Performance analysis of edge detection in an image using cuda

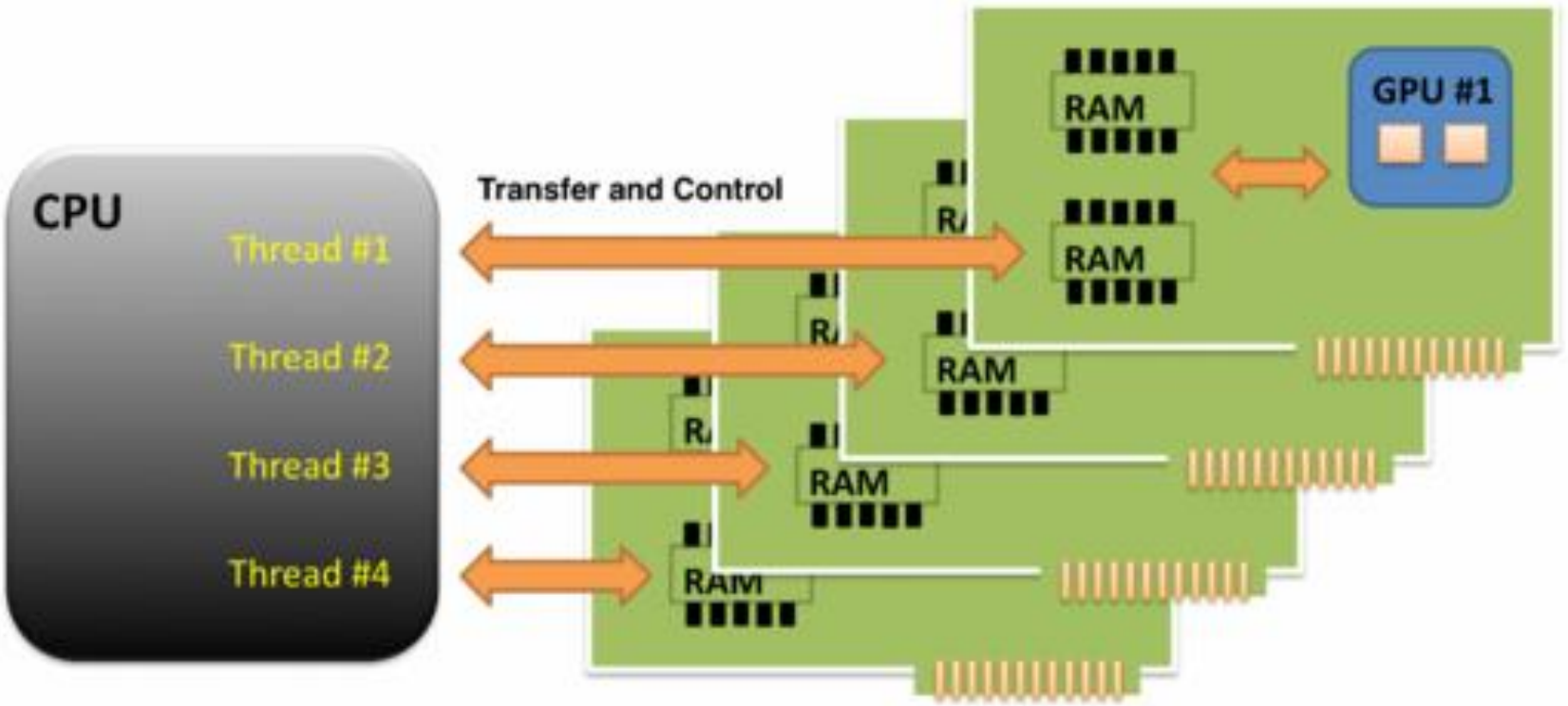
By: Brijesh Kumar
Vikramaditya

Agenda

1. Locality and Parallelism
2. Coordination
3. Purpose of using GPU
4. Machine properties
5. Parallelism
6. Overheads

Locality and Parallelism

- CPU Vs GPU



Purpose of using GPU

- Parallel implementation of edge detection should reduce the time complexity.
- Convolution: Tiled Convolution should further reduce the execution time of algorithm
- Using constant memory for storing various masks and should also enhance the performance

