

<http://poloclub.gatech.edu/cse6242>

CSE6242 / CX4242: **Data** & **Visual** Analytics

MMap (Memory Mapping)

Simple, minimalist approach to scale up computation

Duen Horng (Polo) Chau

Assistant Professor

Associate Director, MS Analytics

Georgia Tech

Partly based on materials by

Professors Guy Lebanon, Jeffrey Heer, John Stasko, Christos Faloutsos, Parishit Ram (GT PhD alum; SkyTree), Alex Gray

**When should you use
Spark/Hadoop, AWS, Azure?**

And when should you **not?**

MMap

Fast Billion-Scale Graph Computation on a PC via **Memory Mapping**



Lead by

Zhiyuan (Jerry) Lin

Georgia Tech CS Undergrad

Now: Stanford 1st year PhD student

MMap: Fast Billion-Scale Graph Computation on a PC via Memory Mapping. Zhiyuan Lin, Minsuk Kahng, Kaeser Md. Sabrin, Duen Horng Chau, Ho Lee, and U Kang. *Proceedings of IEEE BigData 2014 conference*. Oct 27-30, Washington DC, USA.

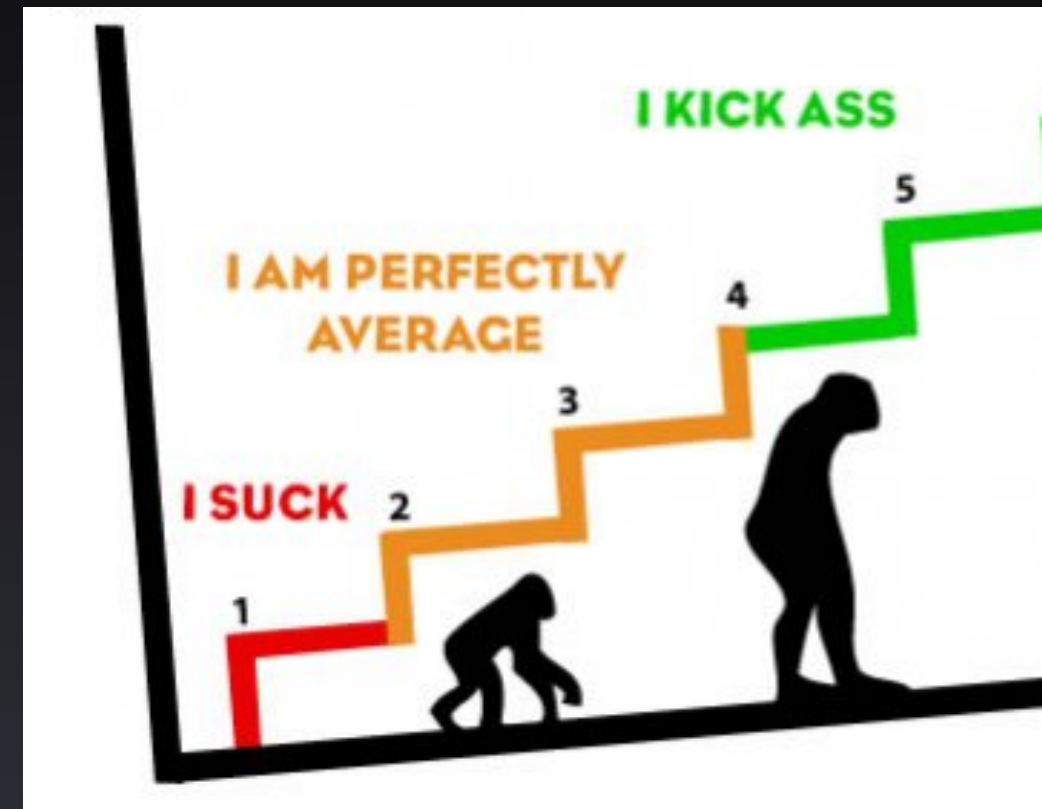
Towards Scalable Graph Computation on Mobile Devices. Yiqi Chen, Zhiyuan Lin, Robert Pienta, Minsuk Kahng, Duen Horng (Polo) Chau. *IEEE BigData 2014 Workshop on Scalable Machine Learning: Theory and Applications*.

Graph Computation on Computer Cluster?

Steep learning curve

Cost

Overkill for smaller graphs



Best-of-breed Single-PC Approaches

- **GraphChi** – OSDI 2012
- **TurboGraph** – KDD 2013

What do they have in common?

