Efficient Parallel Algorithm for Bi-core Decomposition

Claire Wang
cwang23@andover.edu

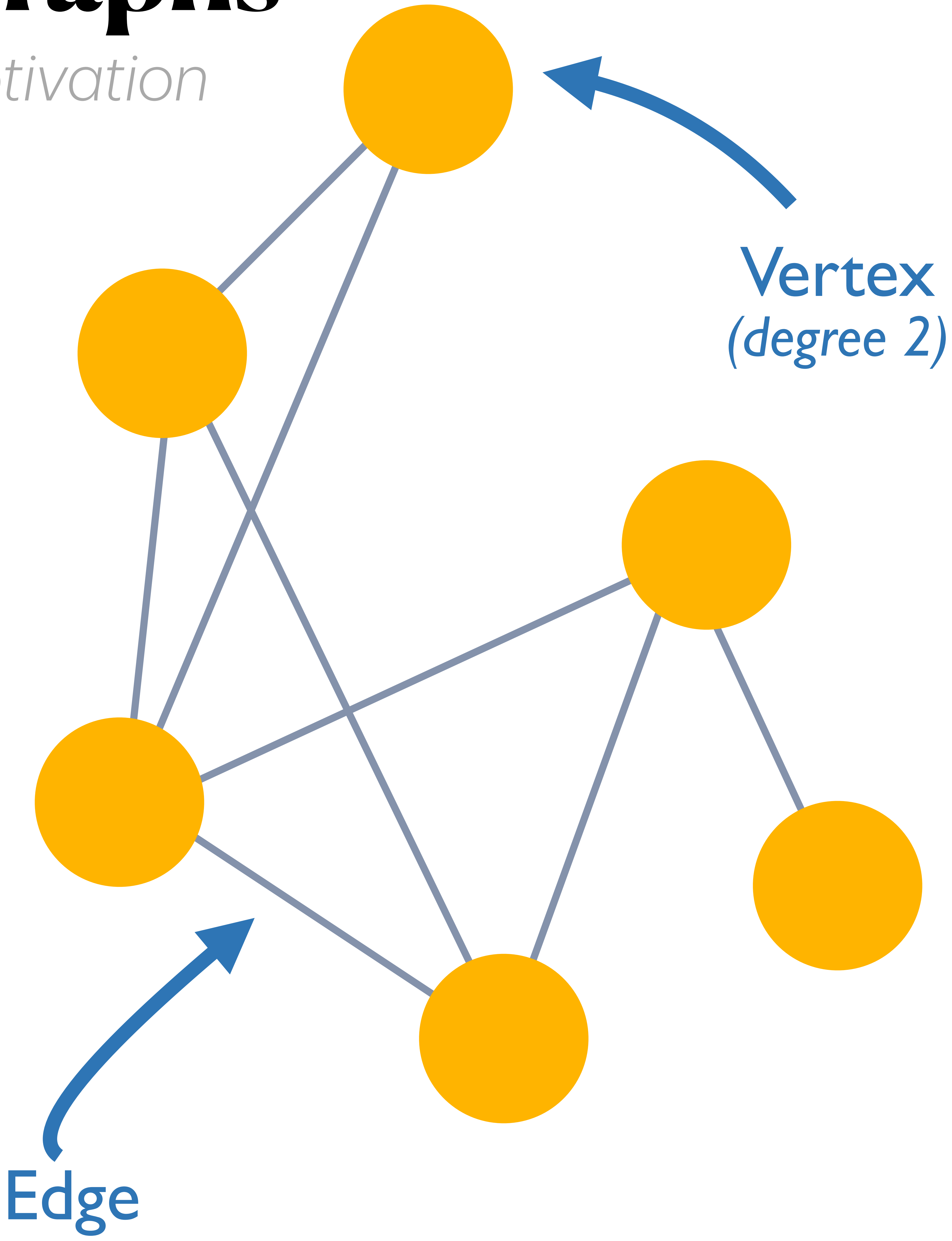
Michael (Yihao) Huang
yhuang23@andover.edu

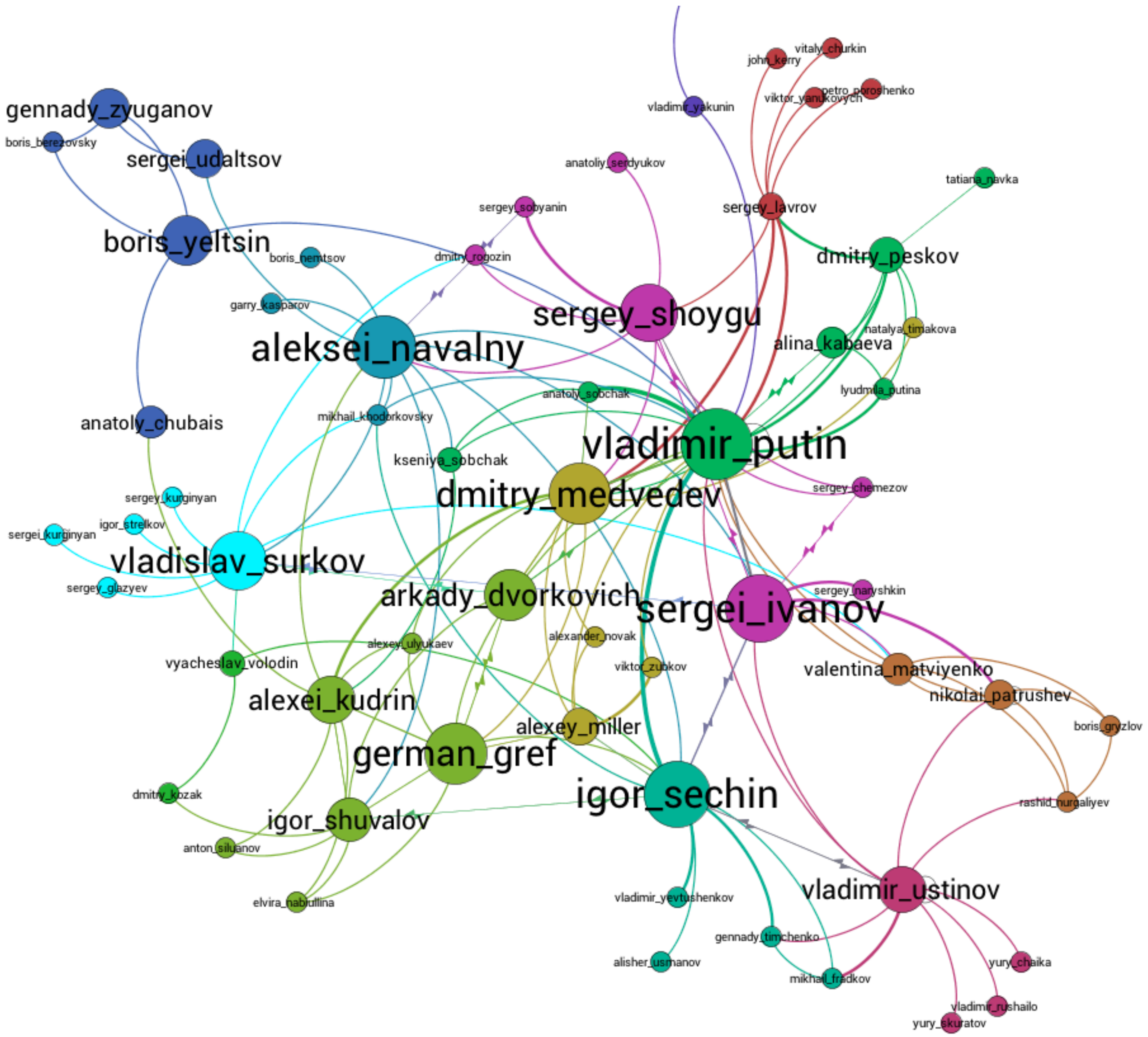
PRIMES 2021
Mentored by: *Jessica Shi, Julian Shun*

Graphs

Motivation

- set of vertices and edges
  - a **vertex** represents an object of interest in a study or dataset
  - an **edge** represents a *relationship* between two vertices.

