GPU-based Graph Traversal on Compressed Graphs

Mo Sha[†], Yuchen Li[‡], Kian-Lee Tan[†]

[†]School of Computing, National University of Singapore
[‡]School of Information Systems, Singapore Management University

Presenter: Mo Sha
Jul 3rd, 2019

NUS
National University
of Singapore

Outline

- Introduction
- Compression Strategies Applied on the GPU
- Graph Traversal on Compressed Graphs on the GPU
- Evaluation
- Conclusion

Introduction

- Graph processing on the GPU offers attractive potential for performance boost.
- The high-bandwidth device memory on GPUs has limited capacity that constrains the size of the graph to be loaded on chip.
- GPU’s device memory is fixed and cannot be extended. The memory size of mainstream GPUs is typically at most 12 GB, and GPU with a large memory size is expensive.
- Existing solutions for graphs which exceed the device memory limit lead to severe performance degradation.

Compression Strategies Applied

- We execute the GPU workloads directly on a compressed graph and only decode necessary data into caches to limit the memory usage.
- We leverage on the massive computing power of GPUs to hide the decoding cost and retain the performance as that in the uncompressed scenario.

Use less device memory to store graphs on-chip:

- Virtual Node Compression (VNC)
- Node Re-ordering (NR)
- Compressed Graph Representation (CGR)

Compressed Graph Representation

- Intervals and Residuals Representation
- Gap Transformation
- Variable-Length Code (VLC) Encoding

Compressed Graph Representation

- Intervals and Residuals Representation
- Gap Transformation
- Variable-Length Code (VLC) Encoding

Example

Original Adjacency List:

16: 12, 18, 19, 20, 21, 24, 27, 28, 29, 101

Compressed Graph Representation

- ***Intervals and Residuals Representation***
- Gap Transformation
- Variable-Length Code (VLC) Encoding

Example

Original Adjacency List:

16: 12, **18, 19, 20, 21**, 24, **27, 28, 29**, 101

Compressed Graph Representation

- ***Intervals and Residuals Representation***
- Gap Transformation
- Variable-Length Code (VLC) Encoding

Example

Original Adjacency List:

16: 12, **18, 19, 20, 21**, 24, **27, 28, 29**, 101

Intervals and Residuals Representation:

16: 10,	2,	(18, 4),	(27, 3),	12,	24,	101
degNum	itvNum	itv0	itv1	res0	res1	res2

Compressed Graph Representation

- Intervals and Residuals Representation
- ***Gap Transformation***
- Variable-Length Code (VLC) Encoding

Example

Original Adjacency List:

16: 12, **18, 19, 20, 21**, 24, **27, 28, 29**, 101

Intervals and Residuals Representation:

16: 10,	2,	(18, 4),	(27, 3),	12,	24,	101
degNum	itvNum	itv0	itv1	res0	res1	res2

Gap Transformation:

16: 10, 2, (2, 4), (6, 3), -4, 12, 77

Compressed Graph Representation

- Intervals and Residuals Representation
- Gap Transformation
- ***Variable-Length Code (VLC) Encoding***

Example

Original Adjacency List:

16: 12, **18, 19, 20, 21**, 24, **27, 28, 29**, 101

Intervals and Residuals Representation:

16: 10,	2,	(18, 4),	(27, 3),	12,	24,	101
degNum	itvNum	itv0	itv1	res0	res1	res2

Gap Transformation:

16: 10, 2, (2, 4), (6, 3), -4, 12, 77

VLC Encoding:

0001010	010	00100	00100	00110	011	0001001	0001100	0000001001101

Compressed Graph Representation

- Intervals and Residuals Representation
- Gap Transformation
- Variable-Length Code (VLC) Encoding

Example

Original Adjacency List:

16: 12, **18, 19, 20, 21**, 24, **27, 28, 29**, 101

Compressed Graph Representation:

0001010	010	00100	00100	00110	011	0001001	0001100	00000001001101
10	2	2	4	6	3	-4	12	77

Compressed Graph Representation

- Intervals and Residuals Representation
- Gap Transformation
- Variable-Length Code (VLC) Encoding

Example

Original Adjacency List:

16: 12, **18, 19, 20, 21**, 24, **27, 28, 29**, 101

10 integer = 320 bits

Compressed Graph Representation:

0001010	010	00100	00100	00110	011	0001001	0001100	0000001001101
10	2	2	4	6	3	-4	12	77

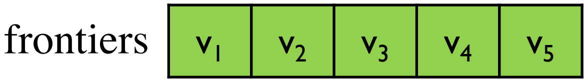
55 bits

Graph Traversal on GPU

- node-centric
- iterative ping-pong frontier queues
- expansion - filtering - contraction

Graph Traversal on GPU

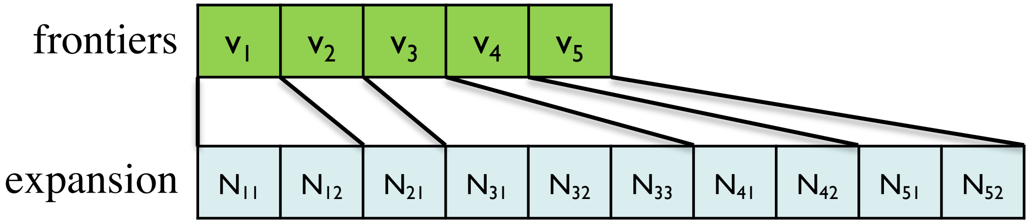
- node-centric
- iterative ping-pong frontier queues
- expansion - filtering - contraction



frontiers v_1 v_2 v_3 v_4 v_5

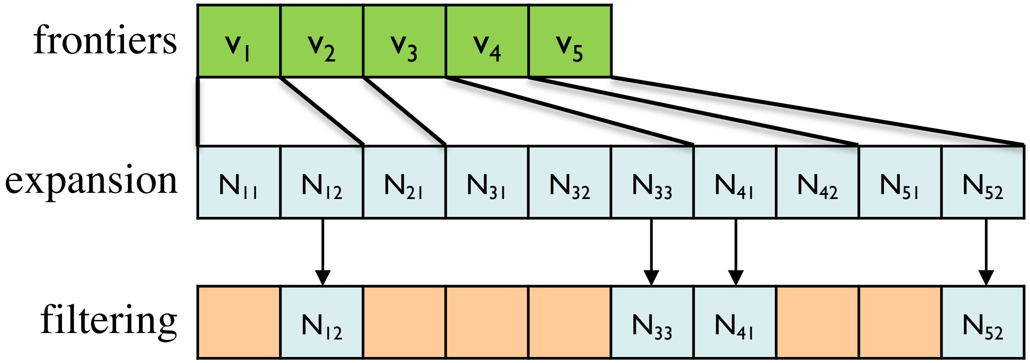
Graph Traversal on GPU

- node-centric
- iterative ping-pong frontier queues
- expansion - filtering - contraction