Coordination

- Computation partition
- Data partition
- Concurrency management

Different approch

Serial approach

- Calculate Laplacian of Gaussian
- Read Input Image
- Convolve the image with mask
- Write image to output

Parallel approach

- Calculate Laplacian of Gaussian in CPU
- Read Input image from CPU
- Copy image and Mask data to GPU
- Convolve the image data and mask
- Copy output data to CPU
- Write image to output

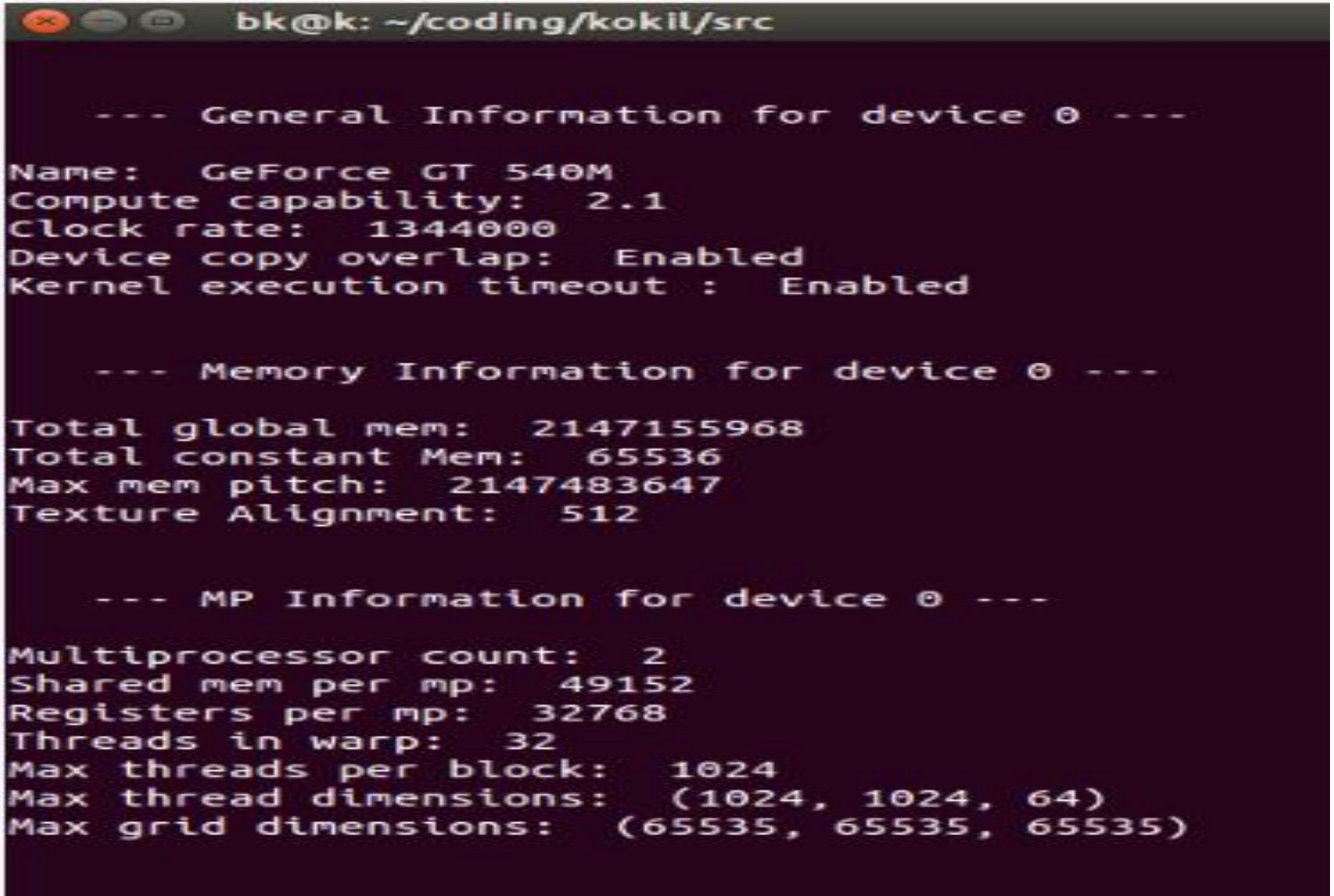
Device Properties

- CPU

Name	i3-380M
Processor Frequency	2.53 GHz
Cache	3MB
Number of core	2
Number of Threads	4
Max Memory Bandwidth	17.1 GB/s