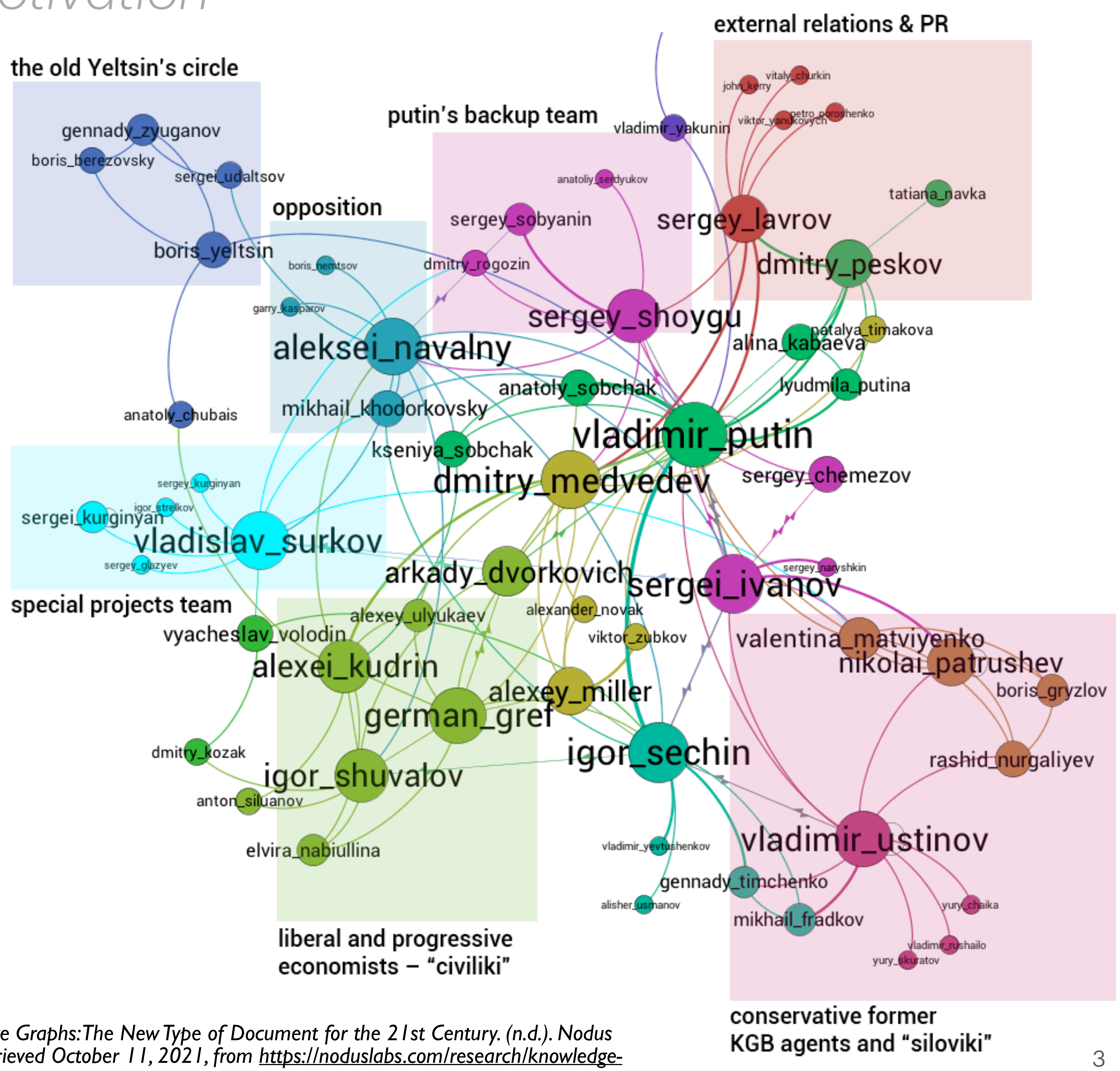




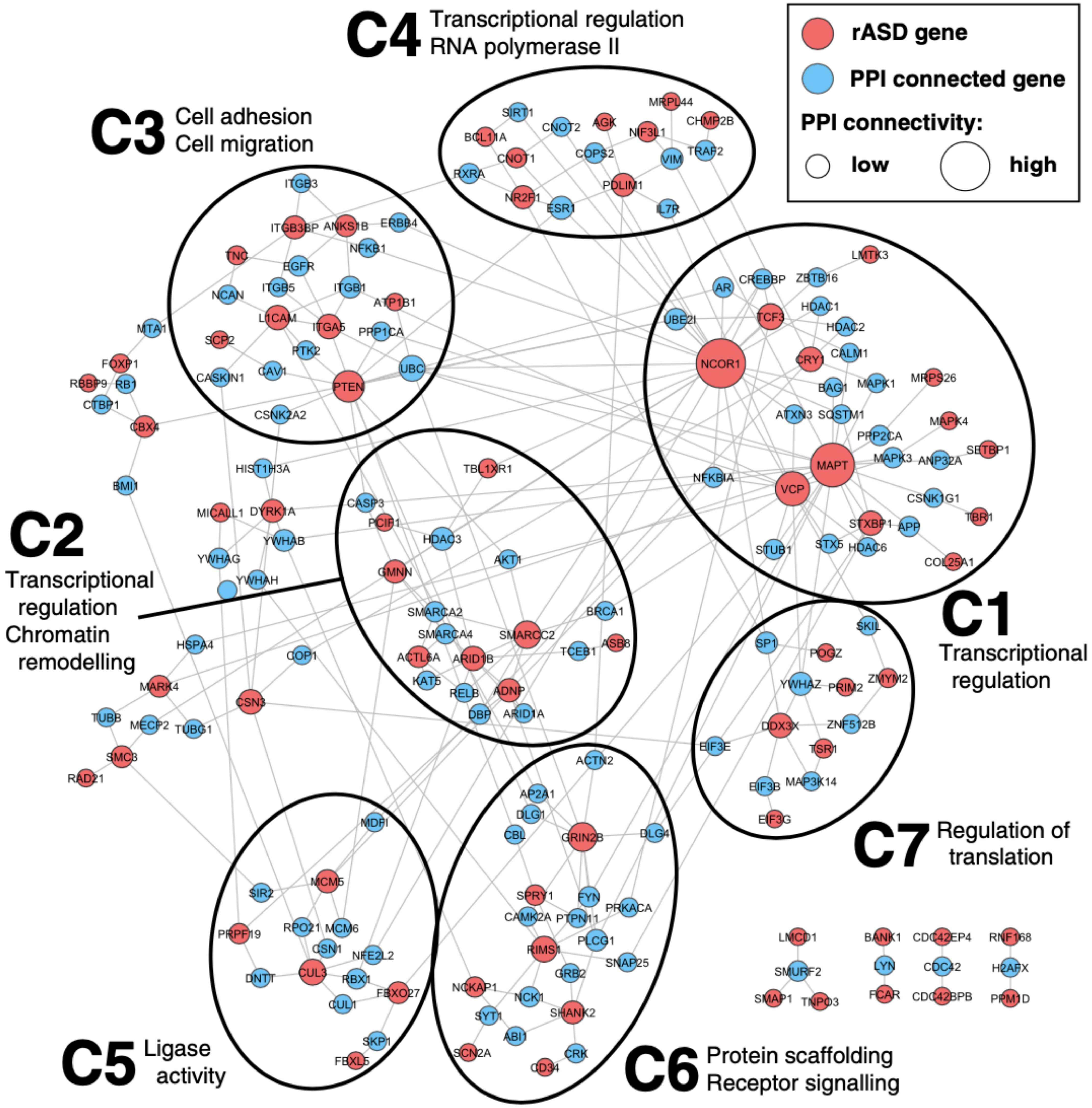
Knowledge Graphs: The New Type of Document for the 21st Century. (n.d.). Nodus Labs. Retrieved October 11, 2021, from https://noduslabs.com/research/knowledge-graphs-type-document/

Dense Subgraph Discovery

Motivation



Knowledge Graphs: The New Type of Document for the 21st Century. (n.d.). Nodus Labs. Retrieved October 11, 2021, from https://noduslabs.com/research/knowledge-graphs-type-document/

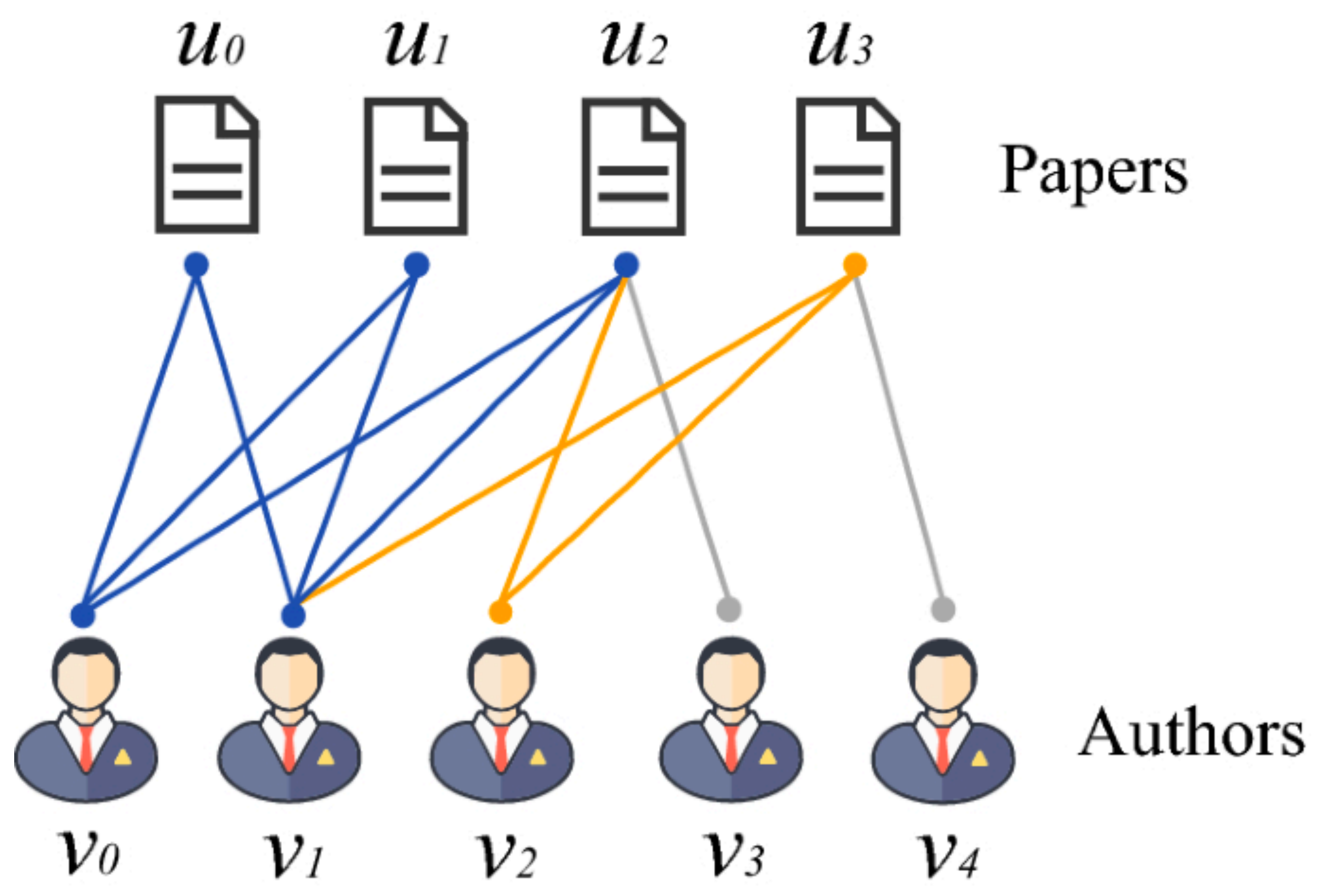


Liu, L., Lei, J., Sanders, S. J., Willsey, A. J., Kou, Y., Cicek, A. E., Klei, L., Lu, C., He, X., Li, M., Muhle, R. A., Ma'ayan, A., Noonan, J. P., Šestan, N., McFadden, K. A., State, M. W., Buxbaum, J. D., Devlin, B., & Roeder, K. (2014). DAWN: a framework to identify autism genes and subnetworks using gene expression and genetics. *Molecular Autism*, 5(1). https://doi.org/10.1186/2040-2392-5-22

Bipartite Graphs

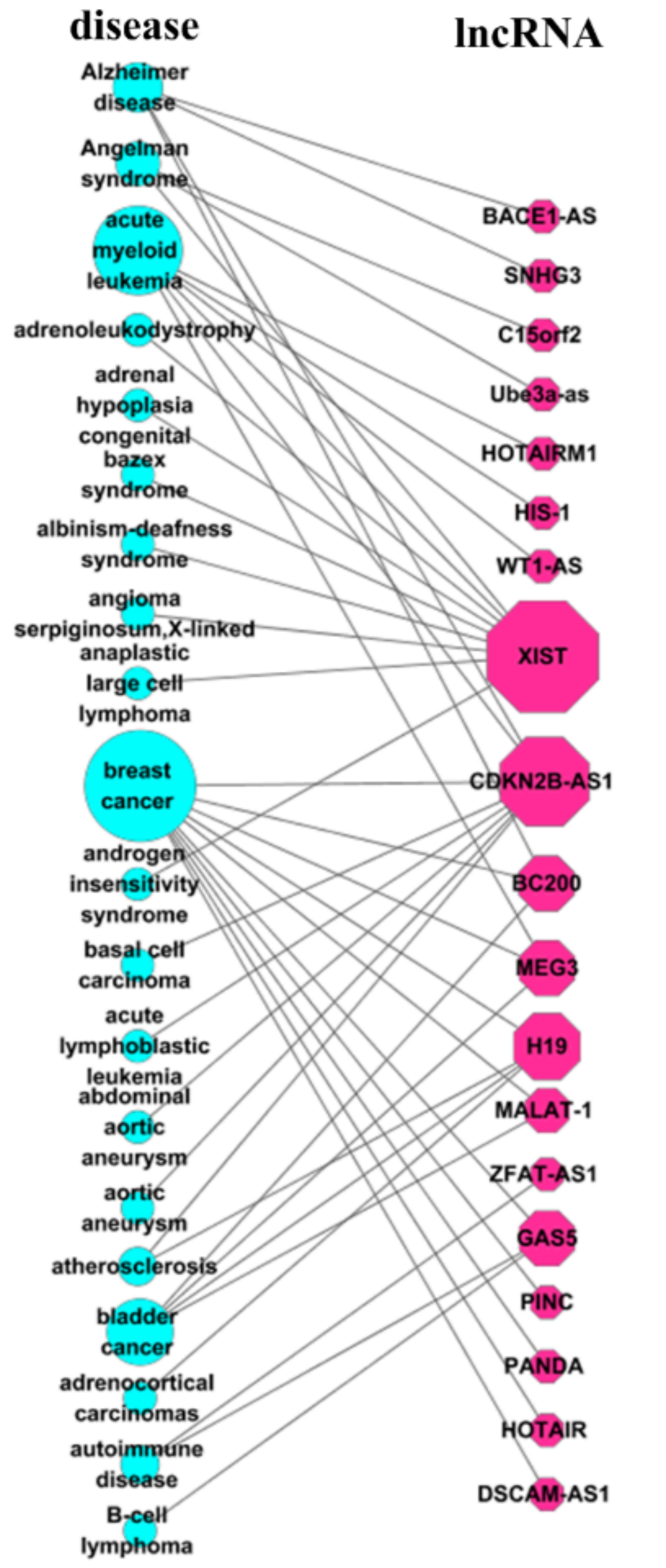
Motivation

- a graph G made up of two mutually exclusive sets of vertices with edges that connect them
- model the relationship between two groups



