X-Stream: Edge-centric Graph Processing using Streaming Partitions

AMITABHA ROY, IVO MIHAILOVIC, WILLY ZWAENEPOEL

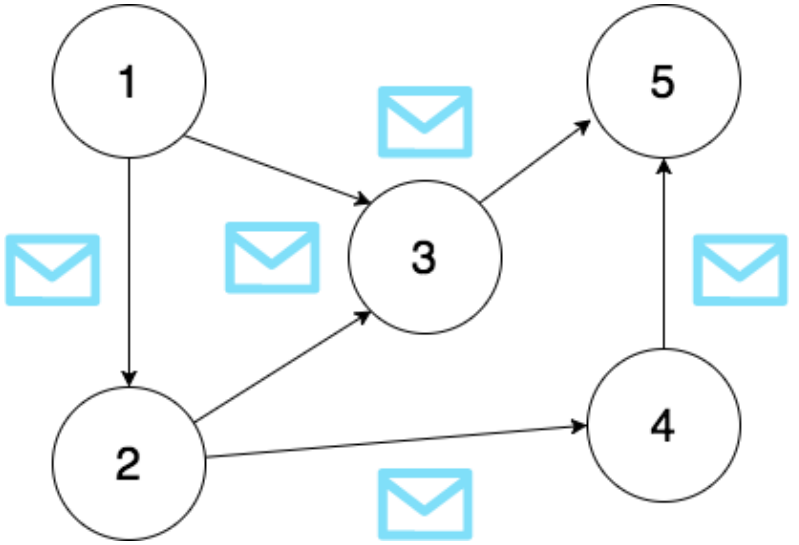
PRESENTED BY: MAREK STRELEC

Motivation

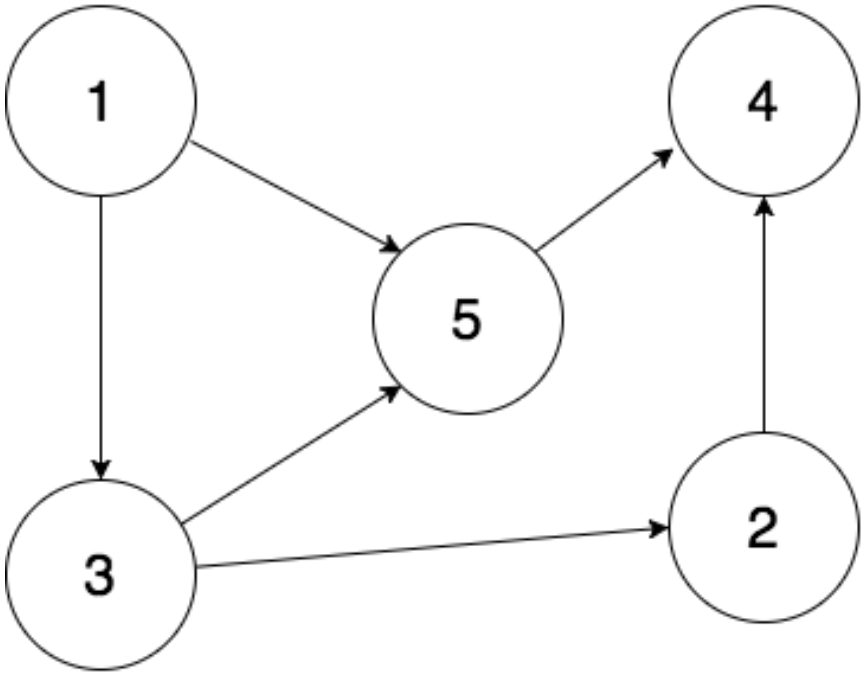
- Large graphs – billions of vertices and edges
- Process on large clusters
  - Pregel, GraphLab, PowerGraph, Niad
  - Complexity and cost
- Process on a single machine
  - GraphChi, X-Stream
- 64 GB RAM, 32 cores, 2 x 200 GB SSD, 3 x 3TB drive

Vertex-centric processing model

- "Think like a vertex"
- Popularized by the Pregel and GraphLab projects
- Mutable states stored in vertices
- Scatter-Gather model
  - Scatter updates along outgoing edges
  - Gather updates from incoming edges



