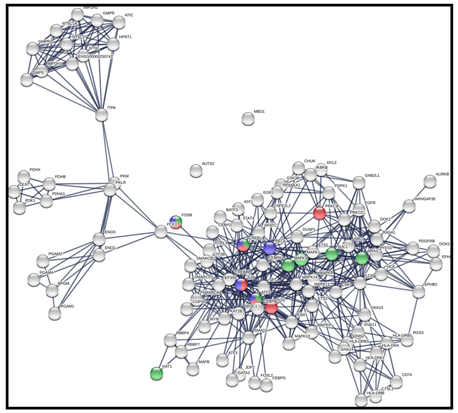
Optimizing Dynamic Graph Processing with the Locality-First Strategy

Helen Xu
Lawrence Berkeley National Laboratory
ACDA Workshop

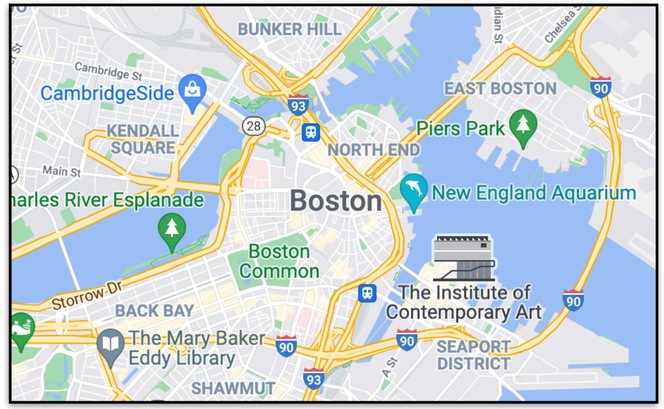
Real-World Graphs Are Sparse

Sparse graphs, which have many fewer edges than the total possible number of edges, underlie most real-world applications.

Social networks



Computational biology



Road networks

…and others!

Sparsity **disrupts locality** due to the presence of many zeroes in the data.

Real-World Graphs Are Also Dynamic

Furthermore, many real-world sparse graphs are **dynamic**: they **change over time**.

Updates

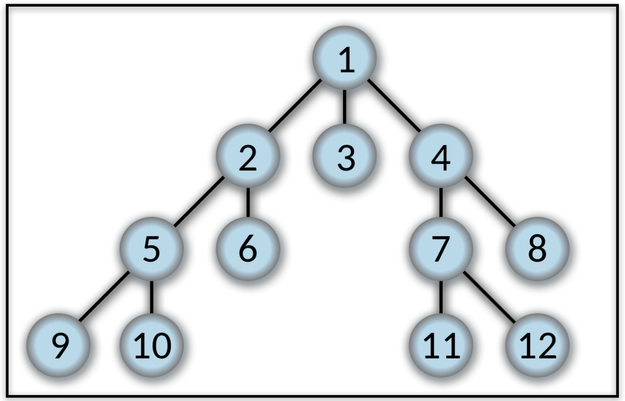
Systems for processing dynamic graphs support **updates** (e.g. edge insertions and deletions) and **queries** (algorithms run on the graph).

Dynamic graphs disrupt locality because of the inherent **tradeoff between colocating and updating** data (e.g., in CSR).

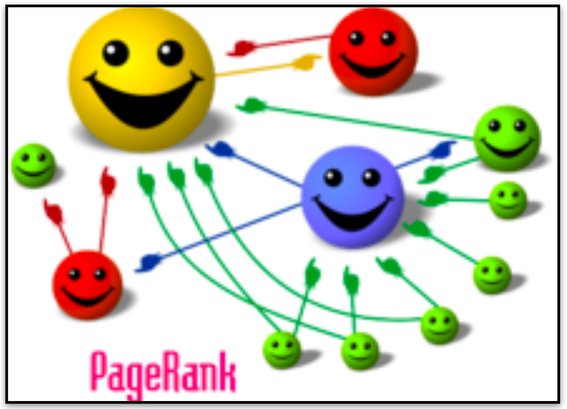
Multicore Optimization Enables Fast Graph Queries and Updates

Despite these challenges to locality, high-performance dynamic-graph-processing systems such as Aspen [DhulipalaShBl19] have taken huge steps towards **efficient queries and updates**.

