RUTGERS

THE STATE UNIVERSITY OF NEW JERSEY

Towards a GPU SDN Controller

Eduard Gibert Renart
Eddy Zheng Zhang
Badri Nath

Overview of the presentation

- Introduction
- Gpu Architecture
- Our approach to a GPU SDN Controller handling warp divergence
- Results
- Future work

Introduction

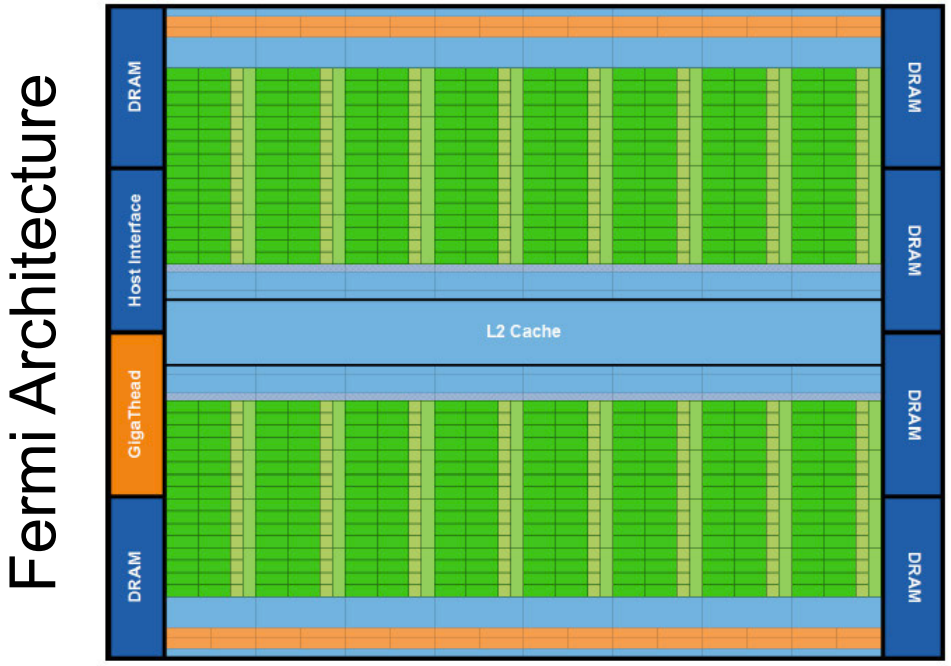
- My team:
  - Myself, Eduard Gibert Renart, 2nd year PhD student
  - Dr. Zheng Zhang, GPU Expert
  - Dr. Badri Nath, Networking Expert

GPU

ARCHITECTURE

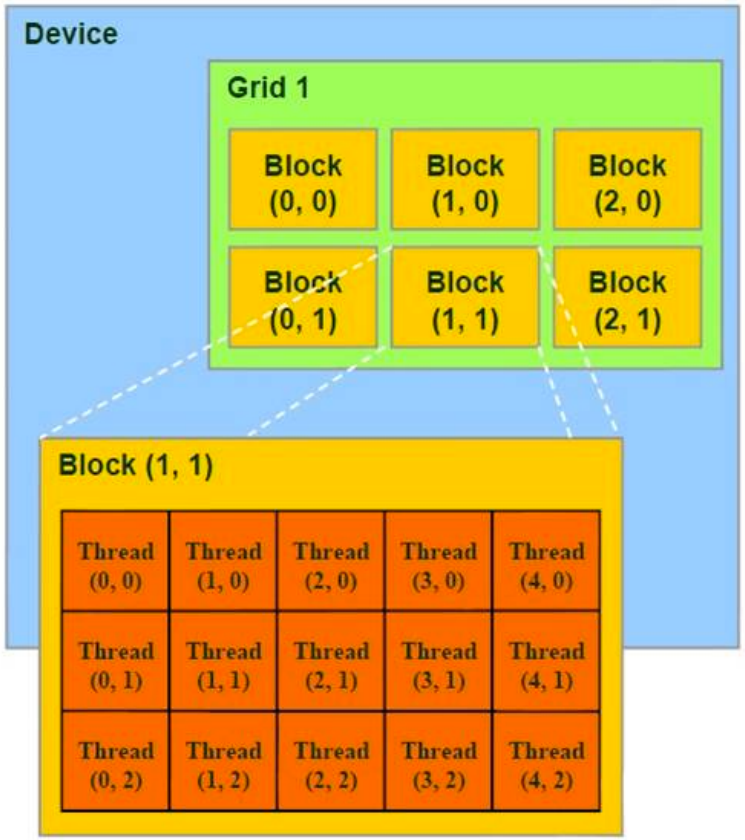
GPU Architecture

- SM - Streaming multiprocessors with multiple processing cores
  - Perform the actual computations
  - Each SM contains 32 processing cores
  - Up to 16 SMs on a card for a maximum of 512 compute cores



