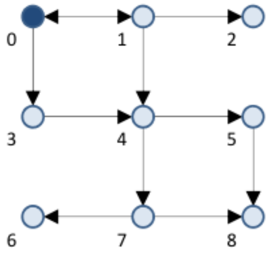
Scalable GPU graph traversal

BFS

Compressed Row Format



C:	1	3	0	2	4	4	5	7	8	6	8
	0	1	2	3	4	5	6	7	8	9	10

R:	0	2	5	5	6	8	9	9	11	11
	0	1	2	3	4	5	6	7	8	9

Traversal from source vertex v_0		
BFS Iteration	***Vertex frontier***	***Edge frontier***
1	{0}	{1,3}
2	{1,3}	{0,2,4,4}
3	{2,4}	{5,7}
4	{5,7}	{6,8,8}
5	{6,8}	{}

Fig. 1. Example sparse graph, corresponding CSR representation, and frontier evolution for a BFS beginning at source vertex v_0.

Sequential BFS

Algorithm 1. The simple sequential breadth-first search algorithm for marking vertex distances from the source s.

Input: Vertex set V, row-offsets array R, column-indices array C, source vertex v_s
Output: Array $dist[0..n\text{-}1]$ with $dist[v_i]$ holding the distance from v_s to v_i
Functions: *Enqueue*(*val*) inserts *val* at the end of the queue instance. *Dequeue*() returns the front element of the queue instance.

```
Q := {}
for i in 0 .. |V|-1:
  dist[i] := ∞
dist[s] := 0
Q.Enqueue(s)
while (Q != {}) :
  i = Q.Dequeue()
  for offset in R[i] .. R[i+1]-1 :
    j := C[offset]
    if (dist[j] == ∞)
      dist[j] := dist[i] + 1;
      Q.Enqueue(j)
```

Parallel BFS

- Quadratic parallelizations - $O(n^2+m)$
- Linear parallelizations - $O(n+m)$
  - Frontiers may be maintained in-core or out-of-core
- Distributed parallelizations
  - partition the graph amongst multiple processors
  - out-of-core edge queues are used for communication
- Our parallelization strategy: out-of-core E&V

Prefix sum

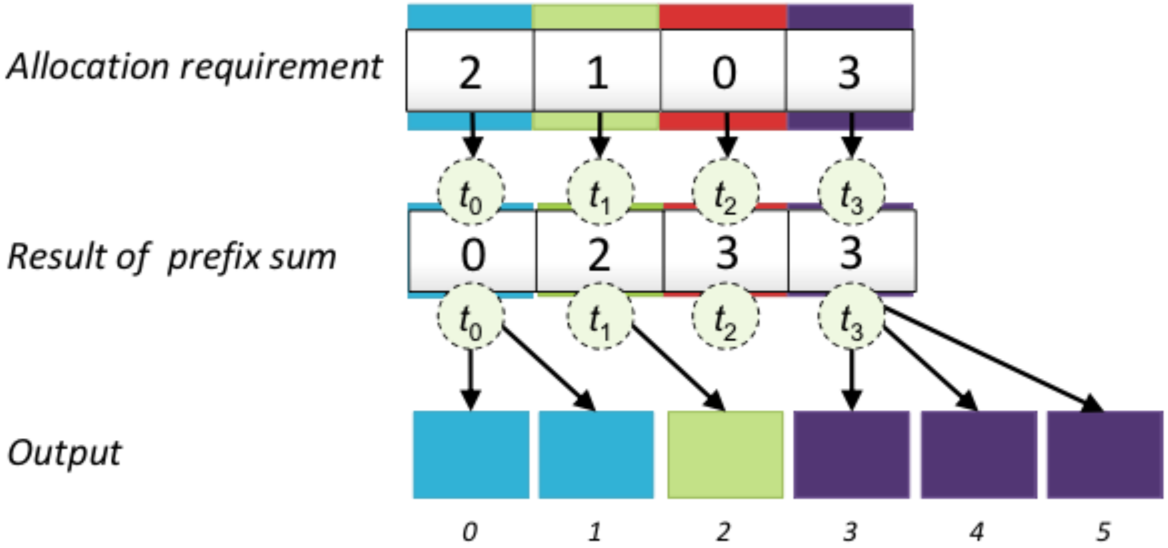


Fig. 2. Example of prefix sum for computing scatter offsets for run-length expansion. Input order is preserved.