Authorship graph

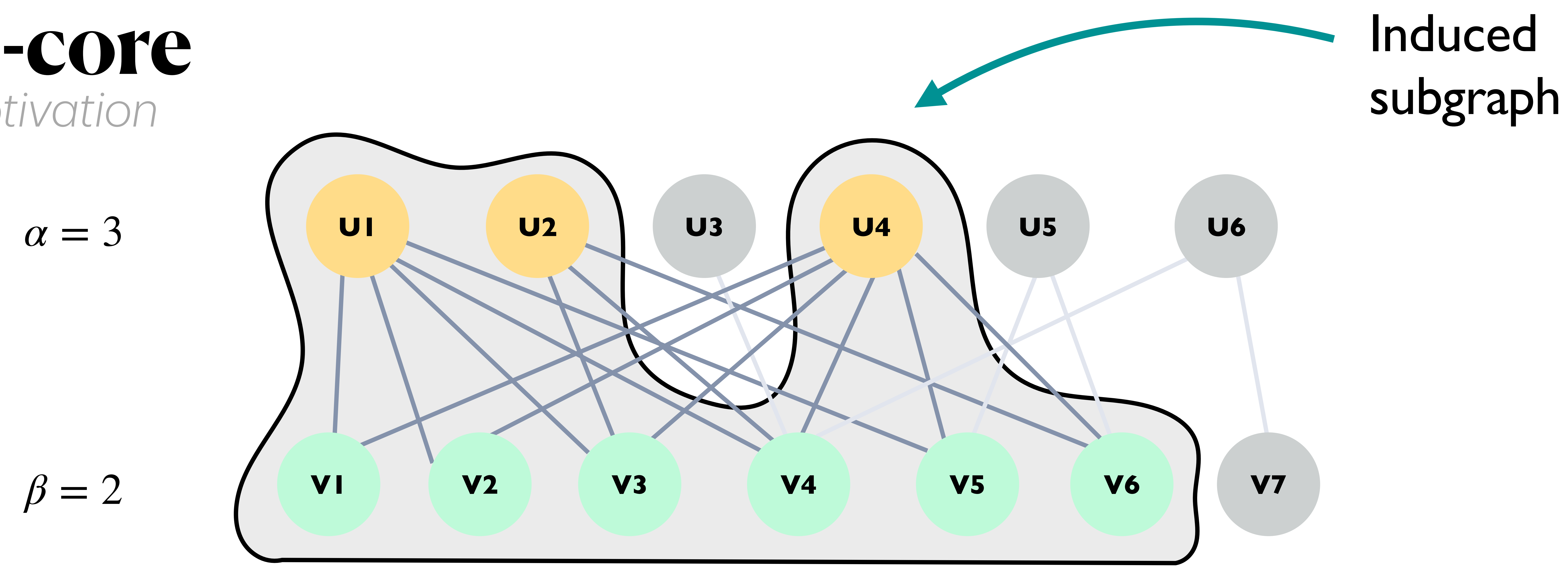
Diseases and their correlated lncRNA loci



https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0087797

Bi-core

Motivation



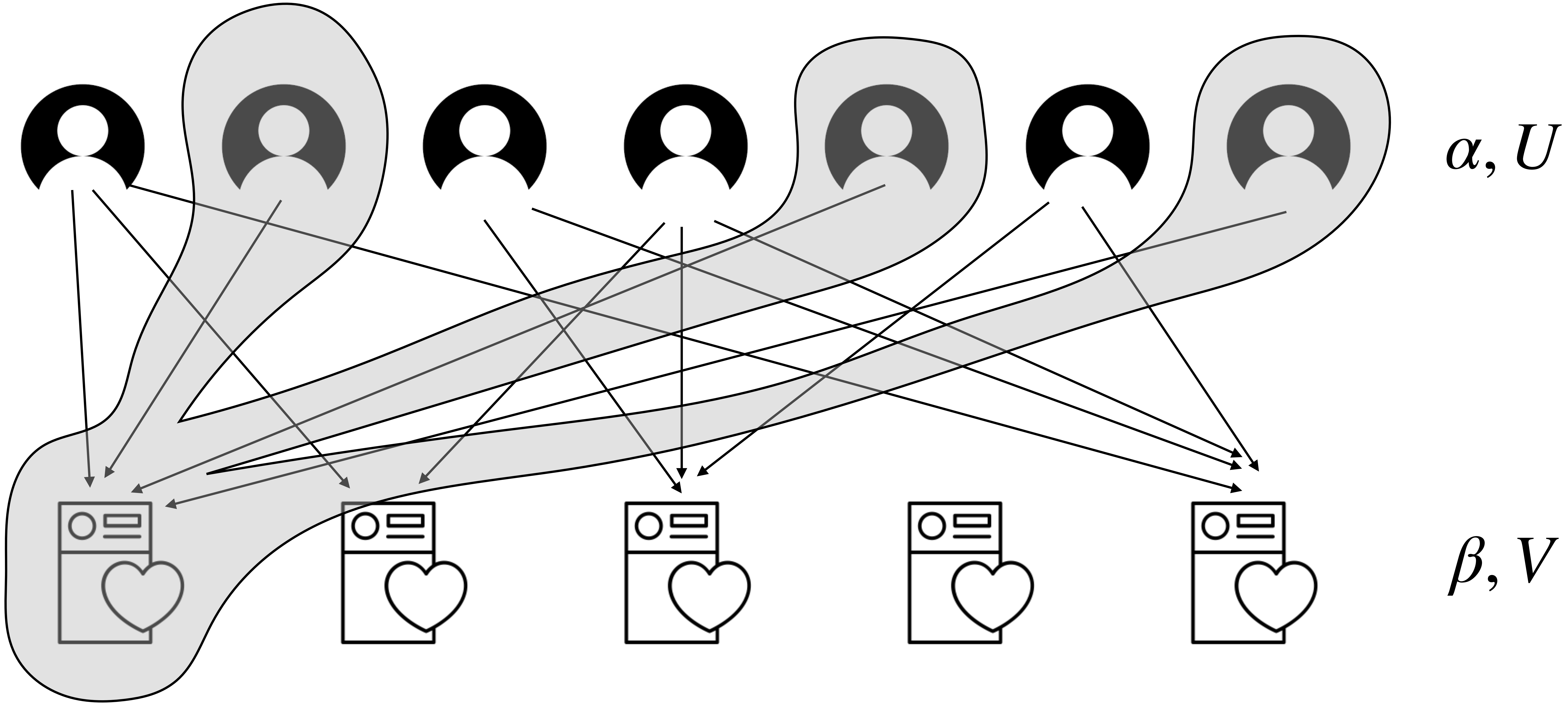
(α, β)-core

*(3,2) core means that every **U** node has at least 3 edges and every **V** node has at least 2 edges within the subgraph*