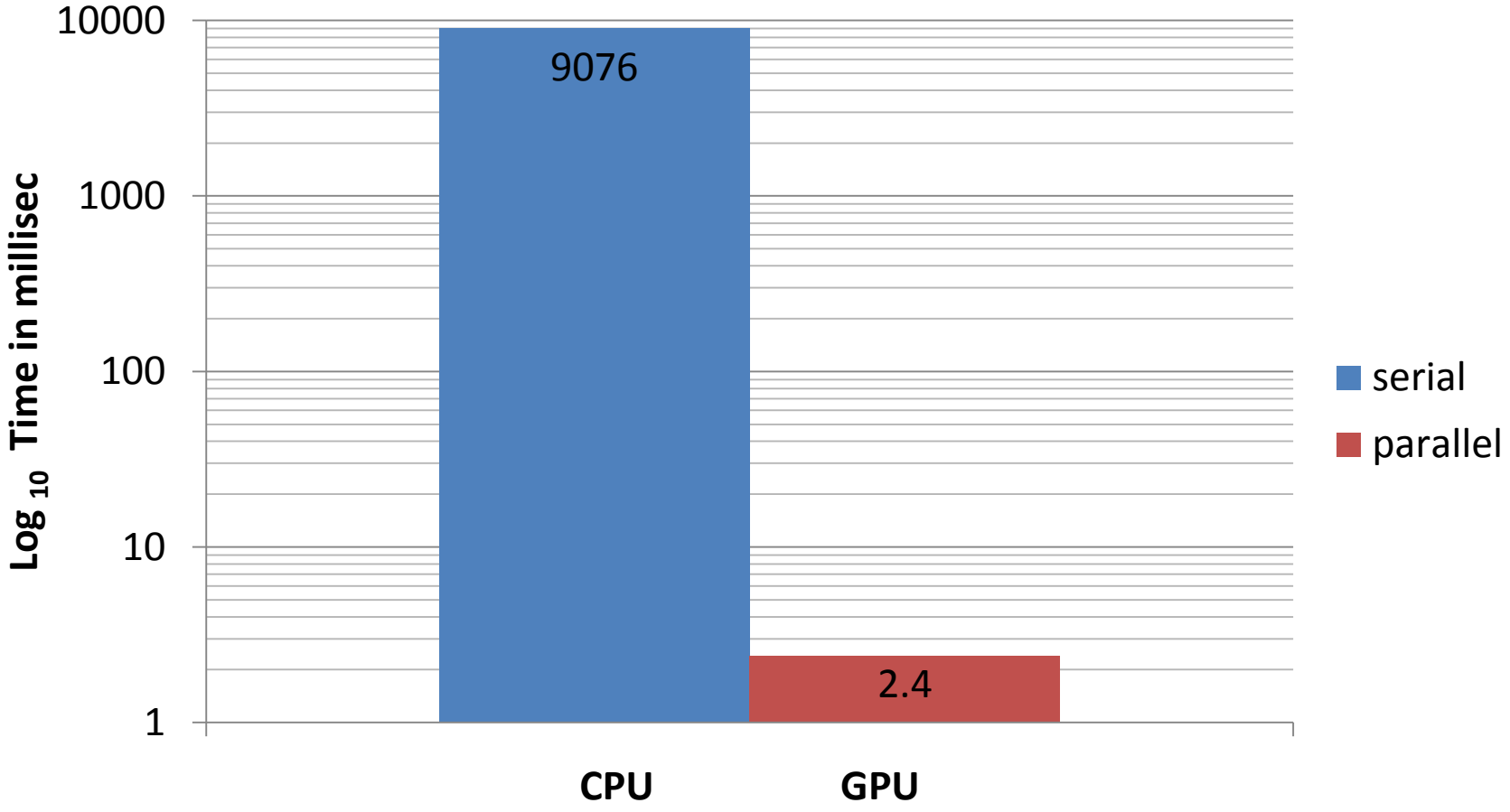
Device Properties

GPU



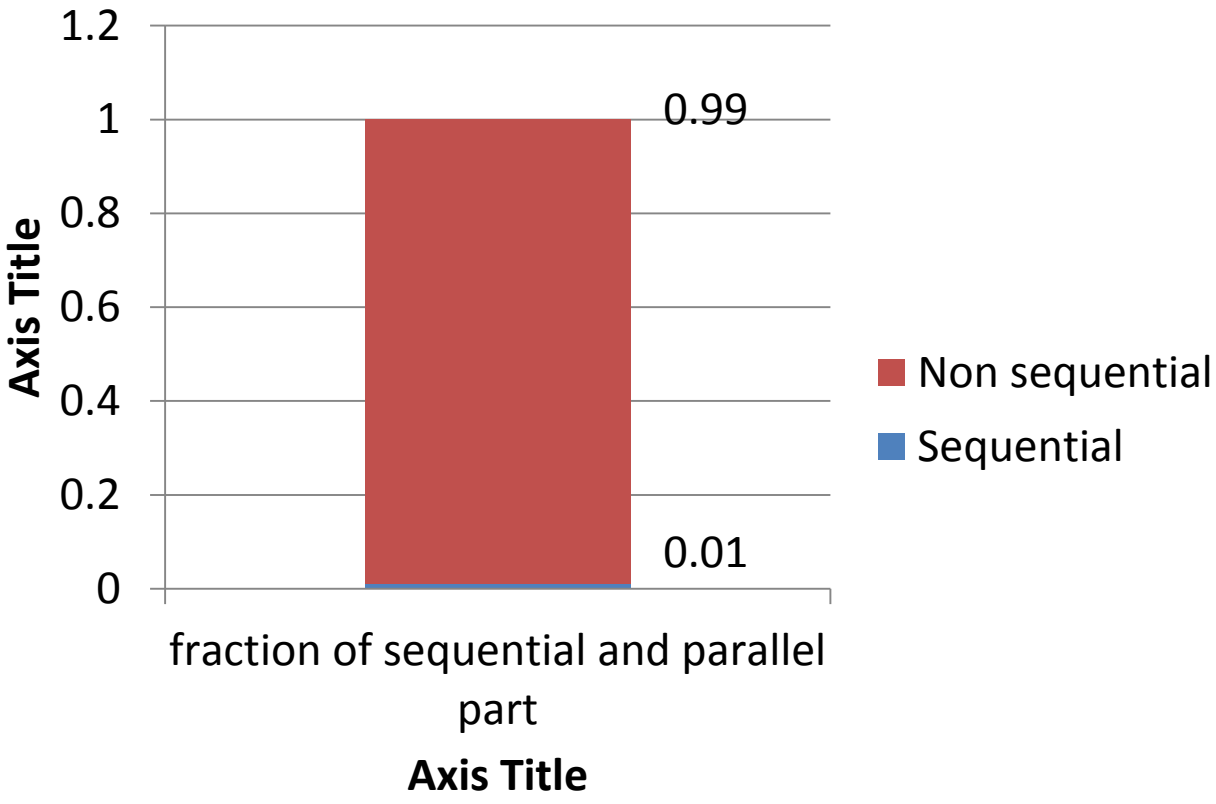