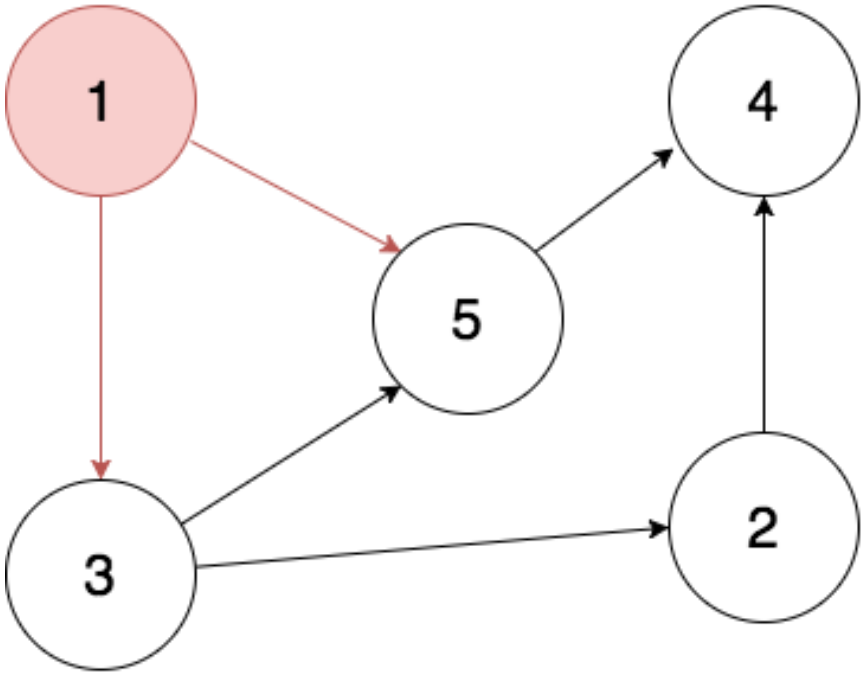
Vertex-centric BFS



Vertices
1
2
3
4
5

Source	Destination
1	3
1	5
2	4
3	2
3	5
5	4

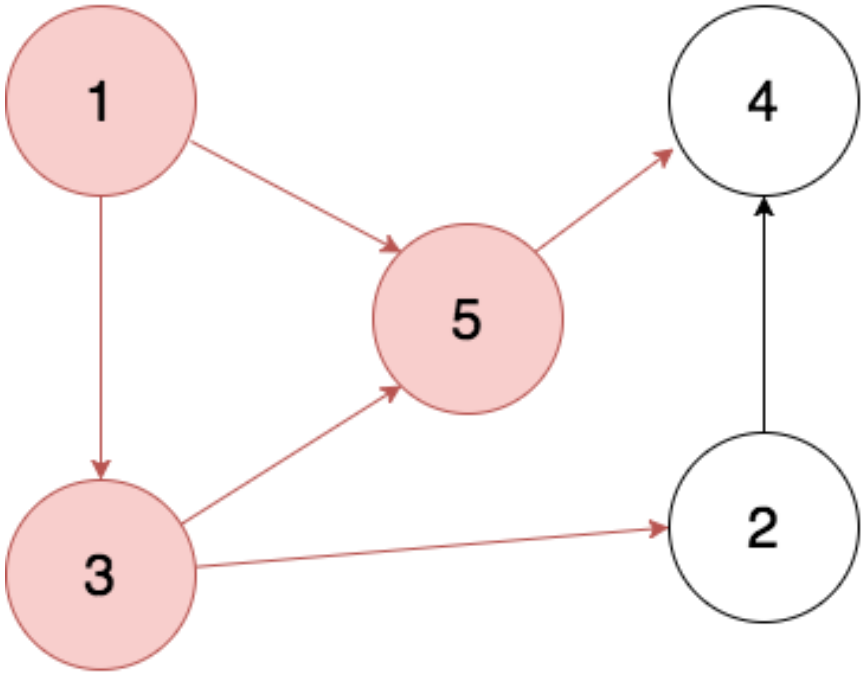
Vertex-centric BFS



Vertices
1
2
3
4
5

Source	Destination
1	3
1	5
2	4
3	2
3	5
5	4

