MapGraph

A High Level API for Fast Development of High Performance Graphic Analytics on GPUs

http://mapgraph.io

Zhisong Fu, Michael Personick and Bryan Thompson

SYSTAP, LLC

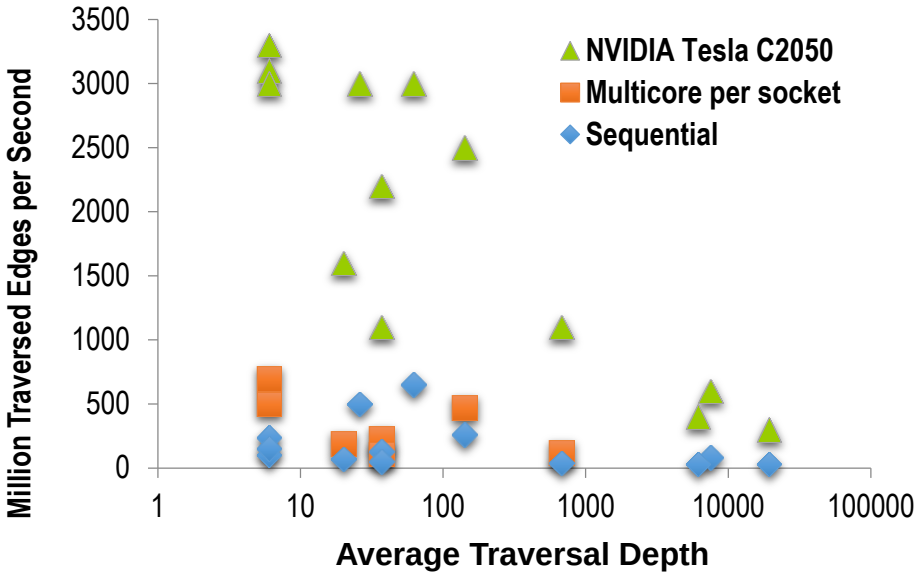
systap™

Outline

- Motivations
- MapGraph overview
- Results
- Summary

GPUs – A Game Changer for Graph Analytics?

- Graphs are everywhere in data, also getting bigger and bigger
- GPUs may be the technology that finally delivers real-time analytics on large graphs
  - 10x flops over CPU
  - 10x memory bandwidth
- This is a hard problem
  - Irregular memory access
  - Load imbalance
- Significant speed up over CPU on BFS [Merrill2013]
  - Over 10x speedup over CPU



Low-level VS. High-level

Low-level approach

- **BFS: [Merrill2013]**
- **PageRank: [Duong2012]**
- **SSSP: [Davidson2014]**

- **Pros: High performance**
- **Cons: Difficulty to develop**
  Reinvent the wheels

