Wexner Medical Center

STREAMER: a Distributed Framework for Incremental Closeness Centrality Computation

A. Erdem Sarıyüce [1,2], Erik Saule [4], Kamer Kaya [1], Ümit V. Çatalyürek [1,3]

[1] Department of Biomedical Informatics

[2] Department of Computer Science & Engineering

[3] Department of Electrical & Computer Engineering

The Ohio State University

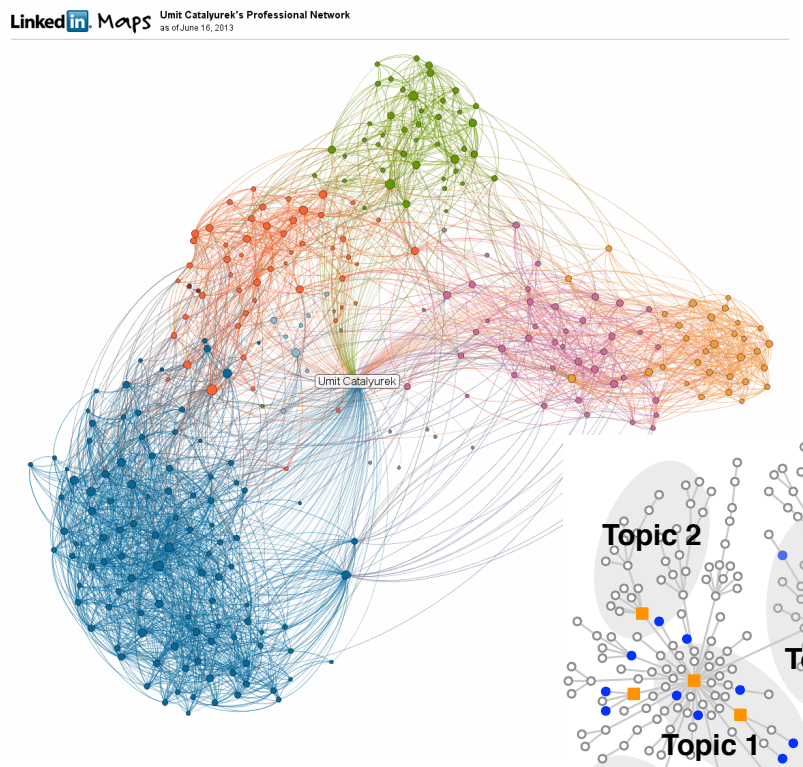
[4] Department of Computer Science

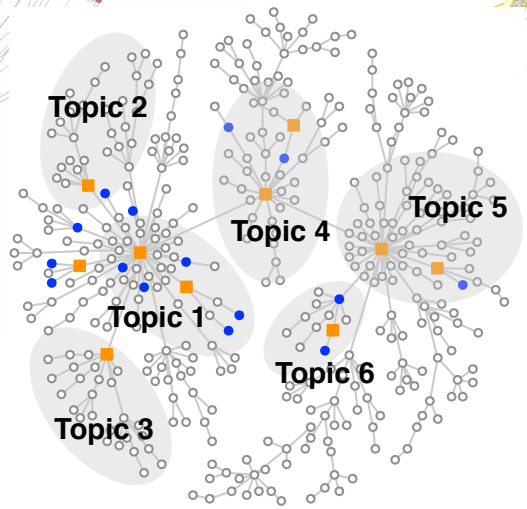
University of North Carolina Charlotte

IEEE Cluster 2013, Indianapolis, IN

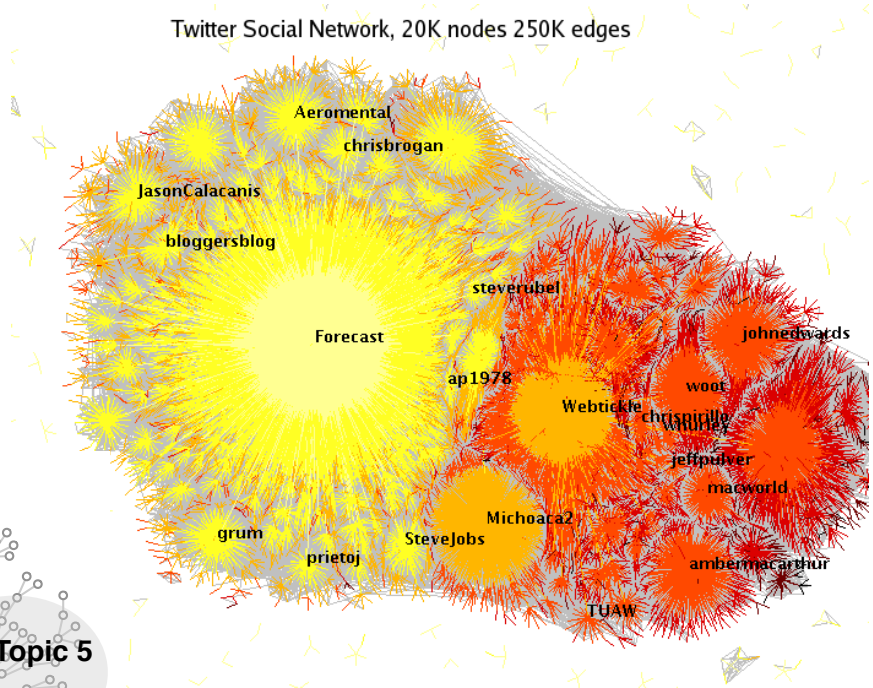
Massive Graphs are everywhere

- Facebook has a billion users and a trillion connections
- Twitter has more than 200 million users



