

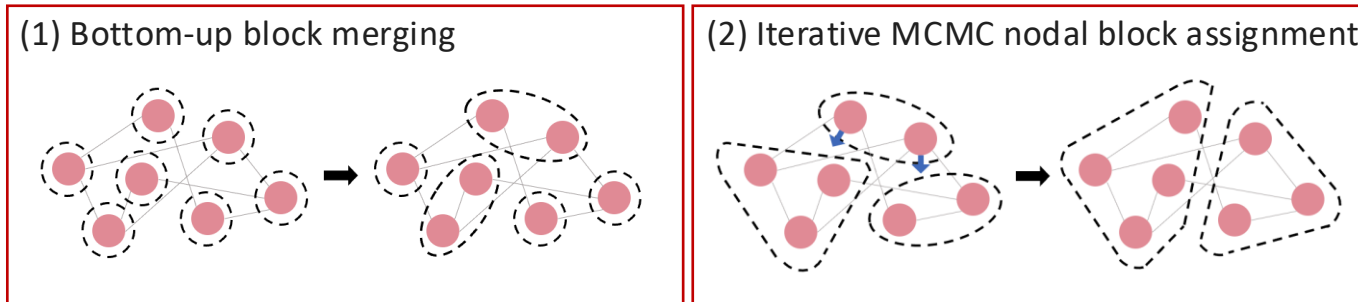


uSAP: An Ultra-Fast Stochastic Graph Partitioner

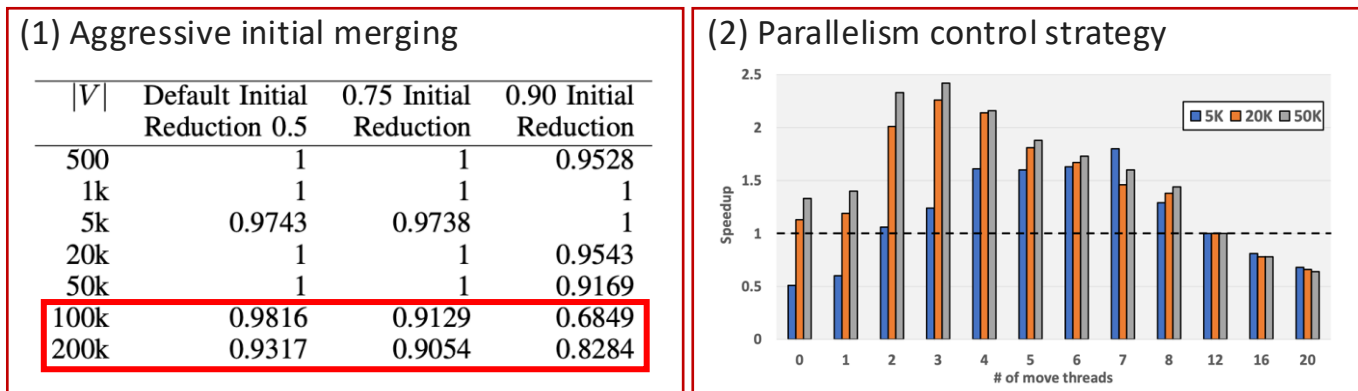
2023 MIT/IEEE/Amazon HPEC Graph Challenge

Challenges

- **Time-consuming** baseline sequential partitioner (PEIXOTO¹)



- The 2021 Champion's results can be improved (FSBP²)



Recall decrease!

Not scalable!