High-level approach

- **GraphLab [Low2012]**
- **Medusa [Zhong2013]**
- **Totem [Gharaibeh2013]**

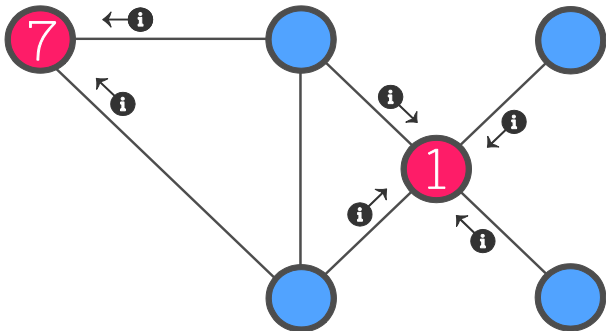
Pros: High programmability

Cons: Low Performance

MapGraph

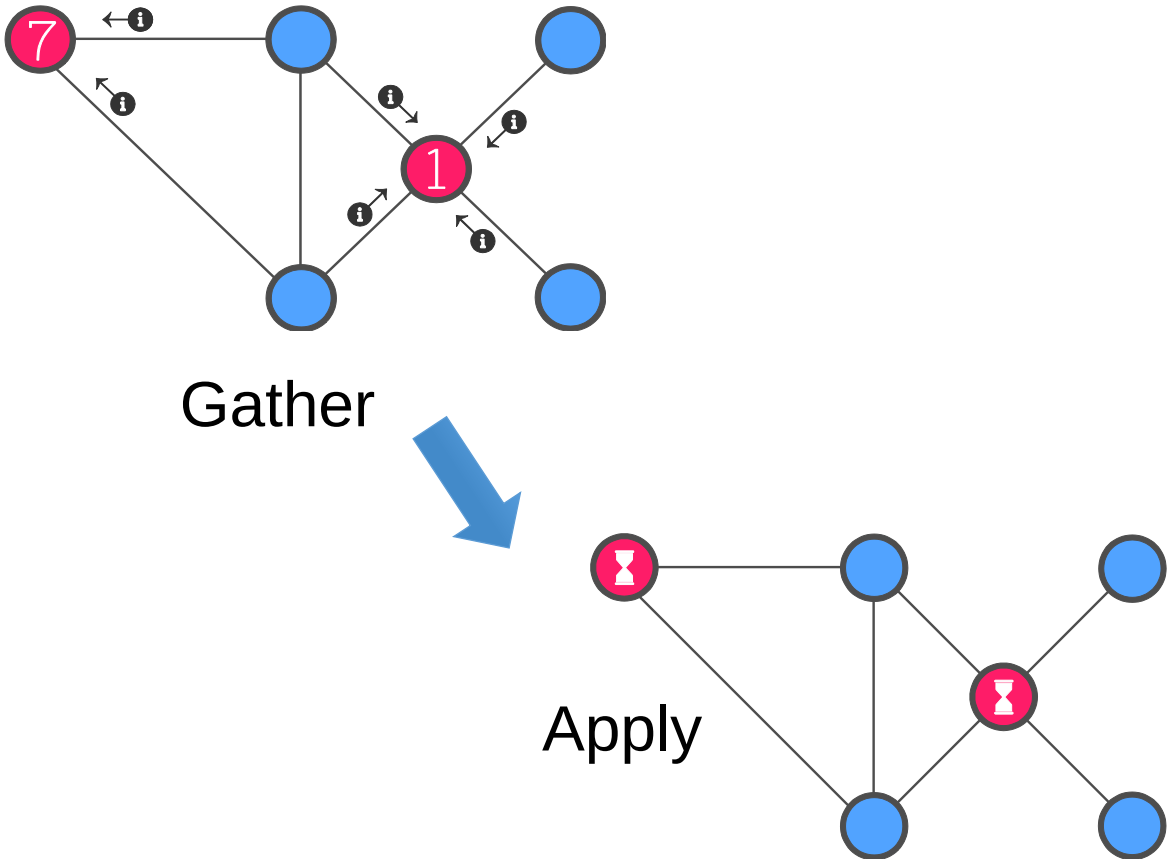
- High-level graph processing framework
  - High programmability: only C++ sequential
    - GPU architecture
    - Optimization techniques
    - CUDA, OpenCL
  - High performance
    - Comparable to low-level approach

