Parallel k-Core Maintenance in Dynamic Graphs

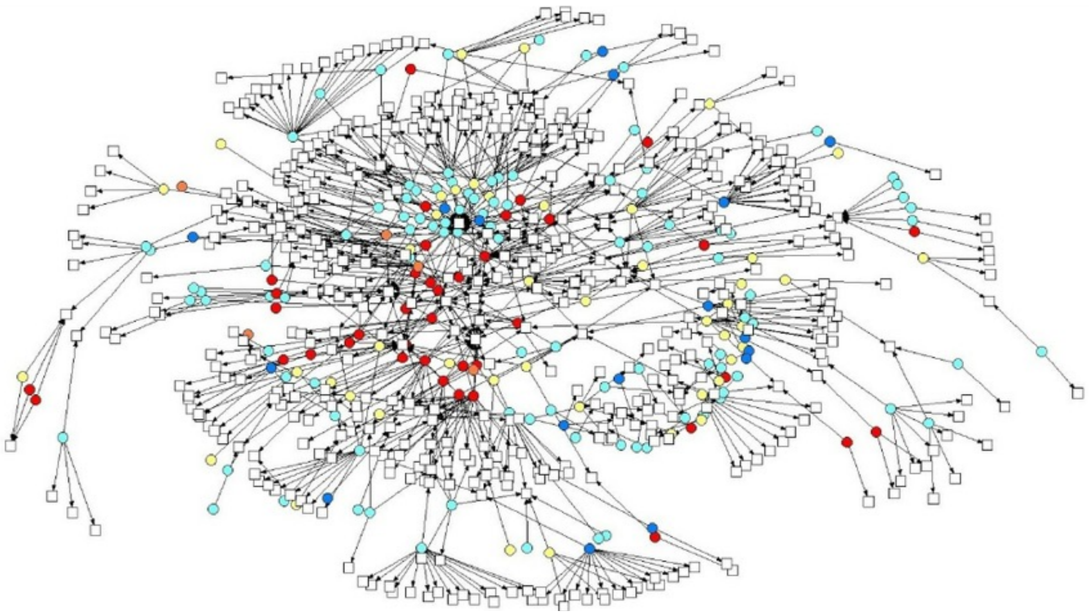
Ph.D. Thesis Defense by Bin Guo

Apr. 19, 2023

Committee Members:
- Dr. Terry Todd (Chair)
- Dr. Emil Sekerinski
- Dr. Ryszard Janicki
- Dr. Nedialko Nedialkov
- Dr. Richard Paige
- Dr. Maryam Mehri Dehnavi (University of Toronto)

Motivation

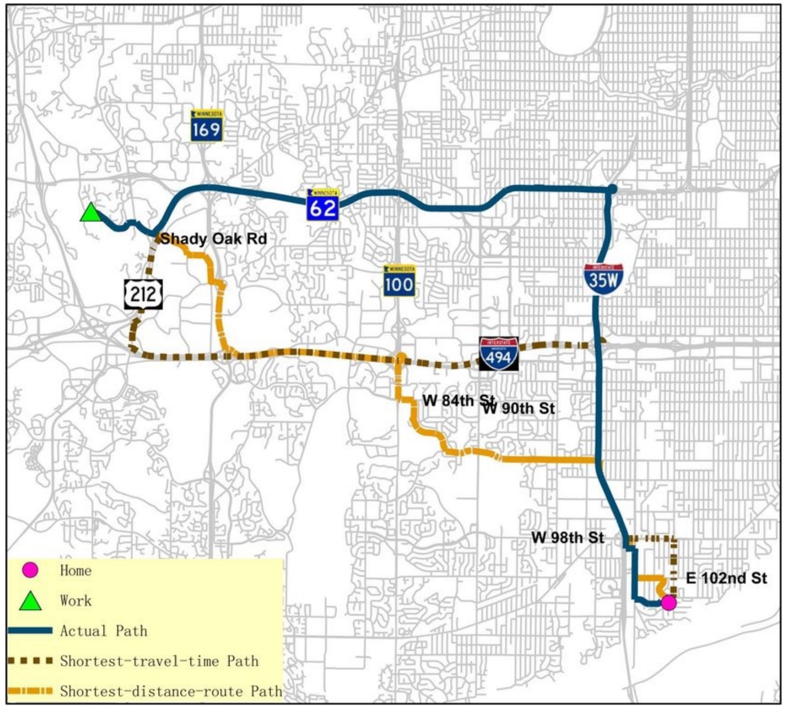
- Graphs are important data structures used in many applications:
  - Social Networks: Facebook, Twitter
  - Knowledge Network: DBpedia
  - Biological Networks and Road Network
- The data graphs are growing larger and larger:
  - Facebook has **2.9 billion** active users
  - Dbpedia has **6.6 million** entities and **13 billion** pieces of information



Visualizations of Social Networks show the employee interactions [1]

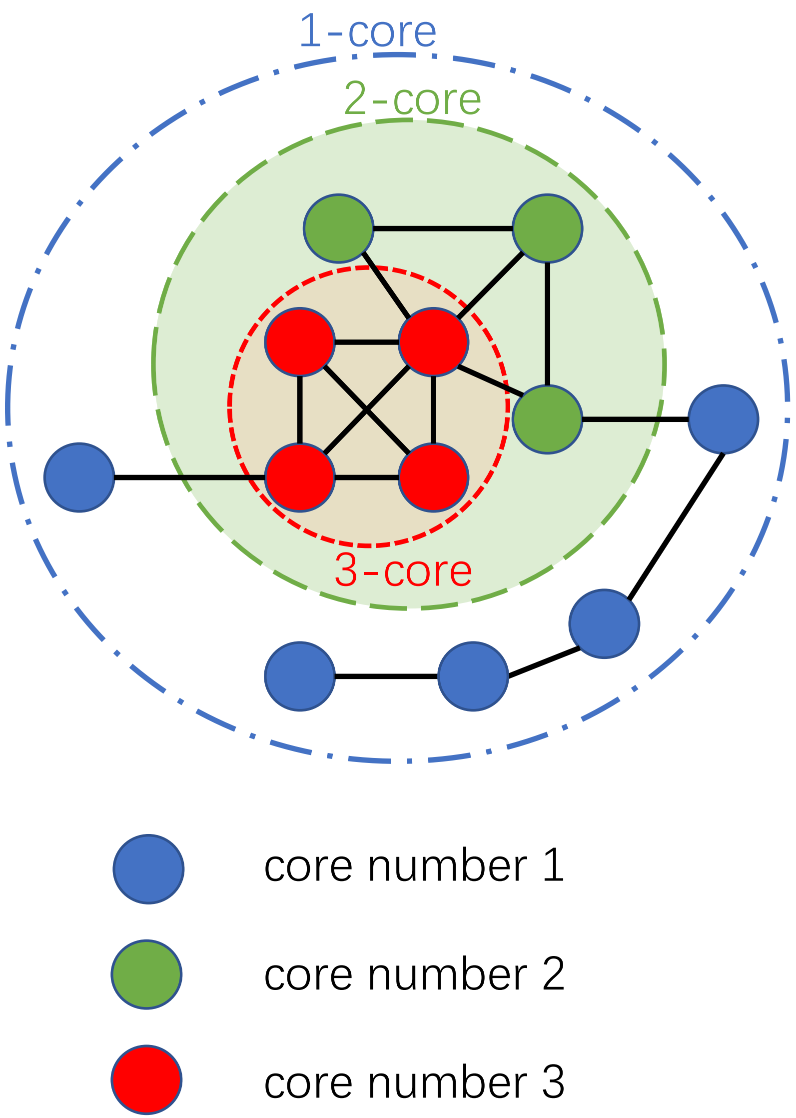
Graph Analytics

- Large Data graphs require data analytics
- Graph databases:
  - **Neo4j**
  - Microsoft SQL Server
  - Amazon Neptune
- Graph algorithms:
  - Strongly Connected Components,
  - Minimum Spanning Forest
  - Shortest Path Distance
  - k-Core



