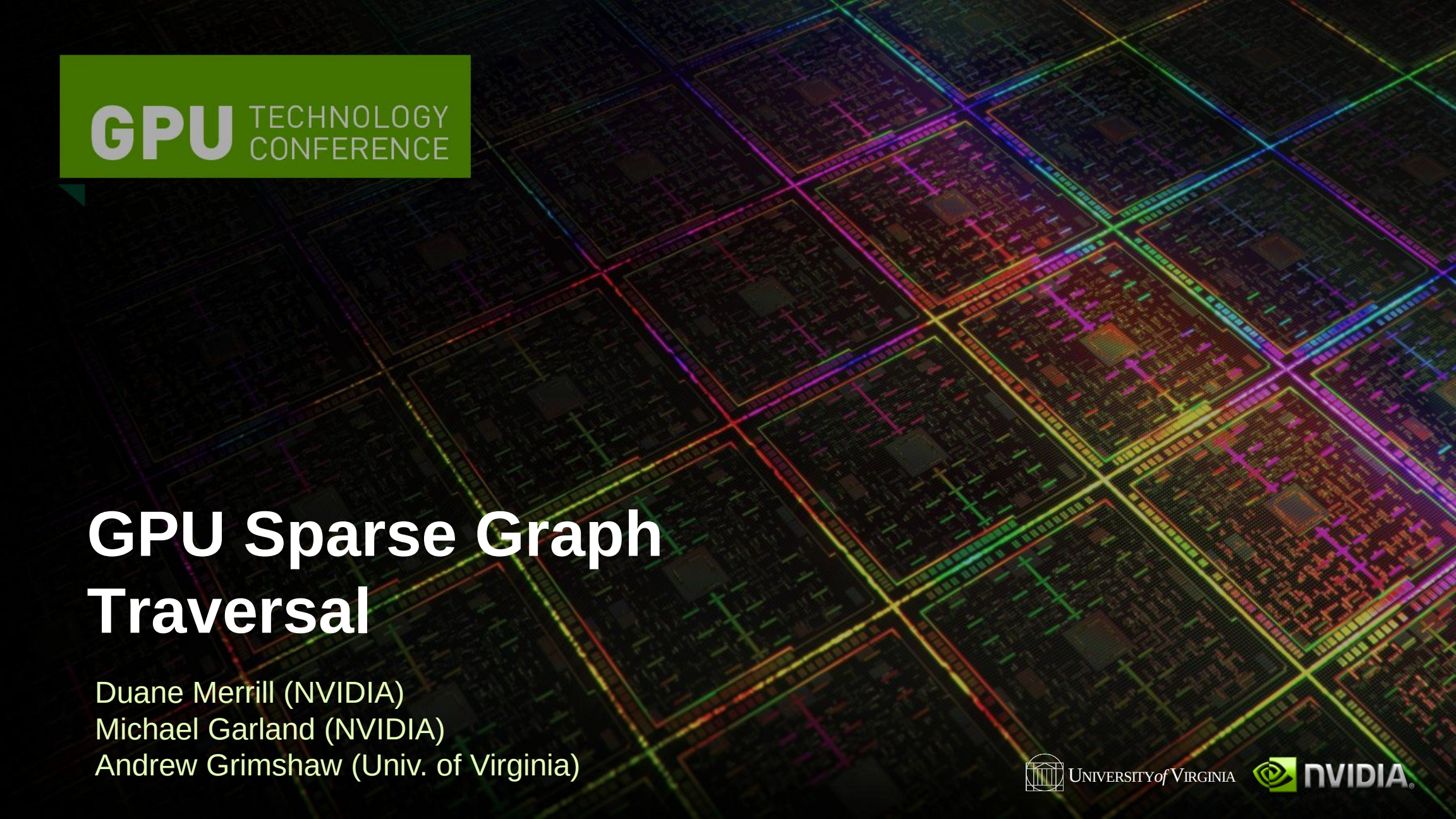


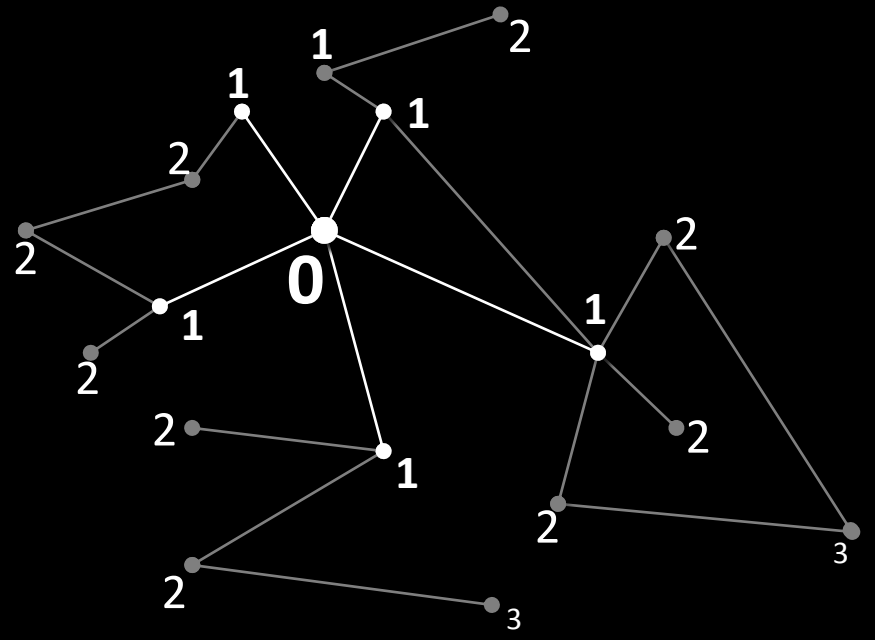
GPU Sparse Graph Traversal

Duane Merrill (NVIDIA)
Michael Garland (NVIDIA)
Andrew Grimshaw (Univ. of Virginia)

Breadth-first search (BFS)

1. Pick a source node
2. Rank every vertex by the length of shortest path from source

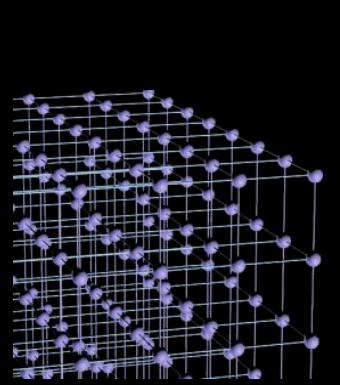
   (or by predecessor vertex)



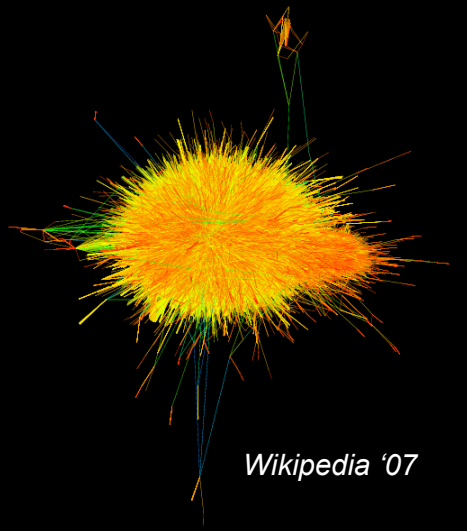
BFS applications

- A common algorithmic building block
  - Tracking reachable heap during garbage collection
  - Belief propagation (statistical inference)
  - Finding community structure
  - Path-finding
- Core benchmark kernel
  - Graph 500
  - Rodinia
  - Parboil

- Simple performance-analog for many applications
  - Pointer-chasing
  - Work queues



