PINCH: Accelerating Distributed GNN Training through In-Kernel Operation Using eBPF

Jianchang Su[1], Yifan Zhang[1], Wei Zhang[1]

[1]University of Connecticut

GNN Training Communication

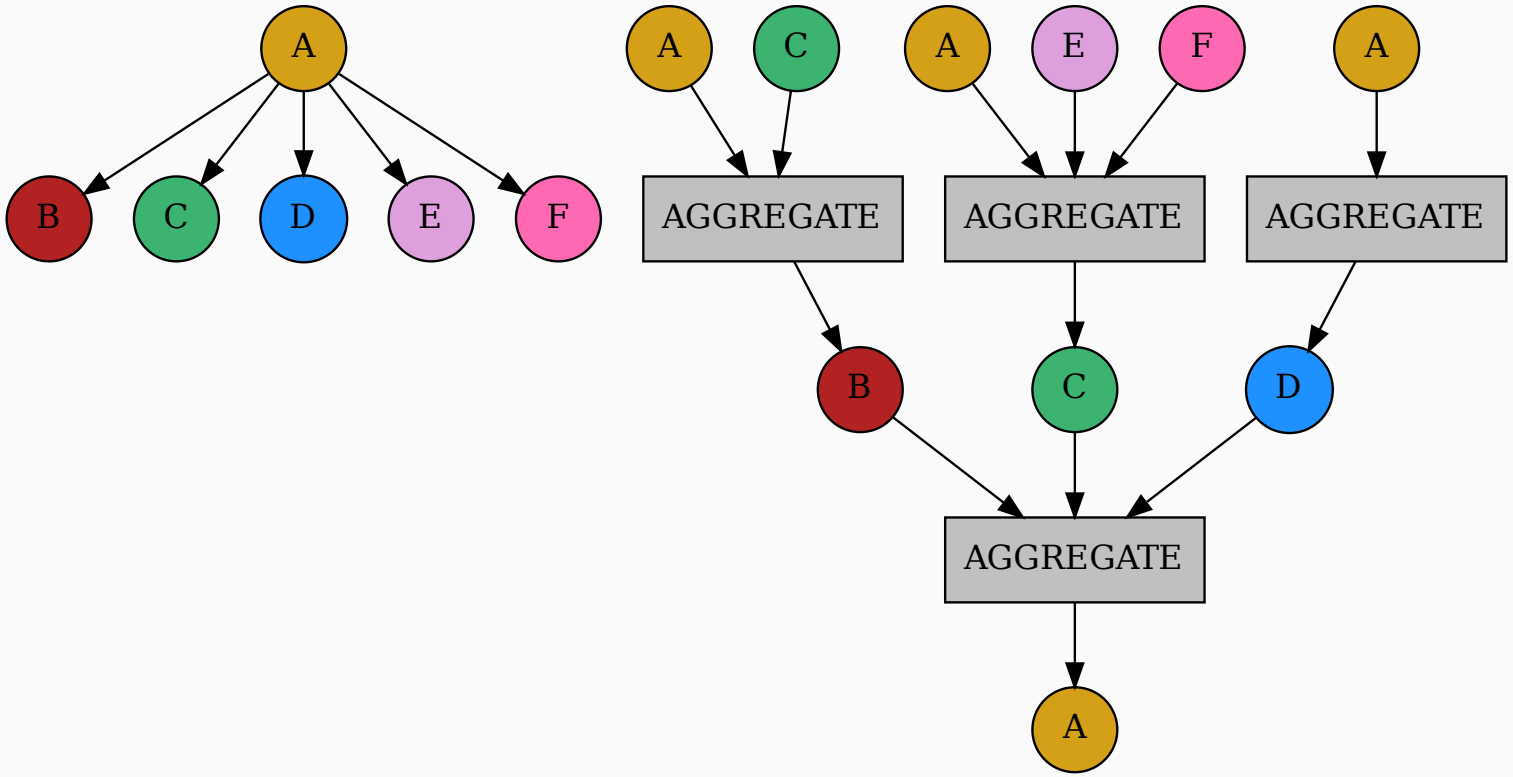

Figure 1: GNN Training Broadcast and Aggregation

Limitations of Existing Distributed GNN Frameworks

- Rely on socket-based communication, incurring high latency and CPU overhead
- Follow the pull-based paradigm, leading to redundant communication and blocking waits
- Communication overhead can take up to 80% of total training time [1, 2, 3]