k-Core Decomposition

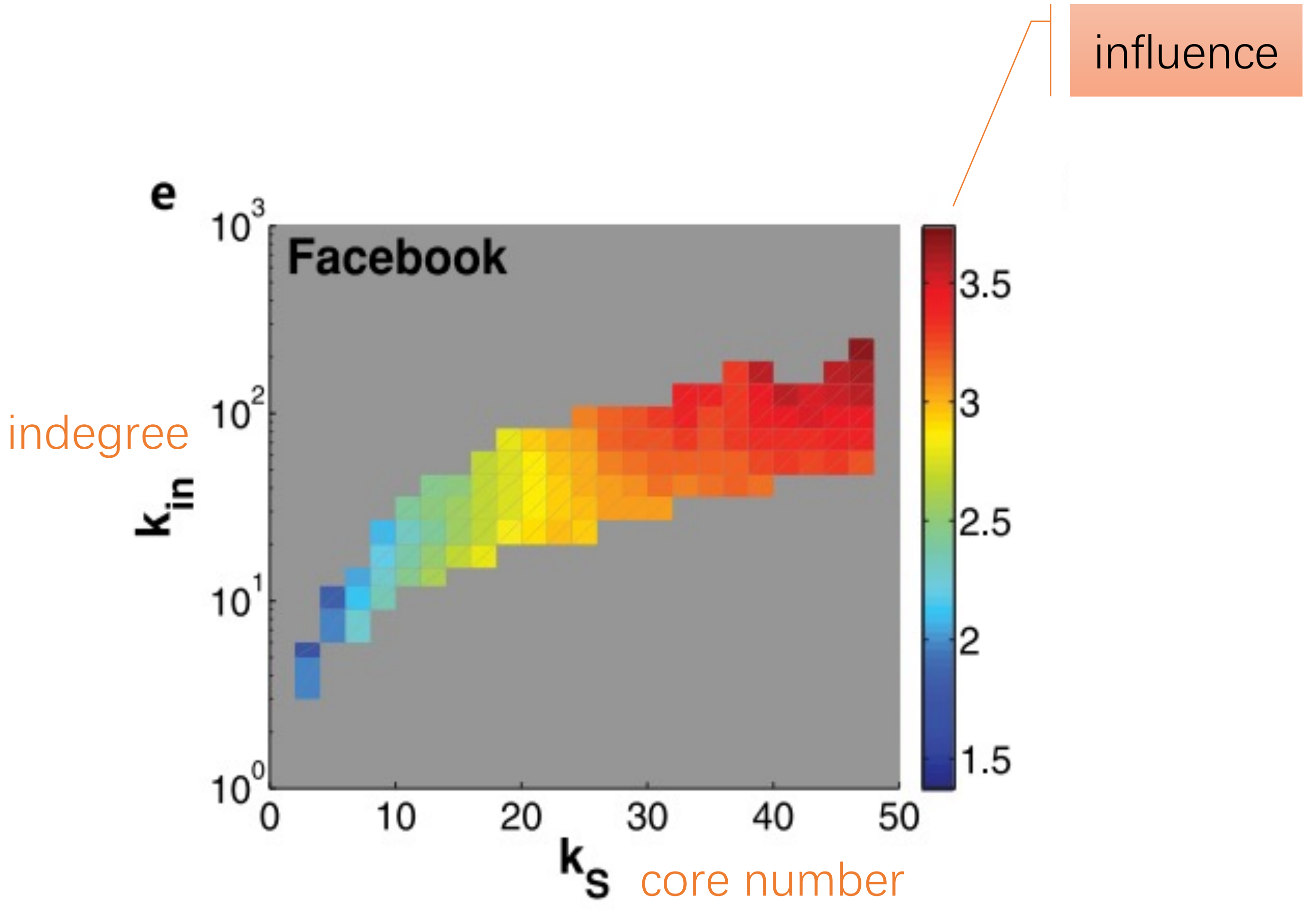
- It is to find the largest subgraph, in which each node has at least k neighbors in the induced subgraph
- The **core number** is the largest value of k



Applications in Social Networks

Social Networks, e.g. Facebook and Twitter	
Vertices	individuals
Edges	relations

- The core numbers of vertices can predict the average influence of spreading [1]
- For vertices, larger core numbers and larger indgrees indicate higher influence



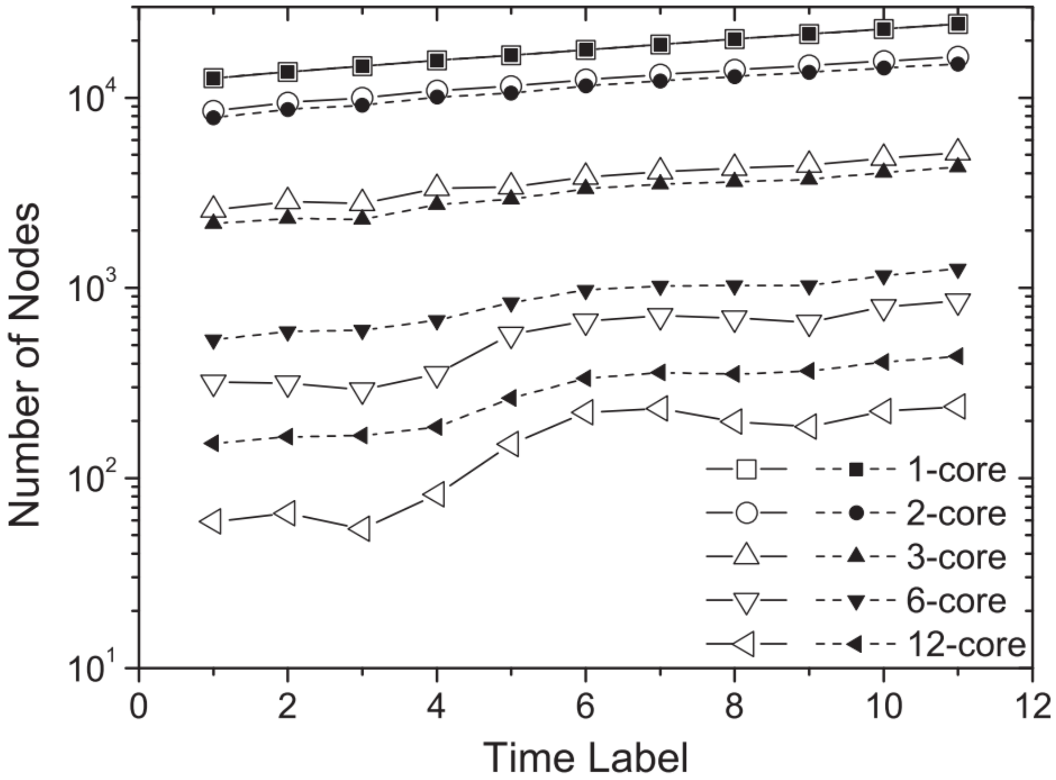
Use core numbers to predicts the influence of spreading in social networks [1]

[1] Kong, Yi-Xiu, et al. "k-core: Theories and applications." Physics Reports 832 (2019): 1-32.

