Matlab2Trace: A Matlab to Trace translator to visualise and analyse concurrent system activities and execution traces

DOI:
10.5281/zenodo.4650066

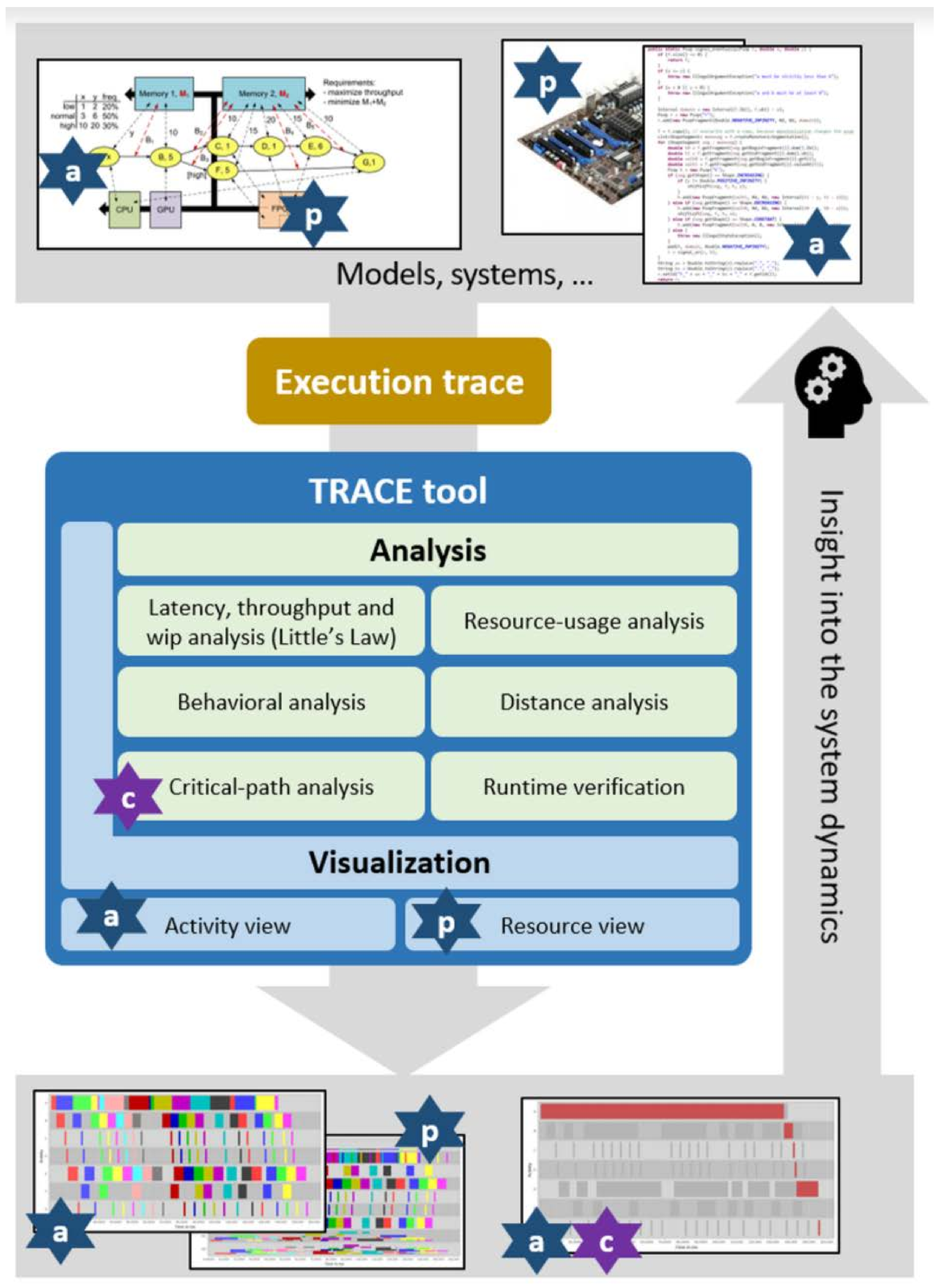
Document status and date:
Published: 01/01/2021