PINCH: Overview

- PINCH: A novel system to accelerate distributed GNN training
- Leverages eBPF and kernel hooks (XDP and TC) to optimize communication
- Key techniques:
  - In-kernel neighborhood aggregation via eBPF and XDP
  - In-kernel broadcasting through eBPF and TC
  - Caching and reusing aggregated embeddings with eBPF maps

eBPF and Kernel Hooks

- eBPF: Allows safely executing custom code inside the Linux kernel
- XDP: eBPF-based packet processing framework for low latency and high throughput
- TC: eBPF hook point for classifying, filtering, and manipulating packets

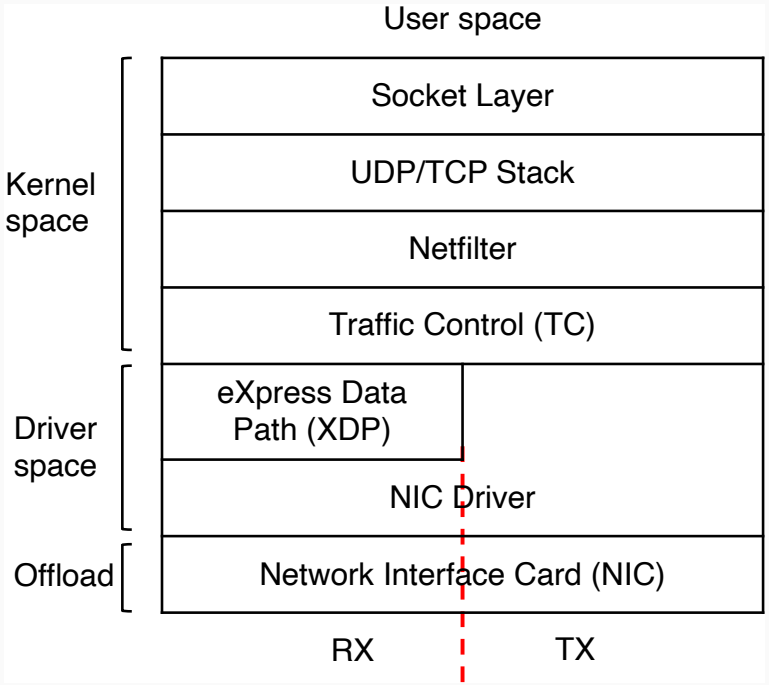

Figure 2: Linux kernel networking stack and eBPF XDP/TC hooks