Applications on Analyzing Real Internet networks

Real Internet Networks	
Vertices	Websites
Edges	Links

- The sizes of k-cores change with time
- The size of the k-core with a larger k was basically unchanged [1]



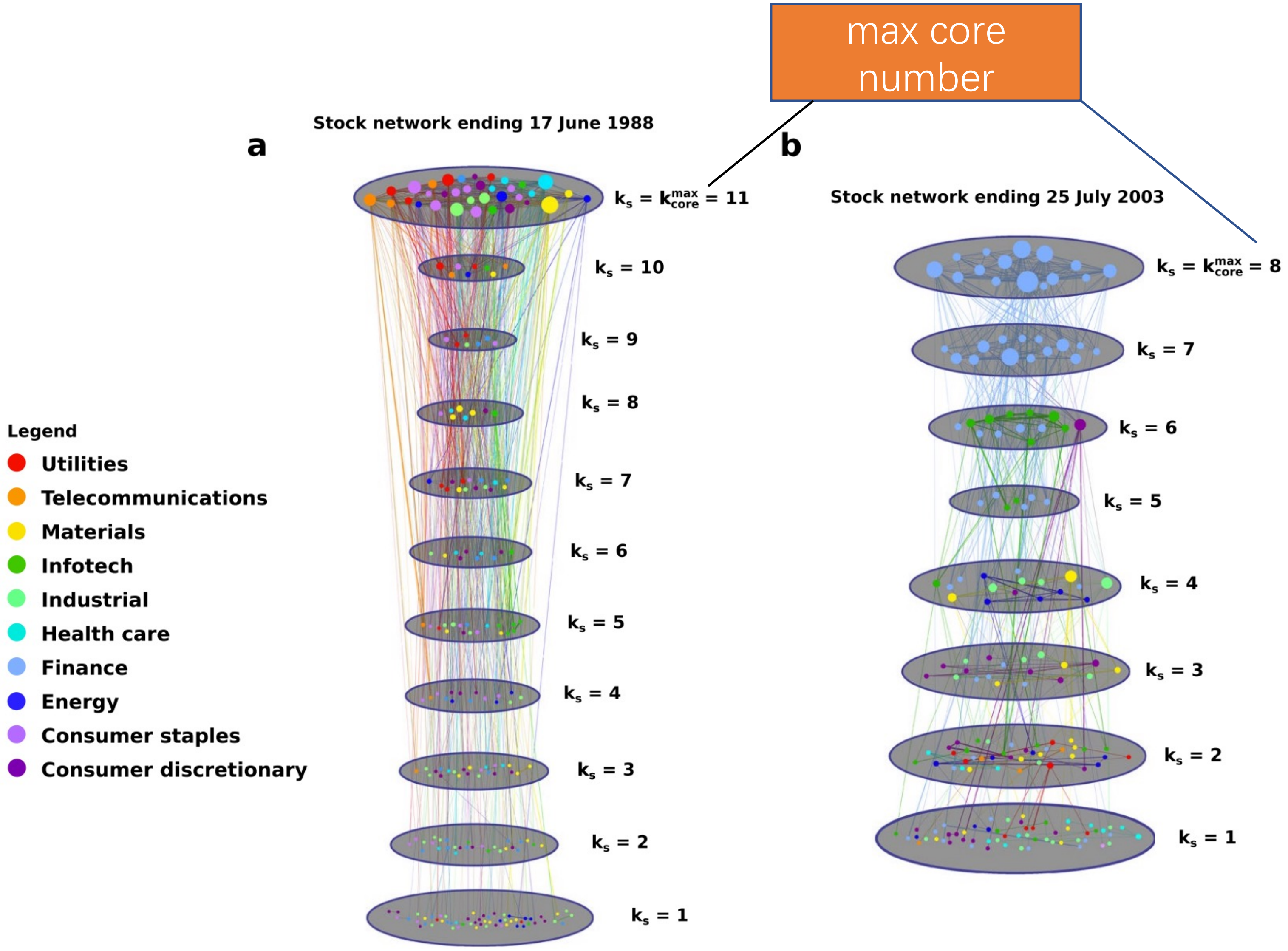
From Dec. 2001 to Dec. 2006 with six months interval

[1] Kong, Yi-Xiu, et al. "k-core: Theories and applications." Physics Reports 832 (2019): 1-32.

Applications in Economics

Stock Networks	
Vertices	Stocks
Edges	Connections

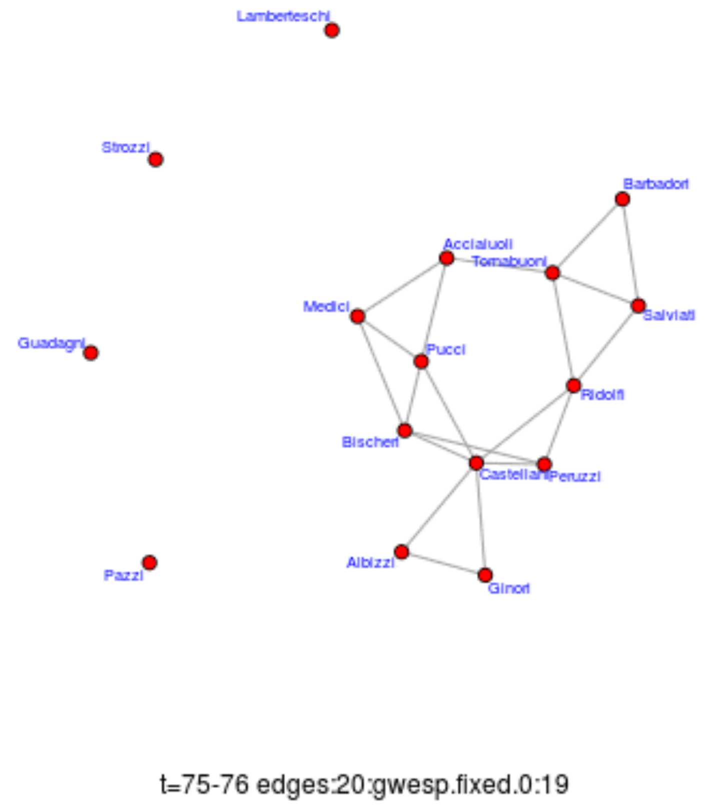
- The max core is dominated by the Finance in 2003 [2]
- The Finance has huge effects



[2] Burleson-Lesser, Kate, et al. "K-core robustness in ecological and financial networks." Scientific reports 10.1 (2020): 1-14.

Dynamic Graphs

- In practice, all above graphs can be dynamic
- Dynamic graphs change with new edges inserted or old edges removed, e.g. temporal graphs
- The core numbers have to be updated



A temporal graph with time-evolving edges [3]. Each edge has a time stamp.

