Zorua

A Holistic Approach to Resource Virtualization in GPUs

Session 2A – Monday, 5:20 PM

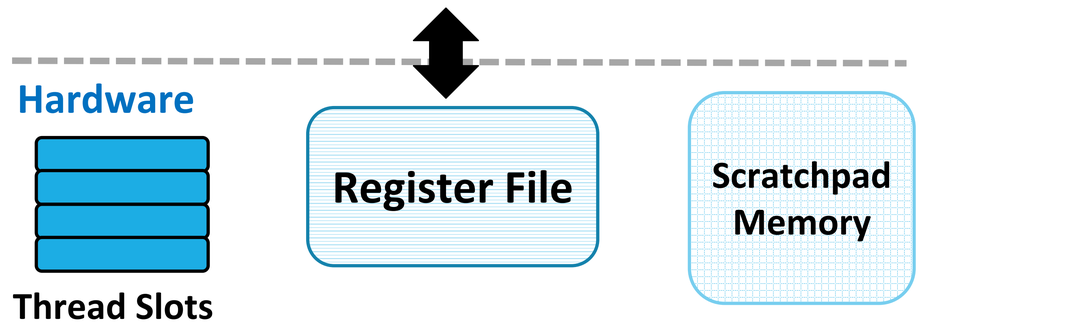
Nandita Vijaykumar, Kevin Hsieh, Gennady Pekhimenko, Samira Khan, Ashish Shrestha, Saugata Ghose, Adwait Jog, Phillip B. Gibbons, Onur Mutlu

Carnegie Mellon University · Microsoft Research · University of Virginia · William & Mary · ETH Zürich

Overview

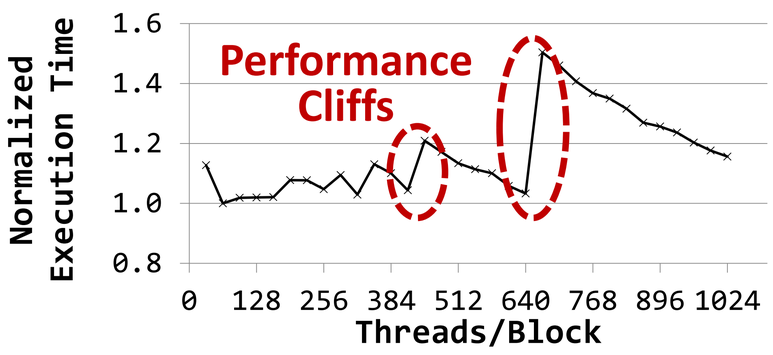
- **Problem:** Major on-chip resources in GPUs are managed by the ***programmer/software***
- **Key Issues:** Leads to several challenges in obtaining high performance:
  - **Programming Ease:** Requires programmer effort to optimize resource usage
  - **Performance Portability:** Optimizations do not port well across different GPU architectures
  - **Resource Inefficiency:** Underutilized resources even in optimized code
- **Our Goal:**
  - Reduce dependence of performance on programmer-specified resource usage
  - Enhance resource efficiency for optimized code
- **Our Approach:** ***Decouple*** the programmer-specified resource usage from the allocation in the hardware

***Problem*: Tight coupling between programmer specified resource usage and hardware allocation**



Tight Coupling Between Resource Specification and Allocation Leads to Several Challenges

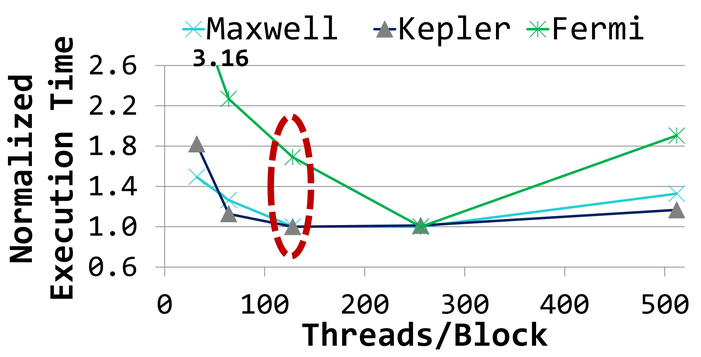
▪ Programming Ease



MST (*Minimum Spanning Tree*)

