Graph Partitioning for Near Memory Processing

Chenfeng Zhao Roger D. Chamberlain Xuan Zhang
McKelvey School of Engineering, Washington University in St. Louis

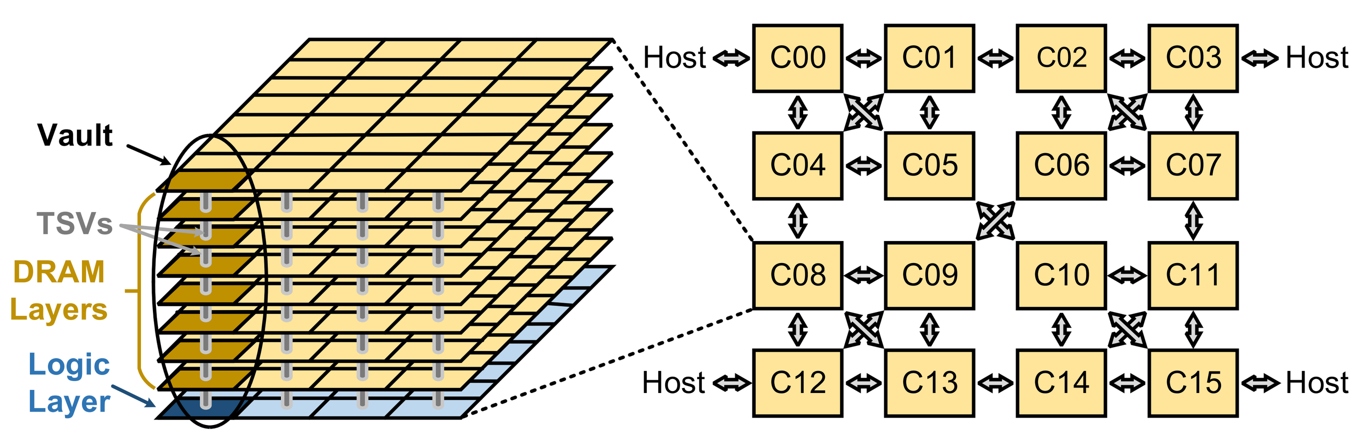
CNS-1739643 & CNS-1763503

Background and Motivation

Graph processing applications have a high memory bandwidth requirement.