PINCH Architecture

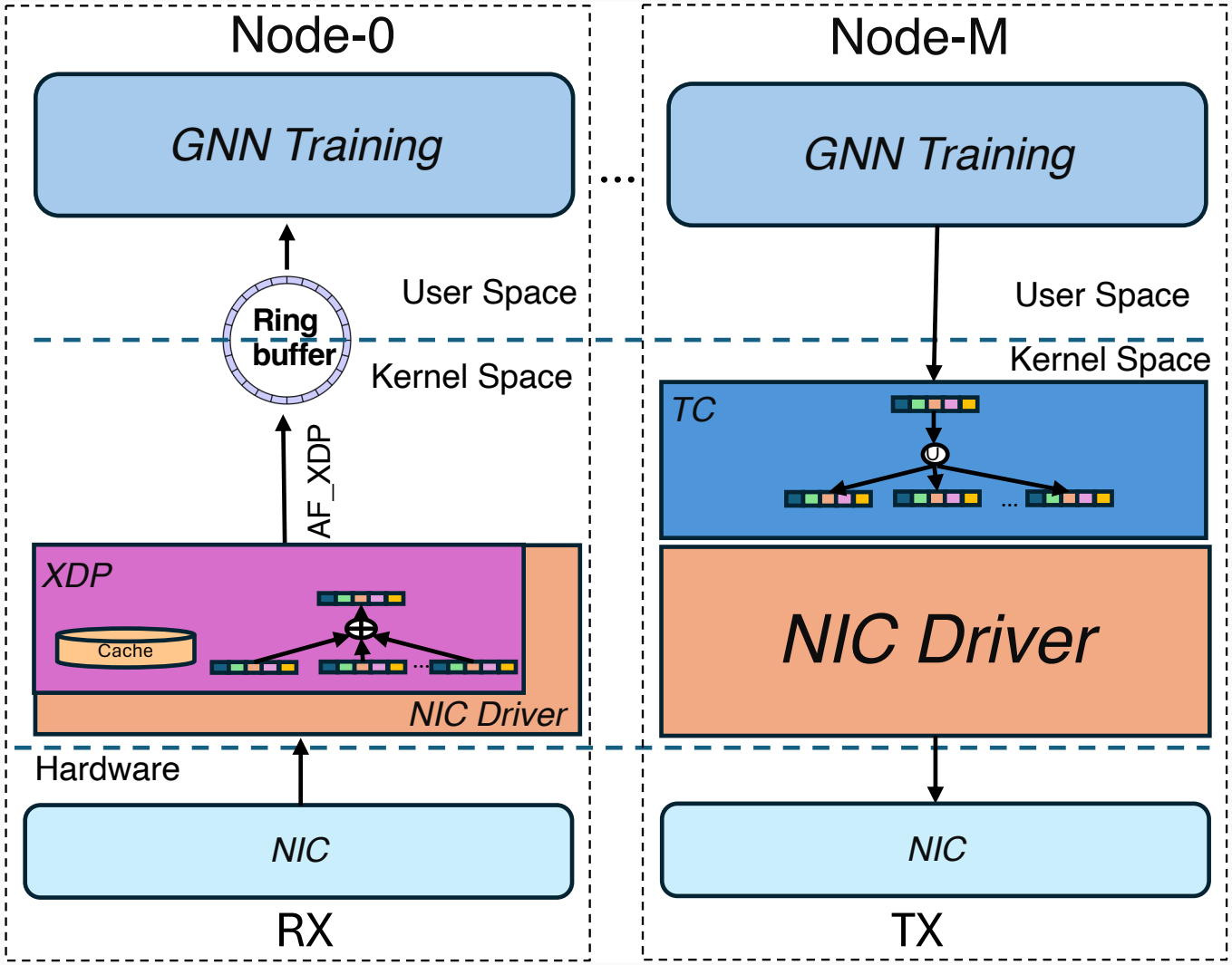

Figure 3: PINCH system architecture

In-XDP Aggregation

- Performs neighbor embedding aggregation directly in the kernel
- Implemented as an eBPF program attached to the XDP hook
- Leverages quantization and tail calls to overcome eBPF limitations
- Caches aggregated embeddings in eBPF maps for reuse

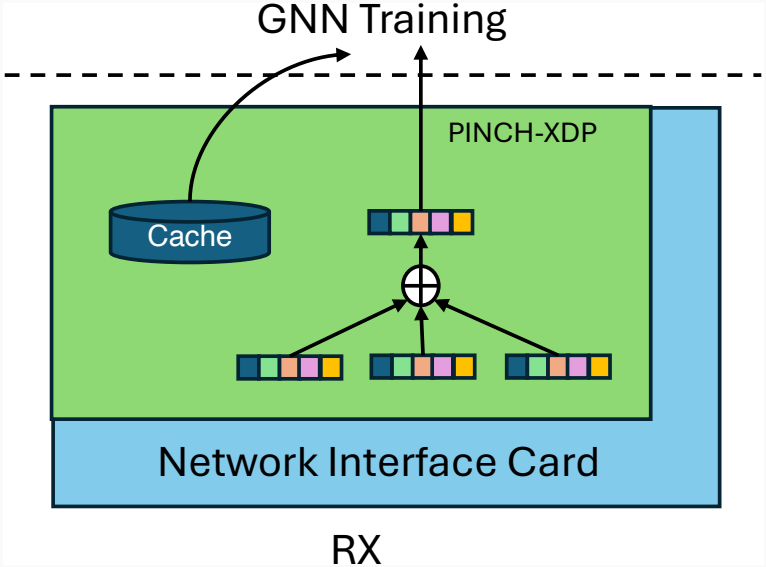

Figure 4: In-XDP aggregation workflow

In-TC Broadcasting

- Constructs and sends broadcast packets containing node embeddings
- Implemented as an eBPF program attached to the TC hook
- Leverages `bpf_clone_redirect()` for efficient packet cloning and redirection

