e,1}\notin B_k,\\ &\qquad\cap_{p\neq p'=1}^{m-1}\{\nu_{k,\bar{S},e,u_{p'}}\neq\nu_{k,\bar{S},e,u_p}\},\{u_p\}_{p=1}^{m-1},\mathcal{G}_{k-1}\Big)\\ &\le\mathbb{P}\Big(\nu_{k,\bar{S},e,u_{m-1}+j}\in B_k\Big|\cap_{p=1}^{m-1}\{\nu_{k,\bar{S},e,u_{m-1}+j}\neq\nu_{k,\bar{S},e,u_p}\},\\ &\qquad\nu_{k,\bar{S},e,u_{m-1}}\in B_k,\dots,\nu_{k,\bar{S},e,u_{m-2}+1}\notin B_k,\nu_{k,\bar{S},e,u_{m-2}}\in B_k,\dots,\nu_{k,\bar{S},e,1}\notin B_k,\\ &\qquad\cap_{p\neq p'=1}^{m-1}\{\nu_{k,\bar{S},e,u_{p'}}\neq\nu_{k,\bar{S},e,u_p}\},\{u_p\}_{p=1}^{m-1},\mathcal{G}_{k-1}\Big)\\ &\le\mathbb{P}\Big(\nu_{k,\bar{S},e,u_{m-1}+j}\in B_k\Big|\cap_{p=1}^{u_{m-1}}\{\nu_{k,\bar{S},e,u_{m-1}+j}\neq\nu_{k,\bar{S},e,p}\},\\ &\qquad\nu_{k,\bar{S},e,u_{m-1}}\in B_k,\dots,\nu_{k,\bar{S},e,u_{m-2}+1}\notin B_k,\nu_{k,\bar{S},e,u_{m-2}}\in B_k,\dots,\nu_{k,\bar{S},e,1}\notin B_k,\\ &\qquad\cap_{p\neq p'=1}^{m-1}\{\nu_{k,\bar{S},e,u_{p'}}\neq\nu_{k,\bar{S},e,u_p}\},\{u_p\}_{p=1}^{m-1},\mathcal{G}_{k-1}\Big)\end{aligned}$$

for all $j\ge 1$, where use the standard properties of probability and the last inequality comes from the fact that for all $p\ge 1$, if $\nu_{k,\bar{S},e,p}\notin B_k$ and $\nu_{k,\bar{S},e,p}=\nu_{k,\bar{S},e,u_{m-1}+j}$, then the probability is zero. Now, the last condition

$$\begin{aligned}&\nu_{k,\bar{S},e,u_{m-1}}\in B_k,\dots,\nu_{k,\bar{S},e,u_{m-2}+1}\notin B_k,\nu_{k,\bar{S},e,u_{m-2}}\in B_k,\dots,\nu_{k,\bar{S},e,1}\notin B_k,\\ &\cap_{p=1}^{u_{m-1}}\{\nu_{k,\bar{S},e,u_{m-1}+j}\neq\nu_{k,\bar{S},e,p}\},\cap_{p\neq p'=1}^{m-1}\{\nu_{k,\bar{S},e,u_{p'}}\neq\nu_{k,\bar{S},e,u_p}\},\{u_p\}_{p=1}^{m-1},\mathcal{G}_{k-1}\end{aligned}$$

says us that B_k includes $\nu_{k,\bar{S},e,u_{m-1}}\neq\ldots\neq\nu_{k,\bar{S},e,u_1}$ (pairwise distinct) and does not include $\nu_{k,\bar{S},e,u_{m-1}-1},\dots,\nu_{k,\bar{S},e,u_{m-2}+1},\nu_{k,\bar{S},e,u_{m-2}-1},\dots,\nu_{k,\bar{S},e,1}$ such that none of them equals to $\nu_{k,\bar{S},e,u_{m-1}+j}$, and $\mathcal{G}_{k-1}$ also might "reveal" the values of $\{B_{k'}\}_{1\le k'\le k-1}$. Conditioned on this information, the probability that $\nu_{k,\bar{S},e,u_{m-1}+j}\in B_k$ is less or equal to $\frac{K-(m-1)}{\max\{d-(k-1)K-u_{m-1},0\}}\le\frac{K}{\max\{d-T-u_{m-1},0\}}\le\frac{K}{\max\{d/2-u_{m-1},0\}}$ since B_k is an uniformly random subset (there are still $K-(m-1)$ coordinates in B_k with "unknown positions" that can uniformly placed in at least $\max\{d-(k-1)K-u_{m-1},0\}$ positions). Therefore,

$$\mathbb{P}\left(\mu_{k,\bar{S},e,m}=j\middle|\mu_{k,\bar{S},e,m-1},\dots,\mu_{k,\bar{S},e,1},\mathcal{G}_{k-1}\right)\le\frac{K}{\max\{\frac{d}{2}-\sum_{p=1}^{m-1}\mu_{k,\bar{S},e,p},0\}}$$

[7] Pedantically, $\mathbb{P}(A|B)$ is not defined if $\mathbb{P}(B)=0$. When we use the inequality $\mathbb{P}(A)\le\mathbb{P}(A|B)$, we take into account the standard convention that $\mathbb{P}(A|B)=0$ whenever $\mathbb{P}(B)=0$.

and

$$\mathbb{P}\left(\mu_{k,\bar{S},e,m} \leq t \middle| \mu_{k,\bar{S},e,m-1}, \ldots, \mu_{k,\bar{S},e,1}, \mathcal{G}_{k-1}\right) \leq \frac{Kt}{\max\{\frac{d}{2} - \sum_{p=1}^{m-1} \mu_{k,\bar{S},e,p}, 0\}}. \tag{45}$$

□

Lemma G.9. *For all $S \in \bar{V}$, $k \geq 1$,$K \geq 1$, and $s \geq \frac{128}{B_h(1/p_\sigma, S)}$,*

$$\mathbb{E}\left[\exp\left(-s\sum_{j=1}^{K} \min_{i\in S} h_i \eta_{k,i,j}\right) \middle| \mathcal{G}_{k-1}\right] \leq \frac{1}{8^K}. \tag{28}$$

Proof. For all $t > 0$ and $j \geq 1$,

$$\begin{aligned}
&\mathbb{E}\left[\exp\left(-s\min_{i\in S} h_i\eta_{k,i,j}\right) \middle| \{\eta_{k,i,j'}\}_{1\leq j'\leq j-1}, \mathcal{G}_{k-1}\right] \\
&\leq e^{-st} + \mathbb{P}\left(\min_{i\in S} h_i\eta_{k,i,j} \leq t \middle| \{\eta_{k,i,j'}\}_{1\leq j'\leq j-1}, \mathcal{G}_{k-1}\right) \\
&\leq e^{-st} + \sum_{i\in S} \mathbb{P}\left(h_i\eta_{k,i,j} \leq t | \{\eta_{k,i,j'}\}_{1\leq j'\leq j-1}, \mathcal{G}_{k-1}\right) \\
&\overset{(44)}{\leq} e^{-st} + \sum_{i\in S} p_\sigma \left\lfloor \frac{t}{h_i} \right\rfloor.
\end{aligned}$$

Taking $t = \frac{1}{32} \times B_h(1/p_\sigma, S)$, we can use Lemma G.11 and the bound on s to get

$$\mathbb{E}\left[\exp\left(-s\min_{i\in S} h_i\eta_{k,i,j}\right) \middle| \{\eta_{k,i,j'}\}_{1\leq j'\leq j-1}, \mathcal{G}_{k-1}\right] \leq \frac{1}{8} \tag{46}$$

for all $j \geq 1, k \geq 1, S \in \bar{V}$. Using the tower rule and (46),

$$\begin{aligned}
&\mathbb{E}\left[\exp\left(-s\sum_{j=1}^{K} \min_{i\in S} h_i\eta_{k,i,j}\right) \middle| \mathcal{G}_{k-1}\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[\exp\left(-s\min_{i\in S} h_i\eta_{k,i,K}\right) \middle| \{\eta_{k,i,j'}\}_{1\leq j'\leq K-1}, \mathcal{G}_{k-1}\right] \exp\left(-s\sum_{j=1}^{K-1} \min_{i\in S} h_i\eta_{k,i,j}\right) \middle| \mathcal{G}_{k-1}\right] \\
&\leq \frac{1}{8}\mathbb{E}\left[\exp\left(-s\sum_{j=1}^{K-1} \min_{i\in S} h_i\eta_{k,i,j}\right) \middle| \mathcal{G}_{k-1}\right].
\end{aligned}$$