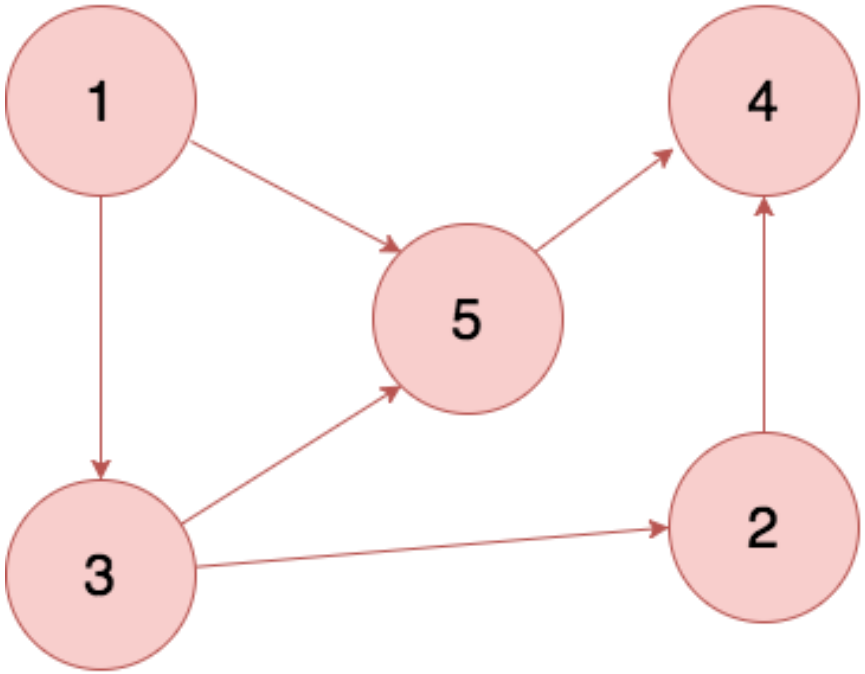
Vertex-centric BFS



Vertices
1
2
3
4
5

Source	Destination
1	3
1	5
2	4
3	2
3	5
5	4

Vertex-centric BFS



Vertices
1
2
3
4
5

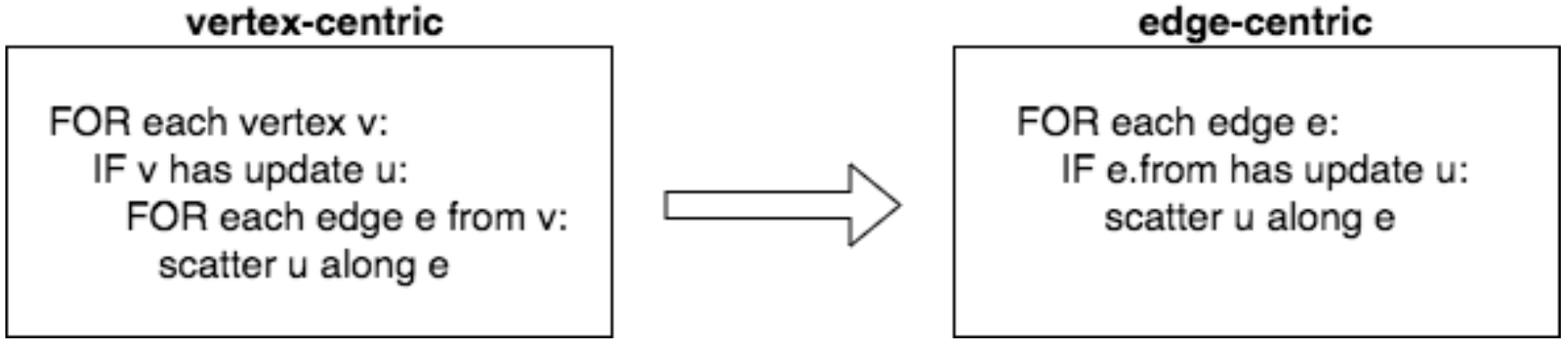
Source	Destination
1	3
1	5
2	4
3	2
3	5
5	4