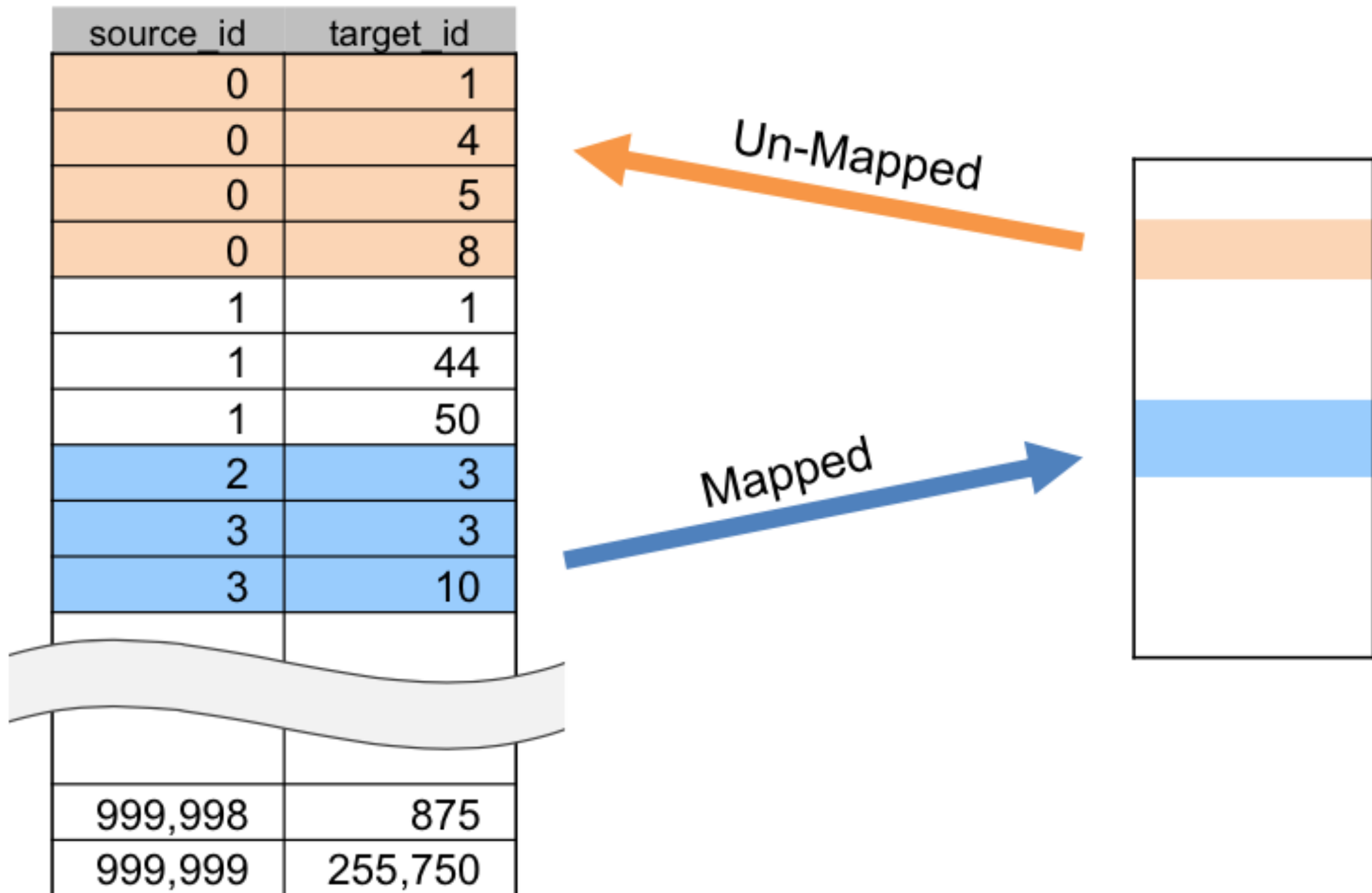
- Sophisticated Data Structures
- Explicit Memory Management

Can We Do Less?

To get same or **better** performance?

e.g., auto memory management, faster, etc.

