MNMGDatalog: A Scalable Multi-Node Multi-GPU Datalog Engine

Ahmedur Rahman Shovon (ashov@uic.edu)
Sidharth Kumar (sidharth@uic.edu)
University of Illinois Chicago

Introduction

Declarative programming focuses on "WHAT" not on "HOW"

Users

UserID	UserName	UserEmail	Country
101	Alice	alice@example.com	USA
102	Bob	bob@example.com	USA
103	Eve	eve@example.com	Canada

WHAT **SELECT** UserID **FROM** Users **WHERE** Country='USA'; HOW

Advanced approach: Logic programming (Datalog)

Datalog rules to compute Transitive Closure (TC) of a relation

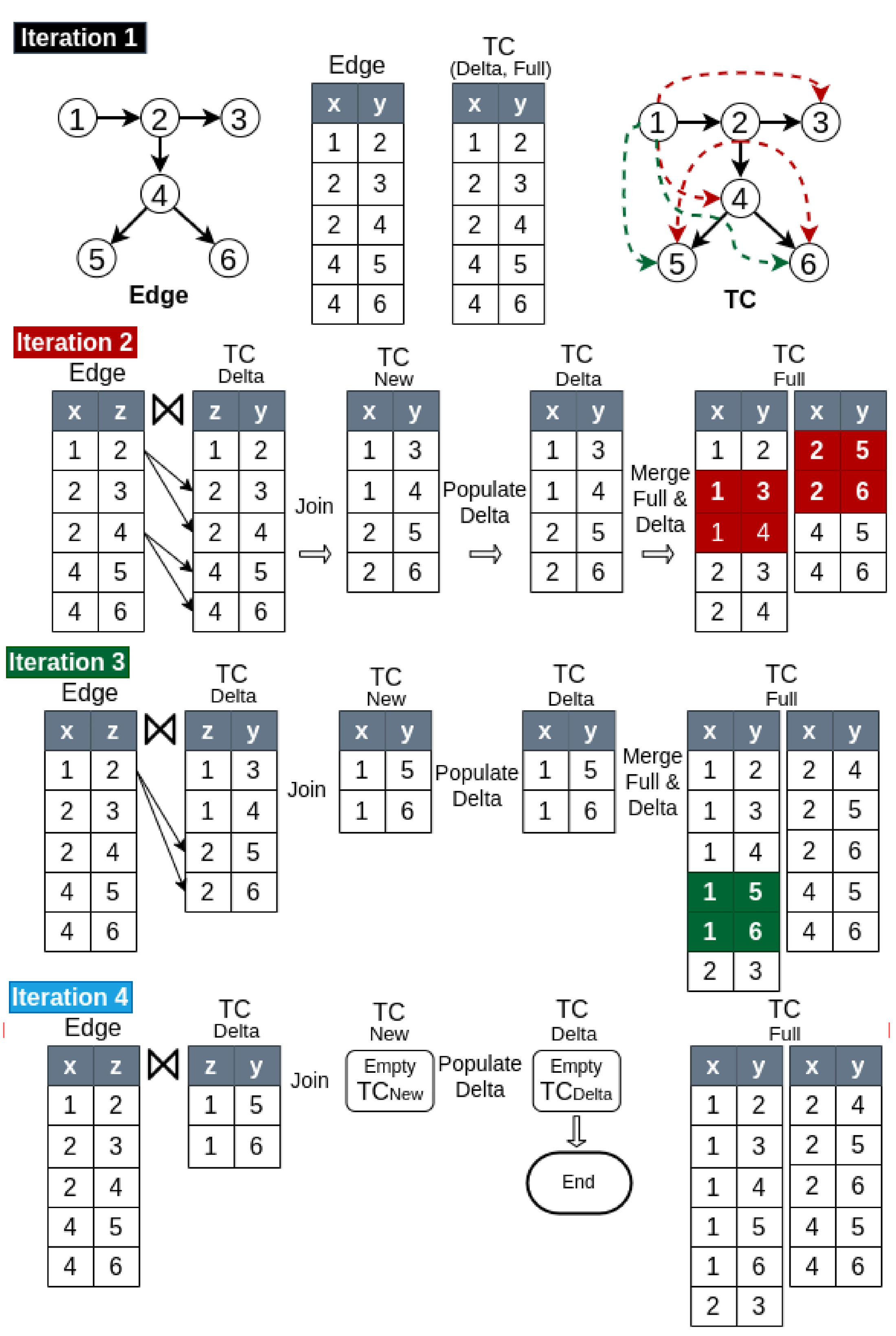
```
TC(x, y) :- Edge(x, y).
TC(x, z) :- TC(x, y), Edge(y, z).
```

