Seraph: Towards Scalable and Efficient Fully-external Graph Computation via On-demand Processing

Tsun-Yu Yang, Yizou Chen, Yuhong Liang, and Ming-Chang Yang

The Chinese University of Hong Kong

Outline

- Introduction
- Background and Motivation
- Seraph
  - Hybrid Format
  - Vertex Passing
  - Selective Pre-computation
- Evaluation

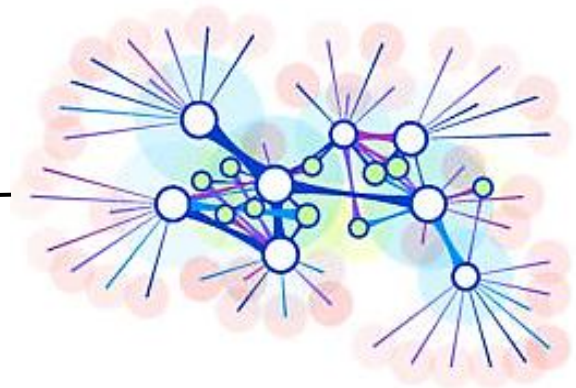
Graph and Graph Systems

- Graph are powerful data structures to describe the relationship.
  - Store the entities as *vertices* and the connection between entities as *edges*.
  - Widely used in different fields, such as networking, social media, and bioinformatics.

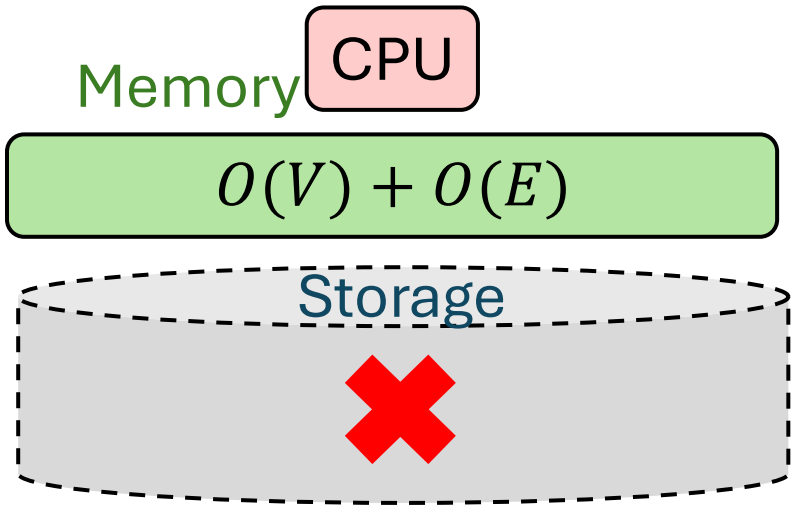
- Against this background, *graph systems* are proposed to optimize the execution of graph algorithms.

- Based on how we are using the memory and storage, the graph systems can be divided into three different types.
  - Shared-memory
  - Semi-external
  - Fully-external

Architectures of Graph Systems

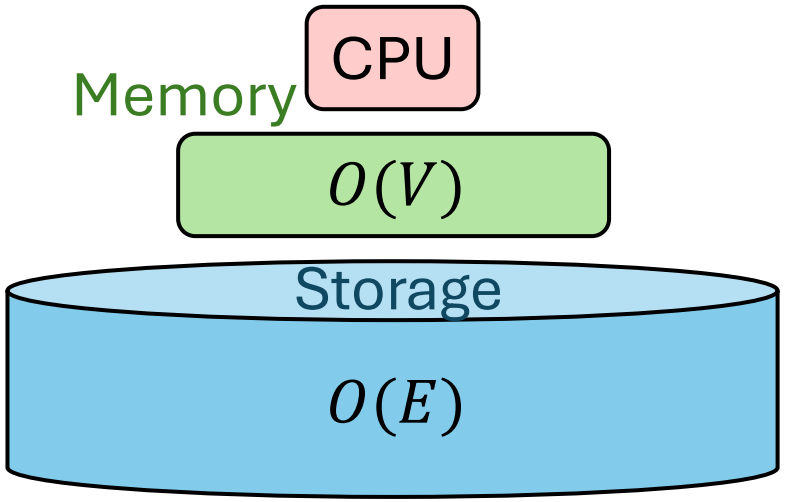


$O(V)$: size of Vertex Data
$O(E)$: size of Edge Data



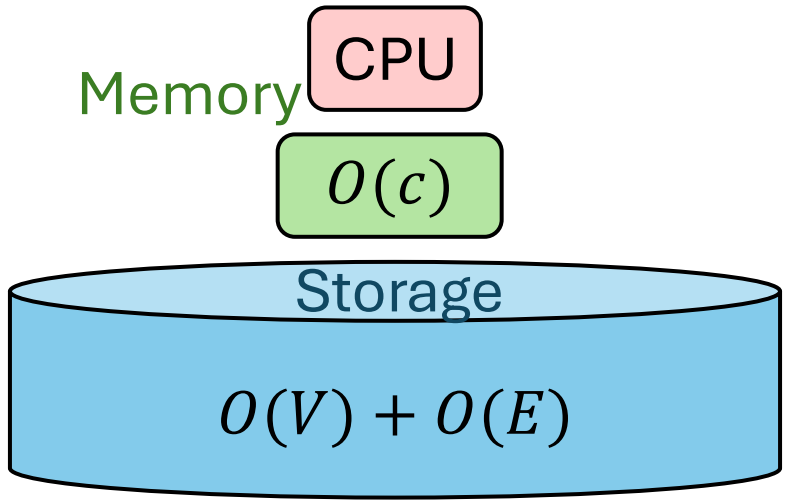
Shared-memory

(e.g., Ligra [PPoPP'13], Galois [SOSP'13])



