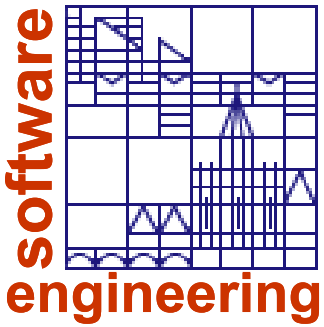
Causality Checking for Complex System Models

Florian Leitner-Fischer

University of Konstanz
Department of Computer and Information Science
Chair for Software Engineering



Joint work with

Stefan Leue

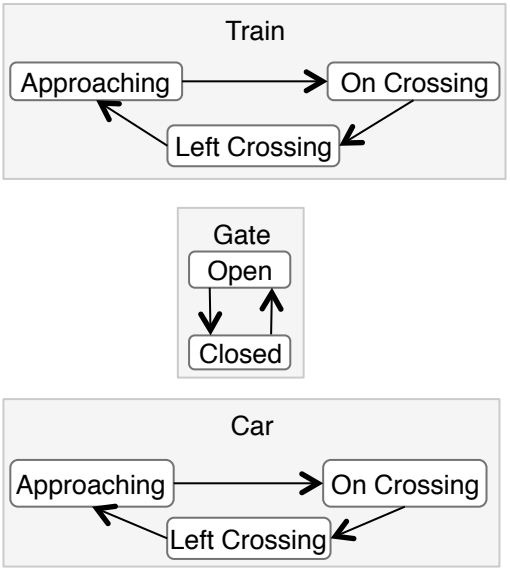
Chair for Software Engineering
Department of Computer and Information Science
University of Konstanz
Germany

Analysis of Complex Systems

- **A Railroad Crossing**

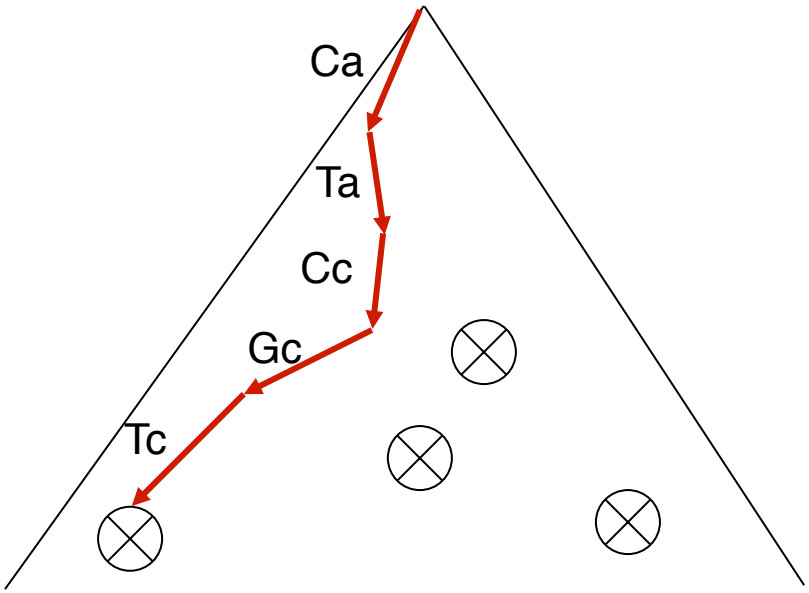
Model Checking

$M \models S$

model of the software (transition system, Kripke structure)	model checking algorithm	requirement specification (assertions, temporal logic, automata)
	state space search (depth-first or breadth-first search)	*there is never a train in the crossing at the same time when there is a car in the crossing* $\varphi = \Box\neg(Tc \wedge Cc)$