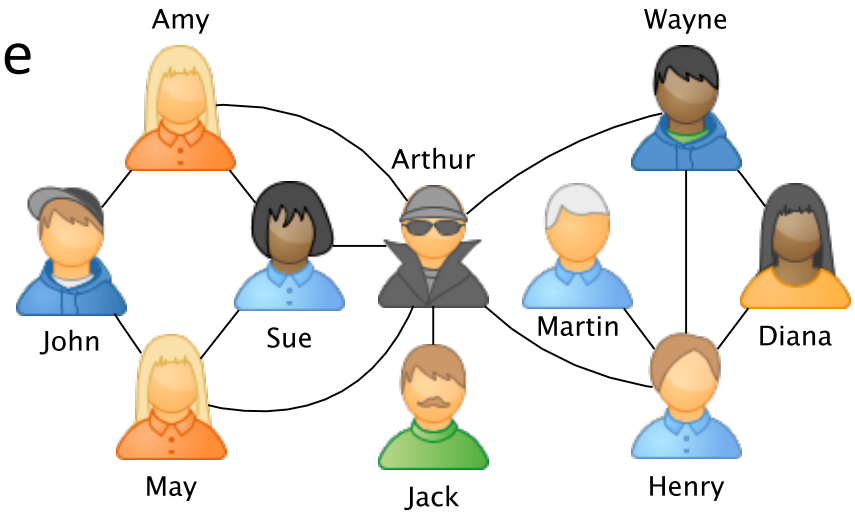


citation graphs



Large(r) Networks and Centrality

- Who is more important in a network? Who controls the flow between nodes?
  - Centrality metrics answer these questions
  - Closeness Centrality (CC) is an intriguing metric



- How to handle changes?
  - Incremental algorithms are good but not enough in practice
  - Parallelism is essential

Closeness Centrality

Wexner Medical Center

- Let *G=(V, E)* be a graph with vertex set *V* and edge set *E*
  - Farness (far) of a vertex is the sum of shortest distances to each vertex

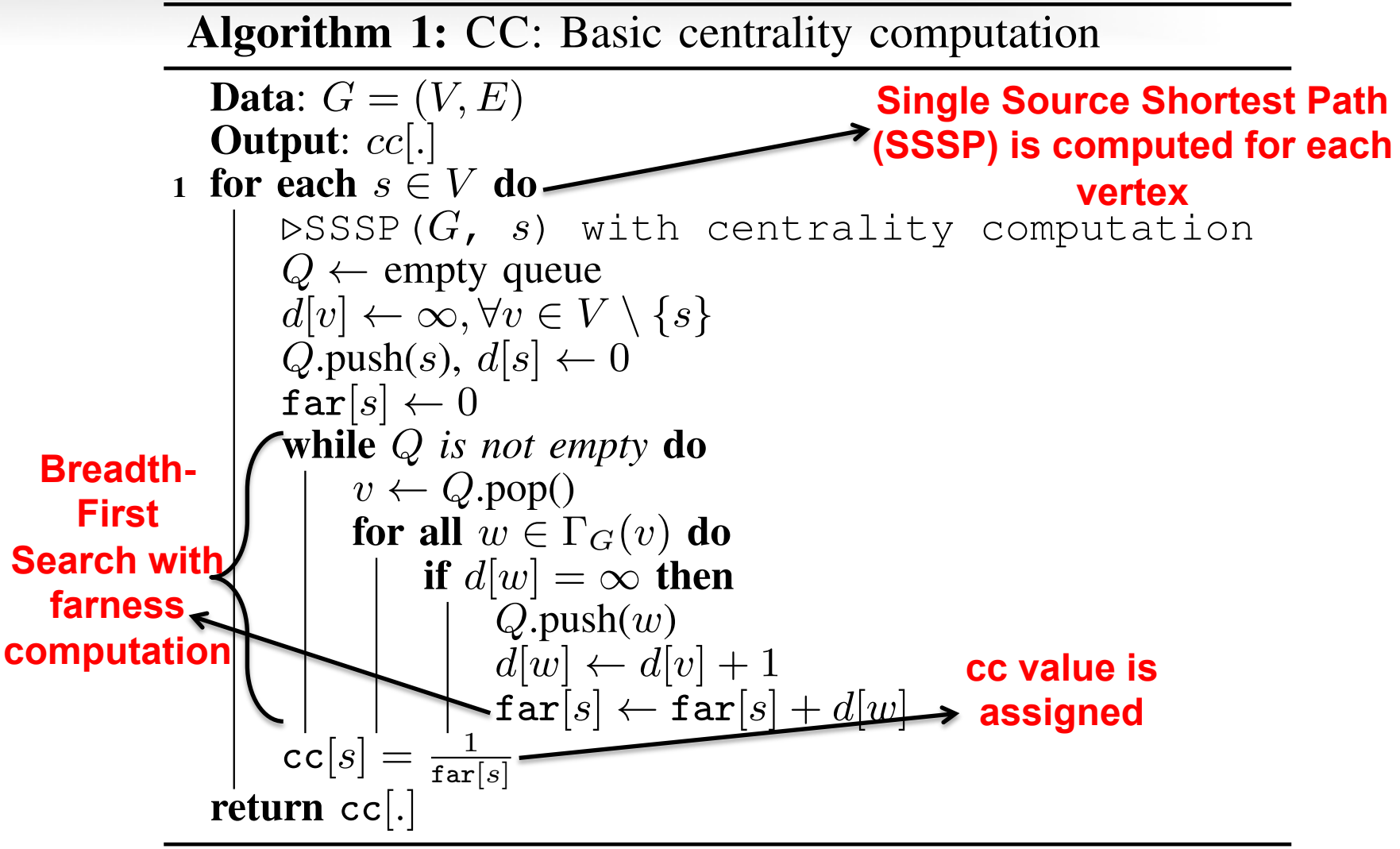
$$\texttt{far}[u] = \sum_{\substack{v \in V \\ d_G(u,v) \neq \infty}} d_G(u, v)$$