Semi-external

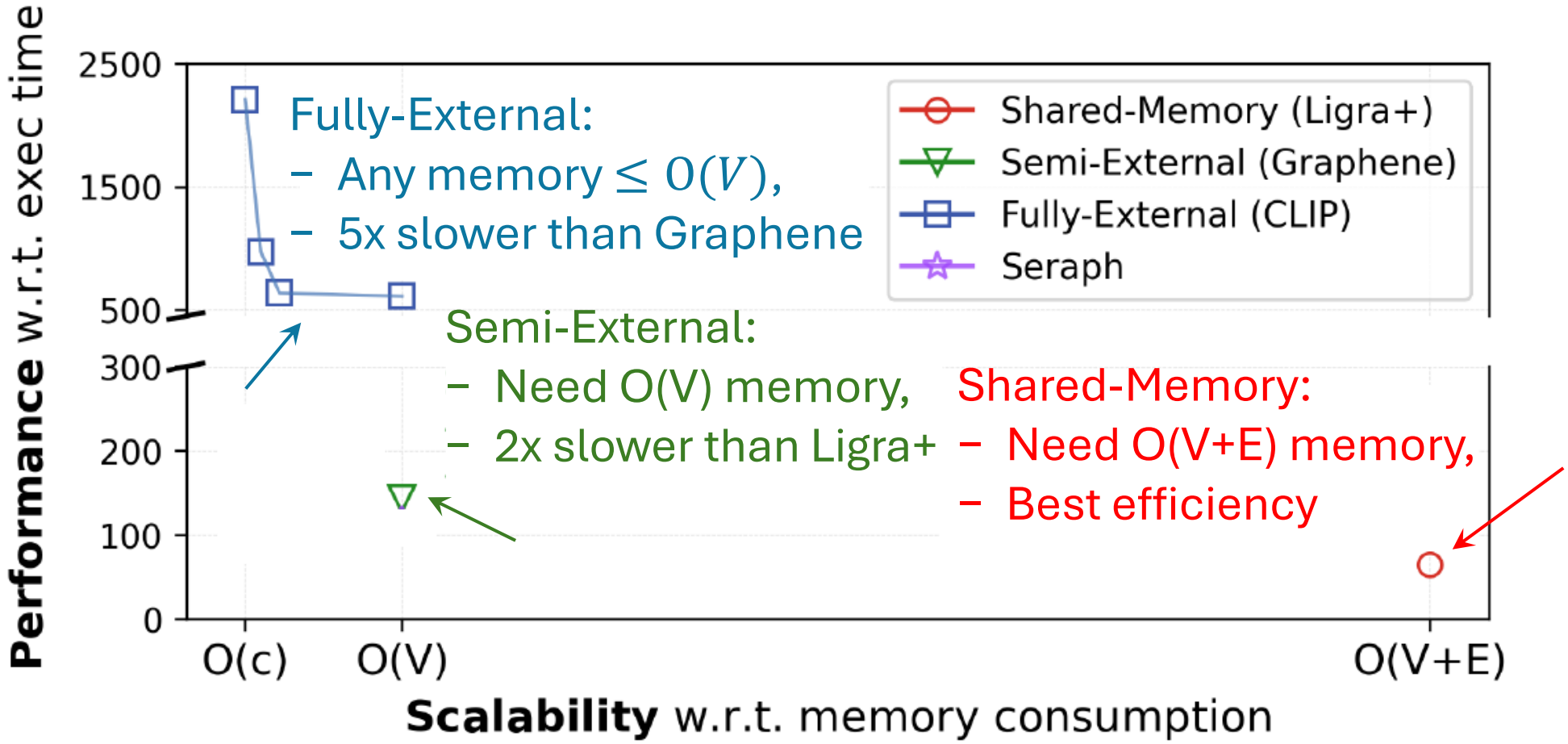
(e.g., FlashGraph [FAST'15], Graphene [FAST'17])



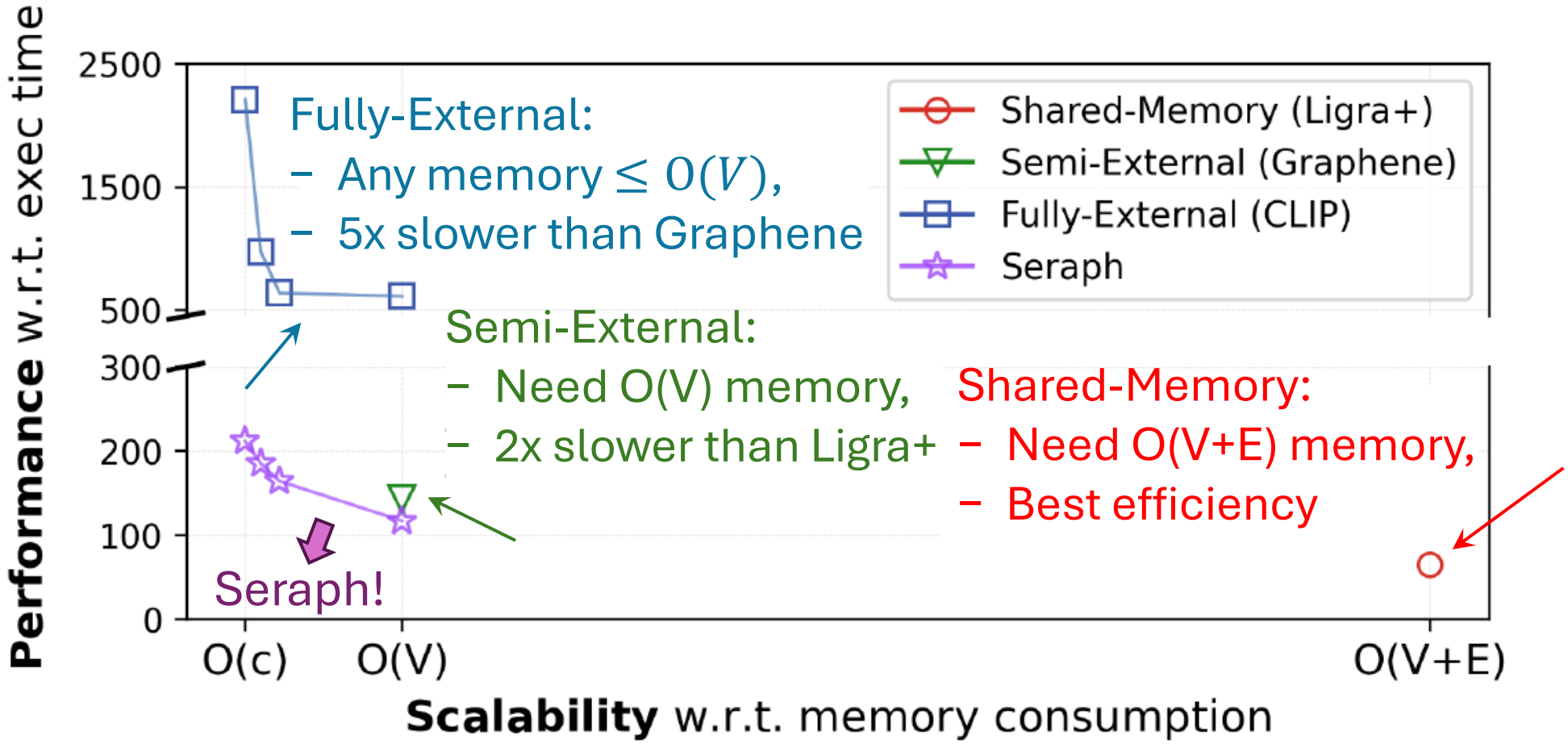
Fully-external

(e.g., GraphChi [OSDI'12], GridGraph [ATC'15])

