# Parallel Kernel Execution on GPUs (PKEG)

By Sahil Gandhi and Matthew Wong

# Table Of Contents

# Motivation

- GPUs have more resources
- Not all layers take up the entire GPU
  - Not enough computation to fill all resources
  - Unnecessary overhead to fill all resources
- What if we could execute kernels in parallel?
  - Supported since CUDA compute 2.0

# Related Work

- In 2010 Nvidia introduced Concurrent Kernels
  - Fermi Architecture
  - Fermi supported 16 maximum concurrent kernels (in theory)
    - 4 in reality
- Today Compute 7.0, can do up to 128 concurrent kernels
- https://ieeexplore.ieee.org/abstract/document/5999803 <-- shared data between kernels
- https://ieeexplore.ieee.org/abstract/document/6319193 <-- proper ordering of kernels

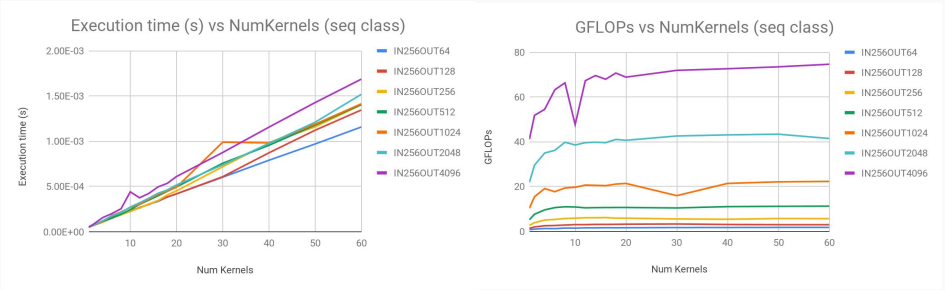
# Methodology

- Create a set of kernels
- Profile GPU utilization
- Compare sequential execution and concurrent execution
- Compare different sizes of layers/problems
- Mix different kernel functions together

# CUDA Streams

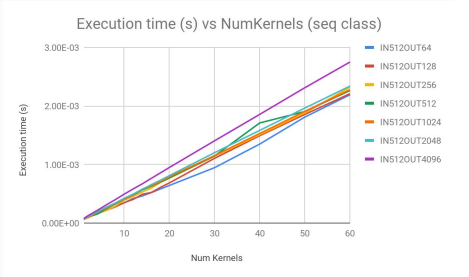
- cudaStream_t streams[ NUM_STREAMS ]
- cudaStreamCreate(&(streams[i]))
- kernel_function<<<dimGrid, dimBlock, 256, streams[i]>>>
  - Dimension and Size of Grid
  - Dimension and Size of each Block
  - Number of bytes of shared memory per Block
  - Stream ID object (defaults to 0)