¹ T. P. Peixoto, "Efficient monte carlo and greedy heuristic for the inference of stochastic block models," in 2014

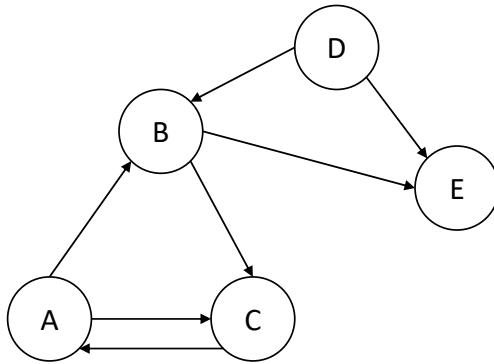
² A. J. Uppal, J. Choi, T. B. Rolinger, and H. H. Huang, "Faster stochastic block partition using aggressive initial merging, compressed representation, and parallelism control," in 2021

Solutions

- **Strongly Connected Components-based initial block merging**

⇒ try to find the blocks with more interactions

(1) Merge the block B



$$M_1^+ = \begin{pmatrix} A & B & C & D & E \\ 1 & 0 & 2 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{matrix} A \\ B \\ C \\ D \\ E \end{matrix}$$

⇒ $\Delta E = 0.35$

(2) Merge the block C

$$M_1^+ = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{matrix} C \\ D \\ E \end{matrix}$$

$$M_1^+ = \begin{pmatrix} A & B & C & D & E \\ 1 & 0 & 2 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{matrix} A \\ B \\ C \\ D \\ E \end{matrix}$$

$$M_2^+ = \begin{pmatrix} A & B & C & D & E \\ 2 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{matrix} A \\ B \\ C \\ D \\ E \end{matrix}$$

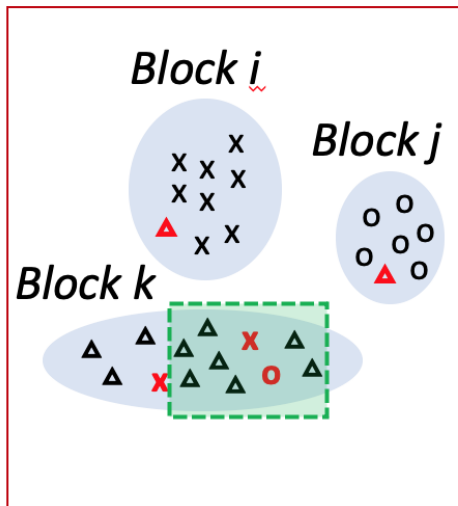
⇒ $\Delta E = -0.43$

We use a tunable threshold t_{SCC} to determine when to stop.