Requires programmer effort to avoid sub-optimal specifications

▪ Performance Portability



DCT (*Discrete Cosine Transform*)

Programs need to be retuned to fit different GPUs

▪ Resource Efficiency

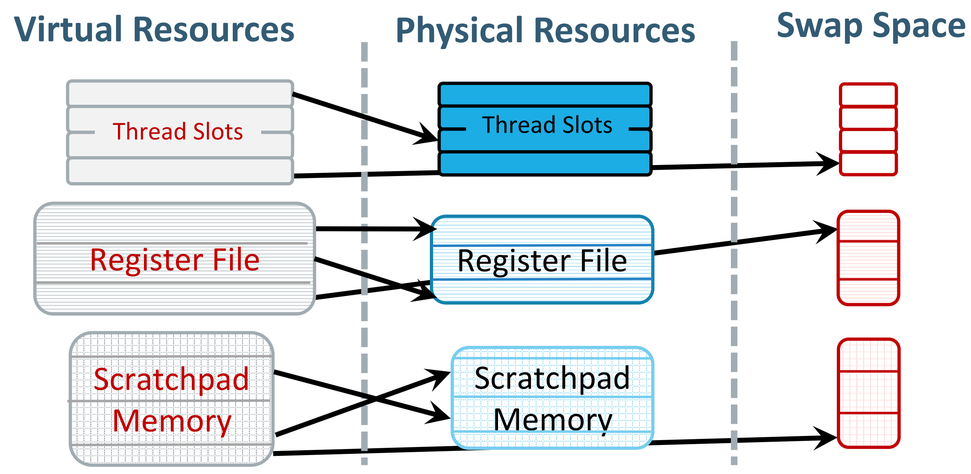
```
__global__ void CUDAkernel2DCT(float *dst, float *src, int I){
   int OffsThreadInRow = threadIdx.y * B + threadIdx.x;
   ...
   for(unsigned int i = 0; i < B; i++)
            bl_ptr[i * X] = src[i * I];

   __syncthreads();
   ...
   CUDAsubroutineInplaceDCTvector(…);

   for(unsigned int i = 0; i < B; i++)
            dst[i *I] = bl_ptr[i * X];
   . ..
}
```

16 regs · 32 regs · 16 regs

Resource inefficiency results from worst-case static allocation

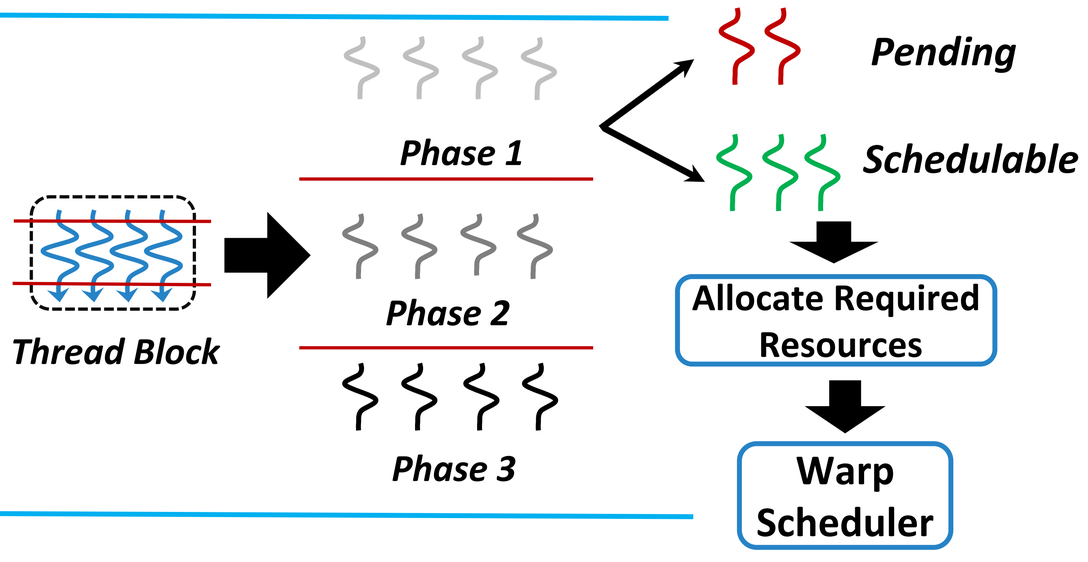
Zorua: Decouple Programmer/Software Resource Specification from Hardware Allocation

