This is the title of your abstract for the 13th Scientific FORTH Retreat

Manos Pavlidakis [1,2#*], Stelios Mavridis [2] , Antonis Chazapis [2], and Angelos Bilas [1,2]

1 Computer Science Department, University of Crete, Greece

2 Institute of Computer Science, Foundation for Research and Technology - Greece

Presenting author: Manos Pavlidakis, email: manospavl@ics.forth.gr

* Corresponding author: Manos Pavlidakis, email: manospavl@ics.forth.gr

ABSTRACT

Future servers will be equipped with a plethora of heterogeneous accelerators to compensate for the increased processing demands of applications. However, this increasing use of different accelerator types escalates the programming effort in applications that use multiple & heterogeneous accelerators. Existing programming models (e.g., SYCL, CUDA) enforce applications to be tightly coupled with accelerators since a developer should manually manage task assignments and data transfers (i.e., accelerator management). Tightly coupling accelerators and applications affect also cloud providers due to accelerator fragmentation. Popular accelerators will be overloaded while others will idle just consuming power. Finally, accelerators improve the compute density of servers if they are utilized. We have to spatial share them among different applications to keep them utilized. Existing spatial sharing services are a black box, without leaving space for cloud providers to implement scheduling policies that reflect their needs. Even if accelerators open their sharing API, this will differ from accelerator to accelerator, making it unmanageable to implement a scheduler for all the different accelerator types.

We present RuSH an RPC-like approach, with a new programming model and runtime that addresses these challenges. According to RuSH's programming model, an application is divided into host-code, accelerator management, and kernel code. The host-code is accelerator agnostic and is placed in RuSH client, while the accelerator management and kernel-code in the server. To abstract accelerator compute resources and type RuSH offers Virtual Accelerators (VAs) to applications. Regarding memory management, the RuSH provides Memory Handles. The handle promises to receive the requested memory, but applications are unaware of the time and the place of the actual allocation. RuSH server assigns VAs and performs the data allocations to accelerators transparently and dynamically to applications. Finally, the RuSH server incorporates a spatial sharing service common for different accelerator types. Due to the common sharing service and the fine-grain view of application tasks, cloud providers can implement their scheduling policies.

We use RuSH to virtualize four processing units. Our spatial sharing mechanism provides comparable performance (up to 20% improvement) to state-of-the-art sharing mechanisms. RuSH can dynamically increase/decrease the accelerators provided to one application improving the application performance by up to 2x. The overhead of RuSH compared to the native execution is 10%, while other approaches imply 30% for the same applications.