Investigations of Graph Systems



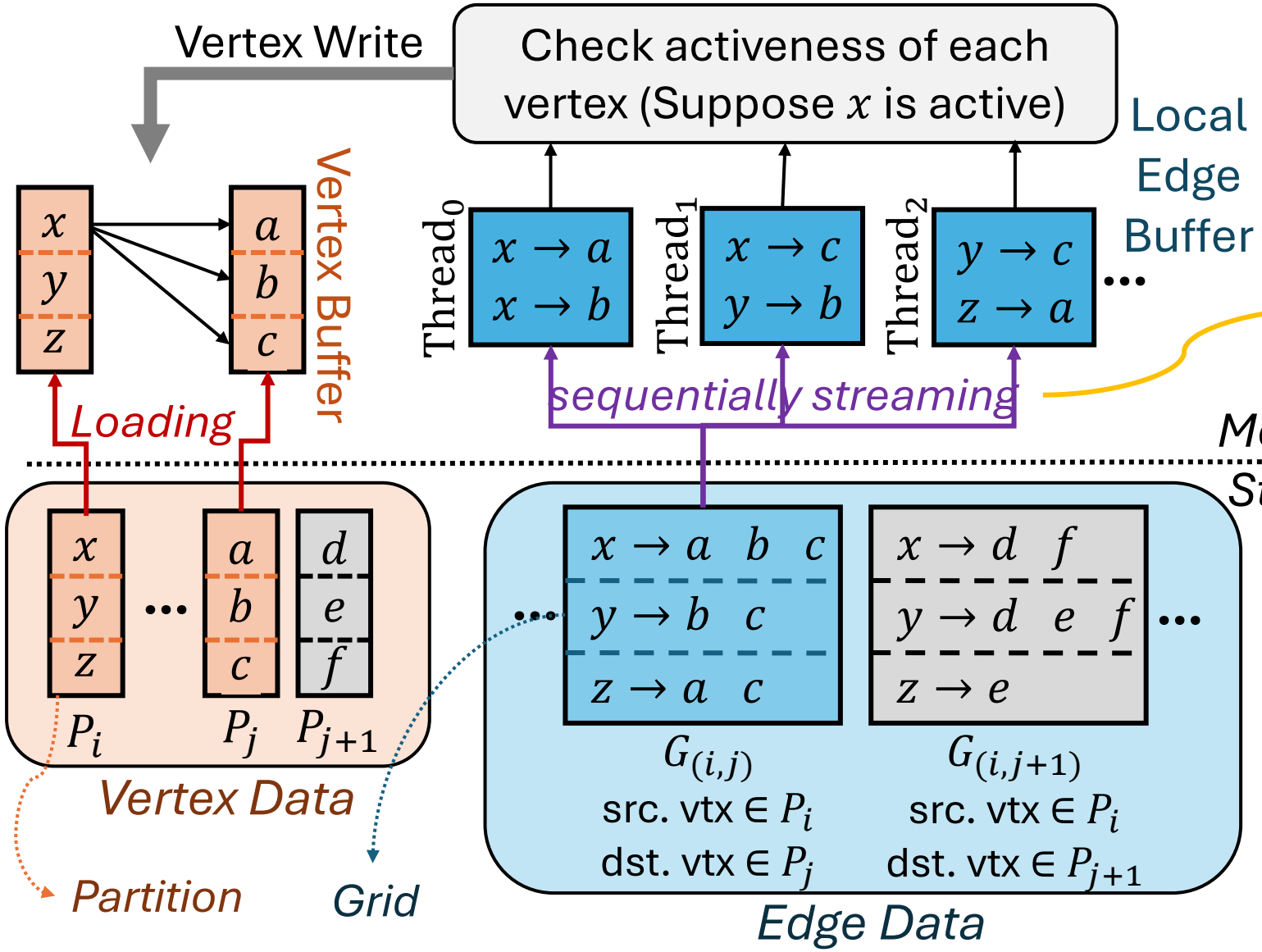
Investigations of Graph Systems



Outline

- Introduction
- Background and Motivation
- Seraph
  - Hybrid Format
  - Vertex Passing
  - Selective Pre-computation
- Evaluation

Existing Fully-external Graph Systems



Streaming-based Processing:

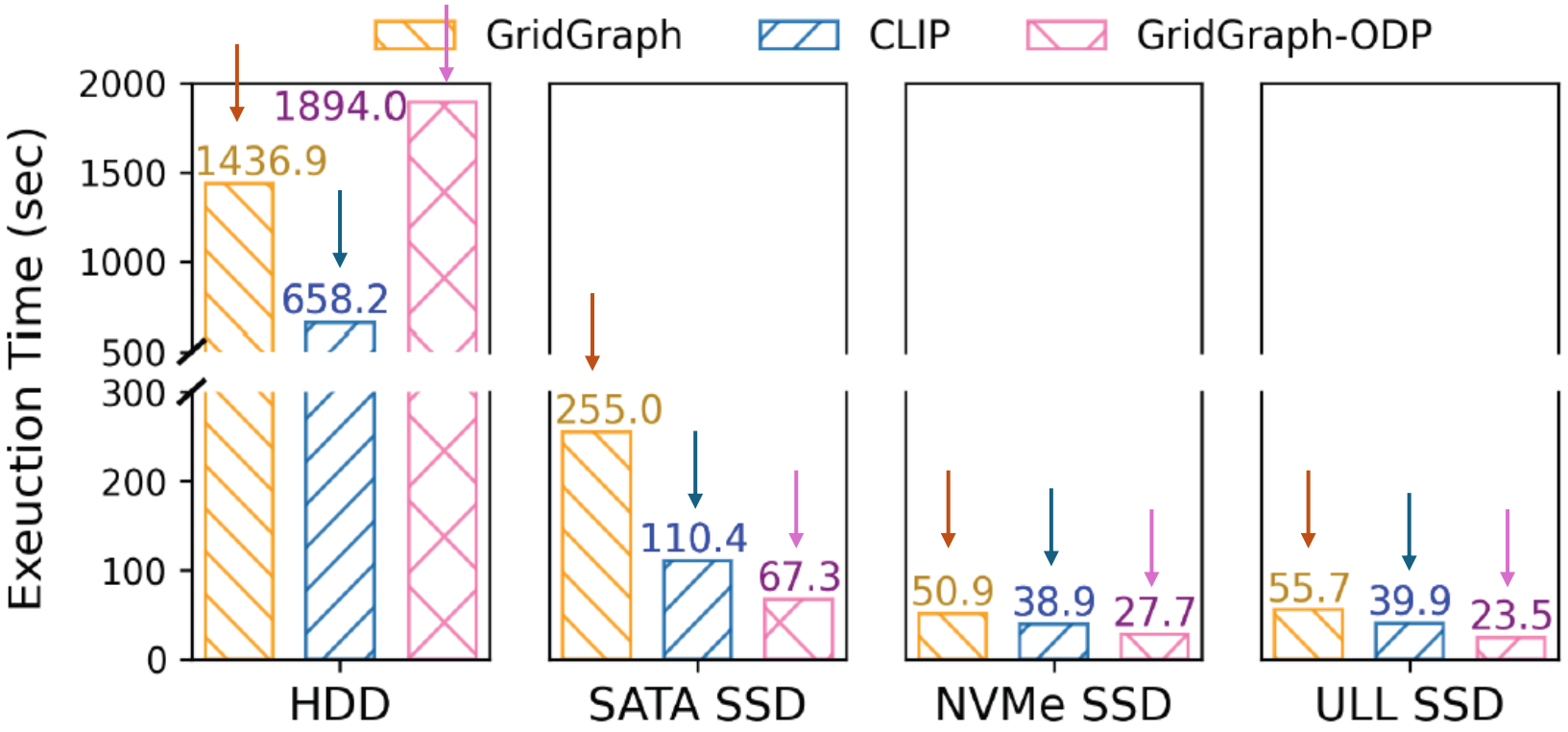
- ☺ Sequentially streaming to enjoy high I/O access speed.
- ☹☹☹ Often load excessive-and-irrelevant data.

Grid-based Format:

- Divide vertices into disjoint *partitions*.
- Based on the partitions, divide edges into disjoint *grids*.
- 2 partitions + 1 grid = *subgraph*

Motivation

- Study GridGraph, CLIP, and GridGraph-ODP.
- Evaluating with four types of storage drives.
  - HDD, SATA SSD, NVMe SSD, and ULL SSD



GridGraph@ATC'15

A baseline system which adopts streaming-based processing.

CLIP@ATC'17

An advanced version of streaming-based processing

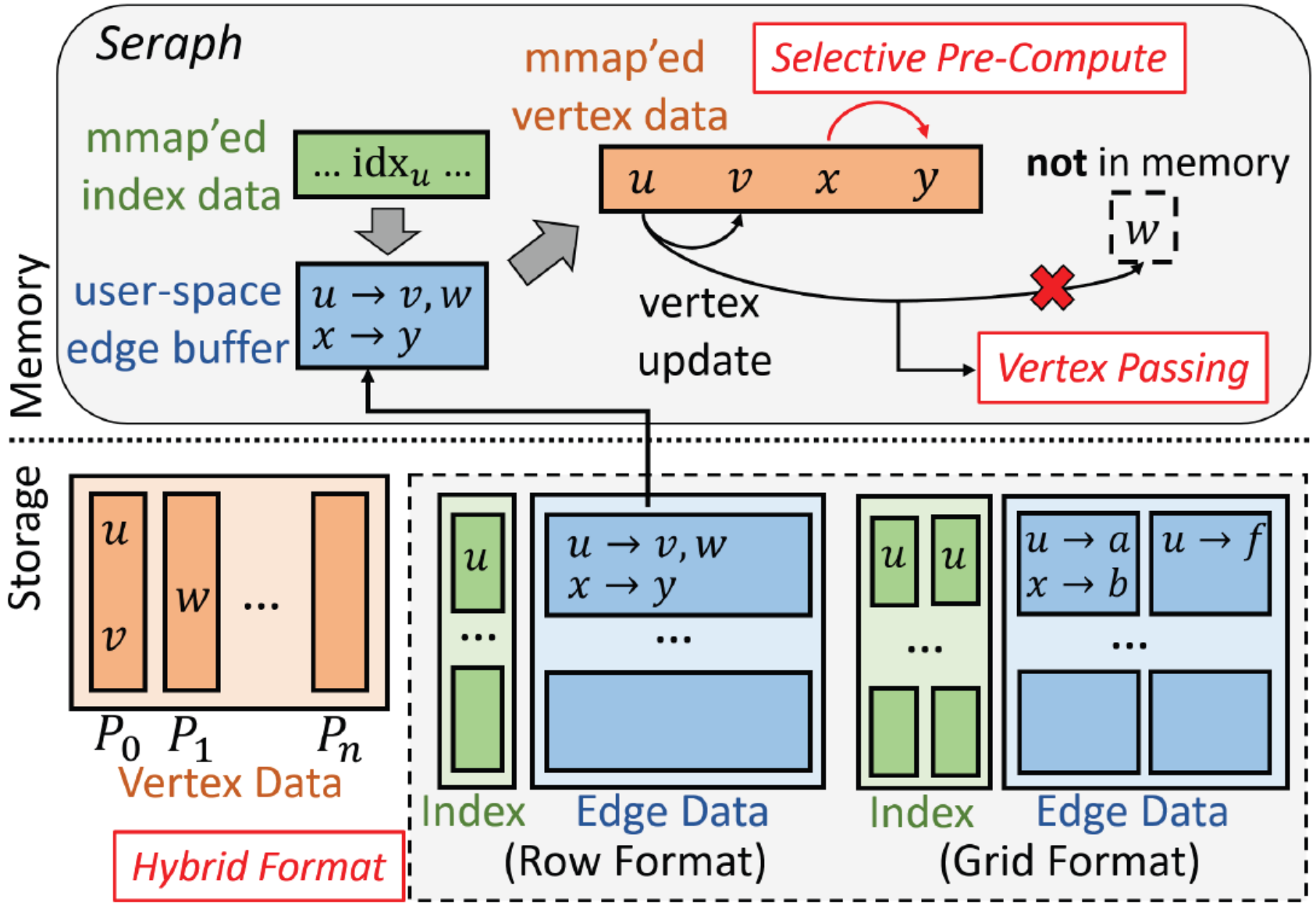
GridGraph-ODP

A baseline system which adopts on-demand processing.