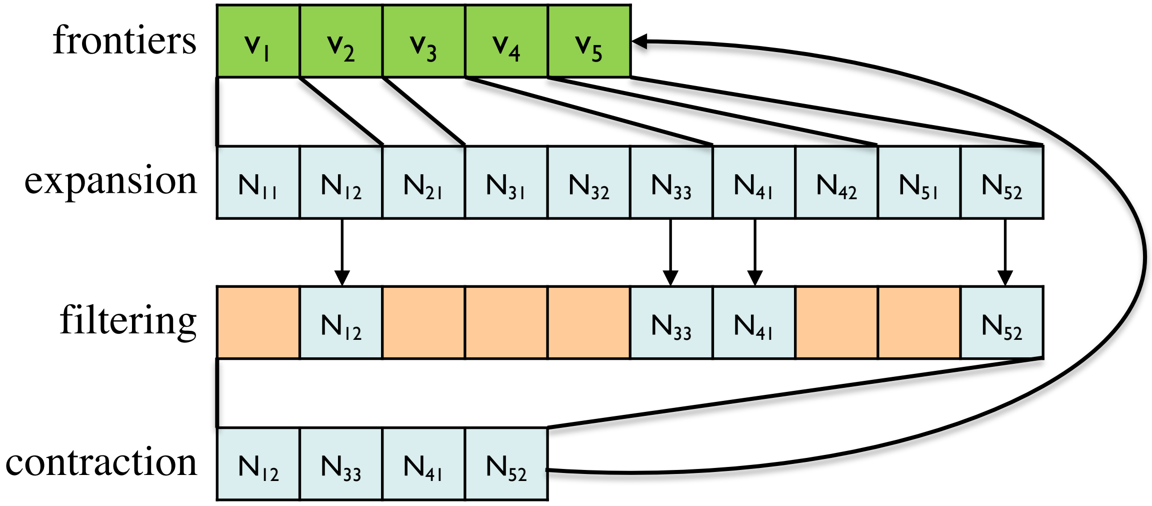
Graph Traversal on GPU

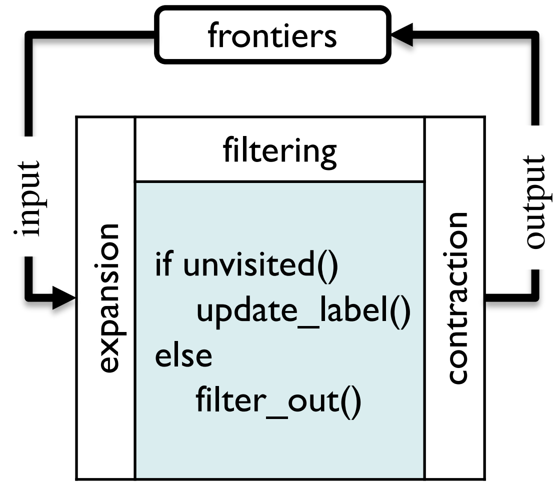
- node-centric
- iterative ping-pong frontier queues
- expansion - filtering - contraction



Graph Traversal on GPU

- node-centric
- iterative ping-pong frontier queues
- expansion - filtering - contraction

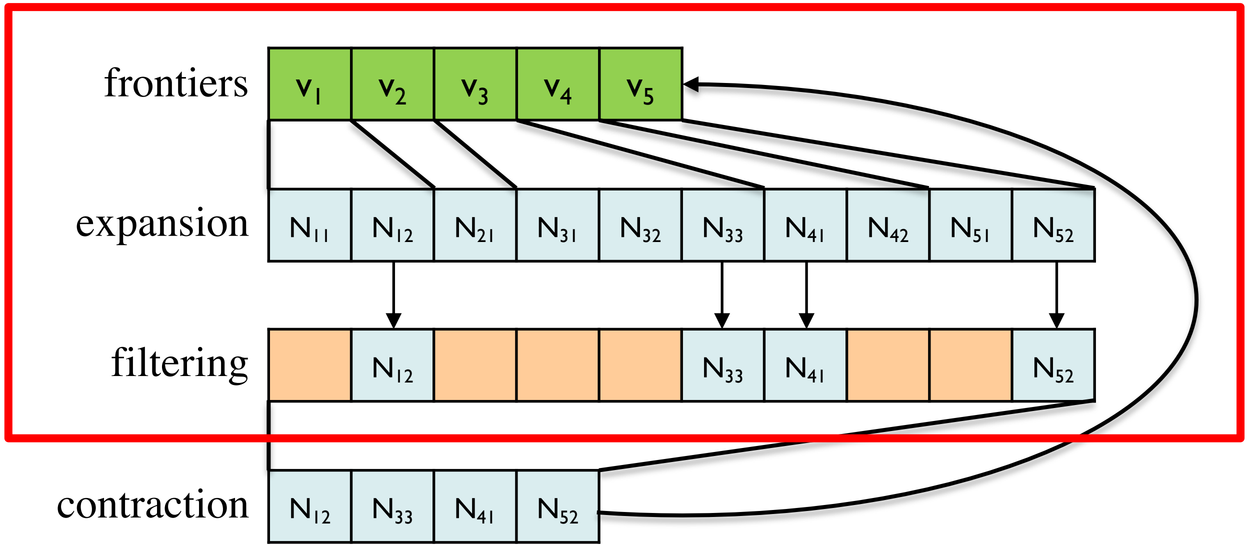


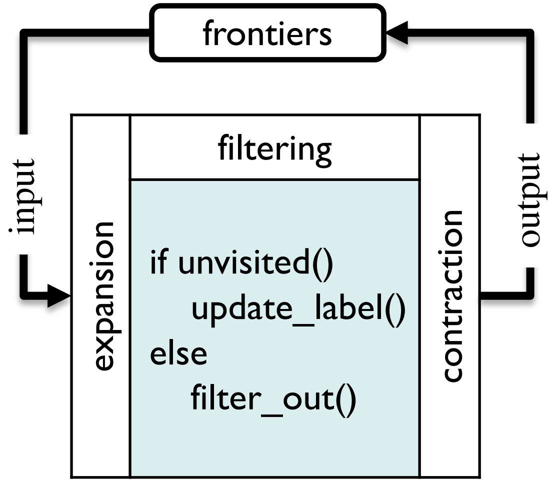


```
if unvisited()
    update_label()
else
    filter_out()
```

Graph Traversal on GPU

- node-centric
- iterative ping-pong frontier queues
- ***expansion*** - filtering - contraction





```
if unvisited()
    update_label()
else
    filter_out()
```