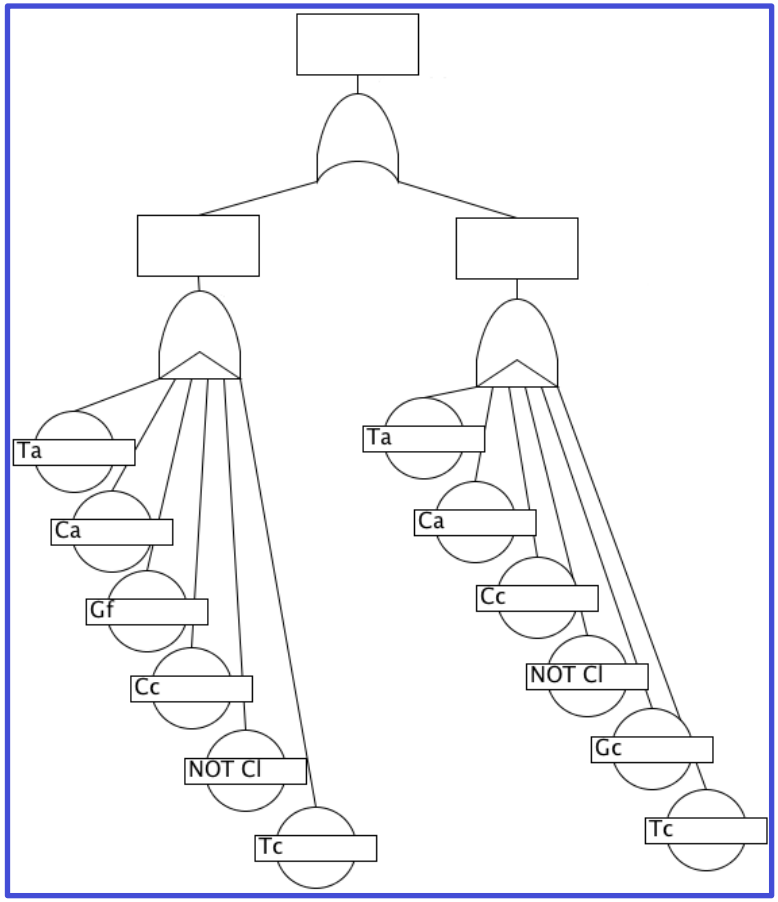
Model Checking

- **Explicit State Model Checking**
  - most common: automatic search of all reachable system states to find property violations using depth-first search (DFS) or breadth-first search (BFS)
  - the path into a property violating state is called an error path or counterexample



Interpreting Counterexamples

- **Railroad Crossing Example:**
  - 11 error-paths (only considering shortest paths)



- all lead into a property violating state (accident)
- for debugging
  - **what is the cause?**
- manual analysis
  - tedious
  - error prone
  - essentially impossible
- our goal:
  - **algorithmic causality computation**

Outline

- Models of Causation
- An Adopted Structural Equation Model
- Causality Checking
- Experimental Evaluation
- Conclusion

Outline

- **Models of Causation**
- An Adopted Structural Equation Model
- Causality Checking
- Experimental Evaluation
- Conclusion

Causality

- **(Naive) Lewis Counterfactual Reasoning**

  c is causal for e (effect / hazard) if, had c not happened, then e would not have happened either

  - logical foundation of some software debugging techniques, e.g.,
    - delta debugging
    - nearest neighbor techniques
  - best suited for single cause failures

Consequences

- **Need for Alternate Worlds**
  - what-if analysis
    - had there been another course of action (= "world") in which the gate had been closed before the car entered the crossing, there would not have been an accident (= a "good" world)
  - "good" world: the effect does not occur
  - "bad" world: the effect occurs

- **Limitations**
  - not suited for effects that have logically complex causal structure
  - We use an adaption of the **Structural Equation Model** by Halpern and Pearl
    - SEM is based on Lewis counterfactional reasoning

Halpern / Pearl Structural Equation Model (SEM)

- **Key Ideas**
  - events are represented by boolean variables
    - specified using structural equations
  - computes minimal boolean disjunction and conjunction of causal events
  - causal dependency of events represented by causal networks
  - reference

J. Halpern and J. Pearl, "Causes and explanations: A structural-model approach. Part I: Causes," *The British Journal for the Philosophy of Science*, 2005.

Halpern / Pearl Structural Equation Model (SEM)

- **Actual Causality Conditions**
  - AC1: ensures that there exists a world where the boolean combination of causal events **c** and the effect **e** occur
  - AC2:
    1. if at least one of the causal events does not happen, the effect **e** does not happen
    2. if the causal events occur, the occurrence of other events can not prevent the effect
  - AC3: no subset of the causal events satisfies AC1 and AC2 (minimality)

Outline

- Models of Causation
- **An Adopted Structural Equation Model**
- Causality Checking
- Experimental Evaluation
- Conclusion

An Adopted Structural Equation Model

- **Main Goals**
  - Consider event order as causal factor
  - Make Structural Equation Model applicable to transition systems

