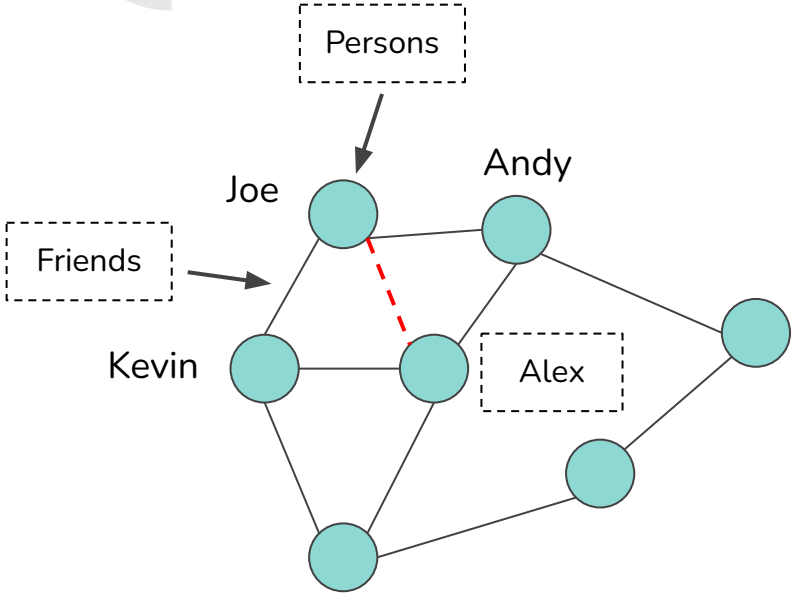
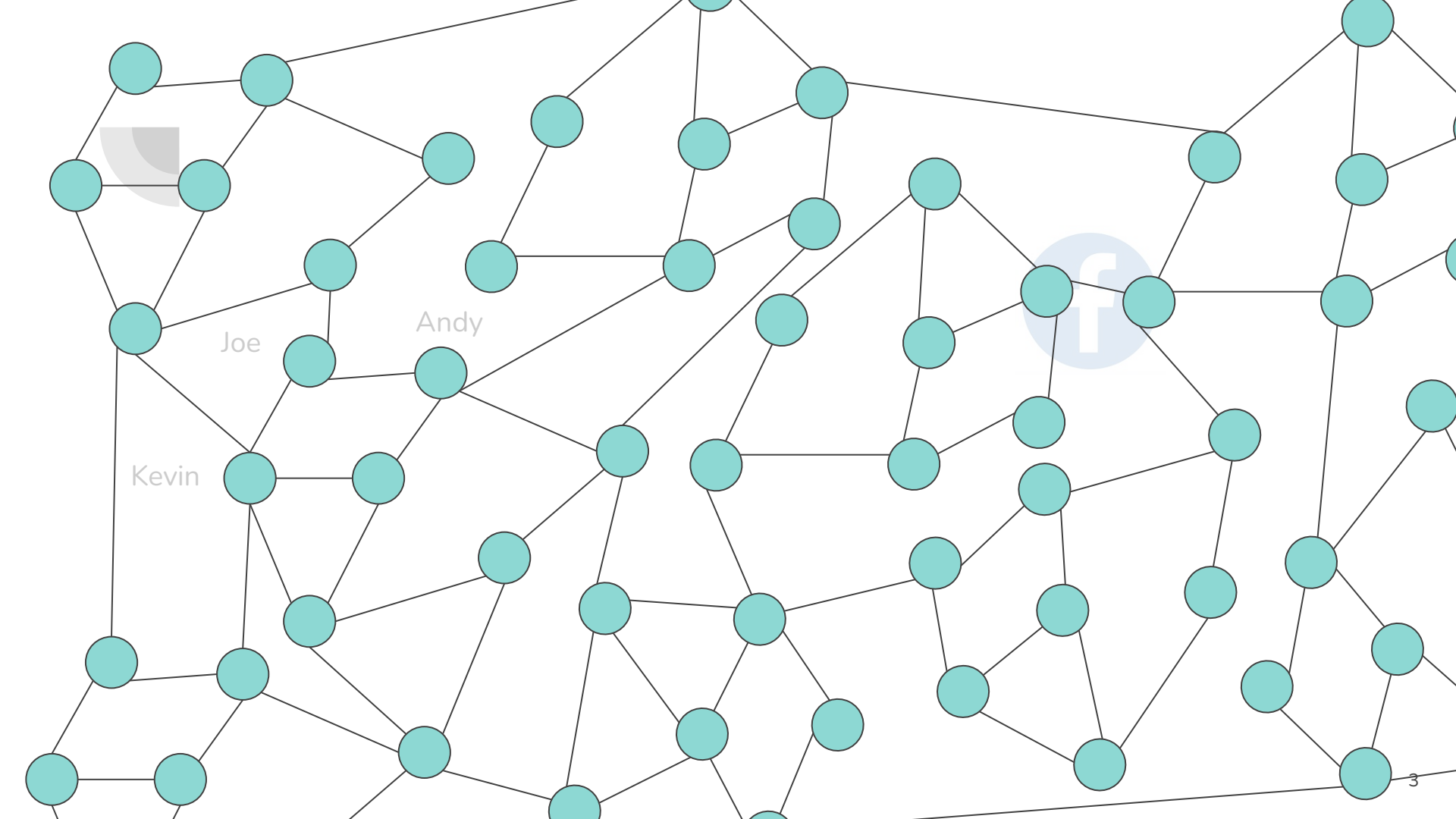
ParViz: Visualizing Graph Partitioners