Alpha and beta maxes

Fraudster Detection

Applications of Bi-core Decomposition

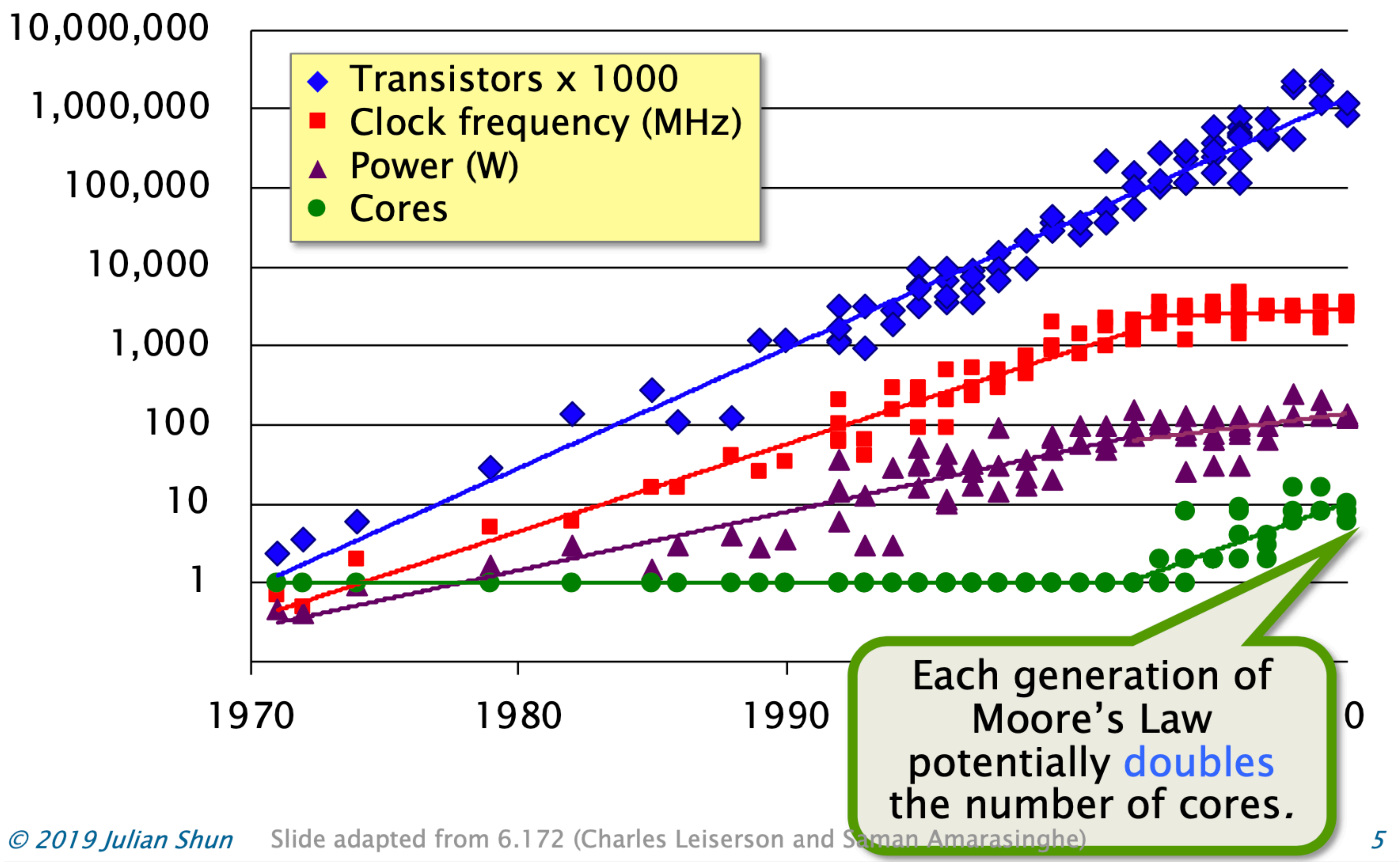


High β low α

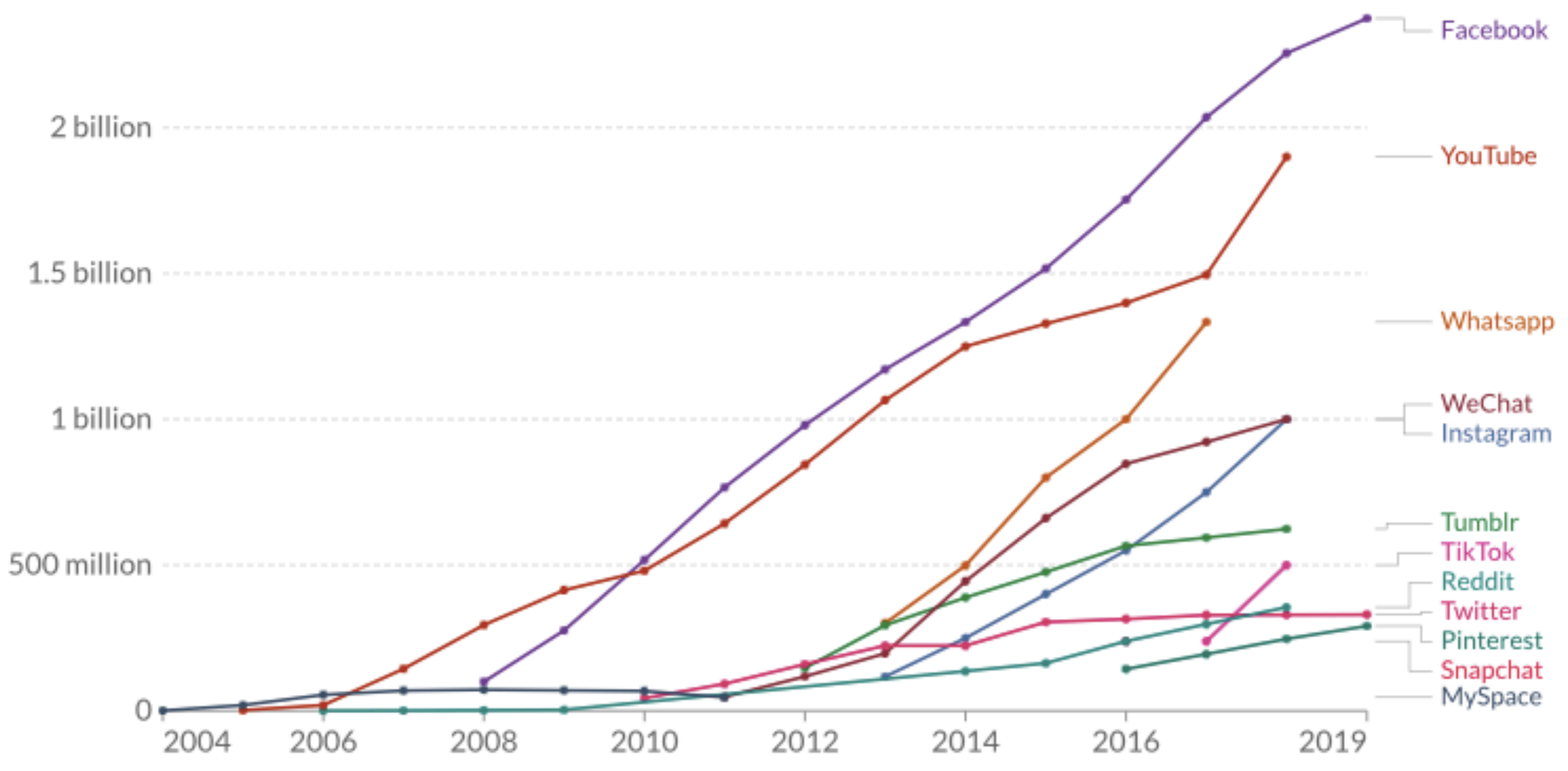
↓

Suspicious and flagged

Parallelism



Increase in size of graphs and # of cores.



Preliminaries

Work-span Model

Preliminary

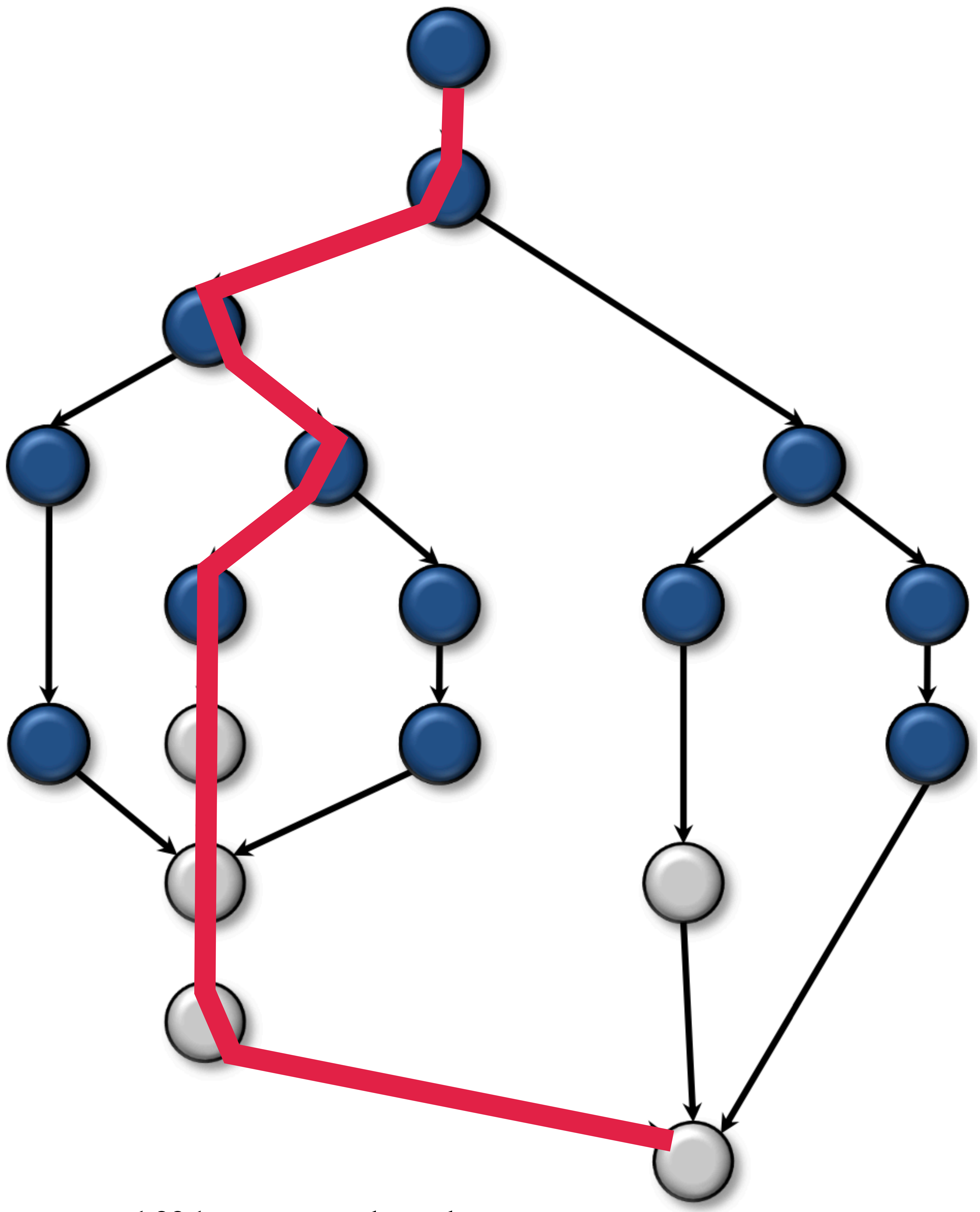
T_p = Runtime with p processors

T_1 = Work

T_∞ = Span

Brent's Law:

$$T_p \leq T_\infty + \frac{T_1 - T_\infty}{p}$$



6.886 Lecture 2 Algorithm Engineering, MIT

Work Efficiency: same Work Complexity as the best sequential algorithm

Bi-core Decomposition

Goal: find $\alpha_{\max\beta}(v)$ for every β and v and find $\beta_{\max\alpha}(u)$ for every α and u

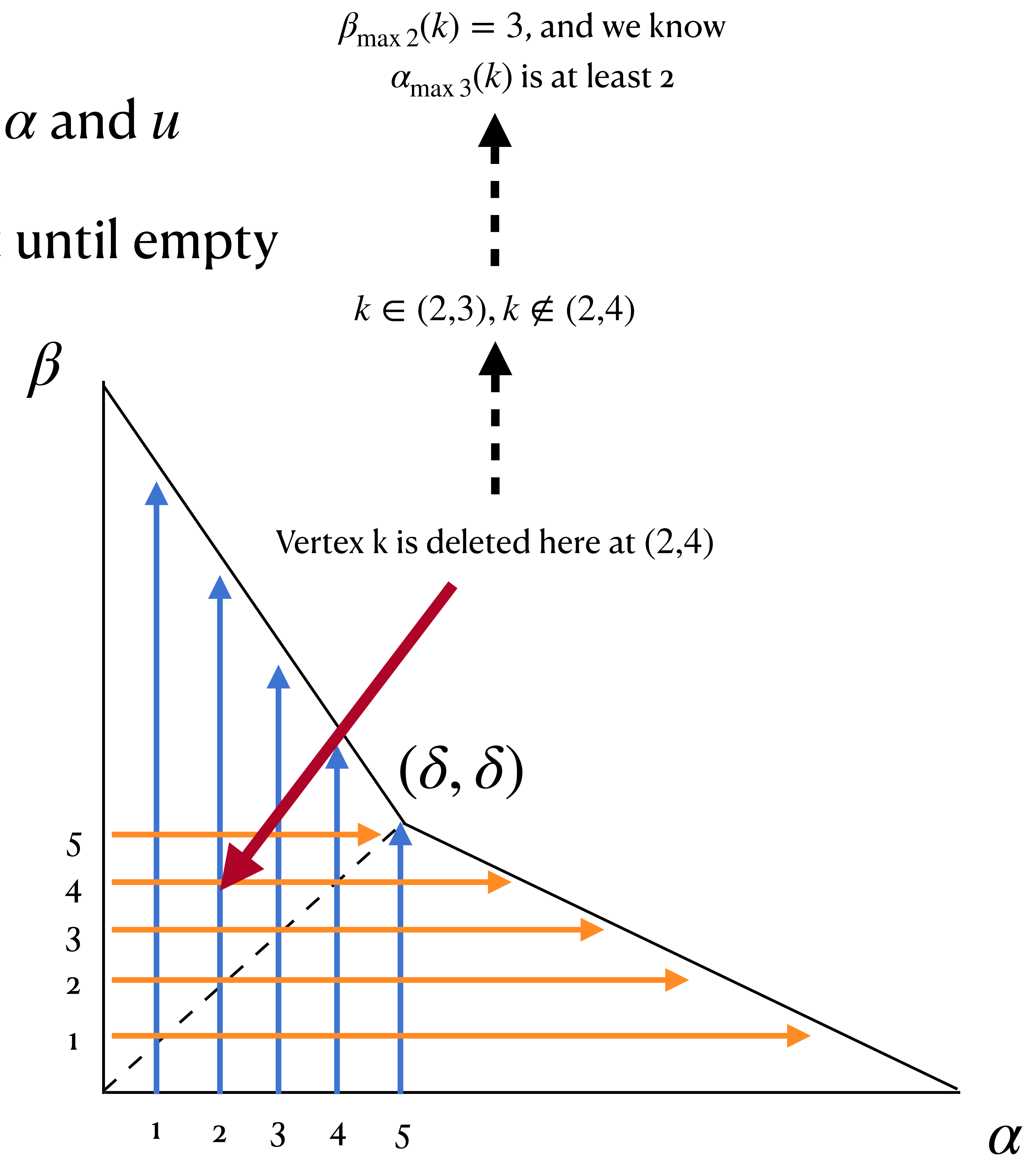
Process: Peeling-based—remove vertices with min degree—repeat until empty

For $\beta = 1$ to δ:

Peel from $\alpha = 1$ to its maximum value

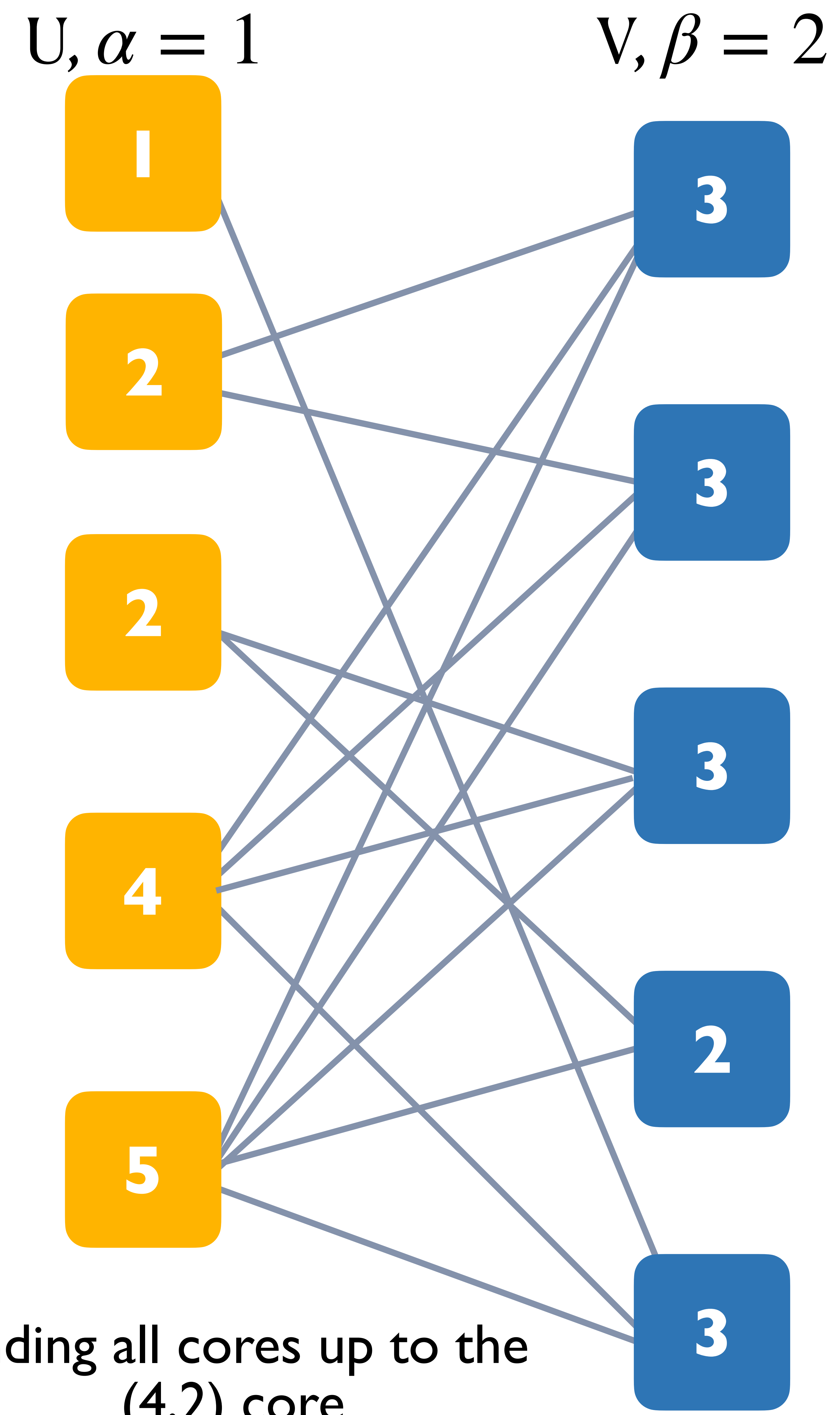
For $\alpha = 1$ to δ:

Peel from $\beta = 1$ to its maximum value



Liu, B., Yuan, L., Lin, X. *et al.* Efficient (α,β)-core computation in bipartite graphs. *The VLDB Journal* **29,** 1075–1099 (2020). https://doi.org/10.1007/s00778-020-00606-9

Sequential Bi-core Decomposition



Finding all cores up to the (4,2) core

In the yellow, U partition find the vertex with minimum induced deg

For each such vertex:

Delete it

Update blue vertex degree

Check the **blue** partition for vertices with degree < β

For each blue node < β:

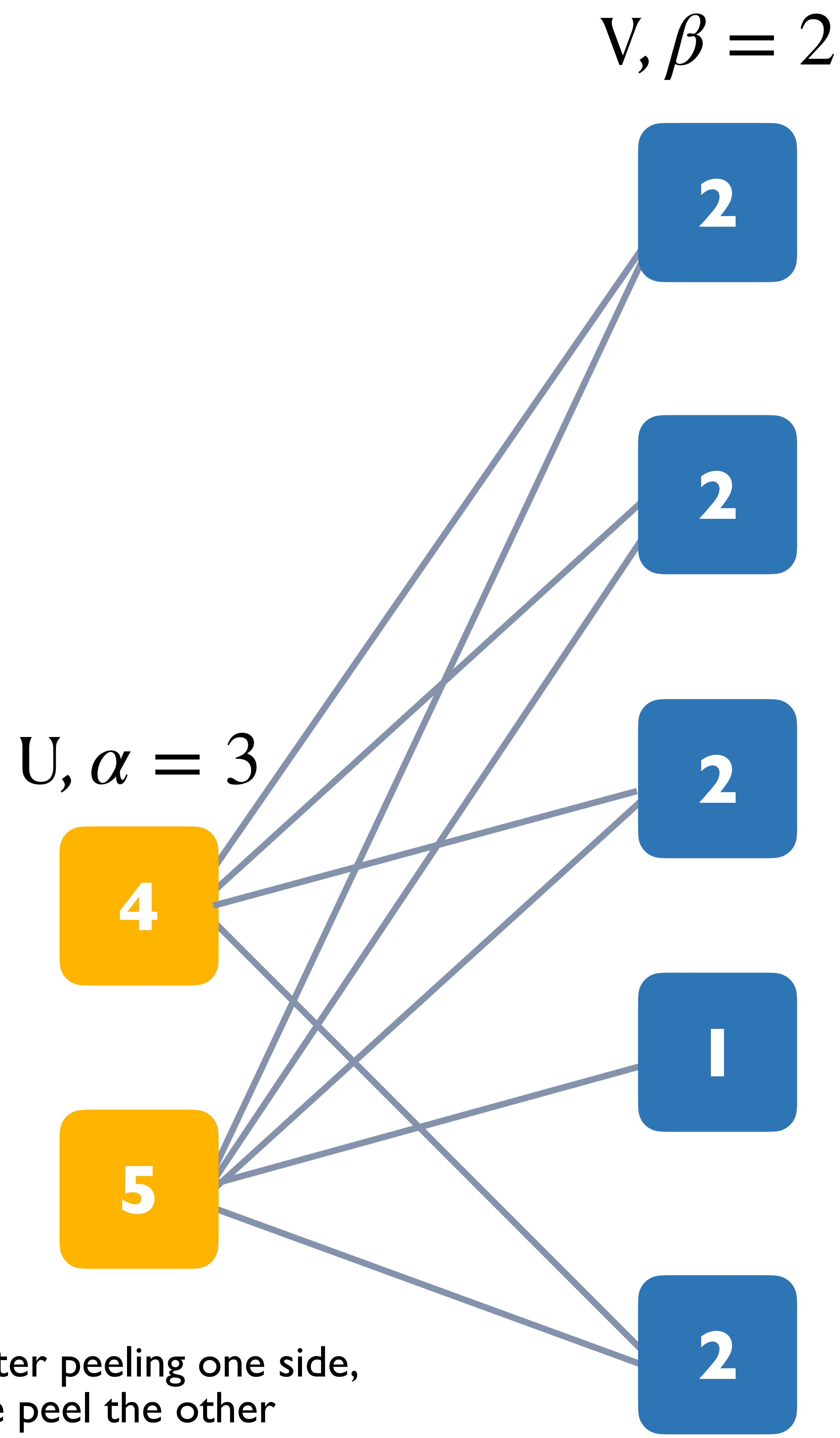
Delete node

Update yellow vertex degree

Update yellow

Liu, B., Yuan, L., Lin, X. *et al.* Efficient (α,β)-core computation in bipartite graphs. *The VLDB Journal* **29,** 1075–1099 (2020). https://doi.org/10.1007/s00778-020-00606-9

Sequential Bi-core Decomposition



After peeling one side, we peel the other

In the **yellow**, **U** partition find the vertex with minimum induced deg

For each such vertex:

  Delete it

  Update blue vertex degree

Check the **blue** partition for vertices with degree < β

For each blue node < β:

  Delete node

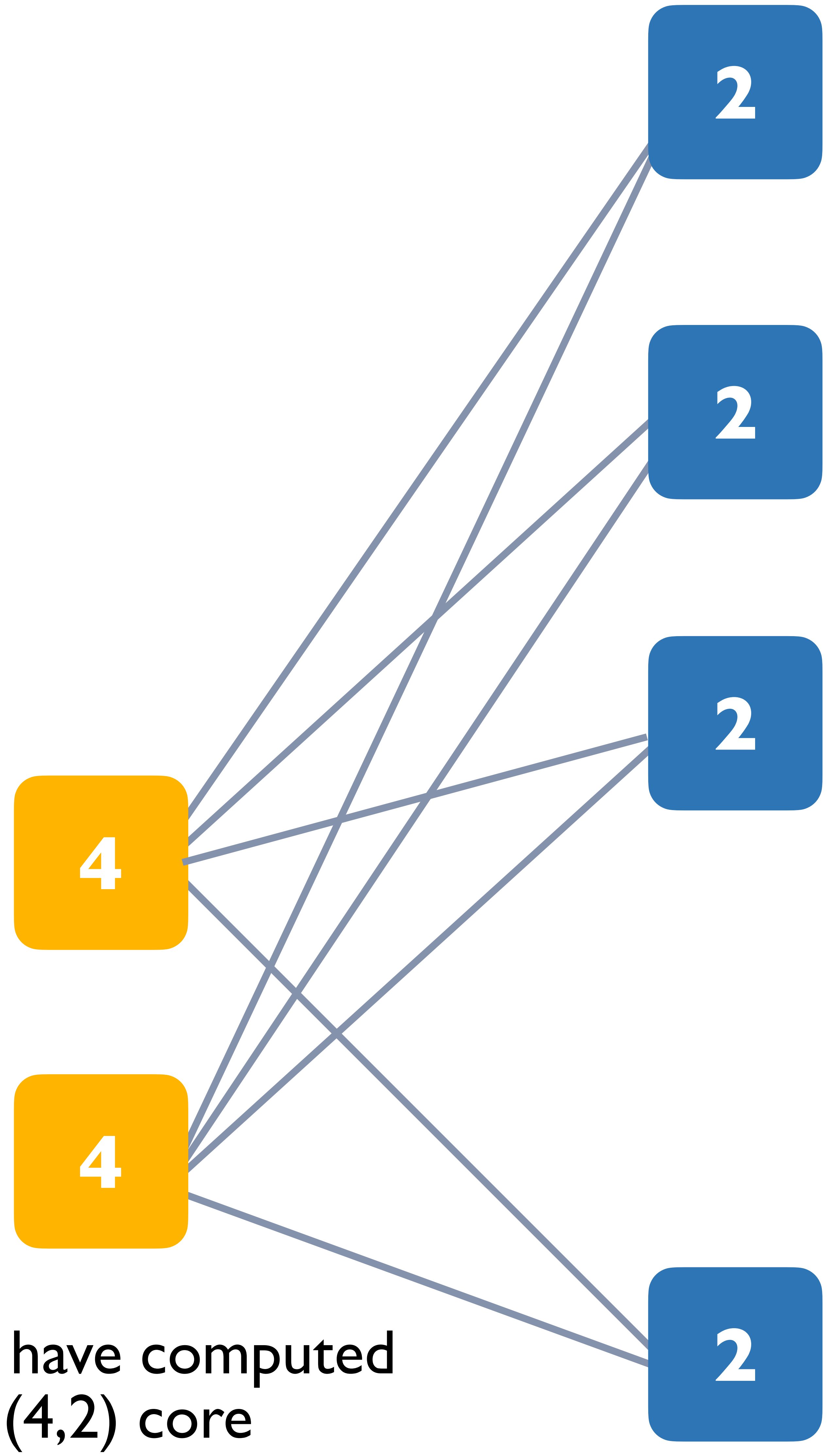
  Update yellow vertex degree

Update yellow

Liu, B., Yuan, L., Lin, X. *et al.* Efficient (α,β)-core computation in bipartite graphs. *The VLDB Journal* **29,** 1075–1099 (2020). https://doi.org/10.1007/s00778-020-00606-9