Outline

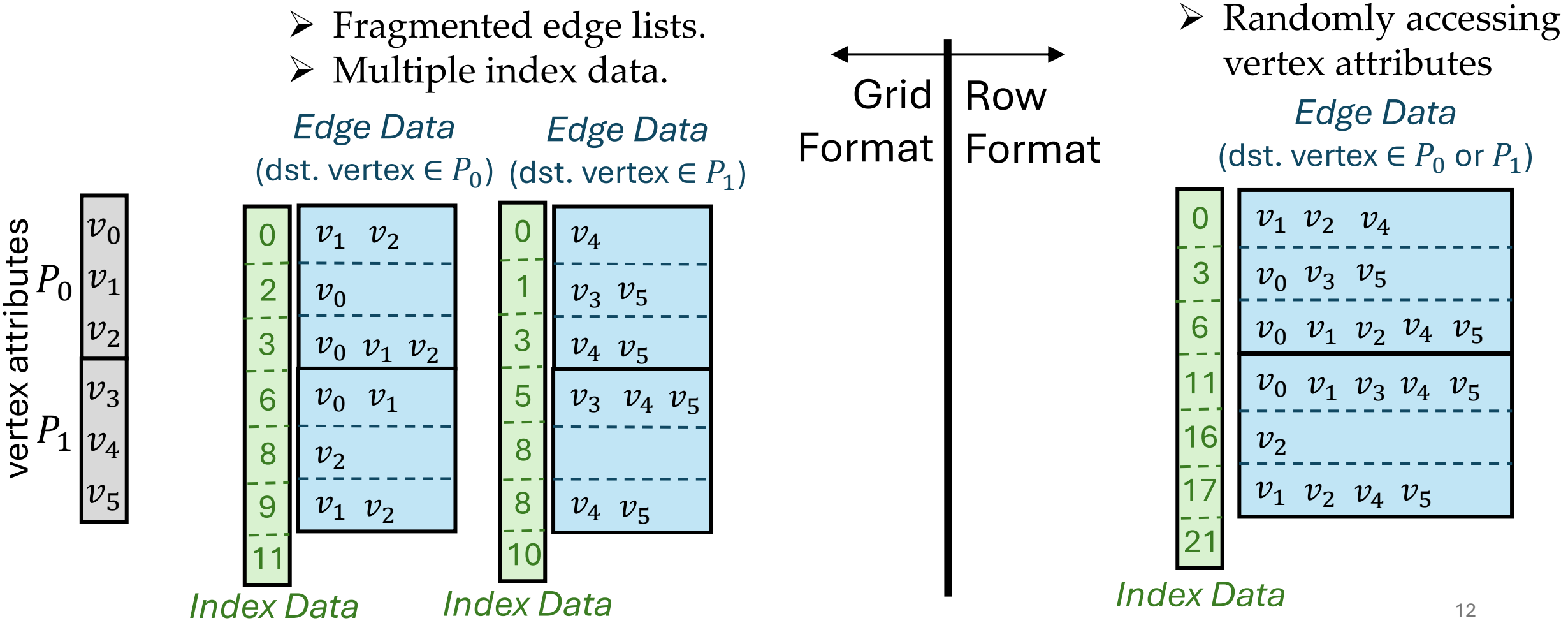
- Introduction
- Background and Motivation
- Seraph
  - Hybrid Format
  - Vertex Passing
  - Selective Pre-computation
- Evaluation

Overview of Seraph



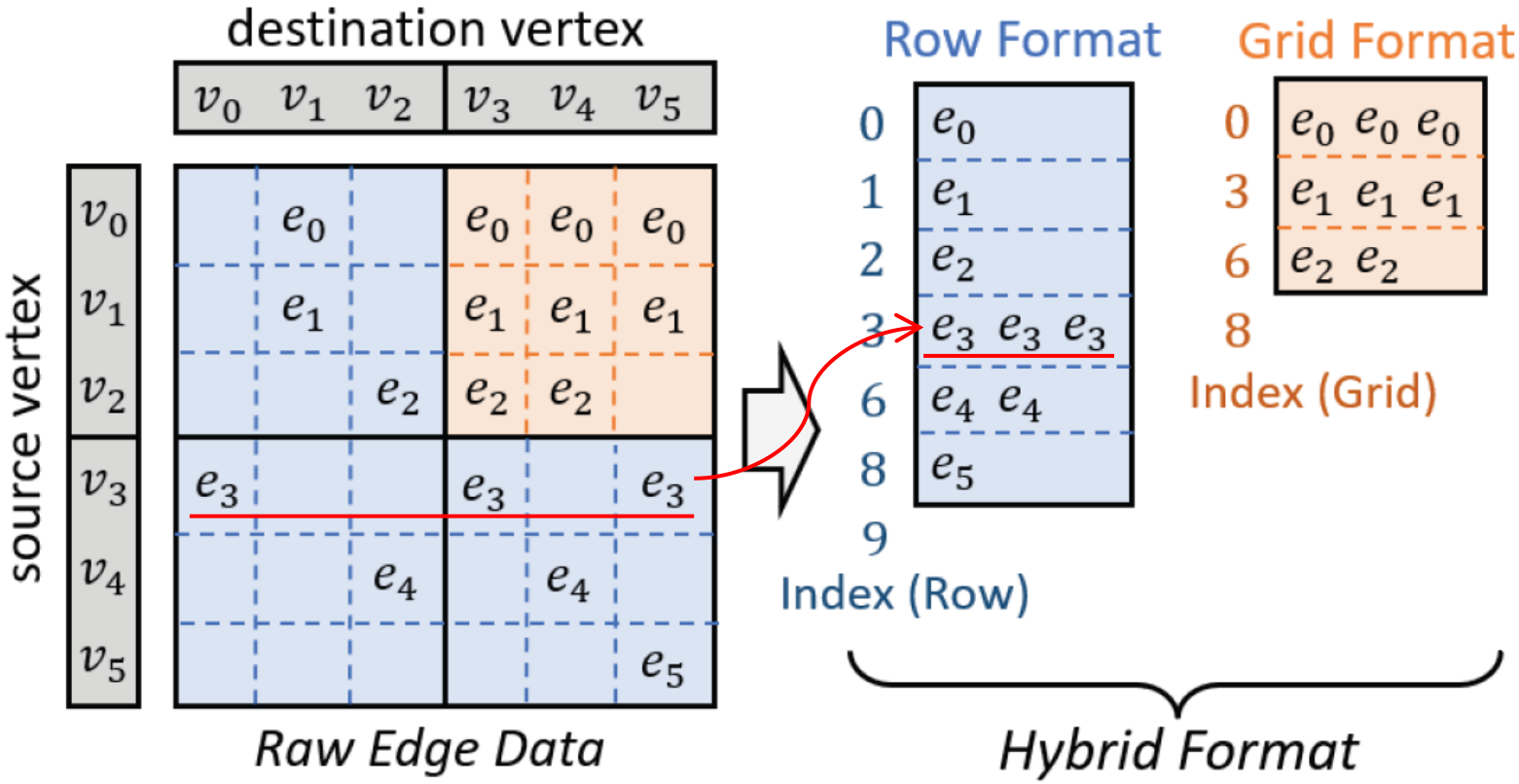
On-demand Processing on Traditional Formats

