Some thoughts on CNNs real-time execution on NVIDIA GPUs

In ML-RT Workshop: Towards a research agenda for learning-enabled safety-critical real-time systems

Mourad Dridi [1] Joshua Dumont [2] Yasmina Abdeddaïm [1]

[1]Univ Gustave Eiffel, CNRS, LIGM, France

[2]Univ Gustave Eiffel, ESIEE Paris, France

July 9, 2024

General motivation and problem

1. **Critical** embedded systems are nowadays required to incorporate artificial intelligence (AI) functionalities.
2. Frameworks have been proposed to optimize performance of embedded AI algorithms in terms of **average computing time**.
3. Frameworks use **high level languages**.
4. In order to be **certified**, critical systems must guarantee a **temporal determinism** that is not guaranteed by the average temporal behavior of the system.

Problem to solve: How to **implement AI algorithms** in critical embedded systems with **real-time constraints**.

A more specif problem

- **Different classes of AI algorithms have different characteristics:** A general approach for real-time execution problem of AI algorithms is likely to be complicated.
- We propose to handle AI algorithms according to the class of AI algorithms to which they belong and the platform where they are executed.

In this presentation:

- **AI algorithms:** convolutional neuronal networks (**CNNs**) during the **inference phase**.
- **Platform:** NVIDIA GPUs.

A more specif problem

- **Different classes of AI algorithms have different characteristics:** A general approach for real-time execution problem of AI algorithms is likely to be complicated.
- We propose to handle AI algorithms according to the class of AI algorithms to which they belong and the platform where they are executed.

In this presentation:

- **AI algorithms:** convolutional neuronal networks (**CNNs**) during the **inference phase**.
- **Platform:** NVIDIA GPUs.

Research directions

We identify 3 research direction:

1. Task models for multiple CNNs with real-time constraints executed on NVIDIA GPUs.
2. Benchmarks for real-time CNNs.
3. Real-time scheduling problems with different granularity.

Research directions

We identify 3 research direction:

1. Task models for multiple CNNs with real-time constraints executed on NVIDIA GPUs.
2. Benchmarks for real-time CNNs.
3. Real-time scheduling problems with different granularity.

Real-time CNN task model

The task model should take into account:

- CNN characteristics
- GPU characteristics
- Real-time constraints

Convolutional neural networks (CNNs) characteristics

- A convolutional neural network (CNN) is a type of **acyclic** (feed-forward) artificial neural network.

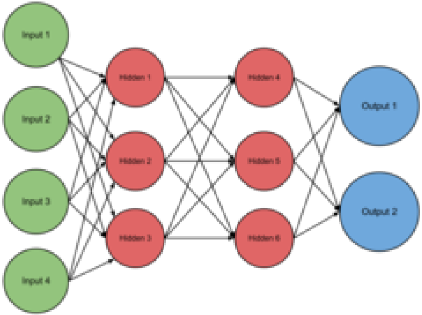


Figure: convolutional neural network (CNN) [1]

- Consist of a layered stack of perceptrons (binary classifier).
- Convolutional neural networks are widely used in image and video recognition.

[1]Wikipedia

NVIDIA GPU architecture characteristics

- NVIDIA GPUs consist of a number of Streaming Multiprocessors (SMs).
- SMs use the SIMD (Single Instruction Multiple Data) processing method: multiple processing elements perform the same operation on different data at the same time.
- SMs are grouped into Tensor Processing Clusters (TPCs).

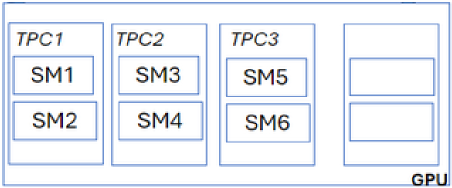
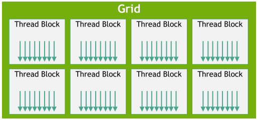


Figure: NVIDIA GPU

CUDA: Kernel, Thread and Block

1. CUDA is a parallel programming library developed by NVIDIA.
2. The code executed by the GPU is referred to as a **kernel**.
3. A **kernel** is made up of **blocks**, which are collections of individual threads.
4. **Threads** run in parallel and operate on different subsets of data.
5. Each block is executed by one SM.
6. An SM can execute several blocks concurrently.
7. CUDA architecture limits the numbers of threads per block.