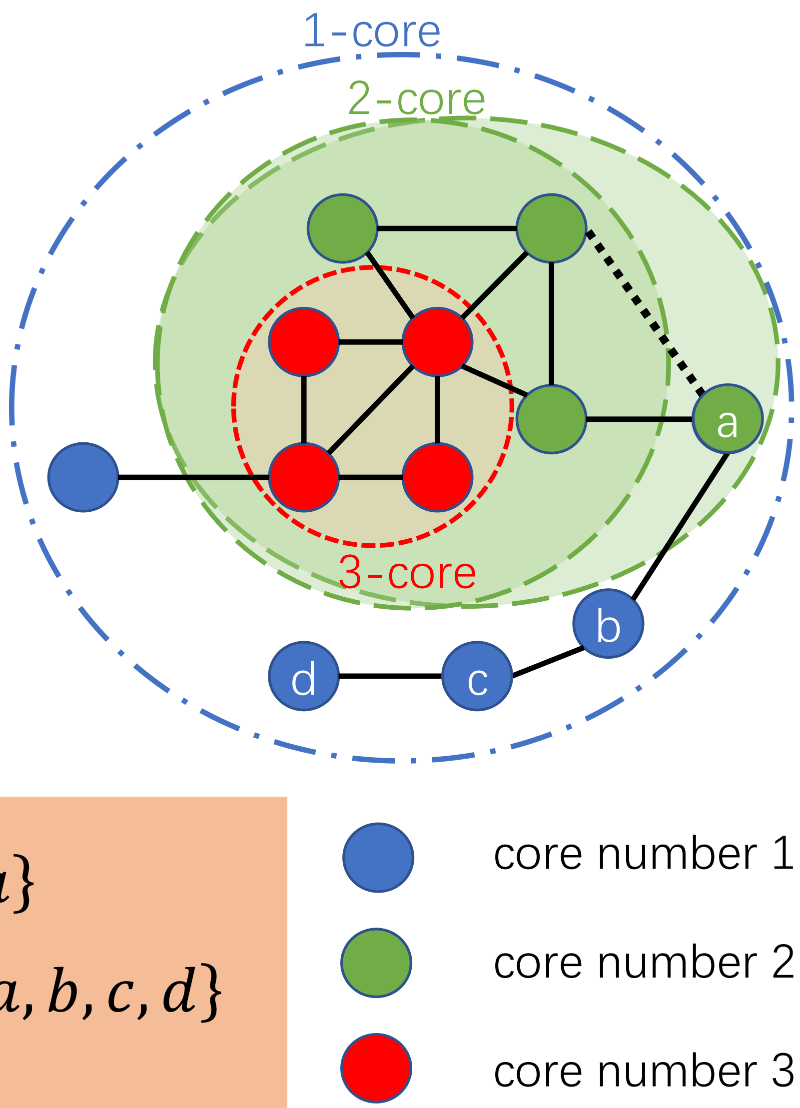
[3] Lotito, Quintino Francesco, and Alberto Montresor. "Efficient Algorithms to Mine Maximal Span-Trusses From Temporal Graphs." *arXiv preprint arXiv:2009.01928* (2020).

k-Core Maintenance

- Maintain the core numbers in dynamic graphs when inserting or removing one edge.
- Identify two set: V^* and V^+

V^*	All vertices with core number changed
V^+	All searched vertices

$$V^* \subseteq V^+$$



$V^* = \{a\}$

$V^+ = \{a, b, c, d\}$

Sequential k-Core Maintenance Algorithms

Insert or remove 100,000 edges

Dataset	Insert (second)						Remove (second)					
	OrderInsert	Trav-2	Trav-3	Trav-4	Trav-5	Trav-6	OrderRemoval	Trav-2	Trav-3	Trav-4	Trav-5	Trav-6
Facebook	**0.16**	*3.52*	4.07	5.91	10.52	16.95	**0.10**	*0.50*	1.63	4.14	9.70	17.77
Youtube	**0.26**	*2.51*	2.88	4.01	6.13	9.71	**0.28**	*0.61*	1.42	3.19	6.28	11.32
DBLP	**0.16**	1.80	*1.20*	2.31	6.32	17.65	**0.11**	*0.21*	0.61	1.88	5.49	15.78
Patents	**0.88**	2,944.14	1,805.98	1,173.20	845.93	*810.00*	**0.38**	*0.92*	4.22	18.57	75.06	276.37
Orkut	**1.14**	954.36	793.82	*780.69*	996.43	1,576.63	**0.71**	*7.75*	36.80	136.78	428.85	1,089.38
LiveJournal	**0.53**	149.56	90.93	*76.57*	125.29	285.50	**0.33**	*1.66*	6.59	24.56	86.10	233.92
Gowalla	**0.18**	*1.04*	1.37	2.21	3.78	6.38	**0.14**	*0.35*	0.84	1.82	3.45	6.22
CA	**0.52**	15.14	4.20	2.08	1.37	*1.11*	0.16	**0.08**	*0.13*	0.19	0.26	0.33
Pokec	**0.77**	1,726.04	*1,603.80*	1,650.37	1,876.48	2,338.78	**0.32**	*4.86*	53.13	259.93	756.40	1,652.88
BerkStan	**0.37**	*6.37*	7.29	9.37	13.14	16.19	**0.52**	*2.55*	5.04	8.33	12.45	17.34
Google	**0.37**	*1.01*	1.25	2.44	4.81	9.27	**0.25**	*0.46*	0.96	2.08	4.32	8.75

- The Order algorithm is much faster than the Traversal algorithm [4]
- The Order algorithm maintains an order for all vertices (k-order) to reduce the size of V^+

[4] Yikai Zhang, Jeffrey Xu Yu, Ying Zhang, and Lu Qin. A fast order-based approach for core maintenance. ICDE, pages 337–348, 2017.