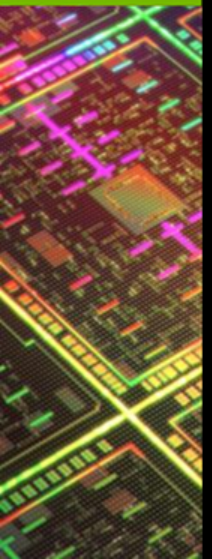
3D lattice



Wikipedia '07

D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP'12.*

GPU TECHNOLOGY CONFERENCE



“Conventional” wisdom

- GPUs would be poorly suited for BFS
  - Not pleasingly data-parallel (if you want to solve it efficiently)
    - Insufficient atomic throughput for cooperation

D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP’12.*

GPU TECHNOLOGY CONFERENCE

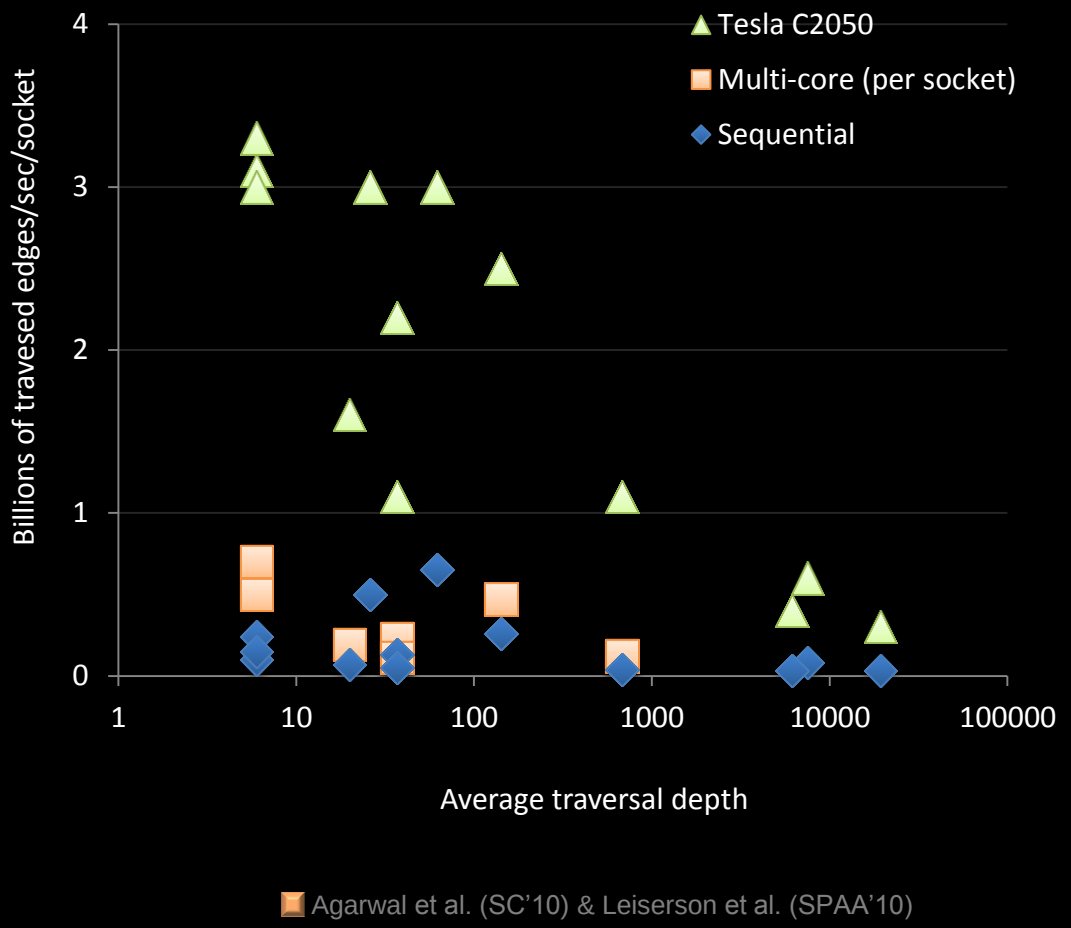
“Conventional” wisdom

- GPUs would be poorly suited for BFS
  - Not pleasingly data-parallel (if you want to solve it efficiently)
    - Insufficient atomic throughput for cooperation
  - Issues with scaling to 10,000s of processing elements
    - Workload imbalance, SIMD underutilization
    - Not enough parallelism in all graphs

D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP’12.*

The punchline

- We show GPUs provide exceptional performance…
  - Billions of traversed-edges per second per socket
  - For diverse synthetic and real-world datasets
  - Linear-work algorithm
- …using efficient ***parallel prefix sum***
  - For cooperative data movement

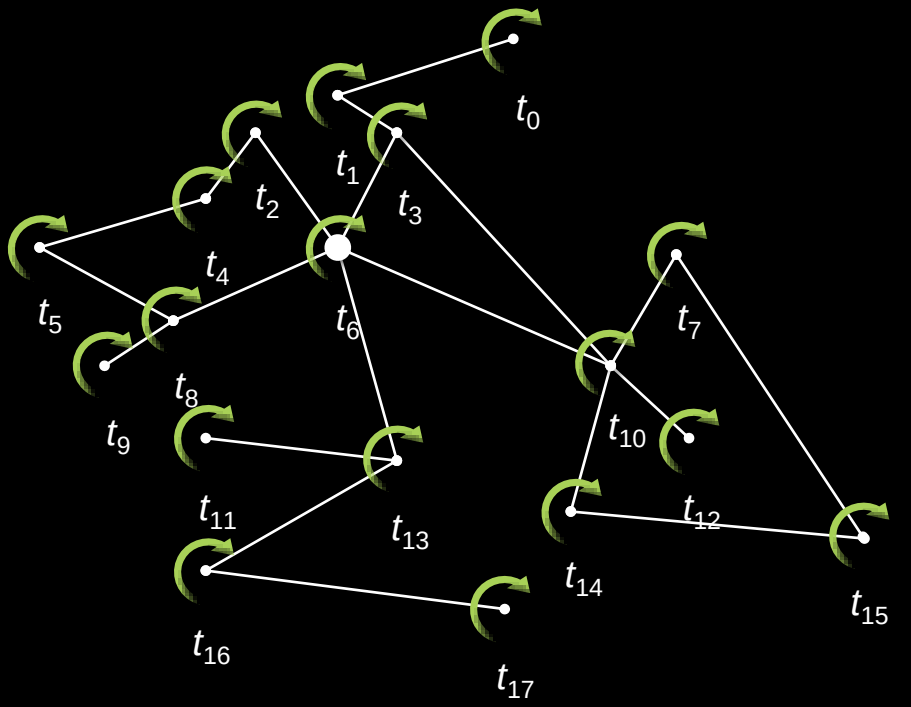



GPU TECHNOLOGY CONFERENCE
Level-synchronous parallelization strategies
UNIVERSITY of VIRGINIA
NVIDIA

(a) Quadratic-work approach

"Typical" GPU BFS (e.g., Harish et al., Deng et al. Hong et al.)