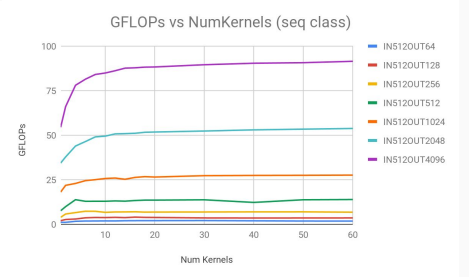
# Evaluation - Sequential Classifier





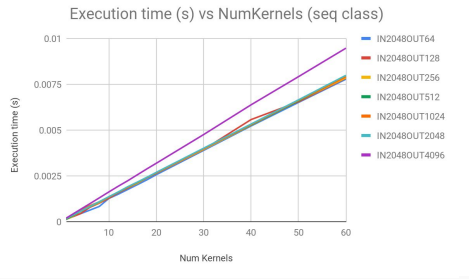


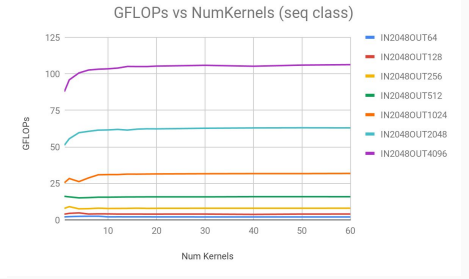




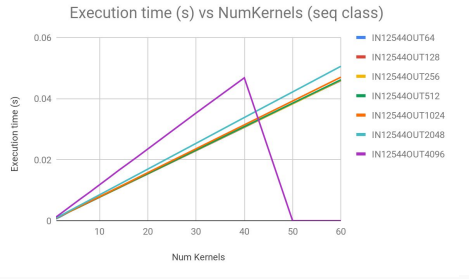


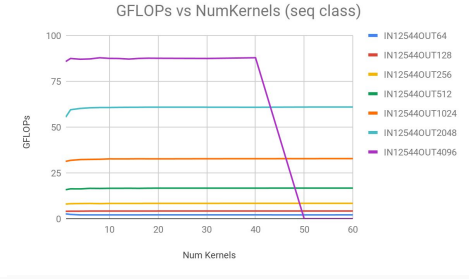



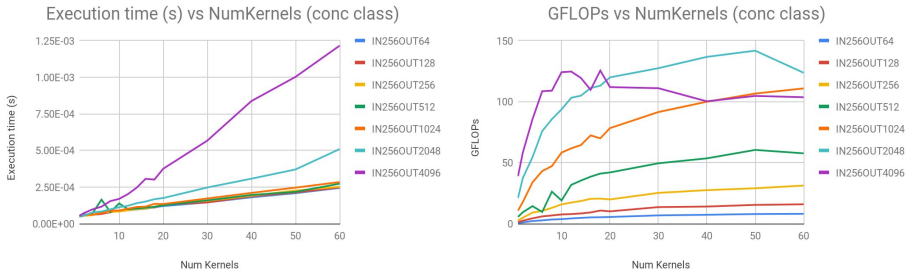




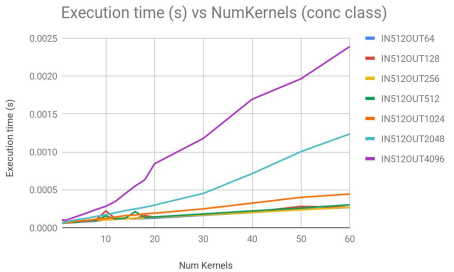



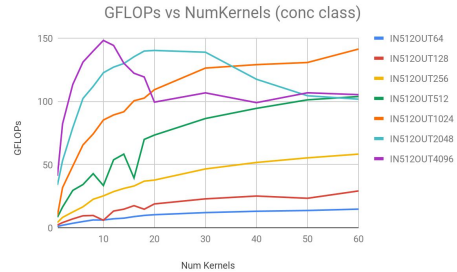
# Evaluation - Concurrent Classifier

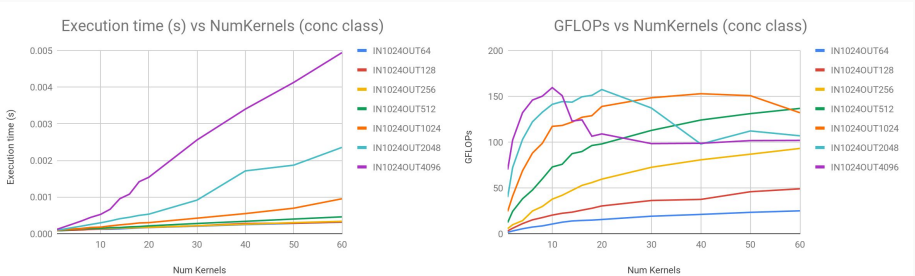






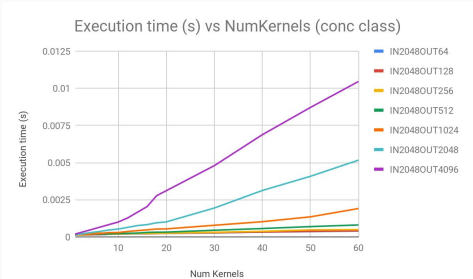


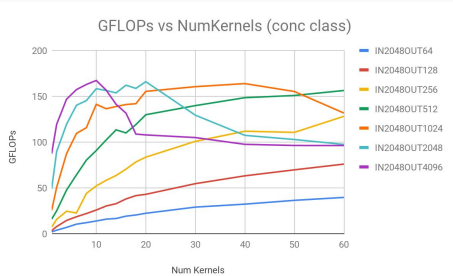



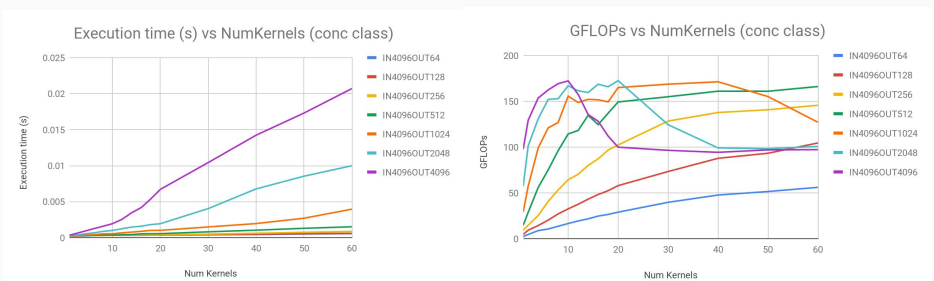







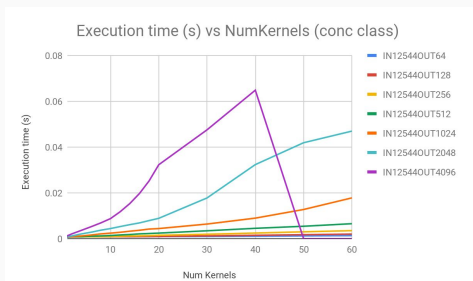


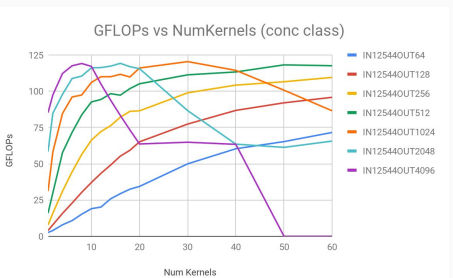



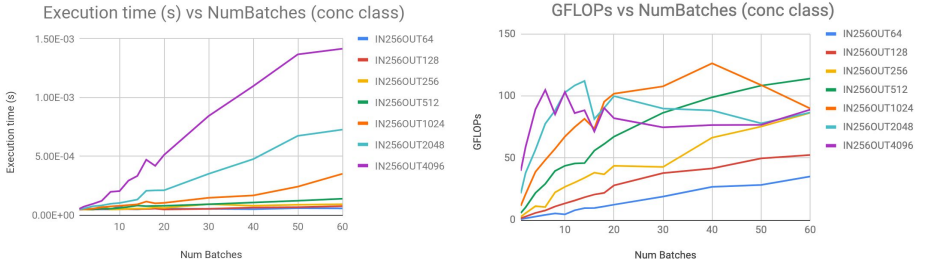






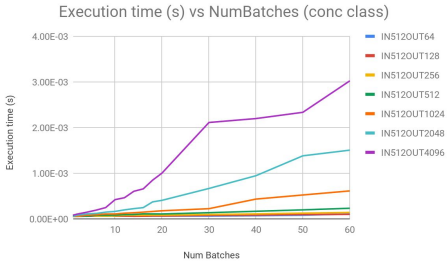



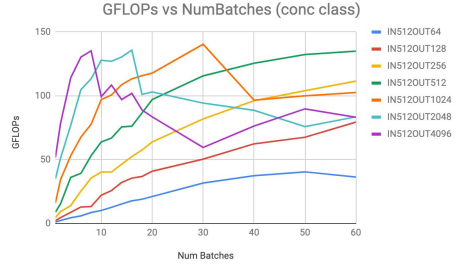
# Evaluation - Batched Classifier

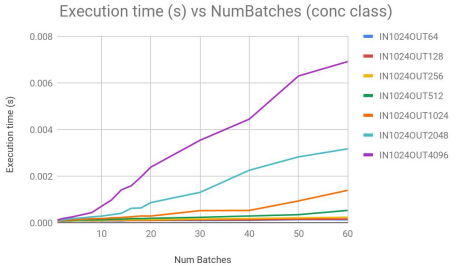


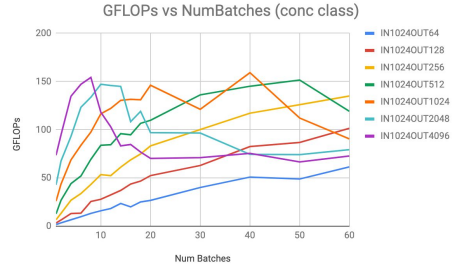


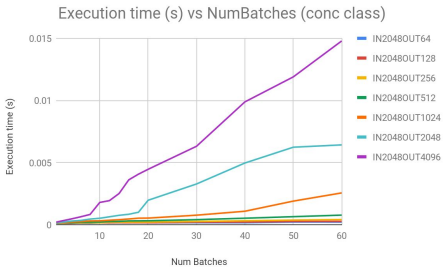


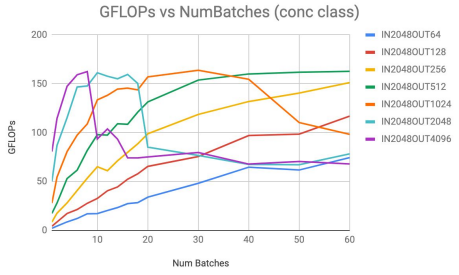


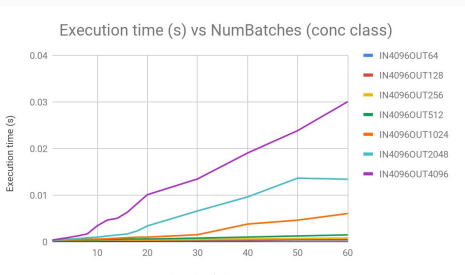



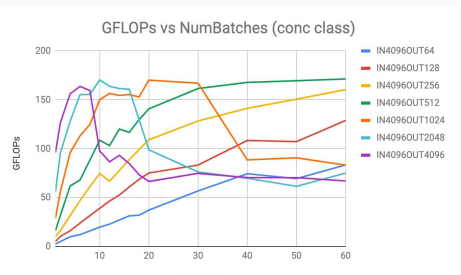