Graph Traversal on CGR

decoding an interval from VLC

decoding a residual from VLC

Graph Traversal on CGR

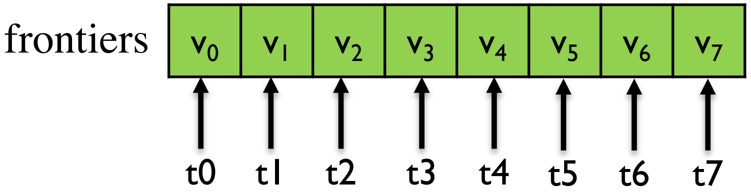
decoding an interval from VLC

handling a neighbor of an interval

decoding a residual from VLC

handling a residual

Graph Traversal on CGR

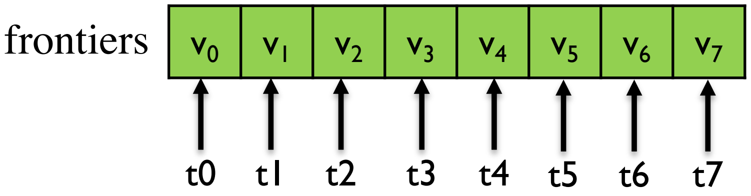


frontiers

v_0 v_1 v_2 v_3 v_4 v_5 v_6 v_7

t0 t1 t2 t3 t4 t5 t6 t7

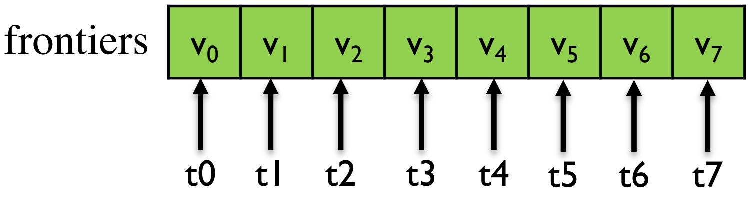
Graph Traversal on CGR



t0	degNum=6	itvNum=1	itv0:len=4	res0	res1			
t1	degNum=1	itvNum=0	res0					
t2	degNum=14	itvNum=1	itv0:len=11	res0	res1	res2		
t3	degNum=2	itvNum=0	res0	res1				
t4	degNum=1	itvNum=0	res0					
t5	degNum=11	itvNum=1	itv0:len=7	res0	res1	res2	res3
t6	degNum=1	itvNum=0	res0					
t7	degNum=1	itvNum=0	res0					

- tX:iY decoding Y-th interval of the node assigned to thread-X
- tX:iY:Z handling Z-th neighbour of Y-th interval of the node assigned to thread-X
- tX:resY decoding Y-th residual of the node assigned to thread-X
- tX:resY handling Y-th residual of the node assigned to thread-X
- ⬜ idle (warp divergence)

Graph Traversal on CGR



t0	degNum=6	itvNum=1	itv0:len=4	res0	res1		
t1	degNum=1	itvNum=0	res0				
t2	degNum=14	itvNum=1	itv0:len=11	res0	res1	res2	
t3	degNum=2	itvNum=0	res0	res1			
t4	degNum=1	itvNum=0	res0				
t5	degNum=11	itvNum=1	itv0:len=7	res0	res1	res2	res3
t6	degNum=1	itvNum=0	res0				
t7	degNum=1	itvNum=0	res0				

tX:iY	decoding Y-th interval of the node assigned to thread-X
tX:iY:Z	handling Z-th neighbour of Y-th interval of the node assigned to thread-X
tX:resY	decoding Y-th residual of the node assigned to thread-X
tX:resY	handling Y-th residual of the node assigned to thread-X
	idle (warp divergence)

step	t0	t1	t2	t3	t4	t5	t6	t7
0	t0:i0		t2:i0			t5:i0		
1		t1:res0		t3:res0	t4:res0		t6:res0	t7:res0
2	t0:i0:0	t1:res0	t2:i0:0	t3:res0	t4:res0	t5:i0:0	t6:res0	t7:res0
3				t3:res1				
4	t0:i0:1		t2:i0:1	t3:res1		t5:i0:1		
5	t0:i0:2		t2:i0:2			t5:i0:2		
6	t0:i0:3		t2:i0:3			t5:i0:3		
7	t0:res0							
8	t0:res0		t2:i0:4			t5:i0:4		
9	t0:res1							
10	t0:res1		t2:i0:5			t5:i0:5		
11			t2:i0:6			t5:i0:6		
12						t5:res0		
13			t2:i0:7			t5:res0		
14						t5:res1		
15			t2:i0:8			t5:res1		
16						t5:res2		
17			t2:i0:9			t5:res2		
18						t5:res3		
19			t2:i0:10			t5:res3		
20			t2:res0					
21			t2:res0					
22			t2:res1					
23			t2:res1					
24			t2:res2					
25			t2:res2					

Optimization Mechanism

- Two-Phase Traversal
- Task Stealing
- Warp-centric Decoding
- Residual Segmentation

Two-Phase Traversal

- Interval Phase and Residual Phase
- Handling continuous neighbors of an interval cooperatively

t0	degNum=6	itvNum=1	itv0:len=4	res0	res1		
t1	degNum=1	itvNum=0	res0				
t2	degNum=14	itvNum=1	itv0:len=11	res0	res1	res2	
t3	degNum=2	itvNum=0	res0	res1			
t4	degNum=1	itvNum=0	res0				
t5	degNum=11	itvNum=1	itv0:len=7	res0	res1	res2	res3
t6	degNum=1	itvNum=0	res0				
t7	degNum=1	itvNum=0	res0				

step	t0	t1	t2	t3	t4	t5	t6	t7
0	t0:i0		t2:i0			t5:i0		
1	t2:i0:0	t2:i0:1	t2:i0:2	t2:i0:3	t2:i0:4	t2:i0:5	t2:i0:6	t2:i0:7
2	t0:i0:0	t0:i0:1	t0:i0:2	t0:i0:3	t2:i0:8	t2:i0:9	t2:i0:10	t5:i0:0
3	t5:i0:1	t5:i0:2	t5:i0:3	t5:i0:4	t5:i0:5	t5:i0:6		
4								
5								
6								
7								
8								
9								
10								
11								

tX:iY decoding Y-th interval of the node assigned to thread-X

tX:iY:Z handling Z-th neighbour of Y-th interval of the node assigned to thread-X