1. Inspect every vertex at every time step
2. If updated, update its neighbors
3. Repeat



- Quadratic O(n2+m) work
- Trivial data-parallel stencils
  - One thread per vertex

D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP'12.*

(b) Linear-work approach

1. Expand neighbors

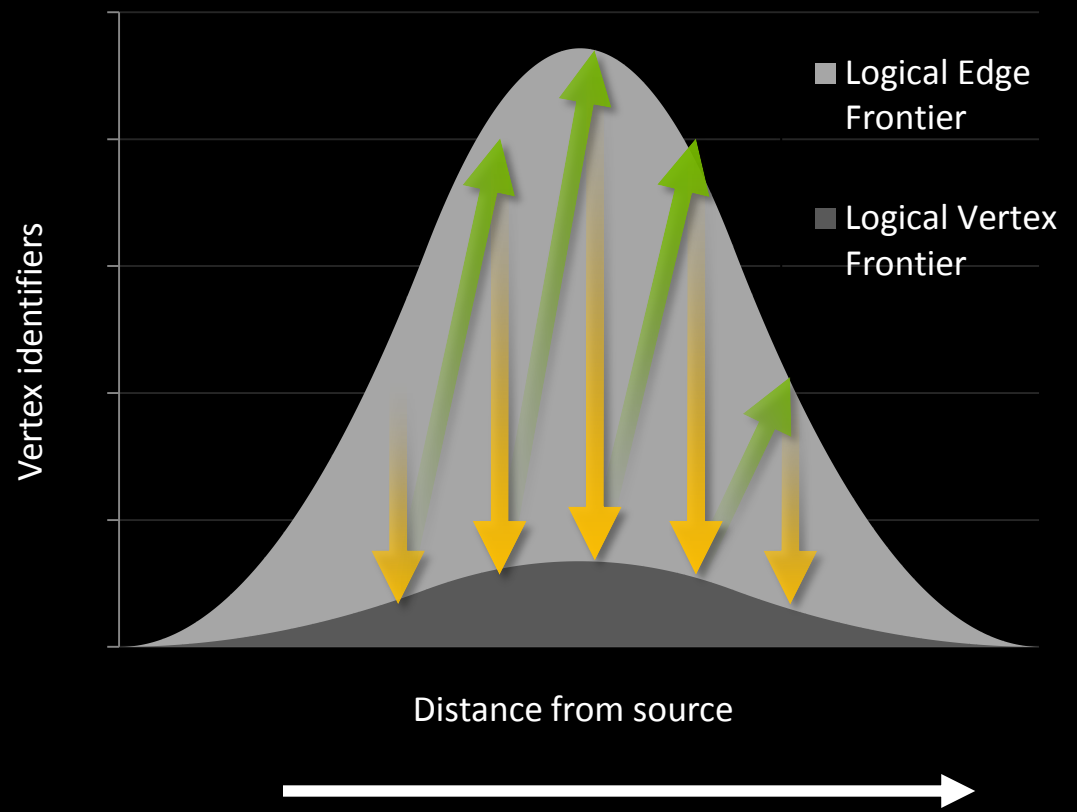
   (vertex frontier → edge-frontier)

2. Filter out previously visited & duplicates

   (edge-frontier → vertex frontier)

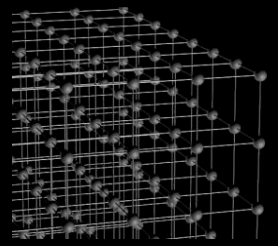
3. Repeat

- *Linear O*(*n*+*m*) work
- Need *cooperative* buffers for tracking frontiers

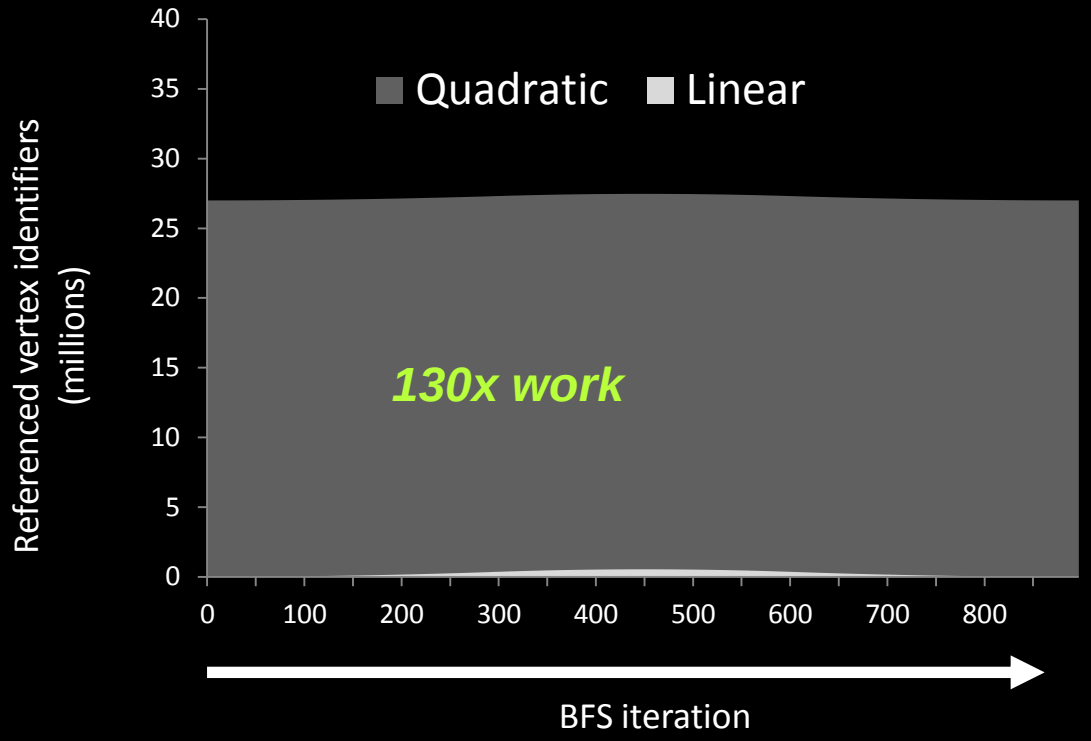


Quadratic vs. linear

Quadratic is too much work!!



3D Poisson Lattice
(300^3 = 27M vertices)

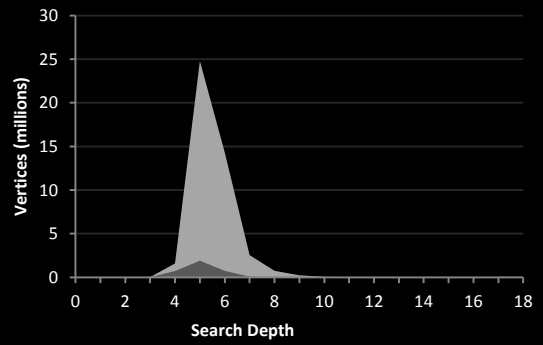


2.5x–2,300x speedup vs. quadratic GPU methods

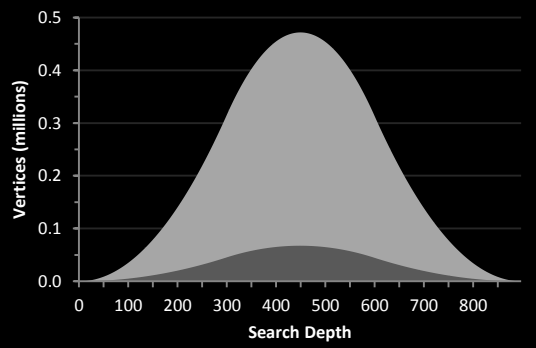
D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP'12.*

Goal: GPU competence on diverse datasets

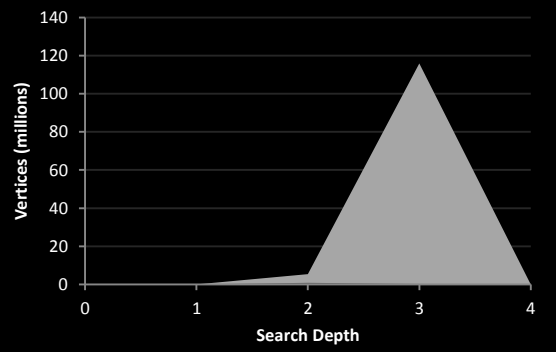
(Not a one-trick pony…)



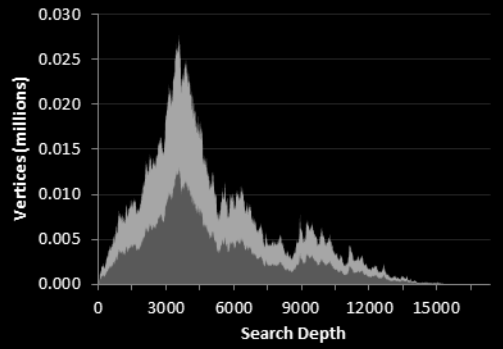
Wikipedia
(social)