GAS Abstraction

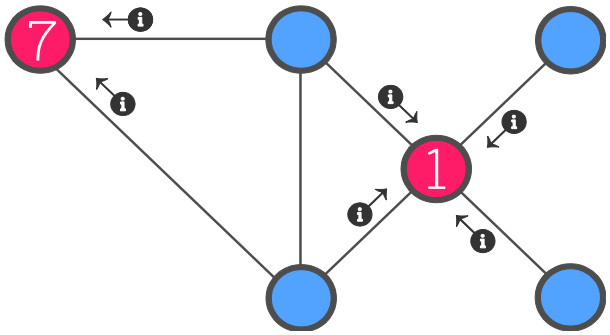


Gather

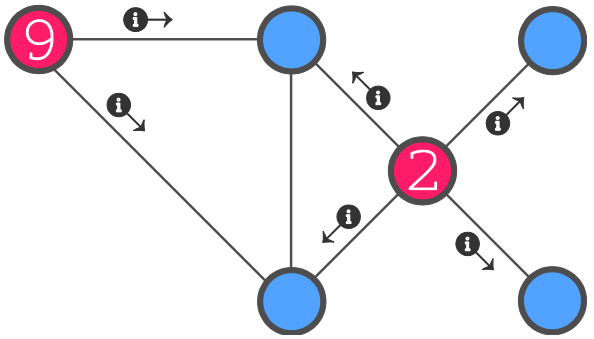
GAS Abstraction



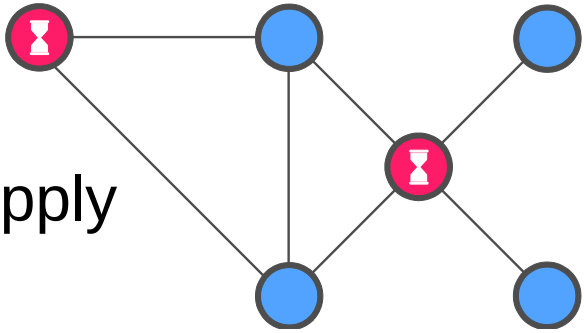
GAS Abstraction



Gather

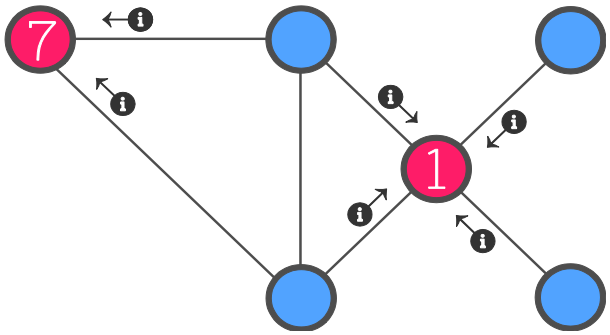