GPU Execution Model

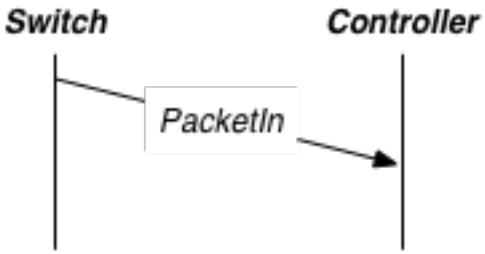
- A grid is composed of blocks which are completely independent
- A block is composed of threads which can communicate within their own block
- 32 threads from a warp

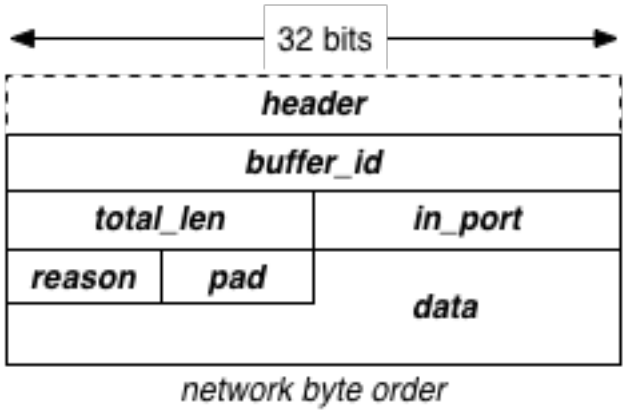


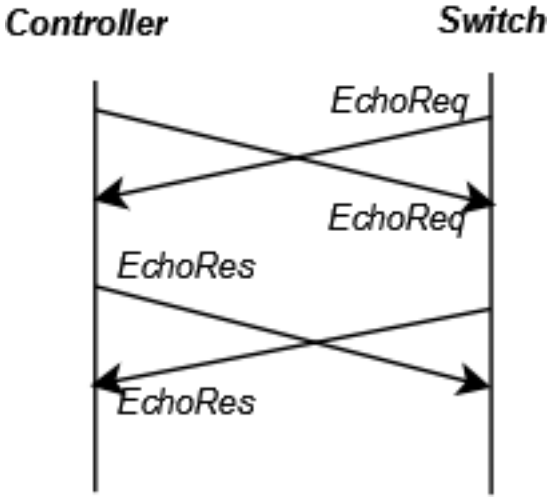
Previous Work

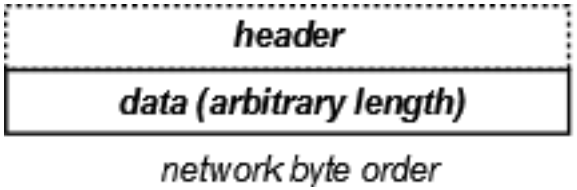
- PacketShader a GPU Software Router [SIGCOMM ’10].
- SSLShader: Cheap SSL Acceleration with Commodity Processors [NSDI ’11].
- Multi-Layer Packet Classification with Graphics Processing Units [CoNEXT ’14].