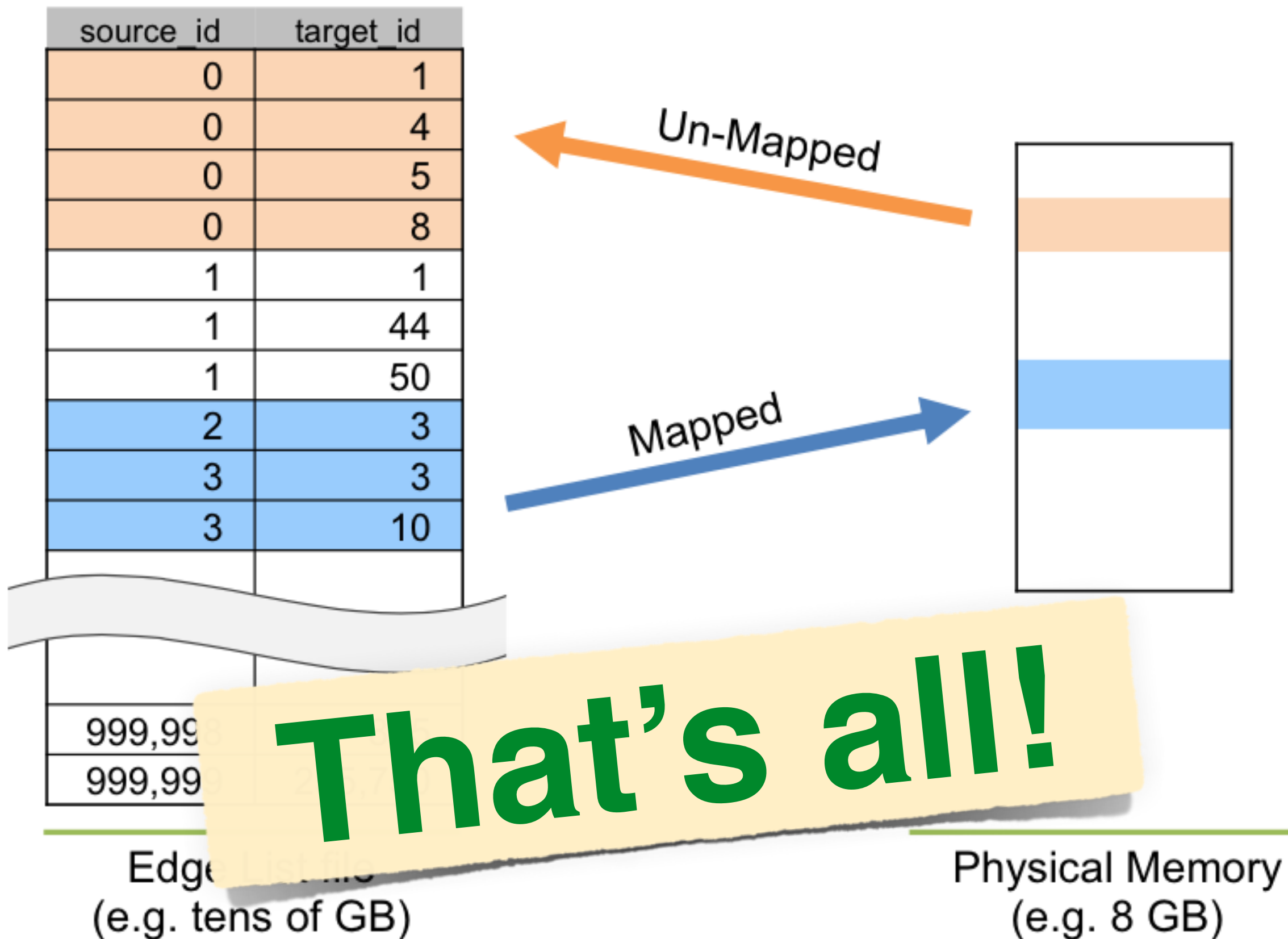
Main Idea: **Memory-mapped** the Graph



Edge List file
(e.g. tens of GB)

Physical Memory
(e.g. 8 GB)

Main Idea: **Memory-mapped** the Graph



How to compute PageRank for huge matrix?

Reminder

Use the power iteration method

http://en.wikipedia.org/wiki/Power_iteration

$$\mathbf{p}' = c \mathbf{B} \mathbf{p} + \frac{(1-c)}{n} \mathbf{1}$$

\mathbf{p}'

\mathbf{B}

\mathbf{p}

$$\begin{bmatrix} p1 \\ p2 \\ p3 \\ p4 \\ p5 \end{bmatrix}$$

=

c

$$\begin{bmatrix} & & 1 & & \\ 1 & & & 1 & \\ & 1/2 & & & 1/2 \\ & & & & 1/2 \\ & 1/2 & & & \end{bmatrix}$$

+ $\frac{(1-c)}{n}$

$$\begin{bmatrix} p1 \\ p2 \\ p3 \\ p4 \\ p5 \end{bmatrix}$$

$$\begin{bmatrix} 1 \\ 1 \\ \dots \\ 1 \end{bmatrix}$$

Can initialize this vector to any non-zero vector, e.g., all “1”s

Example: PageRank (implemented using MMap)