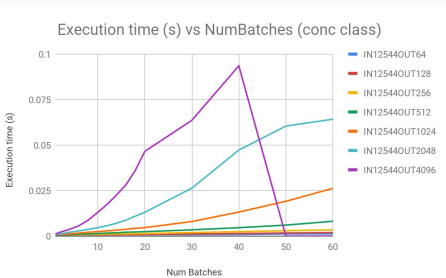



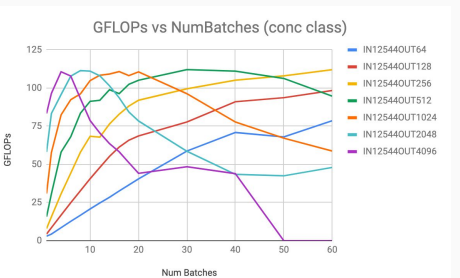



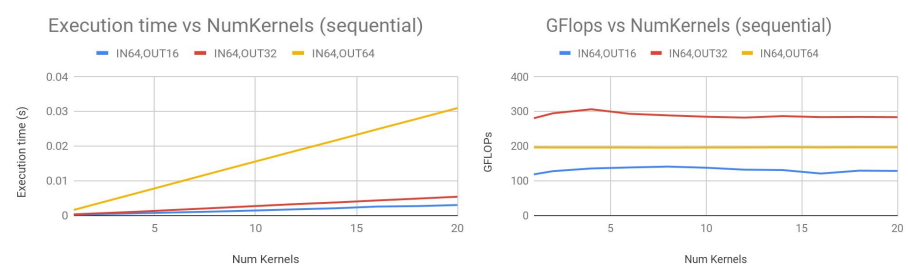



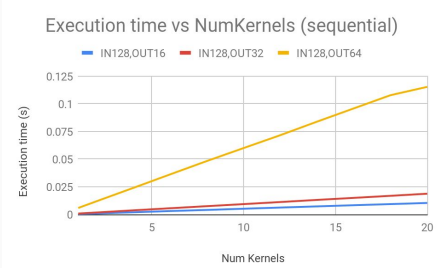



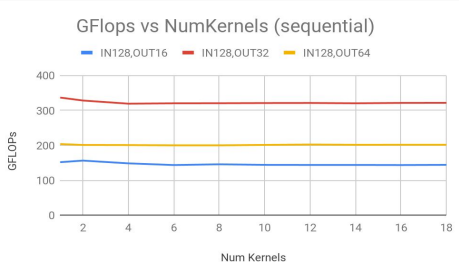
# Evaluation - Sequential Convolution

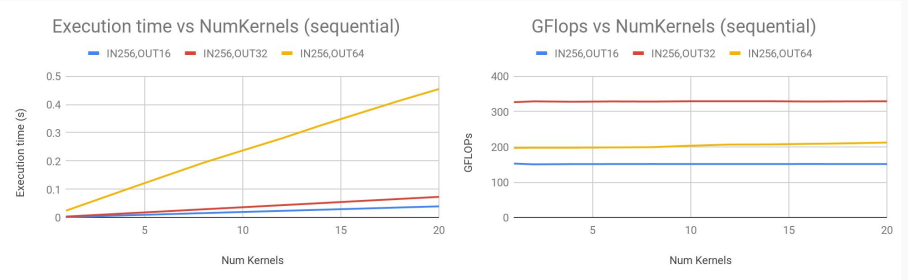






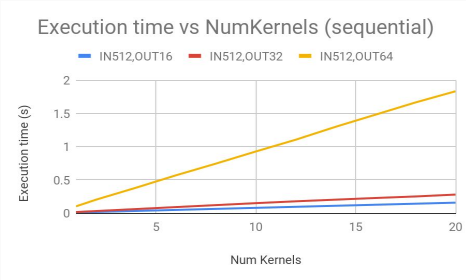


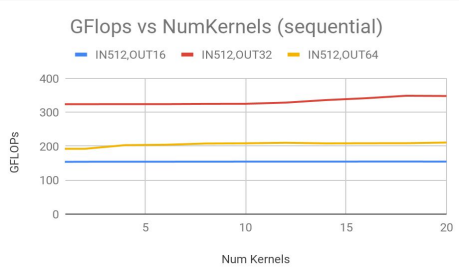



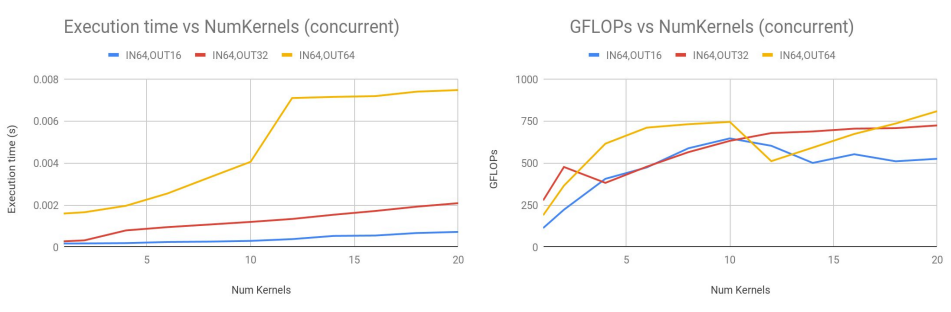






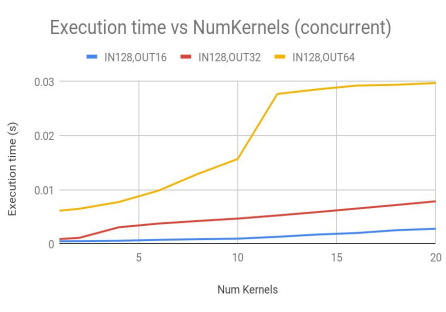



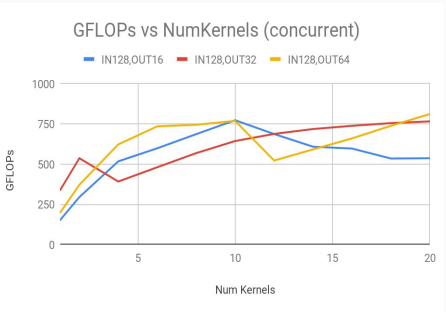
# Evaluation - Concurrent Convolution

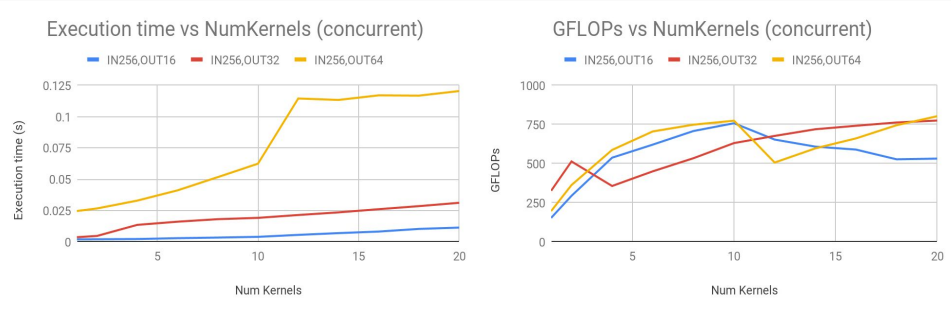






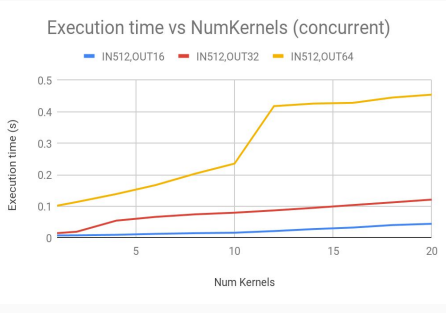


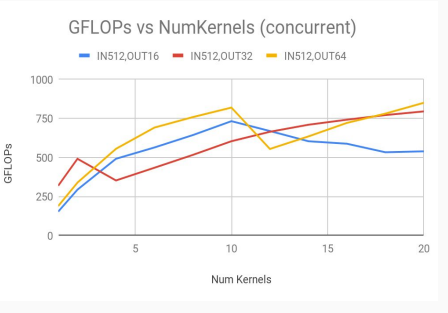







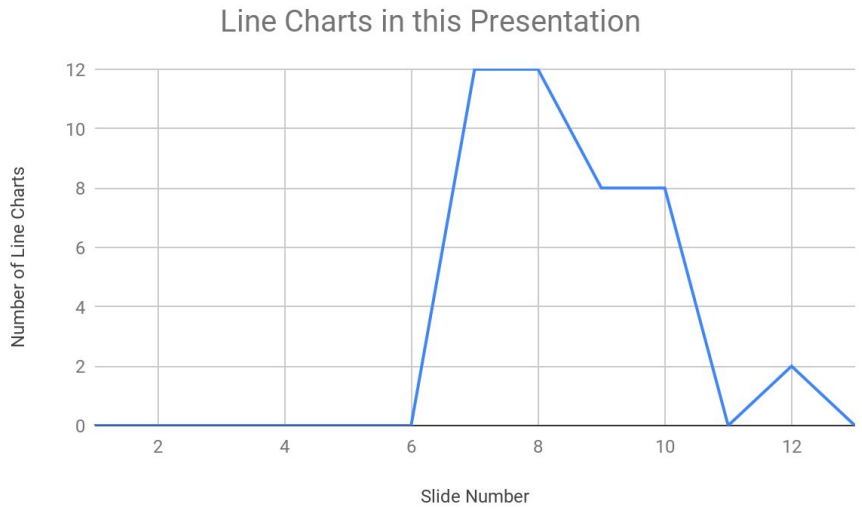


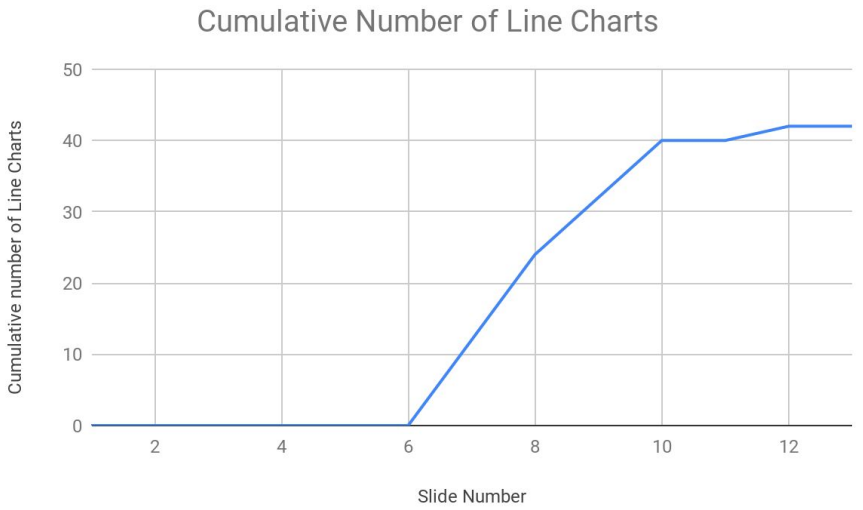



# Conclusion

- Parallel execution helps!
- Mysterious plateau though (perhaps more efficient context switching)
  - Or full switching (ie swap 10 “batches” at once instead of 10 and 3 being rotated around all the time)
- Larger GPUs can do more computation
  - How about smaller ones though?
    - We will find out next week!

