RTSS 2025 – Shepherding process - Revised paper no. 151

Paper No:	151
Title:	Real-Time Multitasking of Deep Neural Networks with Nvidia TensorRT

Dear Shepherd, Dear Reviewers,

Thank you very much for your detailed comments and insightful feedback concerning the manuscript "Real-Time Multitasking of Deep Neural Networks with Nvidia TensorRT" submitted to IEEE RTSS 2025. We deeply appreciate your consideration of our manuscript and the possibility of improving it through the shepherding process.

Please find enclosed a revised version of the manuscript. Main differences in the text with respect to the previously submitted version of the manuscript are highlighted in blue.
In our revision, we tried our best to closely follow the shepherding instructions and to cover as many concerns as possible from the reviewers, and integrated relevant remarks that were anticipated as part of the authors' response during the rebuttal phase. The main changes of the revised manuscript, compared to the original submission, are as follows:

1. We extended the introduction to better describe the reference scenario (Section I).
2. We extended the description of related works and better highlighted the novelty of our work with respect to existing techniques (Section III).
3. We extended the empirical evaluation of the baseline approach with additional models to better justify the single resource abstraction and reworded the text throughout the paper to avoid generalization of these results (mostly Section IV).
4. We added additional implementation details related to both the runtime scheduling system and the optimization module of the proposed framework (Sections V and VII).
5. We added details and clarification on how the profiling is carried out for all relevant measurements presented in the paper (Sections IV, VII, and VIII).
6. We clarified the notion of optimality followed in the optimization method (Section VII).
7. We reinforced and clarified throughout the paper that all experiments were carried out on the three reference Jetson platforms, including running all optimized task sets in the schedulability experiments to validate the related results (Section VIII).
8. We carried out and discussed additional experimental results related to memory usage, startup times, and chunk launch overheads (Section VIII).
9. We checked all the measurements and results in the paper and fixed the results on optimization runtimes, which improperly utilized the profiling cache generated for the optimal method also for the heuristic method, thus producing misleading results (Section VIII).
10. We revised the overall text to fix typos and ensure that the contents fix the page limit (12 pages plus references).
11. Finally, we collected a set of execution traces using the Nvidia Nsight profiler. Unfortunately, it was not possible to fit representative figures and discussions regarding those traces; instead, we report those below to your attention. We could, however, integrate a discussion of those traces into the conference presentation.

Below you can find the commented execution traces and detailed responses to the shepherding instructions.