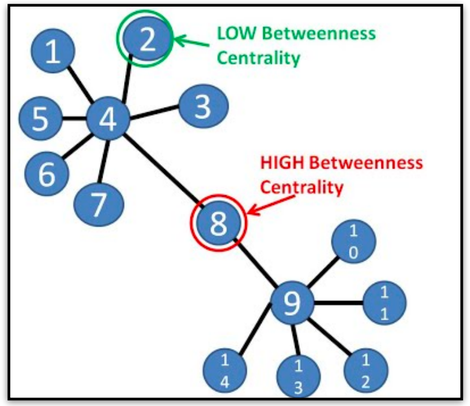
On 48 cores, Aspen runs the following queries on **Twitter** (2.4B edges):



Breadth-first search
0.32s



PageRank
24.03s



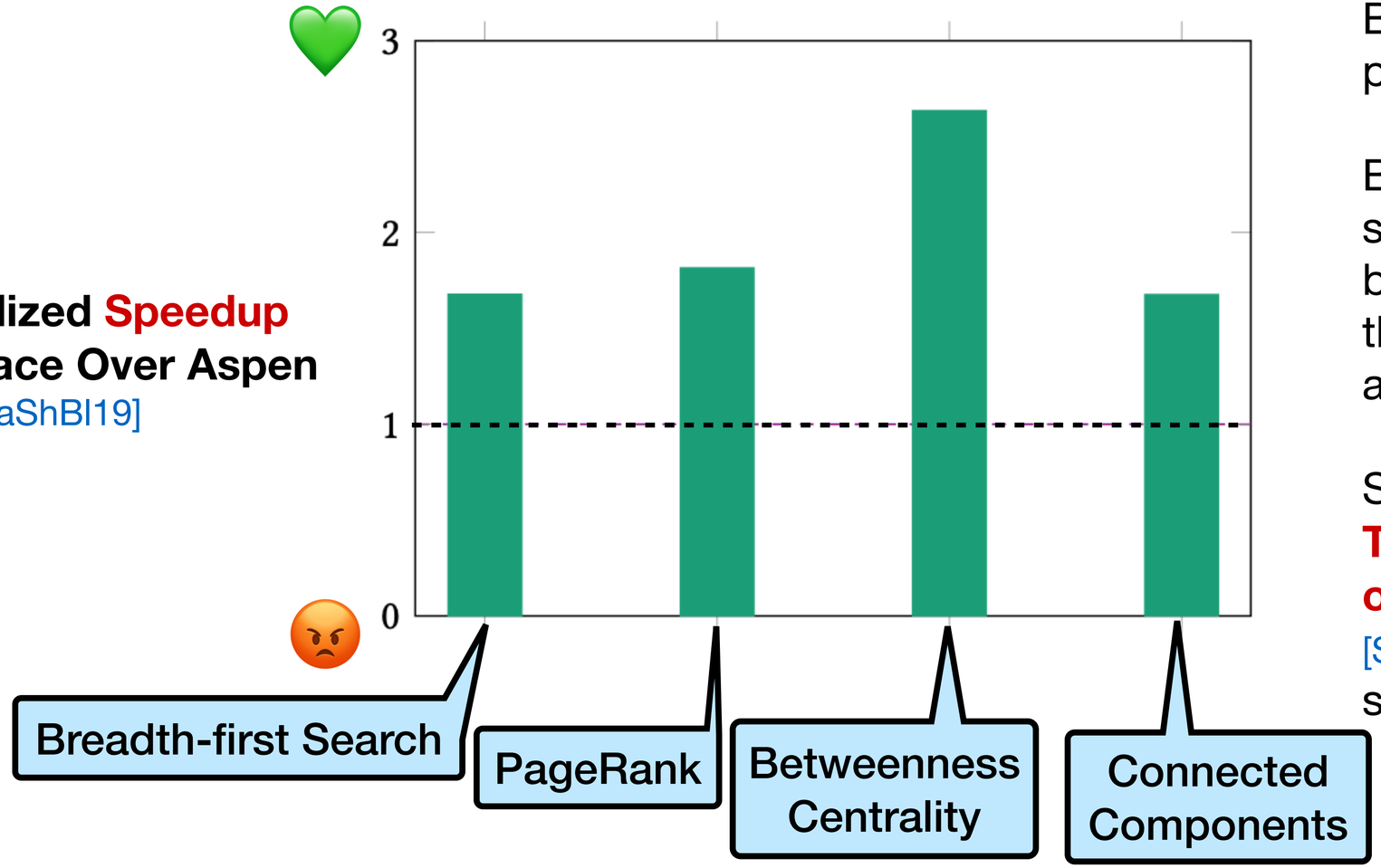
Betweenness centrality
4.72s

Times are **human-measurable** even with parallelism, demonstrating the importance of **efficient processing**

Query Speed in Dynamic Graph Systems

Terrace [PandeyWhXuBu21], a dynamic graph processing system, optimizes further with a **"locality-first design"** that takes advantage of graph structure.

Normalized Speedup of Terrace Over Aspen [DhulipalaShBl19]



Both systems support parallelization.