Parallelism

Speed up time in log scale



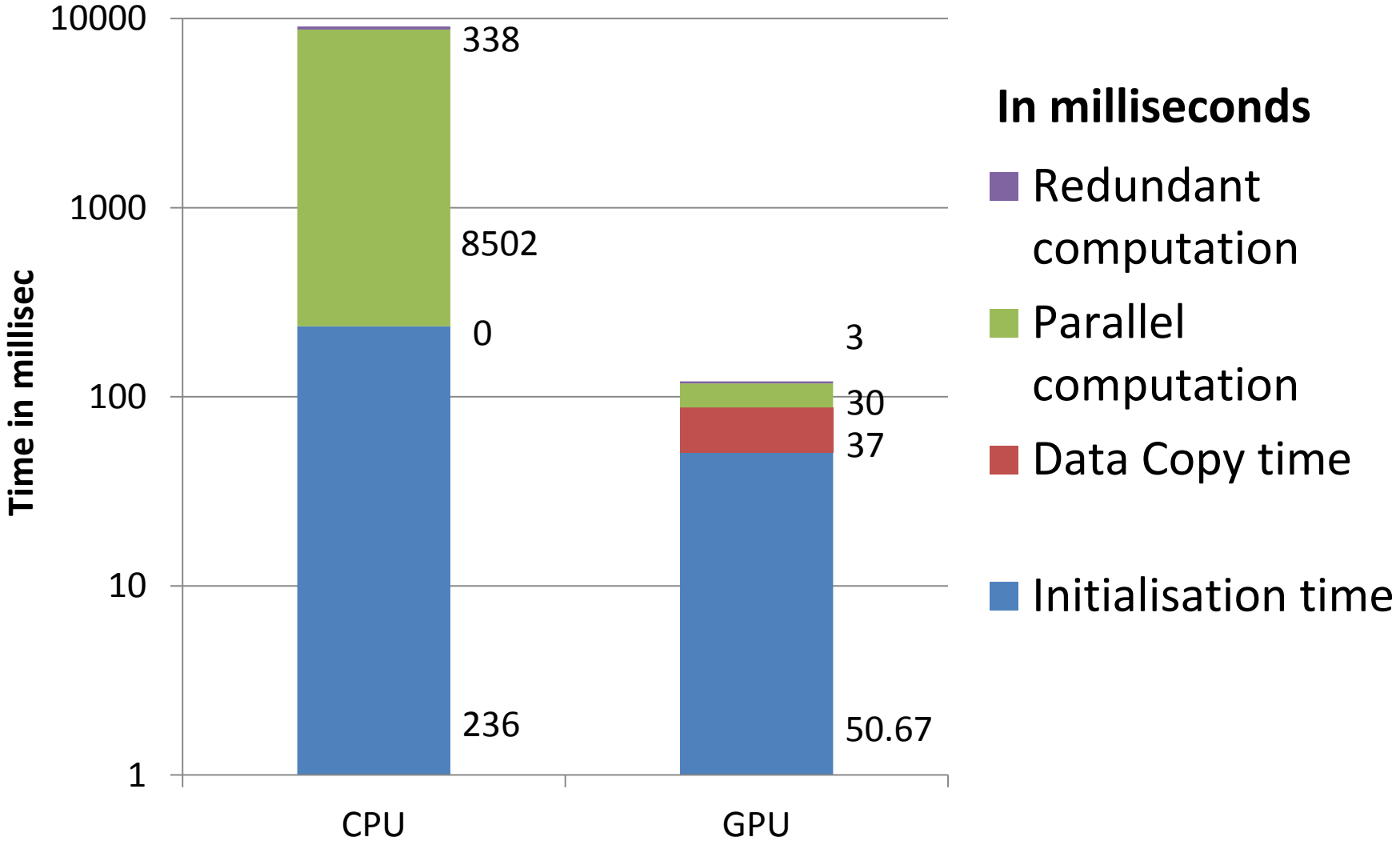