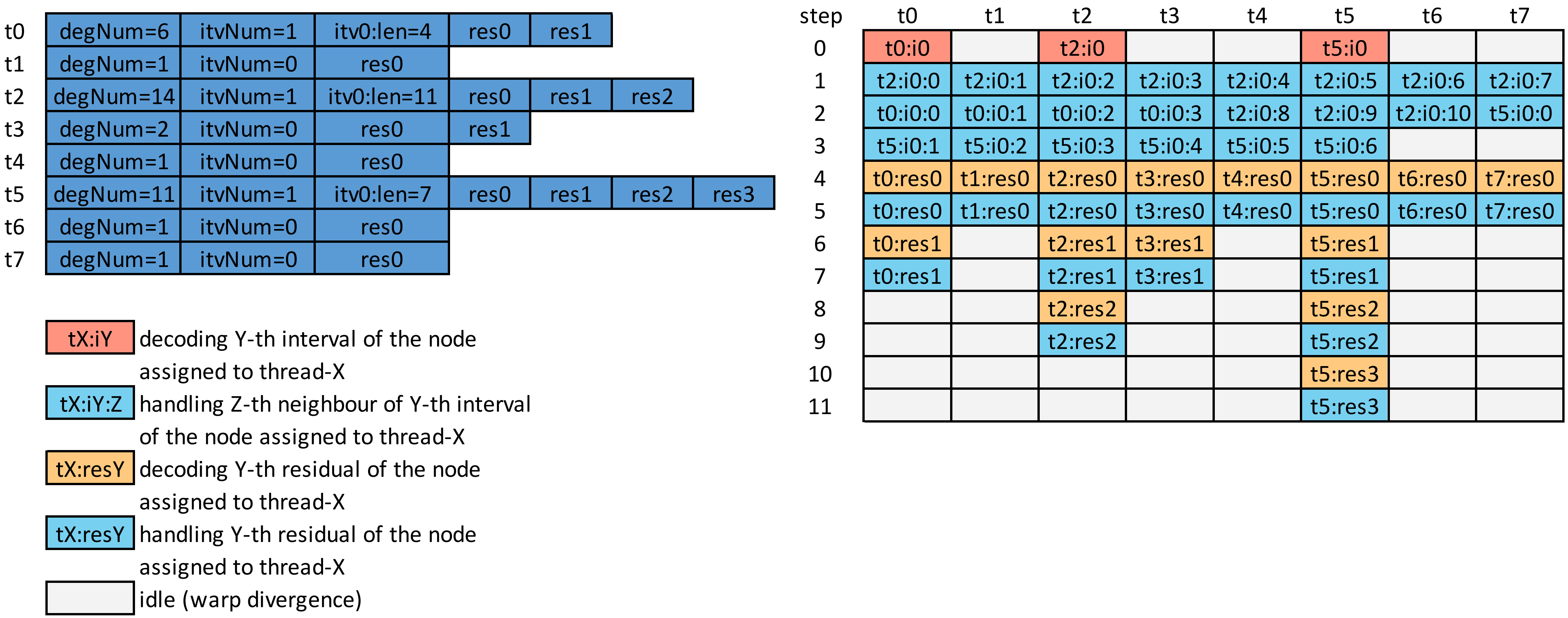
tX:resY decoding Y-th residual of the node assigned to thread-X

tX:resY handling Y-th residual of the node assigned to thread-X

□ idle (warp divergence)

Two-Phase Traversal

- Interval Phase and Residual Phase
- Handling continuous neighbors of an interval cooperatively



t0	degNum=6	itvNum=1	itv0:len=4	res0	res1		
t1	degNum=1	itvNum=0	res0				
t2	degNum=14	itvNum=1	itv0:len=11	res0	res1	res2	
t3	degNum=2	itvNum=0	res0	res1			
t4	degNum=1	itvNum=0	res0				
t5	degNum=11	itvNum=1	itv0:len=7	res0	res1	res2	res3
t6	degNum=1	itvNum=0	res0				
t7	degNum=1	itvNum=0	res0				

tX:iY	decoding Y-th interval of the node assigned to thread-X
tX:iY:Z	handling Z-th neighbour of Y-th interval of the node assigned to thread-X
tX:resY	decoding Y-th residual of the node assigned to thread-X
tX:resY	handling Y-th residual of the node assigned to thread-X
□	idle (warp divergence)

step	t0	t1	t2	t3	t4	t5	t6	t7
0	t0:i0		t2:i0			t5:i0		
1	t2:i0:0	t2:i0:1	t2:i0:2	t2:i0:3	t2:i0:4	t2:i0:5	t2:i0:6	t2:i0:7
2	t0:i0:0	t0:i0:1	t0:i0:2	t0:i0:3	t2:i0:8	t2:i0:9	t2:i0:10	t5:i0:0
3	t5:i0:1	t5:i0:2	t5:i0:3	t5:i0:4	t5:i0:5	t5:i0:6		
4	t0:res0	t1:res0	t2:res0	t3:res0	t4:res0	t5:res0	t6:res0	t7:res0
5	t0:res0	t1:res0	t2:res0	t3:res0	t4:res0	t5:res0	t6:res0	t7:res0
6	t0:res1		t2:res1	t3:res1		t5:res1		
7	t0:res1		t2:res1	t3:res1		t5:res1		
8			t2:res2			t5:res2		
9			t2:res2			t5:res2		
10						t5:res3		
11						t5:res3		

Two-Phase Traversal

- Interval Phase and Residual Phase
- Handling continuous neighbors of an interval cooperatively

t0	degNum=6	itvNum=1	itv0:len=4	res0	res1		
t1	degNum=1	itvNum=0	res0				
t2	degNum=14	itvNum=1	itv0:len=11	res0	res1	res2	
t3	degNum=2	itvNum=0	res0	res1			
t4	degNum=1	itvNum=0	res0				
t5	degNum=11	itvNum=1	itv0:len=7	res0	res1	res2	res3
t6	degNum=1	itvNum=0	res0				
t7	degNum=1	itvNum=0	res0				

tX:iY	decoding Y-th interval of the node assigned to thread-X
tX:iY:Z	handling Z-th neighbour of Y-th interval of the node assigned to thread-X
tX:resY	decoding Y-th residual of the node assigned to thread-X
tX:resY	handling Y-th residual of the node assigned to thread-X
	idle (warp divergence)

step	t0	t1	t2	t3	t4	t5	t6	t7
0	t0:i0		t2:i0			t5:i0		
1	t2:i0:0	t2:i0:1	t2:i0:2	t2:i0:3	t2:i0:4	t2:i0:5	t2:i0:6	t2:i0:7
2	t0:i0:0	t0:i0:1	t0:i0:2	t0:i0:3	t2:i0:8	t2:i0:9	t2:i0:10	t5:i0:0
3	t5:i0:1	t5:i0:2	t5:i0:3	t5:i0:4	t5:i0:5	t5:i0:6		
4	t0:res0	t1:res0	t2:res0	t3:res0	t4:res0	t5:res0	t6:res0	t7:res0
5	t0:res0	t1:res0	t2:res0	t3:res0	t4:res0	t5:res0	t6:res0	t7:res0
6	t0:res1		t2:res1	t3:res1		t5:res1		
7	t0:res1		t2:res1	t3:res1		t5:res1		
8			t2:res2			t5:res2		
9			t2:res2			t5:res2		
10						t5:res3		
11						t5:res3		