Scatter = Expand + Contract

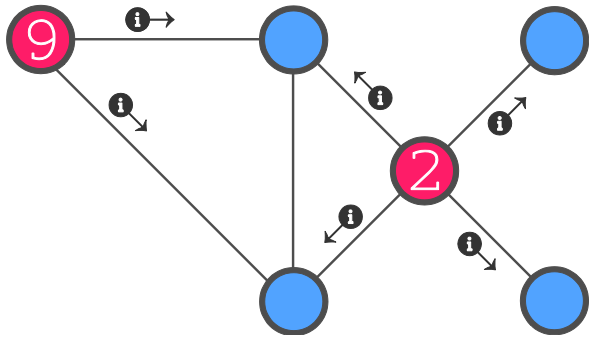


Apply

GAS Abstraction

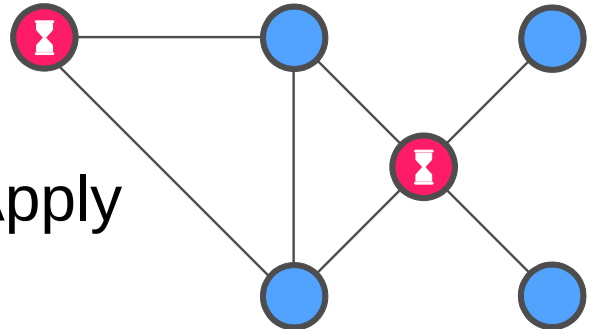


Frontier size > 0



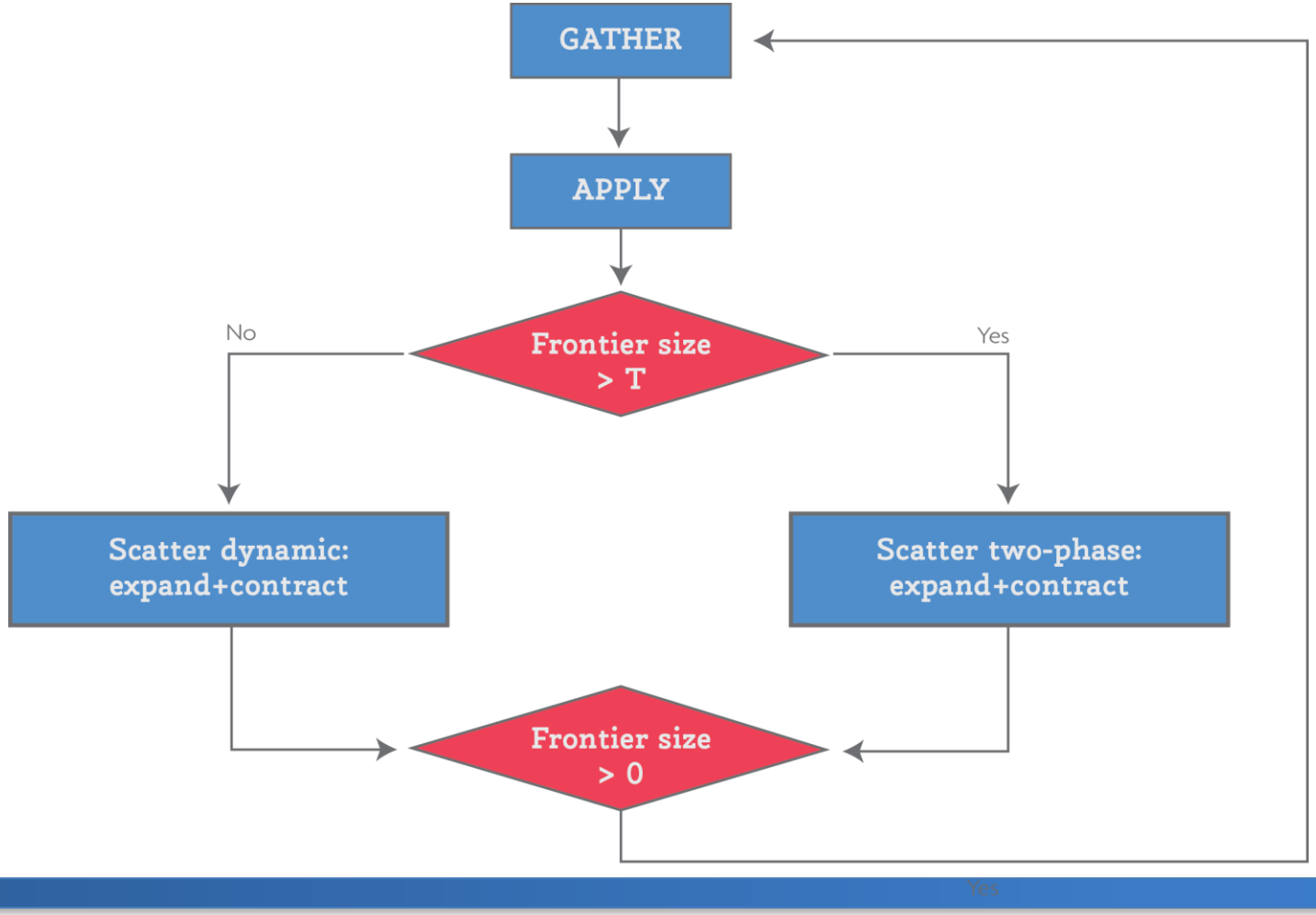
Gather