Sequential vs. Random access

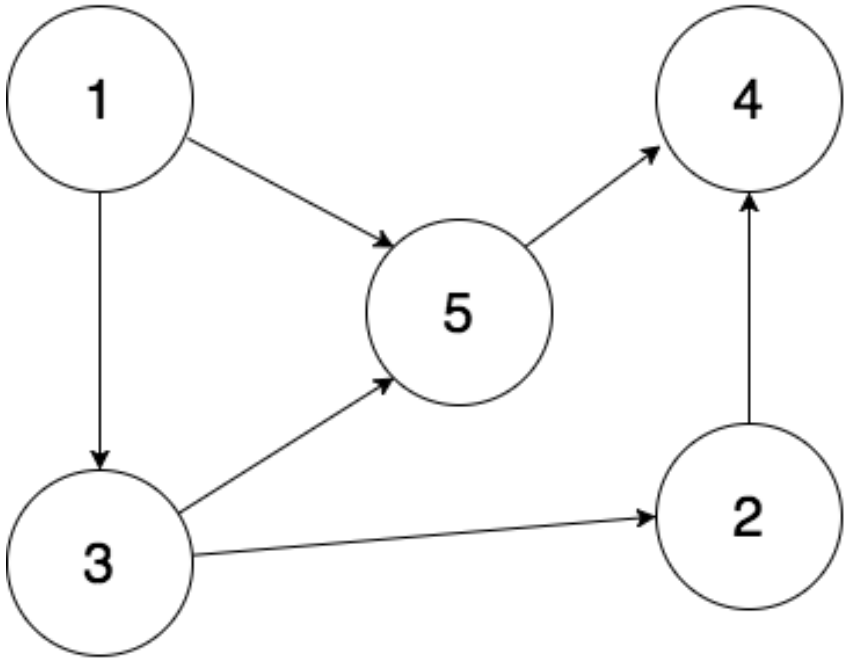
- Graph traversal = Random access
- For all storage media (RAM, SSD, and HDD)
  - Sequential bandwidth >> random access bandwidth
  - HDD - 300x higher
  - SSD - 30x higher
  - RAM (1 core) - 4.6x higher
  - RAM (16 cores) - 1.8x higher

X-stream processing model: Edge-centric

- Input to X-stream is an unordered set of directed edges
  - For undirected graphs - pair of directed edges
- Scatter and Gather phases iterate over ~~vertices~~ edges
- X-stream makes graph access sequential