<http://www.cc.gatech.edu/~dchau/papers/14-bigdata-mmap.pdf>

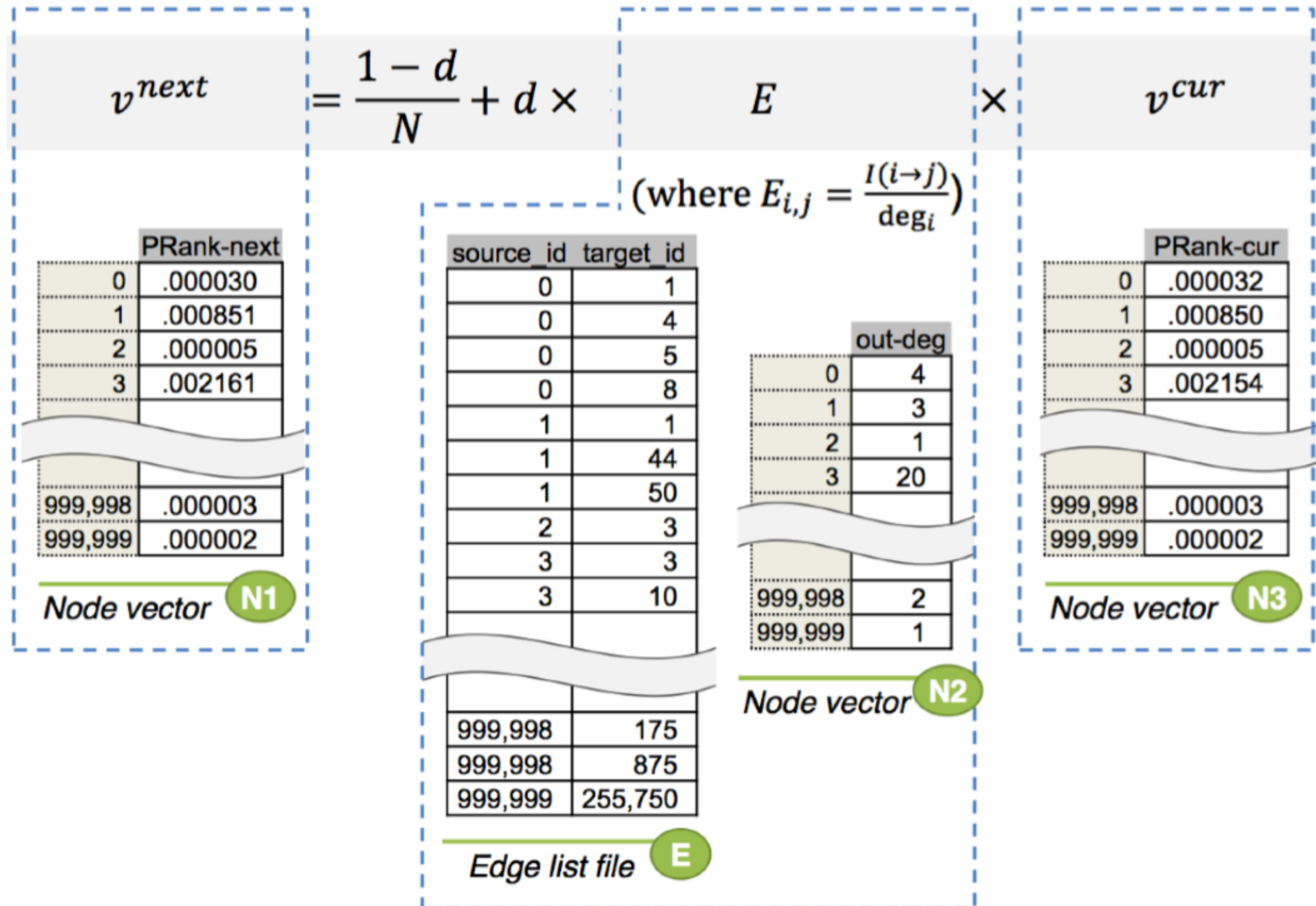