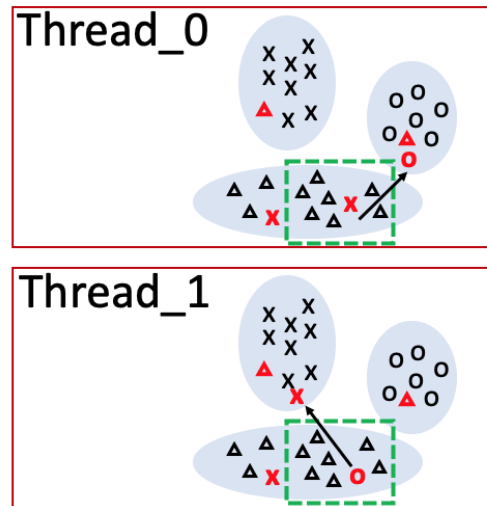
Solutions

- **Batch Parallel Nodal Block Assignment**

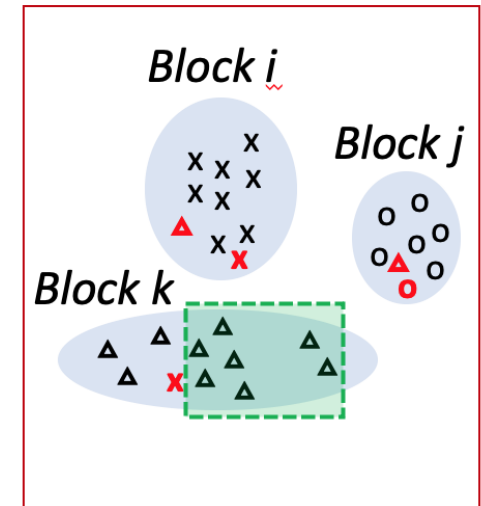
(1) Randomly select a batch



(2) Nodal block assignment in parallel



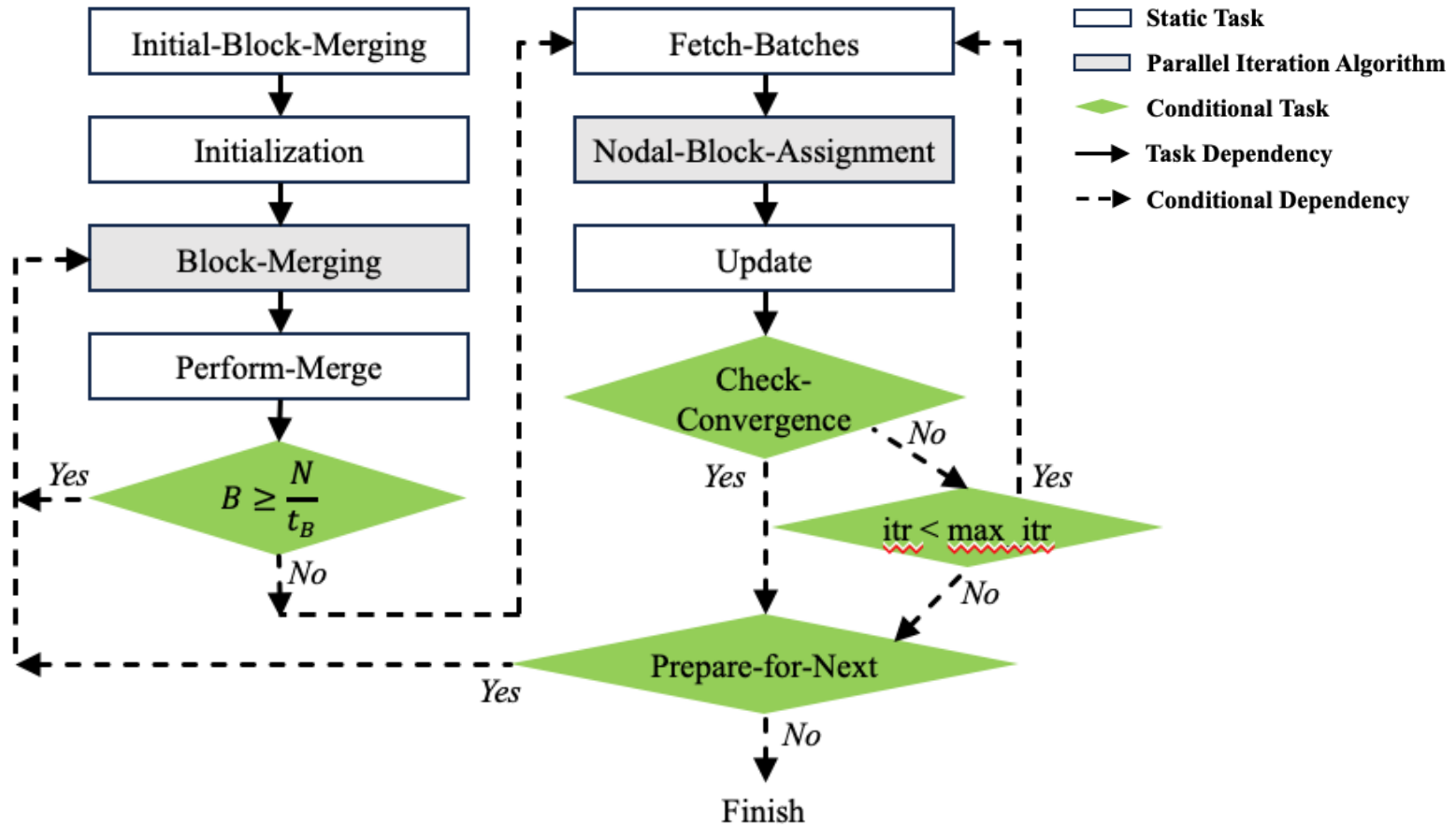
(3) Update the shared state



The results of (2) are saved in a **shared vector**.