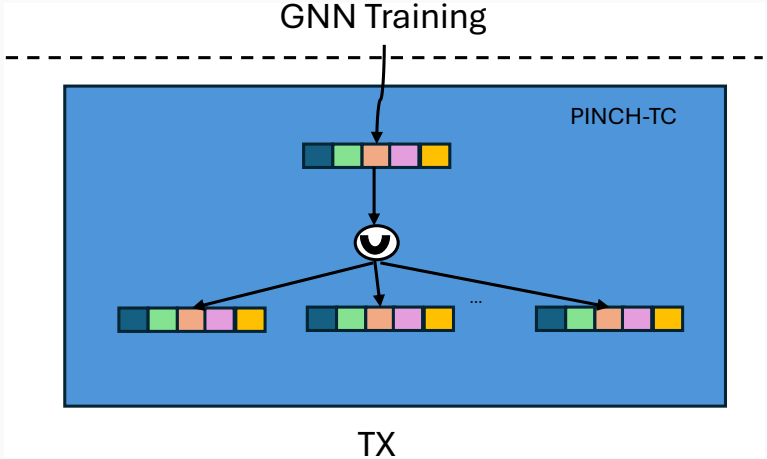

Figure 5: In-TC broadcasting workflow

Kernel-User Crossings in PINCH

- Minimizes crossings by performing aggregation and broadcasting in the kernel
- In-XDP Aggregation: Aggregates node embeddings in the kernel
- In-TC Broadcasting: Constructs and sends broadcast packets in the kernel
- Caches and reuses aggregated embeddings in eBPF maps

Fault Tolerance

- Designed to be resilient to packet loss and duplication
- In-TC Broadcasting: Retransmission and sliding window for packet loss
- In-XDP Aggregation: Idempotent processing using eBPF maps

Limitations and Future Work

- Flexible Aggregation:
  - Limited aggregation functions
  - Future: Extend eBPF VM or use approximate computing
- Communication Patterns:
  - Tailored to message-passing GNNs
  - Future: Support more GNNs and big data apps
- CPU and NUMA:
  - No CPU affinity or NUMA impact consideration
  - Future: Explore CPU pinning, NUMA-aware scheduling, cache-aware data placement

Related Work

- PipeGCN, Cluster-GCN, CAGNET
- CCP, BMC, Syrup, SPRIGHT, XRP, SynCord

Conclusion

- PINCH optimizes distributed GNN training using eBPF and kernel-level packet processing
- Employs in-kernel broadcasting, aggregation, caching, and fault tolerance mechanisms
- Potential for integrating communication layer with kernel network stack
- Future work: Advanced GNN models, eBPF for other distributed system challenges

References i

Zhenkun Cai, Xiao Yan, Yidi Wu, Kaihao Ma, James Cheng, and Fan Yu.
Dgcl: an efficient communication library for distributed gnn training.
In *Proceedings of the Sixteenth European Conference on Computer Systems*, EuroSys '21, page 130–144, New York, NY, USA, 2021. Association for Computing Machinery.

Jingshu Peng, Zhao Chen, Yingxia Shao, Yanyan Shen, Lei Chen, and Jiannong Cao.
Sancus: staleness-aware communication-avoiding full-graph decentralized training in large-scale graph neural networks.
Proc. VLDB Endow., 15(9):1937–1950, may 2022.

References ii


Alok Tripathy, Katherine A. Yelick, and Aydin Buluç.
Reducing communication in graph neural network training.
CoRR, abs/2005.03300, 2020.

Thank You!