Operationalized as a **fixed-point iteration** using F_G

$$F_G(T) \triangleq G \cup \Pi_{1,2}(\rho_{0/1}(T) \bowtie_1 G)$$

Datalog rules compiled down to **iterative relational algebra**

Iterations in Transitive Closure (TC) Computation



Datalog Engine Categories

Multi-threaded	Distributed (Apache Spark)	Multi-node Multi-threaded	Single-GPU	Multi-node Multi-GPUs (MNMG)
Soufflé	RDFox	SLOG	GPUJoin	?
LogicBlox	Radlog	PRAM	GPULog	

Requirements for MNMG Datalog Engine

1. Workload partitioning
2. Data representation
3. Efficient communication
4. Tuple materialization

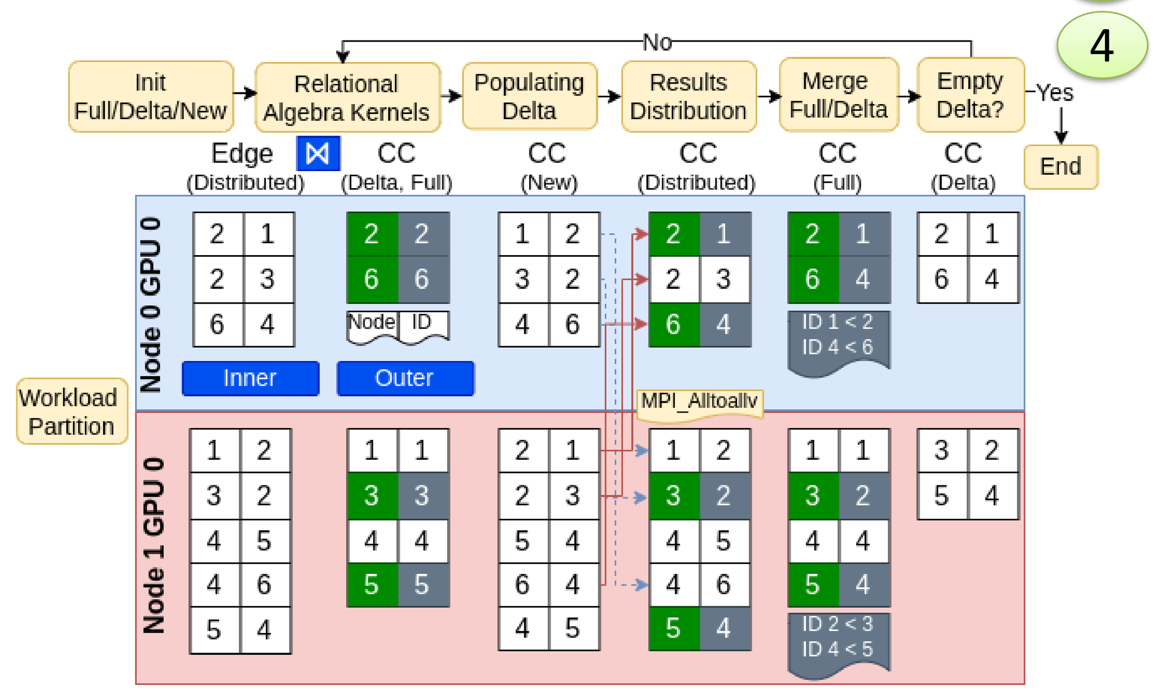
MNMGDatalog is the first MNMG Datalog engine

Highest performant Datalog engine
Single-GPU: Up to 7× speedup over GPULog
Multi-threaded: Up to 33× over Soufflé
Distributed: Up to 31.9× speedup over SLOG