SDN Packet Heterogeneity



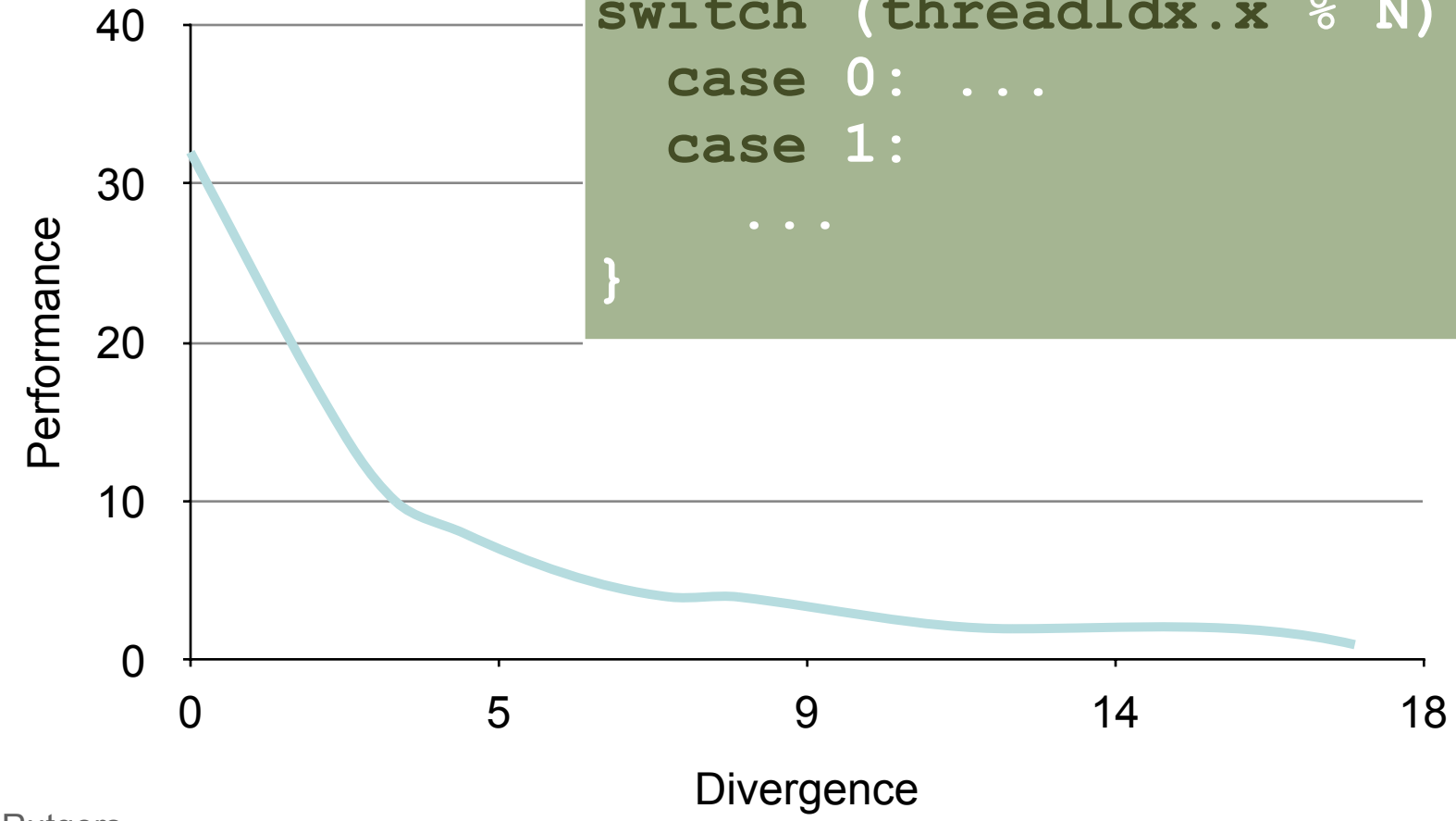






Warp Divergence

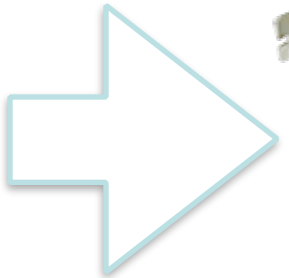
```
switch (threadIdx.x % N) {
  case 0: ...
  case 1:
     ...
}
```

