Rich Counter-Examples for ACTL Model Checking

Simon Busard and Charles Pecheur

ICT, Electronics and Applied Mathematics Institute

Université catholique de Louvain (UCLouvain, Belgium)

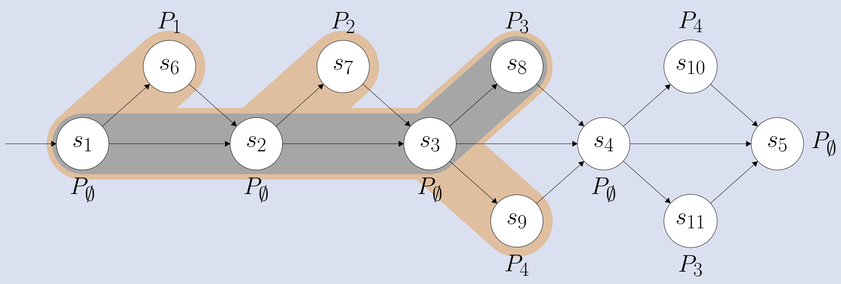
1. Objective

Problem: Standard model checkers return **single execution paths** as counter-examples while CTL has **branching counter-examples** in general.

Approach: Generate **branching** counter-examples **annotated** with parts of (the negation of) the violated property.

Example: A buggy scheduler executing prioritized processes; at each step, the scheduler can execute a process or skip it.

- This model does not satisfy the property that *the scheduler will never be able to run the process with prority* 4 *unless it is the only possibility*, expressed as **A[AX** $\neg P_4$ **W AX** $(P_\emptyset \vee P_4)$**]** in CTL (P_x means that process x is running, $P_\emptyset$ means that no process is running).
- The full counter-example is branching, but model checkers return partial information.



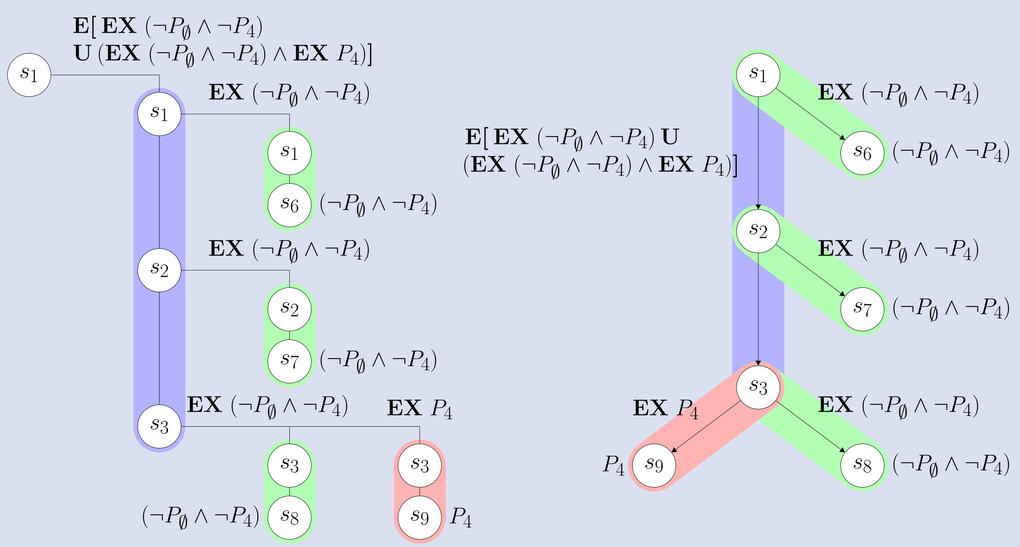
This work is based on Clarke et al. and Rasse.

2. Tree-like Annotated Counter-Examples

TLACEs are full counter-examples for ACTL—and full witnesses for ECTL—i.e. they completely explain a violation. A TLACE is defined by

$$n ::= node(s, \{(b \mid \neg b)^*\}, \{(\mathbf{E}\ \pi : p)^*\}, \{(\mathbf{A}\ \pi)^*\})$$
$$p ::= \langle n^+ \rangle \mid \langle n^+, loop(n) \rangle$$

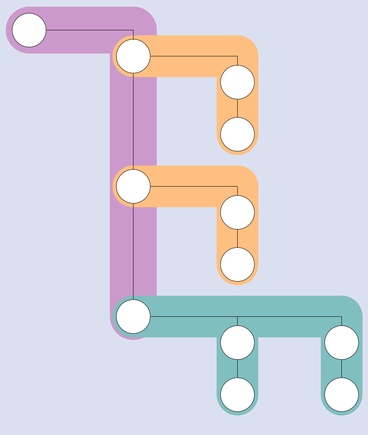
We are interested in **consistent** TLACEs that are **adequate** for a model $\mathcal{M}$ and a property ϕ, i.e. TLACEs that come from $\mathcal{M}$ and correctly explain ϕ.