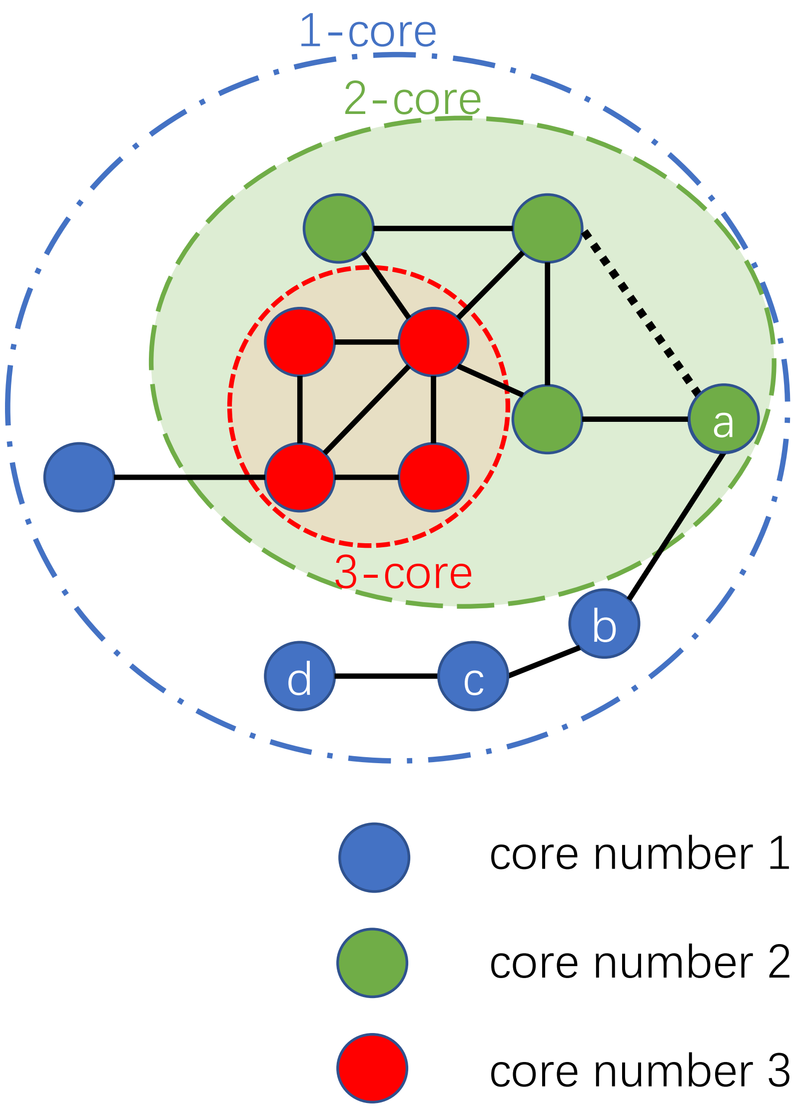
Order vs Traversal

Traversal

$$V^* = \{a\}$$
$$V^+ = \{a, b, c, d\}$$

Order

$$V^* = \{a\}$$
$$V^+ = \{a\}$$



Parallel k-Core Maintenance

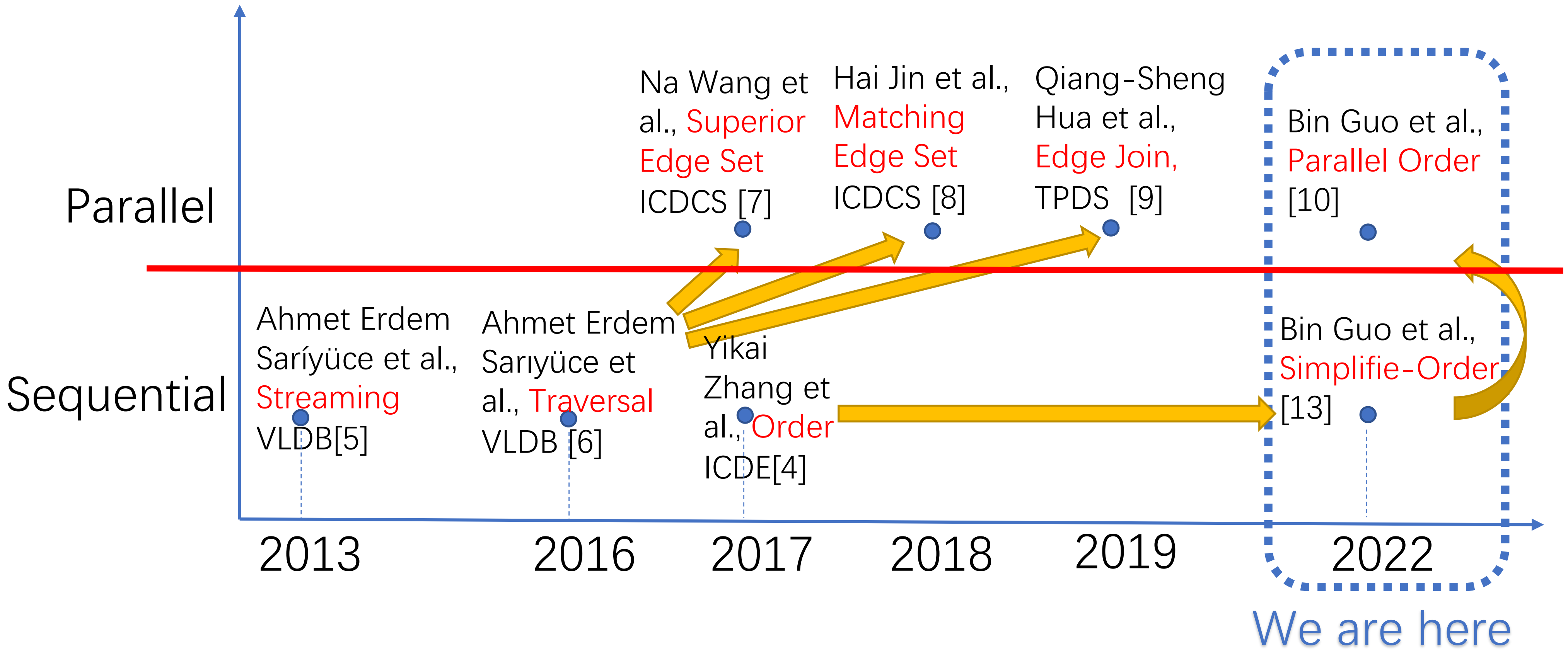
- The existing **parallel** methods [7, 8, 9] are based on Traversal algorithm
- We first propose a Simplified-Order algorithm
- Then, we propose a Parallel-Order algorithm

[7] Na Wang, Dongxiao Yu, Hai Jin, Chen Qian, Xia Xie, and Qiang-Sheng Hua. Parallel algorithm for core maintenance in dynamic graphs. In 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), pages 2366–2371. IEEE, 2017.

[8] Hai Jin, Na Wang, Dongxiao Yu, Qiang Sheng Hua, Xuanhua Shi, and Xia Xie. Core Maintenance in Dynamic Graphs: A Parallel Approach Based on Matching. IEEE Transactions on Parallel and Distributed Systems, 29(11):2416–2428, nov 2018.

[9] Qiang-Sheng Hua, Yuliang Shi, Dongxiao Yu, Hai Jin, Jiguo Yu, Zhipen Cai, Xiuzhen Cheng, and Hanhua Chen. Faster parallel core maintenance algorithms in dynamic graphs. IEEE Transactions on Parallel and Distributed Systems, 31(6):1287–1300, 2019.

The Studies of k-Core Maintenance



Time Complexity

Parallel	Worst-case (O) $\mathcal{W}$	$\mathcal{D}$	Best-case (O) $\mathcal{W}$	$\mathcal{D}$
Insert	$m'\lvert E^+\rvert \log \lvert E^+\rvert$	$m'\lvert E^+\rvert \log \lvert E^+\rvert$	$m'\lvert E^+\rvert \log \lvert E^+\rvert$	$\lvert E^+\rvert \log \lvert E^+\rvert + m'\lvert V^*\rvert$
Remove	$m'\lvert E^*\rvert$	$m'\lvert E^*\rvert$	$m'\lvert E^*\rvert$	$\lvert E^*\rvert + m'\lvert V^*\rvert$