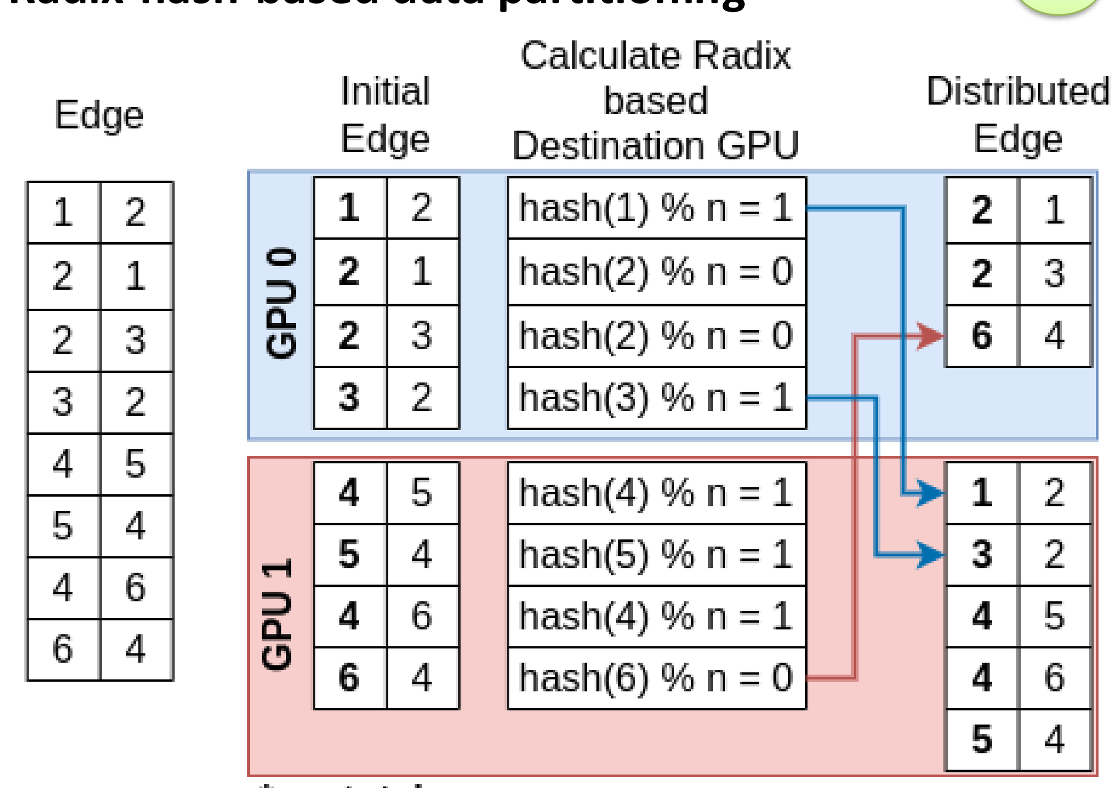
MNMGDatalog Implementation

MNMGDatalog uses radix-hash partitioning and non-uniform all-to-all communication with GPU-aware hash tables for efficient tuple materialization

Tuple materialization in fixed point iteration (2) (4)