Then, repeating these derivations $K-1$ times, we obtain (28). □

Lemma G.10. *For all $S \in \bar{V}$, $e \in \bar{E}(S)$, $k \geq 1$,$K \geq 1$, $w > 0$, and $s \geq \frac{1024Kw}{d}$,*

$$\mathbb{E}\left[\exp\left(-s\sum_{j=1}^{K} \frac{\mu_{k,S,e,j}}{w}\right) \middle| \mathcal{G}_{k-1}\right] \leq \frac{1}{8^K}. \tag{29}$$

Proof. Using the tower rule,

$$\begin{aligned}
I_2^K &:= \mathbb{E}\left[\exp\left(-s\sum_{j=1}^{K} \frac{\mu_{k,S,e,j}}{w}\right) \middle| \mathcal{G}_{k-1}\right] \\
&= \mathbb{E}\left[\underbrace{\mathbb{E}\left[\exp\left(-s\frac{\mu_{k,S,e,K}}{w}\right) \middle| \{\mu_{k,S,e,j}\}_{1\leq j\leq K-1}, \mathcal{G}_{k-1}\right] \exp\left(-s\sum_{j=1}^{K-1} \frac{\mu_{k,S,e,j}}{w}\right)}_{K_2:=} \middle| \mathcal{G}_{k-1}\right].
\end{aligned}$$

If $\sum_{j=1}^{K-1} \mu_{k,S,e,j} \geq \frac{d}{4}$, then

$$K_2 \leq \exp\left(-\frac{sd}{4w}\right).$$

Otherwise, if $\sum_{j=1}^{K-1} \mu_{k,S,e,j} < \frac{d}{4}$, then

$$\begin{aligned} K_2 &\leq \mathbb{E}\left[\exp\left(-s\frac{\mu_{k,S,e,K}}{w}\right)\middle|\, \{\mu_{k,S,e,j}\}_{1\leq j\leq K-1}, \mathcal{G}_{k-1}\right] \exp\left(-s\sum_{j=1}^{K-1}\frac{\mu_{k,S,e,j}}{w}\right) \\ &\leq \left(\exp(-st) + \mathbb{P}\left(\frac{\mu_{k,S,e,K}}{w} \leq t \middle| \{\mu_{k,S,e,j}\}_{1\leq j\leq K-1}, \mathcal{G}_{k-1}\right)\right) \exp\left(-s\sum_{j=1}^{K-1}\frac{\mu_{k,S,e,j}}{w}\right) \end{aligned}$$

for all $t > 0$. Using (45), if $\sum_{j=1}^{K-1} \mu_{k,S,e,j} < \frac{d}{4}$, then

$$\begin{aligned} K_2 &\leq \left(\exp(-st) + \frac{Ktw}{\max\{\frac{d}{2} - \sum_{p=1}^{K-1}\mu_{k,S,e,p}, 0\}}\right) \exp\left(-s\sum_{j=1}^{K-1}\frac{\mu_{k,S,e,j}}{w}\right) \\ &\leq \left(\exp(-st) + \frac{4Ktw}{d}\right) \exp\left(-s\sum_{j=1}^{K-1}\frac{\mu_{k,S,e,j}}{w}\right). \end{aligned}$$

Taking $t = \frac{d}{128Kw}$ and using $s \geq \frac{\log(32)}{t}$,

$$K_2 \leq \left(\frac{1}{32} + \frac{1}{32}\right) \exp\left(-s\sum_{j=1}^{K-1}\frac{\mu_{k,S,e,j}}{w}\right) = \frac{1}{16}\exp\left(-s\sum_{j=1}^{K-1}\frac{\mu_{k,S,e,j}}{w}\right).$$

Combining both cases,

$$\begin{aligned} I_2^K &\leq \mathbb{E}\left[\max\left\{\frac{1}{16}\exp\left(-s\sum_{j=1}^{K-1}\frac{\mu_{k,S,e,j}}{w}\right), \exp\left(-\frac{sd}{4w}\right)\right\}\middle|\, \mathcal{G}_{k-1}\right] \\ &\leq \frac{1}{16}\underbrace{\mathbb{E}\left[\exp\left(-s\sum_{j=1}^{K-1}\frac{\mu_{k,S,e,j}}{w}\right)\middle|\, \mathcal{G}_{k-1}\right]}_{I_2^{K-1}} + \exp\left(-\frac{sd}{4w}\right). \end{aligned}$$

Repeating the same steps $K-1$ more times,

$$I_2^K \leq \frac{1}{16^K} + 2\exp\left(-\frac{sd}{4w}\right) \leq \frac{1}{8^K},$$

where we use that $s \geq \frac{256Kw}{d}$ and $\sum_{j=0}^{\infty}\frac{1}{16^j} \leq 2$. □

Lemma G.11. *For all $h_1, \ldots, h_n \geq 0$, $p_\sigma \in (0,1]$, and $S \subseteq [n]$, we have*

$$p_\sigma \sum_{i\in S}\left\lfloor \frac{t}{h_i}\right\rfloor \leq \frac{1}{16}$$

for

$$t = \frac{1}{32} \times B_h(1/p_\sigma, S),$$

where

$$B_h(1/p_\sigma, S) := \min_{m\in[|S|]}\left[\left(\frac{1}{m}\sum_{i=1}^{m}\frac{1}{h_{\pi_i(S)}}\right)^{-1}\left(1+\frac{1}{p_\sigma m}\right)\right],$$

where $\pi(S)$ is a permutation that sorts $\{h_i\}_{i\in S}$: $h_{\pi_1(S)} \leq \cdots \leq h_{\pi_{|S|}(S)}$.

Proof. If $\lfloor t/h_i \rfloor = 0$ for all $i \in S$, then the lemma is true. Otherwise, let

$$m := \max\Big\{ r \in [|S|] : \ h_{\pi_r(S)} \le t \Big\}. \tag{47}$$

Then $\lfloor t/h_{\pi_i(S)} \rfloor \ge 1$ for $i \le m$ and $\lfloor t/h_{\pi_i(S)} \rfloor = 0$ for $i > m$. Hence,

$$\sum_{i \in S} \left\lfloor \frac{t}{h_i} \right\rfloor = \sum_{i=1}^{m} \left\lfloor \frac{t}{h_{\pi_i(S)}} \right\rfloor \le \sum_{i=1}^{m} \frac{t}{h_{\pi_i(S)}} = t \sum_{i=1}^{m} \frac{1}{h_{\pi_i(S)}}.$$

Let $A_m := \frac{1}{m} \sum_{i=1}^{m} \frac{1}{h_{\pi_i(S)}}$. By the definition of $B_h(1/p_\sigma, S)$ and $t = \frac{1}{32} B_h(1/p_\sigma, S)$,

$$t \le \frac{1}{32} A_m^{-1} \left(1 + \frac{1}{p_\sigma m} \right).$$

Therefore,

$$p_\sigma \sum_{i \in S} \left\lfloor \frac{t}{h_i} \right\rfloor \le p_\sigma t \sum_{i=1}^{m} \frac{1}{h_{\pi_i(S)}} = p_\sigma t m A_m \le \frac{1}{32} p_\sigma \Big(A_m^{-1} \Big(1 + \frac{1}{p_\sigma m} \Big) \Big) m A_m = \frac{1}{32} \big(p_\sigma m + 1 \big).$$

We now show that $p_\sigma m < 1$. If $p_\sigma m \ge 1$, then $1 + \frac{1}{p_\sigma m} \le 2$. Moreover, $A_m \ge \frac{1}{h_{\pi_m(S)}}$, hence $A_m^{-1} \le h_{\pi_m}$. Consequently,

$$t \le \frac{1}{32} A_m^{-1} \times 2 \le \frac{1}{16} h_{\pi_m(S)} < h_{\pi_m(S)},$$

contradicting $h_{\pi_m(S)} \le t$ in (47). Thus $p_\sigma m < 1$, and

$$p_\sigma \sum_{i \in S} \left\lfloor \frac{t}{h_i} \right\rfloor \le \frac{1}{32} (1 + p_\sigma m) < \frac{1}{16}.$$

□

G.4 Proof of Lemma G.7

Lemma G.7 (Proof in Section G.4)**.** *The number of steps $\bar{d}$ in the leaf-branch peeling procedure satisfies*

$$\bar{d} \le \lfloor \log_2(n+2) \rfloor.$$

Proof. For any step $p \in [\bar{d}]$, consider

$$\bar{T}_p := \bar{T} \setminus \bigcup_{i<p} \left(\bar{\mathcal{L}}_i \cup \bar{\mathcal{B}}_i \right), \qquad \text{so that } \bar{T}_1 \equiv \bar{T}.$$

Let $\widetilde{T}_p$ be the tree obtained from $\bar{T}_p$ by suppressing all degree-2 nodes (recursively removing any degree-2 node and merging its two incident edges into one). Denote $\widetilde{m}_p := |V(\widetilde{T}_p)|$.