  - Closeness centrality (cc) of a vertex :

$$\texttt{cc}[u] = \frac{1}{\texttt{far}[u]}$$

- Best algorithm: All-pairs shortest paths
  - O(|V|.|E|) complexity for unweighted networks
- For large and dynamic networks
  - From scratch computation is infeasible
  - Faster solutions are essential

CC Algorithm



Incremental Closeness Centrality

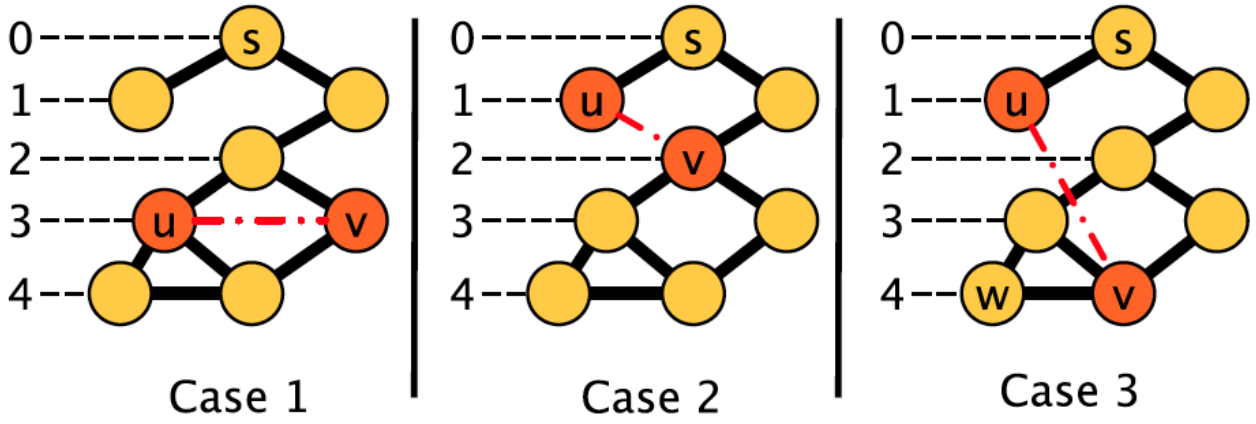
- Computing cc values from scratch after each edge change is very costly
  - Incremental algorithms are used to handle changes
  - Main idea is to reduce number of SSSPs to be executed

- Three filtering techniques are proposed
  - Filtering with level differences
  - Filtering with biconnected components
  - Filtering with identical vertices
- Details can be found at

  "A. E. Sarıyuce, K. Kaya, E. Saule, and Umit V. Catalyurek. Incremental algorithms for Closeness Centrality. IEEE BigData Conference, 2013"

Filtering with level differences

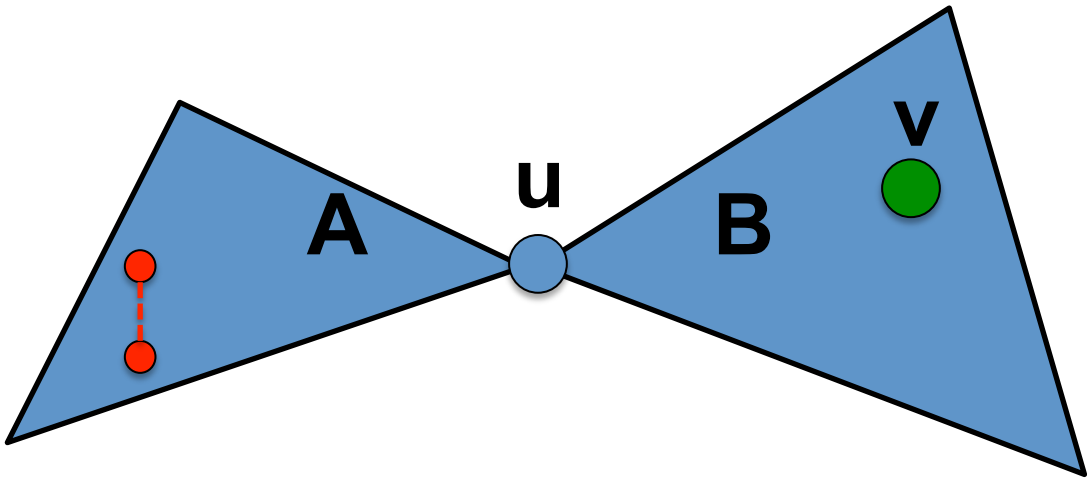
- Upon edge insertion, breadth-first search tree of each vertex will change. Three possibilities:



- Case 1 and 2 will not change cc of s!
  - No need to apply SSSP from them
- Just Case 3
  - BFSs are executed from u and v and level diff is checked

Filtering with biconnected components

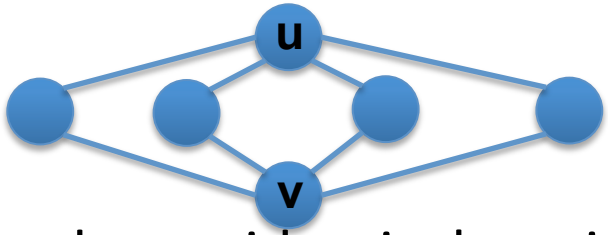
- What if the graph have articulation points?