Parallelism

Amdahl's law



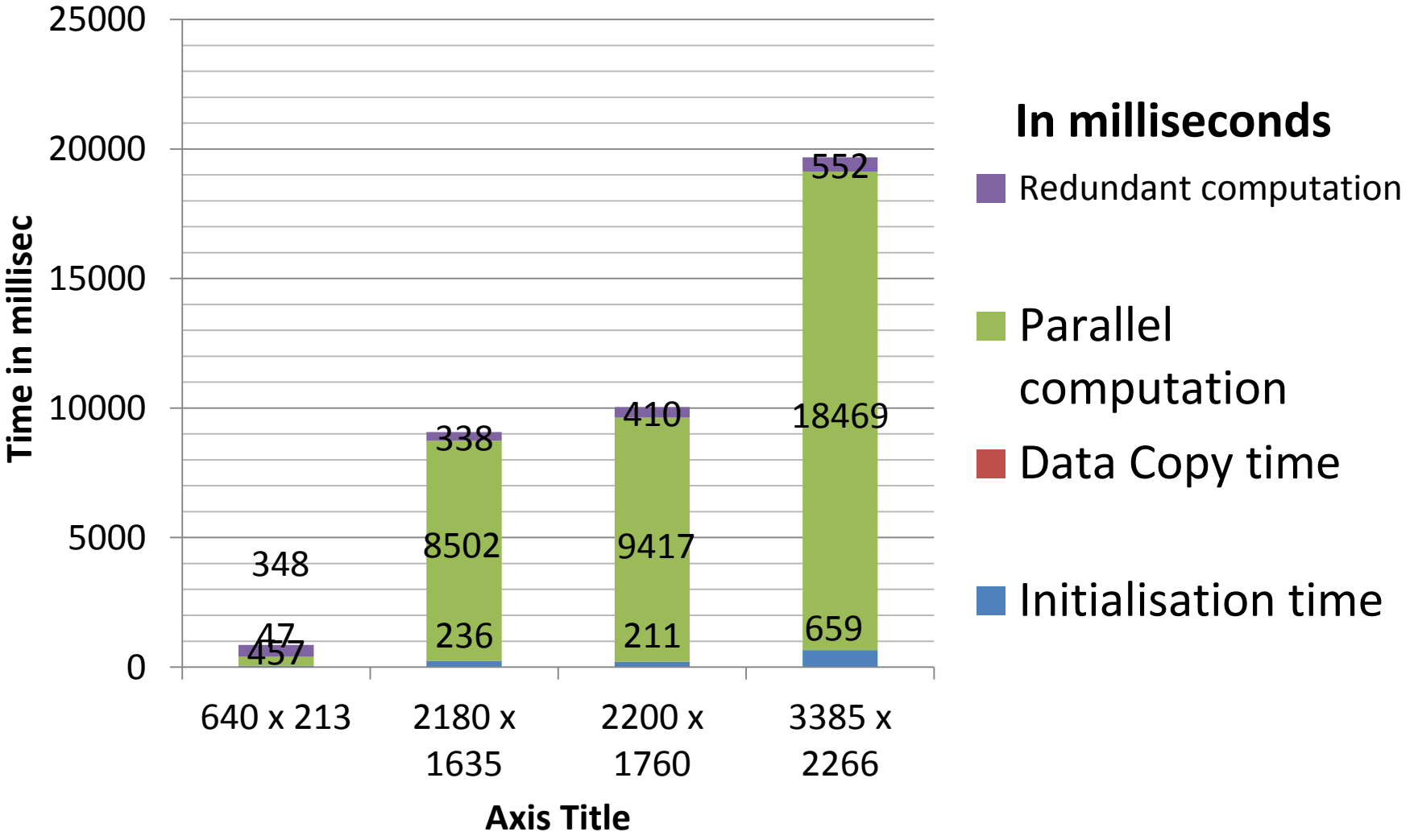
Performance comparison