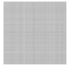
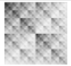
Name	Sparsity Plot	Description	n (10^6)	m (10^6)	$\bar{d}$	Avg. Search Depth
europe.osm		European road network	50.9	108.1	2.1	19314
grid5pt.5000		5-point Poisson stencil (2D grid lattice)	25.0	125.0	5.0	7500
hugebubbles-00020		Adaptive numerical simulation mesh	21.2	63.6	3.0	6151
grid7pt.300		7-point Poisson stencil (3D grid lattice)	27.0	188.5	7.0	679
nlpkkt160		3D PDE-constrained optimization	8.3	221.2	26.5	142
audikw1		Automotive finite element analysis	0.9	76.7	81.3	62
cage15		Electrophoresis transition probabilities	5.2	94.0	18.2	37
kkt_power		Nonlinear optimization (KKT)	2.1	13.0	6.3	37
coPapersCiteseer		Citation network	0.4	32.1	73.9	26
wikipedia-20070206		Links between Wikipedia pages	3.6	45.0	12.6	20
kron_g500-logn20		Graph500 RMAT (*A*=0.57, *B*=0.19, *C*=0.19)	1.0	100.7	96.0	6
random.2Mv.128Me		$G(n, M)$ uniform random	2.0	128.0	64.0	6
rmat.2Mv.128Me		RMAT (*A*=0.45, *B*=0.15, *C*=0.15)	2.0	128.0	64.0	6

Table 1. Suite of benchmark graphs

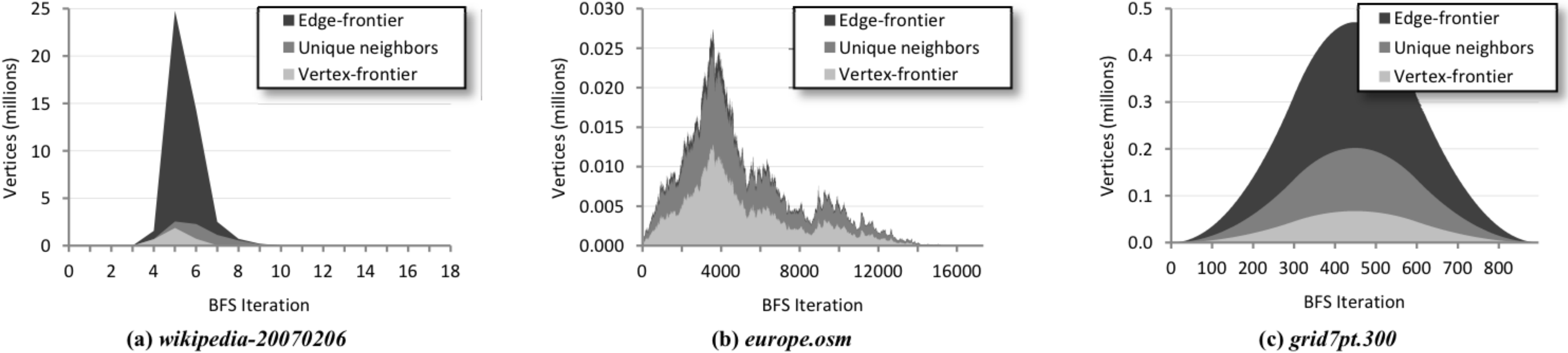


Fig. 3. Sample frontier plots of logical vertex and edge-frontier sizes during graph traversal.

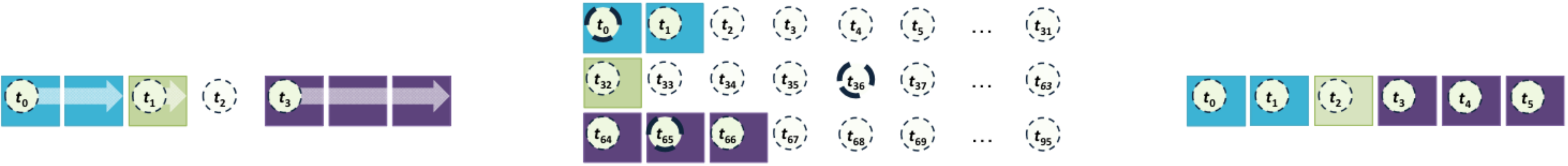
Microbenchmark Analyses

Because edge-frontier is dominant we focus on

- neighbor-gathering
- status-lookup

Isolated neighbor-gathering

- Serial gathering
- Coarse-grained, warp-based gathering
- Fine-grained, scan-based gathering
- Scan+warp+CTA gathering