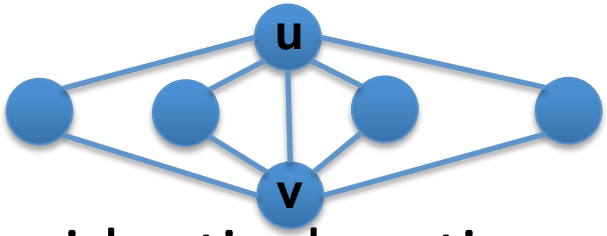
- Change in A can change cc of any vertex in A and B
- Computing the change for u is **enough** for finding changes for any vertex v in B (constant factor is added)

Filtering with identical vertices

- Two types of identical vertices:
  - Type I: u and v are identical vertices if N(u) = N(v), i.e., their neighbor lists are same



  - Type II: u and v are identical vertices if {u} U N(u) = {v} U N(v), i.e., they are also connected



- If u and v are identical vertices, their cc are the same
  - Same breadth-first search trees!

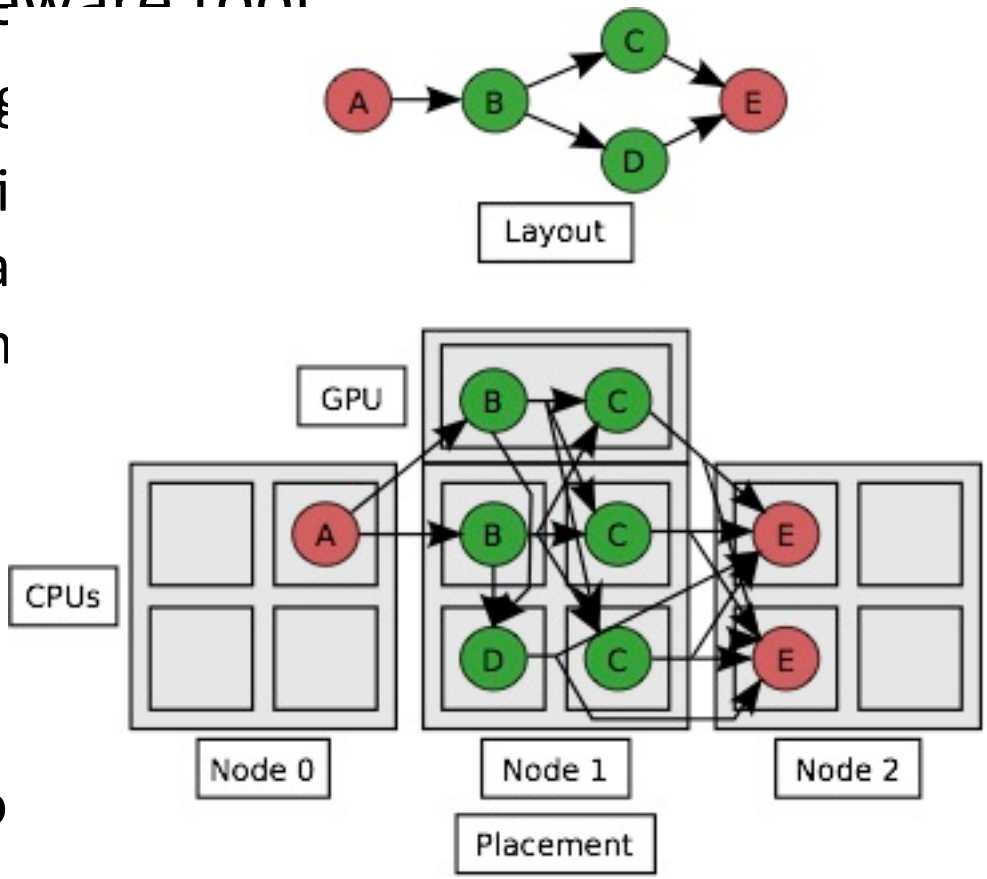
Is it enough?

name	$\|V\|$	$\|E\|$	**Time** (in sec.)
web-NotreDame	325K	1,090K	53.0
amazon0601	403K	2,443K	298.1
web-Google	875K	4,322K	824.4

- Too slow for real-time processing
- The problem is mostly parallel and graphs are relatively small.
  - Source-level parallelism can be used to fill up a cluster

DataCutter

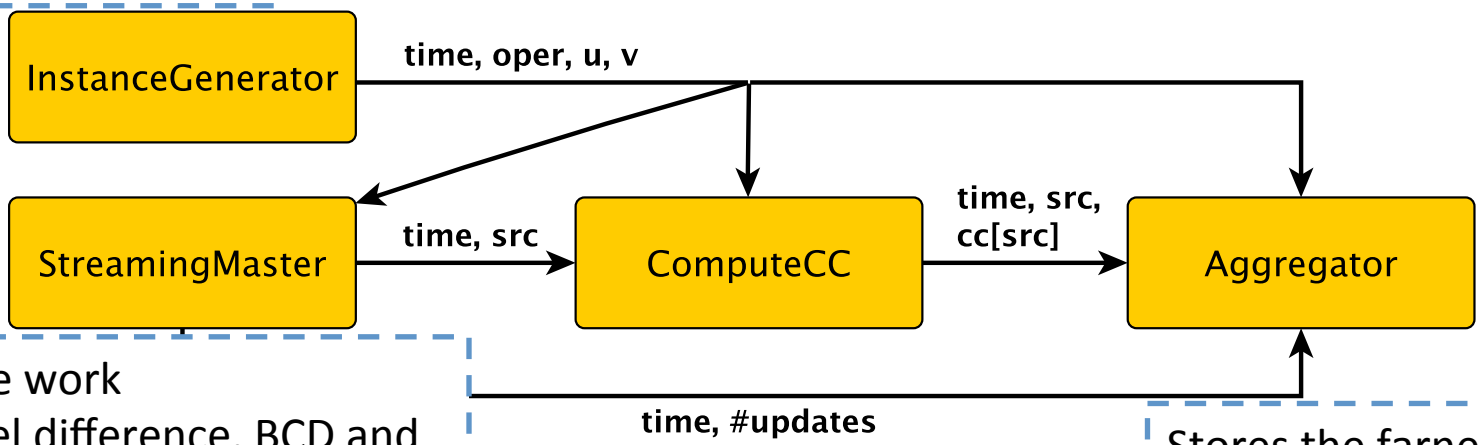
- Component-based middleware tool
  - Supports filter-stream prog
  - Implements the computati of components (filters) tha data through logical strean (unidirectional data flows)
- Layout is a filter ontology
  - Describes the set of tasks, streams and the connectio
  - All replicable