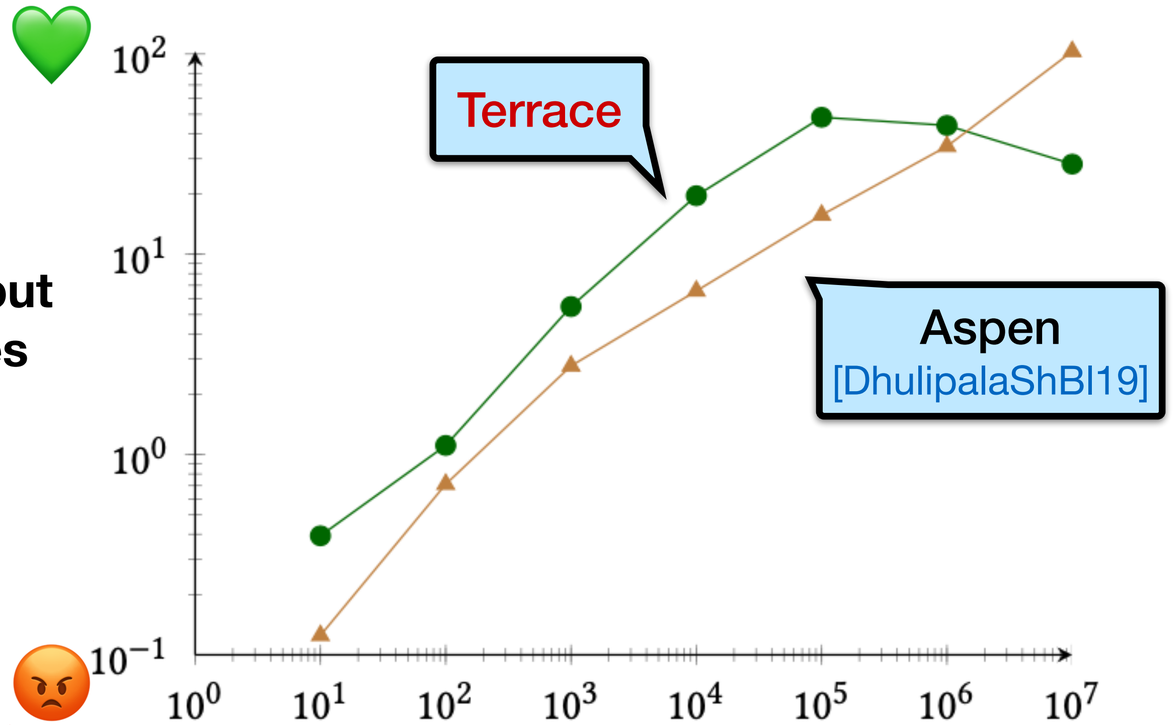
Both systems run the same algorithms by implementing the Ligra [ShunBl13] abstraction.

Surprisingly, in some cases, **Terrace achieves speedup on queries over Ligra** [ShunBl13], a static graph system.

[PandeyWhXuBu21] Pandey, Wheatman, **Xu**, Buluç. "Terrace: A Hierarchical Graph Container for Skewed Dynamic Graphs." SIGMOD '21.

Updatability of Dynamic Graph Systems

Terrace achieves the **best of both worlds** in query and update performance by taking advantage of locality.



Insertion Throughput (in millions of edges per second)

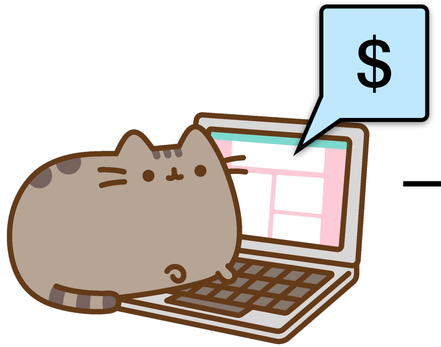
Edges were generated using an rMAT distribution [ChakrabatiZhFa04] and added in batches using the provided API.

Terrace achieves up to 48M inserts per second and up to 9M deletes per second. Future work includes optimizing batch deletions.