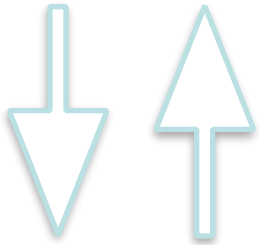


Our approach to a GPU SDN

CONTROLLER HANDLING WARP DIVERGENCE

Basic Idea

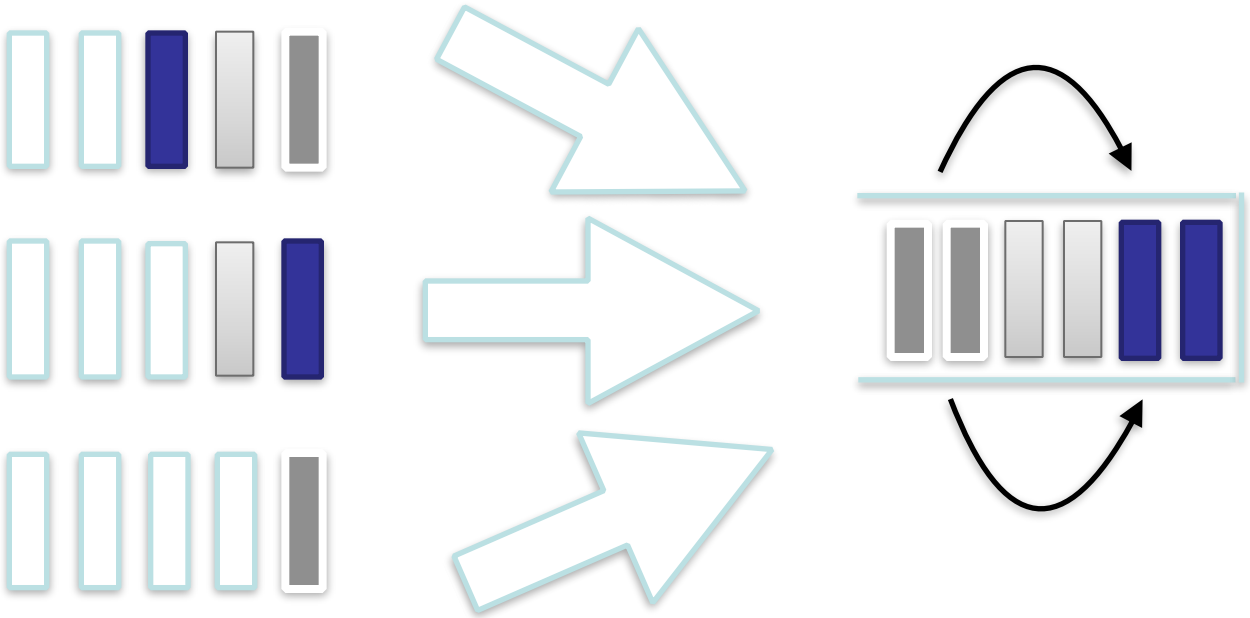




17,000,000 p/s !!!
GPU speedup 5.4x

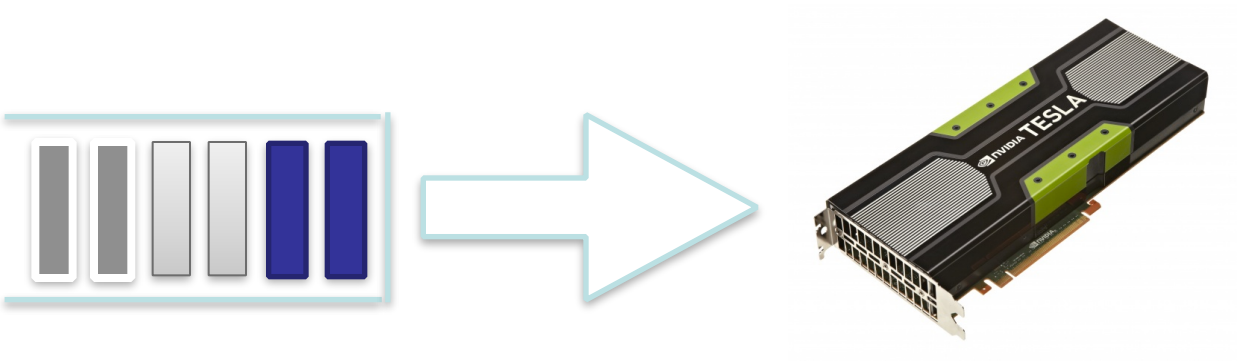
Step 1 - Packet Classification

Rx queue



recv(socket, reply , 2000 , 0);