Sequential Bi-core Decomposition

V, $\beta = 2$



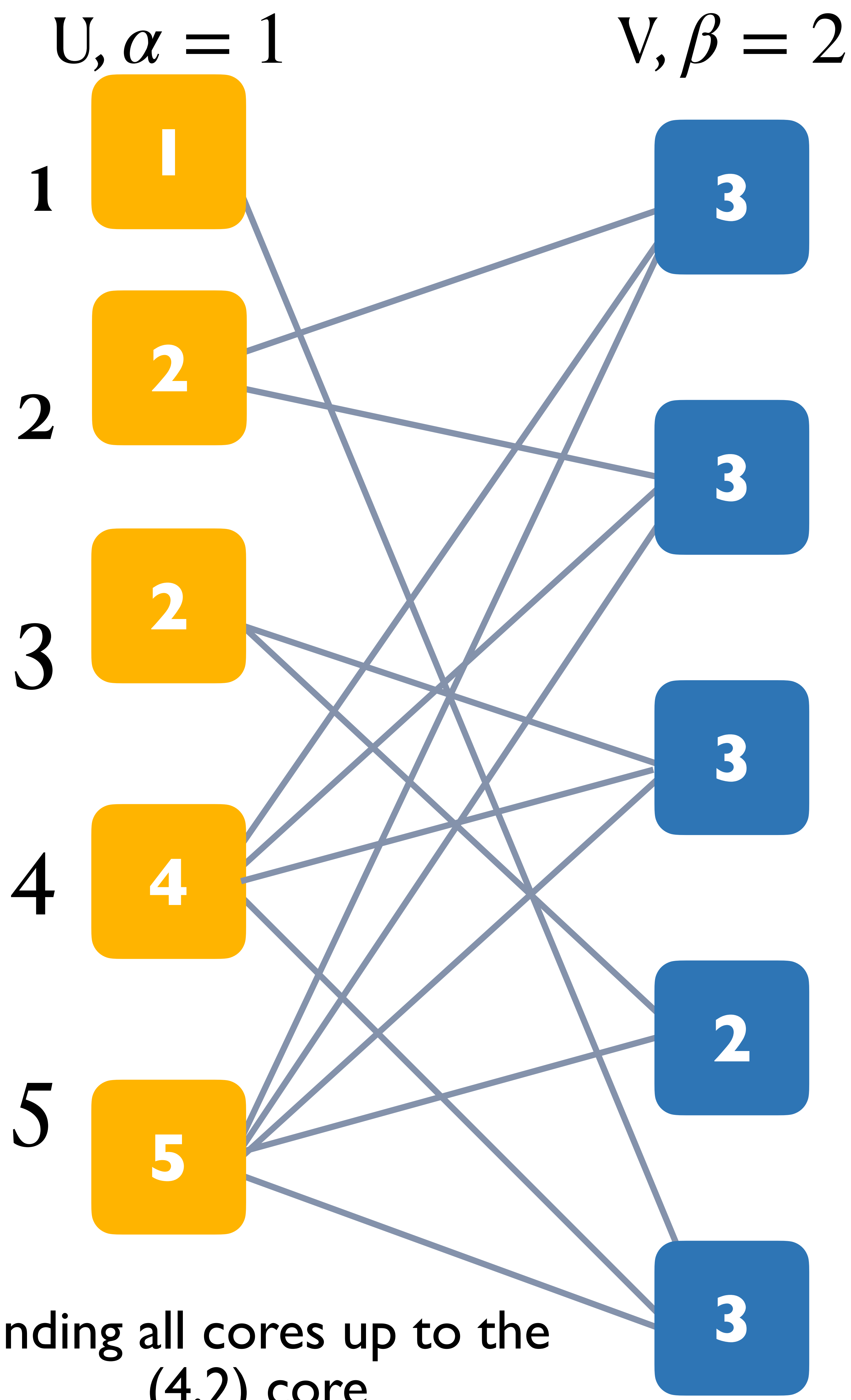
We have computed the (4,2) core

```
In the yellow, U partition
find the vertex with minimum
induced deg

For each such vertex:

  Delete it

  Update blue vertex degree

Check the blue partition for
vertices with degree < β

For each blue node < β:

  Delete node

  Update yellow vertex
  degree

Update yellow
```

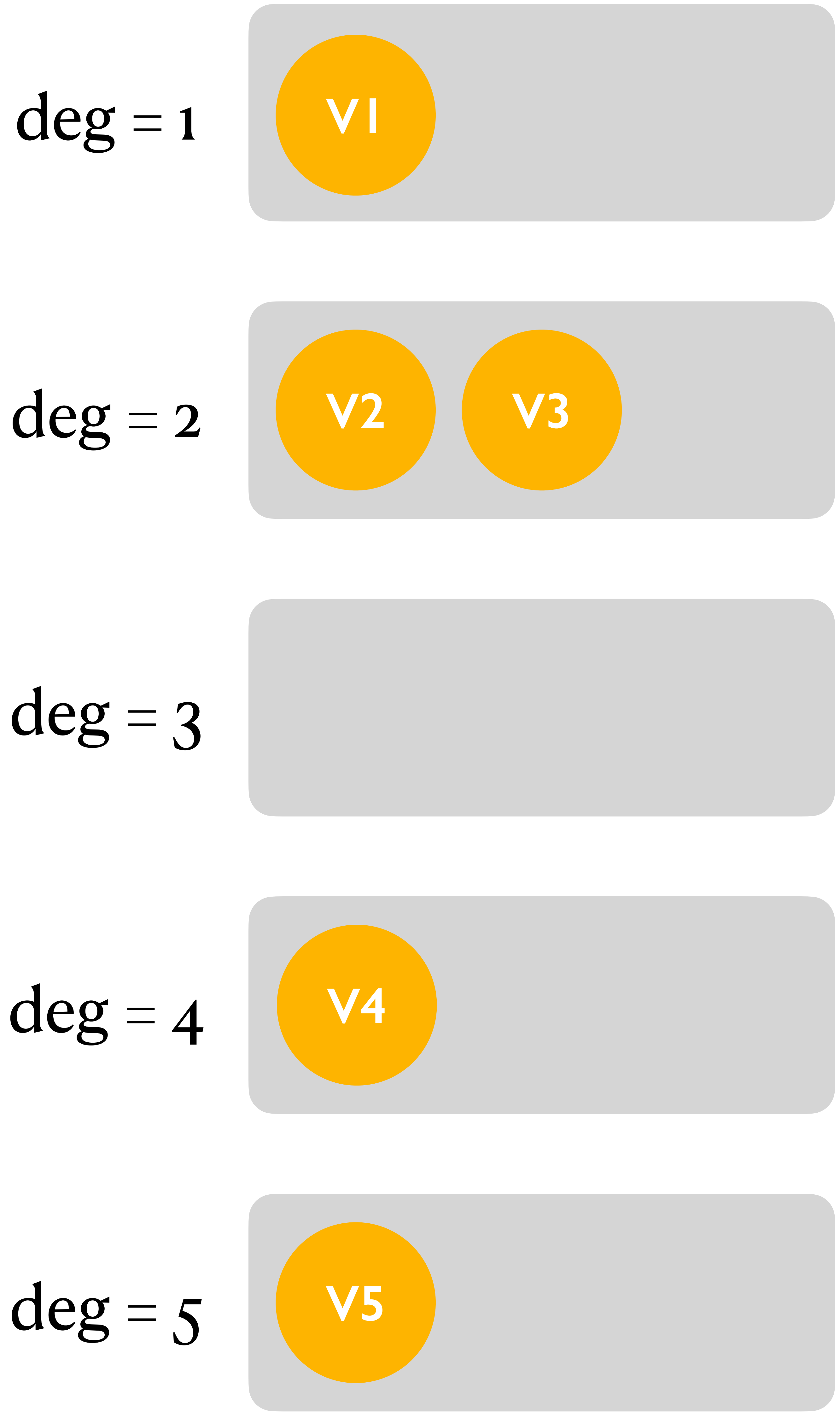

Liu, B., Yuan, L., Lin, X. *et al.* Efficient (α,β)-core computation in bipartite graphs. *The VLDB Journal* **29,** 1075–1099 (2020). https://doi.org/10.1007/s00778-020-00606-9

Algorithm

Parallel Bi-core Decomposition



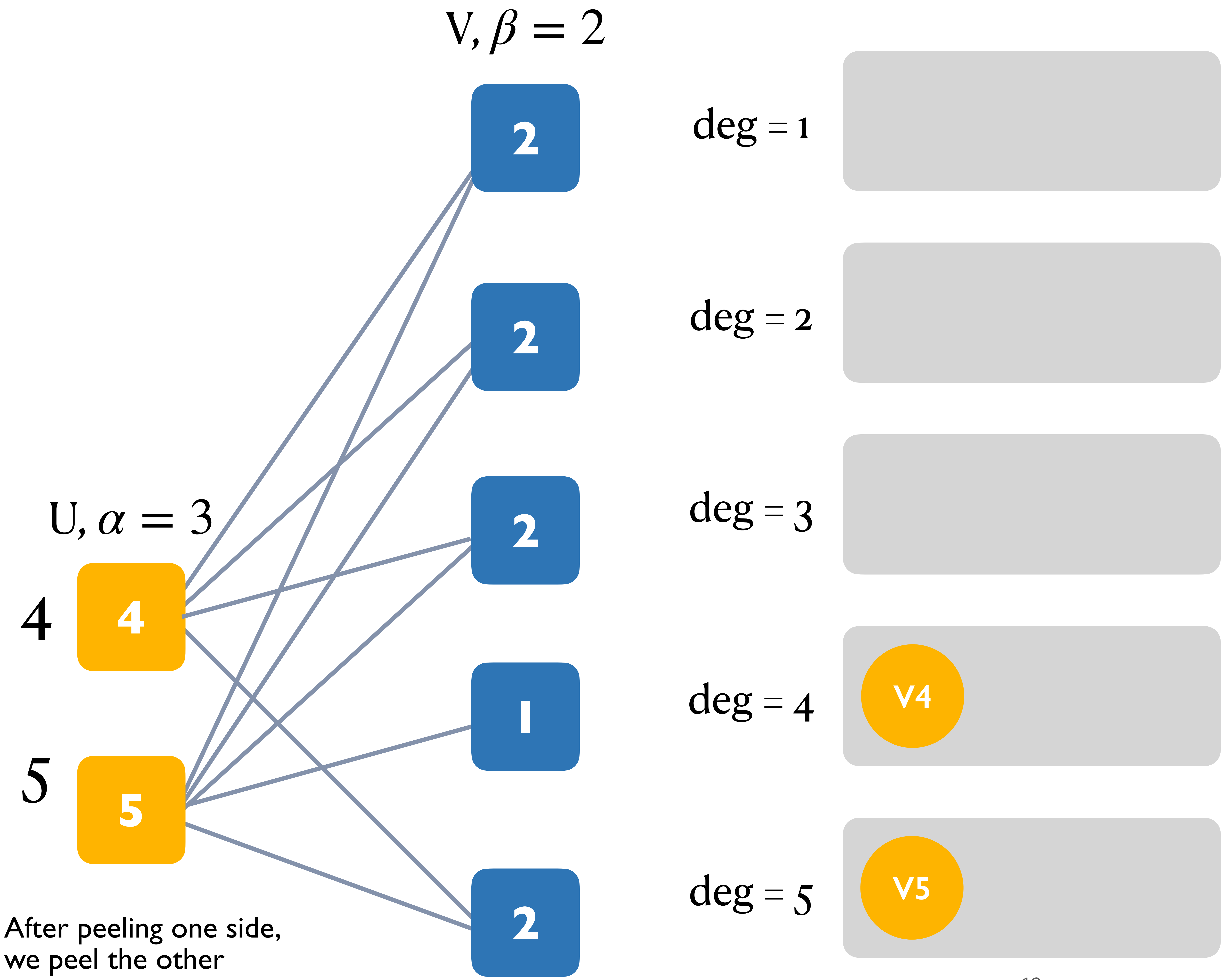
Finding all cores up to the (4,2) core




```
In the yellow, U partition
find all vertices with
minimum induced deg

Parfor each such vertex:

  Delete it

Update blue neighbor
vertex's degree in parallel

Obtain vertices in blue, V
partition with degree < β

Parfor each blue node < β:

  Delete node

Update yellow vertex degree
in parallel
```


