## Software Enabling for Memory-centric Compute Architectures for Datacenter GPUs

Memory-centric Computing Architecture (MCA), presented in the companion paper, opens up an interesting and important front on its programmability and adoption in the software stack. Available Processing-in-Memory (PIM) solutions pick vertical library mode for offloading compute kernels with a driver model and an accelerator view for PIM architecture. Such a mode of operation restricts seamless data sharing between traditional GPU and PIM execution, especially, in scenarios where the end-to-end workload cannot be efficiently mapped onto a PIM architecture (e.g., executing the prefill stage traditionally in GPU compute, or performing pre- or post-processing complex computations in EUs/XeCores). Therefore, we envision lowering software changes in the stack (e.g., compiler, runtime and driver), while providing a unified view of operators across GPU and memory-centric compute to the developer. In this paper, we focus on exploring solutions for 1) shared data layout across GPU and memory-centric compute enabling continuous batching, 2) handling data fragmentation due to memory interleaving and address hashing, 3) exposing programming interface for memory-centric compute, and 4) managing data and computation across virtual and physical address domains. We’ll also present needed hardware enhancements to enable these required software changes.