Step 2 - Host to Device

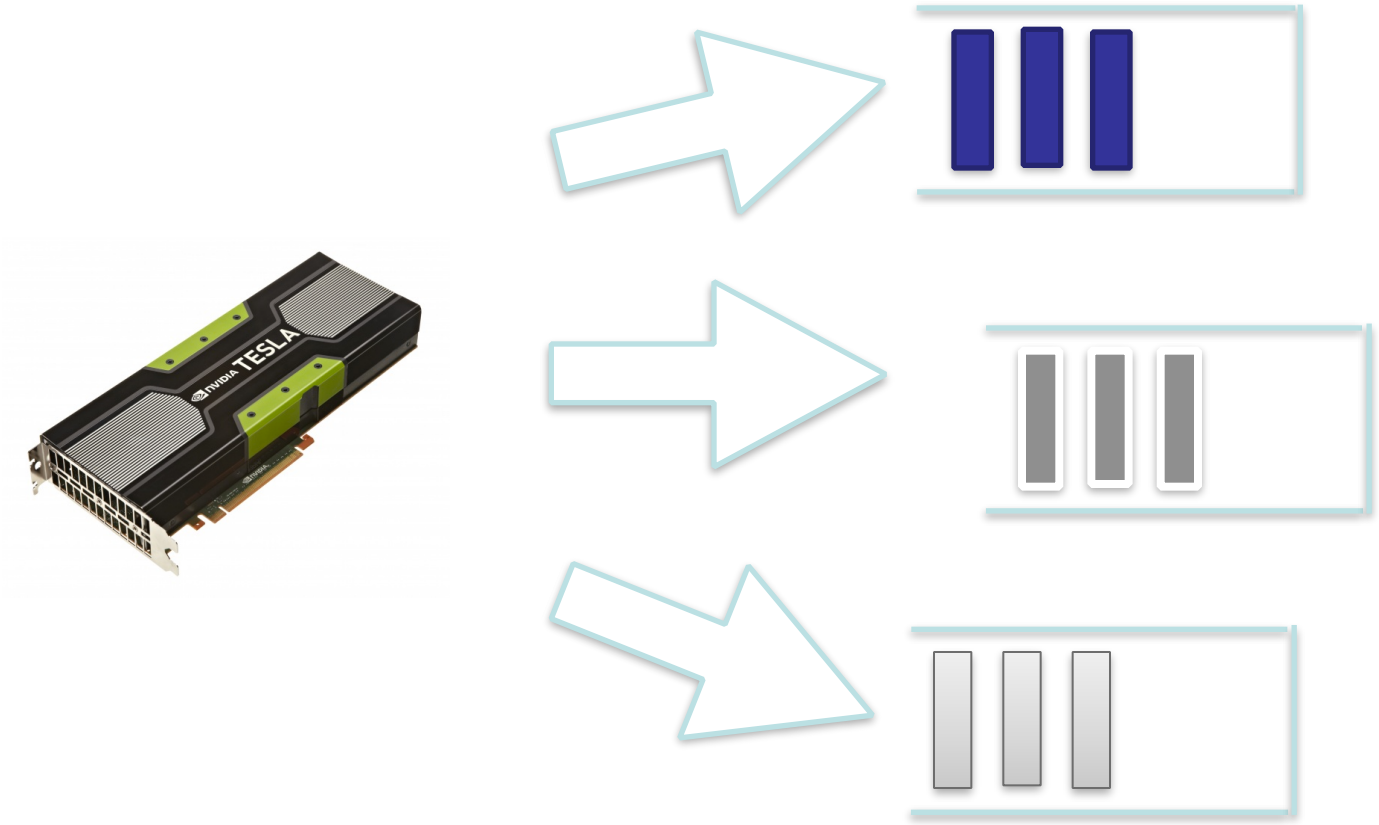


cudaMemcpy(devArray,hostArray,bytes,cudaMemcpyHostToDevice);

Step 3 - Kernel Execution

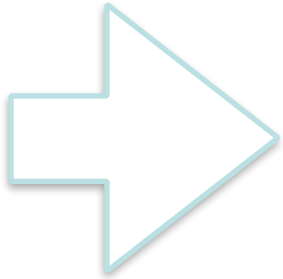

```
kernel<<<gridDimensions,numberOfThreads>>>(dataOut,dataIn);
```

Step 4 - Device to Host



cudaMemcpy(hostArray,devArray,bytes,cudaMemcpyDeviceToHost);

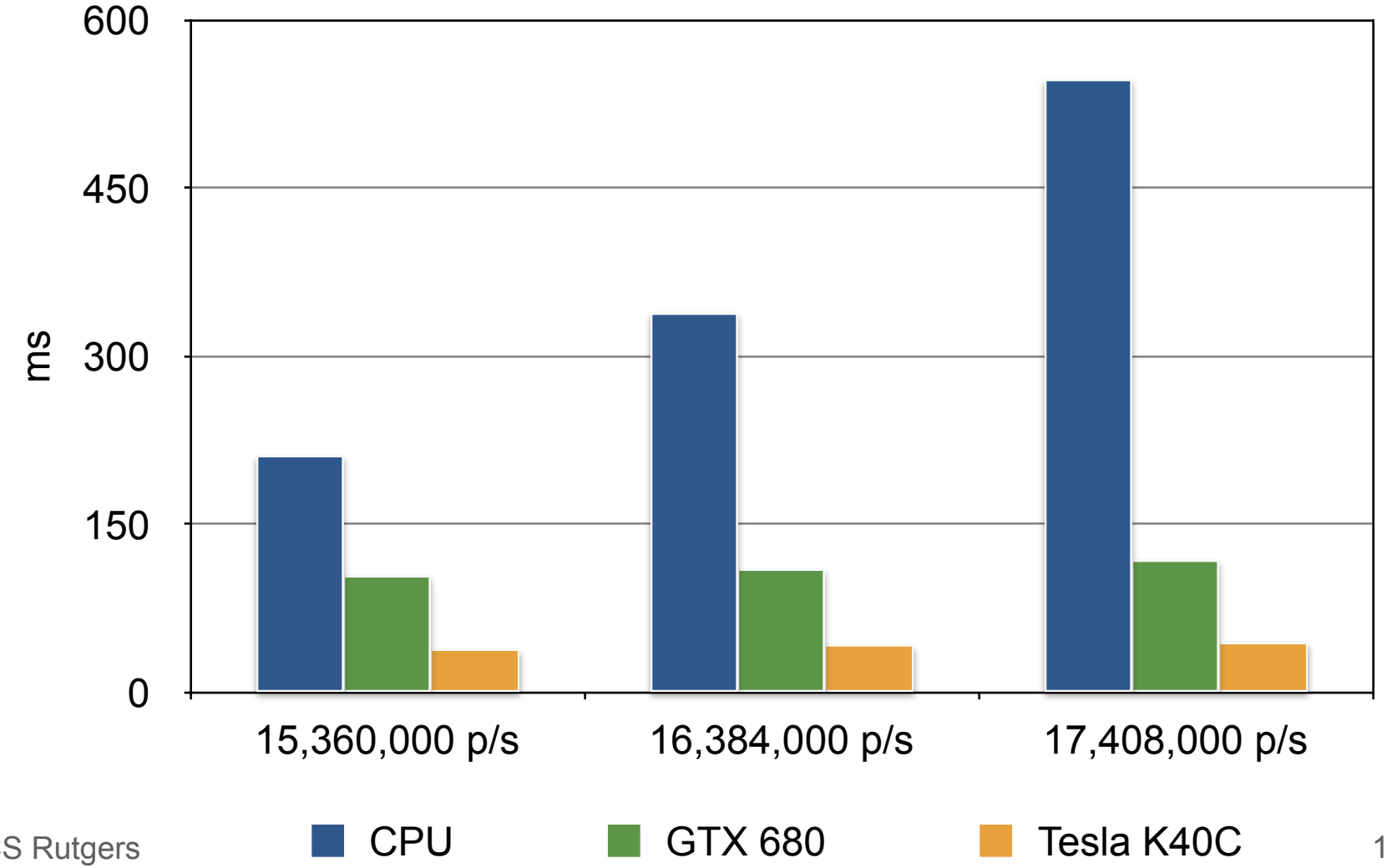
It scales vertically!!