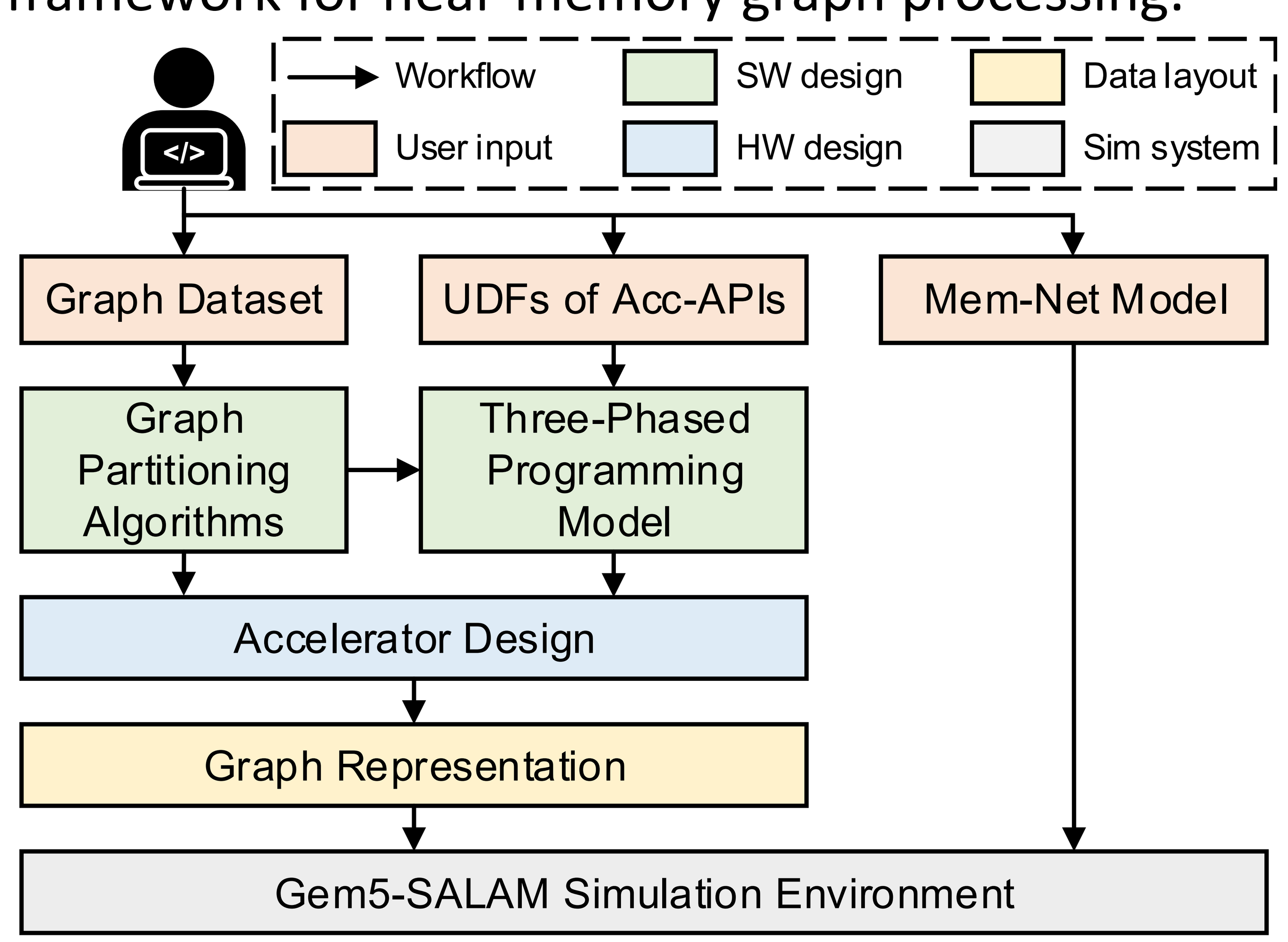
Near-Memory Processing (NMP) architectures based on multiple 3D memory cubes are proposed to accelerate parallel graph processing applications.

However, cross-cube communication is a system bottleneck, taking a significant portion of execution time (12%-78%) and energy consumption (12%-73%).



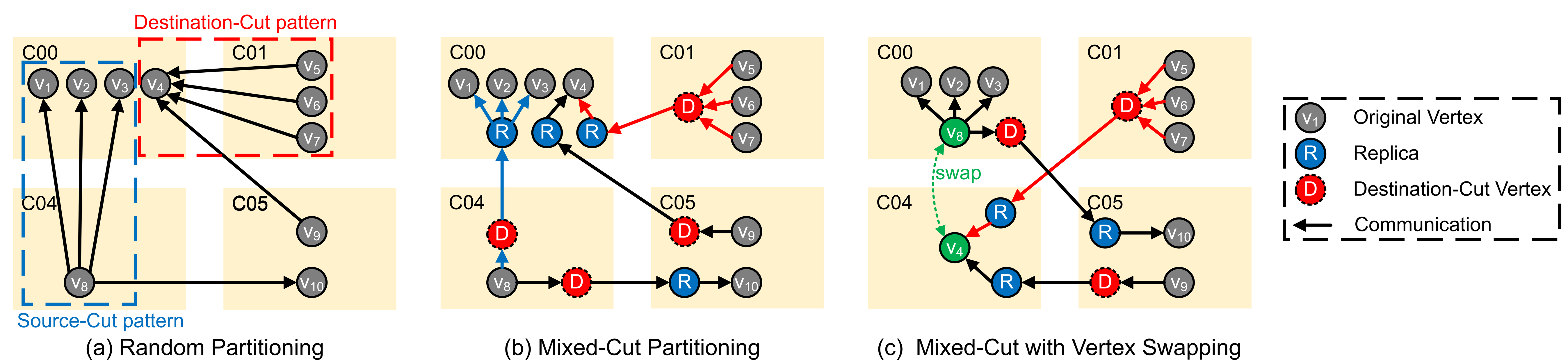
SuperCut

In this work, we propose SuperCut, a co-design framework for near-memory graph processing.