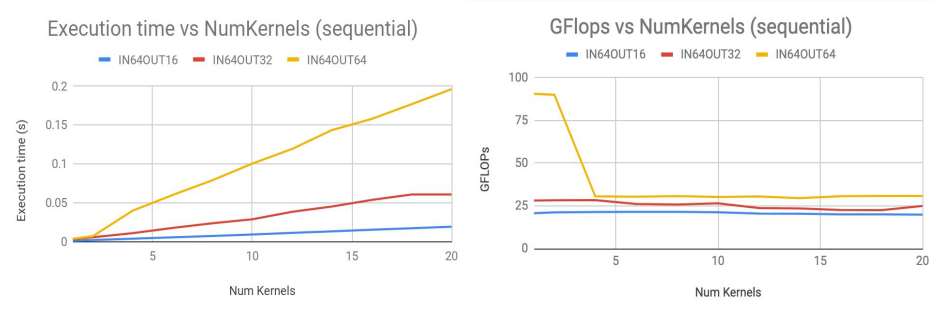
# Too Many Line Charts



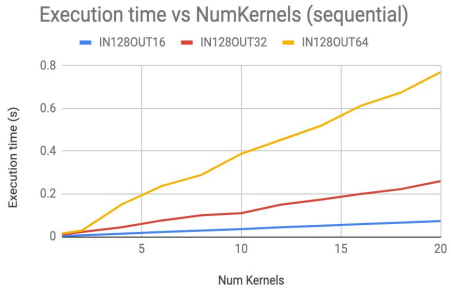
# Questions?