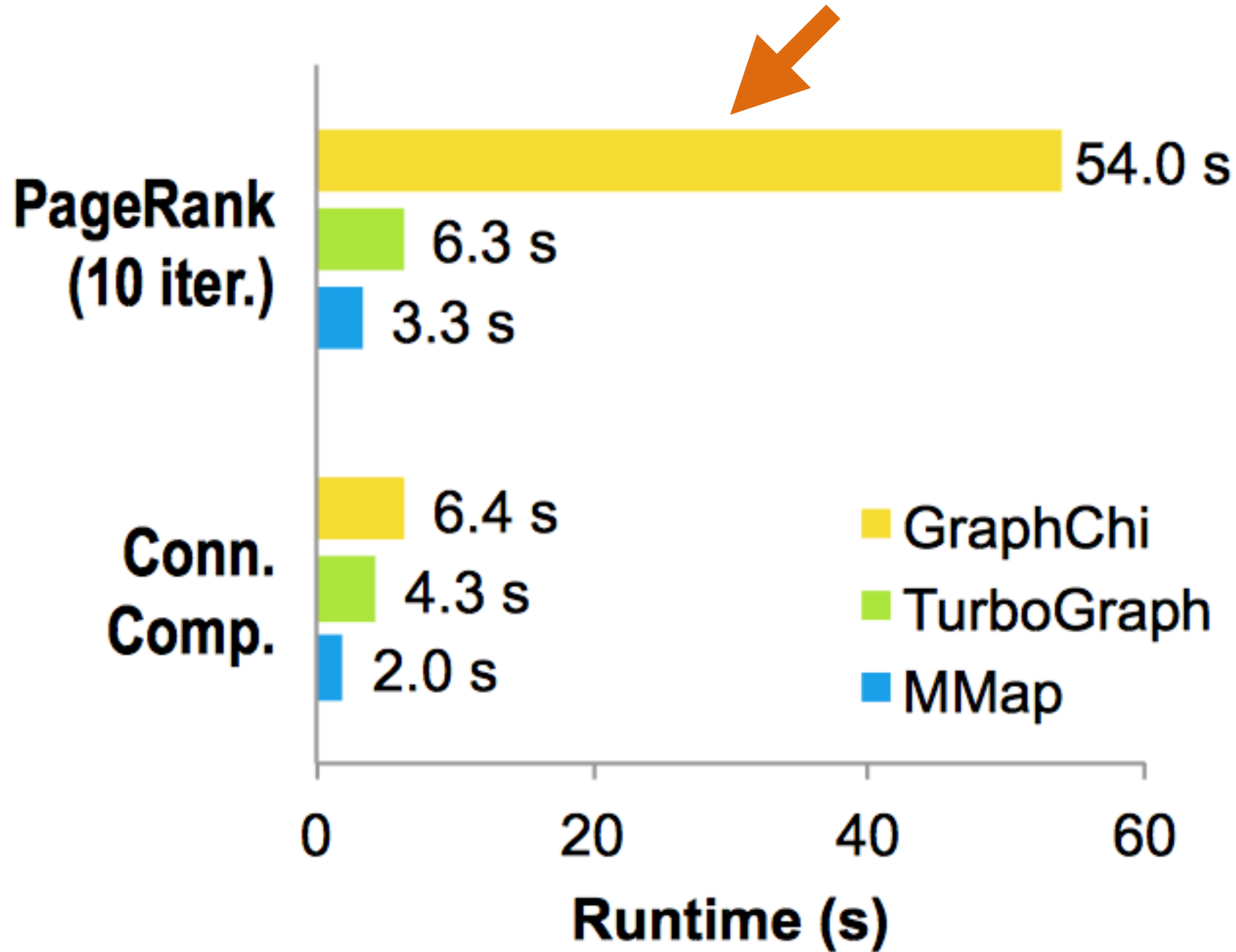
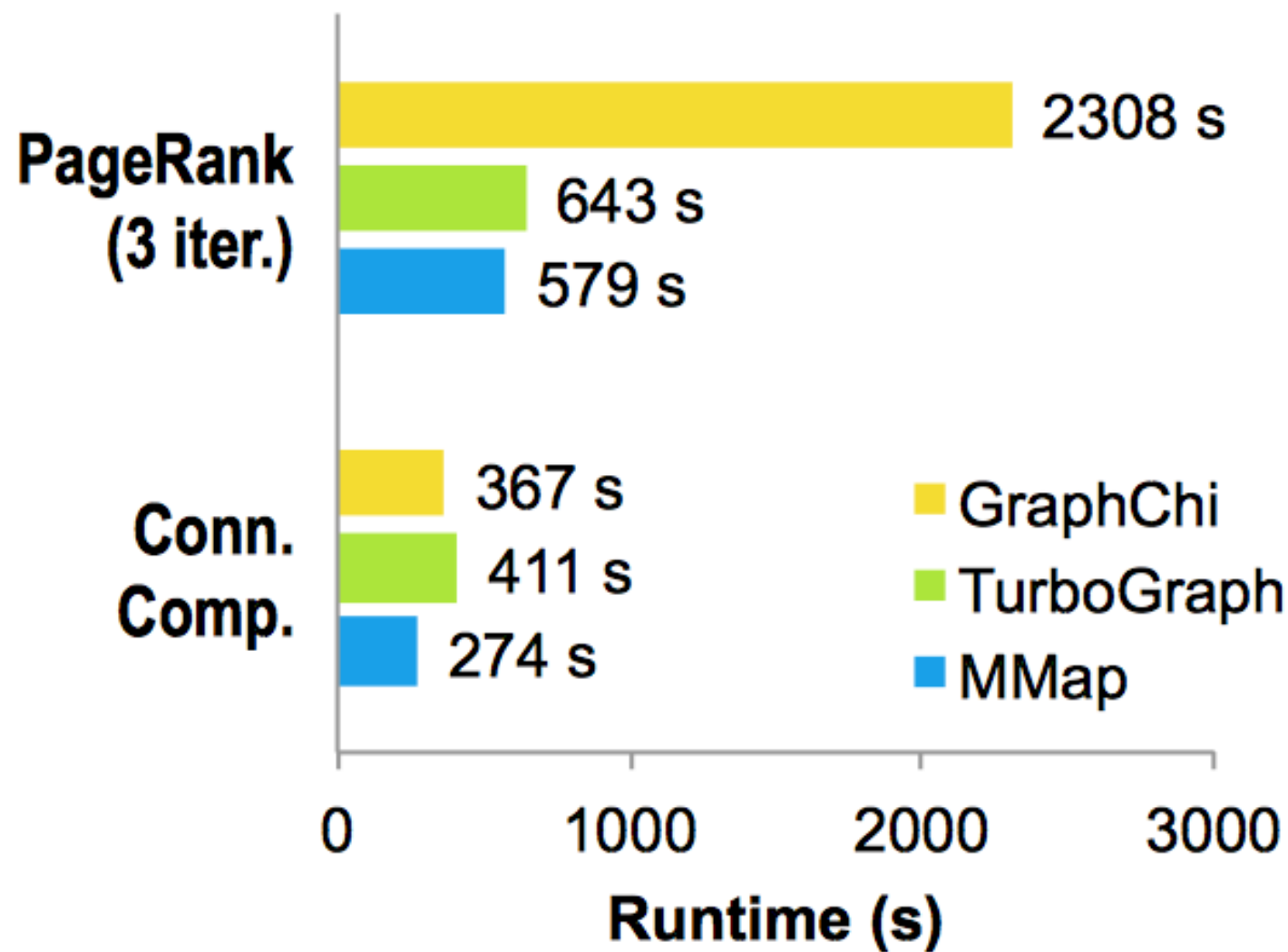