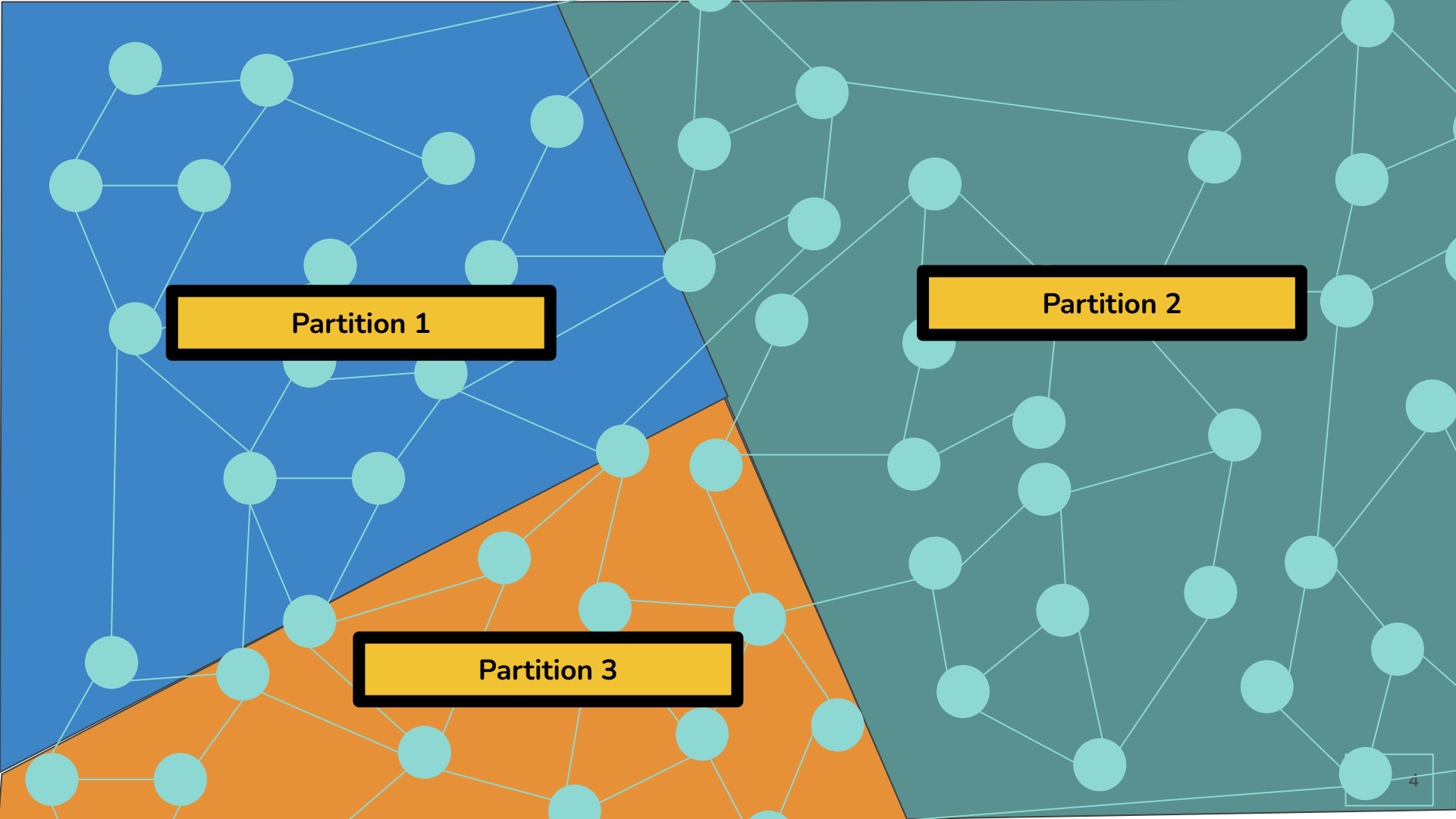
Hadi Sinaee

CPSC 547, December 2021
University of British Columbia

Motivation




Joe
Andy
Kevin


Partition 1
Partition 2
Partition 3

Graph Partitioners

- To create a new partitioner we need to understand the previous ones work
- Different metrics for graph partitioners:
  - Load Balance **(LB)**
  - Edge-Cuts **(EC)**
  - Duplication Factor **(DF)**
- These metrics are measured at the end of the partitioning process
  - It’d help to see how they were evolving

```
nodeId,partNum,df,lb,ec,part_edges,loaded_part_nodes
1404783,1,0.0000002451,0.0000015968,1,1,1
1739659,1,0.0000014709,0.0000031936,5,6,2
68215,1,0.0000029418,0.0000047905,10,12,3
615877,1,0.0000046578,0.0000063873,16,19,4
1037258,1,0.0000056384,0.0000079841,19,23,5
1065836,1,0.0000095608,0.0000095809,34,39,6
1697811,1,0.0000115220,0.0000111777,41,47,7
223282,1,0.0000122574,0.0000127745,43,50,8
706398,1,0.0000125026,0.0000143714,43,51,9
1187234,1,0.0000137283,0.0000159682,47,56,10
1354787,1,0.0000286824,0.0000175650,107,117,11
1405904,1,0.0000318693,0.0000191618,118,130,12
1849008,1,0.0000338305,0.0000207586,125,138,13
170694,1,0.0000357917,0.0000223555,132,146,14