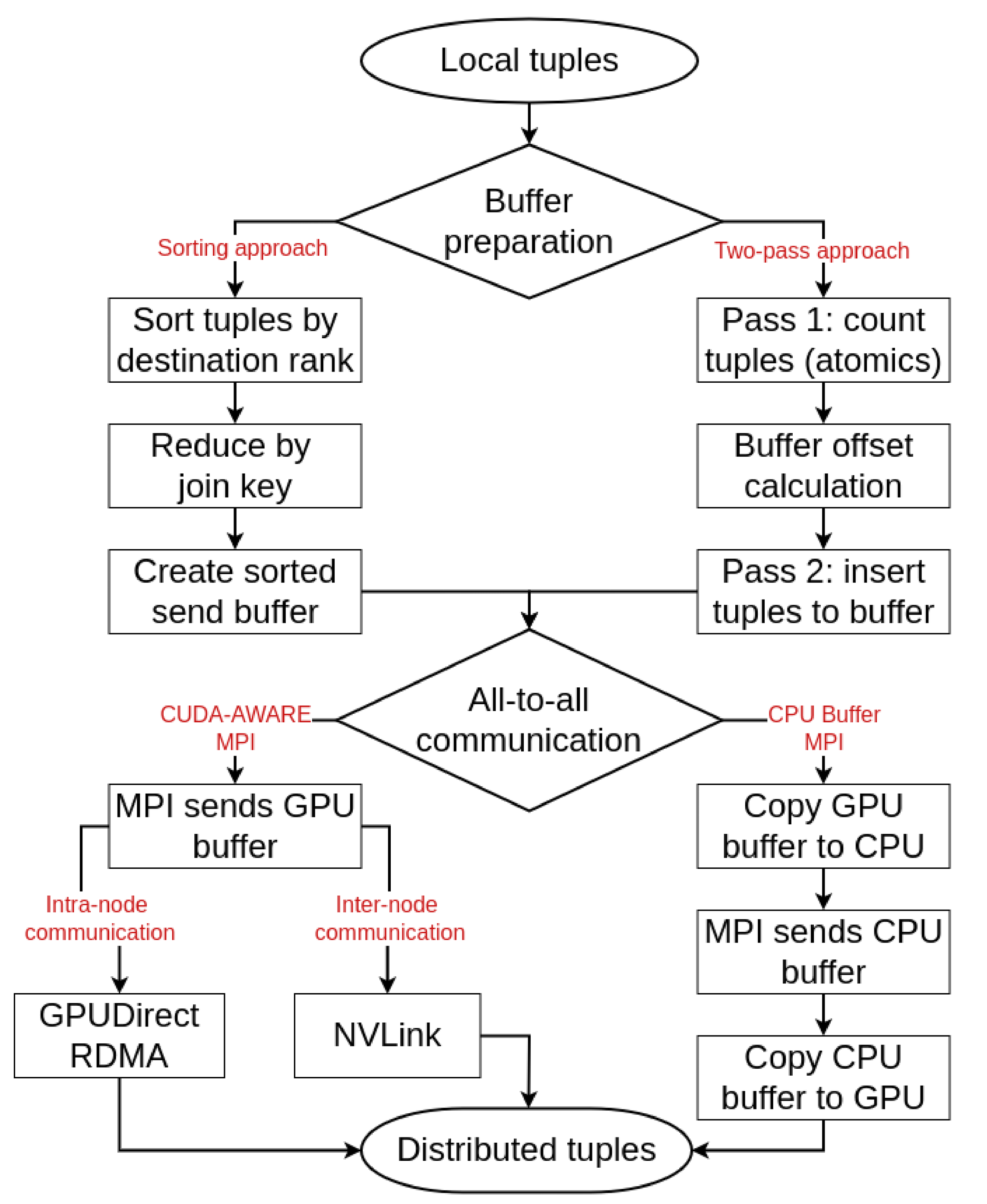


Radix-hash-based data partitioning (1)



*n = total gpus

Configurable all-to-all communication strategy (3)



Evolution from single-GPU to multi-node multi-GPU

Accelerating Datalog applications with cuDF	Towards iterative relational algebra on the {GPU}	Optimizing Datalog for the GPU	Multi-node multi-GPU Datalog
IA³ 2022	USENIX ATC '23	ASPLOS 2025	ICS

Experiments

We evaluate **MNMGDatalog** against state-of-the-art single-GPU, shared-memory, and distributed multi-node Datalog engines up to 32 NVIDIA A100 GPUs

Experiment platform, application, and datasets

- **Polaris** supercomputer from **Argonne National Lab**
- **CPU**: AMD EPYC 7543P processors with 32 cores
- **GPU**: 4 NVIDIA A100 GPUs per node interconnected by NVLink
- **Environment**: CUDA (12), SLOG(32 threads), Soufflé (128 threads)
- **Apps**: Transitive Closure, Same Generation, Connected Component
- **Datasets**: Stanford large network, SuiteSparse, Road network

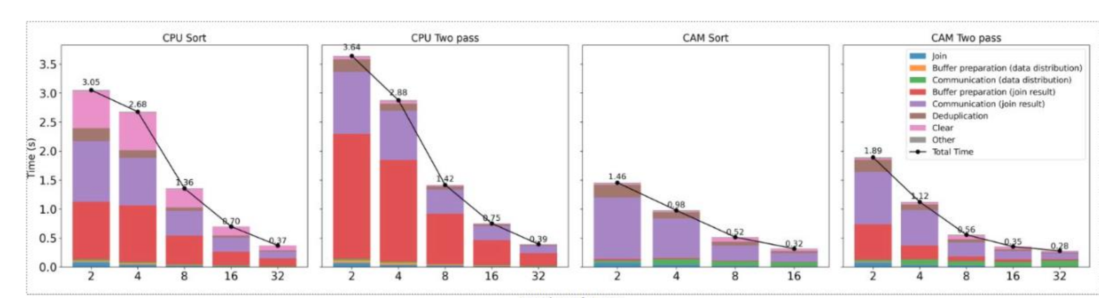
Single-GPU evaluation for Transitive Closure (TC)

Dataset name	*TC* edges	Time (s) MNMGDatalog	GPULog	Soufflé	GPUJoin
com-dblp	1.91B	**13.58**	26.95	232.99	OOM
fe_ocean	1.67B	**66.34**	72.74	292.15	100.30
usroads	871M	**75.07**	78.08	222.76	364.55
vsp_finan	910M	**81.14**	82.75	239.33	125.94
Gnutella31	884M	**4.75**	7.64	96.82	OOM