Dynamic Graph Processing and the Locality-First Strategy

Problem: Dynamic graph processing

Understand locality in dynamic graph processing



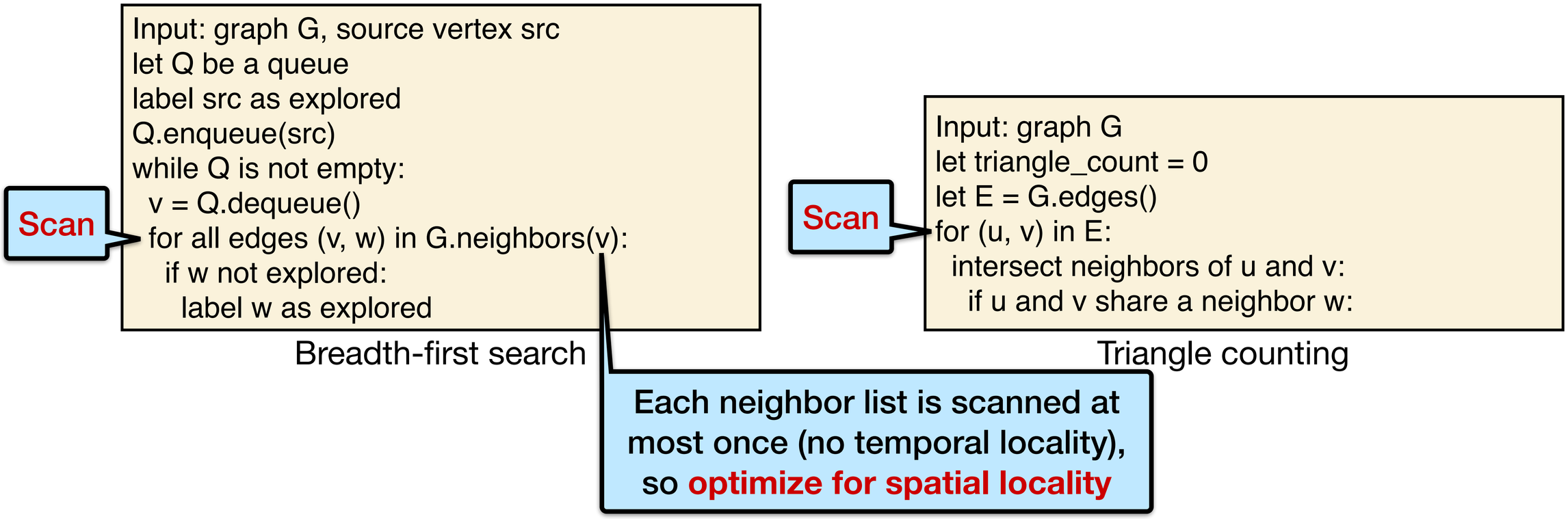
Exploit locality via data structure design for graphs

Add **parallelism** into data structures

Understanding Locality in Graph Queries

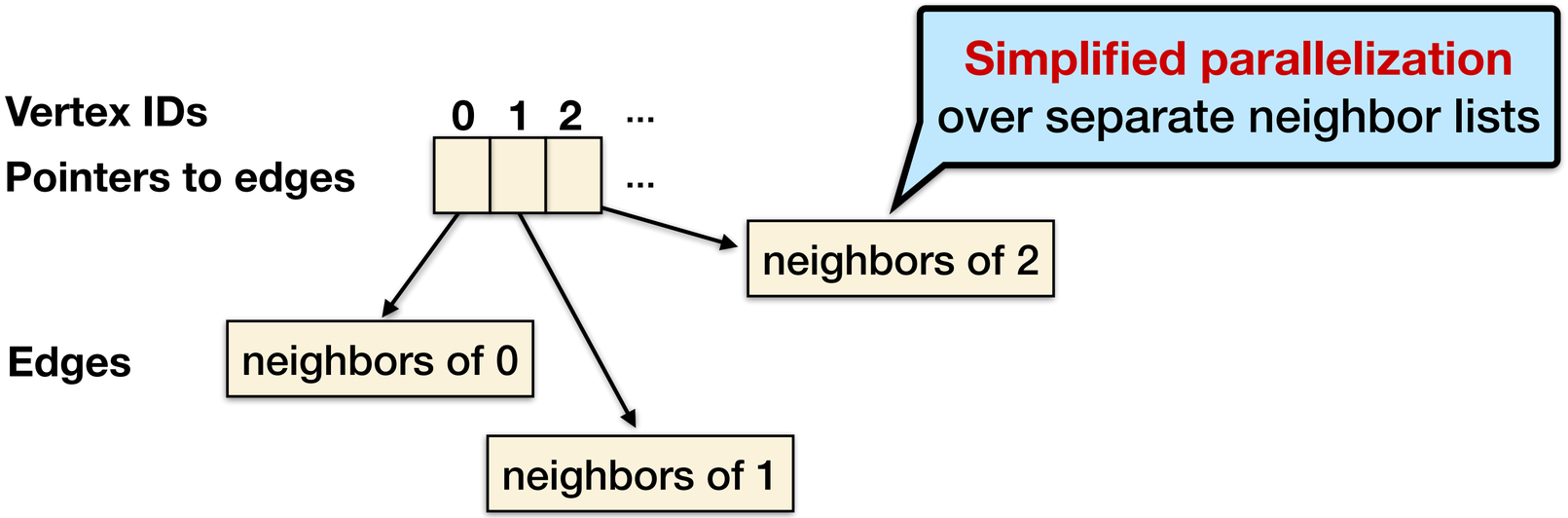
Systems for processing dynamic graphs must support fast graph queries.

Vertex scans, or the **processing of a vertex's incident edges**, are a crucial step in graph queries [ShunBl13].



Most Graph Systems Separate Neighbor Lists for Parallelization

Existing dynamic graph systems optimize for parallelism first with **separate per-vertex data structures** e.g. trees [DhulipalaBlSh19], adjacency lists [EdigerMcRiBa12], and others [KyrolaBlGu12, BusatoGrBoBa18, GreenBa16].