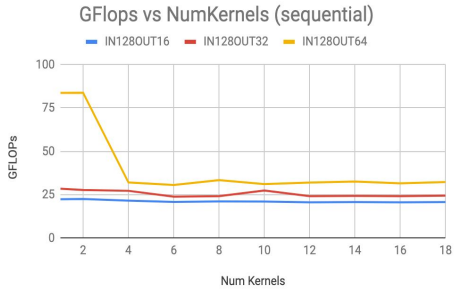
# Evaluation - Sequential Conv 2

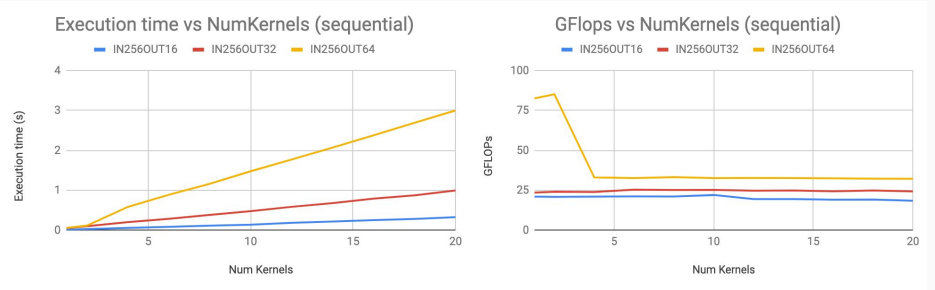






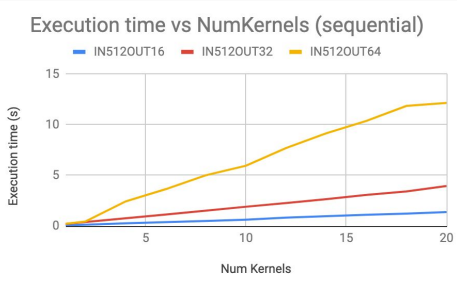


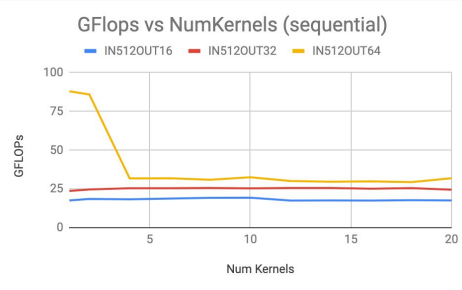



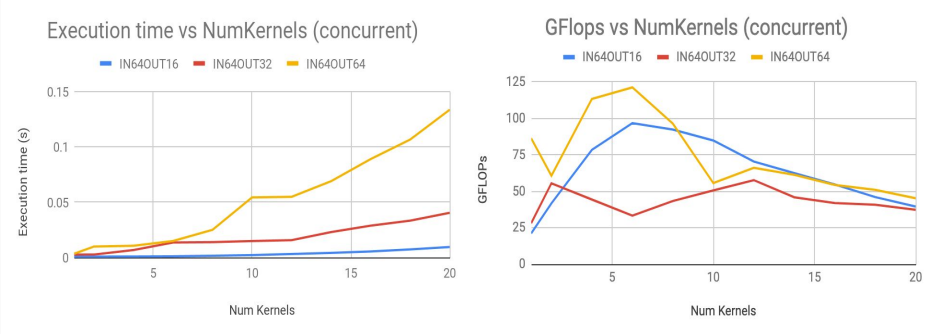






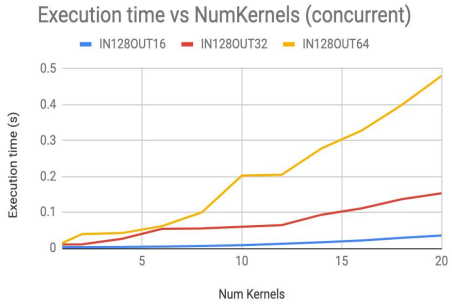



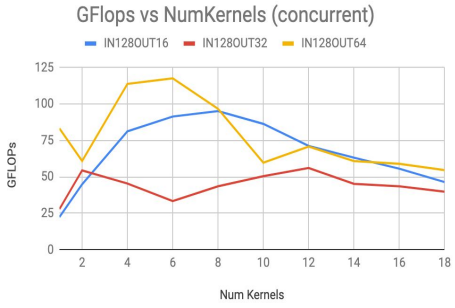
# Evaluation - Concurrent Conv 2

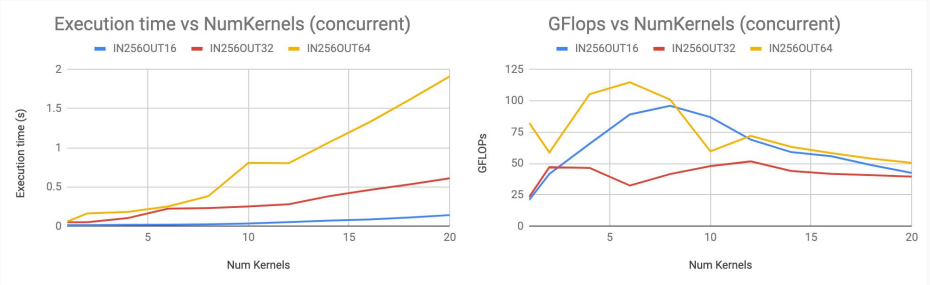






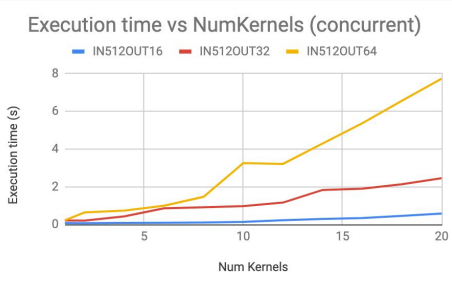


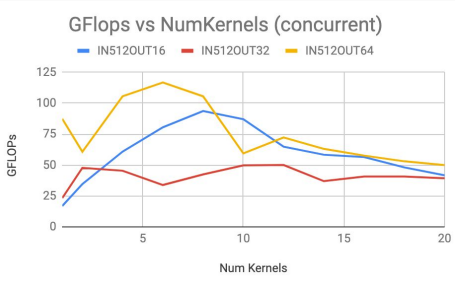



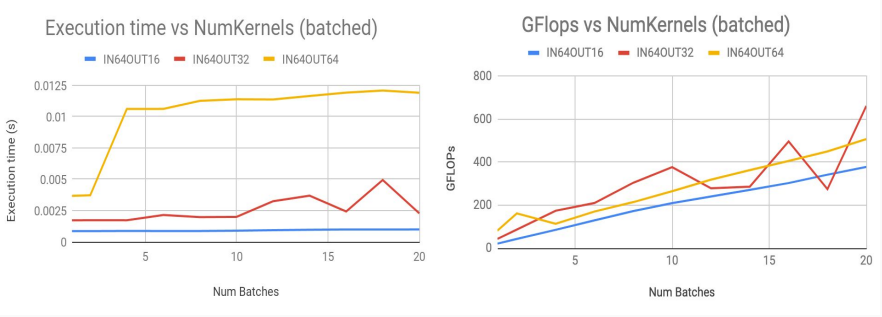






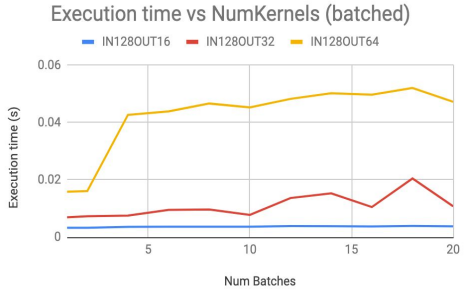



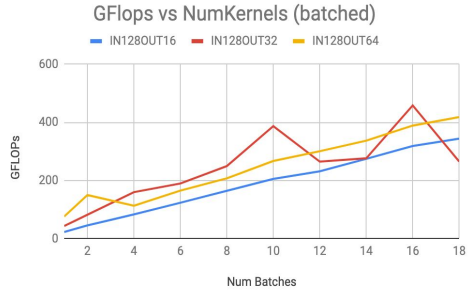
# Evaluation - Batched Conv2

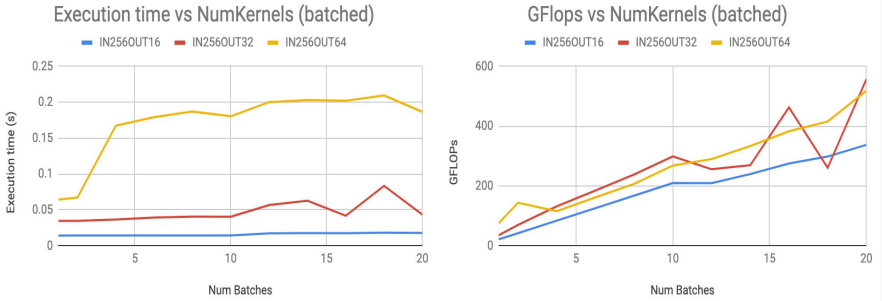






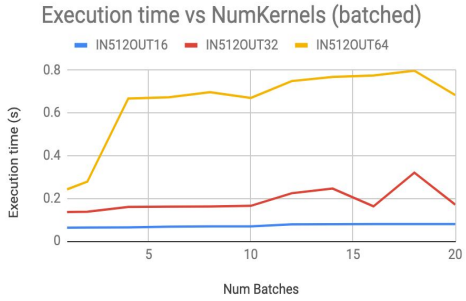


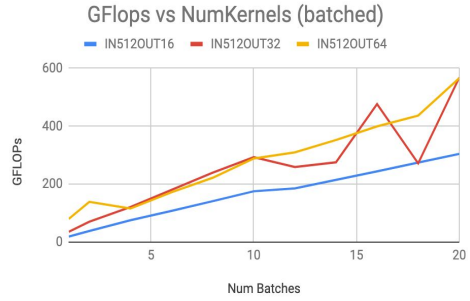





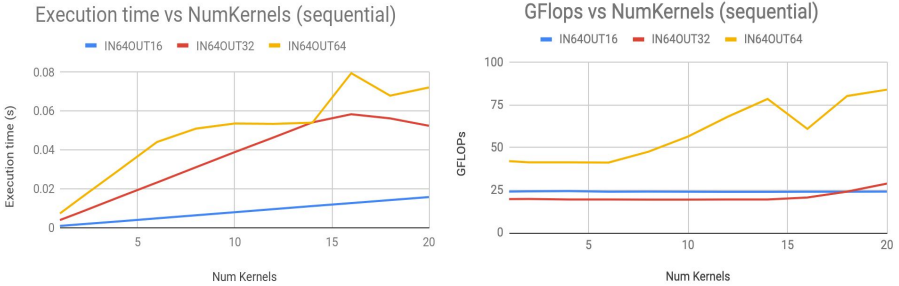