In $\widetilde{T}_p$, every node has degree 1 or at least 3. Let $\widetilde{\mathcal{L}}_p \equiv \bar{\mathcal{L}}_p$ be the set of leaves of $\widetilde{T}_p$ and $\widetilde{\mathcal{I}}_p := V(\widetilde{T}_p) \setminus \widetilde{\mathcal{L}}_p$. Let $\deg_{\widetilde{T}_p}(v)$ be the number of edges of node v in tree $\widetilde{T}_p$. Using the degree-sum identity for trees,

$$\sum_{v \in V(\widetilde{T}_p)} \deg_{\widetilde{T}_p}(v) = 2(\widetilde{m}_p - 1) \ \ge \ |\widetilde{\mathcal{L}}_p| + 3|\widetilde{\mathcal{I}}_p|,$$

which implies $|\widetilde{\mathcal{L}}_p| \ge |\widetilde{\mathcal{I}}_p| + 2$ due to $\widetilde{m}_p = |\widetilde{\mathcal{L}}_p| + |\widetilde{\mathcal{I}}_p|$ and hence

$$|\widetilde{\mathcal{I}}_p| \le \frac{\widetilde{m}_p - 2}{2}.$$

By construction of the leaf-branch peeling step, removing $\bar{\mathcal{L}}_p \cup \bar{\mathcal{B}}_p$ from $\bar{T}_p$ yields $\bar{T}_{p+1}$. Note that $\widetilde{\mathcal{I}}_p$ is a set of nodes belonging to $\bar{T}_{p+1}$ and not suppressed by the degree-2 suppressing procedure in $\bar{T}_p$ (all nodes have degree ≥ 3). Consider the set of degree-2 nodes $\widetilde{\mathcal{K}} = V(\bar{T}_{p+1}) \setminus \widetilde{\mathcal{I}}_p$ such that

$\deg_{\bar{T}_p}(v) = 2$ for all $v \in \widetilde{\mathcal{K}}$. For any $v \in \widetilde{\mathcal{K}}$, the degree of v does not change when we construct $\bar{T}_{p+1}$, because if it changes, then v would belong to $\bar{\mathcal{B}}_p$ (contradiction). Thus, $\deg_{\bar{T}_{p+1}}(v) = 2$ for all $v \in \widetilde{\mathcal{K}}$. It means that when construct $\bar{T}_{p+1}$, vertices $\widetilde{\mathcal{K}}$ still have degree 2. Thus, the degree-2 suppressing procedure on $\bar{T}_{p+1}$ will remove at least $\widetilde{\mathcal{K}}$ and we get

$$\widetilde{m}_{p+1} \leq |\widetilde{\mathcal{I}}_p|.$$

Therefore,

$$\widetilde{m}_{p+1} \leq |\widetilde{\mathcal{I}}_p| \leq \frac{\widetilde{m}_p - 2}{2}.$$

Unrolling the recursion,

$$\widetilde{m}_p + 2 \leq \frac{\widetilde{m}_1 + 2}{2^{p-1}} \leq \frac{n+2}{2^{p-1}}$$

and

$$2^{\bar{d}} \leq n + 2.$$

since $m_{\bar{d}} \geq 0$. Thus

$$\bar{d} \leq \lfloor \log_2(n+2) \rfloor.$$

□

G.5 Proof of Theorem G.12

Theorem G.12 (Lower Bound in the Heterogeneous Setting)**.** *We consider any computational and communication environment satisfying Assumption 1.4. Let $L, \Delta, \varepsilon, n, \sigma^2, d > 0$ be any numbers such that $L\Delta/\varepsilon \geq \bar{c}_1$ and dimension $d \geq \bar{c}_3 L\Delta/\varepsilon$. Consider Protocol 5. For all $i \in [n]$ and $k \geq 0$, mappings $\mathcal{C}_i^k$ satisfy Assumption G.1. Then, for any algorithm A, there exists a function $f \in \mathcal{F}_{\Delta,L}$, exists oracles that satisfy Assumption 5.1, and $\mathbb{E}\left[\inf_{y \in G_t} \|\nabla f(y)\|^2\right] > \varepsilon$ for all*

$$t \leq \bar{c}_2 \times \max\left\{\frac{d}{\bar{w}_1}, \max_{i\in[n]} h_i, \frac{\sigma^2}{n\varepsilon}\left(\frac{1}{n}\sum_{i=1}^n h_i\right)\right\}\frac{L\Delta}{\varepsilon}, \tag{48}$$

where G_t is the set of all possible points that can be constructed by A up to time t based on $\{I_i\}_{i\in[n]}$, and $\bar{w}_1 \equiv \min_{\{i,j\}\in F} w_{ij}$ is the smallest weight in a Gomory-Hu tree $T = (V, F, w)$ of $\bar{G}$. The quantities $\bar{c}_1$, $\bar{c}_2$, and $\bar{c}_3$ are universal constants.

Proof. The lower bound

$$\Omega\left(\max\left\{\max_{i\in[n]} h_i, \frac{\sigma^2}{n\varepsilon}\left(\frac{1}{n}\sum_{i=1}^n h_i\right)\right\}\frac{L\Delta}{\varepsilon}\right) \tag{49}$$

follows from Theorem A.2 by Tyurin & Richtárik (2023), where the authors do not take into account communication times. With communication times, the lower bound can only increase. We now prove the first term in the max .

Unlike the homogeneous setup, it is sufficient to use the deterministic construction from (Carmon et al., 2020) and apply the standard trick of placing the blocks of this function on different nodes. For any $T \in \mathbb{N}$, Carmon et al. (2020) define $F_T : \mathbb{R}^T \to \mathbb{R}$ such that

$$F_T(x) := -\Psi(1)\Phi(x_1) + \sum_{i=2}^T \left[\Psi(-x_{i-1})\Phi(-x_i) - \Psi(x_{i-1})\Phi(x_i)\right], \tag{50}$$

where

$$\Psi(x) = \begin{cases} 0, & x \leq 1/2, \\ \exp\left(1 - \frac{1}{(2x-1)^2}\right), & x \geq 1/2, \end{cases} \quad \text{and} \quad \Phi(x) = \sqrt{e}\int_{-\infty}^x e^{-\frac{1}{2}t^2} dt.$$

Lemma G.13 (Carmon et al. (2020)). *The function F_T satisfies:*

1. $F_T(0) - \inf_{x \in \mathbb{R}^T} F_T(x) \leq \Delta^0 T$, *where* $\Delta^0 := 12$.
2. *The function F_T is l_1–smooth, where* $l_1 := 152$.
3. *For all* $x \in \mathbb{R}^T$, $\text{prog}(\nabla F_T(x)) \leq \text{prog}(x) + 1$.
4. *For all* $x \in \mathbb{R}^T$, *if* $\text{prog}(x) < T$, *then* $\|\nabla F_T(x)\| > 1$.

Find undirected version of $G : \bar{G} = (V, \bar{E}, b)$ where $\{i, j\} \in \bar{E}$ with weight b_{ij} iff $(i, j) \in E$ with weight b_{ij}. For the graph $\bar{G}$, we construct a Gomory-Hu tree $T = (V, F, w)$. Let us take any pair $\{\bar{i}, \bar{j}\} \in F$ of workers such that $\min_{\{i,j\} \in F} w_{ij} = w_{\bar{i}\bar{j}}$. We now split the blocks of the function $F_T(x)$ from (50) and define two new functions.

First, sample a uniformly random subset $R_T = [R_{T,1}, \ldots, R_{T,T}]$ without repetitions from set $[d]$ ($R_{T,1} < \cdots < R_{T,T}$). Let us fix any $\lambda > 0$, then we take $F_{T,1} : \mathbb{R}^d \to \mathbb{R}$ and $F_{T,2} : \mathbb{R}^d \to \mathbb{R}$ such that

$$F_{T,1}(x) := -\Psi(1)\Phi([x]_{R_{T,1}}) + \sum_{i \in \{2,\ldots,T\}, i \,|\, 2=1} \left[\Psi(-[x]_{R_{T,i-1}})\Phi(-[x]_{R_{T,i}}) - \Psi([x]_{R_{T,i-1}})\Phi([x]_{R_{T,i}})\right], \tag{51}$$

and

$$F_{T,2}(x) := \sum_{i \in \{2,\ldots,T\}, i \,|\, 2=0} \left[\Psi(-[x]_{R_{T,i-1}})\Phi(-[x]_{R_{T,i}}) - \Psi([x]_{R_{T,i-1}})\Phi([x]_{R_{T,i}})\right],$$

where $[x]_j$ is the j^{th} coordinate of x. The idea is that we apply a random permutation of the coordinates. Notice that the $d - T$ coordinates are artificial. We consider the following functions f_i :

$$f_i(x) := \begin{cases} \frac{nL\lambda^2}{l_1} F_{T,1}\left(\frac{x}{\lambda}\right), & i = \bar{i}, \\ \frac{nL\lambda^2}{l_1} F_{T,2}\left(\frac{x}{\lambda}\right), & i = \bar{j}, \\ 0, & i \neq \bar{i} \text{ and } i \neq \bar{j}. \end{cases}$$