CPU Vs GPU



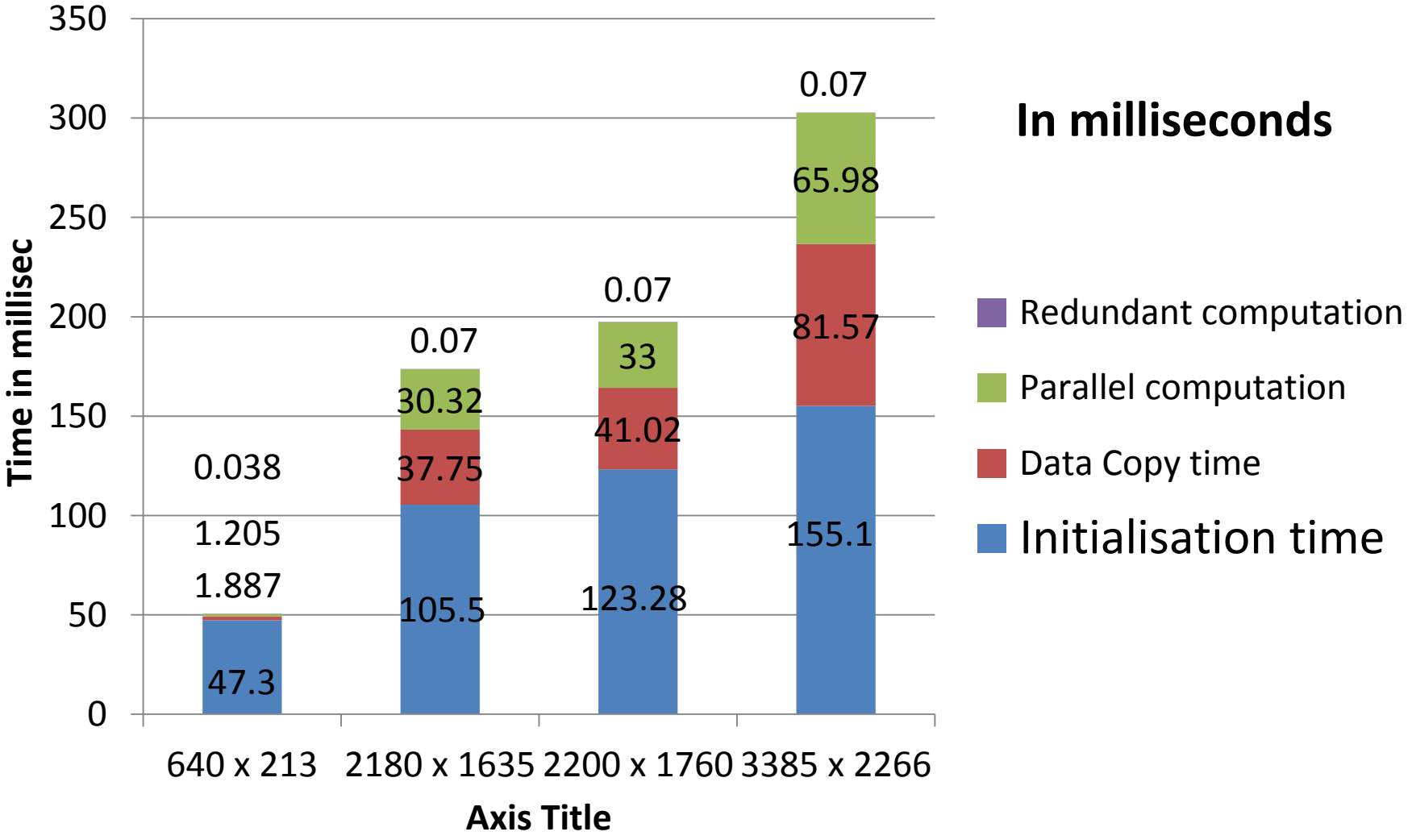
Performance comparison

- CPU execution