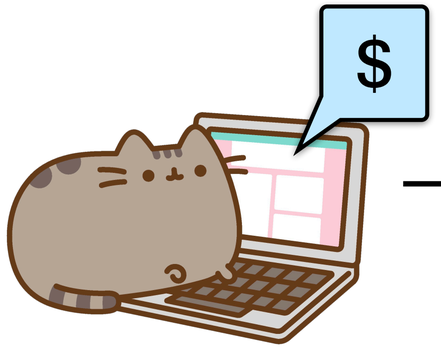


Weakness: Separating the data structures **disrupts spatial locality**.

Dynamic Graph Processing and the Locality-First Strategy

Problem: Dynamic graph processing

Understand locality in dynamic graph processing

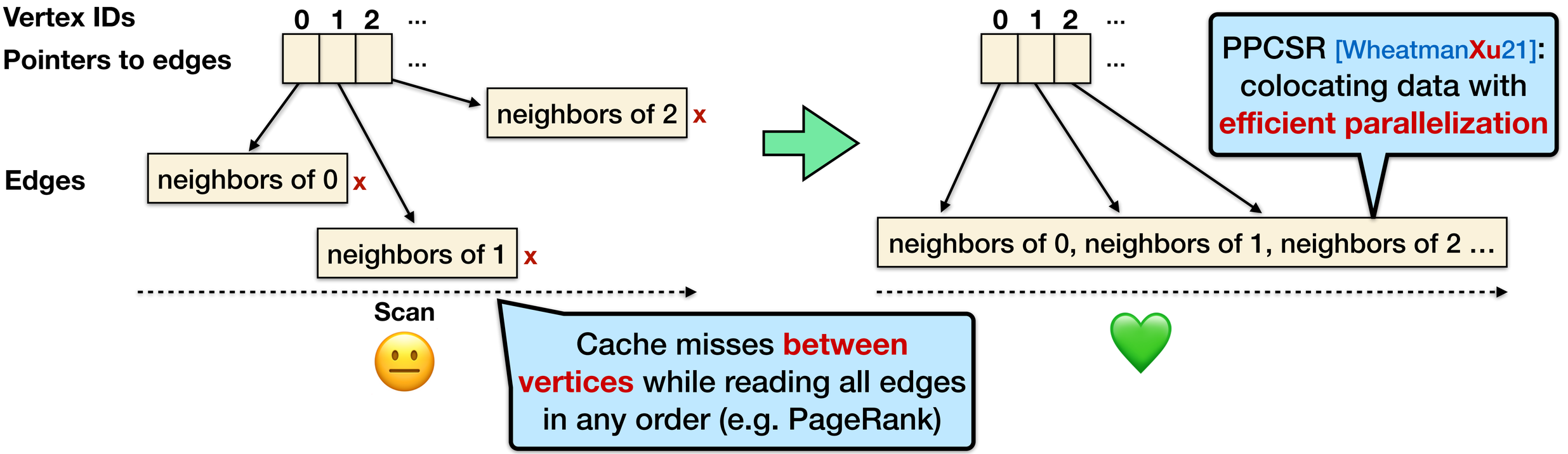


Exploit locality via data structure design for graphs

Add **parallelism** into data structures

Enhancing Spatial Locality by Colocating Neighbor Lists

Idea: Colocate neighbor lists in the same data structure, which **avoids cache misses** when traversing edges in order.



Question: Do these misses actually affect performance, or are they a low-order term?

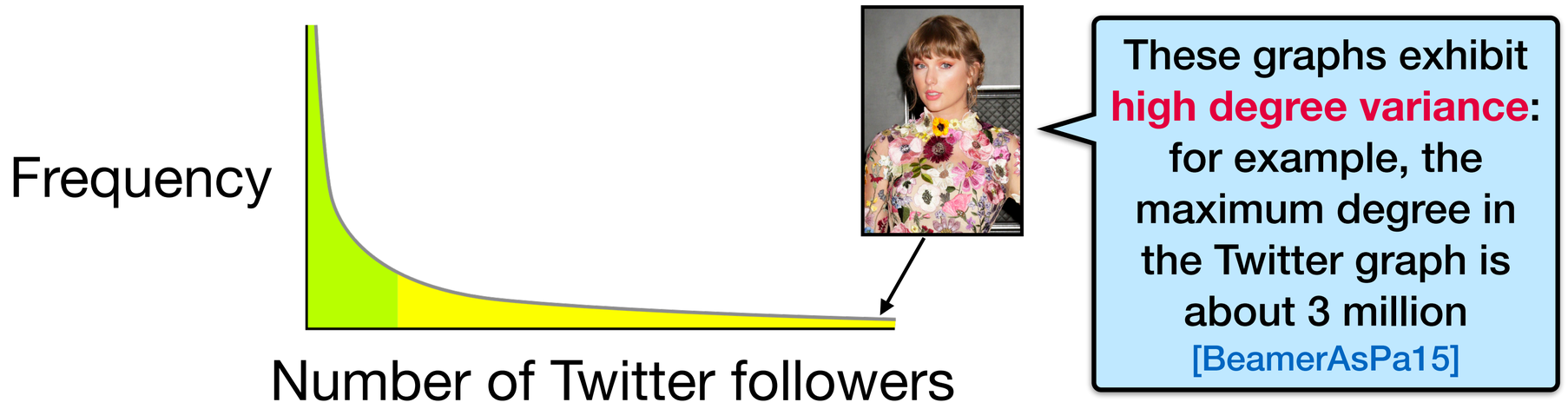
[WheatmanXu21] Wheatman and **Xu**. “A Parallel Packed Memory Array to Store Dynamic Graphs.” ALENEX ’21.