Then, we get

$$f(x) = \frac{1}{n}\sum_{i=1}^n f_i(x) = \frac{1}{n}\left(\frac{nL\lambda^2}{l_1} F_{T,1}\left(\frac{x}{\lambda}\right) + \frac{nL\lambda^2}{l_1} F_{T,2}\left(\frac{x}{\lambda}\right)\right) = \frac{L\lambda^2}{l_1} F_T\left(\frac{x_{[R_T]}}{\lambda}\right),$$

where $x_{[R_T]} \in \mathbb{R}^T$ is the subvector of size T of vector $x \in \mathbb{R}^d$ such that $[x_{[R_T]}]_i = [x]_{R_{T,i}}$ for all $i \in [T]$. The function f is L-smooth since

$$\|\nabla f(x) - \nabla f(y)\| = \frac{L\lambda}{l_1}\left\|\nabla F_T\left(\frac{x_{[R_T]}}{\lambda}\right) - \nabla F_T\left(\frac{y_{[R_T]}}{\lambda}\right)\right\| \leq L\|x - y\|.$$

Let us take

$$T = \left\lfloor \frac{\Delta l_1}{L\lambda^2 \Delta^0} \right\rfloor,$$

then

$$f(0) - \inf_{x \in \mathbb{R}^d} f(x) = \frac{L\lambda^2}{l_1}(F_T(0) - \inf_{x \in \mathbb{R}^T} F_T(x)) \leq \frac{L\lambda^2 \Delta^0 T}{l_1} \leq \Delta.$$

We take

$$\lambda = \frac{l_1\sqrt{\varepsilon}}{L}$$

to ensure that

$$\|\nabla f(x)\|^2 = \frac{L^2\lambda^2}{l_1^2}\left\|\nabla F_T\left(\frac{x_{[R_T]}}{\lambda}\right)\right\|^2 > \frac{L^2\lambda^2}{l_1^2} = \varepsilon$$

for all $x \in \mathbb{R}^T$ such that $\text{prog}(x) < T$. In the last inequality, we use Lemma G.13.

We assume that the workers have access to non-stochastic mappings $\nabla f_i(x)$ that are unbiased and 0-variance-bounded, which is sufficient to derive the lower bound.

Substituting the choice of the parameters,

$$T = \left\lfloor \frac{\Delta L}{l_1 \varepsilon \Delta^0} \right\rfloor.$$

Only workers $\bar{i}$ and $\bar{j}$ possess information about the function f. The function f is constructed as a zero-chain function, and its components are distributed between workers $\bar{i}$ and $\bar{j}$. Because of this partitioning, these two workers must communicate in order to identify the next non-zero coordinate. Initially, only worker $\bar{i}$ can obtain a non-zero value in the first "useful" coordinate $R_{T,1}$ via the gradient of $F_{T,1}$. Subsequently, however, this worker cannot obtain a non-zero value in the second "useful" coordinate $R_{T,2}$ due to the construction in (51). Thus, it has to pass the first "useful" coordinate to worker $\bar{j}$ because only this worker can discover a non-zero value in the second "useful" coordinate. Then, once the second worker discovers the second "useful" coordinate and has to pass it to the first worker, and so forth.

Let $\{\nu_{1,j}\}_{j\geq 1}$ be the sequence that the second worker receives from the first worker during the optimization process. Due to our construction, the function is random with randomly permuted coordinates. We define μ_1 as the number of received coordinates $\{\nu_{1,j}\}_{j\geq 1}$ by the second worker from the first worker until the moment when a received coordinate is $R_{T,1}$. Thus,

$$\begin{aligned}\mathbb{P}(\mu_1 = j) &= \mathbb{P}(\nu_{1,j} = R_{T,1}, \nu_{1,j-1} \neq R_{T,1}, \ldots, \nu_{1,1} \neq R_{T,1}) \leq \mathbb{P}(\nu_{1,j} = R_{T,1}) \\ &= \mathbb{E}\left[\mathbb{P}(\nu_{1,j} = R_{T,1} | \{\nu_{1,j}\}_{j\geq 1})\right].\end{aligned}$$

Due to Assumption G.1, $\{\nu_{1,j}\}_{j\geq 1}$ are independent of $R_{T,1}$. Thus,

$$\mathbb{P}(\mu_1 = j) \leq \frac{1}{d}$$

and

$$\mathbb{P}(\mu_1 \leq t) \leq \frac{t}{d}$$

for all $t \geq 0$, since $R_{T,1}$ is a uniformly random index from $[d]$, given $\{\nu_{1,j}\}_{j\geq 1}$. Let us define y^k as the first moment of time when any of the workers can discover the k^{th} coordinate $R_{T,k}$. Similarly, we define μ_2 as the number of received coordinates $\{\nu_{2,j}\}_{j\geq 1}$ by the first worker from the second worker, after time y^2, until the moment when a received coordinate is $R_{T,2}$, we define μ_3 as the number of received coordinates $\{\nu_{3,j}\}_{j\geq 1}$ by the second worker from the first worker, after time y^3, until the moment when a received coordinate is $R_{T,3}$, where $\{\nu_{k,j}\}_{j\geq 1}$ are the coordinate sent after time y^k. Thus,

$$\mathbb{P}(\mu_k = j | \{\mu_{k'}\}_{1\leq k'<k}) \leq \mathbb{P}(\nu_{k,j} = R_{T,k} | \{\mu_{k'}\}_{1\leq k'<k}).$$

Let us define $\mathcal{B}_{k-1}$ as the sigma-algebra generated by $\{R_{T,k'}\}_{1\leq k'<k}$ and $\{\nu_{k,j}\}_{j\geq 1,k\geq 1}$. Since $\{\mu_{k'}\}_{1\leq k'<k}$ are deterministic knowing $\mathcal{B}_{k-1}$. Using the tower rule,

$$\begin{aligned}&\mathbb{P}(\mu_k = j | \{\mu_{k'}\}_{1\leq k'<k}) \leq \mathbb{P}(\nu_{k,j} = R_{T,k} | \{\mu_{k'}\}_{1\leq k'<k}) \\ &= \mathbb{E}\left[\mathbb{P}(\nu_{k,j} = R_{T,k} | \{R_{T,k'}\}_{1\leq k'<k}, \{\nu_{k,j}\}_{j\geq 1,k\geq 1}) | \{\mu_{k'}\}_{1\leq k'<k}\right].\end{aligned}$$

Conditioned on $\{R_{T,k'}\}_{1\leq k'<k}$, there are at least $d-(k-1)$ positions where $R_{T,k}$ can be placed uniformly. Therefore,

$$\mathbb{P}(\mu_k = j | \{\mu_{k'}\}_{1\leq k'<k}) \leq \mathbb{P}(\nu_{k,j} = R_{T,k} | \{\mu_{k'}\}_{1\leq k'<k}) \leq \frac{1}{d-(k-1)} \leq \frac{2}{d}$$

and

$$\mathbb{P}(\mu_k \leq t | \{\mu_{k'}\}_{1\leq k'<k}) \leq \frac{2t}{d}$$

for all $t \geq 0$ and $1 \leq k \leq T-1$ since $T \leq d/2$.

By our construction, the two workers are separated by the edge with weight $\min_{\{i,j\}\in F} w_{ij}$. Hence, the maximum flow, or in terms of our problem, the maximum number of coordinates per second that nodes $\bar{i}$ and $\bar{j}$ can send to each other is bounded by $\min_{\{i,j\}\in F} w_{ij}$.

Combining all together, the time discover the last T^{th} "useful" is lower bounded by the sum

$$\sum_{j=1}^{T-1} \frac{\mu_j}{\min\limits_{\{i,j\}\in F} w_{ij}}.$$

for an algorithm to find a vector $x \in \mathbb{R}^T$ such that $\text{prog}(x) = T$. We define $w_{\min} := \min\limits_{\{i,j\}\in F} w_{ij}$.