Graph Partitioning algorithms

SuperCut adopts a set of partitioning algorithms to preprocess graph datasets, including mixed-cut partitioning, a stochastic-and-heuristic-based optimization algorithm and partial graph partitioning.



Programming Model

We propose a three-phase programming model supporting the partitioning algorithms.

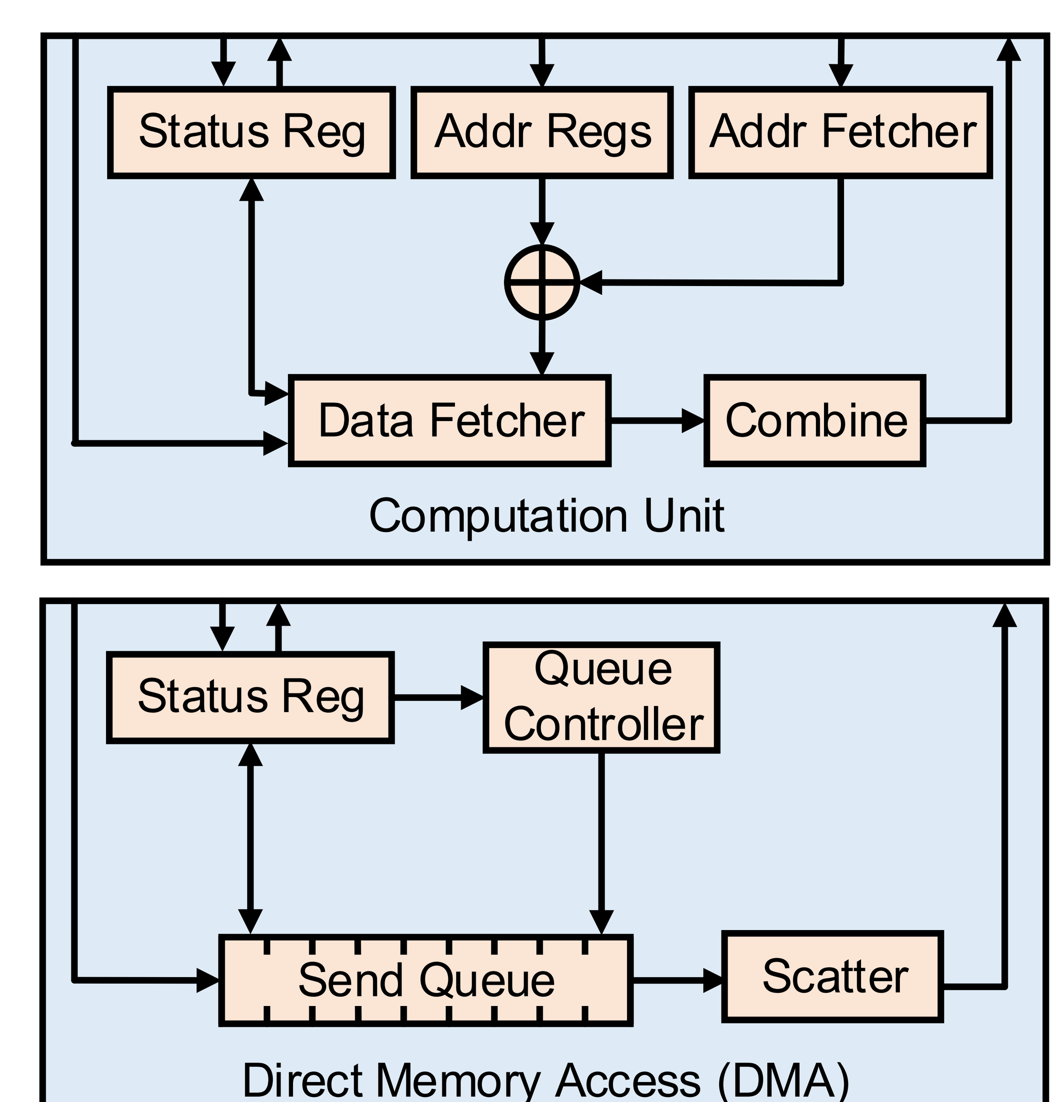
It explicitly handles computation and communication via user-defined functions.