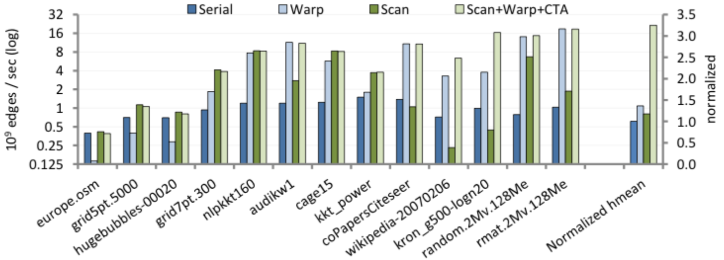


(a) serial

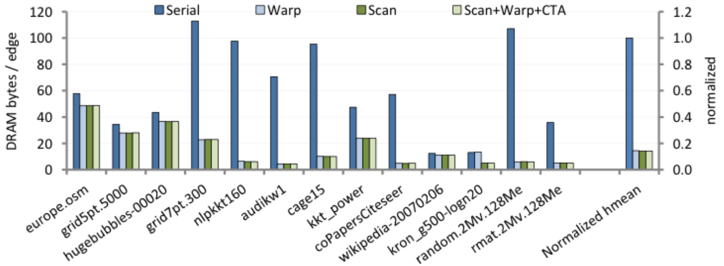
(b) coarse-grained, warp-based cooperative expansion (emphasis on controlling thread)

(c) fine-grained, scan-based cooperative expansion

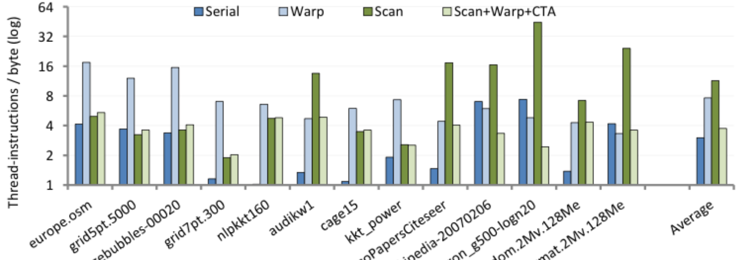
Fig. 4. Alternative strategies for gathering four unexplored adjacency lists having lengths 2, 1, 0, and 3.



(a) Average gather rate (log)



(b) Average DRAM overhead

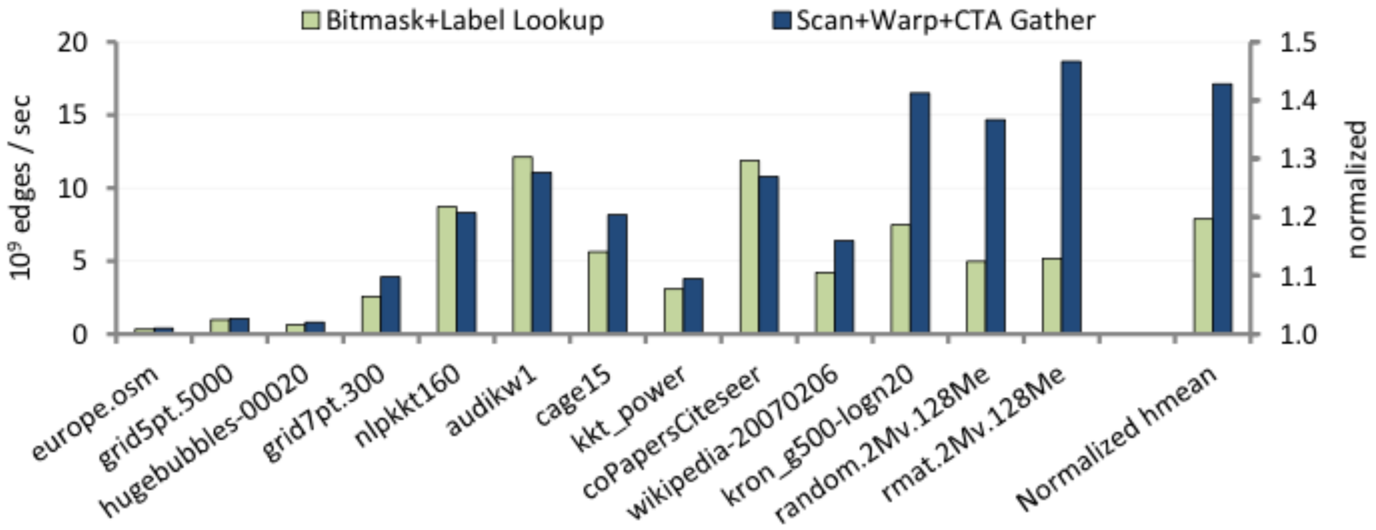


(c) Average computational intensity (log)