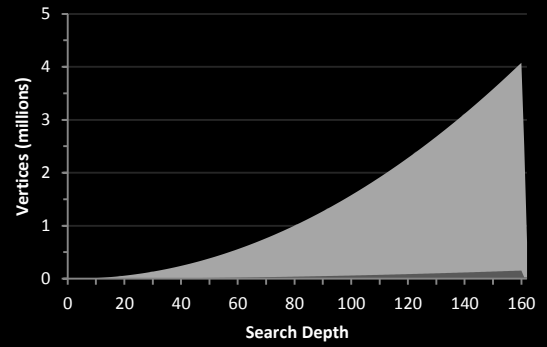
3D Poisson grid
(cubic lattice)



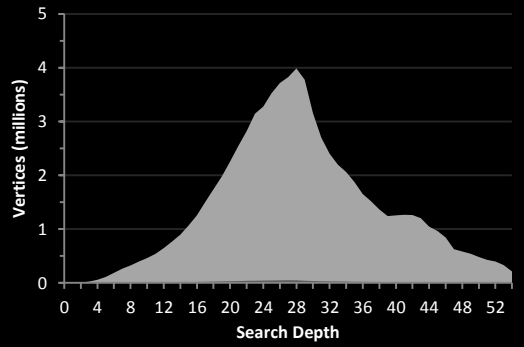
R-MAT
(random, power-law, small-world)



Europe road atlas
(Euclidian space)



PDE-constrained optimization
(non-linear KKT)



Auto transmission manifold
(tetrahedral mesh)


GPU TECHNOLOGY CONFERENCE
Challenges
UNIVERSITY of VIRGINIA
NVIDIA

GPU TECHNOLOGY CONFERENCE

Linear-work challenges for parallelization

- Generic challenges
  1. Load imbalance between cores
     - Coarse workstealing queues
  2. Bandwidth inefficiency
     - Use “status bitmask” when filtering already-visited nodes
- GPU-specific challenges
  1. Cooperative allocation (global queues)
  2. Load imbalance within SIMD lanes
  3. Poor locality within SIMD lanes
  4. Simultaneous discovery (i.e., the benign race condition)

D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP’12.*

GPU TECHNOLOGY CONFERENCE

Linear-work challenges for parallelization

- Generic challenges
  1. Load imbalance between cores
     - Coarse workstealing queues
  2. Bandwidth inefficiency
     - Use “status bitmask” when filtering already-visited nodes
- GPU-specific challenges
  1. Cooperative allocation (global queues)
  2. Load imbalance within SIMD lanes
  3. Poor locality within SIMD lanes
  4. Simultaneous discovery (i.e., the benign race condition)

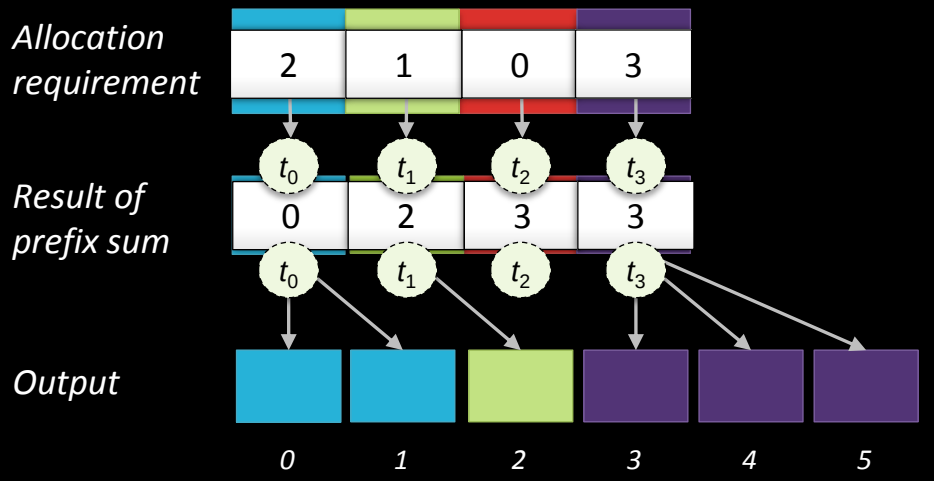
D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP’12.*

(1) Cooperative data movement (queuing)

- Problem:
  - Need shared work “queues”
  - GPU rate-limits (C2050):
    - 16 billion vertex-identifiers / sec (124 GB/s)
    - Only 67M global-atomics / sec (238x slowdown)
    - Only 600M smem-atomics / sec (27x slowdown)
- Solution:
  - Compute enqueue offsets using prefix-sum

D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP’12.*

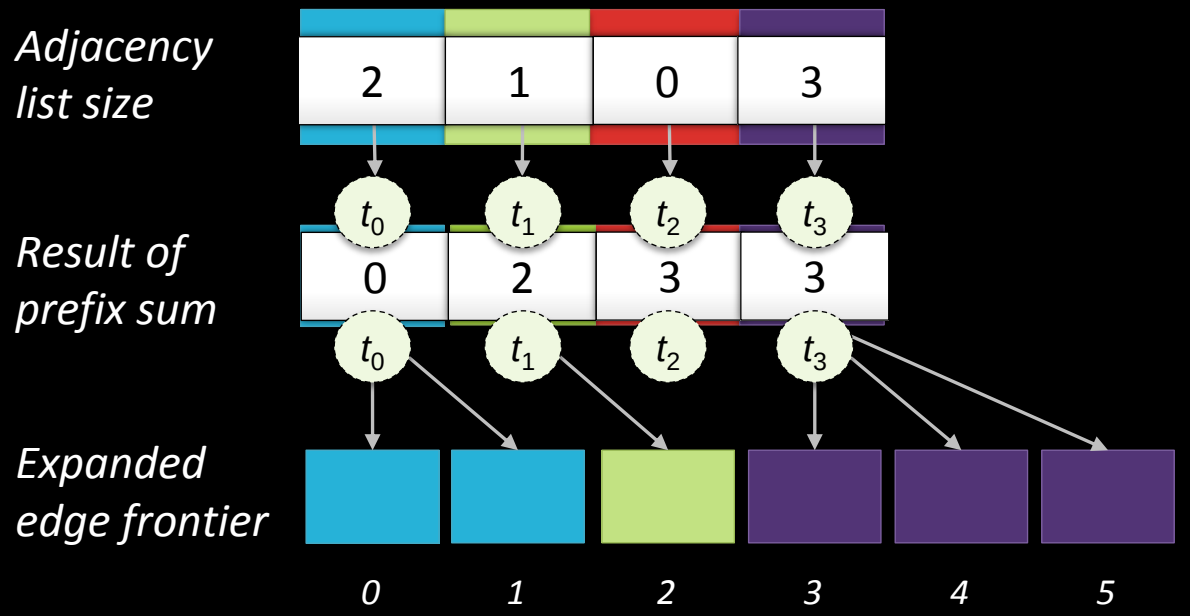
Prefix sum for allocation

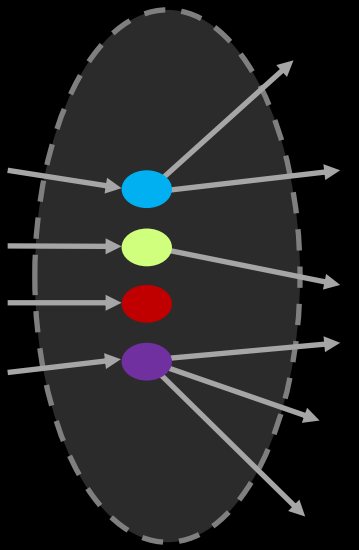


- Each output is computed to be the sum of the previous inputs
  - $O(n)$ work
  - Use results as a scatter offsets
- Fits the GPU machine model well
  - Proceed at copy-bandwidth
  - Only ~8 thread-instructions per input