3. Generating Counter-Examples

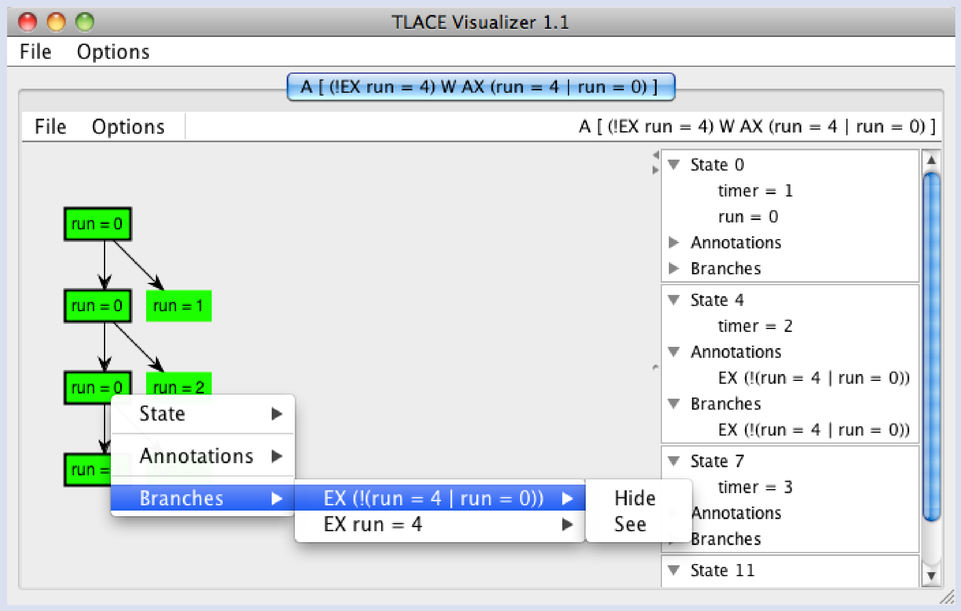
explain **algorithm:** works recursively on the **structure of the formula**; relies on sub-algorithms to extract paths from the model witnessing temporal formulas.

switch ϕ **do**
- **case** ...
- **case** $\boldsymbol{E}[\psi_1\ \boldsymbol{U}\ \psi_2]$
  - $\langle s_1, ..., s_m \rangle \leftarrow EUexplain(\mathcal{M}, s, \psi_1, \psi_2)$
  - $p \leftarrow \langle\rangle$
  - **for** $s_i \in \langle s_1, ..., s_{m-1} \rangle$ **do**
    - $p \leftarrow p + \langle explain(\mathcal{M}, s_i, \psi_1) \rangle$
  - $p \leftarrow p + \langle explain(\mathcal{M}, s_m, \psi_2) \rangle$
  - **return** $node(s, \{\}, \{\boldsymbol{E}\ \pi : p\}, \{\})$
- **case** ...



4. Tools

- **NuSMV:** modified to **generate** and export TLACEs.
- **TLACE Visualizer: displaying** and **manipulating** TLACEs
  - TLACEs displayed as **graphs**, the graph can be rearranged
  - **folding**/**unfolding** branches
  - **displaying** variables on the graph
  - **inspecting nodes:** state information, annotations and branches
  - **displaying** nodes information along a path in the graph



5. Conclusion and further work

Contributions:
- Formalization of branching annotated counter-examples for ACTL;
- Generating algorithm;
- Implementation in a symbolic model checker;
- Tool for visualizing and manipulating the counter-examples.

Further work:
- Extend the formalization to richer logics like epistemic temporal logics;
- Interactive generation of branches: explain only the part relevant for the user;
- Explain **A** operators through interactive game: the user can try to show the satisfaction while the system shows him that it is impossible.

References

- E. M. Clarke, S. Jha, Y. Lu, H. Veith. Tree-like Counterexamples in Model Checking. *LICS'02*, 2002.
- A. Rasse. Error Diagnostics in Finite Communicating Systems. *CAV*, 1991.

A. CTL Model Checking

Model checking: checks whether a (finite state) **model** satisfies a (temporal logic) **property**; if not, returns a **counter-example**.

CTL: branching temporal logic expressing properties about the **execution tree** of the model.

AX ϕ — *all successors satisfy* ϕ

EG ϕ — *there exists a path where all states satisfy* ϕ

A[ϕ **W** ψ **]** — *for all paths,* ϕ *is true up to a state satisfying* ψ

ACTL: only **A** quantifiers and negations applied to atomic propositions.

Grascomp's Day, November 3rd, 2011, Université Libre de Bruxelles, Brussels, Belgium.

Simon.Busard@uclouvain.be

Union européenne

Le Fonds européen de Développement régional et la Wallonie investissent dans votre avenir.

Wallonie