Fig. 3: Data structures used for computing PageRank. In our PageRank implementation, a binary edge list file and three

8000 lines of code

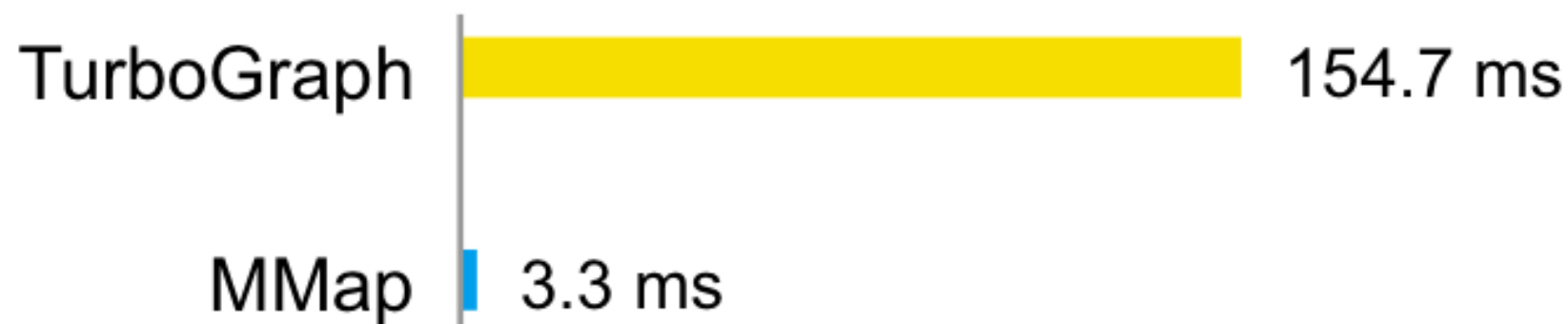


(a) LiveJournal graph (69M edges)



(c) YahooWeb graph (6.6B edges)

1-step Neighbor Query Runtime on YahooWeb Graph (6.6 billion edges)



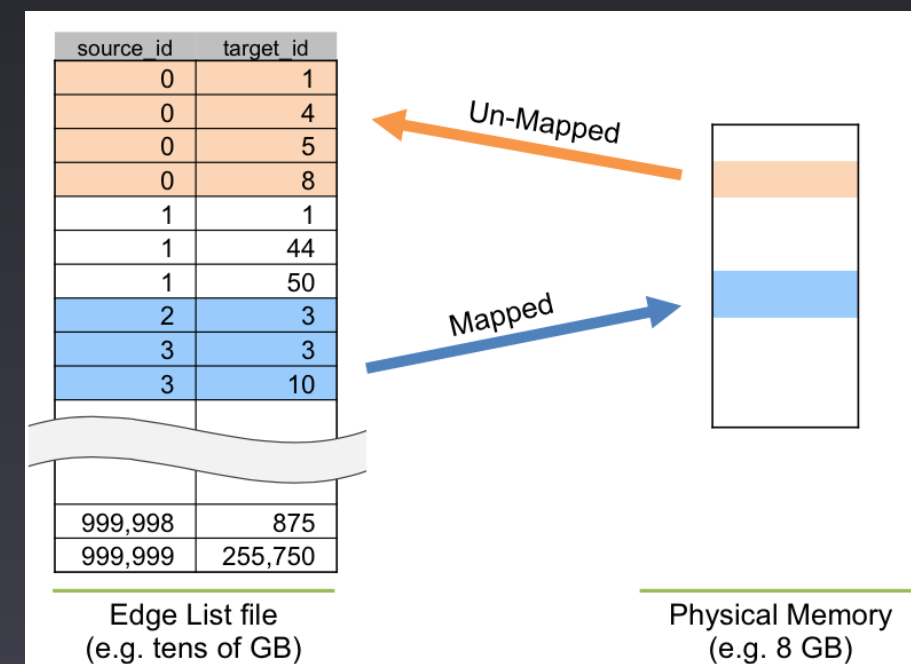
Why Memory Mapping Works?