874187,1,0.0000367723,0.0000239523,135,150,15
905472,1,0.0000421655,0.0000255491,156,172,16
1177215,1,0.0000426558,0.0000271459,157,174,17
1719979,1,0.0000455976,0.0000287427,168,186,18
1780232,1,0.0000544230,0.0000303395,203,222,19
1782500,1,0.0000590808,0.0000319364,221,241,20
790822,1,0.0000598162,0.0000335332,223,244,21
1001695,1,0.0000627580,0.0000351300,234,256,22
1049690,1,0.0000627580,0.0000367268,233,256,23
1313771,1,0.0000632483,0.0000383236,234,258,24
53174,1,0.0000676610,0.0000399205,251,276,25
148393,1,0.0000747703,0.0000415173,279,305,26
307904,1,0.0000769766,0.0000431141,287,314,27
326874,1,0.0000796732,0.0000447109,297,325,28
421588,1,0.0000808990,0.0000463077,301,330,29
457890,1,0.0000894792,0.0000479045,335,365,30
698470,1,0.0000919307,0.0000495014,344,375,31
705699,1,0.0000997754,0.0000510982,375,407,32
857700,1,0.0001002657,0.0000526950,376,409,33
941536,1,0.0001005109,0.0000542918,376,410,34
1184819,1,0.0001039429,0.0000558886,389,424,35
1362294,1,0.0001130134,0.0000574855,425,461,36
1510820,1,0.0001169358,0.0000590823,440,477,37
1552389,1,0.0001174261,0.0000606791,441,479,38
1634687,1,0.0001225742,0.0000622759,460,500,39
```

