Causality-based LTL Model Checking without Automata

**Causality**
A relationship between two events, when the first event is recognized as a necessary prerequisite for the occurrence of the second. Unrolling of the program causality relation can be exponentially more succinct than its state space.

**Concurrent Traces**
Compactly represent sets of program runs, by specifying events that should necessarily occur in the run, and the partial order between them. An arbitrary number of other events can happen in between if they satisfy the edge constraints. Concurrent traces enjoy efficient algorithms for testing language emptiness and inclusion.

**Our Approach**
We consider the problem of verifying LTL properties for infinite-state concurrent programs. Our approach can be seen as a proof by contradiction technique. We assume that the property is violated, and follow causal consequences from that assumption.

- We represent sets of potential counterexample runs as concurrent traces.
- The search is organized into a trace tableau, where vertices are labeled with concurrent traces. The root trace represents all possible counterexamples.
- We unroll the trace tableau by applying causality-based proof rules.
- The property is proven when all tableau leaves are either contradictory, or can be covered by the premises of other vertices in the tableau.

**Proof Rules**

```plaintext
Proof rule is a set \( \tau : (T_1, \ldots, T_n) \) of trace transformations \( T_i : (L \rightarrow R_i) \). A proof rule should preserve the set of system runs: \( \text{L} = \bigcup_{i} \text{L}(R_i) \).
```

**Results**

- **Safety**: our causality-based algorithm is the first that is able to analyze in polynomial time the class of multi-threaded programs with binary locks and arbitrary control flow.
- **Termination**: we have developed the first termination prover, Arctor, that scales to a large number of concurrent threads (see table).
- **LTL Model Checking**: implementation is in progress. Preliminary experience shows that in some cases our approach can achieve exponentially better results compared to automata-based model checking.

**Publications**


Contact: Andrey Kupriyanov
kupriyanov@cs.uni-saarland.de
react.uni-saarland.de/people/kupriyanov.html