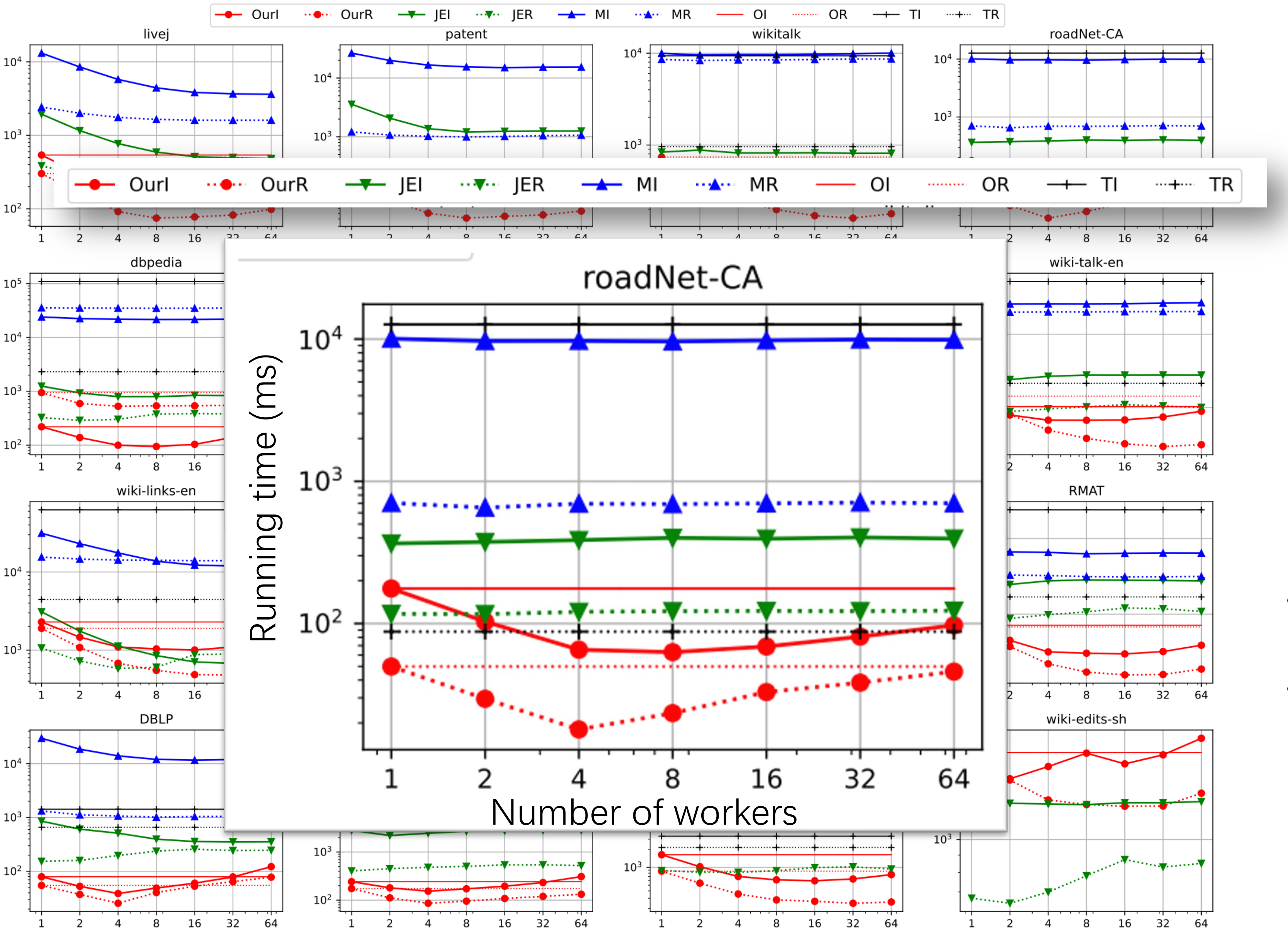
Table 1: The worst-case and best-case work, depth complexities of our parallel core maintenance operations for inserting and removing a batch of edges, where m' is the total number of edges that are inserted or removed in parallel, E^+ is adjacent edges for all vertices in V^+, and E^* is adjacent edges for all vertices in V^*.

- In the worst case, all workers execute as one blocking chain and reduce to sequential version
- The worst case is unlikely to happen over real graphs
- The best case has high speedups

Tested Graphs

Graph	$n = \|V\|$	$m = \|E\|$	AvgDeg	Max k		
livej	4,847,571	68,993,773	14.23	372	Social Networks	Static Graphs
patent	6,009,555	16,518,948	2.75	64		
wikitalk	2,394,385	5,021,410	2.10	131		
roadNet-CA	1,971,281	5,533,214	2.81	3	Road Network	
dbpedia	3,966,925	13,820,853	3.48	20	Social Networks	
baidu	2,141,301	17,794,839	8.31	78		
pokec	1,632,804	30,622,564	18.75	47		
wiki-talk-en	2,987,536	24,981,163	8.36	210	Hyperlink Network	
wiki-links-en	5,710,993	130,160,392	22.79	821		
ER	1,000,000	8,000,000	8.00	11	Synthetic Network	
BA	1,000,000	8,000,000	8.00	8		
RMAT	1,000,000	8,000,000	8.00	237		
DBLP	1,824,701	29,487,744	16.17	286	Temporal Graphs	Dynamic Graphs
Flickr	2,302,926	33,140,017	14.41	600		
StackOverflow	2,601,977	63,497,050	24.41	198		
wiki-edits-sh	4,589,850	40,578,944	8.84	47		

- For static graphs, randomly select 100,000 edges for insertion and removal
- For dynamic graphs, insert or remove 100,000 edges by their time stamps
- Evaluate the the accumulated running times

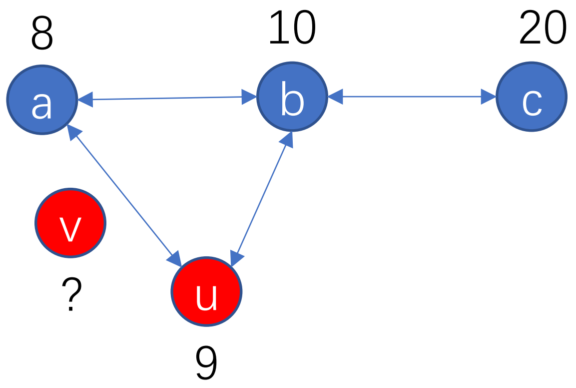


OurI	Our Insert
OurR	Our Remove
JEI	Join Edge Insert
JER	Join Edge Remove
MI	Match Edge Insert
MR	Match Edge Remove
OI	Sequential Order Insert
OR	Sequential Order Remove
TI	Sequential Traversal Insert
TR	Sequential Traversal Remove

- With 1-worker, OurI and OurR is faster than JEI and JER
- With 16-worker, OurI and OurR always has higher speedups than JEI and JER

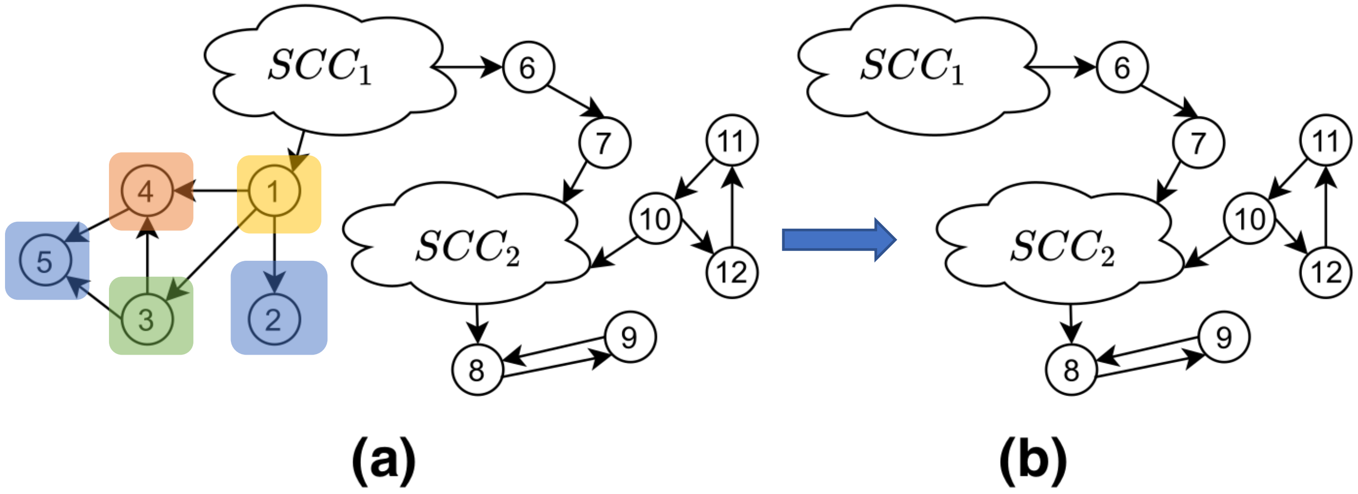
My Third Work: Parallel Order Maintenance

- Maintain an order of all items in parallel by three operations:
  - inserting,
  - deleting, and
  - comparing the order for two items [14]
- All three operations cost amortized $O(1)$ time
- We are the **first** to propose a **parallel** version



[14] **Bin Guo** and Emil Sekerinski. "New Parallel Order Maintenance Data Structure." arXiv preprint arXiv:2208.07800 (2022).