Solutions

- Task Graph Parallelism



T.-W. Huang, D.-L. Lin, C.-X. Lin, and Y. Lin, "Taskflow: A lightweight parallel and heterogeneous task graph computing system," IEEE Transactions on Parallel and Distributed Systems, in 2021



Experimental Setup

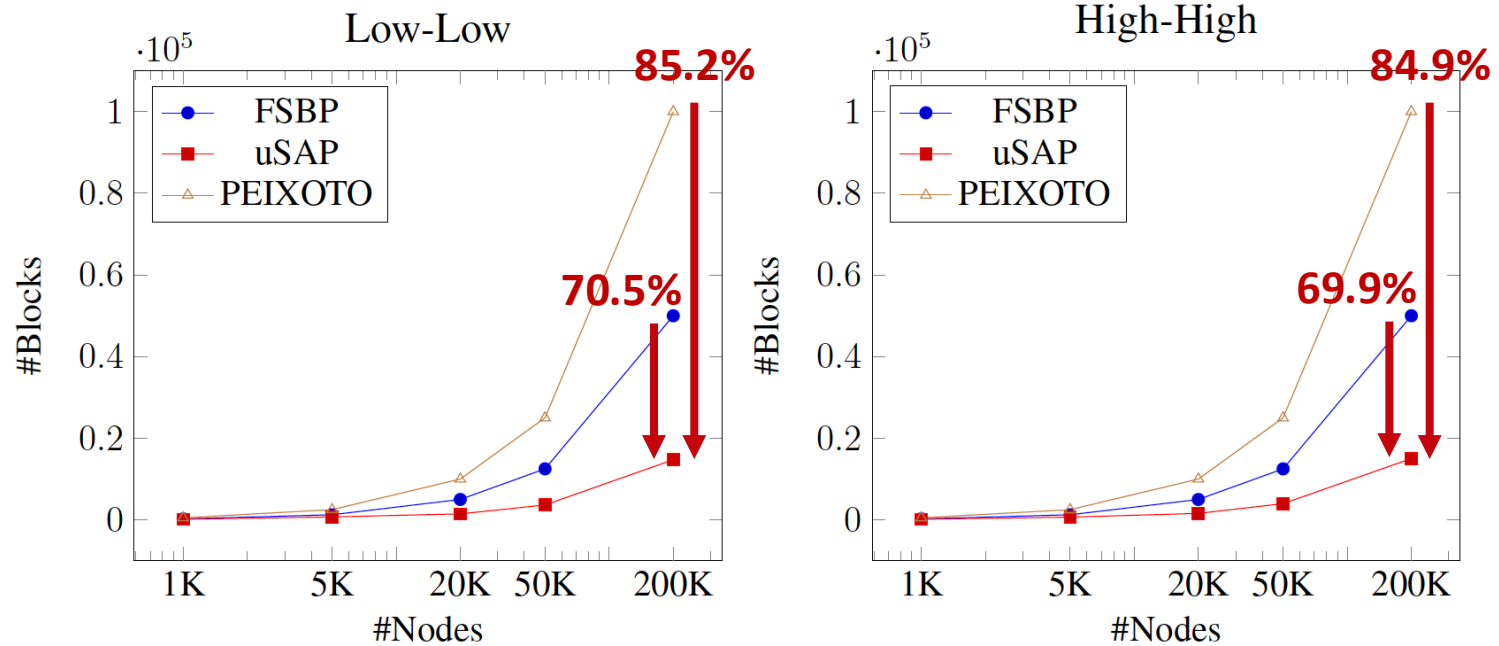
- **Baseline**
 - Example code provided by Graph Challenge (PEIXOTO¹)
 - Faster Stochastic Block Partition (FSBP²)
- **Dataset**
 - 2022 Streaming Graph Partition Dataset provided by Graph Challenge
- **Software**
 - Ubuntu 20.04 Linux 5.15.0-58-generic x86_64 machine
 - GNU GCC-11.3.0 with C++17
- **Hardware**
 - 12-core Intel® CoreTM i7-12700 processor
 - 32GB RAM