Task Stealing

step	t0	t1	t2	t3	t4	t5	t6	t7
0	t0:i0		t2:i0			t5:i0		
1		t1:res0		t3:res0	t4:res0		t6:res0	t7:res0
2	t0:i0:0	t1:res0	t2:i0:0	t3:res0	t4:res0	t5:i0:0	t6:res0	t7:res0
3				t3:res1				
4	t0:i0:1		t2:i0:1	t3:res1		t5:i0:1		
5	t0:i0:2		t2:i0:2			t5:i0:2		
6	t0:i0:3		t2:i0:3			t5:i0:3		
7	t0:res0							
8	t0:res0		t2:i0:4			t5:i0:4		
9	t0:res1							
10	t0:res1		t2:i0:5			t5:i0:5		
11			t2:i0:6			t5:i0:6		
12						t5:res0		
13			t2:i0:7			t5:res0		
14						t5:res1		
15			t2:i0:8			t5:res1		
16						t5:res2		
17			t2:i0:9			t5:res2		
18						t5:res3		
19			t2:i0:10			t5:res3		
20			t2:res0					
21			t2:res0					
22			t2:res1					
23			t2:res1					
24			t2:res2					
25			t2:res2					

step	t0	t1	t2	t3	t4	t5	t6	t7
0	t0:i0		t2:i0			t5:i0		
1	t2:i0:0	t2:i0:1	t2:i0:2	t2:i0:3	t2:i0:4	t2:i0:5	t2:i0:6	t2:i0:7
2	t0:i0:0	t0:i0:1	t0:i0:2	t0:i0:3	t2:i0:8	t2:i0:9	t2:i0:10	t5:i0:0
3	t5:i0:1	t5:i0:2	t5:i0:3	t5:i0:4	t5:i0:5	t5:i0:6		
4	t0:res0	t1:res0	t2:res0	t3:res0	t4:res0	t5:res0	t6:res0	t7:res0
5	t0:res0	t1:res0	t2:res0	t3:res0	t4:res0	t5:res0	t6:res0	t7:res0
6	t0:res1		t2:res1	t3:res1		t5:res1		
7	t0:res1		t2:res1	t3:res1		t5:res1		
8			t2:res2			t5:res2		
9			t2:res2			t5:res2		
10						t5:res3		
11						t5:res3		

step	t0	t1	t2	t3	t4	t5	t6	t7
0	t0:i0		t2:i0			t5:i0		
1	t2:i0:0	t2:i0:1	t2:i0:2	t2:i0:3	t2:i0:4	t2:i0:5	t2:i0:6	t2:i0:7
2	t0:i0:0	t0:i0:1	t0:i0:2	t0:i0:3	t2:i0:8	t2:i0:9	t2:i0:10	t5:i0:0
3	t5:i0:1	t5:i0:2	t5:i0:3	t5:i0:4	t5:i0:5	t5:i0:6		
4	t0:res0	t1:res0	t2:res0	t3:res0	t4:res0	t5:res0	t6:res0	t7:res0
5	t0:res0	t1:res0	t2:res0	t3:res0	t4:res0	t5:res0	t6:res0	t7:res0
6	t0:res1		t2:res1	t3:res1		t5:res1		
7			t2:res2			t5:res2		
8						t5:res3		
9	t0:res1	t2:res1	t2:res2	t3:res1	t5:res1	t5:res2	t5:res3	

Task Stealing

step	t0	t1	t2	t3	t4	t5	t6	t7
0	t0:i0		t2:i0			t5:i0		
1		t1:res0		t3:res0	t4:res0		t6:res0	t7:res0
2	t0:i0:0	t1:res0	t2:i0:0	t3:res0	t4:res0	t5:i0:0	t6:res0	t7:res0
3				t3:res1				
4	t0:i0:1		t2:i0:1	t3:res1		t5:i0:1		
5	t0:i0:2		t2:i0:2			t5:i0:2		
6	t0:i0:3		t2:i0:3			t5:i0:3		
7	t0:res0							
8	t0:res0		t2:i0:4			t5:i0:4		
9	t0:res1							
10	t0:res1		t2:i0:5			t5:i0:5		
11			t2:i0:6			t5:i0:6		
12						t5:res0		
13			t2:i0:7			t5:res0		
14						t5:res1		
15			t2:i0:8			t5:res1		
16						t5:res2		
17			t2:i0:9			t5:res2		
18						t5:res3		
19			t2:i0:10			t5:res3		
20			t2:res0					
21			t2:res0					
22			t2:res1					
23			t2:res1					
24			t2:res2					
25			t2:res2					

step	t0	t1	t2	t3	t4	t5	t6	t7
0	t0:i0		t2:i0			t5:i0		
1	t2:i0:0	t2:i0:1	t2:i0:2	t2:i0:3	t2:i0:4	t2:i0:5	t2:i0:6	t2:i0:7
2	t0:i0:0	t0:i0:1	t0:i0:2	t0:i0:3	t2:i0:8	t2:i0:9	t2:i0:10	t5:i0:0
3	t5:i0:1	t5:i0:2	t5:i0:3	t5:i0:4	t5:i0:5	t5:i0:6		
4	t0:res0	t1:res0	t2:res0	t3:res0	t4:res0	t5:res0	t6:res0	t7:res0
5	t0:res0	t1:res0	t2:res0	t3:res0	t4:res0	t5:res0	t6:res0	t7:res0
6	t0:res1		t2:res1	t3:res1		t5:res1		
7	t0:res1		t2:res1	t3:res1		t5:res1		
8			t2:res2			t5:res2		
9			t2:res2			t5:res2		
10						t5:res3		
11						t5:res3		

step	t0	t1	t2	t3	t4	t5	t6	t7
0	t0:i0		t2:i0			t5:i0		
1	t2:i0:0	t2:i0:1	t2:i0:2	t2:i0:3	t2:i0:4	[illegible]	[illegible]	t2:i0:7
2	t0:i0:0	t0:i0:1	t0:i0:2	t0:i0:3	[illegible]	[illegible]	[illegible]	t5:i0:0
3	t5:i0:1	t5:i0:2	t5:i0:3	t5:i0:4	[illegible]	[illegible]		
4	t0:res0	t1:res0	t2:res0	t3:res0	t4:res0	t5:res0	t6:res0	t7:res0
5	t0:res0	t1:res0	t2:res0	t3:res0	t4:res0	t5:res0	t6:res0	t7:res0
6	t0:res1		t2:res1	t3:res1		t5:res1		
7			t2:res2			t5:res2		
8						t5:res3		
9	t0:res1	t2:res1	t2:res2	t3:res1	t5:res1	t5:res2	t5:res3	

bottleneck

Warp-centric Decoding

Original Adjacency List:

16: 12, **18, 19, 20, 21**, 24, **27, 28, 29**, 101

Intervals and Residuals Representation:

16:	10,	2,	(18, 4),	(27, 3),	12,	24,	101
	degNum	itvNum	itv0	itv1	res0	res1	res2

Gap Transformation:

16: 10, 2, (2, 4), (6, 3), -4, 12, 77

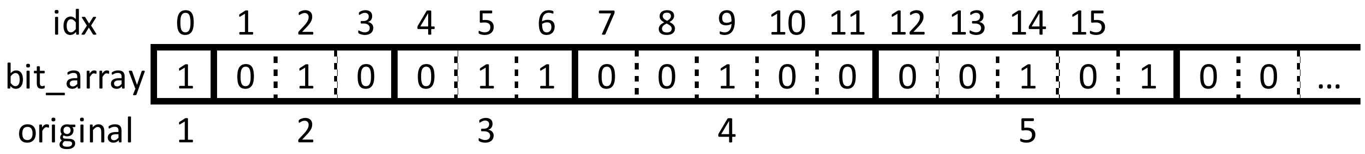
VLC Encoding:

0001010	010	00100	00100	00110	011	0001001	0001100	0000001001101
10	2	2	4	6	3	-4	12	77

step	t0	t1	t2	t3	t4	t5	t6	t7
0	t0:i0		t2:i0			[illegible]	[illegible]	
1	t2:i0:0	t2:i0:1	t2:i0:2	t2:i0:3	t2[illegible]	[illegible]	[illegible]	t2:i0:7
2	t0:i0:0	t0:i0:1	t0:i0:2	t0:i0:3	[illegible]	[illegible]	[illegible]	t5:i0:0
3	t5:i0:1	t5:i0:2	t5:i0:3	t5:i0:[illegible]	[illegible]	[illegible]		
4	t0:res0	t1:res0	t2:res0	t3:res0	t4:res0	t5:res0	t6:res0	t7:res0
5	t0:res0	t1:res0	t2:res0	t3:res0	t4:res0	t5:res0	t6:res0	t7:res0
6	t0:res1		t2:res1	t3:res1		t5:res1		
7			t2:res2			t5:res2		
8						t5:res3		
9	t0:res1	t2:res1	t2:res2	t3:res1	t5:res1	t5:res2	t5:res3	

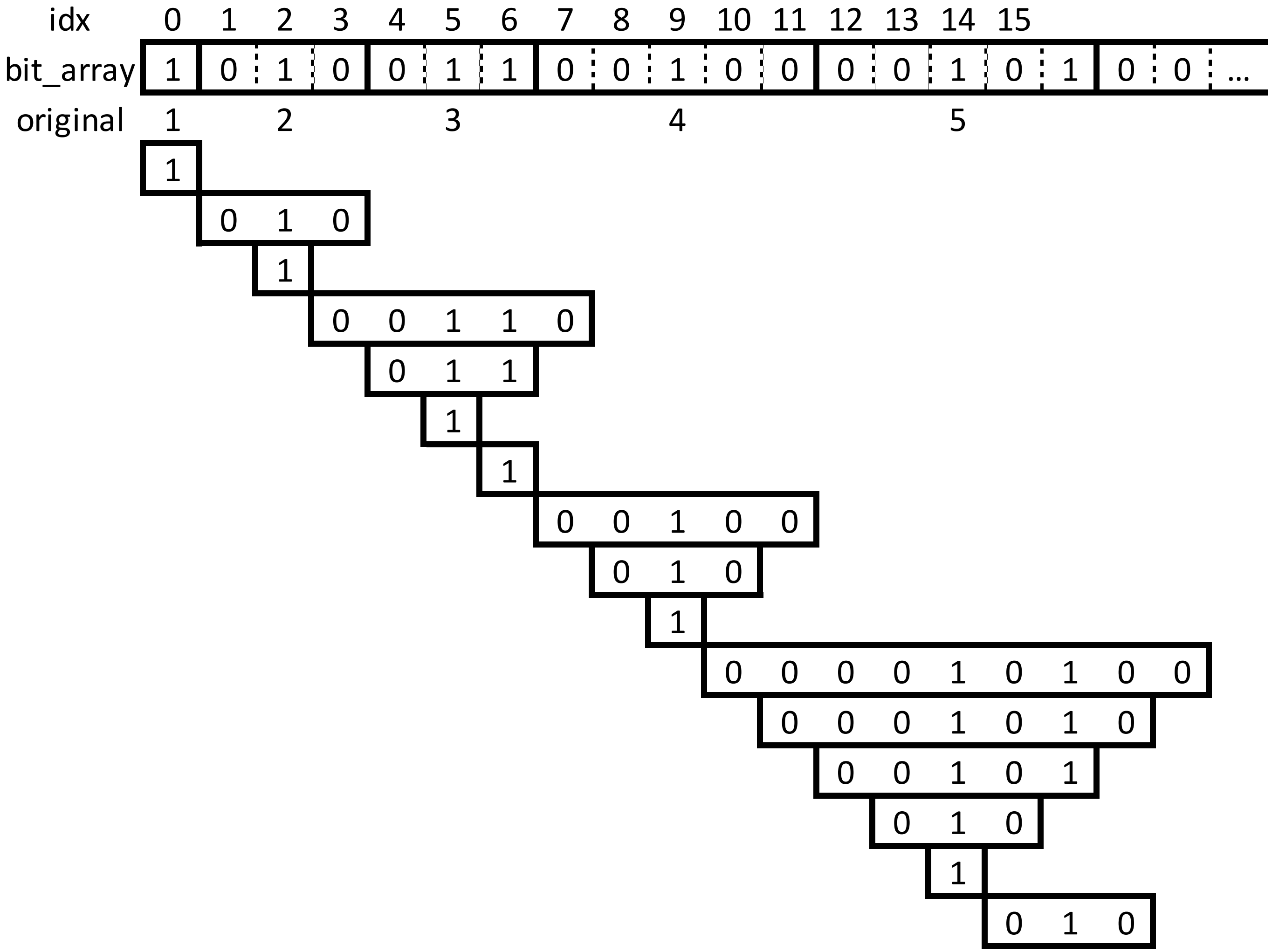
bottleneck

Warp-centric Decoding



idx	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15				
bit_array	1	0	1	0	0	1	1	0	0	1	0	0	0	0	1	0	1	0	0	...
original	1		2			3				4					5					

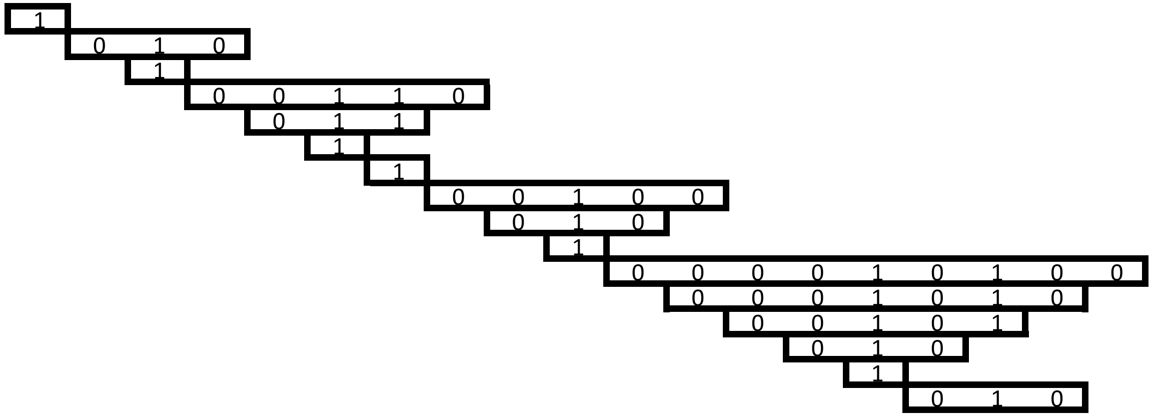
Warp-centric Decoding



idx	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15				
bit_array	1	0	1	0	0	1	1	0	0	1	0	0	0	0	1	0	1	0	0	...
original	1		2			3				4					5					
	1																			
		0	1	0																
			1																	
				0	0	1	1	0												
					0	1	1													
						1														
							1													
								0	0	1	0	0								
									0	1	0									
										1										
											0	0	0	0	1	0	1	0	0	
												0	0	0	1	0	1	0		
													0	0	1	0	1			
														0	1	0				
															1					
																0	1	0		