- **Virtualizing** major on-chip GPU resources:
  - **Dynamic** allocation/deallocation of resources
  - Careful **oversubscription** of resources using a swap space in the memory hierarchy
- Two design challenges
  - **Control** the **extent of oversubscription**
  - **Coordinate** virtualization of **multiple** on-chip resources

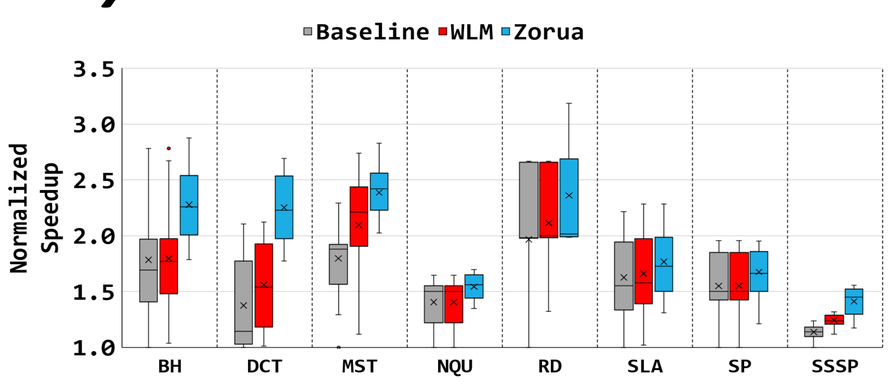


Key Components of Zorua

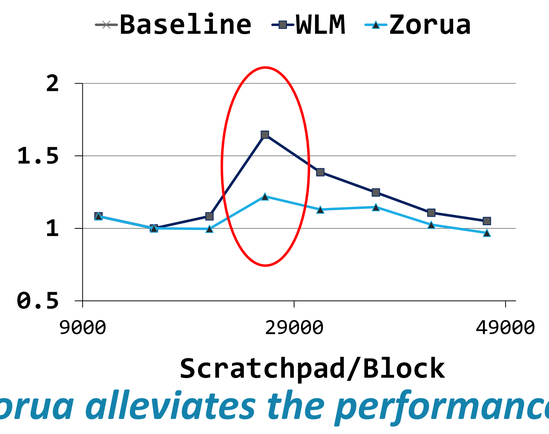
- **Compiler**: Statically finds program ***phases***, annotates per-phase resource needs
- **Coordinator**: Adaptive runtime system that makes oversubscription & resource allocation decisions



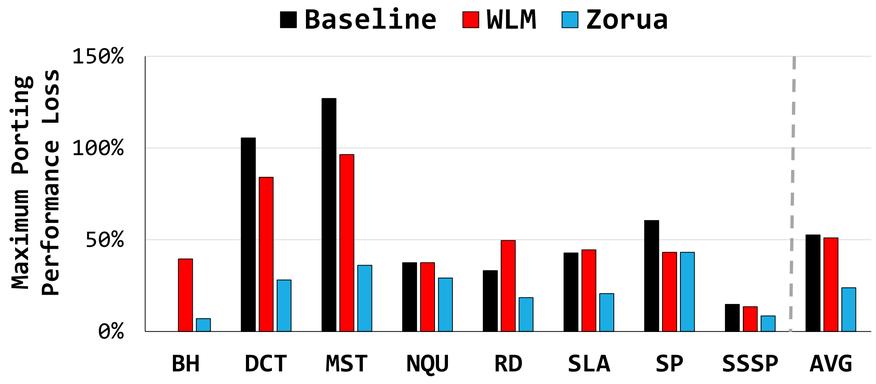
Key Results



Zorua reduces the dependence of performance on resource specification



Zorua alleviates the performance cliffs resulting from un-optimized specifications



Zorua reduces the performance loss from porting across GPU architectures