Event Order Logic

- **Boolean Event Occurrence Conditions**
  - $a \wedge b$, $a \vee b$, $\neg a$
- **Event Ordering Conditions**
  - $a \wedge\!\!\cdot\, b$
    - a and b occur, and a occurs before b
- **Interval Operators**
  - $a \wedge\!\!\cdot\, [\, b$
    - a occurs until eventually b will hold in every state
  - $a \wedge\!\!\cdot\,]\, b$
    - a always holds until eventually b occurs
  - $a \wedge\!\!\cdot_{<} b \wedge\!\!\cdot_{>} c$
    - in the interval delimited by a and c, b always holds
- **Model-theoretic Semantics**
  - Event Order Logic is an LTL

Event Order Logic

- **Representation of Traces**

$$\sigma = \text{“Ta, Ca, Gf, Cc, Tc”}$$

$$\psi = \mathrm{Ta} \mathbin{\dot{\wedge}} \mathrm{Ca} \mathbin{\dot{\wedge}} \mathrm{Gf} \mathbin{\dot{\wedge}} \mathrm{Cc} \mathbin{\dot{\wedge}} \mathrm{Tc}$$

- **Representation of Ordering Constraints**

$$(\mathrm{Ta} \wedge \mathrm{Ca}) \mathbin{\dot{\wedge}} \mathrm{Gf}$$

$$\mathrm{Cc} \mathbin{\dot{\wedge}}_{<} \neg \mathrm{Cl} \mathbin{\dot{\wedge}}_{>} \mathrm{Tc}$$

Adopted SEM

- Is $\psi = \mathrm{Ta} \wedge \mathrm{Ca} \wedge \mathrm{Gf} \wedge \mathrm{Cc} \wedge \mathrm{Tc}$
  causal for the violation of $\varphi = \Box\neg(\mathrm{Tc} \wedge \mathrm{Cc})$?

- **AC1**
  - there exists σ so that both $\sigma \models \psi$ and $\sigma \not\models \neg\varphi$

- **Remarks**
  - the "positive" side of counterfactual test
  - True if there exists a error-path σ = Ta, Ca, Gf, Cc, Tc

Adopted SEM

- Is $\psi = \mathrm{Ta} \wedge \mathrm{Ca} \wedge \mathrm{Gf} \wedge \mathrm{Cc} \wedge \mathrm{Tc}$ causal for the violation of $\varphi = \Box\neg(\mathrm{Tc} \wedge \mathrm{Cc})$?

- **AC2 (1)**
  - $\exists\ \sigma'$ where the order and occurrence of events is different from σ and φ is not violated on σ'

- **Remarks**
  - this is the counterfactual test
  - AC2 (1) fulfilled by $\psi = \mathrm{Ta} \wedge \mathrm{Ca} \wedge \mathrm{Gf} \wedge \mathrm{Cc} \wedge \mathrm{Tc}$
    - since there exists σ' = Ta, Ca, Gc, Tc so that
      - σ' is different from σ = Ta, Ca, Gf, Cc, Tc
      - $\sigma' \models \varphi$ (σ' is a "good" path)

Adopted SEM

- Is $\psi = \mathrm{Ta} \wedge \mathrm{Ca} \wedge \mathrm{Gf} \wedge \mathrm{Cc} \wedge \mathrm{Tc}$ causal for the violation of $\varphi = \Box\neg(\mathrm{Tc} \wedge \mathrm{Cc})$?

- **AC2 (2)**
  - for a sequence of events to be causal it cannot be possible to add an event so that causality is voided

- **Remarks (1)**
  - Motivation
    - serves to reveal that non-occurrence is causal
  - consider σ" = "Ta, Ca, Gf, Cc, Cl, Tc"
    - for σ" the property φ is not violated since Cl occurs before Tc
  - consequence:
    - ψ is not causal (AC2 (2) fails)
    - the non-occurrence of an event (Cl) is causal

Adopted SEM

- Is $\psi = \mathrm{Ta} \barwedge \mathrm{Ca} \barwedge \mathrm{Gf} \barwedge \mathrm{Cc} \barwedge \mathrm{Tc}$
  causal for the violation of φ = □¬(Tc ∧Cc) ?