D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP'12.*

Prefix sum for edge expansion

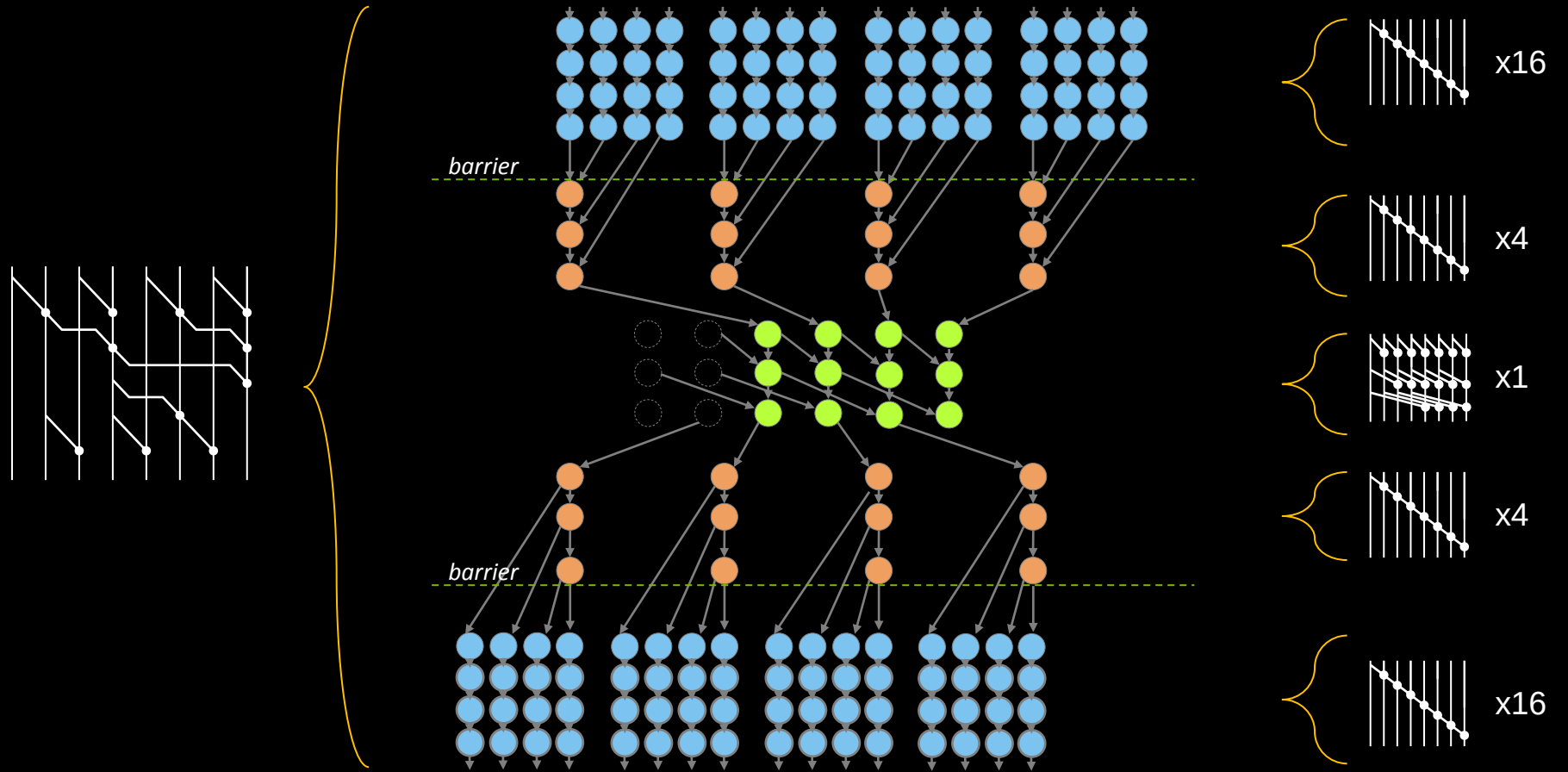




D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP'12.*

Efficient local scan

(granularity coarsening)

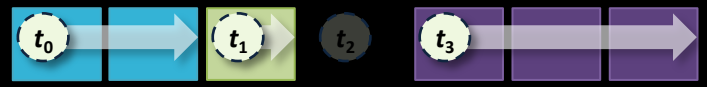


GPU TECHNOLOGY CONFERENCE

(2) Poor load balancing within SIMD lanes

- Problem:
  - Large variance in adjacency list sizes
    - Exacerbated by wide SIMD widths
- Solution:
  - Cooperative neighbor expansion
    - Enlist nearby threads to help process each adjacency list in parallel

a) **Bad:** *Serial expansion & processing*



b) **Slightly better:** *Coarse warp-centric parallel expansion*

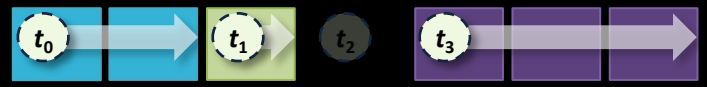
c) **Best**: *Fine-grained parallel expansion (packed by prefix sum)*

D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP'12.*

(2) Poor load balancing within SIMD lanes

- Problem:
  - Large variance in adjacency list sizes
    - Exacerbated by wide SIMD widths
- Solution:
  - Cooperative neighbor expansion
    - Enlist nearby threads to help process each adjacency list in parallel

a) **Bad:** *Serial expansion & processing*



b) **Slightly better:** *Coarse warp-centric parallel expansion*

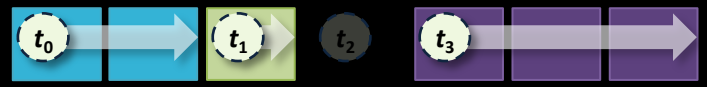
c) **Best**: *Fine-grained parallel expansion (packed by prefix sum)*

D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP'12.*

(3) Poor locality within SIMD lanes

- Problem:
  - The referenced adjacency lists are arbitrarily located
    - Exacerbated by wide SIMD widths
    - Can't afford to have SIMD threads streaming through unrelated data
- Solution:
  - Cooperative neighbor expansion