Dynamic Graphs Are Often Skewed

Real-world dynamic graphs, e.g. social network graphs, often follow a **skewed** (e.g. power-law) distribution with a **few high-degree vertices and many low-degree vertices** [BarabasiAl99].

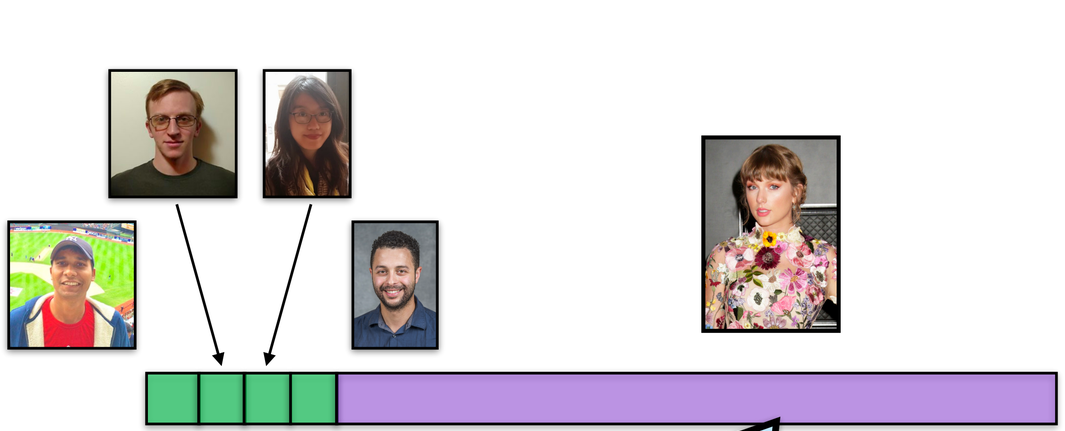
Example power law:

Graph	% < 10 neighbors	% < 1000 neighbors
Twitter	64.56	99.51

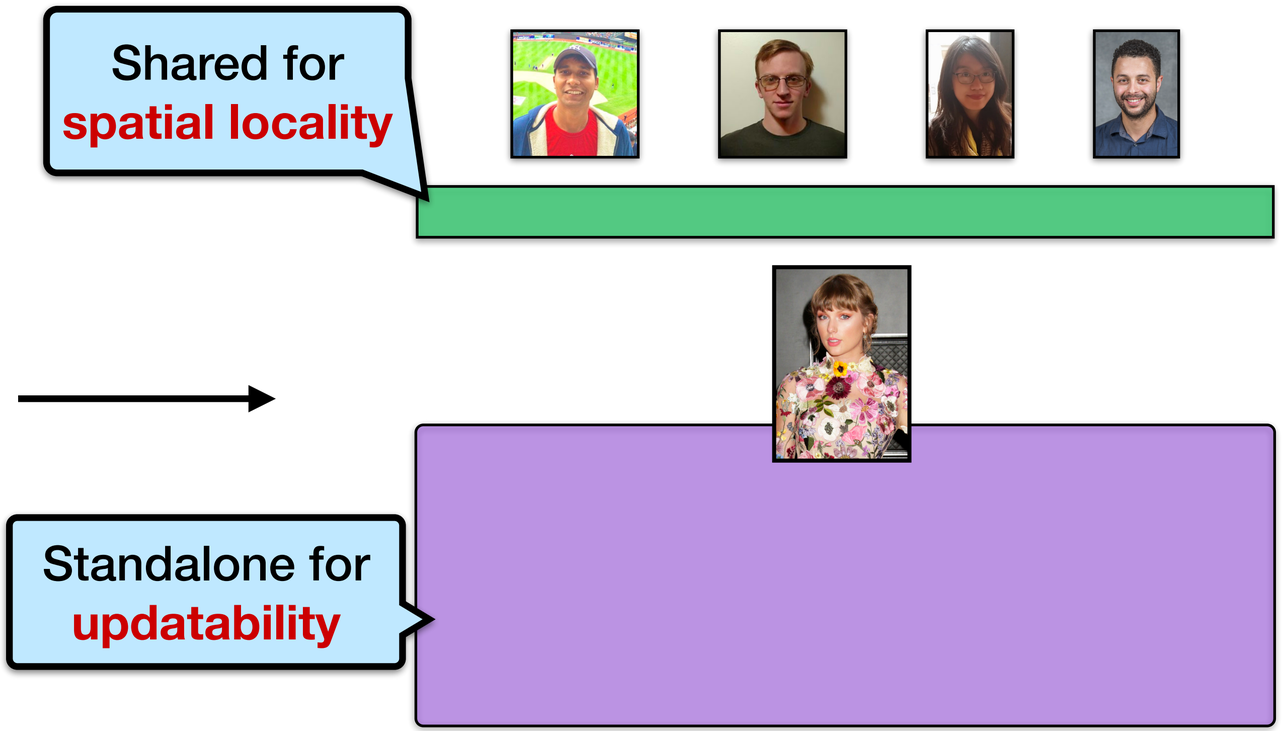


Insight: Locality-First Skew-Aware Design

Next step: **refine the solution** with a **hierarchical design** that takes advantage of skewness while maintaining locality as much as possible.

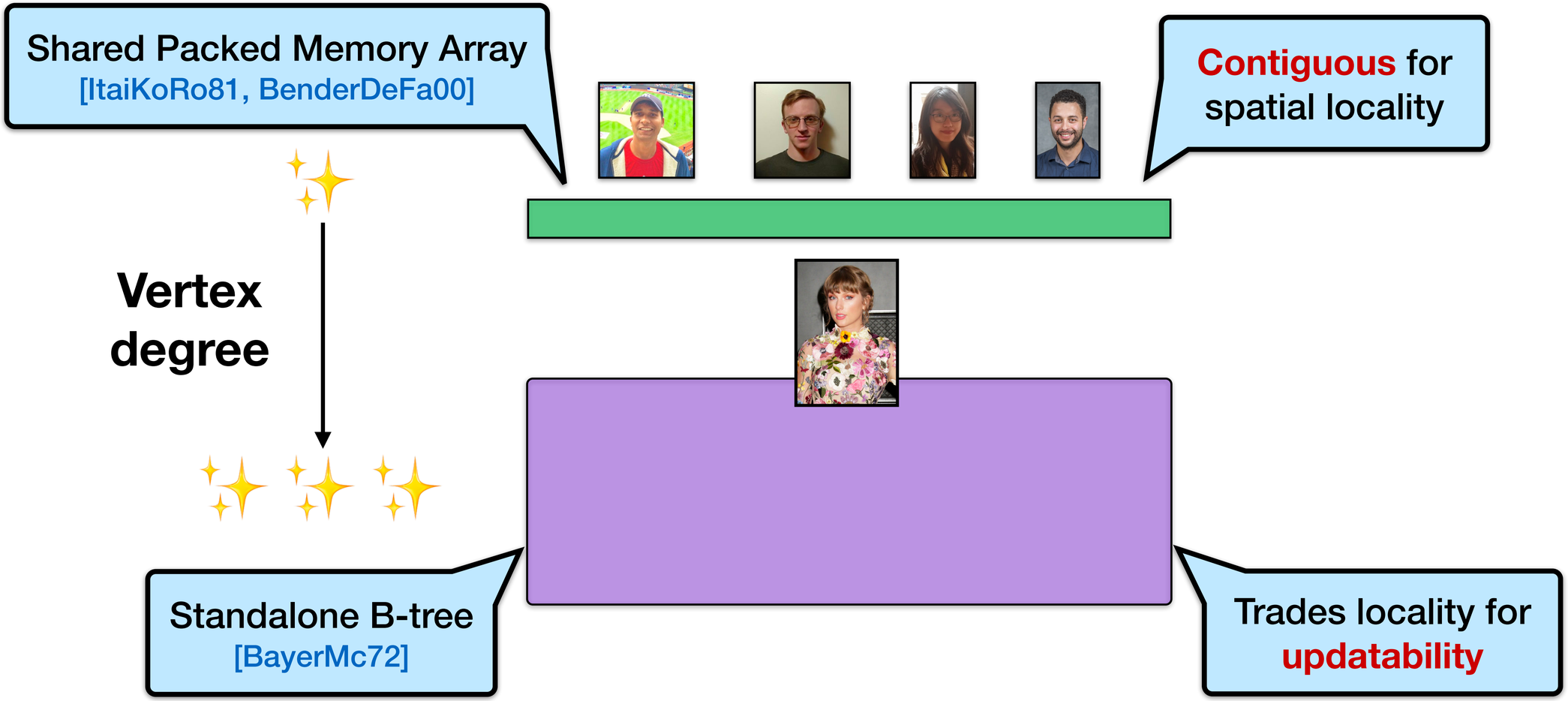


Problem: High-degree vertices **slow down updates** for all vertices in the shared data structure