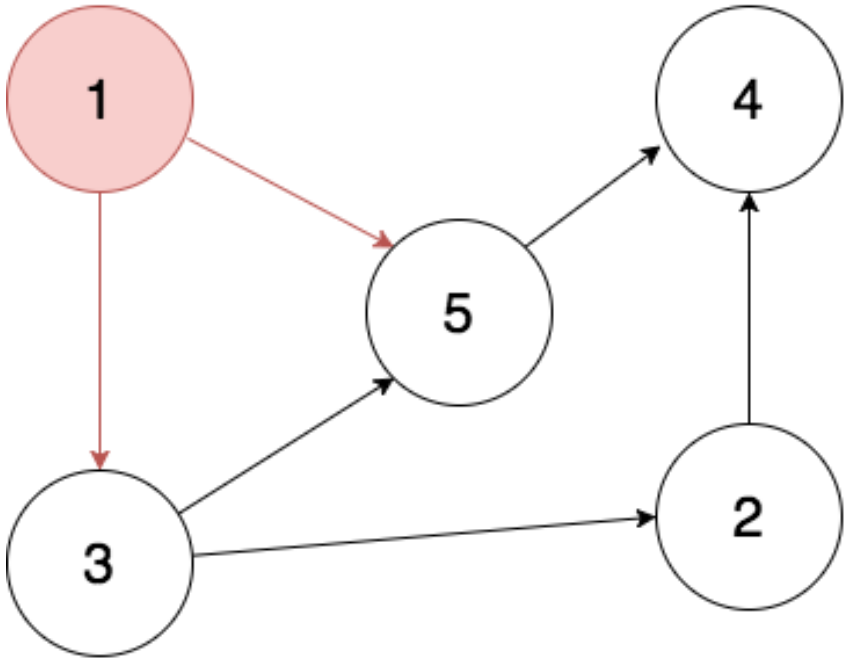
Edge-centric BFS



Vertices
1
2
3
4
5

Source	Destination
1	3
1	5
2	4
3	2
3	5
5	4

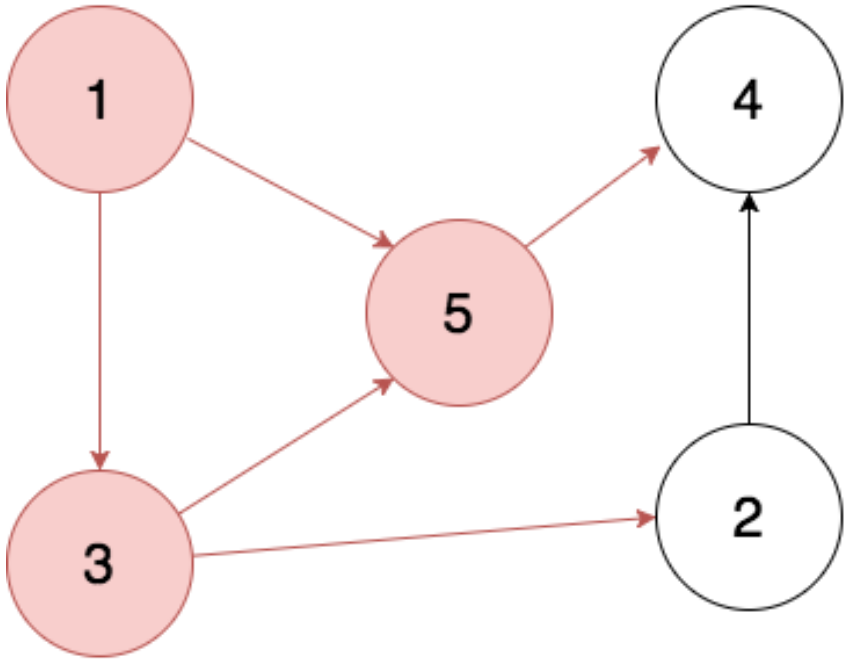
Edge-centric BFS



Vertices
1
2
3
4
5

Source	Destination
1	3
1	5
2	4
3	2
3	5
5	4

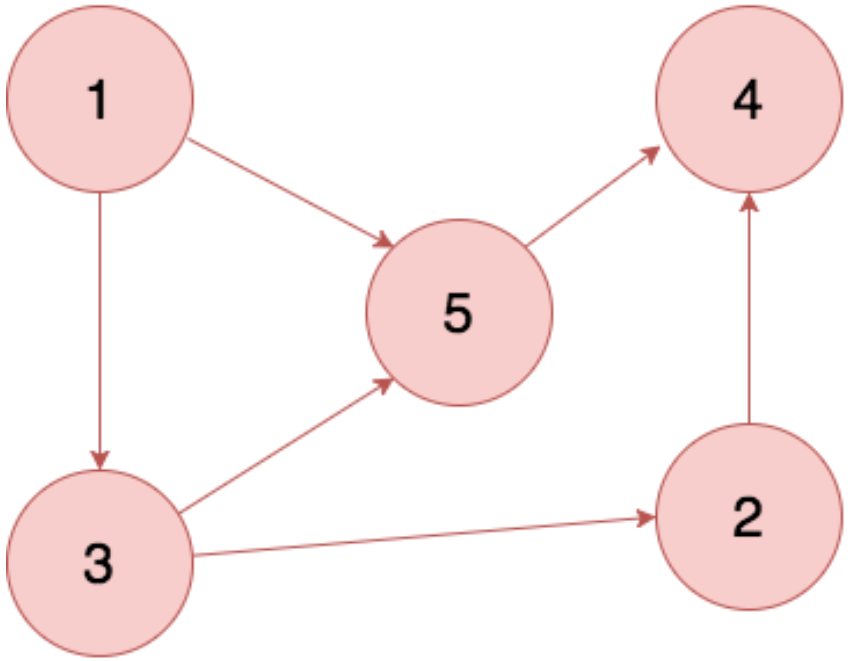
Edge-centric BFS

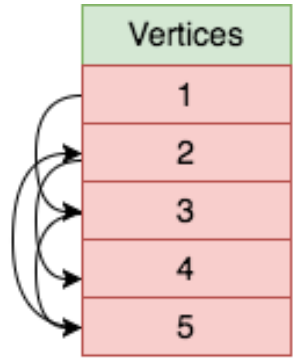


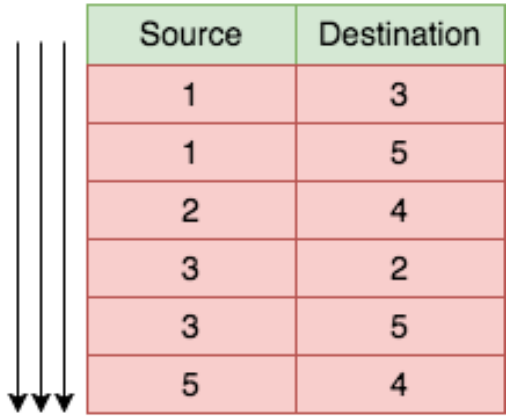
Vertices
1
2
3
4
5

Source	Destination