a) **Bad:** *Serial expansion & processing*

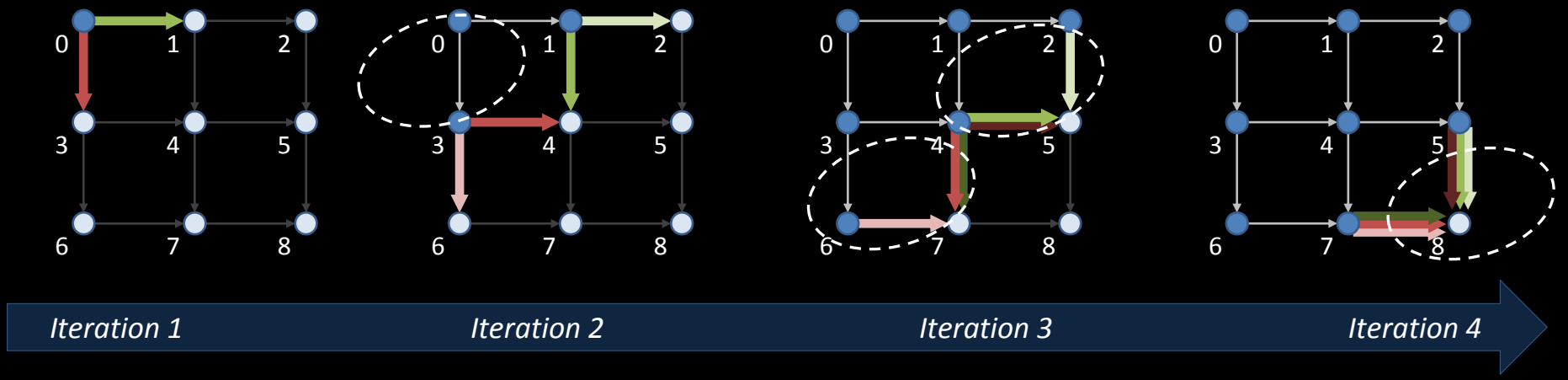


b) **Slightly better:** *Coarse warp-centric parallel expansion*

c) **Best**: *Fine-grained parallel expansion (packed by prefix sum)*

D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP'12.*

(4) SIMD simultaneous discovery

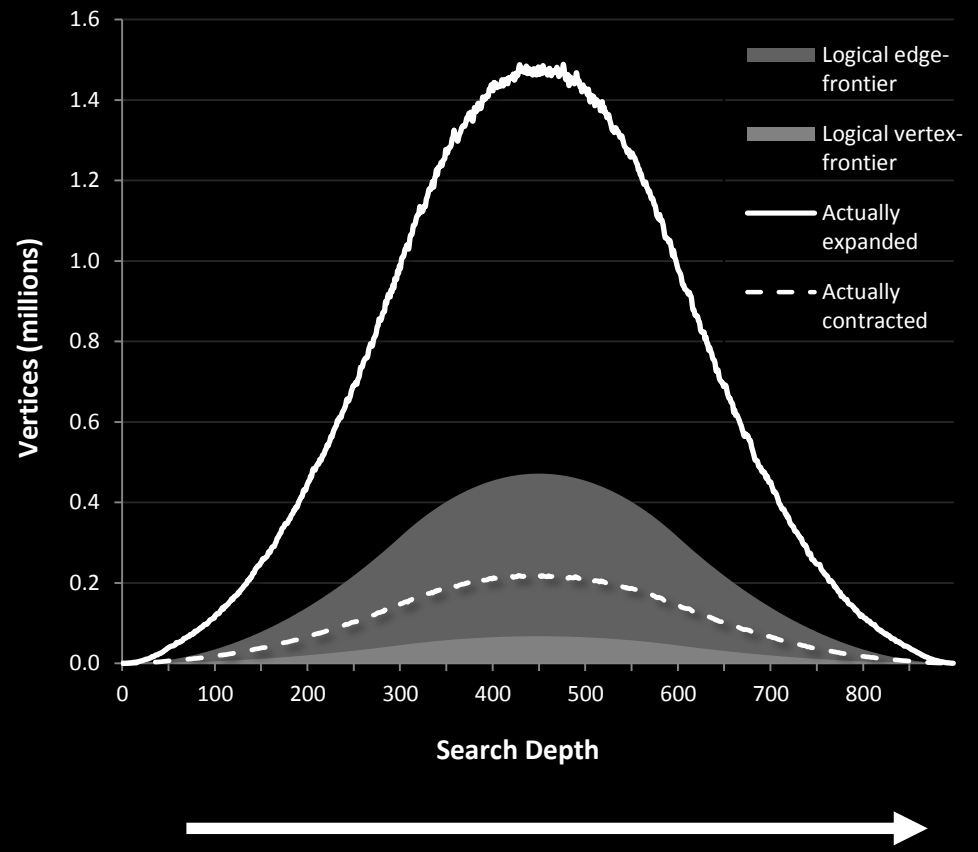


- "Duplicates" in the edge-frontier → redundant work
  - Exacerbated by wide SIMD
  - Compounded every iteration

D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP'12.*

Simultaneous discovery

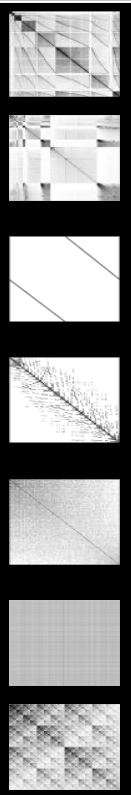
- Normally not a problem for
  - CPU implementations
  - Serial adjacency list inspection
- A ***big*** problem for cooperative SIMD expansion
  - Spatially-descriptive datasets
  - Power-law datasets
- Solution:
  - Localized duplicate removal using hashes in local scratch



Absolute and relative performance

GPU TECHNOLOGY CONFERENCE

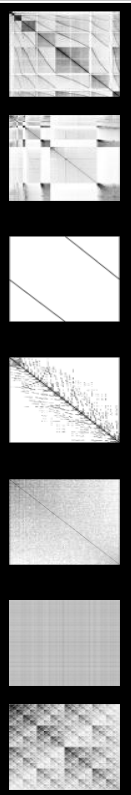
Single-socket comparison

Graph	Spy Plot	Average Search Depth	Nehalem		Tesla C2050	
			Billion TE/s	Parallel speedup	Billion TE/s	Parallel speedup†††
europe.osm		19314			0.3	11 x
grid5pt.5000		7500			0.6	7.3 x
hugebubbles		6151			0.4	15 x
grid7pt.300		679	0.12 (4-core†)	2.4 x	**1.1**	28 x
nlpkkt160		142	0.47 (4-core†)	3.0 x	2.5	10 x
audikw1		62			3.0	4.6 x
cage15		37	0.23 (4-core†)	2.6 x	2.2	18 x
kkt_power		37	0.11 (4-core†)	3.0 x	1.1	23 x
coPapersCiteseer		26			3.0	5.9 x
wikipedia-2007		20	0.19 (4-core†)	3.2 x	1.6	25 x
kron_g500-logn20		6			3.1	13 x
random.2Mv.128Me		6	0.50 (8-core††)	7.0 x	3.0	29 x
rmat.2Mv.128Me		6	0.70 (8-core††)	6.0 x	3.3	22 x

† 2.5 GHz Core i7 4-core (Leiserson *et al.*) †† 2.7 GHz EX Xeon X5570 8-core (Agarwal *et al*) ††† vs 3.4GHz Core i7 2600K (Sandybridge)

GPU TECHNOLOGY CONFERENCE

Single-socket comparison

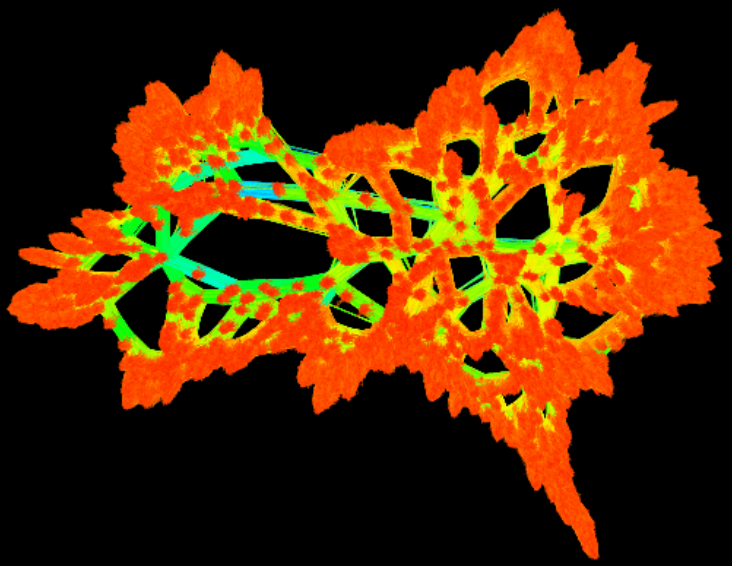
Graph	Spy Plot	Average Search Depth	Nehalem		Tesla C2050	
			Billion TE/s	Parallel speedup	Billion TE/s	Parallel speedup†††
europe.osm		19314			0.3	11 x
grid5pt.5000		7500			0.6	7.3 x
hugebubbles		6151			0.4	15 x
grid7pt.300		679	0.12 (4-core†)	2.4 x	**1.1**	28 x
nlpkkt160		142	0.47 (4-core†)	3.0 x	2.5	10 x
audikw1		62			3.0	4.6 x
cage15		37	0.23 (4-core†)	2.6 x	2.2	18 x
kkt_power		37	0.11 (4-core†)	3.0 x	1.1	23 x
coPapersCiteseer		26			3.0	5.9 x
wikipedia-2007		20	0.19 (4-core†)	3.2 x	1.6	25 x
kron_g500-logn20		6			3.1	13 x
random.2Mv.128Me		6	0.50 (8-core††)	7.0 x	3.0	29 x
rmat.2Mv.128Me		6	0.70 (8-core††)	6.0 x	3.3	22 x