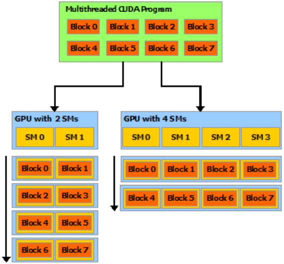


Figure: Threads, Blocks and SM [2]

[2] Nvidia web site

CUDA: Kernel, Thread and Block

A kernel is described by its **number of blocks**, the **number of threads per block**, the **shared memory between blocks** and the **stream used to execute the kernel**. Kernels assigned to different streams can be executed in parallel. A **priority** can be assigned to a stream.

$$kernel <<< nbBlocks, blockSize, shared\_mem*, stream* >>>$$

```
// Kernel definition
__global__ void VecAdd(float* A, float* B, float* C)
{
    int i = threadIdx.x;
    C[i] = A[i] + B[i];
}
MatAdd
int main()
{
    ...
    // Kernel invocation with N threads
    VecAdd<<<1, N>>>(A, B, C);
    ...
}
```

Real-time CNN task model: The structure

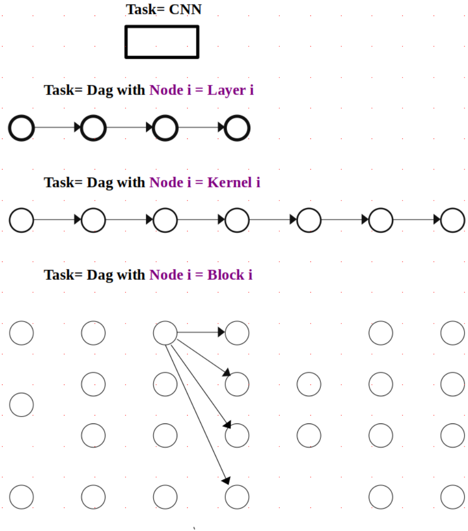
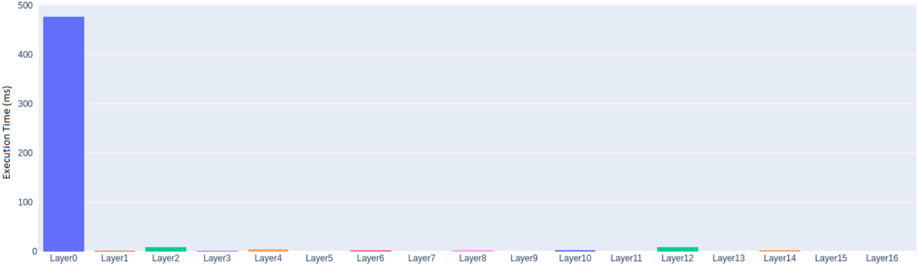


Figure: Task models at different granularity

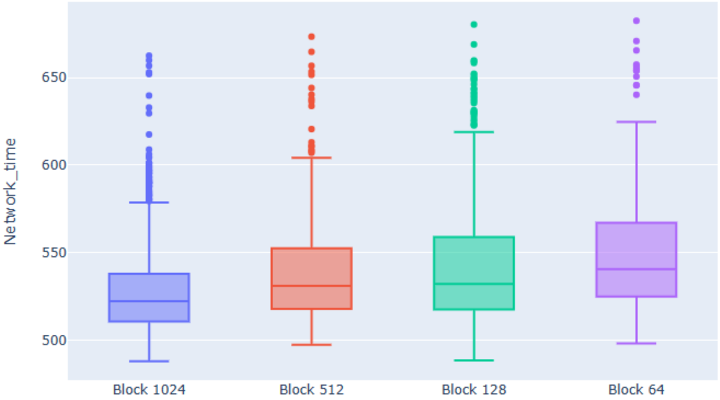
Real-time CNN task model: Execution time

- Yolov3_tiny model run on the Nvidia Jetson TX2 card.
- Execution time per layer.



