Example Quote from Citations

Alexander C. Berg
UNC Chapel Hill

improve utilization [3, 13, 19, 35]. A framework called Kernelet [34] falls into this category, but is of particular interest to us due to the fact that GPU co-scheduling is considered in order to improve utilization. Kernelet, however, requires heavy instrumentation and does not consider co-scheduling unmodified workloads. Additionally,

Otterness, Nathan, Ming Yang, Sarah Rust, Eunbyung Park, James H. Anderson, F. Donelson Smith, Alex Berg, and Shige Wang. “An evaluation of the NVIDIA TX1 for supporting real-time computer-vision workloads.” **RTAS, 2017.**

Murali Annavaram
USC

Zhong and He [10] proposed a dynamic kernel slicing that partitions a kernel into several smaller kernels so that multiple kernels can more efficiently share the resources. Adriaens et al. [12] proposed spatial multitasking, which runs multiple applications on different sets of SMs. Ukidave

Xu, Qiumin, Hyeran Jeon, Keunsoo Kim, Won Woo Ro, and Murali Annavaram. "Warped-slicer: efficient intra-SM slicing through dynamic resource partitioning for GPU multiprogramming.“ **ISCA, 2016.**

Example Quote from Citations

Dimitrios Nikolopoulos
Virginia Tech

gScale [Xue et al. 2016] solves gVirt's scalability limitation. gVirt partitions the global graphics memory (2 GB) into several fixed size regions and allocates them to vGPUs. Due to the recommended memory allocation for each vGPU (e.g. 448 MB in Linux), gVirt limits the total number of vGPUs to four. gScale overcomes this limitation by making the GPU memory shareable. For the high graphics memory in Intel GPUs, gScale allows each vGPU to maintain its own private shadow graphics translation table (GTT). Each private GTT translates the vGPU's logical graphics address to

Hong, Cheol-Ho, Ivor Spence, and Dimitrios S. Nikolopoulos. "GPU virtualization and scheduling methods: A comprehensive survey." **CSUR, 2017.**