High-degree nodes' info automatically cached/kept in memory for future frequent access

Read-ahead paging preemptively loads edges from disk.

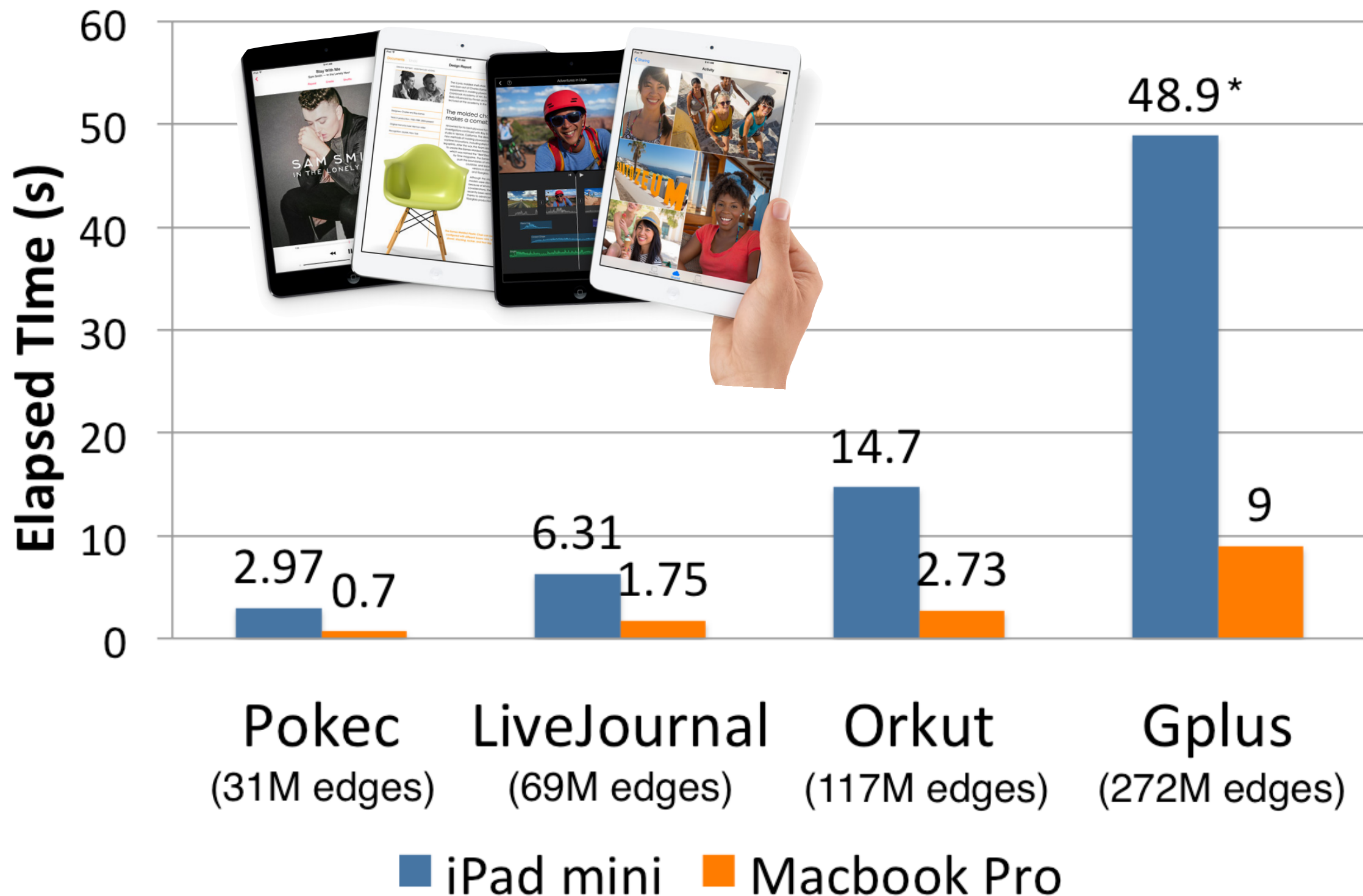
Highly-optimized by the OS

No need to explicitly manage memory
(less book-keeping)