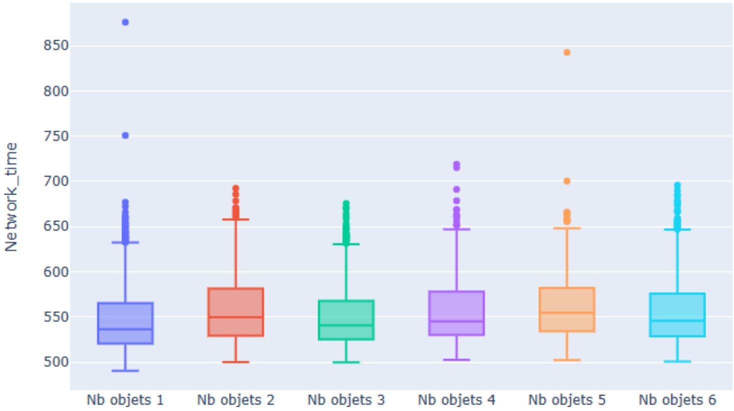
Real-time CNN task model: Execution time

- Yolov3_tiny model run on the Nvidia Jetson TX2 card.
- 100 measurements were taken for different block sizes of kernel.



Real-time CNN task model: Execution time

- Yolov3_tiny model run on the Nvidia Jetson TX2 card.
- 100 measurements were taken for different numbers of objects to detect.



Research directions

We identify 3 research direction:

1. A task model for multiple CNNs with real-time constraints executed on NVIDIA GPUs.
2. Build benchmarks for real-time CNNs.
3. Real-time scheduling problems with different granularity.

Benchmarks for Real-time CNNs

1. Classic benchmarks, such as Mälardalen and TacleBench benchmarks, are commonly used in real-time systems in evaluation.
2. There is a need for specific benchmarks to assess the performance of CNN inference algorithms.
3. **Darknet** is an open source framework written in C and CUDA designed to facilitate the rapid and efficient development of neural networks.
4. Darknet allows the construction and execution of **various trained types of neural networks**.
5. Not easy to use it for real-time execution of multiple CNNs.

Benchmarks for Real-time CNNs

1. Classic benchmarks, such as Mälardalen and TacleBench benchmarks, are commonly used in real-time systems in evaluation.
2. There is a need for specific benchmarks to assess the performance of CNN inference algorithms.
3. **Darknet** is an open source framework written in C and CUDA designed to facilitate the rapid and efficient development of neural networks.
4. Darknet allows the construction and execution of **various trained types of neural networks**.
5. Not easy to use it for real-time execution of multiple CNNs.

Benchmarks for Real-time CNNs

We started the **enhancement of Darknet** to build a new version for real-time CNN:

1. Real-time execution of **multiple CNNs**.
2. Collecting crucial information needed for the real- time scheduling:
    1. **Execution times** at different granularity: layer, kernel, block.
    2. **TPC** assignment. [3]
    3. **SM** identification.

```
159326 Block 3 is executed on SM 6 in kernel scale_bias_kernel - Execution time of a thread: 0.0000002569 seconds. idx_block =160982. idx_kernel : 68
159327 Block 1 is executed on SM 6 in kernel scale_bias_kernel - Execution time of a thread: 0.0000006523 seconds. idx_block =160983. idx_kernel : 68
159328 Block 4 is executed on SM 2 in kernel scale_bias_kernel - Execution time of a thread: 0.0000006854 seconds. idx_block =160984. idx_kernel : 68
159329 Block 6 is executed on SM 2 in kernel scale_bias_kernel - Execution time of a thread: 0.0000006931 seconds. idx_block =160985. idx_kernel : 68
159330 Block 5 is executed on SM 1 in kernel scale_bias_kernel - Execution time of a thread: 0.0000014346 seconds. idx_block =160986. idx_kernel : 68
159331 Block 7 is executed on SM 8 in kernel scale_bias_kernel - Execution time of a thread: 0.0000018869 seconds. idx_block =160987. idx_kernel : 68
159332 Block 9 is executed on SM 6 in kernel scale_bias_kernel - Execution time of a thread: 0.0000006323 seconds. idx_block =160988. idx_kernel : 68
159333 Block 8 is executed on SM 12 in kernel scale_bias_kernel - Execution time of a thread: 0.0000017954 seconds. idx_block =160989. idx_kernel : 68
159334 Block 11 is executed on SM 9 in kernel scale_bias_kernel - Execution time of a thread: 0.0000002331 seconds. idx_block =160990. idx_kernel : 68
159335 Block 10 is executed on SM 0 in kernel scale_bias_kernel - Execution time of a thread: 0.0000037762 seconds. idx_block =160991. idx_kernel : 68

159337 Start time of kernel scale_bias_kernel : 26627.819498624 seconds, idx_kernel = 69
159338 Block 13 is executed on SM 12 in kernel im2col_gpu_kernel - Execution time of a thread: 0.0000030385 seconds. idx_block =48384. idx_kernel : 36
159339 Block 34 is executed on SM 12 in kernel im2col_gpu_kernel - Execution time of a thread: 0.0000038408 seconds. idx_block =48395. idx_kernel : 36
159340 Block 7 is executed on SM 6 in kernel im2col_gpu_kernel - Execution time of a thread: 0.0000035738 seconds. idx_block =48402. idx_kernel : 36
159341 Block 12 is executed on SM 12 in kernel im2col_gpu_kernel - Execution time of a thread: 0.0000027415 seconds. idx_block =48381. idx_kernel : 36
159342 Block 31 is executed on SM 6 in kernel im2col_gpu_kernel - Execution time of a thread: 0.0000042538 seconds. idx_block =48408. idx_kernel : 36
```