Scatter = Expand + Contract



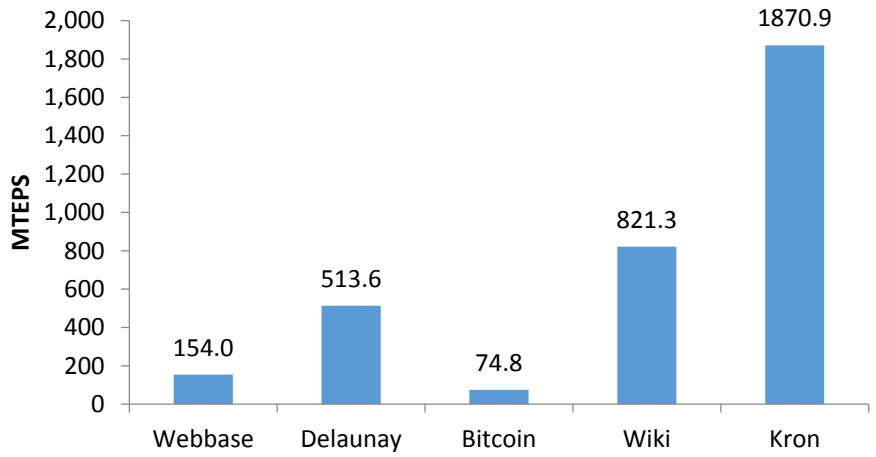
Apply

MapGraph Runtime Pipeline

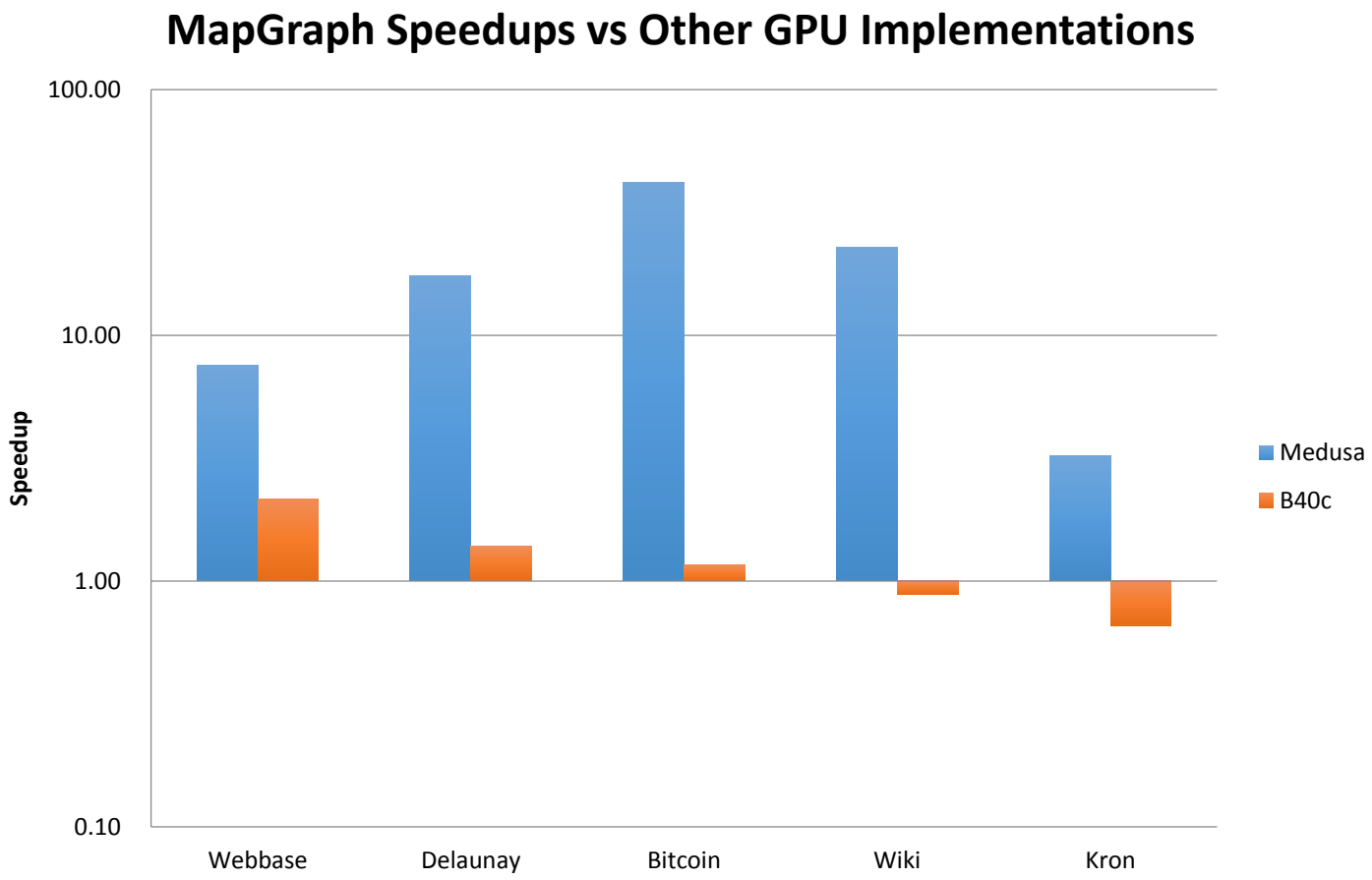


Experiment Datasets

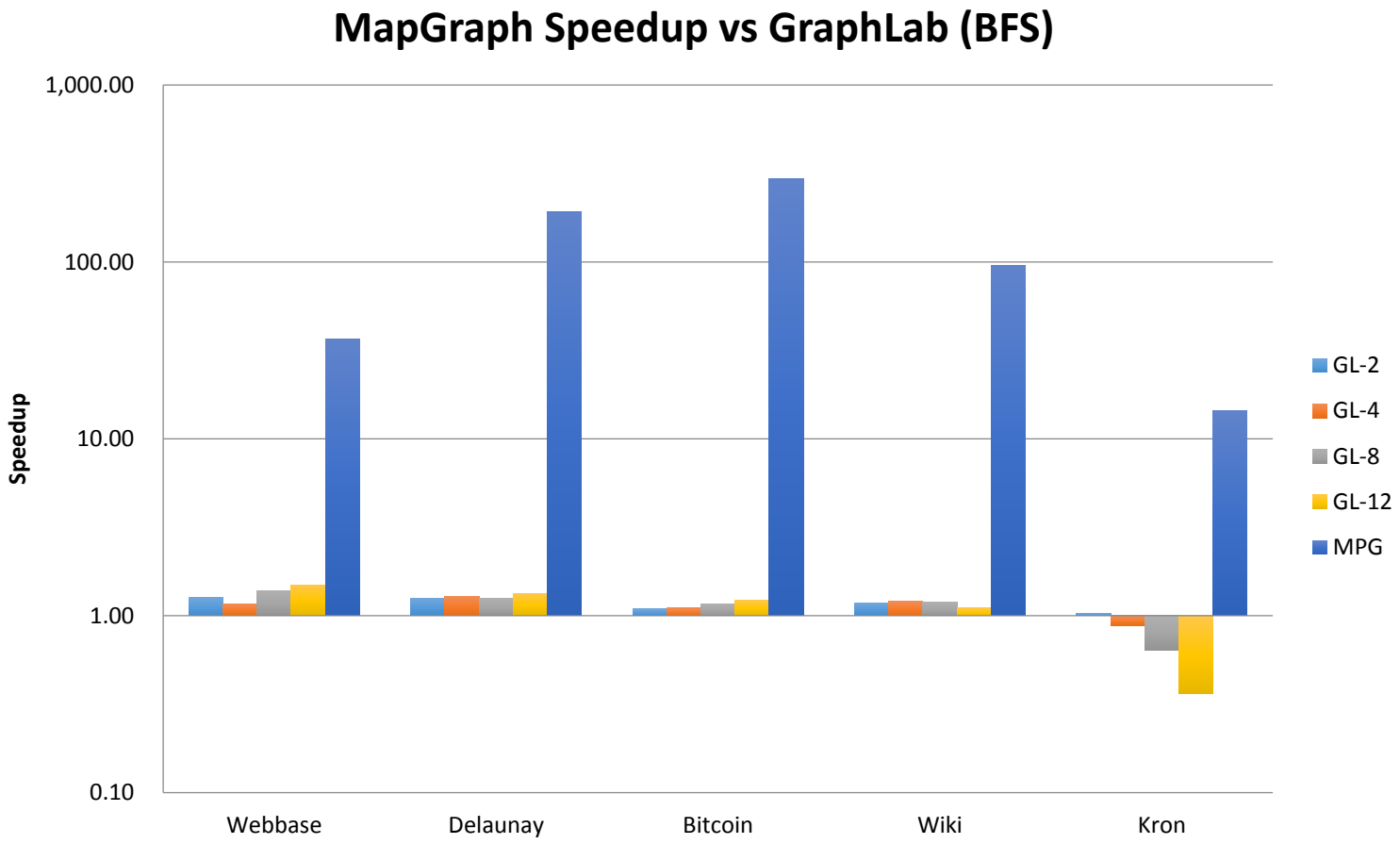
Dataset	#vertices	#edges	Max Degree	MTEPS (BFS)
Webbase	1,000,005	3,105,536	23	514
Delaunay	2,097,152	6,291,408	4,700	154
Bitcoin	6,297,539	28,143,065	4,075,472	75
Wiki	3,566,907	45,030,389	7,061	821
Kron	1,048,576	89,239,674	131,505	1,871