Input: The SuperCut graph H and original graph G
Output: Results of graph processing applications

for each original vertex $v_{org} \in G$ **do**
  gather_combine(v_{org})
end for
for each destination-cut vertex $v_{dc} \in H$ **do**
  update $\leftarrow$ gather_combine(v_{dc})
  scatter(*update*)
end for
for each original vertex v_{org} and replica v_r **do**
  apply(v_{org}); apply(v_r)
end for

Accelerator Design

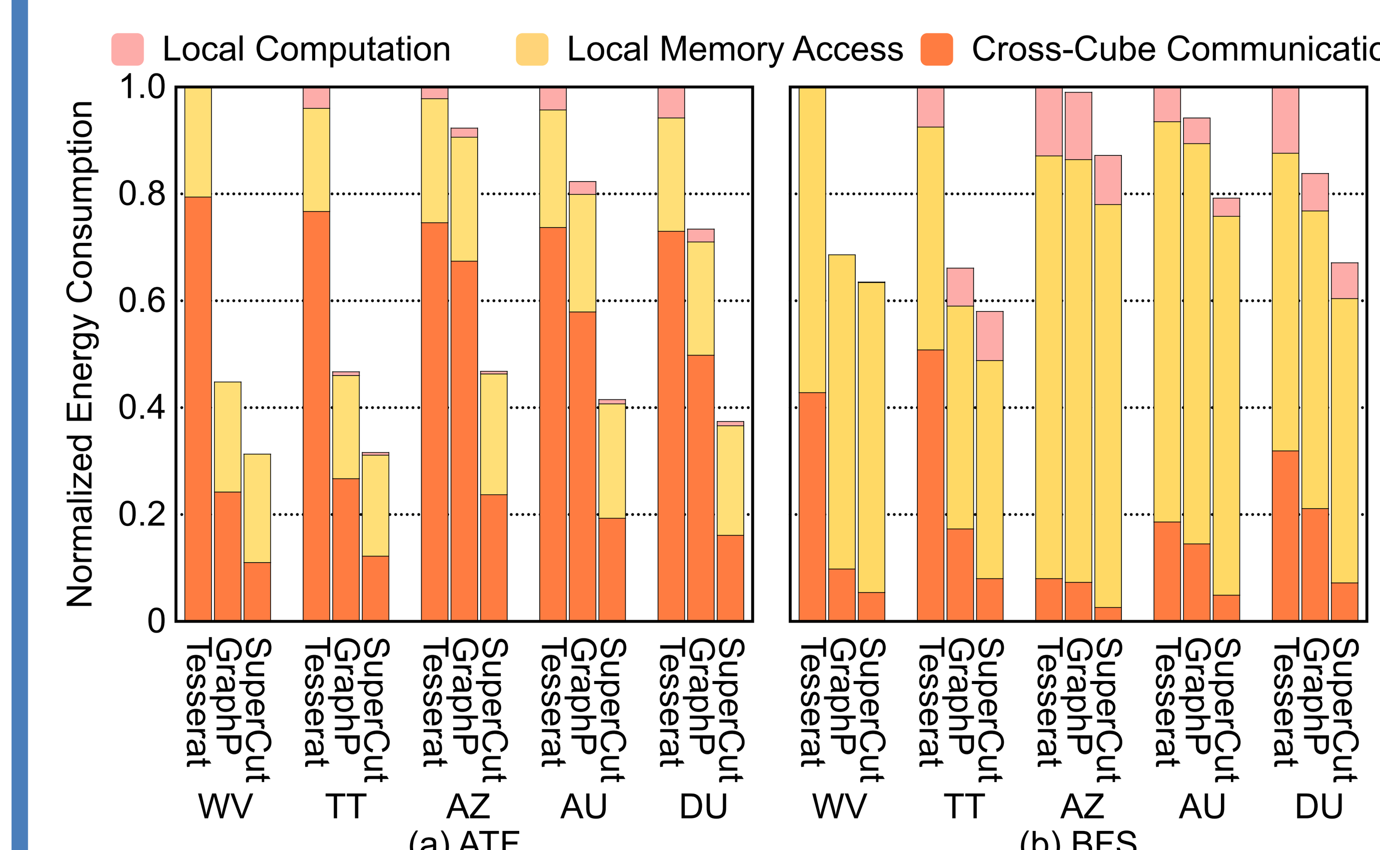
Specialized accelerators are proposed based on our programming model, all of which are mapped to FPGA resources on the logic layer of 3D memory cubes.