¹ T. P. Peixoto, "Efficient monte carlo and greedy heuristic for the inference of stochastic block models," in 2014

² A. J. Uppal, J. Choi, T. B. Rolinger, and H. H. Huang, "Faster stochastic block partition using aggressive initial merging, compressed representation, and parallelism control," in 2021

Experimental Result

- Our SCC-based initial block merging strategy significantly reduces the number of initial blocks to speed up the algorithm. ($t_{\text{scc}} = 10$)





Overall Runtime

- uSAP significantly outperforms PEIXOTO and FSBP in the static graph categories.
 - 1,700x for 1K-node, 80.2x for 5K-node, 103.3x for 20K-node, and 129.4x for 50K-node faster than FSBP.

Static Graph Categories												
Nodes	Low-Low			Low-High			High-Low			High-High		
	PEIXOTO	FSBP	uSAP	PEIXOTO	FSBP	uSAP	PEIXOTO	FSBP	uSAP	PEIXOTO	FSBP	uSAP
	sec MB	sec MB	sec MB	sec MB	sec MB	sec MB	sec MB	sec MB	sec MB	sec MB	sec MB	sec MB
1K	47.3	8.5	0.005	46.8	8.6	0.006	59.9	9.9	0.007	58.9	10.1	0.007
	86.6	49.1	6.9	86.4	48.0	7.3	86.7	49.3	7.4	86.6	48.8	7.2
5K	470.0	56.2	0.7	522.8	70.5	0.6	596.5	75.8	1.2	550.7	71.3	1.1
	244.6	78.9	13.6	244.7	78.5	15.7	249.9	78.6	14.3	239.1	79.3	16.7
20K	5305.5	640.6	6.2	5450.9	641.2	7.0	5681.6	689.2	10.4	5176.6	607.7	10.1
	2074.4	378.2	126.1	1951.7	386.4	136.1	2073.3	385.3	136.1	2033.6	382.0	132.6
50K	>36000	3558.6	27.5	>36000	3462.6	27.4	>36000	3381.4	53.5	>36000	3419.2	41.4
	-	942.6	318.9	-	981.5	310.5	-	947.3	310.5	-	970.8	322.5
200K	>36000	>36000	299.8	>36000	>36000	338.8	>36000	>36000	1243.8	>36000	>36000	1381.2
	-	-	1222.6	-	-	1228.8	-	-	1228.8	-	-	1334.5