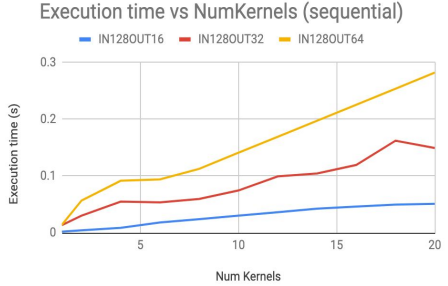
# AWS M60 GPU Results

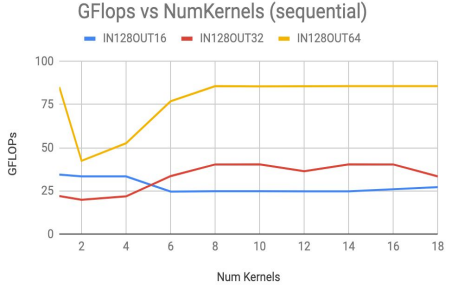
# M60 Conv Sequential

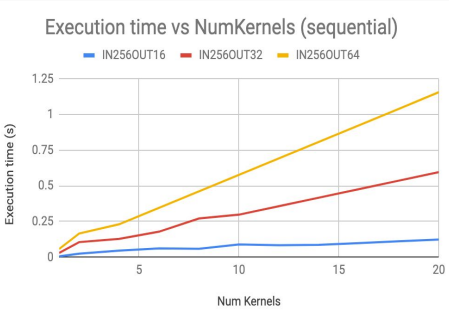


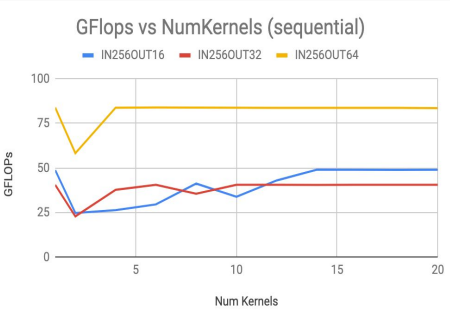


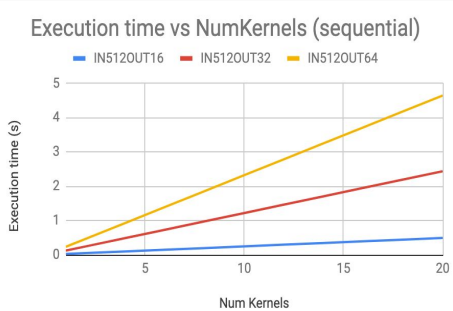


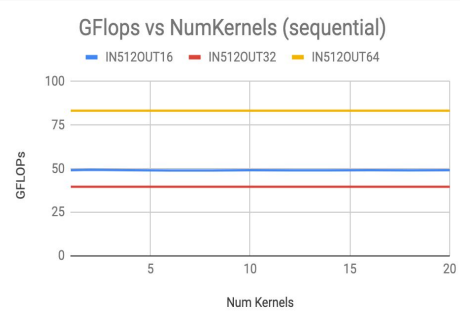


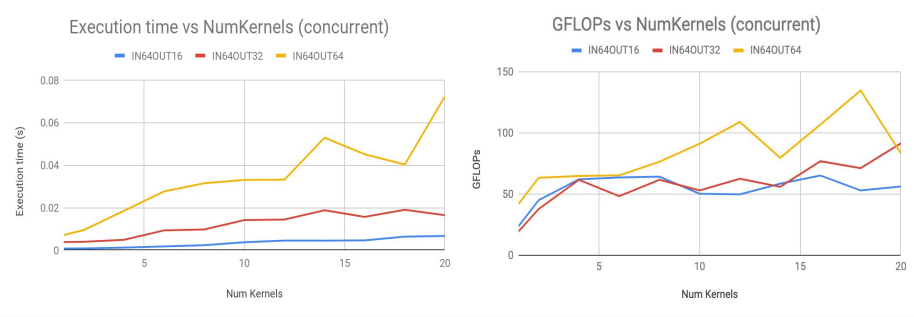



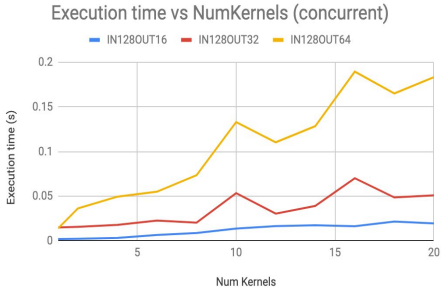



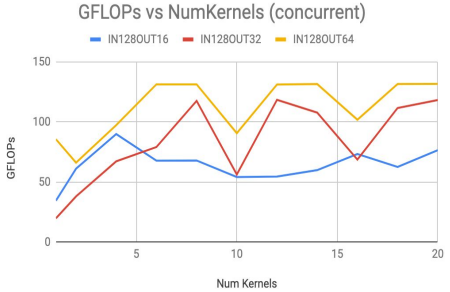
# M60 Conv Concurrent

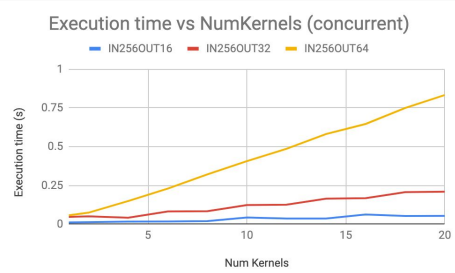


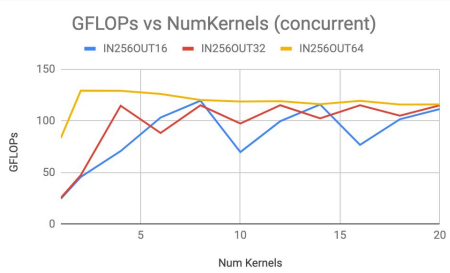


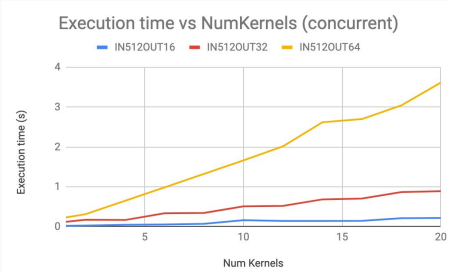


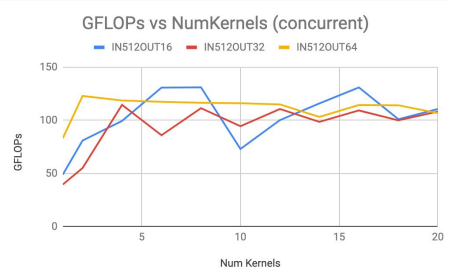


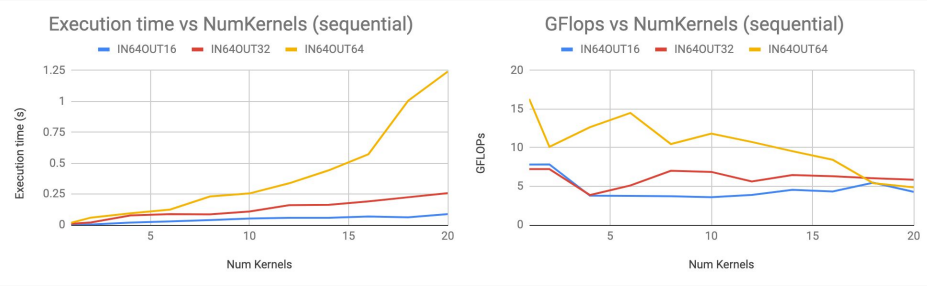



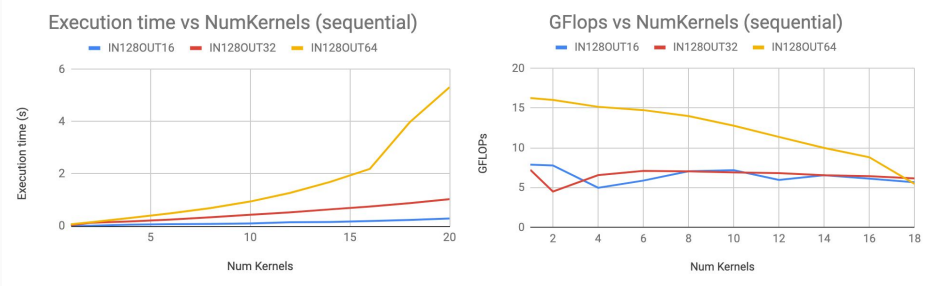



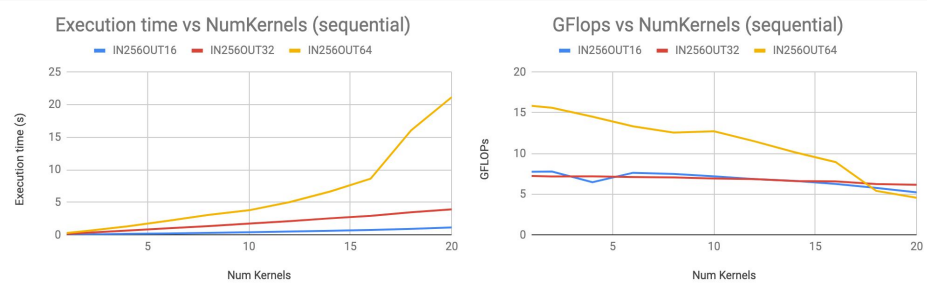



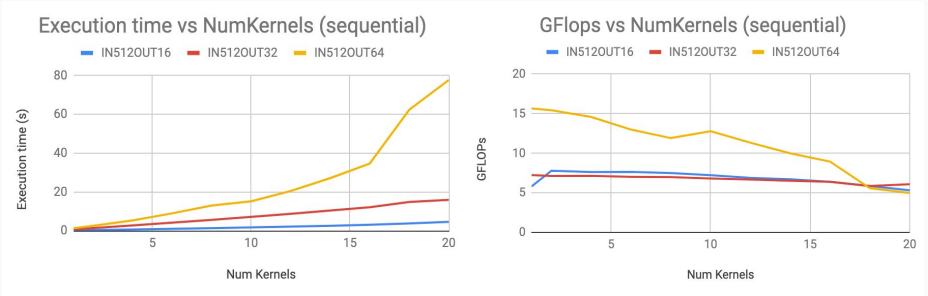



# M60 Conv2 Sequential

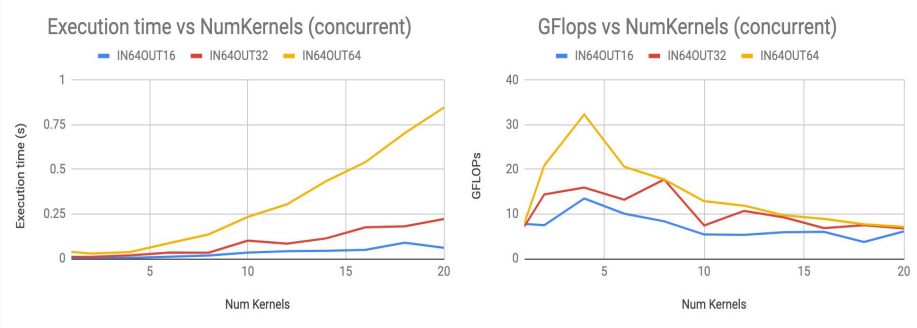