Figure: Information collected from the enhanced Darknet

[3] Joshua Bakita and James H. Anderson, Hardware Compute Partitioning on NVIDIA GPUs, in Proceedings of the 2023 IEEE 29th Real-Time and Embedded Technology and Applications Symposium (RTAS), 2023

Research directions

We identify 3 research direction:

1. A task model for multiple CNNs with real-time constraints executed on NVIDIA GPUs.
2. Build benchmarks for real-time CNNs.
3. Real-time scheduling problems with different granularity.

Real-time scheduling problems

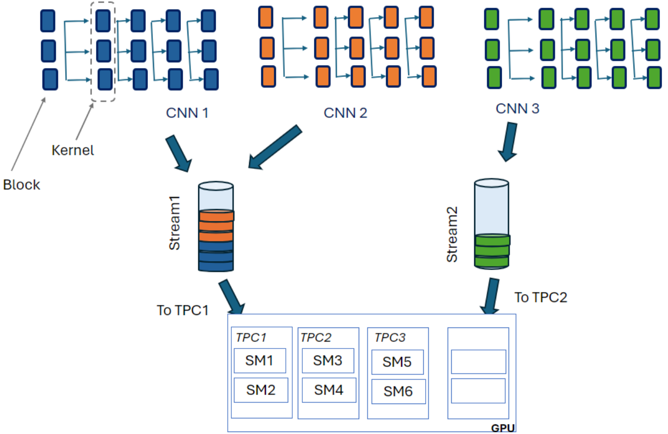
Several granularity of real-time scheduling problems to consider:

1. Real-time scheduling of blocks on streaming multiprocessors (SMs).
2. Assigning kernels to streams.
3. Assigning streams to Thread Processing Clusters (TPCs).
4. Choosing the block size.

Real-time scheduling problems

Several granularity of real-time scheduling problems to consider:

1. Real-time scheduling of blocks on streaming multiprocessors (SMs).
2. Assigning kernels to streams.
3. Assigning streams to Thread Processing Clusters (TPCs).
4. Choosing the block size.



Conclusion: Why this presentation ?

- Several research on real-time scheduling of AI algorithms already exist.
- There are also research more specific to the NVIDIA GPU.
- Experiments have already been carried out using Darknet.

However

- A common theoretical framework is lacking.
- Difficult to situate the different approaches in terms of the problems addressed.
- Difficult to compare the solutions proposed.

Conclusion: Why this presentation ?

- Several research on real-time scheduling of AI algorithms already exist.
- There are also research more specific to the NVIDIA GPU.
- Experiments have already been carried out using Darknet.

However

- A common theoretical framework is lacking.
- Difficult to situate the different approaches in terms of the problems addressed.
- Difficult to compare the solutions proposed.