Hence, for any $\bar{t} > 0$ and $s > 0$, by Chernoff's method,

$$\begin{aligned}\mathbb{P}\left(\sum_{j=1}^{T-1} \frac{\mu_j}{w_{\min}} \leq \bar{t}\right) &= \mathbb{P}\left(\exp\left(-s\sum_{j=1}^{T-1} \frac{\mu_j}{w_{\min}}\right) \geq e^{-s\bar{t}}\right) \\ &\leq e^{s\bar{t}}\mathbb{E}\left[\exp\left(-s\sum_{j=1}^{T-1} \frac{\mu_j}{w_{\min}}\right)\right].\end{aligned}$$

Using the tower rule and the bound $\mathbb{P}\left(\mu_k \leq t | \{\mu_{k'}\}_{k'<k}\right) \leq \frac{2t}{d}$, for any $t \geq 0$,

$$\begin{aligned}\mathbb{E}\left[\exp\left(-s\frac{\mu_k}{w_{\min}}\right)\middle|\, \{\mu_{k'}\}_{k'<k}\right] &\leq \exp(-st) + \mathbb{P}\left(\mu_k \leq w_{\min} t | \{\mu_{k'}\}_{k'<k}\right) \\ &\leq \exp(-st) + \frac{2w_{\min}t}{d}.\end{aligned}$$

Using the tower rule,

$$\mathbb{E}\left[\exp\left(-s\sum_{j=1}^{T-1} \frac{\mu_j}{w_{\min}}\right)\right] \leq \left(\exp(-st) + \frac{2w_{\min}t}{d}\right)^{T-1},$$

and

$$\mathbb{P}\left(\sum_{j=1}^{T-1} \frac{\mu_j}{w_{\min}} \leq \bar{t}\right) \leq \exp(s\bar{t})\left(\exp(-st) + \frac{2w_{\min}t}{d}\right)^{T-1}.$$

Choosing $t = \frac{1}{s}$, gives

$$\mathbb{P}\left(\sum_{j=1}^{T-1} \frac{\mu_j}{w_{\min}} \leq \bar{t}\right) \leq \exp(s\bar{t})\left(e^{-1} + \frac{2w_{\min}}{sd}\right)^{T-1}.$$

Now we choose

$$s = \frac{2ew_{\min}}{d}$$

to get

$$\mathbb{P}\left(\sum_{j=1}^{T-1} \frac{\mu_j}{w_{\min}} \leq \bar{t}\right) \leq \exp\left(\frac{2ew_{\min}}{d}\,\bar{t} - \frac{1}{4}(T-1)\right)$$

For any $\delta \in (0, 1]$, if

$$\bar{t} \;\leq\; \frac{d}{2ew_{\min}}\left(\frac{1}{4}(T-1) - \log\left(\frac{1}{\delta}\right)\right),$$

then $\mathbb{P}\left(\sum_{j=1}^{T-1} \frac{\mu_j}{w_{\min}} \leq \bar{t}\right) \leq \delta$.

For $\delta = \frac{1}{2}$,

$$\frac{d}{2ew_{\min}}\left(\frac{1}{4}(T-1) - \log\left(\frac{1}{\delta}\right)\right) \geq \frac{1}{2e} \times \frac{d}{w_{\min}}\left(\frac{1}{4}\left\lfloor\frac{\Delta L}{l_1\varepsilon\Delta^0}\right\rfloor - 2\right).$$

In the theorem, we assume that $\frac{L\Delta}{\varepsilon} \geq \bar{c}_1$ for some universal constant $\bar{c}_1$ (we take $\bar{c}_1$ large enough to ensure that $\frac{1}{4}\left\lfloor\frac{\Delta L}{l_1\varepsilon\Delta^0}\right\rfloor - 2 \geq \frac{\Delta L}{8l_1\varepsilon\Delta^0}$). Therefore,

$$\frac{d}{2ew_{\min}}\left(\frac{1}{4}(T-1) - \log\left(\frac{1}{\delta}\right)\right) \geq \frac{1}{16el_1\Delta^0} \times \frac{d}{w_{\min}} \cdot \frac{\Delta L}{\varepsilon}.$$

Thus, we get

$$\mathbb{E}\left[\inf_{x\in G_t}\|\nabla f(x)\|^2\right] > 2\varepsilon\mathbb{P}\left(\sum_{j=1}^{T-1}\frac{\mu_j}{w_{\min}} > t\right) \geq \varepsilon$$

for

$$t = \bar{c}_1 \times \frac{d}{w_{\min}} \cdot \frac{\Delta L}{\varepsilon},$$

where $\bar{c}_1$ is a universal constant.

It is left to recall that f is random. Nevertheless, since

$$\mathbb{E}\left[\mathbb{E}\left[\inf_{x\in G_t}\|\nabla f(x)\|^2 \middle| f\right]\right] > \varepsilon,$$

there exists a deterministic function $\bar{f}$ such that

$$\mathbb{E}\left[\inf_{x\in G_t(\bar{f})}\left\|\nabla \bar{f}(x)\right\|^2\right] > \varepsilon,$$

where $G_t(\bar{f})$ are outputs of the algorithm given $\bar{f}$ (in the statement of the theorem, we rename $\bar{f}$ to f). □

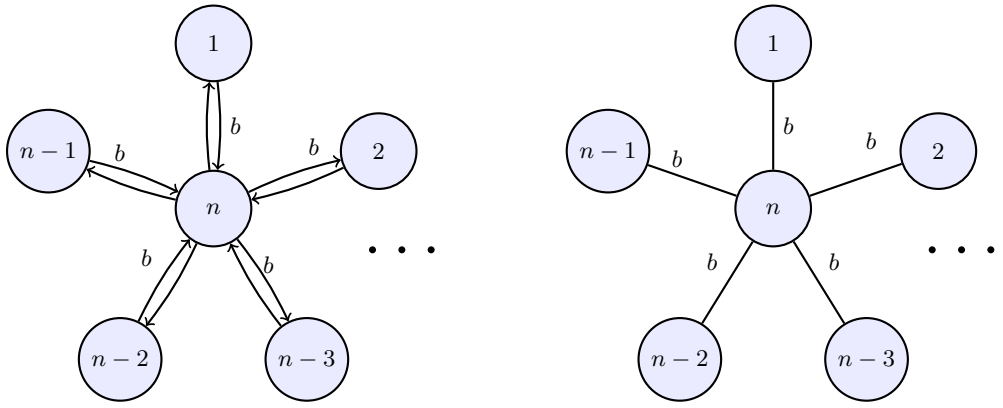
H ONE STEP IN SYNCHRONOUS SGD

In this section, we provide more details on how to implement Synchronous SGD and obtain (2), as discussed in Section 1.1. One way to do it is to fix any pivot worker (server) v that aggregates the stochastic gradients. Then, consider any other worker i. Choose any path p between workers i and v in G. Worker i sends $\nabla f(x^k;\xi_i^k)$ along this path p, which takes at most $d/\min_{(i,j)\in p} b_{ij} \leq d/\min_{(i,j)\in E} b_{ij}$ seconds. There is an important caveat: all workers send their vectors to worker v in parallel, potentially occupying the same edges and causing congestion (e.g., if two workers send their vectors through the same edge, it leads to a $2\times$ slowdown because the edge would process $2 \times d$ coordinates). One known way to fix this is to perform *online in-network aggregation*: each worker streams its vector coordinate-by-coordinate, while intermediate nodes wait for the first coordinate from their children, aggregate these values, and immediately forward the result further. Thus, every edge carries only a single aggregated stream instead of multiple separate vectors, avoiding additional congestion factors, and at most d coordinates pass through each edge. The broadcast operation can be performed in a similar way: every node, upon receiving the new vector x^{k+1}, immediately broadcasts its first coordinates further, before receiving the last ones.

I EXAMPLES OF GRAPHS

In this section, we present examples of graphs that represent the communication topologies of the workers.

I.1 STAR GRAPH



Original graph G Gomory–Hu tree T

Figure 5: Star graph G in the centralized setting and a Gomory–Hu tree of the corresponding undirected graph $\bar{G}$.