† 2.5 GHz Core i7 4-core (Leiserson *et al.*) †† 2.7 GHz EX Xeon X5570 8-core (Agarwal *et al*) ††† vs 3.4GHz Core i7 2600K (Sandybridge)

GPU TECHNOLOGY CONFERENCE

Summary

- Quadratic-work approaches are uncompetitive
- Dynamic workload management:
  - Prefix sum (vs. atomic-add)
  - Utilization requires fine-grained expansion and contraction
- GPUs can be very amenable to dynamic, cooperative problems

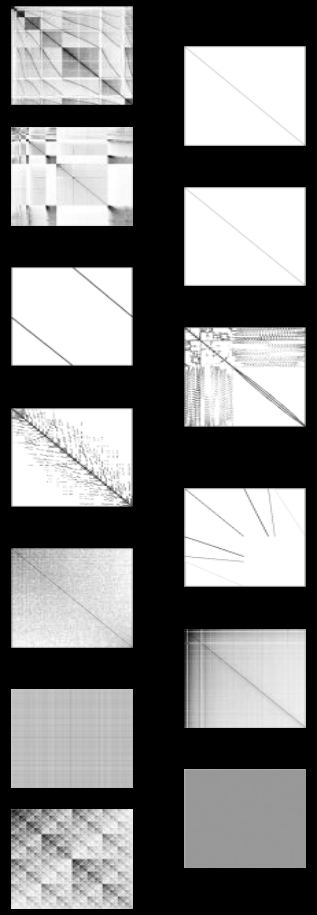


Zaoui@kkt_power. 2063494 nodes, 6482320 edges.

D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP'12.*

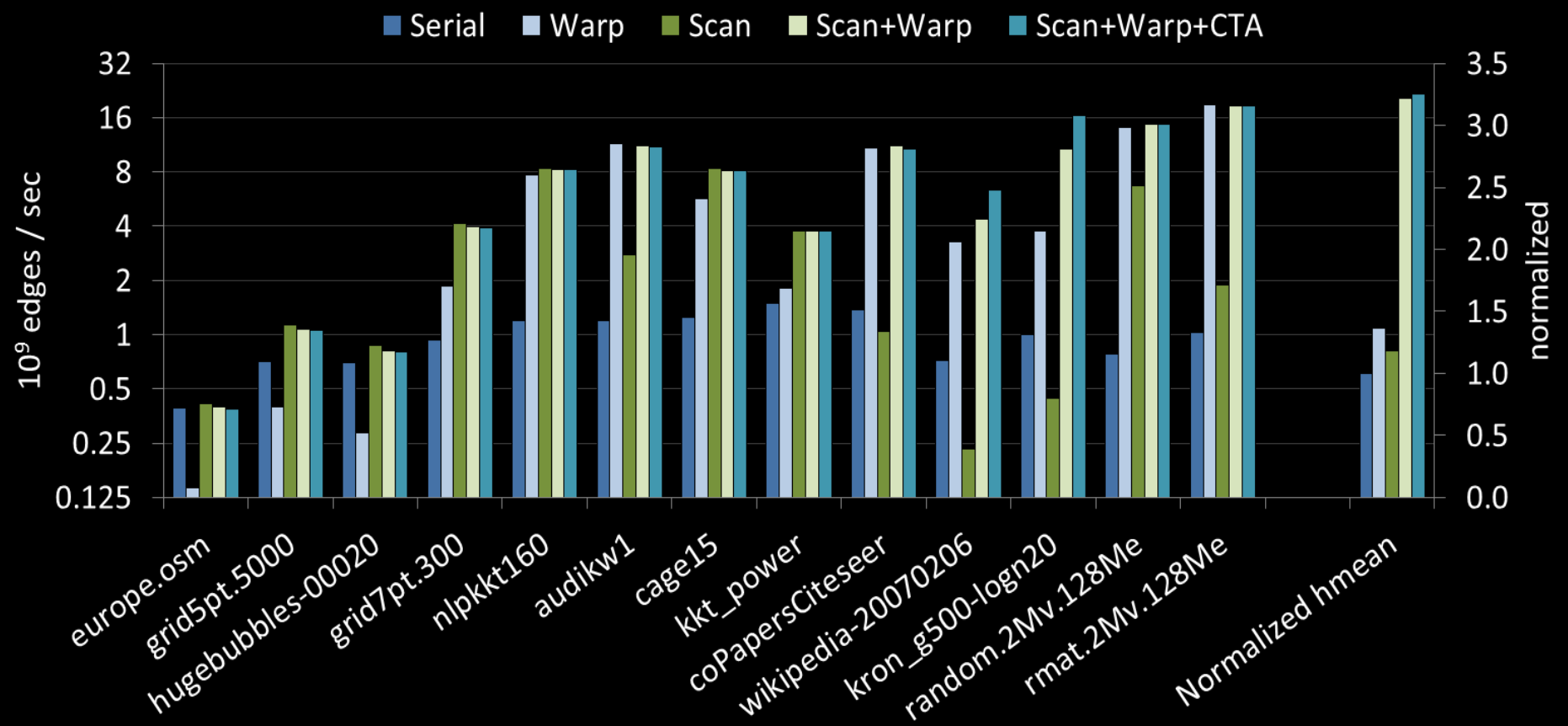

GPU TECHNOLOGY CONFERENCE
Questions?
UNIVERSITY of VIRGINIA
NVIDIA

Experimental corpus

Graph	Spy Plot	Description	Average Search Depth	Vertices (millions)	Edges (millions)
europe.osm		Road network	19314	50.9	108.1
grid5pt.5000		2D Poisson stencil	7500	25.0	125.0
hugebubbles-00020		2D mesh	6151	21.2	63.6
grid7pt.300		3D Poisson stencil	679	27.0	188.5
nlpkkt160		Constrained optimization problem	142	8.3	221.2
audikw1		Finite element matrix	62	0.9	76.7
cage15		Transition prob. matrix	37	5.2	94.0
kkt_power		Optimization (KKT)	37	2.1	13.0
coPapersCiteseer		Citation network	26	0.4	32.1
wikipedia-20070206		Wikipedia page links	20	3.6	45.0
kron_g500-logn20		Graph500 random graph	6	1.0	100.7
random.2Mv.128Me		Uniform random graph	6	2.0	128.0
rmat.2Mv.128Me		RMAT random graph	6	2.0	128.0

GPU TECHNOLOGY CONFERENCE

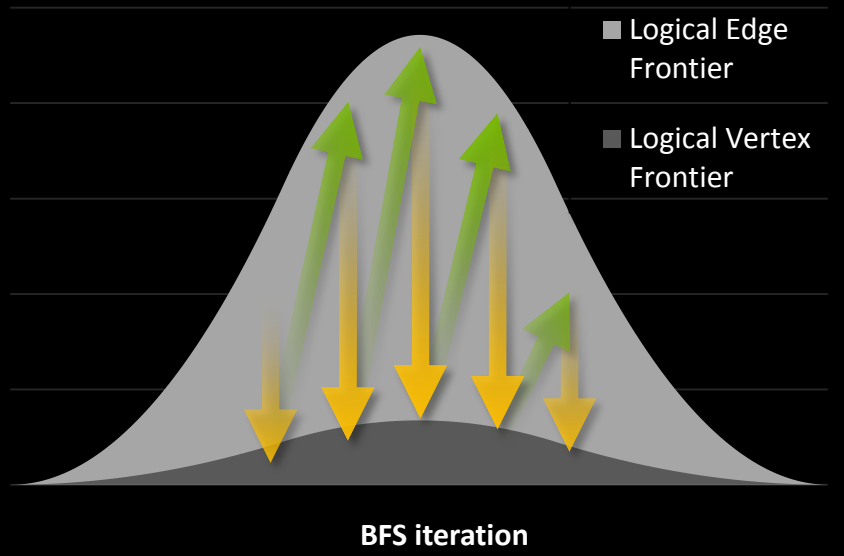
Comparison of expansion techniques



D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP'12.*

Coupling of expansion & contraction phases

- Alternatives:
  - Expand-contract
    - Wholly realize vertex-frontier in DRAM
    - Suitable for all types of BFS iterations
  - Contract-expand
    - Wholly realize edge-frontier in DRAM
    - Even better for small, fleeting BFS iterations
  - Two-phase
    - Wholly realize both frontiers in DRAM
    - Even better for large, saturating BFS iterations (surprising!!)