STREAMER Framework

Wexner Medical Center

Sends the updates on the graph to everyone



Filters the work using level difference, BCD and identical vertices

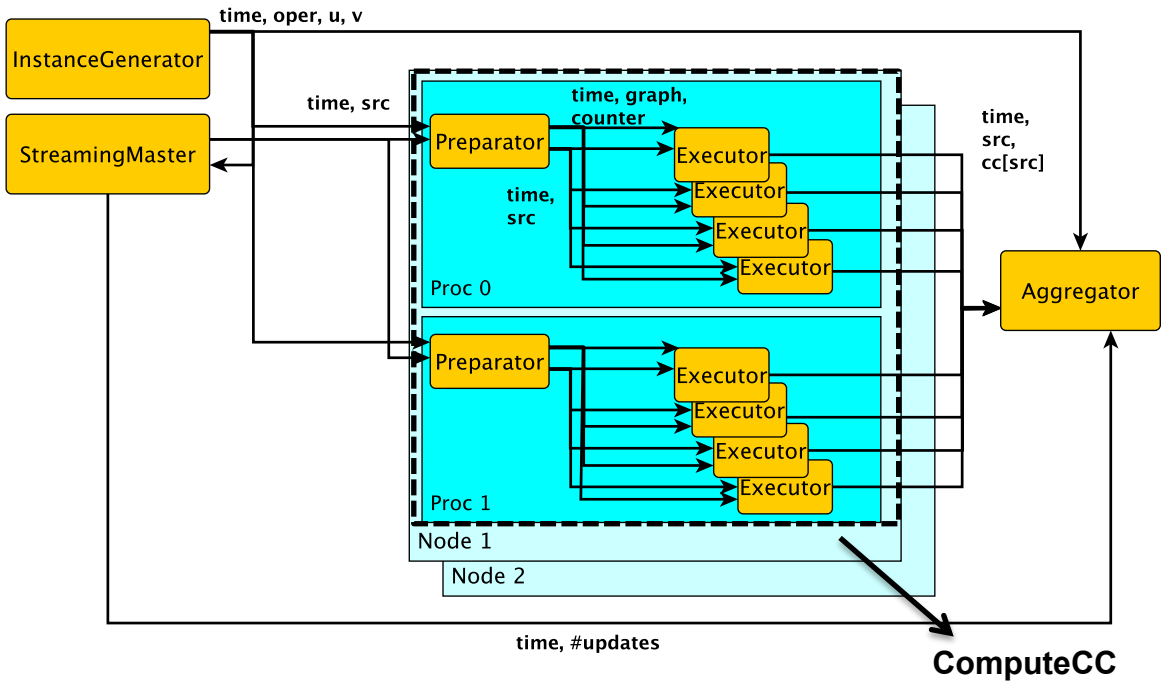
- List of vertices needing SSSP update are sent to ComputeCC
- # of updates are sent to Aggregator

Computes the farness of the vertex

- Most expensive part
- Most replicated part

Stores the farness values of all vertices and does adjustments to identical vertices and biconnected components