1	3
1	5
2	4
3	2
3	5
5	4

Edge-centric BFS







Source	Destination
1	3
1	5
2	4
3	2
3	5
5	4

Edge-centric properties

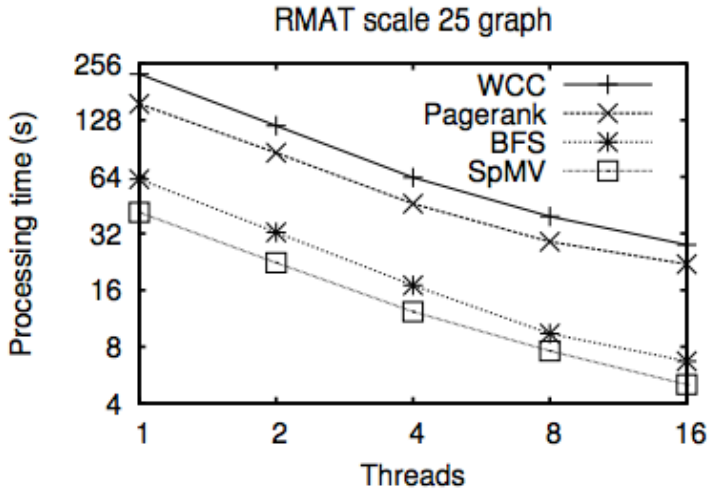
- Many sequential scans of the edge list
- The order of edges is irrelevant
- Tradeoff
  - Sequential access is faster
  - More Scatter/Gather iterations
- The number of iterations might be fever if the edge set >> vertex set
- Problem: still have random access to vertex set