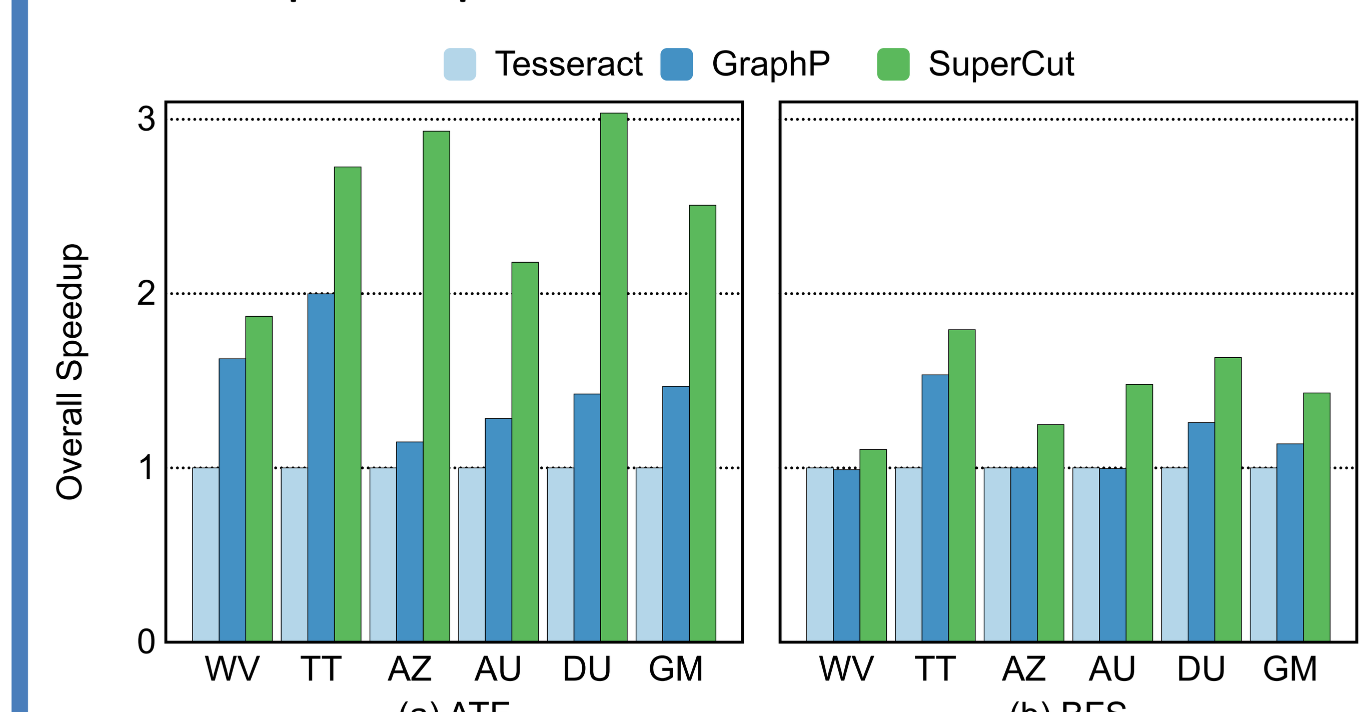
Evaluation Results

Case study: Average Teenage Follower (ATF) and Breadth-First Search (BFS) on 5 graph datasets .

Energy Evaluation: SuperCut achieves 1.09x to 2.0x total energy reduction relative to the state-of-the-art.



Performance Evaluation: SuperCut achieves 1.12x to 2.6x speedup relative to the state-of-the-art.



[1] J. Ahn et al., A scalable processing-in-memory accelerator for parallel graph processing. In Proc. of 42nd International Symposium on Computer Architecture, pages 105–117, 2015.
[2] Mingxing Zhang et al., GraphP: Reducing communication for PIM-based graph processing with efficient data partition. In Proc. of International Symposium on High Performance Computer Architecture (HPCA), pages 544–557. IEEE, 2018.