Multicore architecture and NUMA effects



- Preparator makes the actual graph
- Pointers are shared between Executors

Experiments

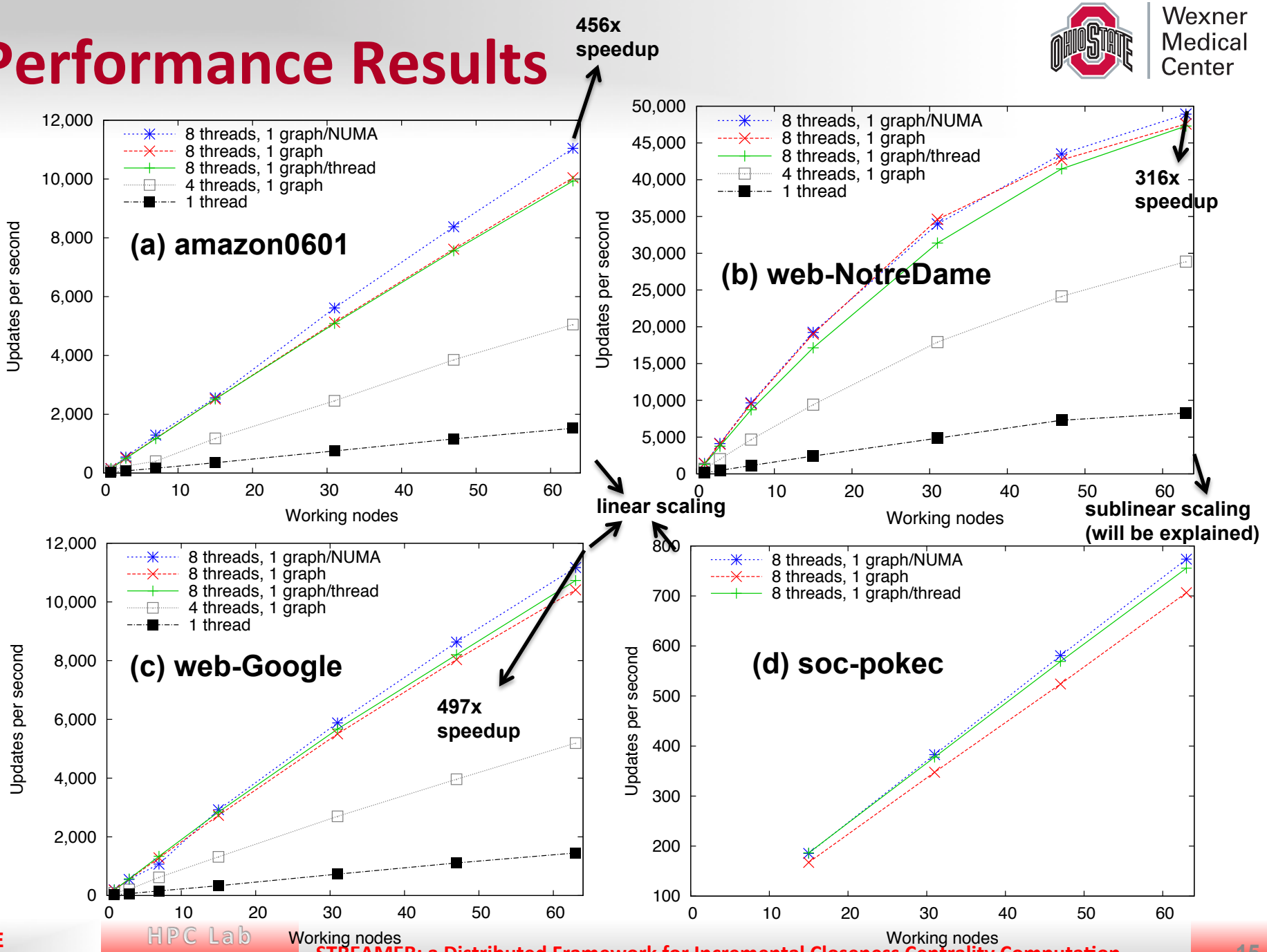
Wexner Medical Center

- Dataset

Name	$\|V\|$	$\|E\|$	% of computation saved
web-NotreDame	325,729	1,090,008	97.5
amazon0601	403,394	2,443,308	92.3
web-Google	916,428	4,321,958	94.4
soc-pokec	1,632,804	30,622,464	93.9

- 64 node cluster
  - Each with dual Intel Xeon E5520 Quad-Core processor
  - 8MB L3 cache per processor
  - 48GB main memory, 20Gbps Infiniband Connection
  - Compiled with GCC 4.5.2 with –O3 flag