Implementing the Hierarchical Skew-Aware Design

Terrace implements the locality-first hierarchical design with **cache-friendly data structures.**



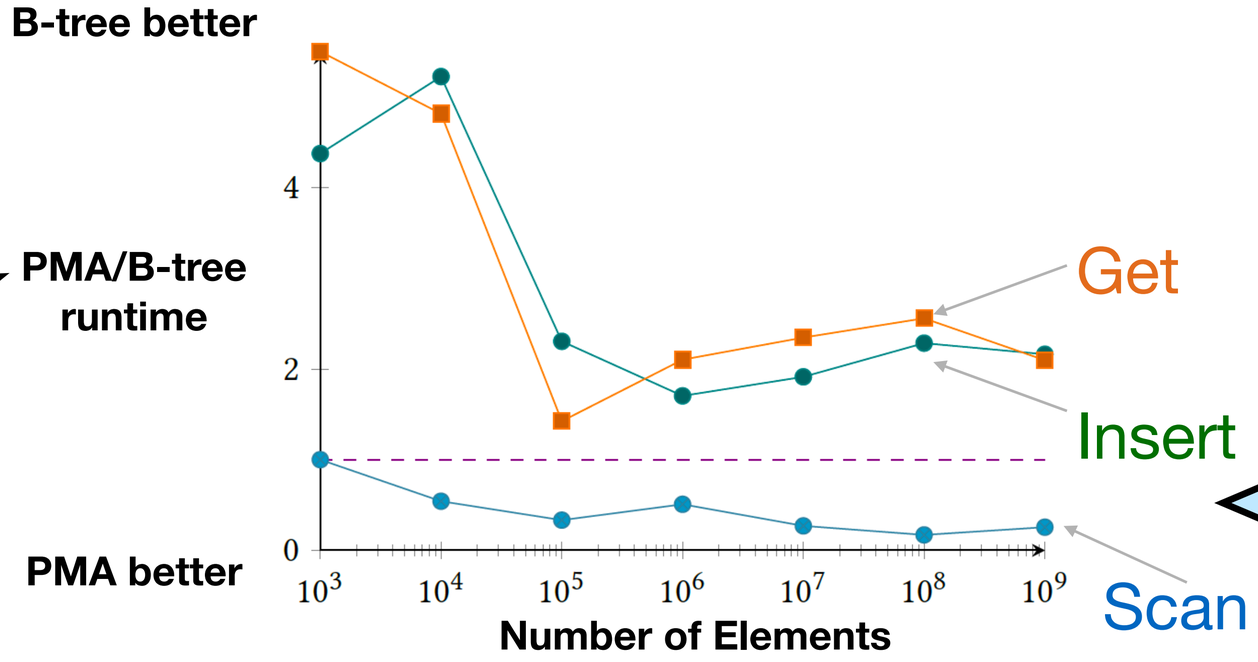
Selecting Data Structures for Dynamic Graphs

In theory, B-trees [BayerMc72] **asymptotically dominate** Packed Memory Arrays (PMA) [ItaiKoRo81, BenderDeFa00] in the classical external-memory model [AggarwalVi88].

Given a cache block size B and input size N, B-trees and PMAs take $\Theta(N/B)$ **block transfers** to scan.

B-tree inserts take $O(\log_B(N))$ transfers, while PMA inserts take $O(\log^2(N))$.

The theory does not capture **sequential vs random access**



Problem: **Neither data structure clearly wins** for dynamic graphs because graphs require fast updates and scans

Solution: **use both**, depending on degree

Exploiting Skewness for Cache-Friendliness

The **locality-first design** in Terrace **reduces cache misses** during graph queries.

Additional optimization: store some edges **in-place for extra spatial locality**

Static

Dynamic

On the LiveJournal graph

Query	Ligra [ShunBl13]	Aspen [DhulipalaShBl19]	Terrace [PandeyWhXuBu21]
Breadth-first Search	3.5M	6.3M	1.1M
PageRank	174M	197M	128M

Cache-friendliness translates into **graph query performance**

Terrace: Applying the Locality-First Strategy to Dynamic Graph Processing

In practice, Terrace is about **2x faster on graph query algorithms** than Aspen while **maintaining similar updatability**.

Terrace's **cache-friendly design** demonstrates the impact of **the locality-first strategy** in graph processing.

Problem: Dynamic graph processing

Understand locality: opportunities for spatial locality due to **skewness**

Exploit spatial locality with a **cache-friendly skew-aware** data structure

Implementation of Terrace, a **parallel** dynamic-graph-processing system based on the **skew-aware design** [PandeyWhXuBu21]

https://github.com/PASSIONLab/terrace