time at various size



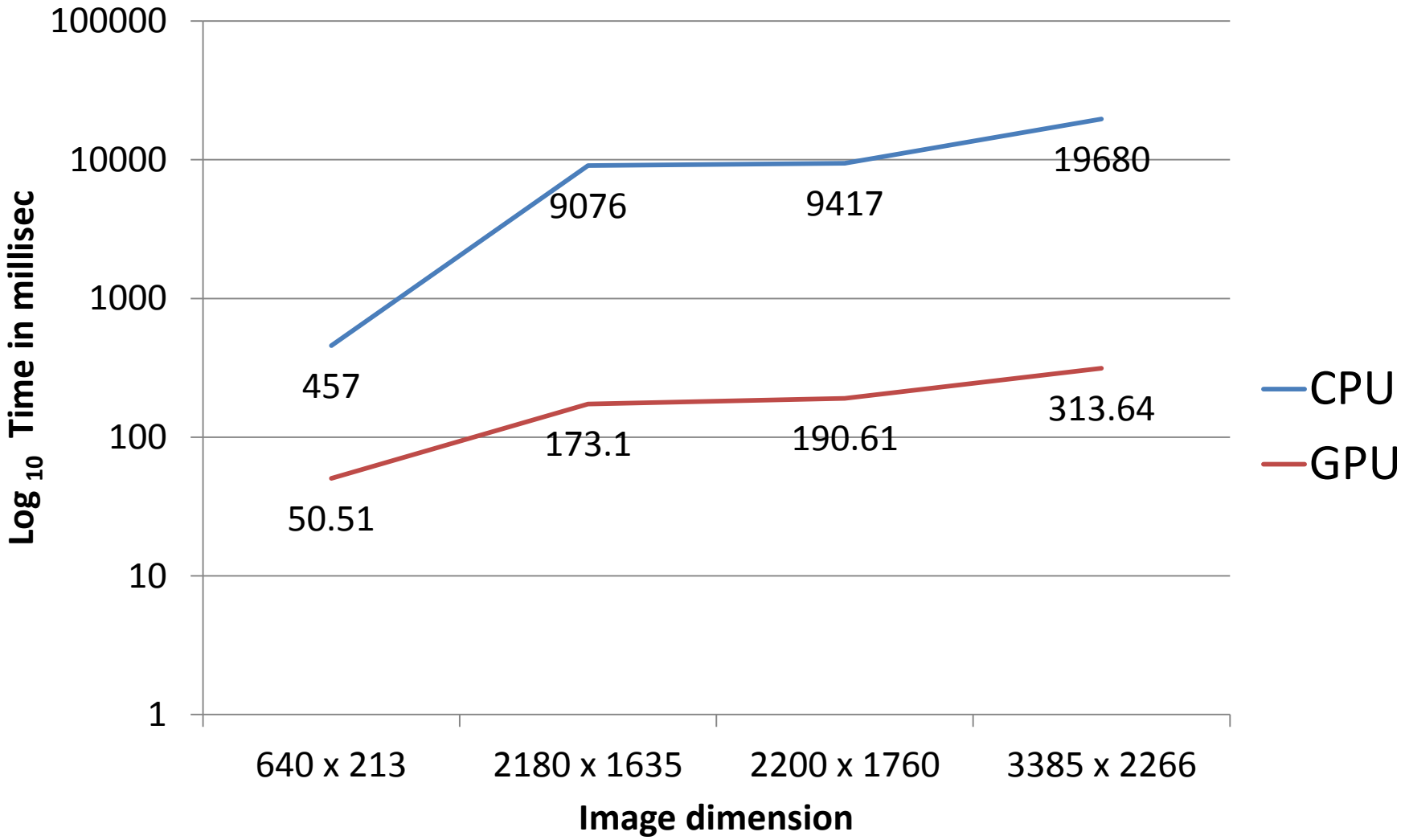
Performance comparison

- GPU execution time at various size



Overheads

- CPU Vs GPU execution time at various size



Demo