Performance Results



NUMA awareness

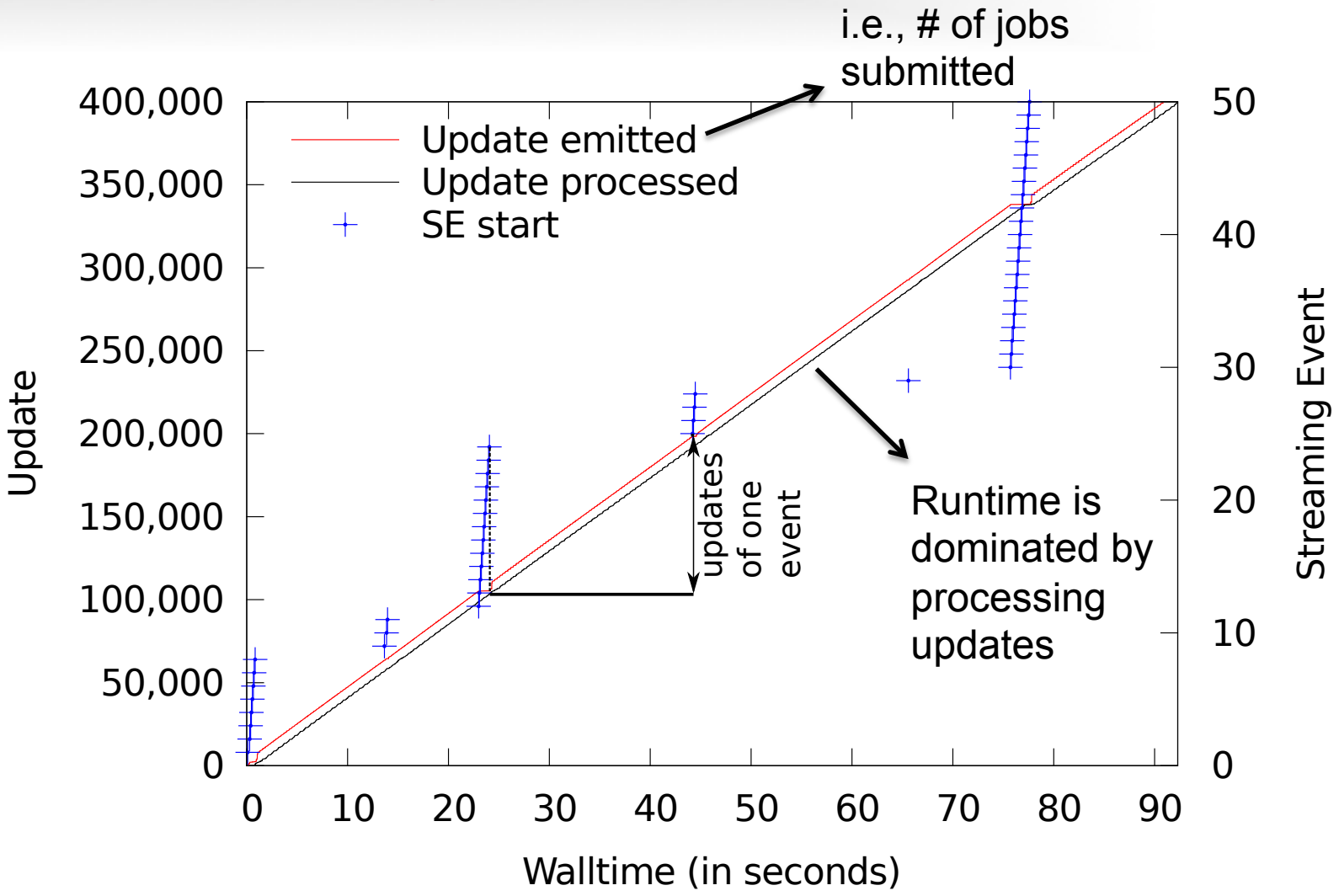
Wexner Medical Center

THE PERFORMANCE OF STREAMER WITH 31 WORKER NODES AND DIFFERENT NODE-LEVEL CONFIGURATIONS NORMALIZED TO 1 THREAD CASE (PERFORMANCE ON `soc-pokec` IS NORMALIZED TO 8 THREADS, 1 GRAPH/THREAD). THE LAST COLUMN IS THE ADVANTAGE OF SHARED MEMORY AWARENESS (RATIO OF COLUMNS 5 AND 3).

Name	4 threads	8 threads, 1 graph per			Shared Mem. awareness
		thread	node	NUMA	
web-NotreDame	3.69	6.46	7.13	6.99	1.08
amazon0601	3.26	6.75	6.81	7.45	1.10
web-Google	3.69	7.77	7.55	8.06	1.03
soc-pokec	-	1.00	0.92	1.01	1.01

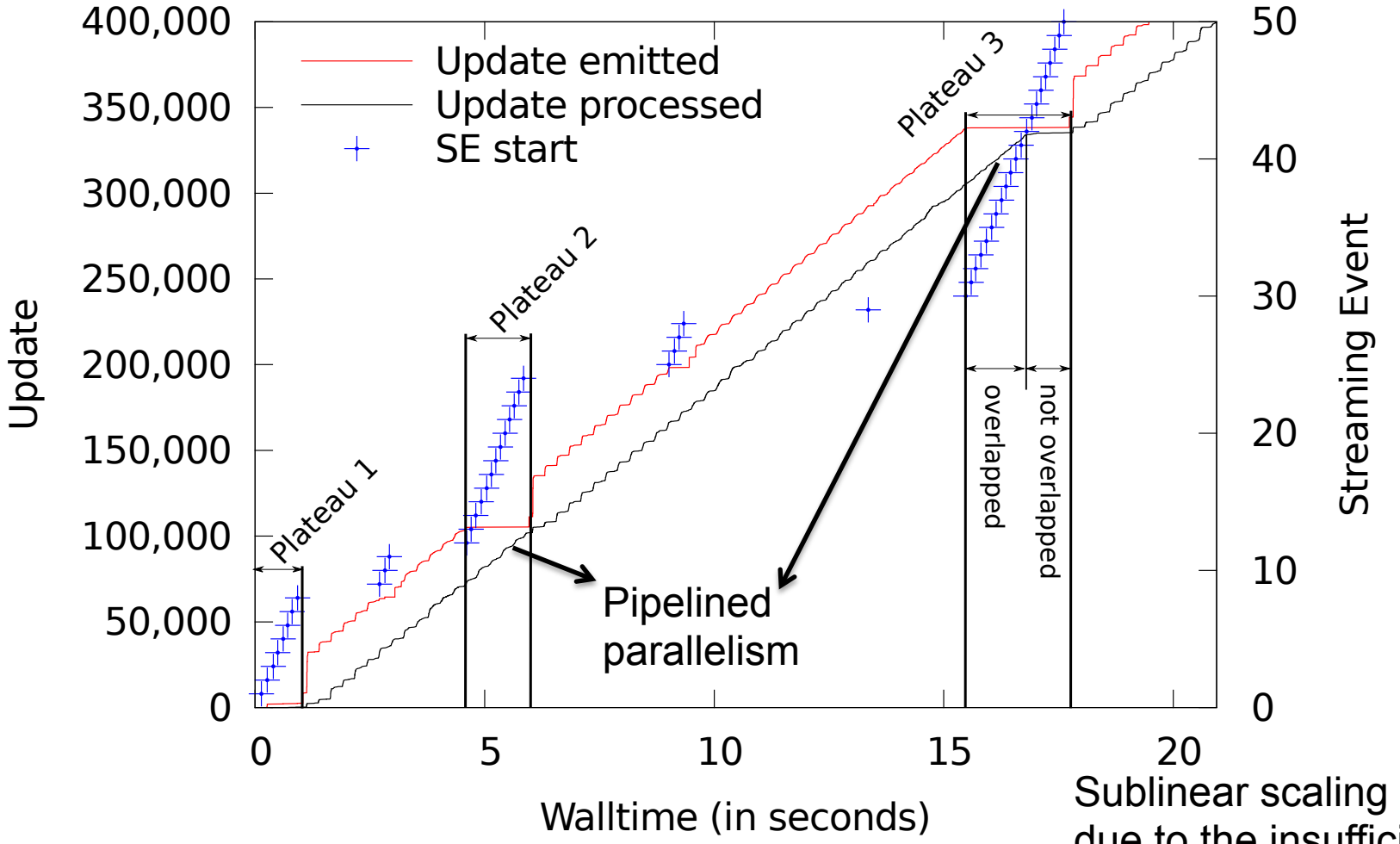
- Exploiting multiple cores and properly taking the shared-memory aspect brings significant improvement