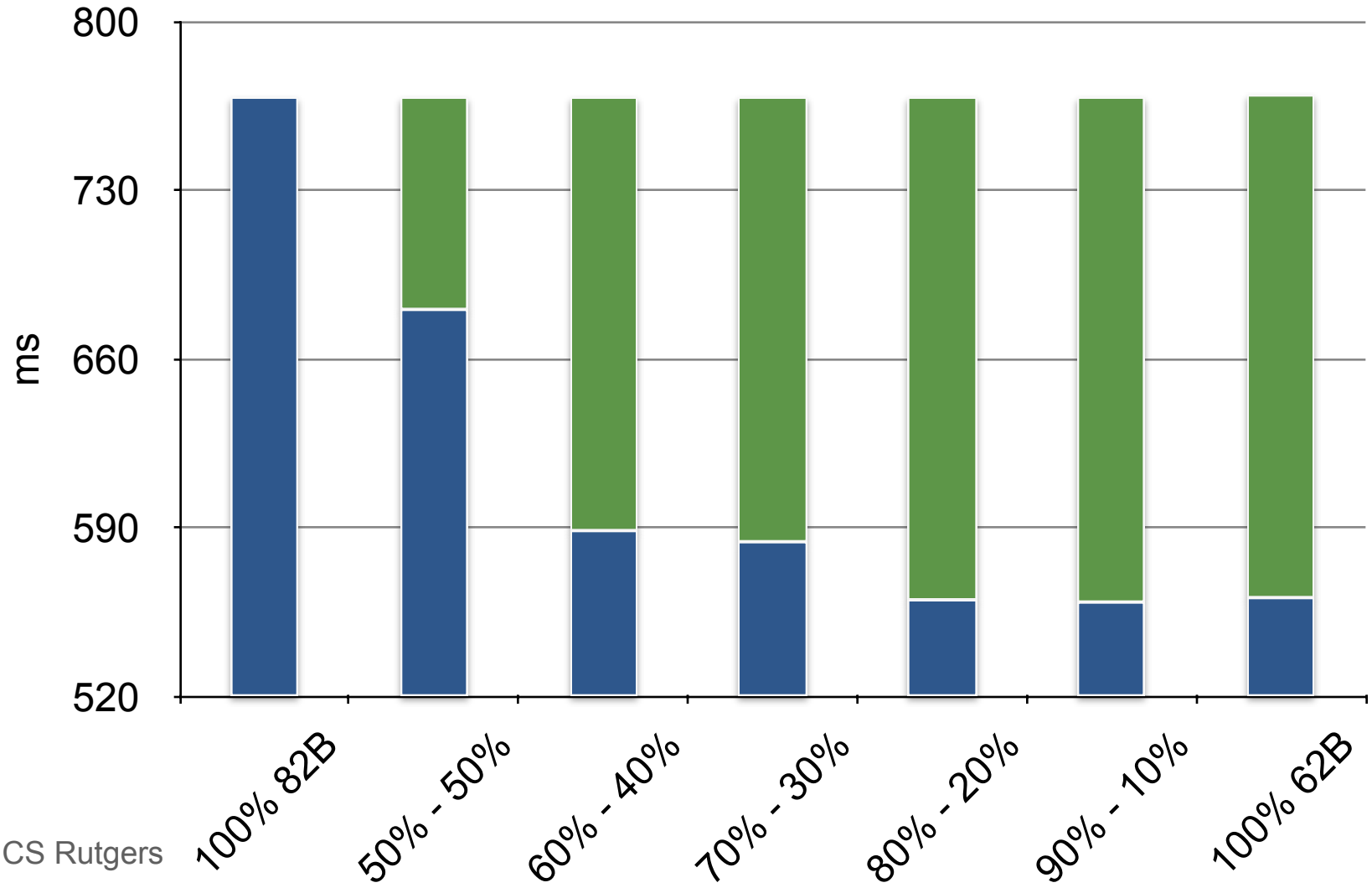
We only need to add more GPUs.