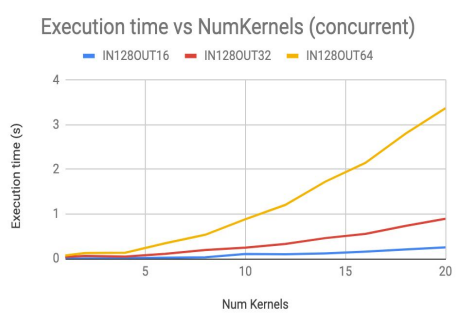



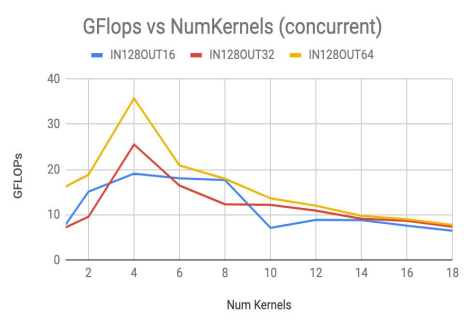
# M60 Conv2 Concurrent

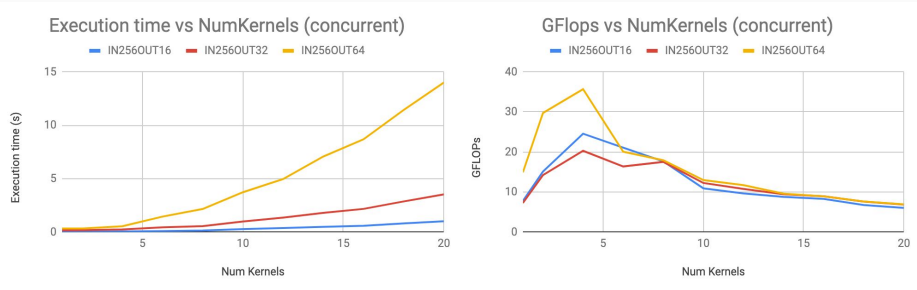






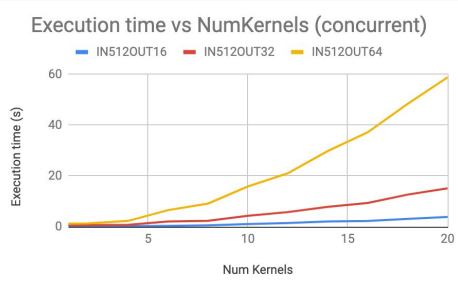


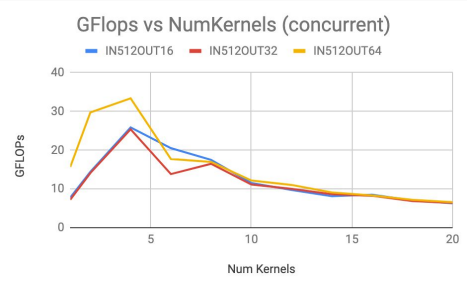



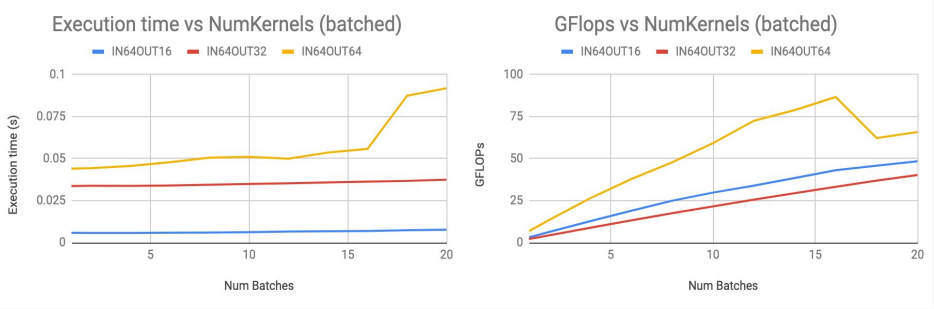






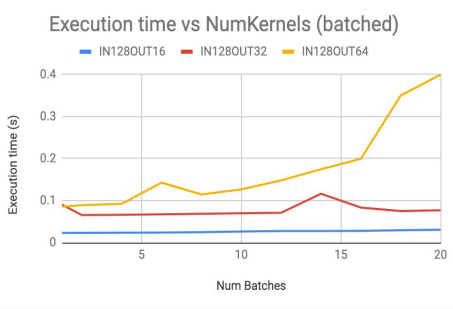



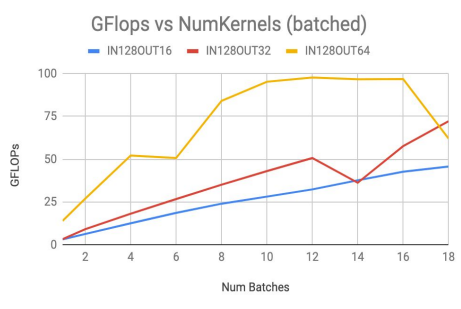
# M60 Conv2 Batched

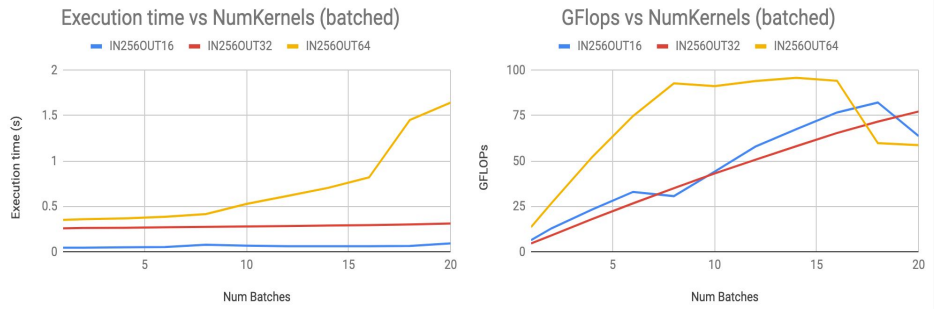






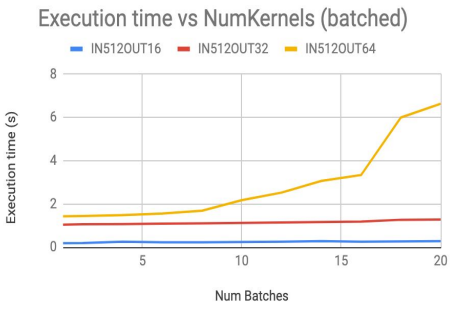


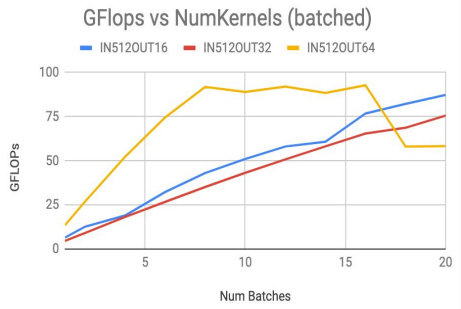



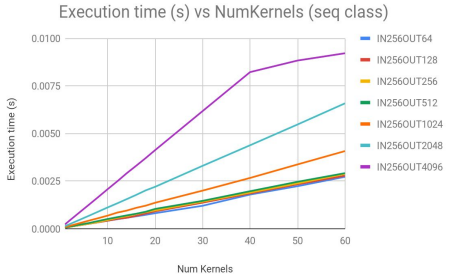


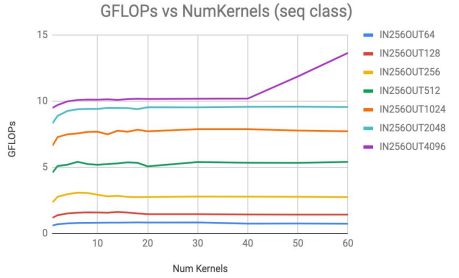


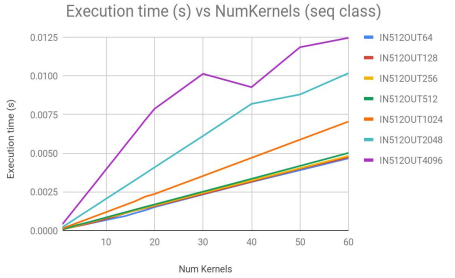


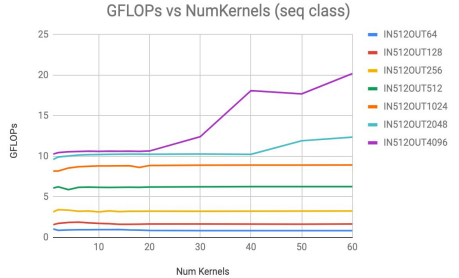



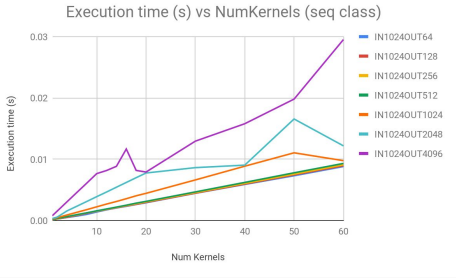
# M60 Classifier Sequential

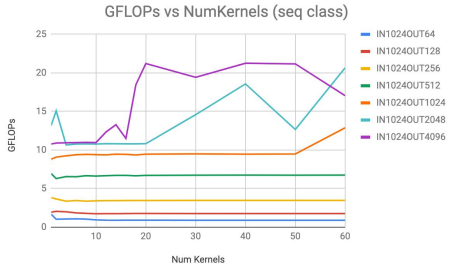