Warp-centric Decoding

idx	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15				
bit_array	1	0	1	0	0	1	1	0	0	1	0	0	0	0	1	0	1	0	0	...
original	1		2			3				4					5					

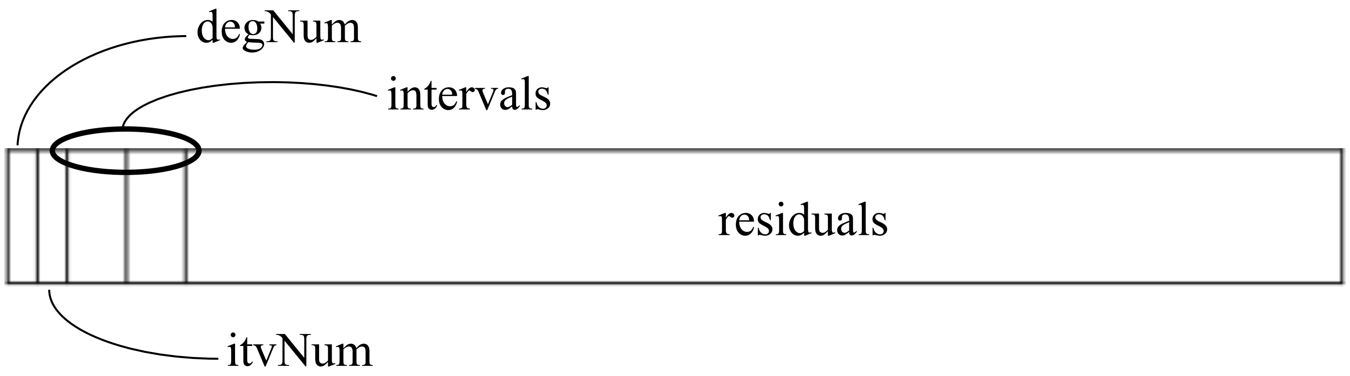


idx	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
vals	1	2	1	6	3	1	1	4	2	1	20	10	5	2	1	2	
poss	1	4	3	8	7	6	7	12	11	10	19	18	17	16	15	18	Round 0
flags	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
poss	4	7	8	11	12	7	12	17	18	19	x	x	x	x	18	x	Round 1
flags	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
poss	12	17	18	x	x	17	x	x	x	x	x	x	x	x	x	x	Round 2
flags	1	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	
poss	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	Round 3
flags	1	1	0	0	1	0	0	1	0	0	0	0	1	0	0	0	

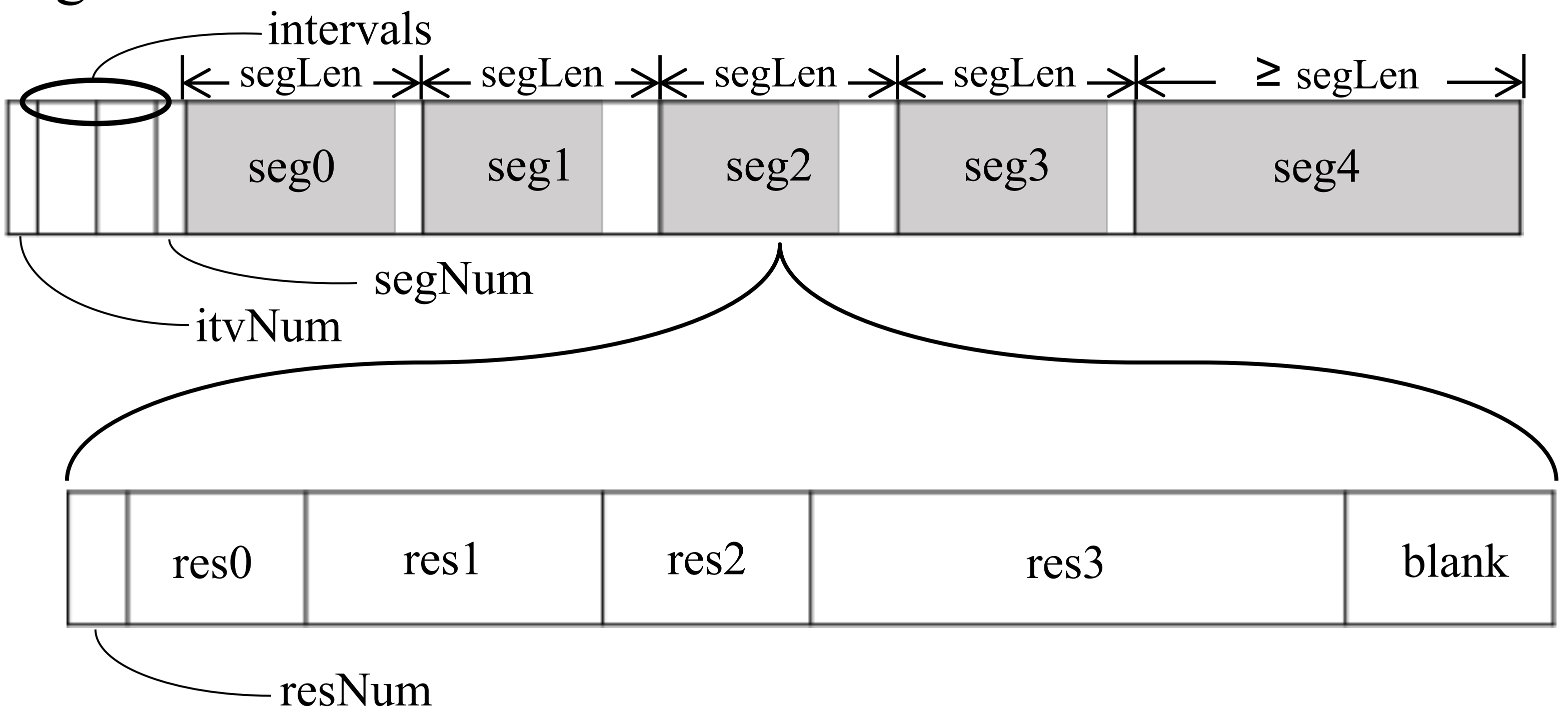
$O(\log_2 T)$

Residual Segmentation

Unsegmented:



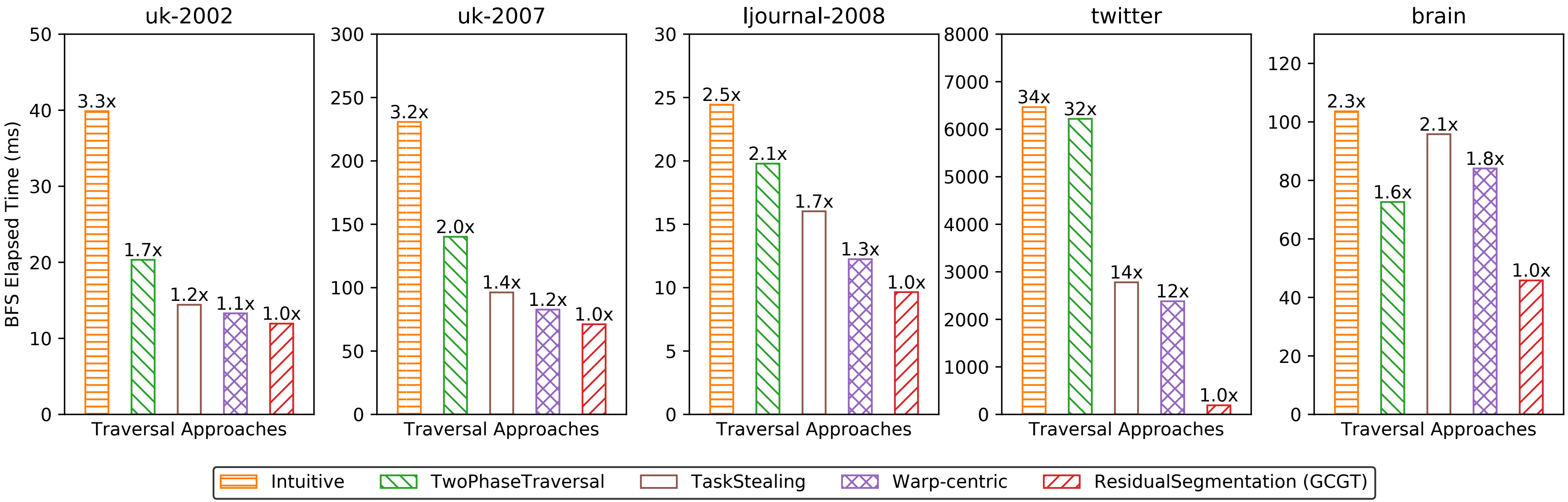
Segmented:



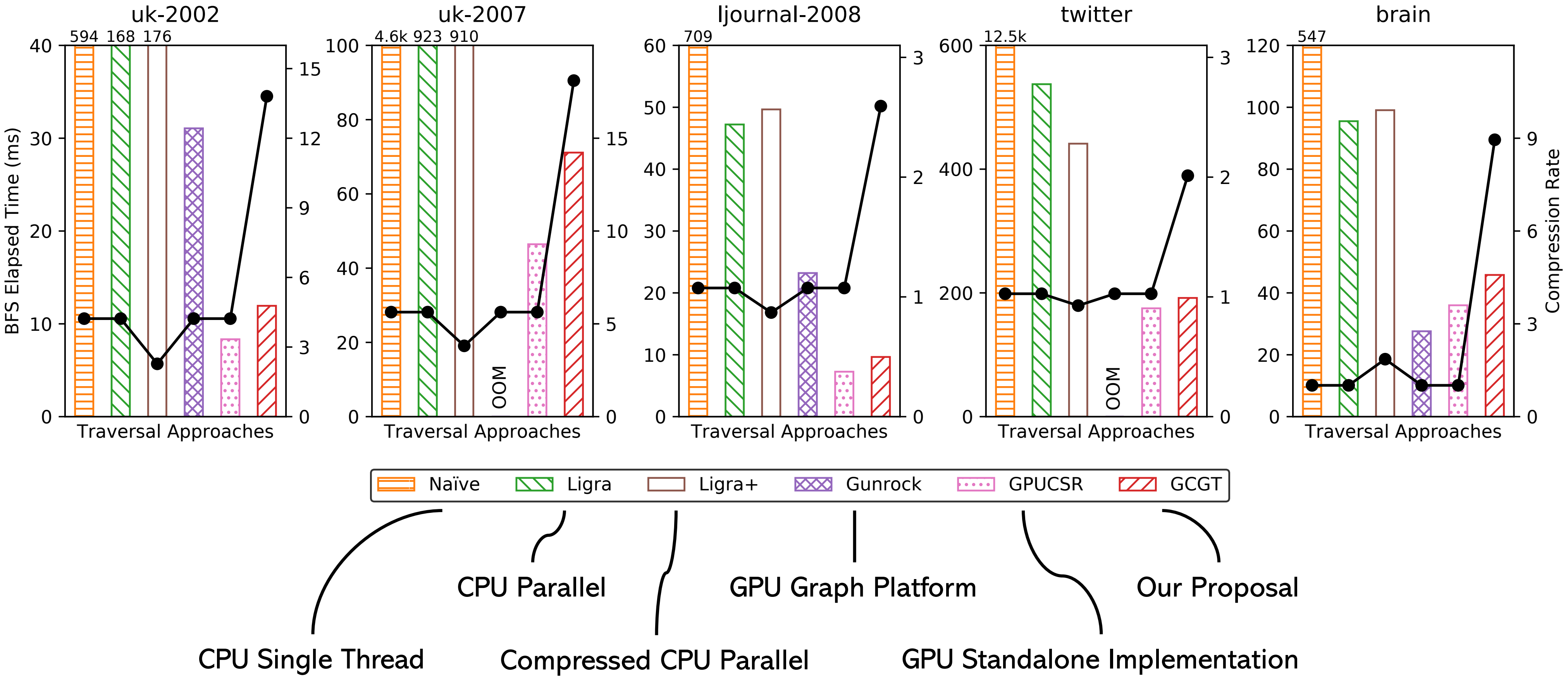
Evaluation Datasets

Dataset	Category	$\lvert V\rvert$	$\lvert E\rvert$
uk-2002	Web	18.5M	298M
uk-2007	Web	105M	3.73B
ljournal	Social Network	5.3M	79M
twitter	Social Network	41.6M	1.46B
brain	Biology	784K	267M

Optimization Impact



Baselines Comparison



CPU Parallel

GPU Graph Platform

Our Proposal

CPU Single Thread

Compressed CPU Parallel

GPU Standalone Implementation

Conclusion

- We introduce compressed graphs into GPU-based graph computation for optimized device memory usage.
- Designed towards GPU's SIMT architecture, we propose a series of implementations when decoding the compressed graph.
- Extensive experiments show that with 2x-18x compression rate, our proposed GPU-based graph traversal on compressed graphs achieves competitive efficiency compared with the state-of-the-art graph traversal approaches on non-compressed graphs.

Thank you!