Consider Figure 5 where workers communicate through a server (another worker). One can show that a Gomory-Hu tree T of G is shown in Figure 5. In the tree, sorting the weights, we get $\bar{w}_1 = b, \ldots, \bar{w}_{n-1} = b, \bar{w}_n = \infty$. For $k = 1$ in Algorithm 2, $S_{1,1} = [n]$ and $\min_{S \in \mathcal{S}_1} t_1(S) = \left(d/b + h\sigma^2/n\varepsilon\right) + h$. For $k = 2$, $S_{2,1} = \{1\}$ and $S_{2,2} = \{2, \ldots, n\}$ because we remove the edge corresponding to the first worker (w.l.o.g., we could have chosen any other worker), and $\min_{S \in \mathcal{S}_2} t_2(S) = \left(d/b + h\sigma^2/(n-1)\varepsilon\right) + h$. Notice that $\min_{S \in \mathcal{S}_2} t_2(S) \leq \min_{S \in \mathcal{S}_1} t_1(S)$. Repeating the same procedure and removing the edges, we get $\min_{S \in \mathcal{S}_1} t_1(S) \leq \min_{S \in \mathcal{S}_k} t_k(S)$ for all $1 \leq k \leq n-1$. However, for $k = n$, we get $\bar{w}_n = \infty$ and $S_{n,1} = \{1\}, S_{n,2} = \{2\}, \ldots, S_{n,n} = \{n\}$, and $\min_{S \in \mathcal{S}_n} t_n(S) = h\sigma^2/\varepsilon + h$, which can be smaller than $\min_{S \in \mathcal{S}_1} t_1(S)$. Thus, the total time complexity is

$$\mathcal{O}\left(\min\left\{\left(\frac{d}{b} + \frac{h\sigma^2}{n\varepsilon}\right)\frac{L\Delta}{\varepsilon} + \frac{hL\Delta}{\varepsilon}, \frac{h\sigma^2 L\Delta}{\varepsilon^2} + \frac{hL\Delta}{\varepsilon}\right\}\right).$$

I.2 K CLUSTERS

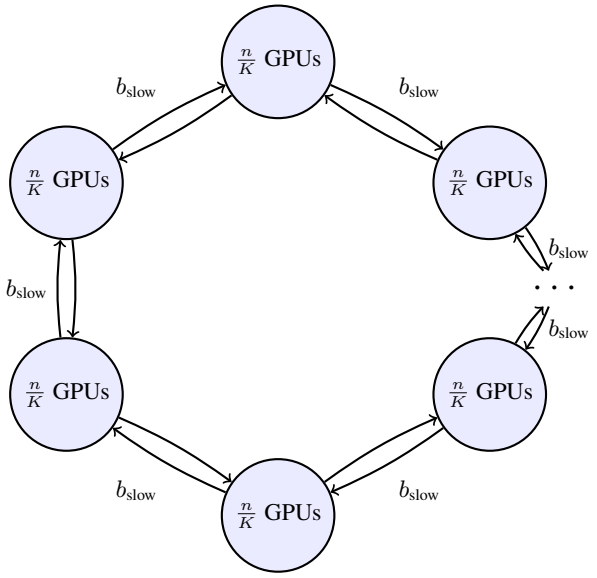


Figure 6: Example of K clusters arranged in a ring. Each cluster contains n/K workers with fast intra-cluster communication (all-to-all with bandwidth ∞). Communication between neighboring clusters occurs with bandwidth b_{slow}.

Example: K clusters in the homogeneous setup. Consider a practical setup (Figure 6), where we have K clusters of n/K workers. The *intra*-communication is fast within each cluster, is all-to-all, and has bandwidth ∞. However, the *inter*-communication between the clusters is relatively slow, with bandwidth b_{slow} (since, for instance, the clusters may be located in different cities). In this case, Theorem 1.5 yields the time complexity $\mathcal{O}\left(\min\left\{\frac{h\sigma^2 L\Delta}{(n/K)\varepsilon^2} + \frac{hL\Delta}{\varepsilon}, \frac{dL\Delta}{b_{\text{slow}}\varepsilon} + \frac{h\sigma^2 L\Delta}{n\varepsilon^2} + \frac{hL\Delta}{\varepsilon}\right\}\right)$, which formalizes the simple idea that either we use one cluster $S^* = [n/K]$ with n/K workers, or we use all clusters ($S^* = [n]$) together but pay the communication cost $dL\Delta/b_{\text{slow}}\varepsilon$ (which may or may not dominate). Notice that we can even consider the case where clusters use different GPUs. In this case, the complexity becomes $\mathcal{O}\left(\min\left\{\min_{i\in[K]} h_i\left(1 + \frac{\sigma^2}{(n/K)\varepsilon}\right), \frac{d}{b_{\text{slow}}} + \min_{m\in[K]}\left(\frac{1}{m}\sum_{i=1}^{m}\frac{1}{h_i}\right)^{-1}\left(1 + \frac{\sigma^2}{\varepsilon m(n/K)}\right)\right\}\frac{L\Delta}{\varepsilon}\right)$, where $h_1 \leq \cdots \leq h_K$ are the computation times of the clusters.

I.3 p-TORUS

Let $p \geq 1$ and $k \geq 2$. The p-Torus is the directed graph $G = (V, E, b)$ defined as follows.

$$V = [k]^p = \{(v_1, \ldots, v_p) : v_i \in [k]\}.$$

For every vertex $v = (v_1, \ldots, v_p)$ and every coordinate $i \in [p]$, there are directed edges

$$v \to (v_1, \ldots, v_i + 1 \pmod{k}, \ldots, v_p)$$

and

$$v \to (v_1, \ldots, v_i - 1 \pmod{k}, \ldots, v_p).$$

Each edge $(u, v) \in E$ has weight $b_{uv} = b > 0$ (for simplicity).

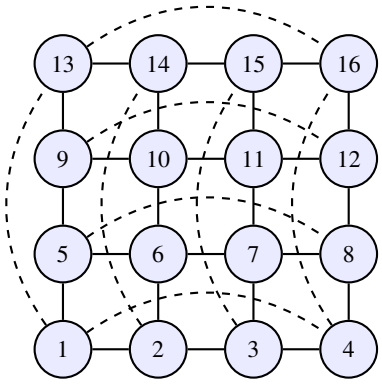


Figure 7: Example of a 2-Torus with $k = 4$. Each node communicates with its four neighbors, and the dashed edges indicate the wrap-around connections. All links have bandwidth b. In the visualization, two arcs merged and visualized with one undirected edge.

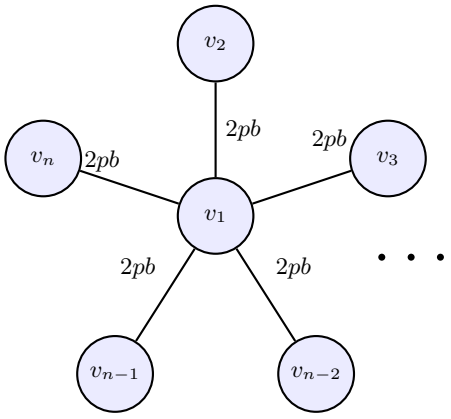


Figure 8: A Gomory–Hu tree T of the undirected graph $\bar{G}$ corresponding to a p-Torus with bandwidths b. The fact that T is a tree with weights $2pb$ follows from the fact that every value of a min-cut in $\bar{G}$ is $2pb$.

J EXAMPLES WITH DIFFERENT GRAPH STRUCTURES IN THE HETEROGENEOUS SETTING

Similarly to Section 4, we consider the same examples in the heterogeneous setting. We also assume that $h_i = h$ for all $i \in [n]$.

Example: One Worker. Assume that $n = 1$, i.e., G is a graph with one node. In this case, Gomory-Hu T is the same as G, $\bar{w}_1 = \infty$ and $S_{1,1} = \{1\}$ in Algorithm 2, and (5) $= \mathcal{O}\left(h\sigma^2 L\Delta/\varepsilon^2 + hL\Delta/\varepsilon\right)$, and we get the same result as in the homogeneous setup.

Example: Star Graph (Centralized Setting; Figure 5). In this case, $\bar{w}_1 = b$. Thus,

$$(5) = \mathcal{O}\left(\left(\frac{d}{b} + \frac{h\sigma^2}{n\varepsilon}\right)\frac{L\Delta}{\varepsilon} + \frac{hL\Delta}{\varepsilon}\right).$$

Example: p-Torus. Since $\bar{w}_1 = 2pb$ in Figure 8,

$$(5) = \mathcal{O}\left(\left(\frac{d}{pb} + \frac{h\sigma^2}{n\varepsilon}\right)\frac{L\Delta}{\varepsilon} + \frac{hL\Delta}{\varepsilon}\right).$$

Example: all-to-all. In the all-to-all graph G,

$$(5) = \mathcal{O}\left(\left(\frac{d}{(n-1)b} + \frac{h\sigma^2}{n\varepsilon}\right)\frac{L\Delta}{\varepsilon} + \frac{hL\Delta}{\varepsilon}\right),$$

for all $n \geq 2$.

Example: K clusters. Using Figure 6, one can show that $\bar{w}_1$ and Theorem 1.7 yields the time complexity

$$\mathcal{O}\left(\frac{dL\Delta}{b_{\text{slow}}\varepsilon} + \frac{h\sigma^2 L\Delta}{n\varepsilon^2} + \frac{hL\Delta}{\varepsilon}\right).$$

Unlike the homogeneous setting (Section I.2), this complexity tends to ∞ if $b_{\text{slow}} \to 0$.