ParViz: Visualizing Graph Partitioners

“Build a viz tool to help algorithm designer understand the partitioning process of a partitioner.”

Task Abstraction

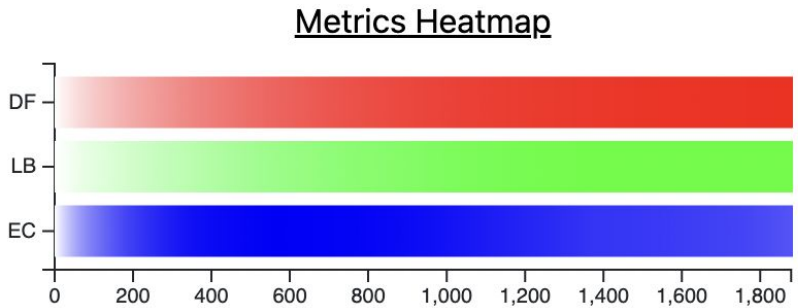
1. **How do partitioning metrics evolve during the partitioning process?**
2. **How does the size of partitions change over time?**
3. What does the final partitioned graph look like?

Dataset

1. The input graph is the Yahoo! Messenger dataset which is a public dataset
    a. An undirected graph
    b. ~1.9M nodes and 4M edges
    c. Nodes are users and the edges show at least one the nodes is a contact of another one
2. Number of partitions is three
3. The graph was partitioned by a vertex partitioner named Fennel
    a. We assign each node and all of its edges into one partition
4. Dataset type is Table with Items and Attributes
    a. Input format is CSV

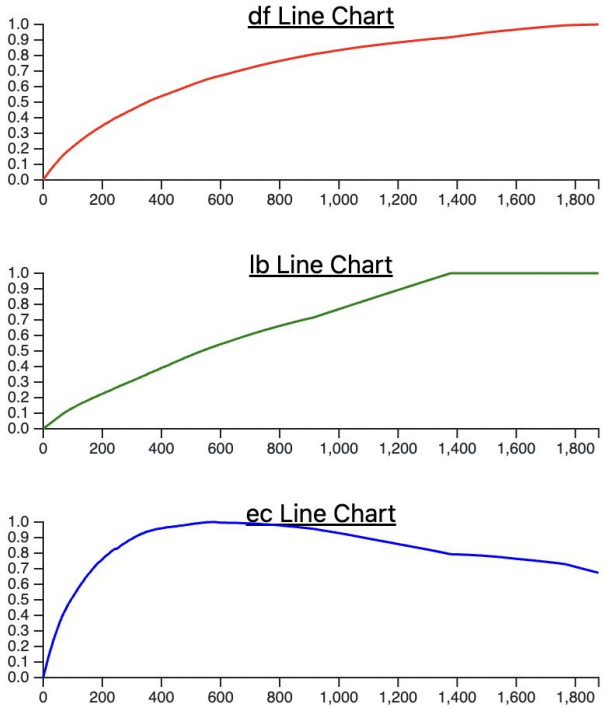
What-How-Why

What: Data	Table; **DF**, **LB**, **EC** quantitative attributes
What: Derived	• Partitioner Steps: Ordered key attribute • Normalized Values of **DF**, **LB**, **EC**
How: Encode	Express **DF**, **LB**, **EC** horizontally; using a heat map with different hues and changing saturation
How: Reduce	Sampled at every 1000 point
Why: Task	Overview of changes in metrics
Scale	Items: ~200K