Sublinear scaling case, 3 nodes



(a) 3 worker nodes

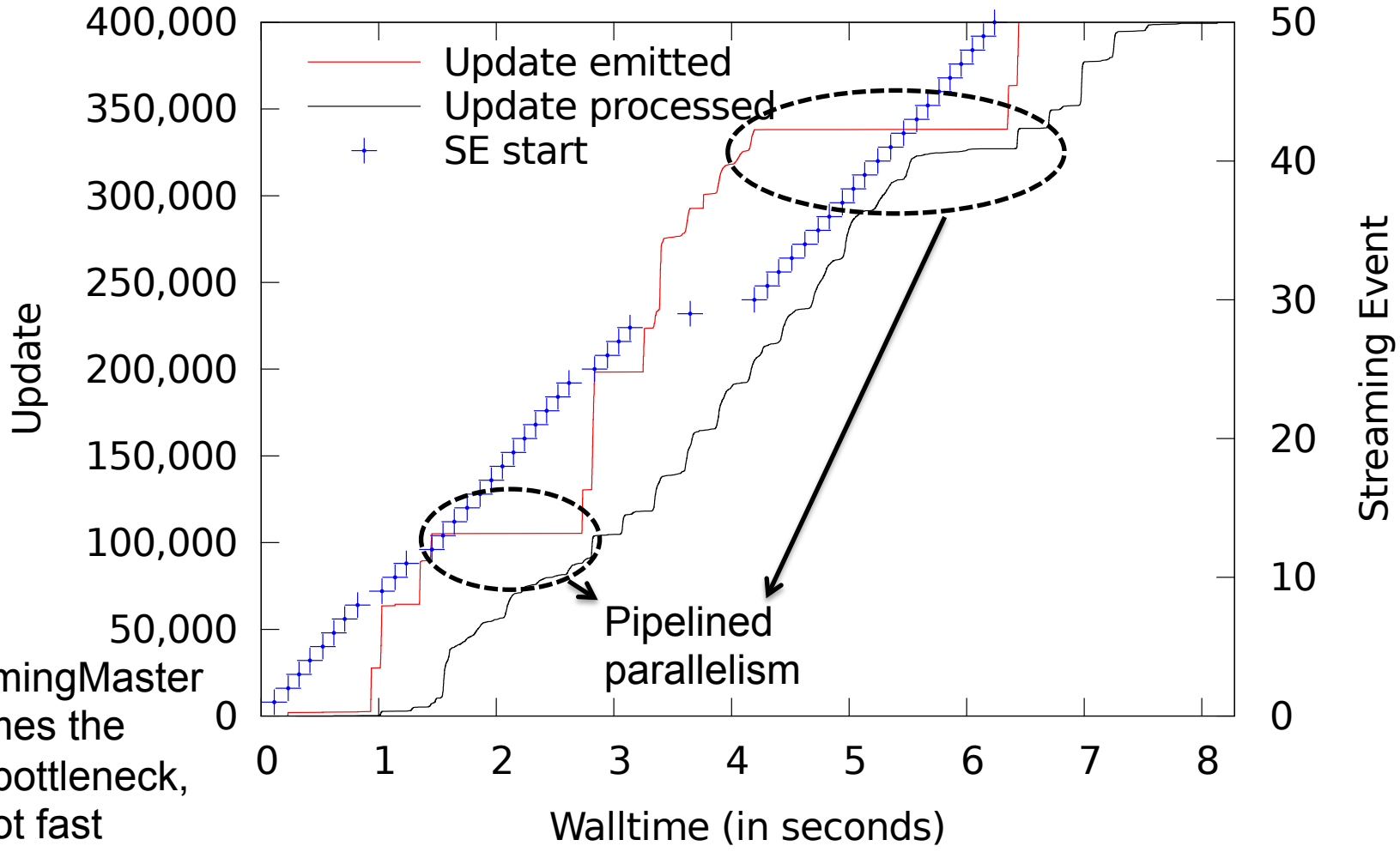
Sublinear scaling case, 15 nodes



(b) 15 worker nodes

Sublinear scaling is due to the insufficient number of emitted updates!

Sublinear scaling case, 63 nodes



(c) 63 worker nodes

StreamingMaster becomes the main bottleneck, i.e., not fast enough to send updates to workers

Conclusion

- STREAMER, a distributed-memory framework, proves to be an effective solution for fast and exact incremental closeness centrality computation
  - Exploits replicated and pipelined parallelism
  - Scales well
  - Reaches speedup of 497 with 64 nodes and 8 cores/node

- Future Work
  - StreamingMaster and Aggregator can be replicated and work can be partitioned
  - Biconnected Decomposition, main part of Aggregator, can be parallelized as well

Thanks

Wexner Medical Center

- For more information
  - Email umit@bmi.osu.edu
  - Visit http://bmi.osu.edu/~umit or http://bmi.osu.edu/hpc

- Acknowledgement of Support