My Forth Work: Parallel Graph Trimming



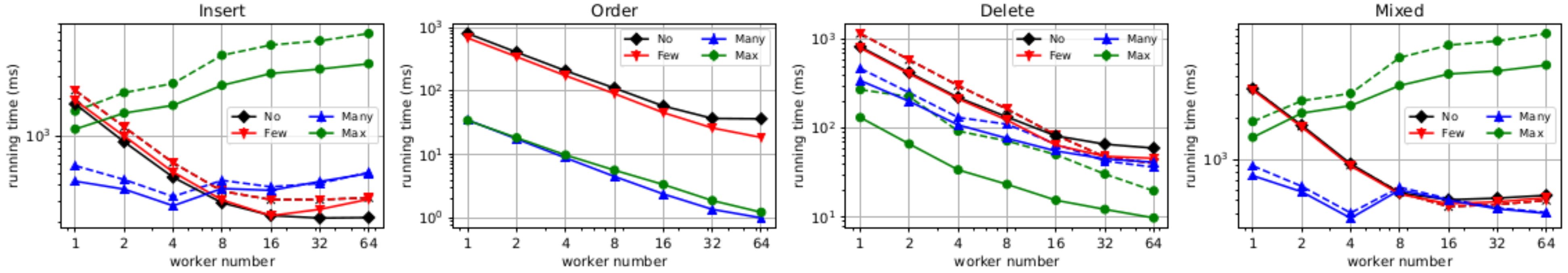
- Repeatedly remove all vertices without out-going edges [11]
  - We compare three algorithms: AC3Trim, AC4Trim and AC6Trim.
- Can be used on parallel SCC decomposition to remove size-1 SCC.

[11] "Efficient parallel graph trimming by arc-consistency" **Bin Guo**, Emil Sekerinski - **The Journal of Supercomputing**, 2022

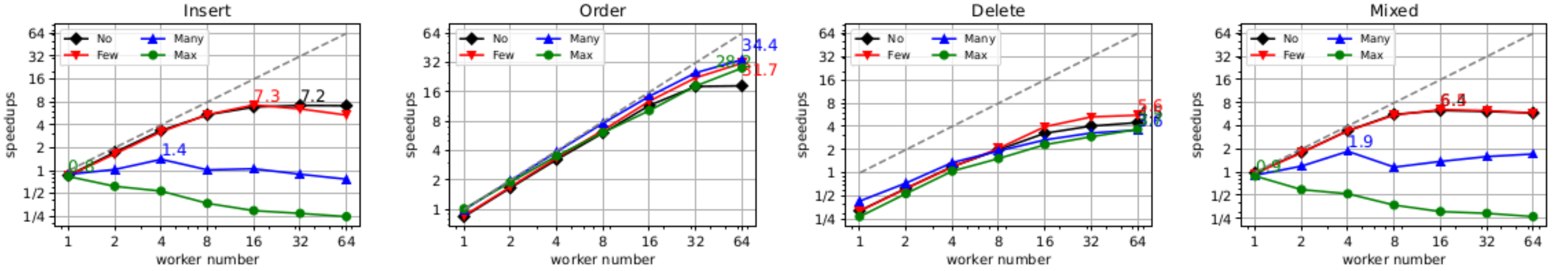
Reference

- [1] Kong, Yi-Xiu, et al. "k-core: Theories and applications." Physics Reports 832 (2019): 1-32.
- [2] Burleson-Lesser, Kate, et al. "K-core robustness in ecological and financial networks." Scientific reports 10.1 (2020): 1-14.
- [3] Lotito, Quintino Francesco, and Alberto Montresor. "Efficient Algorithms to Mine Maximal Span-Trusses From Temporal Graphs." *arXiv preprint arXiv:2009.01928* (2020).
- [4] Yikai Zhang, Jeffrey Xu Yu, Ying Zhang, and Lu Qin. A fast order-based approach for core maintenance. In Proceedings - International Conference on Data Engineering, pages 337–348, 2017.
- [5] Ahmet Erdem Saríyüce, Buğra Gedik, Gabriela Jacques-Silva, Kun-Lung Wu, and Ümit V Çatalyürek. Streaming algorithms for k -core decomposition. Proceedings of the VLDB Endowment, 6(6):433–444, 2013.
- [6] Ahmet Erdem Sarıyüce, Buğra Gedik, Gabriela Jacques-Silva, Kun-Lung Wu, and Ümit V Çatalyürek. Incremental k -core decomposition: algorithms and evaluation. The VLDB Journal, 25(3):425–447, 2016

- [7] Na Wang, Dongxiao Yu, Hai Jin, Chen Qian, Xia Xie, and Qiang-Sheng Hua. Parallel algorithm for core maintenance in dynamic graphs. In 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), pages 2366–2371. IEEE, 2017.
- [8] Hai Jin, Na Wang, Dongxiao Yu, Qiang Sheng Hua, Xuanhua Shi, and Xia Xie. Core Maintenance in Dynamic Graphs: A Parallel Approach Based on Matching. IEEE Transactions on Parallel and Distributed Systems, 29(11):2416–2428, nov 2018.
- [9] Qiang-Sheng Hua, Yuliang Shi, Dongxiao Yu, Hai Jin, Jiguo Yu, Zhipen Cai, Xiuzhen Cheng, and Hanhua Chen. Faster parallel core maintenance algorithms in dynamic graphs. IEEE Transactions on Parallel and Distributed Systems, 31(6):1287–1300, 2019.
- [10] Parallel Order-Based Core Maintenance in Dynamic Graphs B Guo, E Sekerinski - arXiv preprint arXiv:2210.14290, 2022 All 2 versions
- [11] Efficient parallel graph trimming by arc-consistency B Guo, E Sekerinski - The Journal of Supercomputing, 2022
- [12] δ-Transitive closures and triangle consistency checking: a new way to evaluate graph pattern queries in large graph databases Y Chen, B Guo, X Huang - The Journal of Supercomputing, 2020
- [13] Guo, Bin, and Emil Sekerinski. "Simplified Algorithms for Order-Based Core Maintenance." *arXiv preprint arXiv:2201.07103* (2022).
- [14] Guo, Bin, and Emil Sekerinski. "New Parallel Order Maintenance Data Structure." arXiv preprint arXiv:2208.07800 (2022).