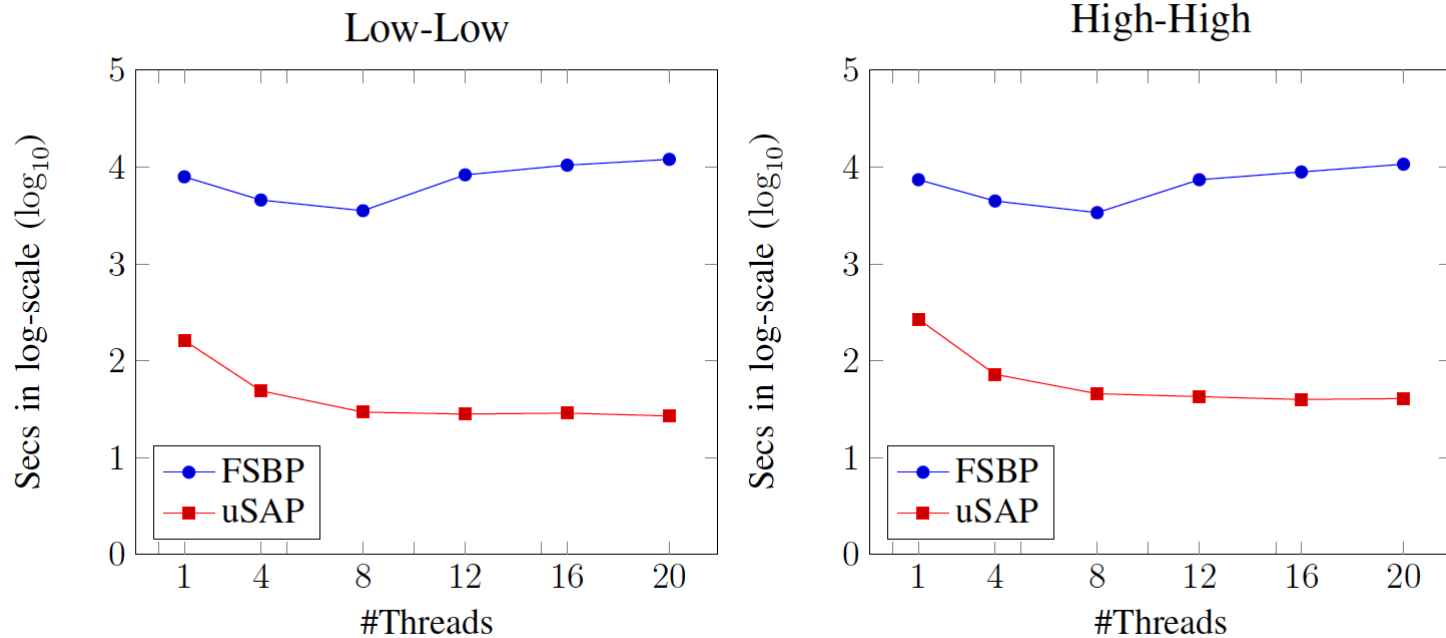
Pairwise Precision and Recall

- The quality of the partition results is better than PEIXOTO and FSBP with our SCC-based initial merging strategy.

Nodes	Static Graph Categories											
	Low-Low			Low-High			High-Low			High-High		
	PEIXOTO PP PR	FSBP PP PR	uSAP PP PR	PEIXOTO PP PR	FSBP PP PR	uSAP PP PR	PEIXOTO PP PR	FSBP PP PR	uSAP PP PR	PEIXOTO PP PR	FSBP PP PR	uSAP PP PR
1K	0.994 0.996	0.994 0.996	0.998 0.998	0.939 0.990	0.811 0.995	0.952 0.993	0.717 0.883	0.719 0.878	0.747 0.971	0.686 0.972	0.710 0.653	0.843 0.954
5K	0.940 1.000	1.000 1.000	1.000 1.000	0.976 0.769	0.970 0.850	0.987 0.998	0.861 0.816	0.641 0.666	0.983 0.997	0.676 0.789	0.689 0.721	0.861 0.801
20K	0.984 1.000	1.000 1.000	1.000 1.000	0.721 0.671	0.921 0.479	0.948 0.999	0.950 0.740	0.875 0.705	0.982 0.999	0.889 0.995	0.789 0.943	0.803 0.778
50K	- -	0.988 0.855	1.000 1.000	- -	0.849 0.997	0.921 0.998	- -	0.757 0.862	0.946 0.994	- -	0.766 0.64	0.456 0.981
200K	- -	- -	0.983 1.000	- -	- -	0.768 0.922	- -	- -	0.832 0.314	- -	- -	0.386 0.885

Scalability (Runtime v.s. #Threads)

- uSAP outperforms FSBP regardless of the number of threads. FSBP achieves its best performance when utilizing 8 threads.





Conclusion

- **SCC-based Initial block merging**
- **Batch parallel nodal block assignment**
- **Task graph parallelism**
- **Experimental Result**
 - Better runtime (129.4x faster than FSBP on 50K-node graph)
 - Better pairwise precision and recall
 - Better scalability compared to FSBP



Thank you!

—
uSAP: An Ultra-Fast Stochastic Graph Partitioner

<https://github.com/gary30404/uSAP.git>