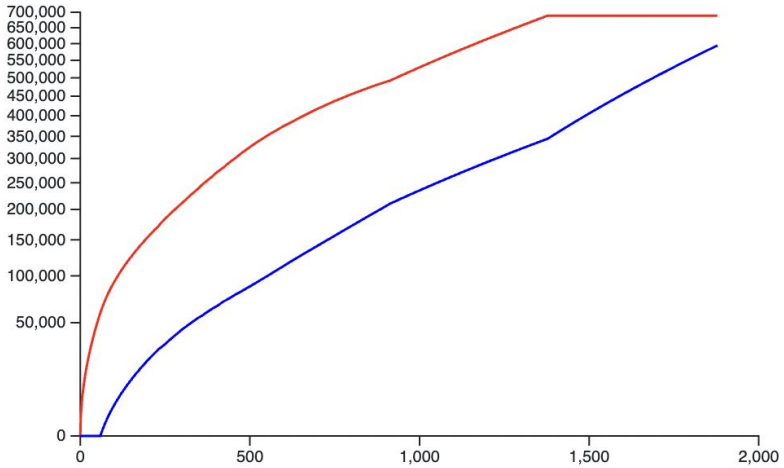
What-How-Why

What: Data	Table; **DF** quantitative attributes
What: Derived	• Partitioner Step: Ordered key attributed • Normalized **DF**
How: Encode	Express **DF** horizontally; the x-axis is the steps and y-axis shows the DF value
How: Reduce	Sampled every 1000 steps
Why: Task	Overview of changes in DF
Scale	Items: ~200K



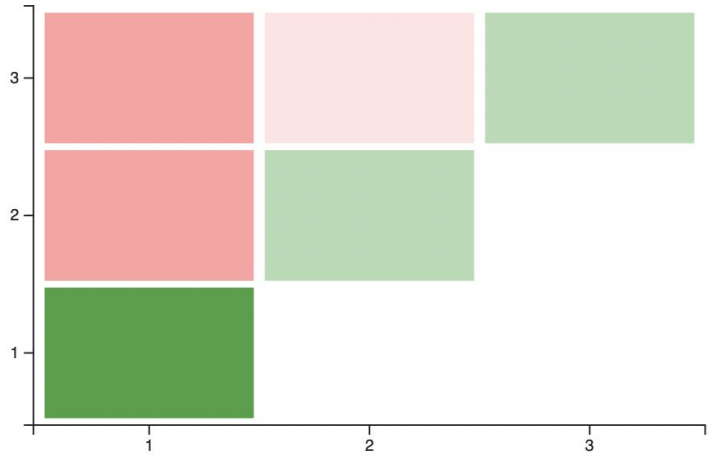
What-How-Why

What: Data	Table; **NodeId** key attribute **PartNum** categorical key attribute
What: Derived	• Matrix of the size: 200K * 3 • Each cell is shows the number nodes at a specific step in a specific partition
How: Encode	Facet; Superimpose the size of partitions for at each step
How: Reduce	Sampled every 1000 steps
Why: Task	Overview of changes in partitions size
Scale	Items: ~200K * 3



What-How-Why

What: Data	Table; **Part1, Part2** categorical key attribute **ec** quantitative attribute **nedge_part** quantitative attribute **nedge** quantitative attribute
What: Derived	• Matrix of the size: 3 * 3 • Each cell is shows the normalized ec between two corresponding partitions
How: Encode	Express the normalized ec value using a heat map
Why: Task	Overview of ec in partitions size
Scale	Items: 3 * 3



Limitations/Critiques

- Limitations
  - Limited to vertex partitioners
  - Doesn’t scale for large graphs with more than 1M nodes after sampling
  - Doesn’t scale for large number of partitions, e.g. 256.
  - Lack of interactivity
  - Cannot compare two different algorithms against each other
- Critiques
  - Using RGB for different categories
  - D3 was not the right choice for the mentioned tasks
  - Some tasks are done manually, such changing the file
  - Lack of proper legends
  - Too many idioms