Example: optimization with switches. The heterogeneous setting can also support switches, but this would require extending the setup described in Section 5. Briefly, under Assumption 1.4, we assume that there are two subsets, S_w and S_s, such that $S_w \cup S_s = [n]$ and $S_w \cap S_s = \emptyset$, where S_w is the set of real workers and S_s is the set of switches. Then, we solve

$$\min_{x \in \mathbb{R}^d} \left\{ f(x) := \frac{1}{|S_w|} \sum_{i \in S_w} f_i(x) \right\}. \tag{52}$$

Using a modified version of Algorithm 4, where only $|S_w|$ workers compute stochastic gradients and run AllReduce, one can extend Theorem 1.7 and prove the time complexity

$$\mathcal{O}\left(\max\left\{\frac{d}{\alpha_{\bar{G}}(S_w)}, \max_{i \in S_w} h_i, \frac{\sigma^2}{|S_w|\varepsilon}\left(\frac{1}{|S_w|}\sum_{i \in S_w} h_i\right)\right\}\frac{L\Delta}{\varepsilon}\right)$$

seconds, where $\alpha_{\bar{G}}(S_w)$ is the minimum value of an S_w-cut in $\bar{G}$ (Definition E.5 and Theorem F.1). When $S_w = [n]$ and $S_s = \emptyset$, this complexity reduces to (5).

K Practical Guidelines and Numerical Experiments

In this section, we discuss practical guidelines and conduct numerical experiments of the new algorithm, Grace SGD. As we explain in Section 2, Grace SGD is simple. It is just a stochastic gradient method with preprocessing (Algorithm 2) and the optimal-bandwidth AllReduce (Algorithm 3). Implementing Algorithm 2 is straightforward. Finding a Gomory-Hu tree is a standard graph problem, and there are many open-source libraries with implementations (for instance, use `gomory_hu_tree` from `NetworkX` (Hagberg et al., 2007)). Besides that, Algorithm 2 is a standard one loop function where we find all steps can be implemented in Python.

At the same time, in general, implementing the optimal-bandwidth AllReduce algorithm described in Algorithm 3 is arguably trickier. The first non-trivial step is to solve the *Steiner Tree Packing* problem. While it is a classical problem in computer science, Lau (2004) only recently proposed a polynomial-time algorithm that finds at least $\lfloor \alpha_{\bar{G}}(S)/26 \rfloor$ edge-disjoint trees that contain every node in $S \subseteq V$, where $\alpha_{\bar{G}}(S)$ is the minimum value of an S-cut in $\bar{G}$ (Definition E.5 and Theorem F.1). Once the Steiner Tree Packing is solved (up to a constant factor), the reduce and broadcast steps can be implemented in practice using the standard communication libraries.

The description of the polynomial-time algorithm in Lau (2004) is somewhat non-trivial. Fortunately, for standard graph structures, implementing an optimal-bandwidth AllReduce is more straightforward. In particular, consider the 2-Torus G from Figure 7. Without loss of generality, assume that $b = 1$ and $n = (2k+1)^2$ for some $k \geq 1$. We fix the node with coordinate (k, k) and refer to it as the pivot node (see Figure 9). In this graph, every node has 4 outgoing edges. The idea of this optimal-bandwidth AllReduce is that every node i splits its local vector into four blocks $(a_{i,1}, a_{i,2}, a_{i,3}, a_{i,4})$ and sends them in different directions. The goal is now to choose the directions in such a way that each block eventually reaches the pivot worker without cycles, while intermediate nodes use online in-network aggregation to combine blocks with the same index and avoid congestion.

Formally, for all (i, j) with $i \neq k$ and $j \neq k$, the node sends the first block to $(i+1 \bmod (2k+1),\ j)$, the second block to $(i,\ j+1 \bmod (2k+1))$, the third block to $(i-1 \bmod (2k+1),\ j)$, and the fourth block to $(i,\ j-1 \bmod (2k+1))$. For nodes with $i = k$ or $j = k$, the node sends the first block to $(i,\ j+1 \bmod (2k+1))$, the second block to $(i+1 \bmod (2k+1),\ j)$, the third block to $(i,\ j-1 \bmod (2k+1))$, and the fourth block to $(i-1 \bmod (2k+1),\ j)$.

See visualization for $k = 2$ in Figure 9. For instance, node $(3, 0)$ sends the first block to $(4, 0)$, which aggregates its own block with it, and sends the sum to $(0, 0)$, which also adds its first block. And it happens until the total sum arrives to $(2, 2)$. Notice that $(2, 1)$ aggregates the sum from nodes $(2, 0)$ and $(1, 1)$.

Compared to the naive strategy, where every worker sends the full vectors to the pivot worker, using this routing algorithm and online in-network aggregation, we can speed up the reduce operation by $4\times$, since each block arrives at the central node via different and independent paths. The improvement by $4\times$ is expected since the value of a min-cut in the graph is 4. The broadcast can be implemented similarly by reversing the edges. We could have obtained a 2–4$\times$ speedup using Algorithm 3, but the strategy specialized for the 2-Torus is arguably simpler. For the ring graph, the algorithm is similar and one get a $2\times$ communication speed up. In general, for a p-Torus, the speedup should be $\Theta(p)$ (at least with Algorithm 3 due to Theorem F.1).

K.1 Numerical experiments

We now consider Grace SGD and compare it to Synchronous SGD. The environment was emulated in Python 3 with one Intel(R) Xeon(R) Platinum 8168 CPU @ 2.70GHz. We assume that i^{th} worker requires $h_i = 1$ second to calculate one stochastic gradient. We assume that the graph G is 2-Torus (see Figure 7) with bandwidth $b = 0.1$ for all the edges. We consider the setup with $n = 100$. In all methods, we tune the step sizes from the set $\{2^i \mid i \in [-20, 20]\}$. In (1), we consider the standard logistic regression problem with *MNIST* dataset (LeCun et al., 2010), where every worker samples one sample from the dataset and calculates a stochastic gradient.

Naive synchronization vs. optimal-bandwidth AllReduce. In this part, we compare Synchronous SGD with naive synchronization, where full vectors are sent to neighbors (a similar problem in gossip methods), and Grace SGD with optimal-bandwidth AllReduce. In Figure 10, we set $\sigma^2/\varepsilon = 100$ for

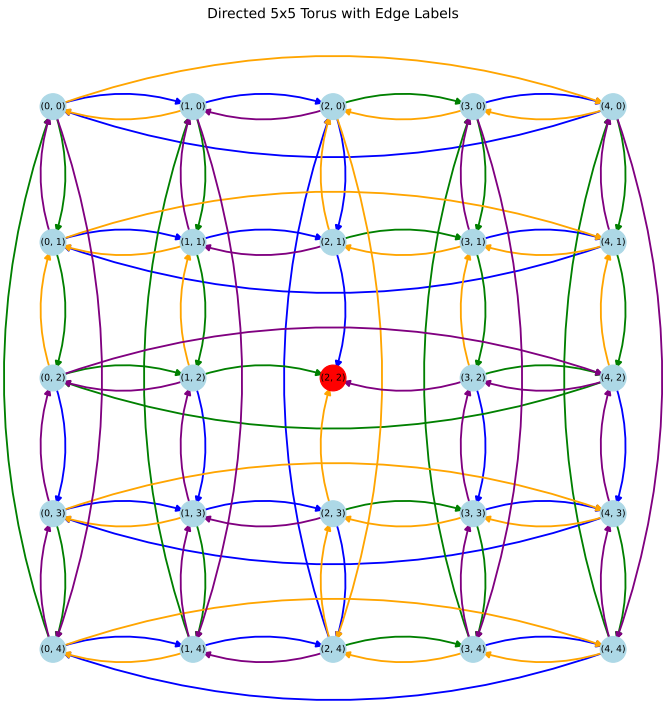


Figure 9: 2-Torus and Reduce routing. Each color corresponds to one of the four blocks.

Algorithm 1 to ensure that, in both Grace SGD and Synchronous SGD, all workers compute one stochastic gradient per iteration, providing a fair comparison of the communication strategies. In practice, we observe that Grace SGD converges faster.

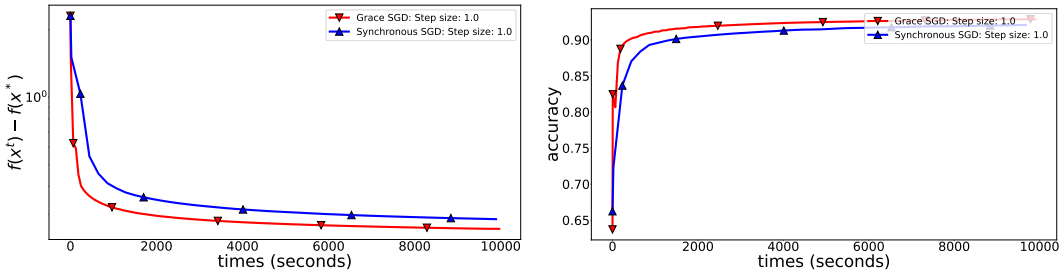


Figure 10: Experiments with Synchronous SGD and Grace SGD

K clusters. We now consider the practical setup from Figure 6 to test the effectiveness of Algorithm 2 and the usefulness of local training within a single cluster when b_{slow} is small. We take $n = 100$ and $K = 10$. For this setup, Algorithm 2 can either return $S^* = [n]$ (all workers) or $S^* = [n/K]$ (one cluster), depending on the value of b_{slow} (see the discussion in Section I.2). In Figure 11, we compare the performance of these two options and observe that $S^* = [n]$ has faster convergence, which concurs with the discussion in Section I.2. However, as we start decreasing the bandwidth in Figures 12 and 13, we observe that local training in one cluster is faster in Figure 13, which supports the fact that $S^* = [n/K]$, as discussed in Section I.2.