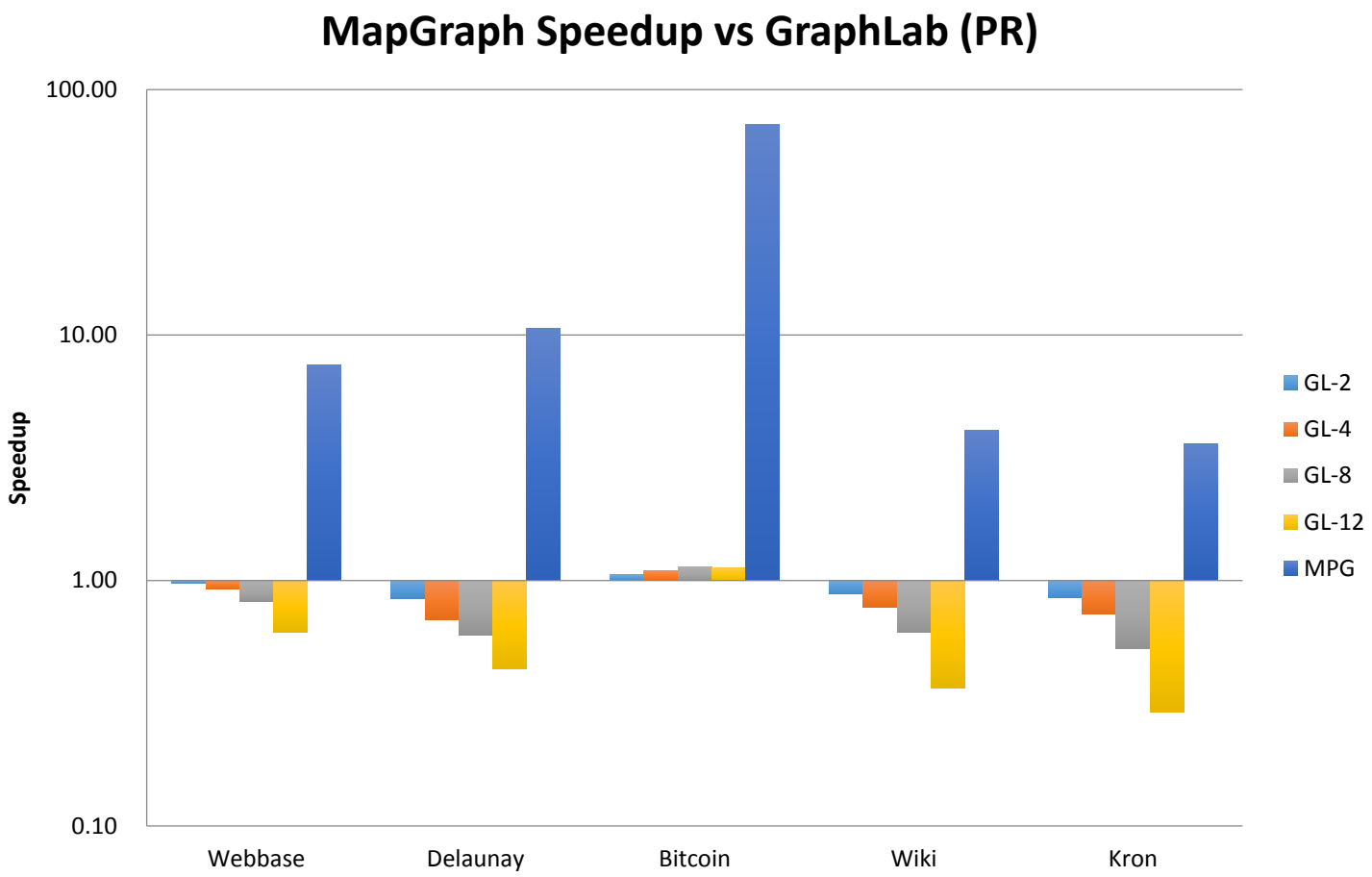
Results: Compare to Other GPU implementations