RESULTS

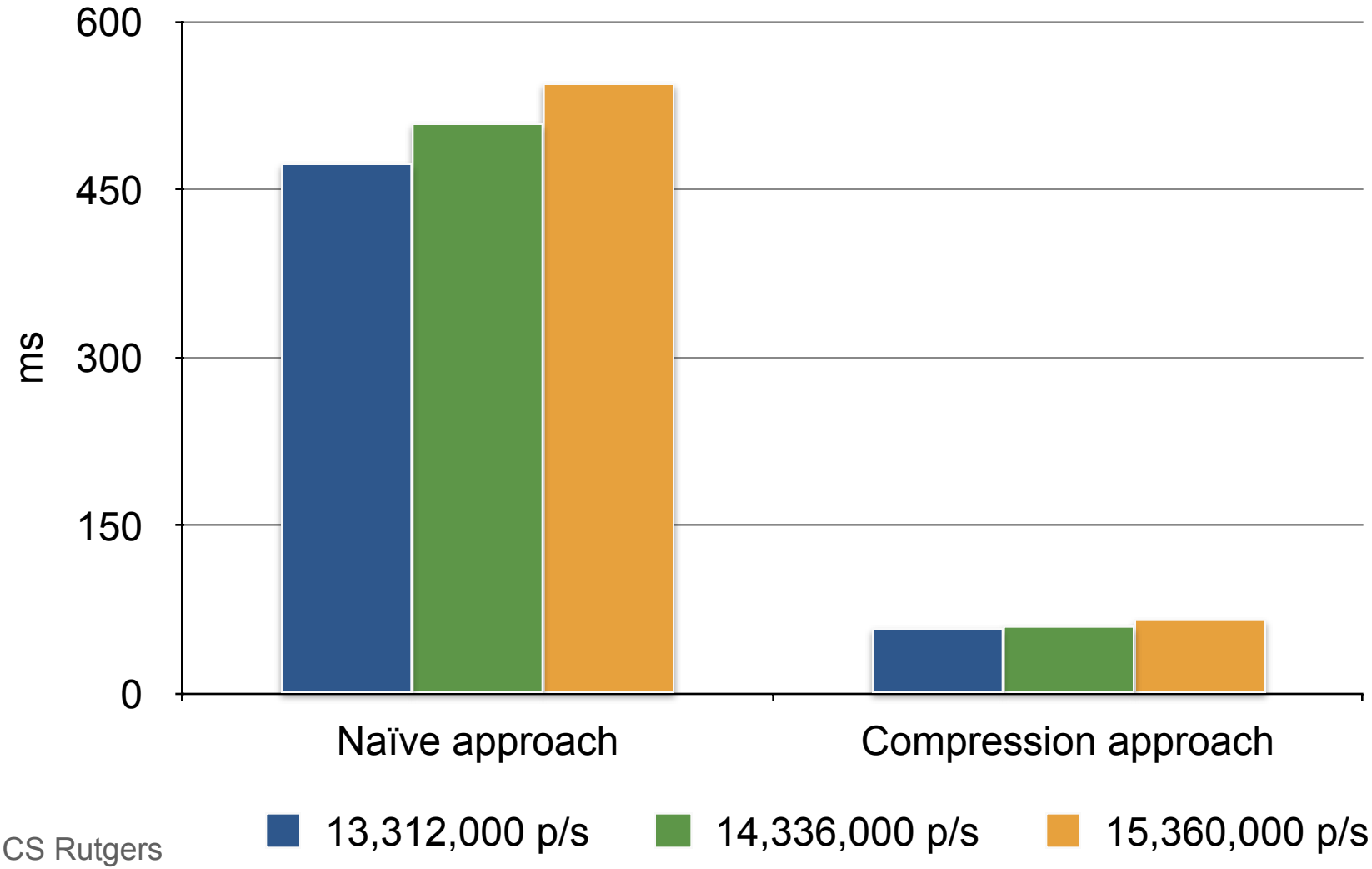
Results (Kernel only)