- There is a trade-off to consider when applying on-demand processing on the traditional formats.



Hybrid Format

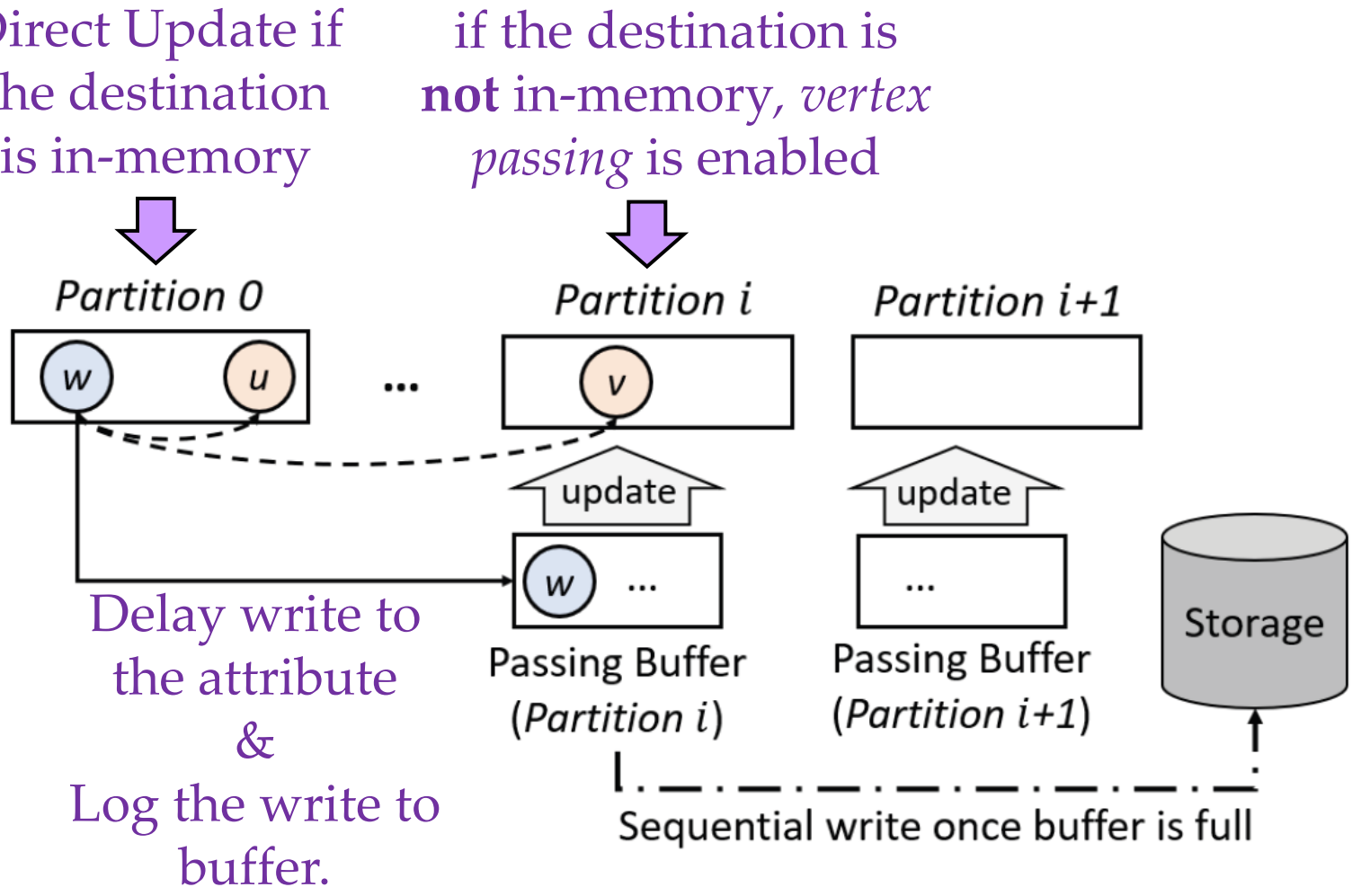
- Hybrid format ➔ Take advantage of both Row & Grid.

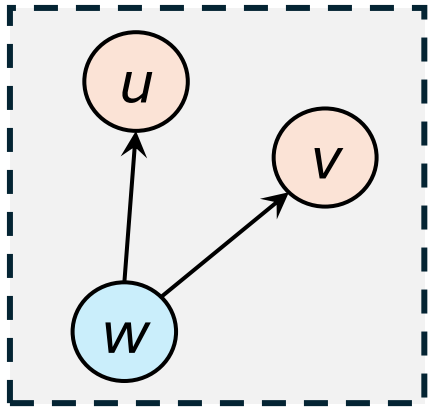


Thus, the execution on **Row Format** still leads to bad locality of access.