Isolated status-lookup

Use bitmask to reduce size of status data from 32 bit to 1 bit.

Avoid atomic operations therefore bitmask is conservative approximation.

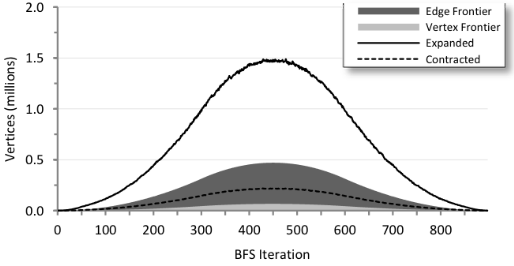


Bitmask+Label Lookup
Scan+Warp+CTA Gather
10^9 edges / sec
normalized
0
5
10
15
20
1.0
1.1
1.2
1.3
1.4
1.5
europe.osm
grid5pt.5000
hugebubbles-00020
grid7pt.300
nlpkkt160
audikw1
cage15
kkt_power
coPapersCiteseer
wikipedia-20070206
kron_g500-logn20
random.2Mv.128Me
rmat.2Mv.128Me
Normalized hmean

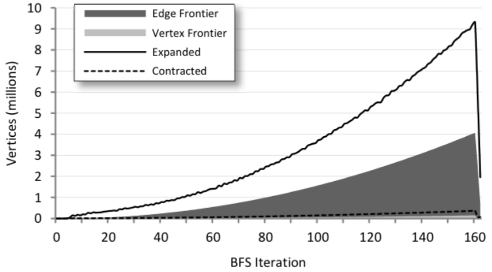
Concurrent discovery

Key: number of duplicate vertices in the edge-frontier.

