Parameter servers for machine learning do not scale?
Well, **one size does not fit all.**

NuPS: A Parameter Server for Machine Learning with Non-Uniform Parameter Access

Alexander Renz-Wieland, Rainer Gemulla, Zoi Kaoudi, Volker Markl

To appear in SIGMOD 2022

Motivation

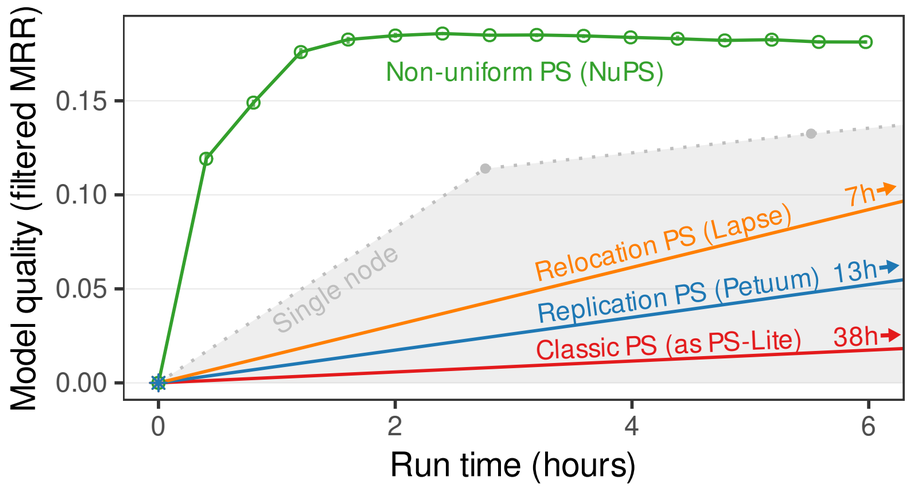
- Parameter Servers (PSs) facilitate distributed machine learning
- But existing PSs are inefficient for non-uniform access:
  - Skew: PSs inefficient because they manage all parameters identically
  - Sampling: PSs inefficient because sampling entails randomized access

NuPS

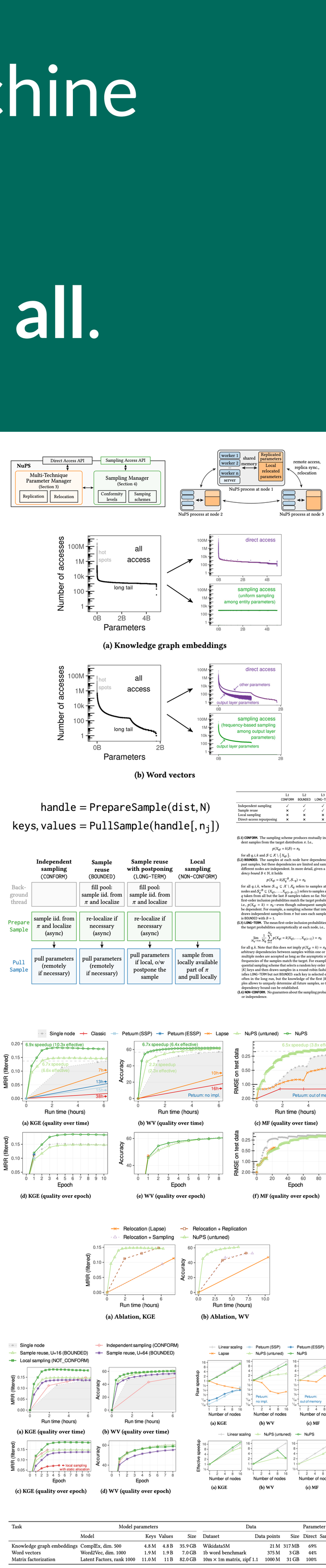
1. Supports multiple management techniques and picks a suitable one for each parameter:
   - Replication is efficient for hot spot parameters
   - Relocation is efficient for long tail parameters
2. Supports sampling directly via suitable primitives and sampling schemes that allow for a controlled quality—efficiency trade-off

Results

1. NuPS outperformed existing PSs by up to one order of magnitude across multiple ML tasks
2. NuPS provided up to linear scalability
3. Most efficient was to replicate a small set of hotspot parameters and relocate all others



Parameter server performance for training large knowledge graph embeddings on an 8-node cluster.



	L1 CONFORM	L2 BOUNDED	L3 LONG-TERM
Independent sampling	✓	✓	✓
Sample reuse	×	✓	✓
Local sampling	×	×	×
Direct-access repurposing	×	×	×

(L1) **CONFORM.** The sampling scheme produces mutually independent samples from the target distribution π. I.e.,

$$P(X_{qi} = k|S) = \pi_k$$

for all q, i, k and $S \subseteq \mathcal{X} \setminus \{X_{qi}\}$

(L2) **BOUNDED.** The samples at each node have dependencies on past samples, but these dependencies are limited and samples at different nodes are independent. In more detail, given a *dependency bound* $B \in \mathbb{N}$, it holds

$$P(X_{qi} = k|S^{qi}_{\mathcal{X}}, \mathcal{S}_{\neg q}) = \pi_k$$

for all q, i, k, where $\mathcal{S}_{\neg q} \subseteq \mathcal{X} \setminus \mathcal{X}_q$ refers to samples at other nodes and $S^{qi}_{\mathcal{X}} \subseteq \{X_{q1}, \ldots, X_{q(i-B-1)}\}$ refers to samples at node q taken from all but the last B samples taken so far. Note that first-order inclusion probabilities match the target probabilities—i.e., $p(X_{qi} = k) = \pi_k$—even though subsequent samples may be dependent. For example, a sampling scheme that internally draws independent samples from π but uses each sample twice is BOUNDED with $B = 1$.

(L3) **LONG-TERM.** The mean first-order inclusion probabilities match the target probabilities asymptotically at each node, i.e.,

$$\lim_{N_q \to \infty} \frac{1}{N_q} \sum_{i=1}^{N_q} p(X_{qi} = k|X_{q1}, \ldots, X_{q(i-1)}) = \pi_k \quad (1)$$

for all q, k. Note that this does *not* imply $p(X_{qi} = k) = \pi_k$. Also, arbitrary dependencies between samples within one or across multiple nodes are accepted as long as the asymptotic relative frequencies of the samples match the target. For example, a sequential sampling scheme that selects a random key order for the $|K|$ keys and then draws samples in a round-robin fashion satisfies LONG-TERM but not BOUNDED: each key is selected equally often in the long run, but the knowledge of the first $|K|$ samples allows to uniquely determine all future samples, so that no dependency bound can be established.

(L4) **NON-CONFORM.** No guarantees about the sampling probabilities or independence.

Task	Model parameters			Data			Parameter access	
	Model	Keys	Values	Size	Dataset	Data points	Size	Direct	Sampling
Knowledge graph embeddings	ComplEx, dim. 500	4.8 M	4.8 B	35.9 GB	Wikidata5M	21 M	317 MB	69%	31%
Word vectors	Word2Vec, dim. 1000	1.9 M	1.9 B	7.0 GB	1b word benchmark	375 M	3 GB	44%	56%
Matrix factorization	Latent Factors, rank 1000	11.0 M	11 B	82.0 GB	10m × 1m matrix, zipf 1.1	1000 M	31 GB	100%	0%

Technische Universität Berlin

BIFOLD

software campus

Read the full paper