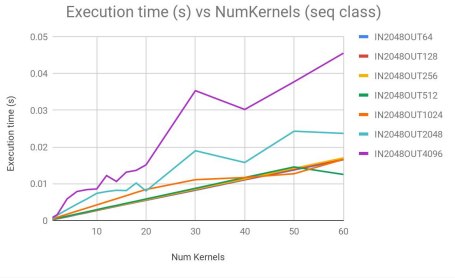



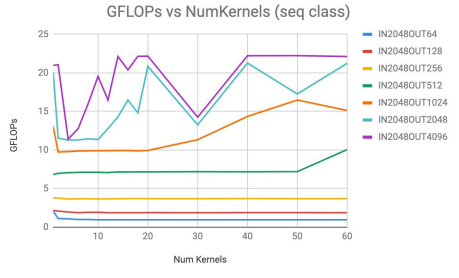



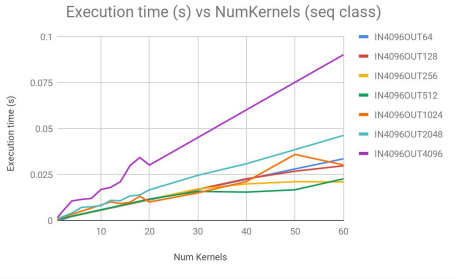



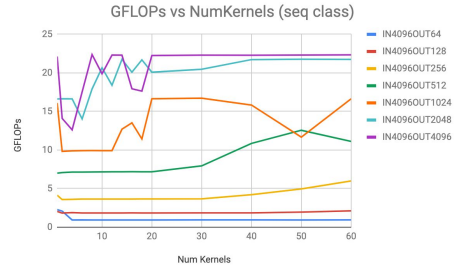



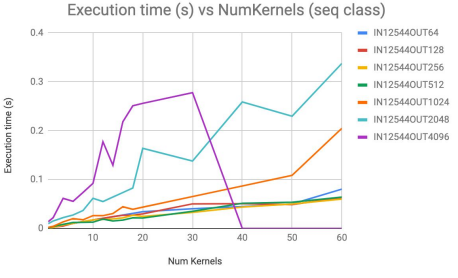



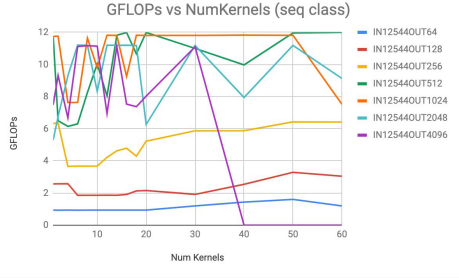



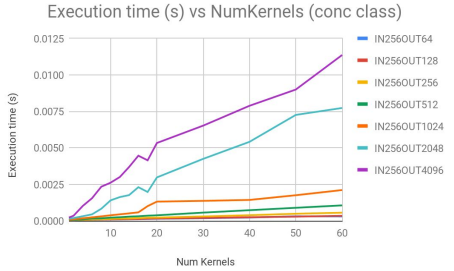



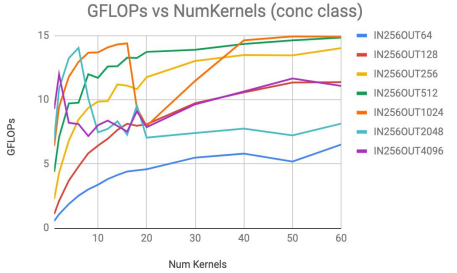



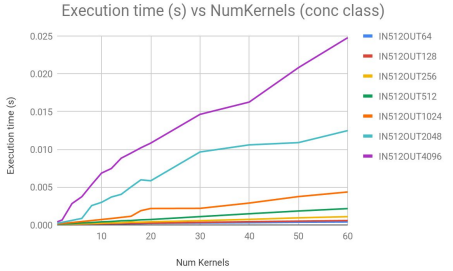



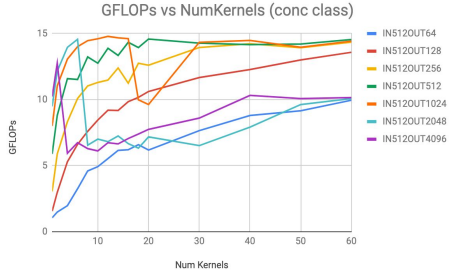



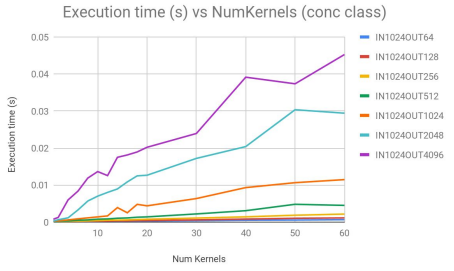
# M60 Classifier Concurrent

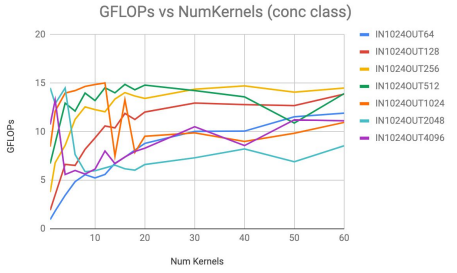


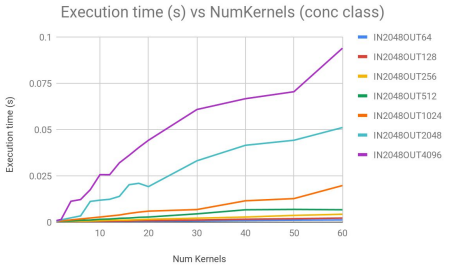



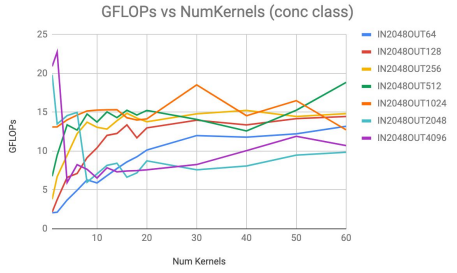



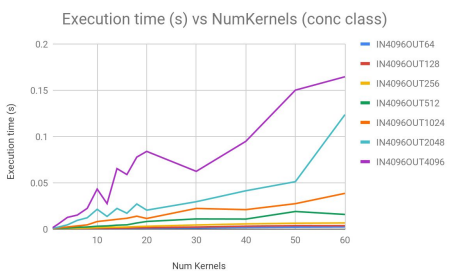



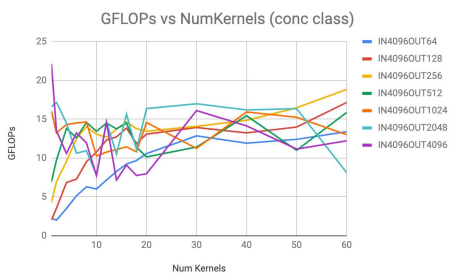



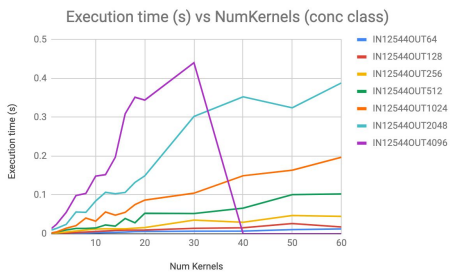



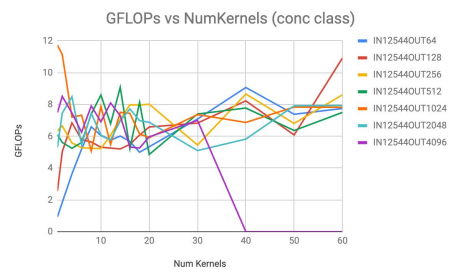



# M60 Classifier Batched