Ultimate Paralizer: Parallel Trace Abstraction

Max Barth[1], Daniel Dietsch[2], Matthias Heizmann[3], and Marie-Christine Jakobs[1]

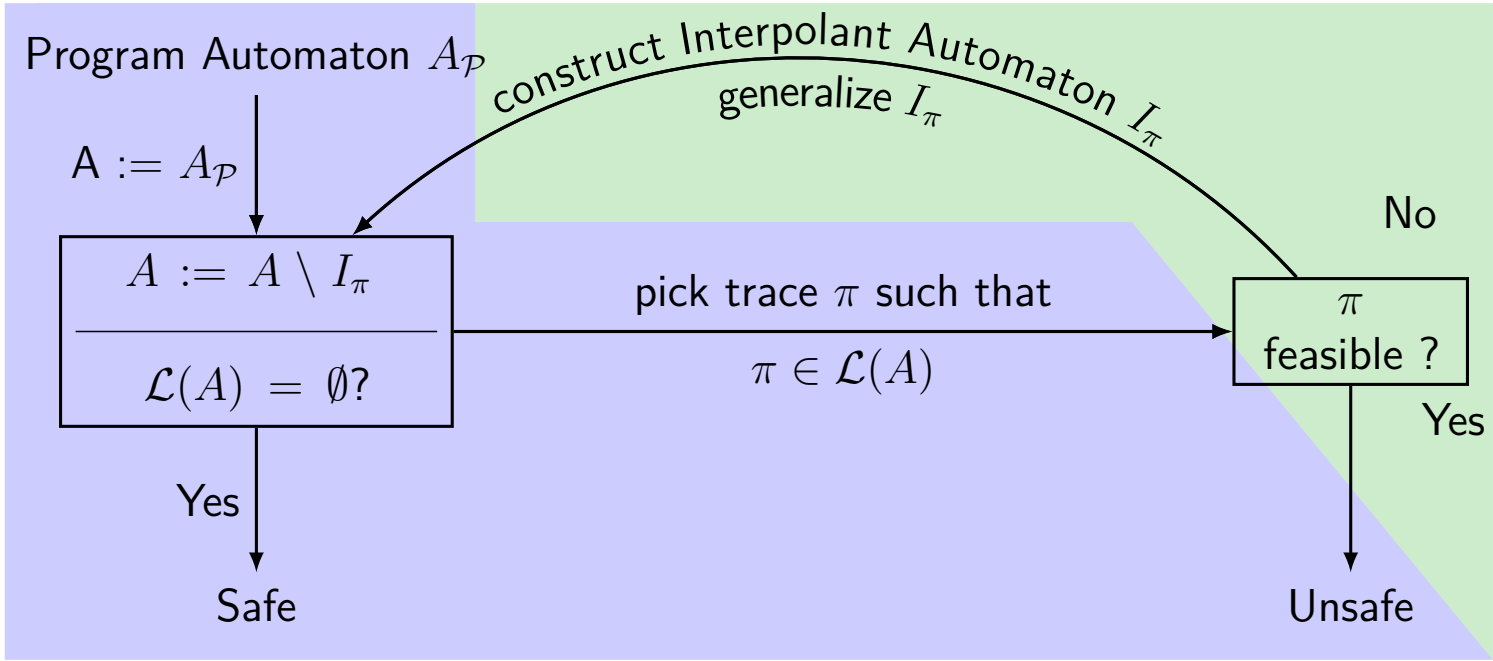
LMU Munich, Munich, Germany
University of Freiburg, Freiburg, Germany
University of Stuttgart, Stuttgart, Germany