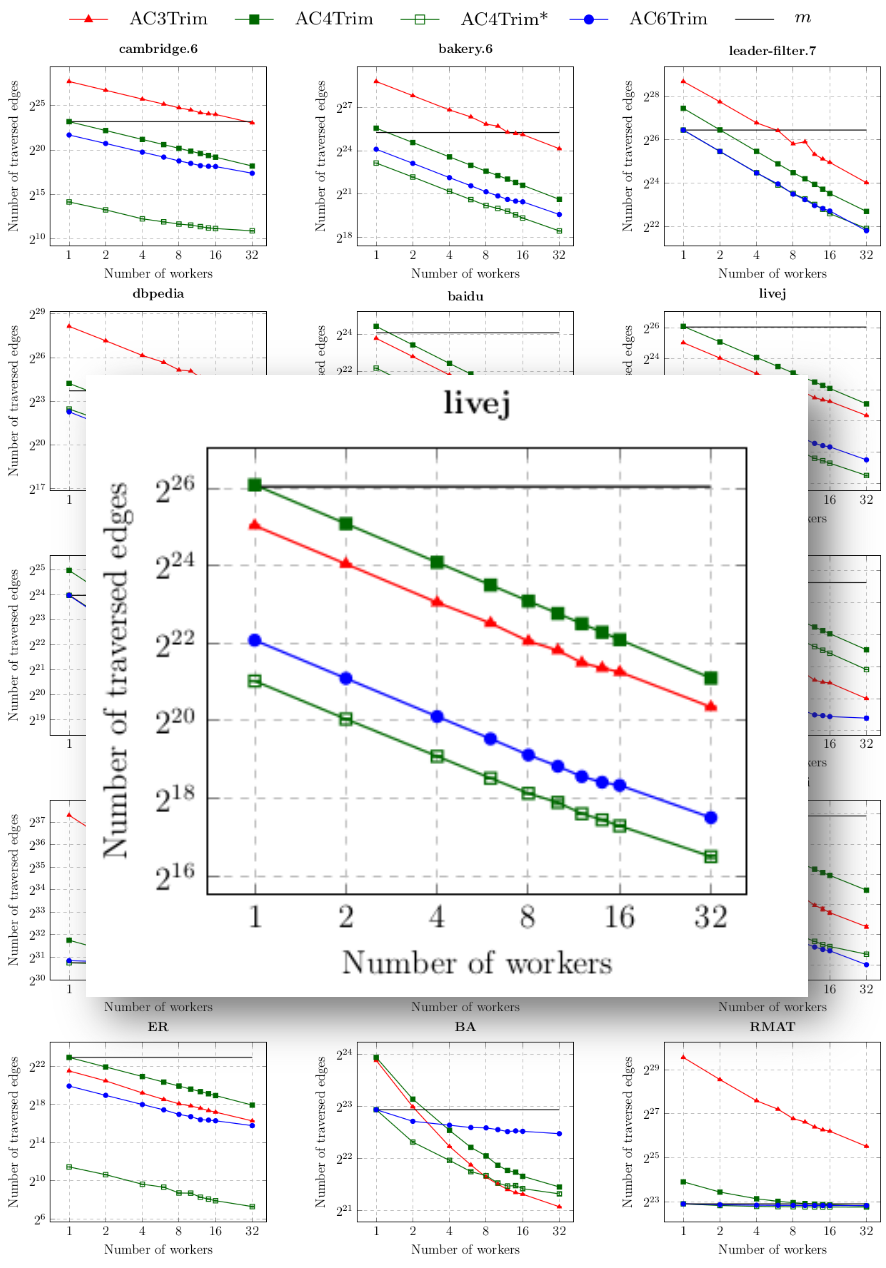
(a) The running times



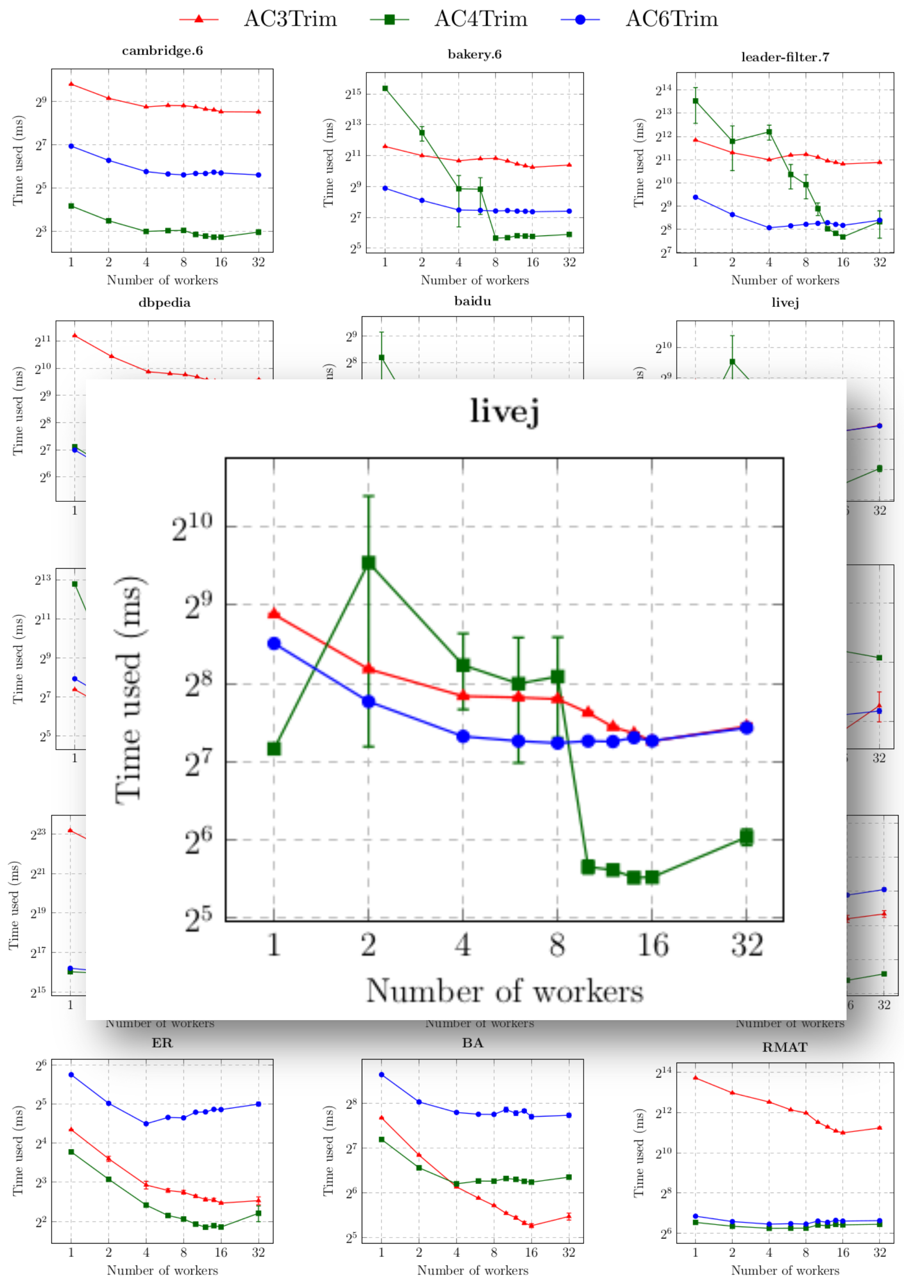
(b) The speedups

- Insert 10 million items the Order list
- Compare the Order of 10 million pair of items
- Delete 10 million Items
- Insert 10 million items, mixed with 100 million Order operations

No Case	10 million random position, no relabel
Few Case	1 million random position, few relabel
Many Case	1000 random position, many relablel
Max Case	1 random posion, maximum relabale



Evaluate the number of traversed edges



Evaluate the real running time