BFS Results: Compare to GraphLab

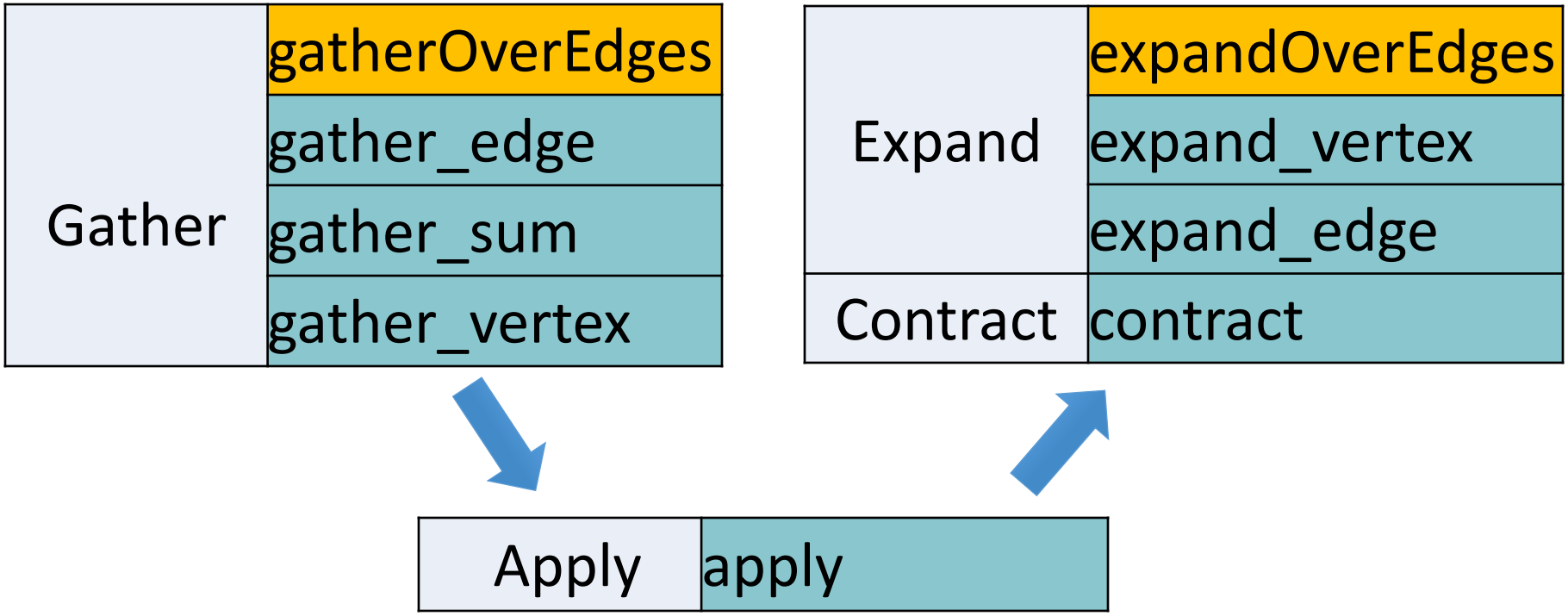


PageRank Results: Compare to GraphLab



MapGraph API

Scatter = Expand + Contract



Example: PageRank Implementation

- Gather, Apply, Scatter phases

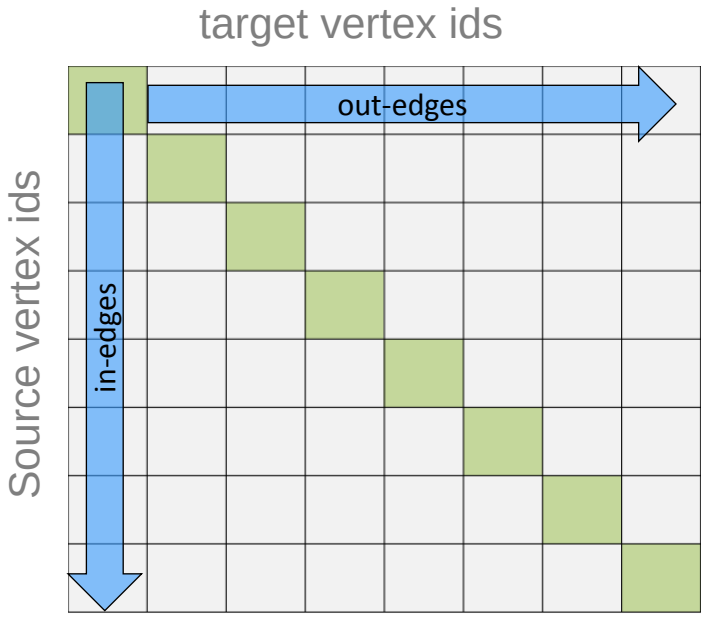
User Data	VertexType	float* d_ranks; int* d_num_out_edge;
Gather	gatherOverEdges	return GATHER_IN_EDGES;
	gather_edge	float nb_rank = d_dists[neighbor_id]; new_rank = nb_rank / d_num_out_edge[neighbor_id];
	gather_sum	return left + right;

Apply	apply	float old_value = d_ranks[vertex_id];
		float new_value = 0.15f + (1.0f - 0.15f) * gathervalue;
		changed = fabs(old_value – new_value) >= 0.01f;
		d_dists[vertex_id] = new_value;

Expand	expandOverEdges	return EXPAND_OUT_EDGES;
	expand_vertex	return changed;
	expand_edge	frontier = neighbor_id;

Future Work

- GPU cluster: 2D partitioning (aka vertex cuts)
  - In collaboration with SCI Institute of the University of Utah
  - Compute grid defined over virtual nodes.
  - Patches assigned to virtual nodes based on source and target identifier of the edge.
- Topology, message and data compression



Summary

- MapGraph: high-level graph processing framework
  - http://mapgraph.io
- High programmability:
  - GAS abstraction
  - Simple and flexible API
- High performance:
  - Hybrid scheduling strategy
  - Structure Of Arrays

Acknowledgement

- This work was (partially) funded by the DARPA XDATA program under AFRL Contract #FA8750-13-C-0002.

- This work is also supported by the DARPA under Contract No. D14PC00029.

- Many thanks to Dr. Christopher White for the support.