Liu, B., Yuan, L., Lin, X. *et al.* Efficient (α,β)-core computation in bipartite graphs. *The VLDB Journal* **29,** 1075–1099 (2020). https://doi.org/10.1007/s00778-020-00606-9

Parallel Bi-core Decomposition



After peeling one side,
we peel the other

In the **yellow, U** partition find all vertices with minimum induced deg

Parfor each such vertex:

  Delete it

Update blue neighbor vertex's degree in parallel

Obtain vertices in **blue, V** partition with degree < β

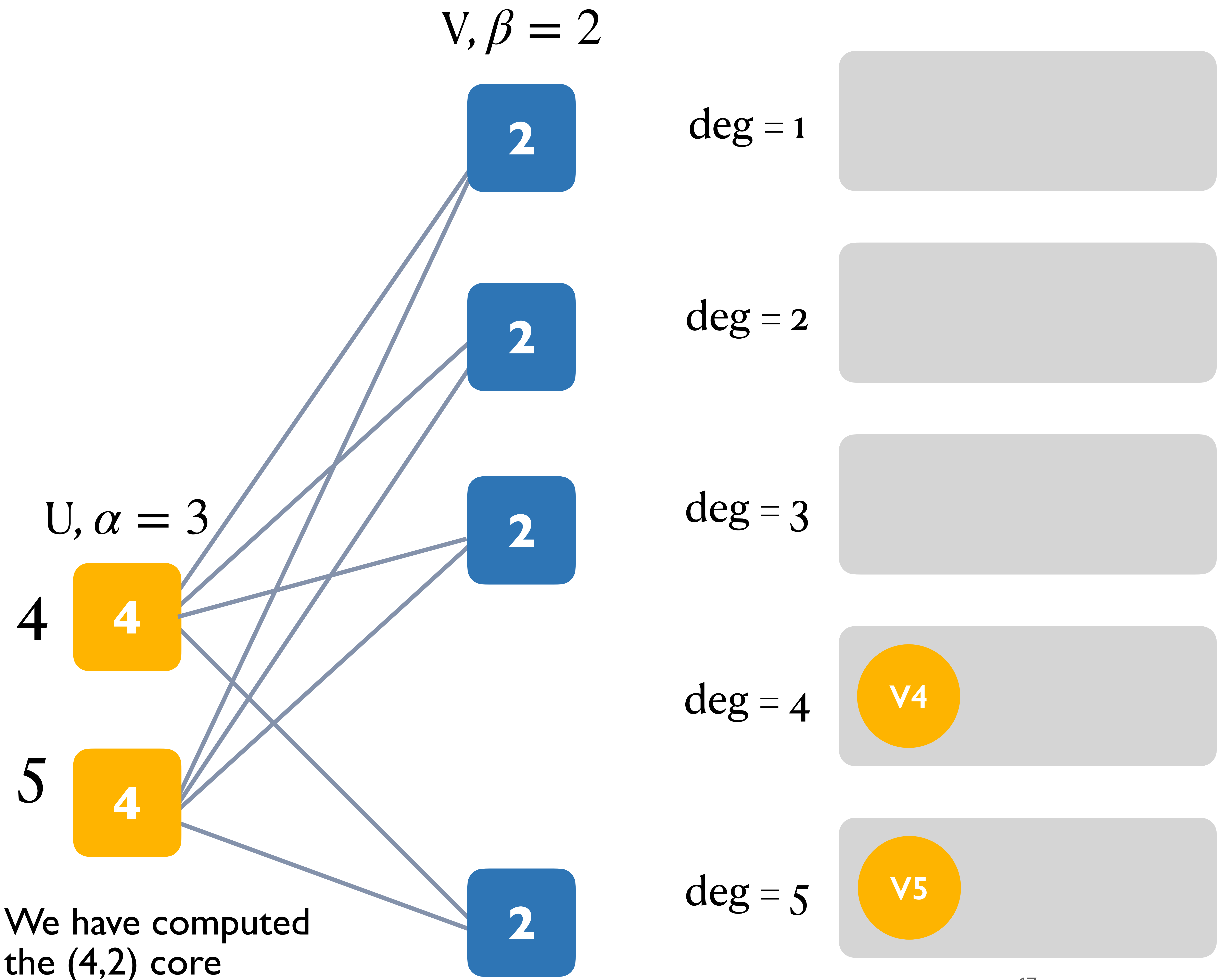
Parfor each blue node < β:

  Delete node

Update yellow vertex degree in parallel

Liu, B., Yuan, L., Lin, X. *et al.* Efficient (α,β)-core computation in bipartite graphs. *The VLDB Journal* **29,** 1075–1099 (2020). https://doi.org/10.1007/s00778-020-00606-9

Parallel Bi-core Decomposition



We have computed the (4,2) core

In the **yellow, U** partition find all vertices with minimum induced deg

Parfor each such vertex:

Delete it

Update blue neighbor vertex's degree in parallel

Obtain vertices in **blue, V** partition with degree < β

Parfor each blue node < β:

Delete node

Update yellow vertex degree in parallel

Liu, B., Yuan, L., Lin, X. *et al.* Efficient (α,β)-core computation in bipartite graphs. *The VLDB Journal* **29,** 1075–1099 (2020). https://doi.org/10.1007/s00778-020-00606-9

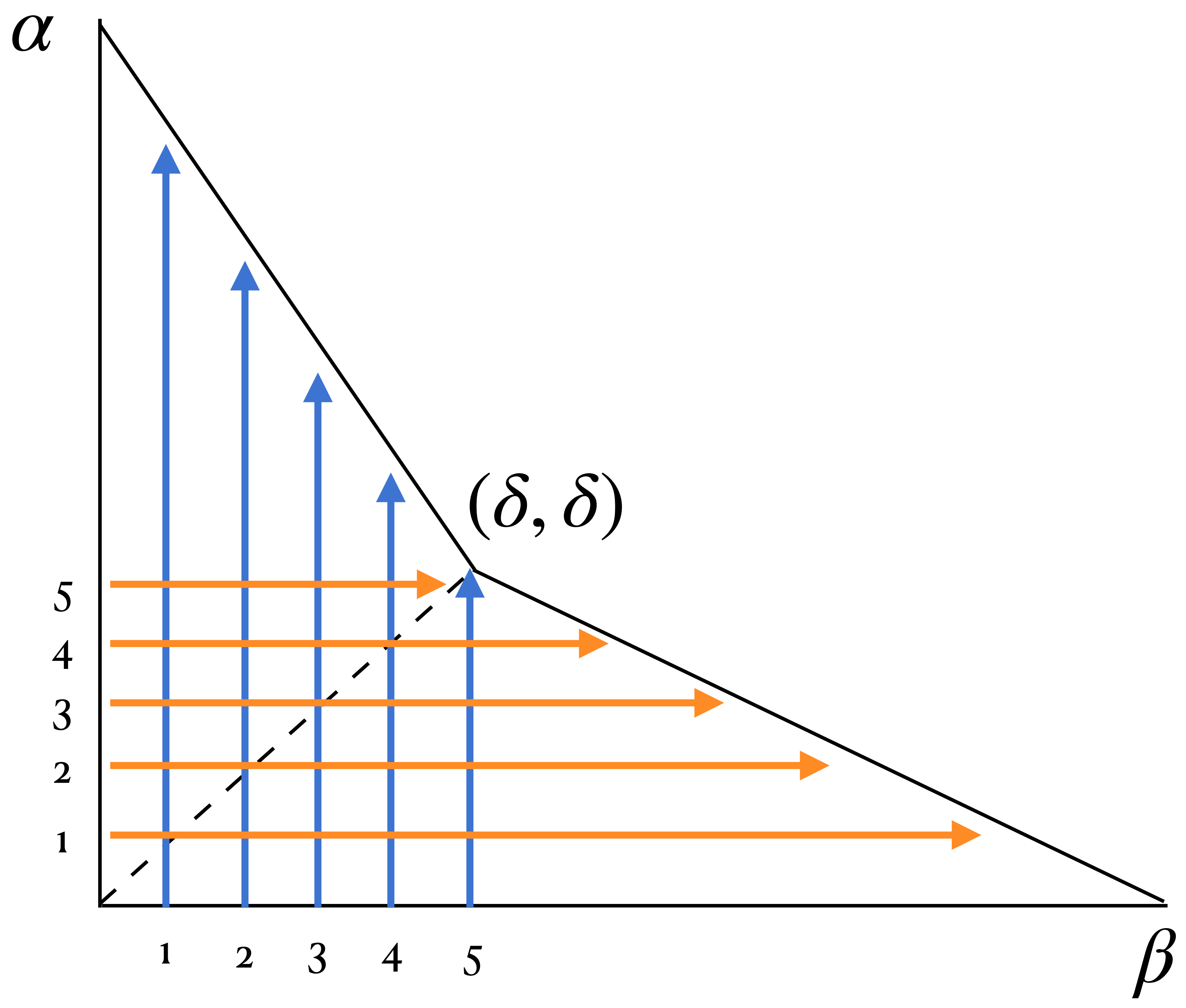
Complexity Results

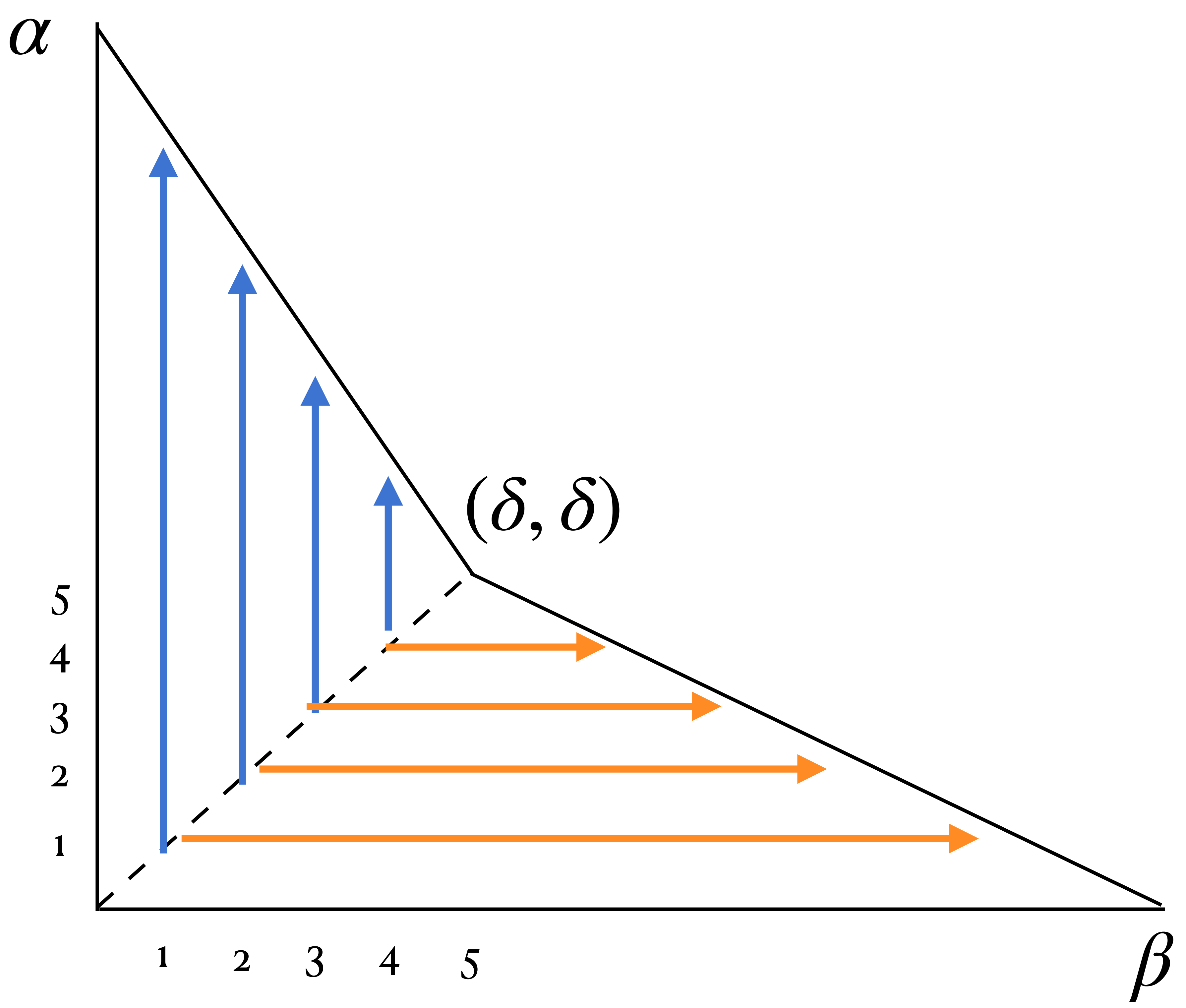
	Liu et al.	Ours
Work	$O(\delta m)$ or $O(m^{1.5})$	$O(\delta m)$ or $O(m^{1.5})$
Span	$O(m)$	$O(\rho \log n)$