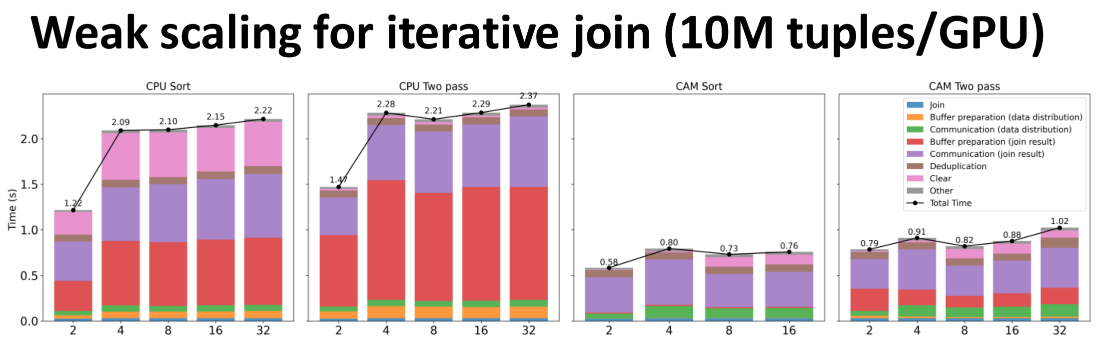
Single-GPU evaluation for Same Generation (SG)

Dataset name	*SG* size	Time (s) MNMGDatalog	GPULog	Soufflé	cuDF
fe_body	408M	**9.08**	18.41	74.26	OOM
loc-Brightkite	92.3M	**1.66**	11.67	48.18	OOM
fe_sphere	205M	**3.55**	7.88	48.12	OOM
CA-HepTH	74M	**0.60**	4.79	20.12	21.24

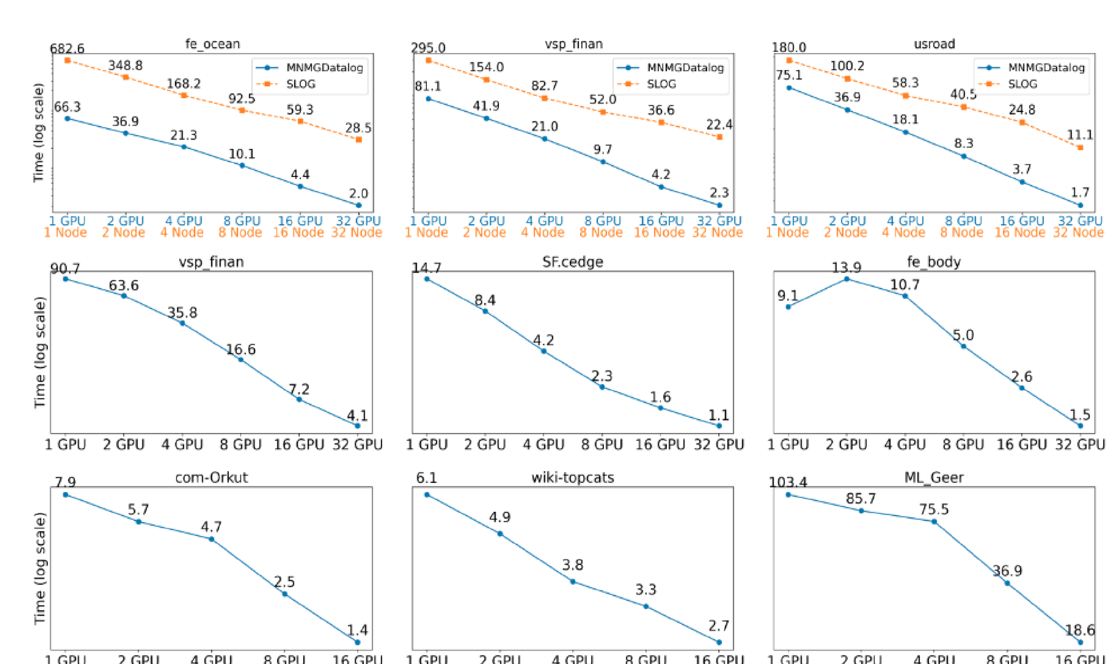
Strong scaling for iterative join (total 10M tuples)



Weak scaling for iterative join (10M tuples/GPU)



Multi-node evaluation for TC, SG, and CC



Conclusion

Our contributions:

- First ever Datalog engine designed for multi-node multi-GPU HPC systems, outperforming state-of-the-art shared-memory, distributed-memory, and GPU-based engines
- Introduces novel GPU-Aware communication and buffer preparation strategies for scalable recursive query evaluation
- Supports recursive aggregation for Datalog rules using high-throughput GPU kernels (Accepted at ICS 2025)

Acknowledgement

NSF · Argonne National Laboratory · The HARP Lab

Future plan

We are working on:

- Spatial and temporal load balancing
- GPU-Aware HIP and OneAPI implementations
- Application to Neurosymbolic programming