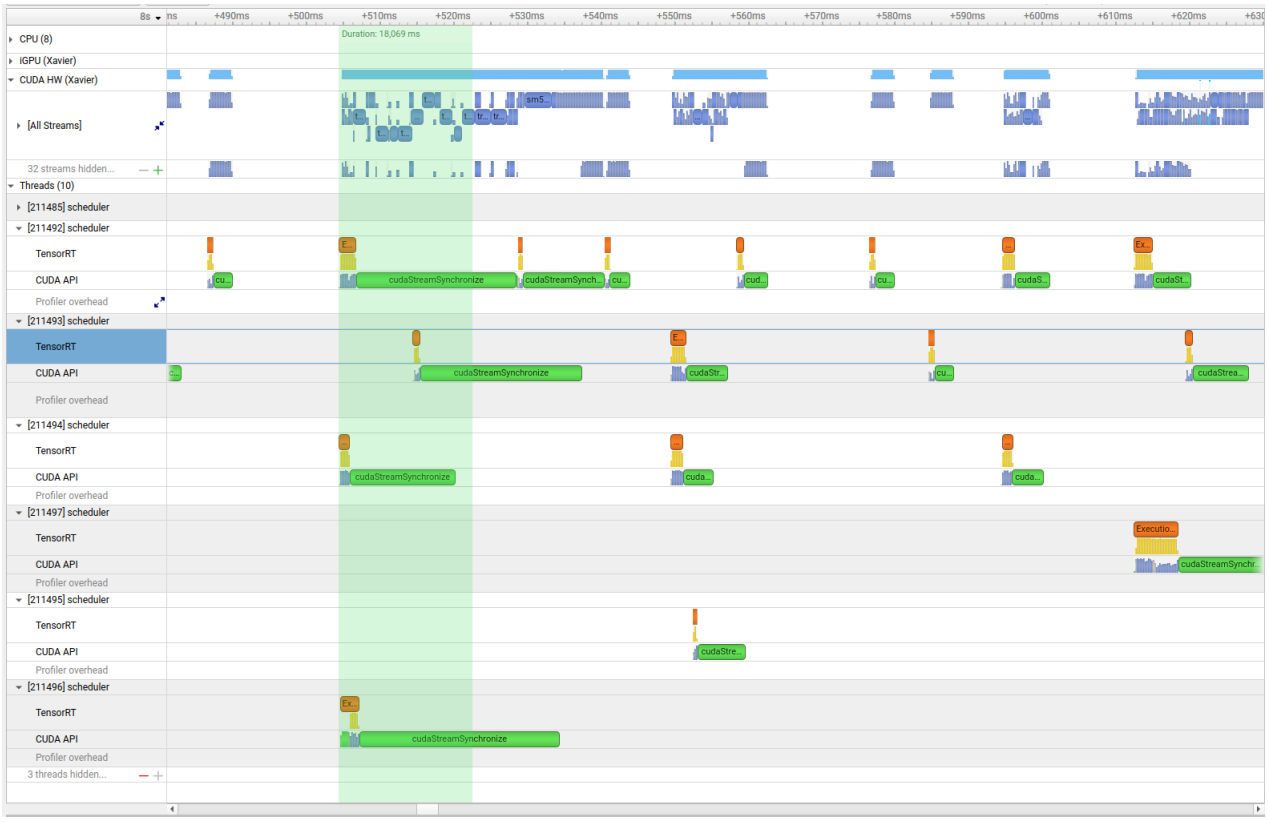
Execution traces

Please find below two execution traces captured using the Nvidia Nsight Profiler on the Nvidia Jetson AGX Xavier platform. Both traces consider a portion of the schedule obtained when running a task set with 6 tasks generated and evaluated to extract the results in Figure 4 (Section VIII).

The first trace shows the execution behavior under the baseline approach. From top to bottom, the "Threads" timelines show the evolution of:

- Task 1 - ResNet18, 18ms period;
- Task 2 - ResNet18, 35ms period;
- Task 3 - Alexnet, 45ms period;
- Task 6 - InceptionV4, 756ms period;
- Task 4 - Alexnet, 96ms period;
- Task 5 - VGG19, 216ms period.

The highlighted portion of the trace shows the deadline for the second job of task 1. Specifically, task 1 is the highest-priority task according to rate monotonic ordering, but suffers a deadline miss which causes overrun and cascaded effects on the following task instances.
Furthermore, note the limited parallelism on the GPU by inspecting the "CUDA HW" – "All Streams" timeline, even in the presence of concurrent streams.