(α, β)-core decomposition is P-complete when $\alpha \geq 3$ or $\beta \geq 3$

Peeling-space Pruning

Optimization





Evaluation

- 30-core, 2-way hyperthreading, CPU @3.1 GHz
  - has 60 vCPUs and 240 GB of memory
- We used the GBBS (graph based benchmark suite) to implement our parallel code
- Graphs were from the KONECT graph database
- Largest graph run: orkut (327 million edges)

Graph Name	Type	$\|U\|$	$\|V\|$	n	m	dmax	δ	ρ_{max}
Orkut	Membership	2.78M	8.73M	11.51M	327M	318K	466	12100
Web Trackers	Inclusion	27.7M	284K	40.43M	140.6M	11.57M	437	4542
LiveJournal	Membership S	3.20M	7.49M	13.89M	112M	1.05M	108	6831
TREC	Inclusion	556K	1.17M	1.73M	83.6M	457K	508	6029
Reuters	Inclusion	781K	284K	1.06M	60.6M	345K	192	4767
Epinions	Rating	120K	755K	880k	13.67M	162K	151	3049
Flickr	Membership	396K	104K	500k	8.55M	35K	147	2300

Table 2. Graphs Statistics

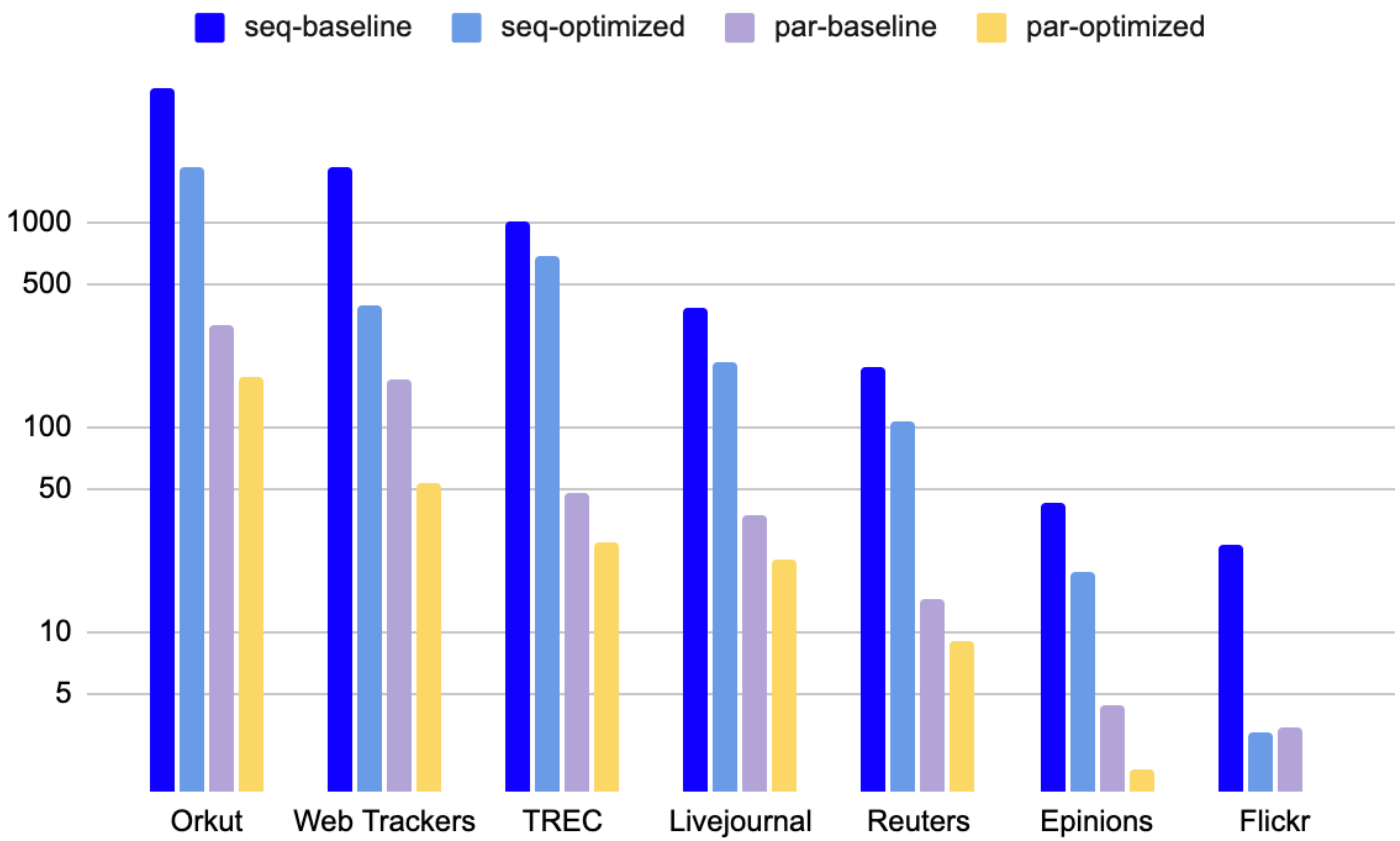
Google Cloud Platform

KONECT -- The Koblenz Network Collection.
Jerone Kunegis 2013. konect.cc/networks

Theoretically Efficient Parallel Graph Algorithms Can Be Fast and Scalable: https://github.com/ParAlg/gbbs, 2018

Runtime comparison

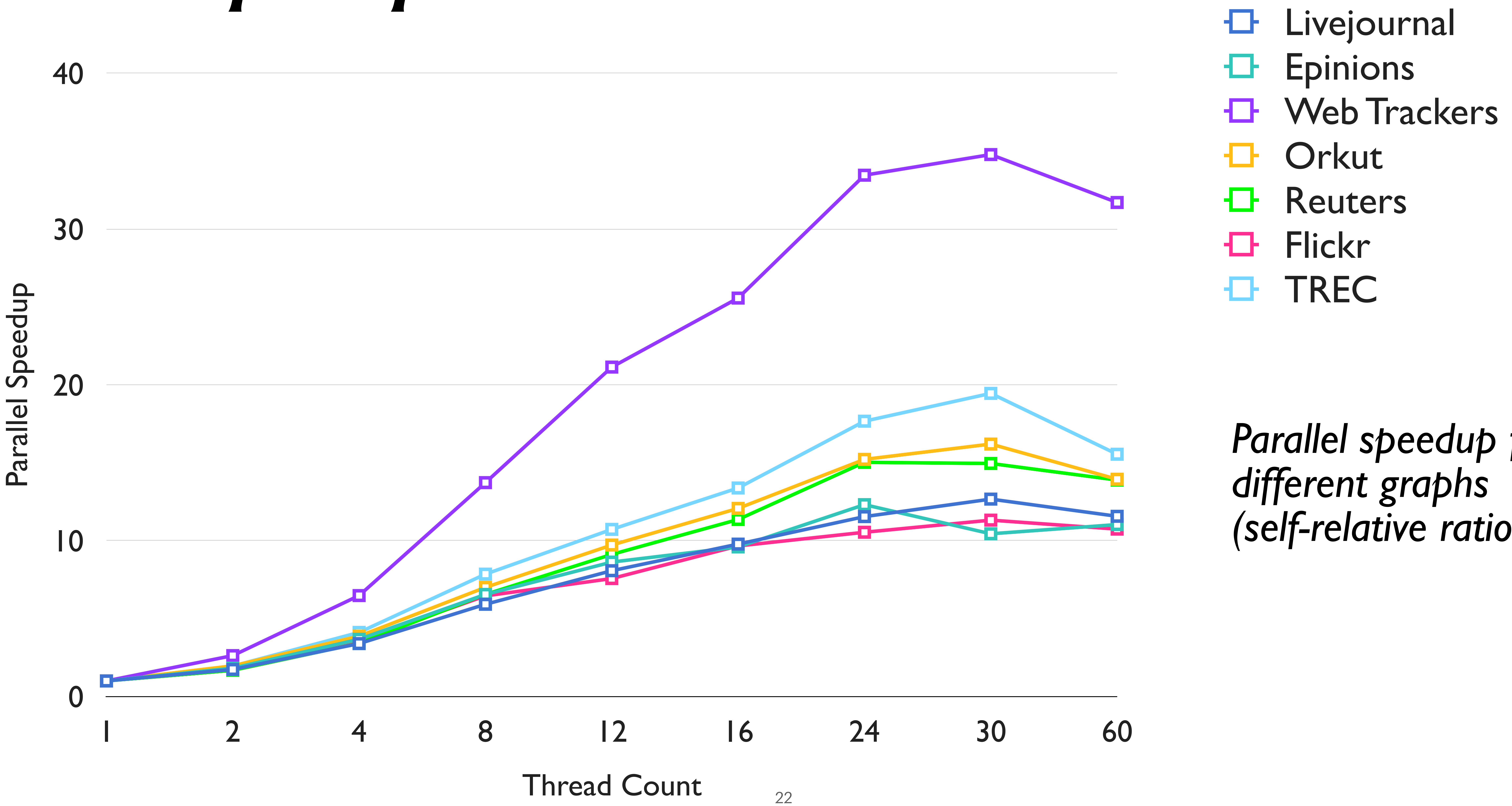
- *4.1x speedup over Liu et al.'s parallelization*
- *16.2—35.5x self-relative speedup*



sequential vs parallel run times

Log scale

Parallel Speedup



Parallel speedup for different graphs (self-relative ratios).

Conclusion

- A work-efficient shared memory algorithm that improves upon the span of previous work
- We achieve 35.5x max self-relative speedup
- Github: https://github.com/clairebookworm/gbbs

Future Work

- Dynamic bi-core peeling
- Extrapolate to bi-clique decomposition (which is a generalization of butterfly decomposition)
- Study the tradeoff between work-efficiency and practical speed

Acknowledgements

We'd like to thank Jessica Shi and Prof. Julian Shun for their support and mentorship, as well as the MIT PRIMES program for this opportunity.

Any questions?