2026-04-13

A Concurrent Framework for Trace Abstraction

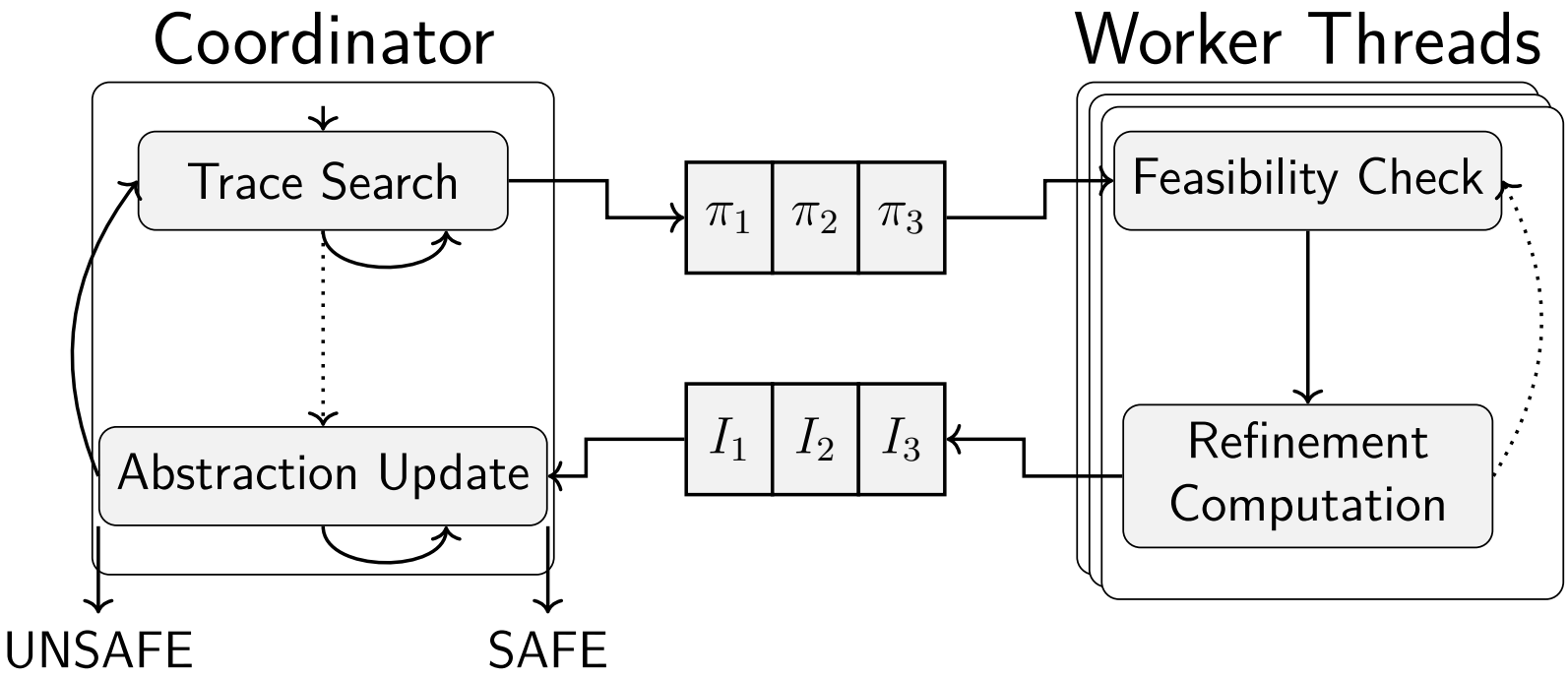
- Software Model Checking is time-consuming.
- Ultimate Paralizer utilizes a concurrent framework for Trace Abstraction.
- It is based on CEGAR and interpolation.

CEGAR Loop of Trace Abstraction



- Blue: Coordinator, maintains A and searches traces
- Green: Worker, feasibility check and creation of I_π

The Concurrent Framework for Trace Abstraction



- Buffer for traces π_i and Interpolation automata I_i
- Dotted lines indicate a thread is waiting.

A Search for a Diverse Set of Traces

We need to find diverse traces in the same Abstraction.
Idea:

- Maintain a set $\mathcal{S}_{\pi_{\text{ex}}}$ of traces that are actively checked.
- Select a fresh trace $\pi \in \mathcal{L}(A) \setminus \mathcal{S}_{\pi_{\text{ex}}}$.

Search Algorithm:

- Track which $\pi \in \mathcal{S}_{\pi_{\text{ex}}}$ are relevant during exploration.
- First explore branches taken by the least amount of $\pi \in \mathcal{S}_{\pi_{\text{ex}}}$.
- If a branch is taken by none, do BFS.

UParalizer at SV-Comp26

Participated in Reach-Safety and No-Overflows

- Won No-Overflows
- In Reach-safety we used less wall-time than UAutomizer and solved more tasks.

If you have any questions or comments, feel free to ask me in person or write an email:

Max.Barth@lmu.de