Vertex Passing

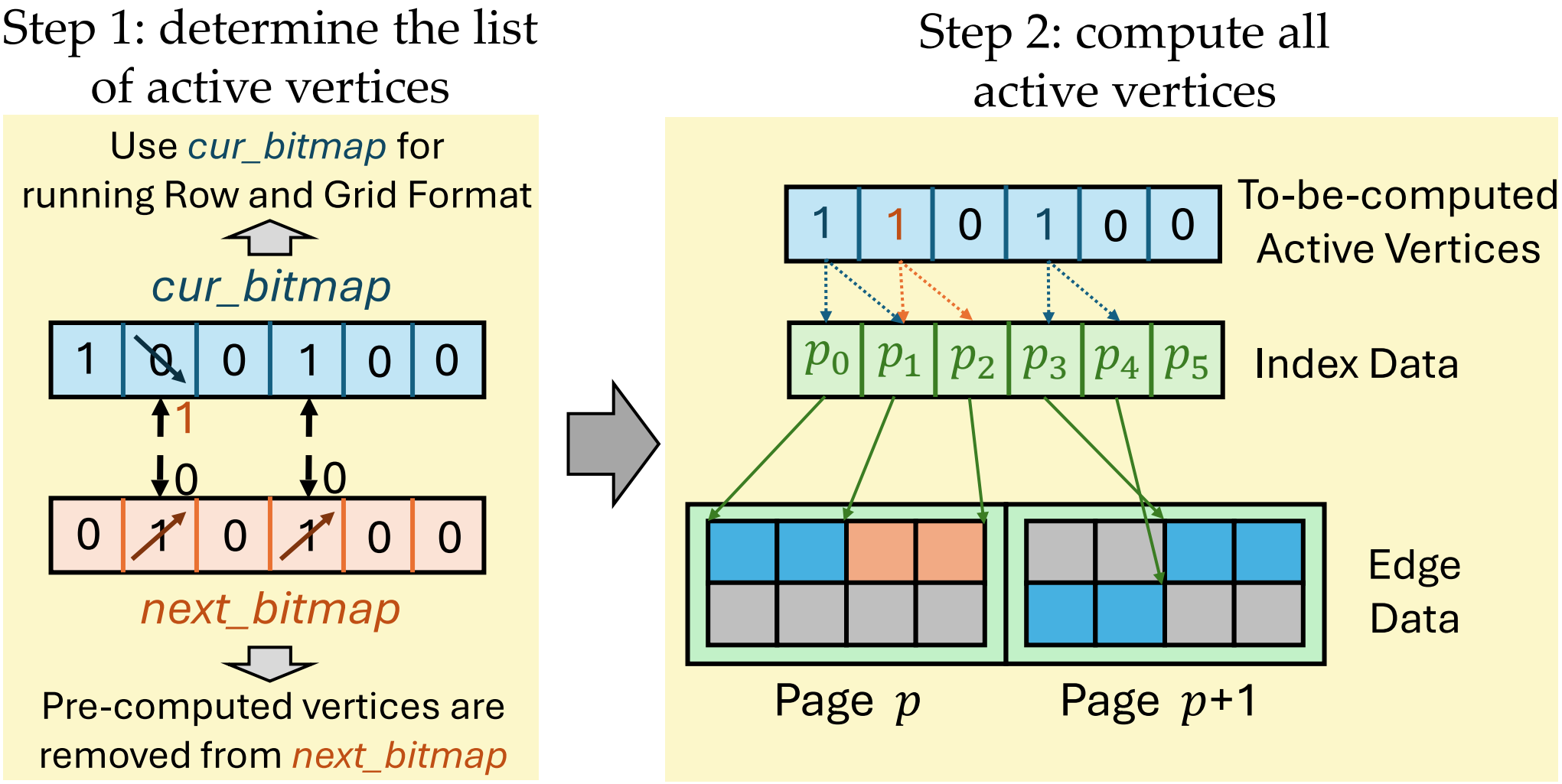
- The main concept of Vertex Passing is to delay vertex writes as logs and then create good locality of attribute access.





Selective Pre-Computation

- Given that a common I/O block is typically larger than the edge list, we introduce selective pre-computation to increase the utilization of loaded data.



Outline

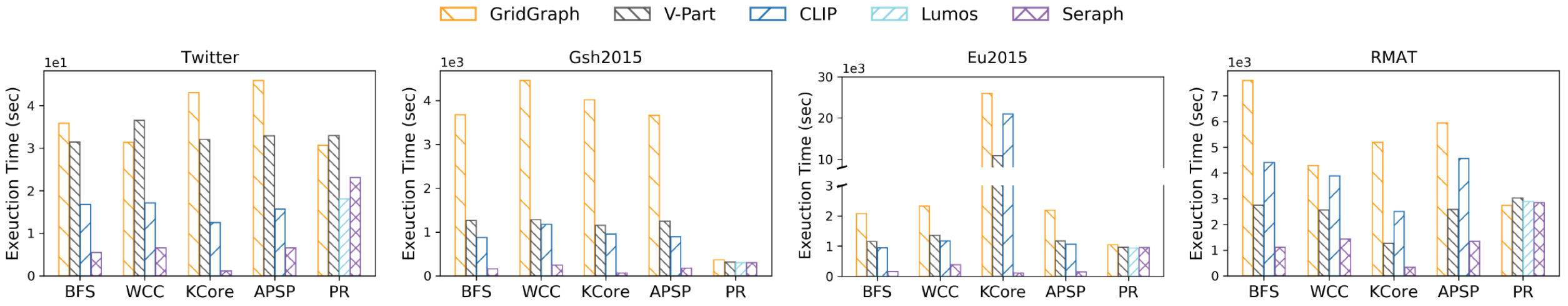
- Introduction
- Background and Motivation
- Seraph
  - Hybrid Format
  - Vertex Passing
  - Selective Pre-computation
- Evaluation

Evaluation Setup

- Evaluated SOTA graph systems:
  - Fully-external: GridGraph@ATC'15, V-Part@FAST'19, CLIP@ATC'17, and Lumos@ATC'19
  - Semi-external: Graphene@FAST'17
  - Shared-memory: Ligra+@DCC'15
- Five graph algorithms are evaluated:
  - Breath-first Search (BFS), Weakly Connected Component (WCC), KCore, All-Pair Shortest Pair (APSP), and PageRank (PR).
- Graph datasets:

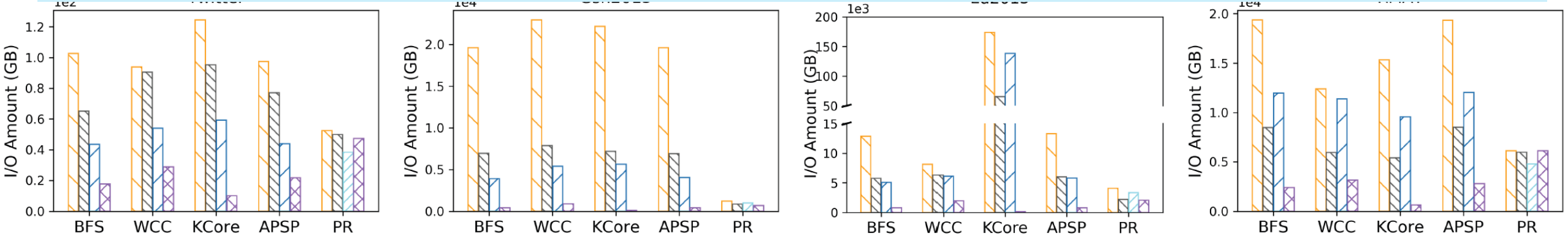
Name	V	E	Graph Size
Twitter	42 M	1.4 B	11.2 GB
Gsh2015	988 M	33.88 B	271 GB
Eu2015	1.1 B	91.8 B	734 GB
RMAT	8.6 B	112 B	1.7 TB

Fully-External Graph System Comparisons



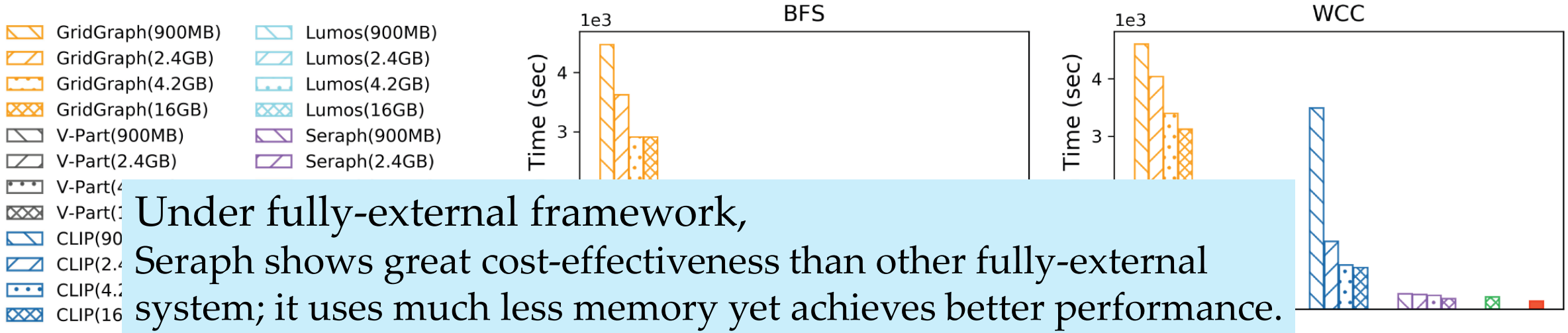
(a) Execution time on Twitter. (b) Execution time on Gsh2015. (c) Execution time on Eu2015. (d) Execution time on RMAT.

Seraph outperforms GridGraph, V-Part, and CLIP by 8.9x, 4.9x, and 4.0x on average



(e) I/O amount on Twitter. (f) I/O amount on Gsh2015. (g) I/O amount on Eu2015. (h) I/O amount on RMAT.

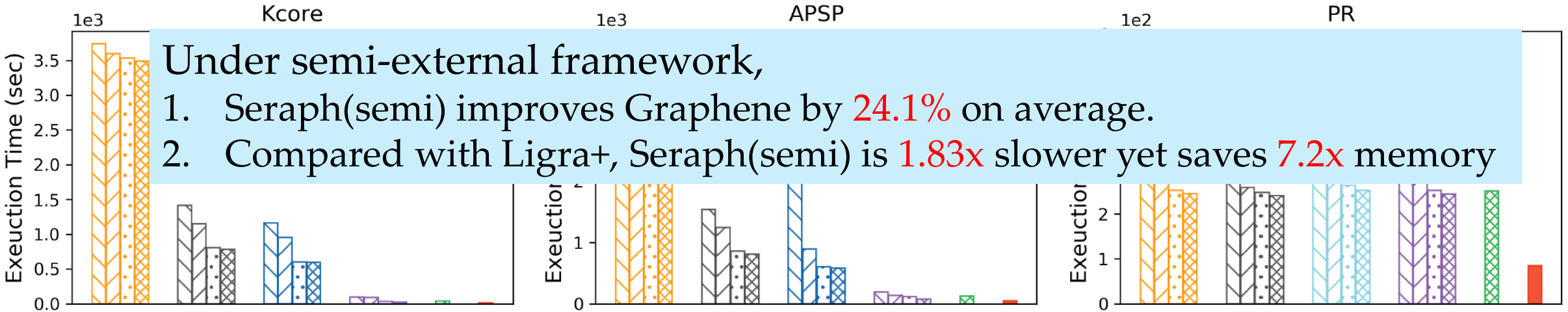
Memory Scalability



Under fully-external framework,
Seraph shows great cost-effectiveness than other fully-external system; it uses much less memory yet achieves better performance.

(a) Execution time on BFS.

(b) Execution time on WCC.



Under semi-external framework,
1. Seraph(semi) improves Graphene by 24.1% on average.
2. Compared with Ligra+, Seraph(semi) is 1.83x slower yet saves 7.2x memory

(c) Execution time on Kcore.

(d) Execution time on APSP.

(e) Execution time on PR.

Conclusion

- This work develops Seraph, an efficient fully-external graph computation system enable scaling graph processing for large-scale graphs on single machines.

- Seraph is developed based on the principle of on-demand processing. Three designs are proposed in Seraph for further performance improvement.
  - Hybrid Format.
  - Vertex Passing.
  - Selective Pre-computation.

- The evaluation shows that Seraph is an efficient fully-external graph system.

Thank you for your attention
Q&A