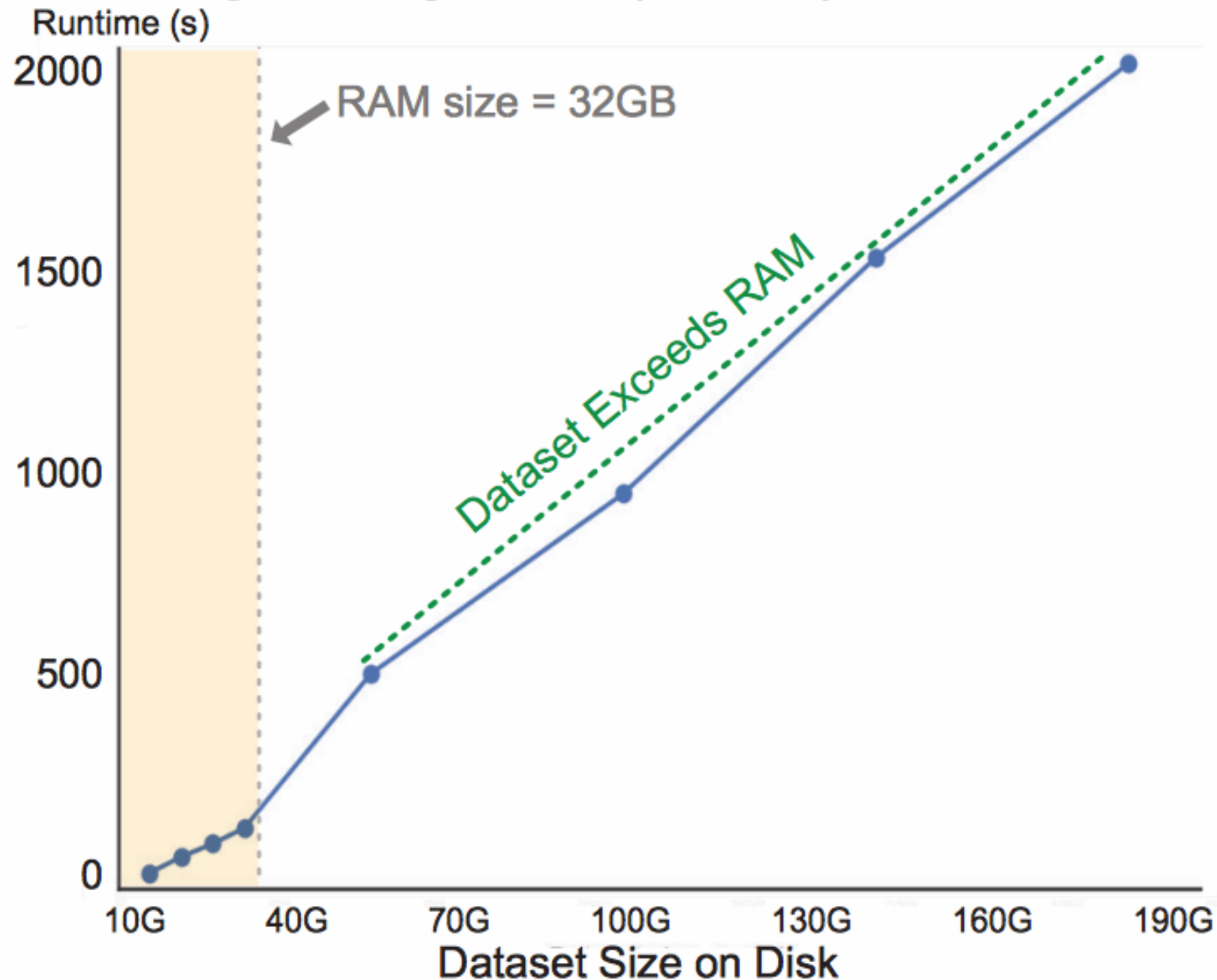


Also works on tablets! (If you want.)

Big Data on Small Devices (270M+ Edges)

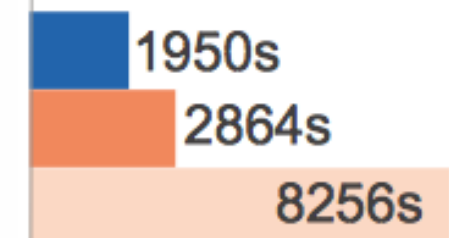


a M3 Runtimes (10 Iterations) for Logistic Regression (L-BFGS)

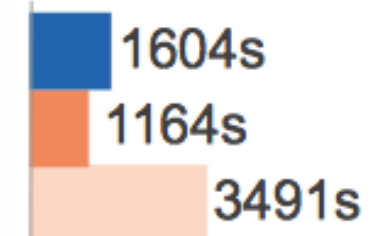


b M3 v.s. Spark (4 & 8 Instances)

L-BFGS



K-Means



- M3
- 8x Spark
- 4x Spark



Scalable Machine Learning & Graph Mining via Virtual Memory

Memory Mapping based computation is a minimalist approach that forgoes sophisticated data structures, explicit memory management, and optimization techniques but still achieve high speed and scalability, by leveraging the fundamental memory mapping (MMap) capability found on operating systems.

Broader Impacts of this Project

Large datasets in terabytes or petabytes are increasingly common, calling for new kinds of scalable machine learning approaches. While state-of-the-art techniques often use complex designs, specialized methods to store and work with large datasets, this project proposes a minimalist approach that forgoes such complexities, by leveraging the fundamental virtual memory capability found on all modern operating systems, to load into the virtual memory space the large datasets



Faster I/O Operations



Less Overhead