Probabilistic Schedulability Analysis for Mixed-Criticality DAG Tasks on Multiprocessors (Artifact)

Hiroto Takahashi
Graduate School of Science and Engineering, Saitama University, Japan

Atsushi Yano
Graduate School of Science and Engineering, Saitama University, Japan
TIER IV Inc., Tokyo, Japan

Takuya Azumi
Graduate School of Science and Engineering, Saitama University, Japan
TIER IV Inc., Tokyo, Japan

Abstract

This artifact accompanies the ECRTS 2026 paper "Probabilistic Schedulability Analysis for Mixed-Criticality DAG Tasks on Multiprocessors". The package provides the implementation of the proposed probabilistic schedulability analysis and baseline analyses, together with the DAG-generation wrapper, experiment drivers, plotting scripts, and input data needed to reproduce the main evaluation results of the paper. The primary reproduction workflow covers the paper-scale experiments for Figures 3–5, while additional check and smoke modes provide faster functional validation. The artifact was evaluated in the ECRTS 2026 Artifact Evaluation process and received all four artifact-evaluation seals: Artifact Available, Artifact Evaluated – Functional, Artifact Evaluated – Reusable, and Artifact Evaluated – Results Reproduced.

2012 ACM Subject Classification Computer systems organization → Real-time systems

Keywords and phrases Mixed-Criticality Systems, DAG Tasks, Probabilistic Analysis, Federated Scheduling, Real-Time Systems, Artifact Evaluation

Digital Object Identifier 10.4230/DARTS.12.2.11

Funding This work was supported by JST CREST Grant Number JPMJCR23M1 and JST FOREST Program Grant Number JPMJFR242G.

Acknowledgements We thank the ECRTS 2026 Artifact Evaluation Committee and the reviewers for their constructive feedback.

Related Article Hiroto Takahashi, Atsushi Yano, and Takuya Azumi, "Probabilistic Schedulability Analysis for Mixed-Criticality DAG Tasks on Multiprocessors", in 38th European Conference on Real-Time Systems (ECRTS 2026), LIPIcs, Vol. 375, pp. 3:1–3:23, 2026.
`https://doi.org/10.4230/LIPIcs.ECRTS.2026.3`

Related Conference 38th European Conference on Real-Time Systems (ECRTS 2026), July 7–10, 2026, Lund, Sweden

1 Scope

The artifact supports the experimental claims in the companion paper. In particular, the artifact reproduces the main quantitative results reported in Figures 3–5:

- the required number of cores across critical-path-ratio bins for the proposed analysis and baselines;
- the sensitivity of the required cores to the system-level failure budget parameter F_S; and
- the effect of increasing the clustering optimization time budget from 1 s to 10 s.

The artifact is not intended to independently validate all mathematical proofs in the paper. The artifact's purpose is to provide executable evidence for the reported evaluation pipeline, including task generation, schedulability analysis, result aggregation, and figure generation.

2 Content

The artifact package includes:

- the Python source code for the federated and multipath schedulability analyses, probabilistic clustering, simulated annealing, DAG and task-system models, and the RD-Gen conversion wrapper;
- experiment drivers for the main ECRTS 2026 evaluation path under `experiments/ex05/`;
- plotting scripts for the paper figures under `experiments/ex06/`;
- the RD-Gen DAG generator [1] as a Git submodule under `external/RD-Gen/`;
- locked Python dependency metadata through `pyproject.toml` and `uv.lock`;
- a `Dockerfile` and a top-level `reproduce.sh` script for containerized and native reproduction; and
- generated and pre-generated data under `data/`, including a pre-generated DAG pool that can be used to skip the most time-consuming generation step.

3 Getting the artifact

The artifact endorsed by the Artifact Evaluation Committee is available free of charge on the Dagstuhl Research Online Publication Server (DROPS). In addition, the artifact is also available as an archived package at: `https://github.com/azu-lab/ecrts26-artifact/releases/download/ecrts2026-darts-artifact/ecrts2026-artifact.zip`.

4 Tested platforms

The reference environment used for the paper evaluation was Linux with Python 3.13 on an AMD Ryzen 7 PRO 8840U processor with 8 cores at 2.9 GHz and 14 GiB of RAM. For native reproduction from the archived package, the artifact requires Python 3.13 and the `uv` package manager. For containerized reproduction, the artifact requires Docker or another OCI-compatible container runtime. Git submodule support is needed only when checking out the repository instead of using the archived package.

Full reproduction can take several hours, primarily because the workflow regenerates DAG pools and runs simulated annealing for each task set. The degree of parallelism affects wall-clock time for DAG generation but should not affect the resulting data because the artifact uses deterministic seeds. For quick validation, the artifact provides `./reproduce.sh check` and `./reproduce.sh smoke`; the paper-scale workflow can be run as `WORKERS=$(nproc) ./reproduce.sh full`.

5 License

The artifact source code developed for this work is available under the MIT License. Third-party components included as Git submodules or external dependencies are distributed under their respective terms.

6 MD5 sum of the artifact

47bf2e3d15908da90479091ef2d630d2

7 Size of the artifact

1.61 MiB (1,688,279 bytes)

A Reproduction Workflow

The full artifact workflow first prepares the DAG pool and then runs the three paper-scale research-question workflows. Figure 3 is reproduced by generating task sets from the pre-generated DAG pool, evaluating the required cores for fan-in and chain DAGs, and plotting the resulting CPR-binned distributions. Figure 4 is reproduced by sweeping the system-level failure budget values 10^{-7}, 10^{-8}, and 10^{-9}. Figure 5 is reproduced by reusing the RQ1 task sets and comparing 1 s and 10 s clustering optimization budgets.

Each experiment records command metadata, deterministic task-set seeds, per-taskset JSON files, CSV result tables, and generated plots. Existing results are skipped by default so that interrupted runs can be resumed; the `--no-skip-existing` option forces recomputation when needed.

References

1 Atsushi Yano and Takuya Azumi. RD-Gen: Random DAG generator considering multi-rate applications for reproducible scheduling evaluation. In *Proceedings of the 26th IEEE International Symposium on Real-Time Distributed Computing (ISORC)*, pages 21–31, 2023. doi:10.1109/ISORC58943.2023.00015.