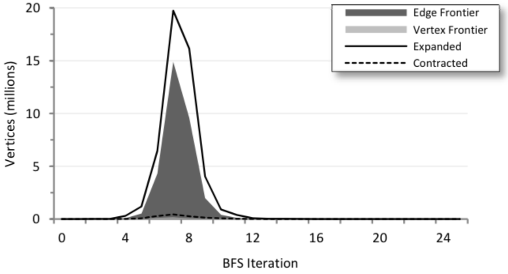
- Warp culling
- History culling



(a) ***grid7pt.300***

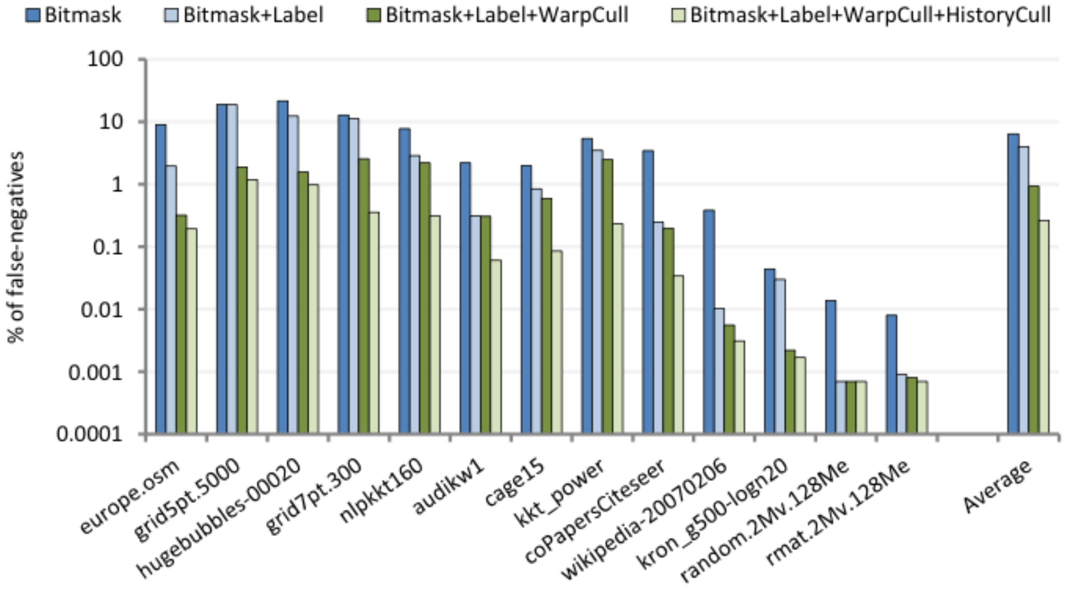


(b) ***nlpkkt160***

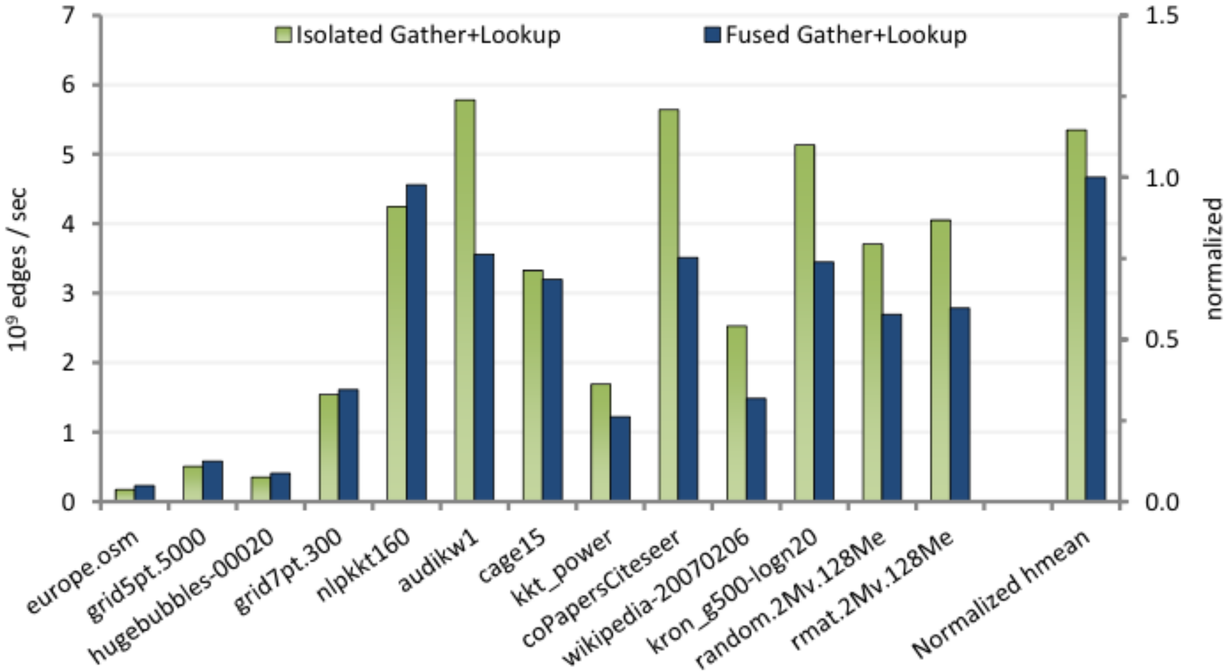


(c) ***coPapersCiteseer***

Fig. 7. Actual expanded and contracted queue sizes without local duplicate culling, superimposed over logical frontier sizes. The redundant expansion factors are 2.6x, 1.7x, and 1.1x for the *grid7pt.300*, *nlpkkt160*, and *coPapersCiteseer* datasets, respectively.