Packet Size Heterogeneity



Results (Full Process)



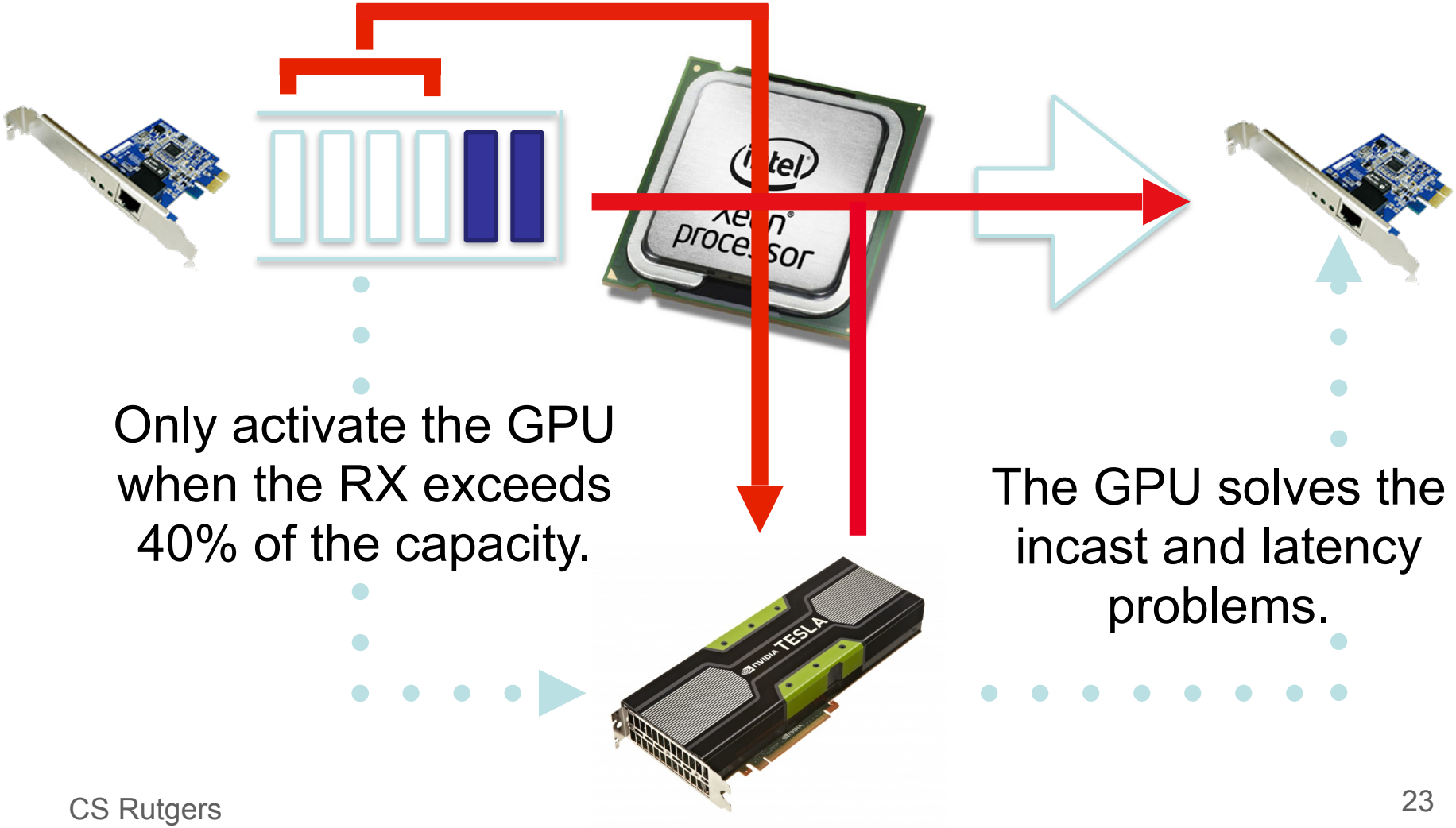
Future

WORK

Hybrid Controller

Traditional SDN Controller only CPU.

Hybrid Controller



Only activate the GPU when the RX exceeds 40% of the capacity.

The GPU solves the incast and latency problems.

Any

QUESTIONS?