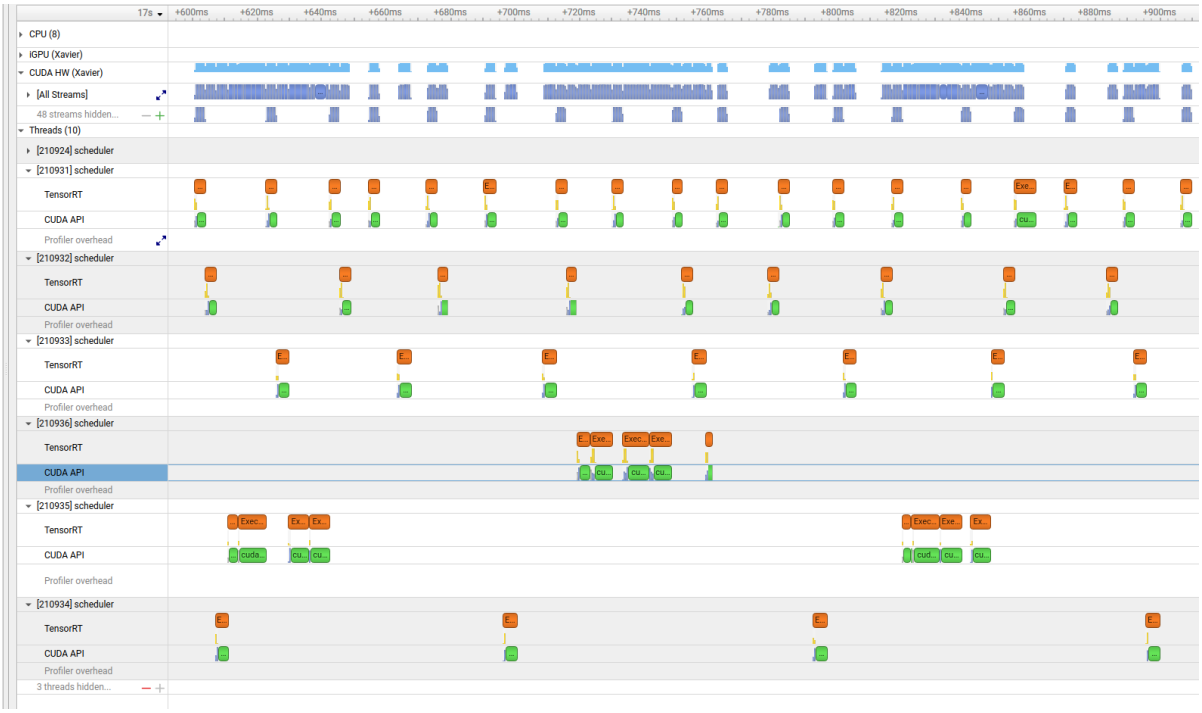
1 - Recorded execution trace under the baseline approach.

The second trace shows the execution behavior of the same task set under the proposed optimal approach. From top to bottom, the “Threads” timelines show the evolution of:

- Task 1 - ResNet18, 18ms period;
- Task 2 - ResNet18, 35ms period;
- Task 3 - Alexnet, 45ms period;
- Task 6 - InceptionV4, 756ms period;
- Task 5 - VGG19, 216ms period;
- Task 4 - Alexnet, 96ms period;

where Tasks 5 and 6 are both split into four chunks to satisfy system schedulability, and the task set was deemed schedulable by both the limited-preemptive analysis and the validation on the AGX Xavier platform.

The timeline highlights a timely behavior and an absence of deadline miss. Also note the limited-preemptive behavior exhibited by chunked tasks 5 and 6 whenever higher-priority tasks are released, and the blocking time which shifts the start time of higher-priority tasks from the release time (although release times are not captured by the trace since job release events are managed by the framework scheduler).



2 - Recorded execution trace under the optimal approach.

Comments of the Shepherd

Shepherding instruction 1: “The paper currently lacks many implementation details, such as how the DNNs are split, how to handle the intermediate data after splitting, which APIs/tools are used to implement the framework, how the DNN chunks are launched. Please clarify.”

Response: We integrated multiple comments in the paper regarding implementation details. Please see point 4 in the above list of changes.

Shepherding instruction 2: “Clarify that experiments are run on real hardware including both profiled values and measurement method for actual inference latency and inter-launch overhead measurements (e.g. which tool or timer) in Sec IV and VI. Please also add information regarding the measurement points for any profiling that the authors have done, plus fix all the potential mistakes made in the paper, as the authors may be looking at the wrong numbers. Additionally, it will be appreciated if an example runtime schedule can be provided, e.g. a screenshot of NVIDIA nsight profiler.”

Response: We integrated multiple details regarding how profiling was carried out and validated all the experiments. Please see points 5, 7, 8, and 9 in the above list of changes. See the above section for commented execution traces.

Shepherding instruction 3: “Add additional metric beyond deadline misses as Reviewer E suggested.”

Response: We added additional experimental results concerning memory, overheads, and startup latency. Please see point 8 in the above list of changes.

Shepherding instruction 4: “Multiple reviewers felt the contribution was incremental unless better justified. Please explicitly state how this work differs from prior DNN splitting frameworks.”

Response: We extended the section on related works to clarify the contribution with respect to existing splitting frameworks. Please see point 2 in the above list of changes.

Shepherding instruction 5: “Some confusion existed over what “optimal” means in the context of the splitting algorithm. Please clarify in Section VII.”

Response: We added clarifications regarding the notion of optimality in the paper. Please see point 6 in the above list of changes.