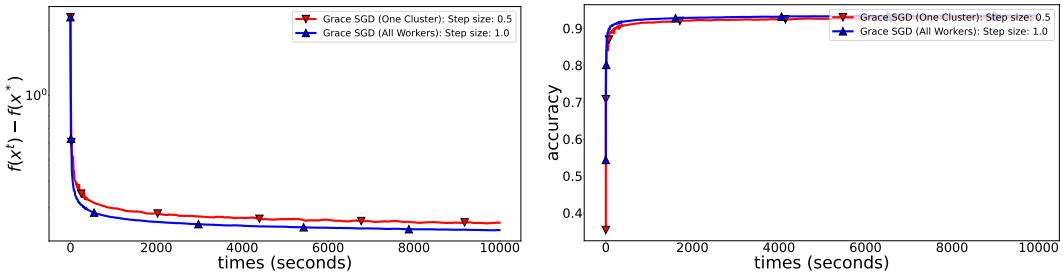


Figure 11: Experiments with Grace SGD and $b_{\text{slow}} = \infty$

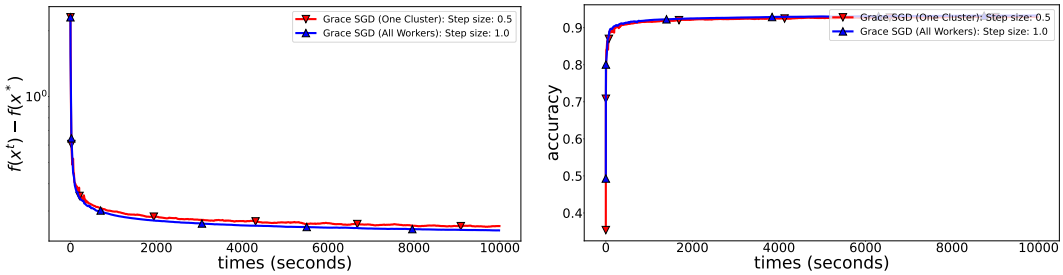


Figure 12: Experiments with Grace SGD and $b_{\text{slow}} = 1$

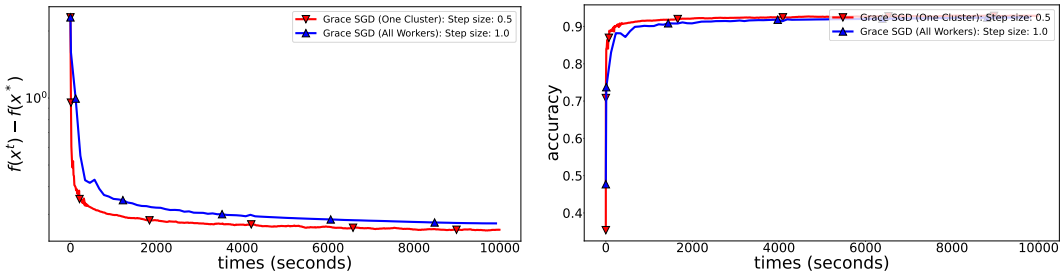


Figure 13: Experiments with Grace SGD and $b_{\text{slow}} = 0.1$

L Graph-Bandwidth Communication Model with Latencies and Fluid Assumption

Graph-Bandwidth Communication Model with Latencies. We can extend Graph-Bandwidth Communication Model by assuming that each edge has a latency $\ell_{ij} \geq 0$, and use the standard α–β model, where transferring s coordinates through edge (i, j) takes $\ell_{ij} + s/b_{ij}$ seconds. Notice that the first term does not depend on the amount of transmitted information s, and since workers typically send large volumes of data, the latency term does not dominate.

Formally, under this model, instead of the main upper bound results in Theorems 1.5 and 1.7, we obtain (3) $+ \mathcal{O}\left(\ell_{\max} L\Delta/\varepsilon\right)$ and (5) $+ \mathcal{O}\left(\ell_{\max} L\Delta/\varepsilon\right)$, respectively, where $\ell_{\max} = \max_{(i,j)\in E} \ell_{ij}$. The only result that needs to be adjusted is Theorem F.1, where the reduce and broadcast operations take $\mathcal{O}\left(\ell_{\max} + d/p\right)$ seconds instead of $\mathcal{O}\left(d/p\right)$ (the asymptotic rate remains $\mathcal{O}\left(d/p\right)$ if d is large). For instance, if $h_i = h$ for all $i \in [n]$, then the time complexity of Grace SGD in the homogeneous case is

$$\mathcal{O}\left(\min\left\{\min_{k\in[n-1]}\left(\ell_{\max} + \frac{d}{\bar{w}_k} + \min_{p\in[k]} \frac{h\sigma^2}{|S_{k,p}|\,\varepsilon} + h\right)\frac{L\Delta}{\varepsilon}\right\}, \frac{h\sigma^2 L\Delta}{\varepsilon^2} + \frac{hL\Delta}{\varepsilon}\right).$$

Thus, if d is large, which is the case in modern training, the term involving $\ell_{\max}$ does not dominate, and all our results and conclusions remain valid. For clarity, and because d is large in practice, we ignore latencies in the main part. Nevertheless, deriving tight lower bounds under the model with large latencies is an important direction for future work.

Fluid Assumption. In the main model, we assume that transmitted information is infinitely divisible and can flow continuously through the network. In particular, a vector or a number can be split into arbitrarily small parts, and an intermediate worker can start forwarding the received parts without waiting for the entire vector. Notice that in theory it is true since all numbers are an infinite sequence of bits that we can infinitely split into chunks (unlike practical systems, where all numbers are represented by 32/64 bits).

In practice, vectors and numbers are transmitted in finite packets or chunks. However, when d is large or a number is represented by many bits, a vector or a number can be divided into many small chunks and communicated in a pipelined manner. Thus, the fluid model captures the bandwidth-dominated regime while ignoring packet granularity and the corresponding lower-order synchronization overhead, assuming that the chunks are infinitely small.

L.1 Relation to Previous Communication Model Assumptions

Our Graph-Bandwidth Communication Model is related to classical bandwidth–latency models used in parallel and distributed computing, such as LogP and LogGP (Culler et al., 1993; Alexandrov et al., 1995). It is also related to the literature on topology-aware collective communication. Previous works optimize collective communication operations in MPI (Thakur et al., 2005), develop efficient AllReduce algorithms (Patarasuk & Yuan, 2009), create collective algorithms for specific network topologies (Cai et al., 2021), exploit heterogeneous communication links (Wang et al., 2020), and construct collective schedules for general networks (Zhao & Krishnamurthy, 2023). Moreover, in-network aggregation, which is allowed in our model, has been implemented in practical distributed learning systems (Sapio et al., 2021).

The operations listed in Section 1 (e.g., split vector, interleaving, in-network aggregation) and supported by Graph-Bandwidth Communication Model are commonly used in practical collective communication. In particular, efficient AllReduce algorithms rely on message splitting and pipelining (Thakur et al., 2005; Patarasuk & Yuan, 2009), while in-network aggregation has also been implemented in distributed learning systems (Sapio et al., 2021).

While there are indeed connections to previous works and distributed problems, our algorithms and lower bounds are very specific to non-convex stochastic optimization. The constructed lower bound uses the special zero-chain functions introduced by (Carmon et al., 2020; Tyurin, 2026), and analyzes how algorithms might transfer coordinates optimally to disseminate the last non-zero coordinates as quickly as possible. The concept of "non-zero" coordinates is very specific to optimization (Nesterov, 2018). Moreover, it is important that the information aggregated by the workers consists of vectors that can be summed or averaged. Therefore, these vectors can be aggregated using an

optimal-bandwidth AllReduce algorithm. Many distributed problems do not admit such a simple associative aggregation and may require substantially more complicated communication protocols or the transmission of additional information. In our approach, it is also critical that we synchronize the workers and run AllReduce simultaneously. In fact, it is non-obvious that we can achieve the optimal complexity without full asynchronicity. Finally, in the homogeneous setting, another specific factor is that we ignore some workers, whereas there are many problems for which it is necessary to obtain information from all of them.