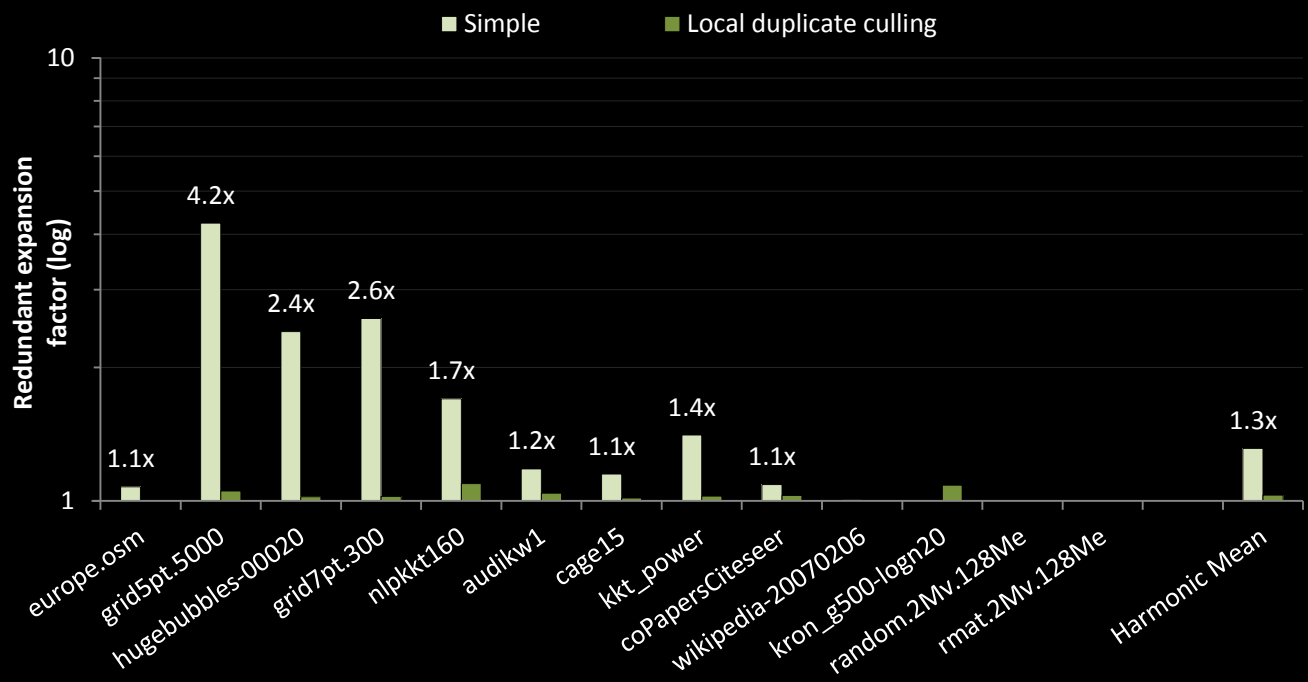
D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP'12.*

Comparison of coupling approaches

D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP'12.*

Local duplicate culling

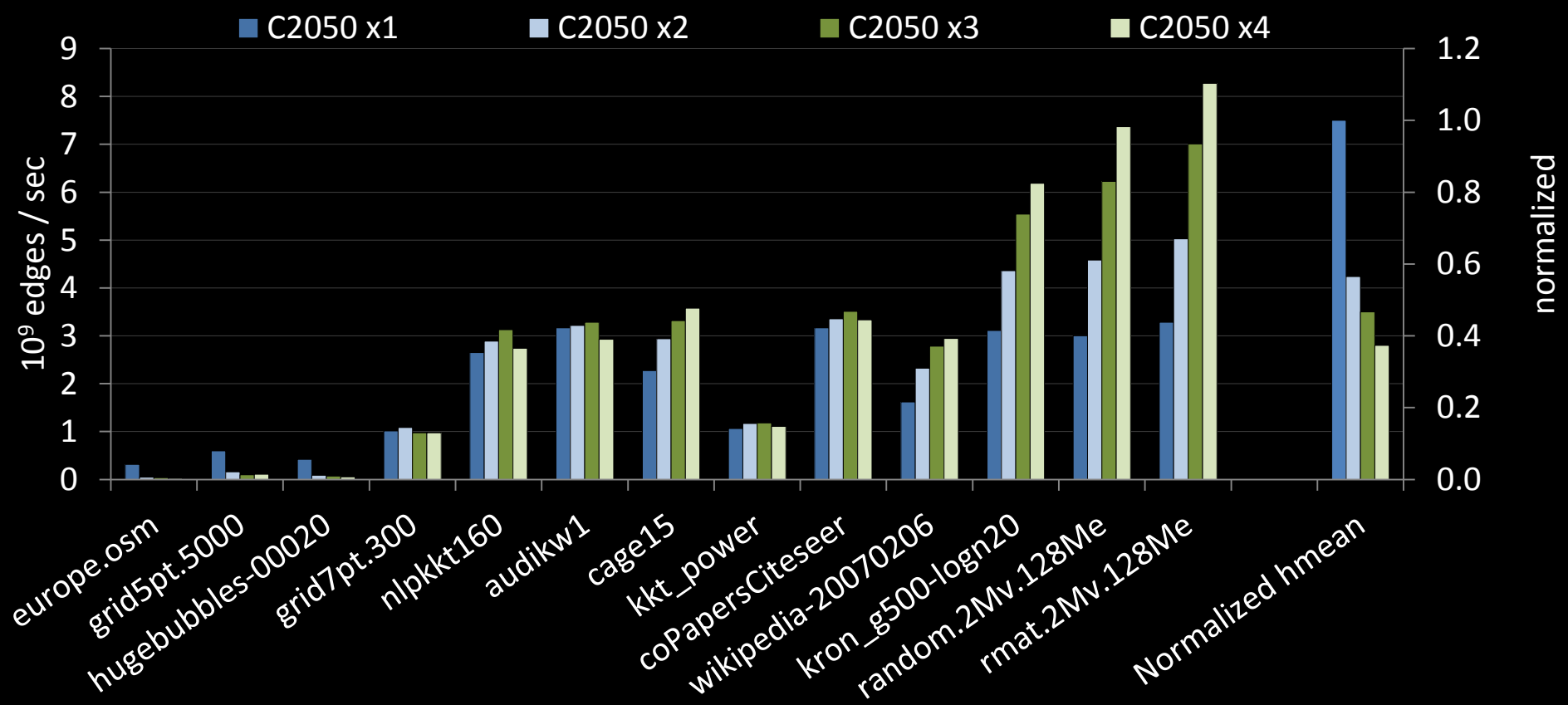
- Contraction uses local collision hashes in smem scratch space:
  - Hash per warp (instantaneous coverage)
  - Hash per CTA (recent history coverage)
- Redundant work < 10% in all datasets
- No atomics needed



D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP'12.*

Multi-GPU traversal

Expand neighbors → sort by GPU (with filter) → read from peer GPUs (with filter) → repeat



D. Merrill, M. Garland, A. Grimshaw. Scalable GPU Graph Traversal. *PPoPP'12.*