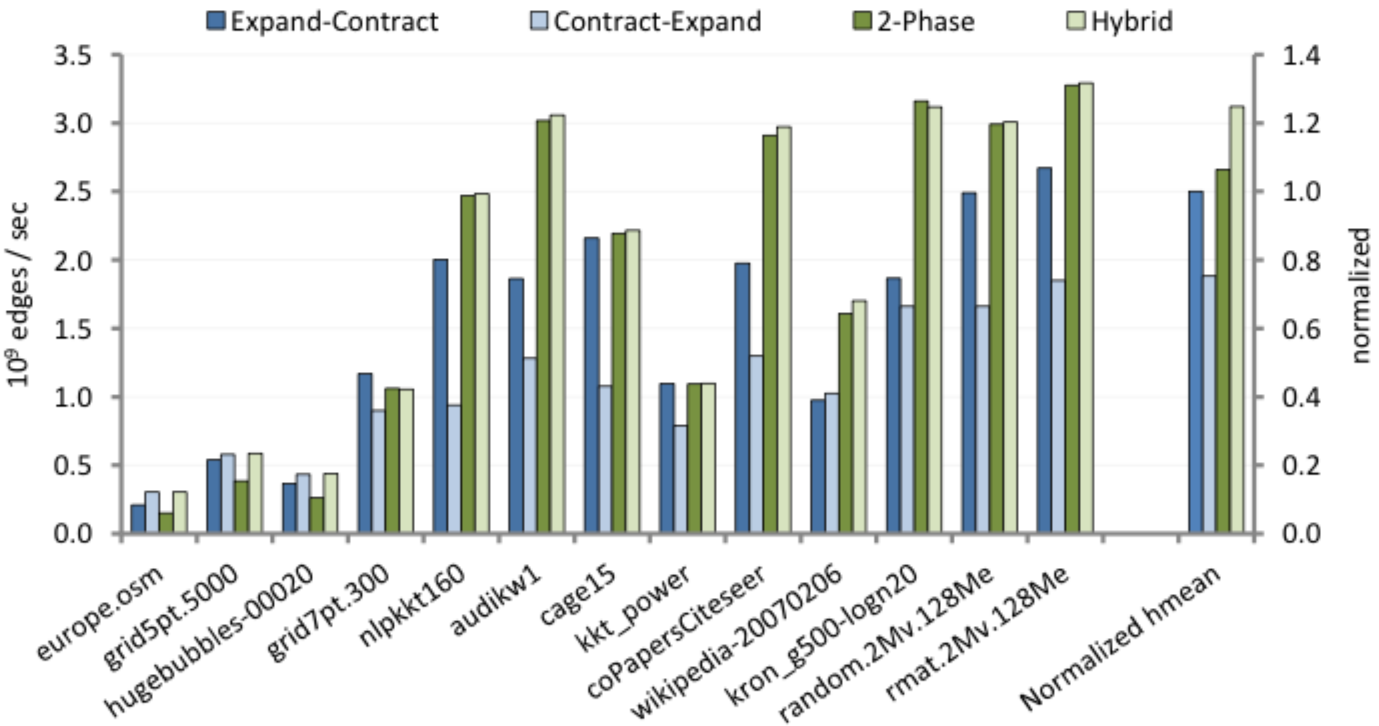


Fused neighbor-gathering and lookup



Single-GPU parallelizations

- Expand-contract (out-of-core vertex queue)
- Contract-expand (out-of-core edge queue)
- Two-phase (both queues out-of-core)
- Hybrid (contract-expand + two-phase)