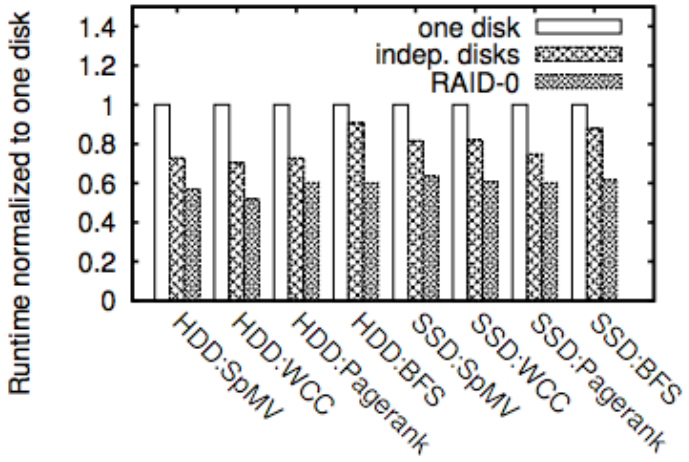
Streaming partitions

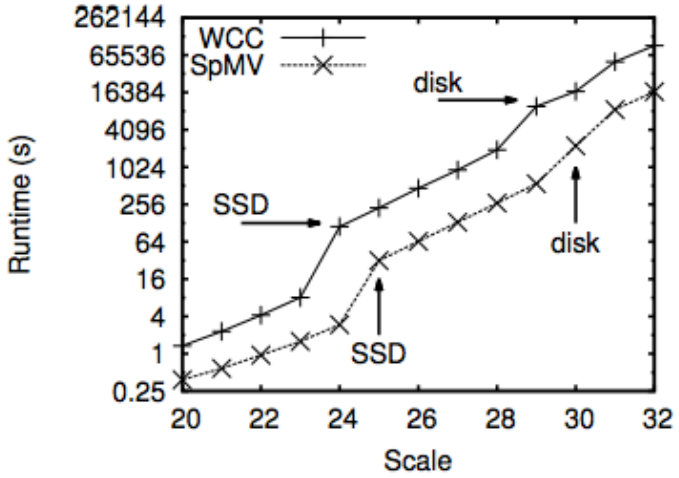
- Partition the graph into streaming partitions
  - vertex set: a subset of vertices that fit into RAM
  - edge list: all edges whose source vertex is in the partition's vertex set
  - update list: all updates whose destination vertex is in the partition's vertex set
- Streaming partitions can be processed in parallel
- Vertices (random access) => fast storage, Edges (sequential access) => slow storage
- The number of partitions is crucial for performance
- Shuffle phase - updates must be re-arranged after the scatter phase

Scalability

- Increasing thread count
- Increasing number of I/O devices
- Across devices







Traversal algorithms – BFS, WCC
Multiplication algorithms – PageRank, SpMW

Comparison with Other Systems: Ligra

- Ligra
  - In-memory graph processing system
  - Requires pre-processing

Threads	Ligra (s)	X-Stream (s)	Ligra-pre (s)
	BFS		
1	11.10	168.50	1250.00
2	5.59	86.97	647.00
4	2.83	45.12	352.00
8	1.48	26.68	209.40
16	0.85	18.48	157.20
	Pagerank		
1	990.20	455.06	1264.00
2	510.60	241.56	654.00
4	269.60	129.72	355.00
8	145.40	83.42	211.40
16	79.24	50.06	160.20

Comparison with Other Systems: GraphChi

- GraphChi
  - Traditional vertex-centric approach
  - Out-of-core data structure, parallel sliding windows, to reduce the amount of random access to disk
  - needs time to pre-sort the graph into shards

	Pre-Sort (s)	Runtime (s)	Re-sort (s)
Twitter pagerank			
X-Stream (1)	*none*	397.57 ± 1.83	–
Graphchi (32)	752.32 ± 9.07	1175.12 ± 25.62	969.99
Netflix ALS			
X-Stream (1)	*none*	76.74 ± 0.16	–
Graphchi (14)	123.73 ± 4.06	138.68 ± 26.13	45.02
RMAT27 WCC			
X-Stream (1)	*none*	867.59 ± 2.35	–
Graphchi (24)	2149.38 ± 41.35	2823.99 ± 704.99	1727.01
Twitter belief prop.			
X-Stream (1)	*none*	2665.64 ± 6.90	–
Graphchi (17)	742.42 ± 13.50	4589.52 ± 322.28	1717.50

System	Graphchi (shard)	Graphchi (run)	X-Stream
	LABOS		
Intel SSDs	486 ± 6.762	908.966 ± 16.667	417.213 ± 3.037
Disk	591.848 ± 19.885	1507 ± 13.656	616.795 ± 2.271
	Cambridge		
Samsung 840	389.569 ± 41.879	943.246 ± 19.754	588.613 ± 5.259
2xSamsung 840	375.729 ± 35.975	811.359 ± 23.706	443.396 ± 40.446
OCZ Vertex	423.104 ± 5.218	1079.138 ± 20.600	843.023 ± 276.625
Disk	590.584 ± 55.165	1879 ± 93.368	1613.174 ± 106.151

Table 2: Results for pagerank

Criticism

- Assumes that the number of edges is larger than the number of vertices
- Performs well only on graphs with a low diameter
- Workload imbalance as the partitions can have different numbers of edges assigned to them
  - Is work stealing sufficient?

Thank you!