Matlab2Trace: A Matlab to Trace tool translator

SAJID MOHAMED*, DIP GOSWAMI*, TWAN BASTEN*^
{s.mohamed, d.goswami, a.a.basten}@tue.nl
*Electronic Systems Group, Eindhoven University of Technology; ^ESI, TNO, The Netherlands

1. TRACE tool

TRACE [1,2] (https://esi.nl/research/output/tools/trace) is a software that specialises in visualising and analysing concurrent system activities and execution traces.



source of image: https://esi.nl/research/output/tools/trace

2. Why Matlab2Trace?

- Matlab provides an environment to analyse and visualise data.
- However, there is limited support for visualising and analysing system activities executing concurrently, for instance, on a multiprocessor platform.
- The runtime behaviour of switched controllers in a multiprocessor platform [6,8,10] is difficult to visualise in Matlab. The challenge is compounded when there are variations in the timing behaviour [3-5,7,9].
- Pipelined multiprocessor control systems implementation considering timing variations [6] is hard to visualise with Matlab.

3. Matlab2Trace

- We present Matlab to TRACE tool translator that directly generates a trace-input file from the Matlab environment.
- Concurrent system activities and execution traces of the algorithms developed inside the Matlab environment can be visualised and analysed in TRACE tool using the generated trace-input file.
- Matlab2Trace translator takes as input the logical or absolute starting and ending time of the algorithmic execution, and the number (and labels) of processing cores.

4. TRACE advantages

- TRACE visualises concurrent activities in Gantt-chart-like view which provides colouring, grouping and filtering options.
- TRACE provides the following analysis methods:
  - Critical-path analysis – to detect tasks and resources that are bottleneck for performance.
  - Distance analysis – to compare execution traces with respect to structure, e.g. to check a model trace against an implementation trace.
  - Behavioral analysis – formally specify and verify properties of execution traces using Metric Temporal Logic. It is useful to express and check, for instance, performance properties such as "the processing latency is at most 50 ms".
  - Resource-usage analysis – gives quick insight into the details of the processor resource usage.

5. Accessing Matlab2Trace

Matlab2Trace: A Matlab to TRACE tool translator to visualise and analyse concurrent system activities and execution traces is open-sourced and can be accessed from https://github.com/sajid-mohamed/Matlab2Trace.

Acknowledgement

This work is part of the From the cloud to the edge–smart IntegraTion and OPtimisation Technologies for highly efficient Image and VIdeo processing Systems (FitOptiVis) project funded by the Electronic Components and Systems for European Leadership (ECSEL) Joint Undertaking under grant number H2020-ECSEL-2017-2-783162.

References

[1] Martijn Hendriks, and Twan Basten, "Performance engineering with TRACE," Bits & Chips 5, 14/09/2018. https://bits-chips.nl/artikel/performance-engineering-with-trace/
[2] Martijn Hendriks, Marc Geilen, Amir Behrouzian, Twan Basten, Hadi Alizadeh, and Dip Goswami. "Checking metric temporal logic with TRACE," in ACSD, 2016.
[3] Sayandip De, et. al., "Hardware- and Situation-Aware Sensing for Robust Closed-Loop Control Systems," In Design, Automation and Test in Europe Conference (DATE), 2021.
[4] Sayandip De, Sajid Mohamed, Dip Goswami, and Henk Corporaal, "Approximation-Aware Design of an Image-Based Control System," In IEEE Access, 2020.
[5] Sajid Mohamed, et. al., "A scenario-and platform-aware design flow for image-based control systems," In Microprocessors and Microsystems (MICPRO), 2020.
[6] Sajid Mohamed, et. al., "Adaptive predictive control for pipelined multiprocessor image-based control systems considering workload variations," In CDC, 2020.
[7] Sayandip De, et. al., "Approximation trade offs in an image-based control system," In Design, Automation and Test in Europe Conference (DATE), 2020.
[8] Sajid Mohamed, Asad Ullah Awan, Dip Goswami, and Twan Basten, "Designing image-based control systems considering workload variations," In CDC, 2019.
[9] Sajid Mohamed, et. al., "IMACS: a framework for performance evaluation of image approximation in a closed-loop system," In MECO, 2019.
[10] Majid Zamani, et. al., "Scheduling of Controllers' Update-Rates for Residual Bandwidth Utilization," In FORMATS, 2016.