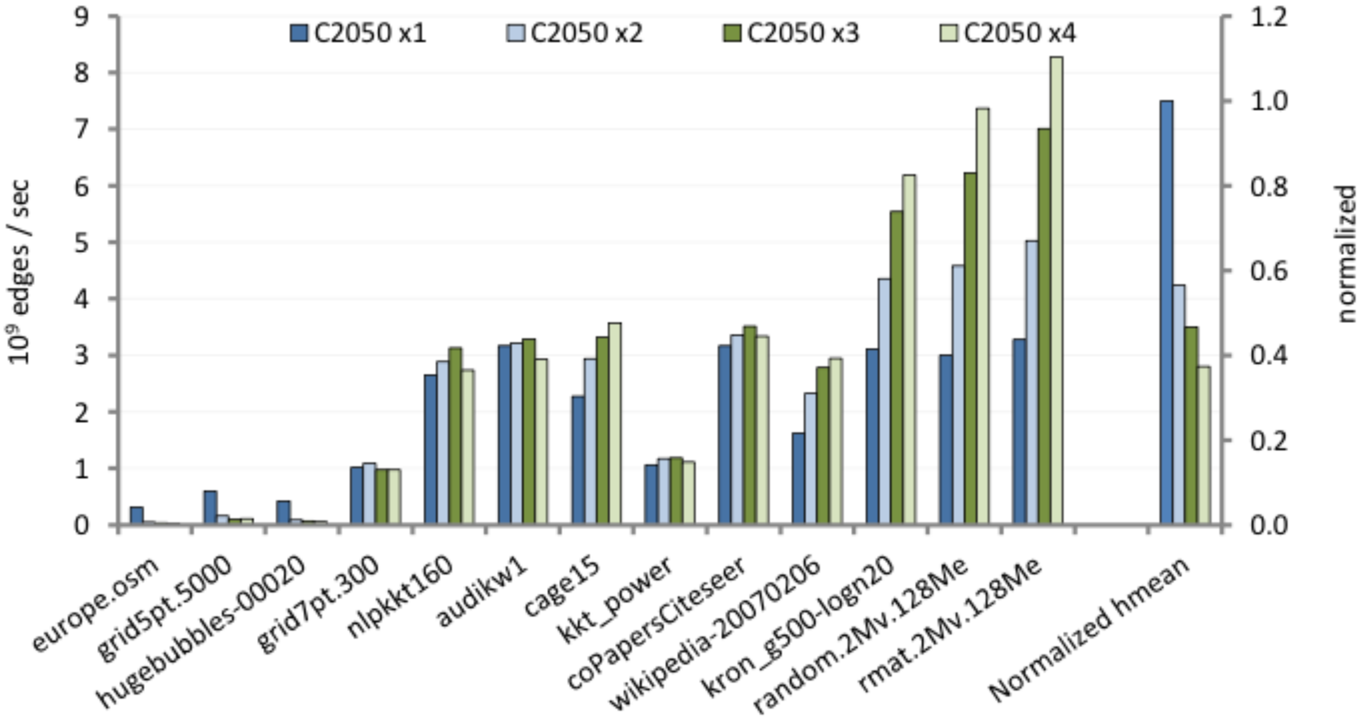


Expand-Contract
Contract-Expand
2-Phase
Hybrid
10^9 edges / sec
normalized
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
europe.osm
grid5pt.5000
hugebubbles-00020
grid7pt.300
nlpkkt160
audikw1
cage15
kkt_power
coPapersCiteseer
wikipedia-20070206
kron_g500-logn20
random.2Mv.128Me
rmat.2Mv.128Me
Normalized hmean

Graph Dataset	CPU Parallel (linear-work)		GPU* (quadratic-work [18])	GPU* (linear-work *hybrid* strategy)	
	*Distance BFS rate*** [21]	*Predecessor BFS rate**** [3]	*Distance BFS rate*	*Distance BFS rate (sequential speedup*****)*	*Predecessor BFS rate (sequential speedup*****)*
europe.osm			0.00014	0.31 (11x)	0.31 (11x)
grid5pt.5000			0.00078	0.6 (7.4x)	0.57 (7.0x)
hugebubbles-00020			0.00061	0.43 (15x)	0.42 (15x)
grid7pt.300	0.12		0.012	1.1 (29x)	0.97 (26x)
nlpkkt160	0.47		0.21	2.5 (9.7x)	2.1 (8.2x)
audikw1			1.2	3.0 (4.6x)	2.5 (3.9x)
cage15	0.23		0.50	2.2 (18x)	1.9 (15x)
kkt_power	0.11		0.18	1.1 (23x)	1.0 (21x)
coPapersCiteseer			2.2	3.0 (6.0x)	2.5 (5.0x)
wikipedia-20070206	0.19		0.39	1.6 (25x)	1.4 (22x)
kron_g500-logn20			1.5	3.1 (13x)	2.5 (10x)
random.2Mv.128Me		0.50	1.2	3.0 (29x)	2.4 (23x)
rmat.2Mv.128Me		0.70	1.3	3.3 (22x)	2.6 (17x)

Table 2. Average single-socket graph traversal rates (10^9 TE/s). * NVIDIA 14-core 1.15 GHz Tesla C2050. ** Intel 4-core 2.5 GHz Core i7. *** Intel 8-core 2.7 GHz Xeon X5570. **** GPU speedup versus sequential method on Intel 3.4GHz Core i7 2600K.

Multi-GPU



C2050 x1
C2050 x2
C2050 x3
C2050 x4
10^9 edges / sec
normalized
0
1
2
3
4
5
6
7
8
9
0.0
0.2
0.4
0.6
0.8
1.0
1.2
europe.osm
grid5pt.5000
hugebubbles-00020
grid7pt.300
nlpkkt160
audikw1
cage15
kkt_power
coPapersCiteseer
wikipedia-20070206
kron_g500-logn20
random.2Mv.128Me
rmat.2Mv.128Me
Normalized hmean