- **Causality of Non-Occurrence (what if AC2(2) fails?)**
  - steps
    - find minimal set of causal non-occurrence events
    - add, depending on the position of the event in a_α this set
      - $\neg a_\alpha \barwedge]$ at the beginning of ψ
      - $\barwedge_[\neg a_\alpha$ at the end of ψ
      - $\barwedge_< \neg a_\alpha \barwedge_>$ in the middle of ψ
      - perform test AC2 (2) again
  - example

$$\psi = \mathrm{Ta} \barwedge \mathrm{Ca} \barwedge \mathrm{Gf} \barwedge \mathrm{Cc} \barwedge_< \neg \mathrm{Cl} \barwedge_> \mathrm{Tc}$$

Adopted SEM

- Is $\psi = \mathrm{Ta} \wedge \mathrm{Ca} \wedge \mathrm{Gf} \wedge \mathrm{Cc} \wedge \mathrm{Tc}$
  causal for the violation of $\varphi = \Box\neg(\mathrm{Tc} \wedge \mathrm{Cc})$?

- **AC3**
  - ψ is minimal: no subset of ψ satisfies conditions AC1 and AC2

Adopted SEM

- Is $\psi = \mathrm{Ta} \mathbin{\dot{\wedge}} \mathrm{Ca} \mathbin{\dot{\wedge}} \mathrm{Gf} \mathbin{\dot{\wedge}} \mathrm{Cc} \mathbin{\dot{\wedge}} \mathrm{Tc}$ causal for the violation of $\varphi = \Box\neg(\mathrm{Tc} \wedge \mathrm{Cc})$?

- **OC1 Causality of Event Order**
  - let Ψ an eol formula over some events in ψ
  - for some eol formula Ψ, replace the ordered operator $\dot{\wedge}$ by the unordered $\wedge$ yielding $\Psi_{\wedge}$
  - the order expressed by Ψ is not causal if
    - $\sigma \vDash \Psi \wedge \exists \sigma' \in \Sigma_B : \sigma' \nvDash \Psi \wedge \sigma' \vDash \Psi_{\wedge}$
  - example
    - order of events Gf, Cc, ¬Cl, Tc is important for causing property violation
    - relative order of Ta and Ca is not important, but they need to precede the above events
    - resulting formula $\psi = (\mathrm{Ta} \wedge \mathrm{Ca}) \mathbin{\dot{\wedge}} \mathrm{Gf} \mathbin{\dot{\wedge}} \mathrm{Cc} \mathbin{\dot{\wedge}}_{<} \neg\mathrm{Cl} \mathbin{\dot{\wedge}}_{>} \mathrm{Tc}$

Outline

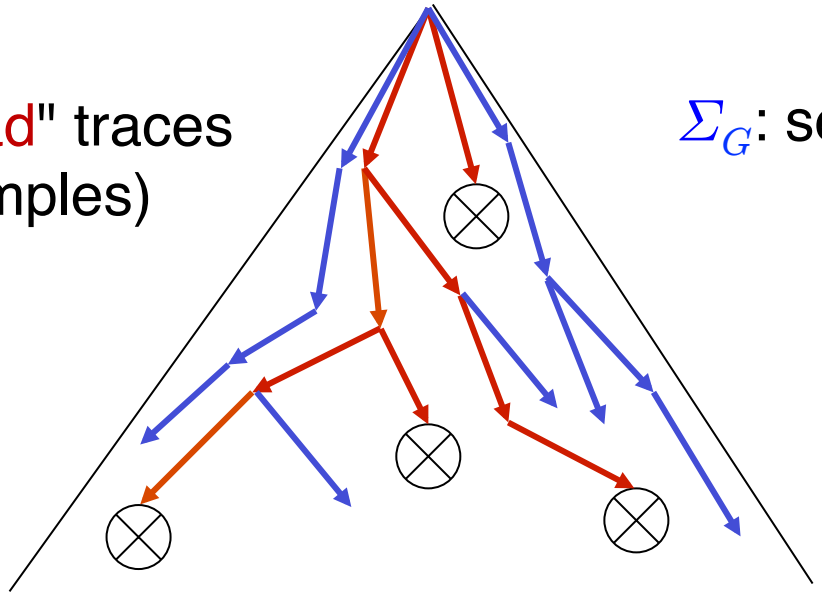
- Models of Causation
- An Adopted Structural Equation Model
- **Causality Checking**
- Experimental Evaluation
- Conclusion

Execution Traces and Counterfactuals

- **Traces Define (Alternate) Worlds**

- **Computed by State Space Search**
  - model checking
    - traverse state space using BFS or DFS
  - applicable to reachability properties
    - no meaningful behavior after property violation is observed

Σ_B: set of "bad" traces (counterexamples)



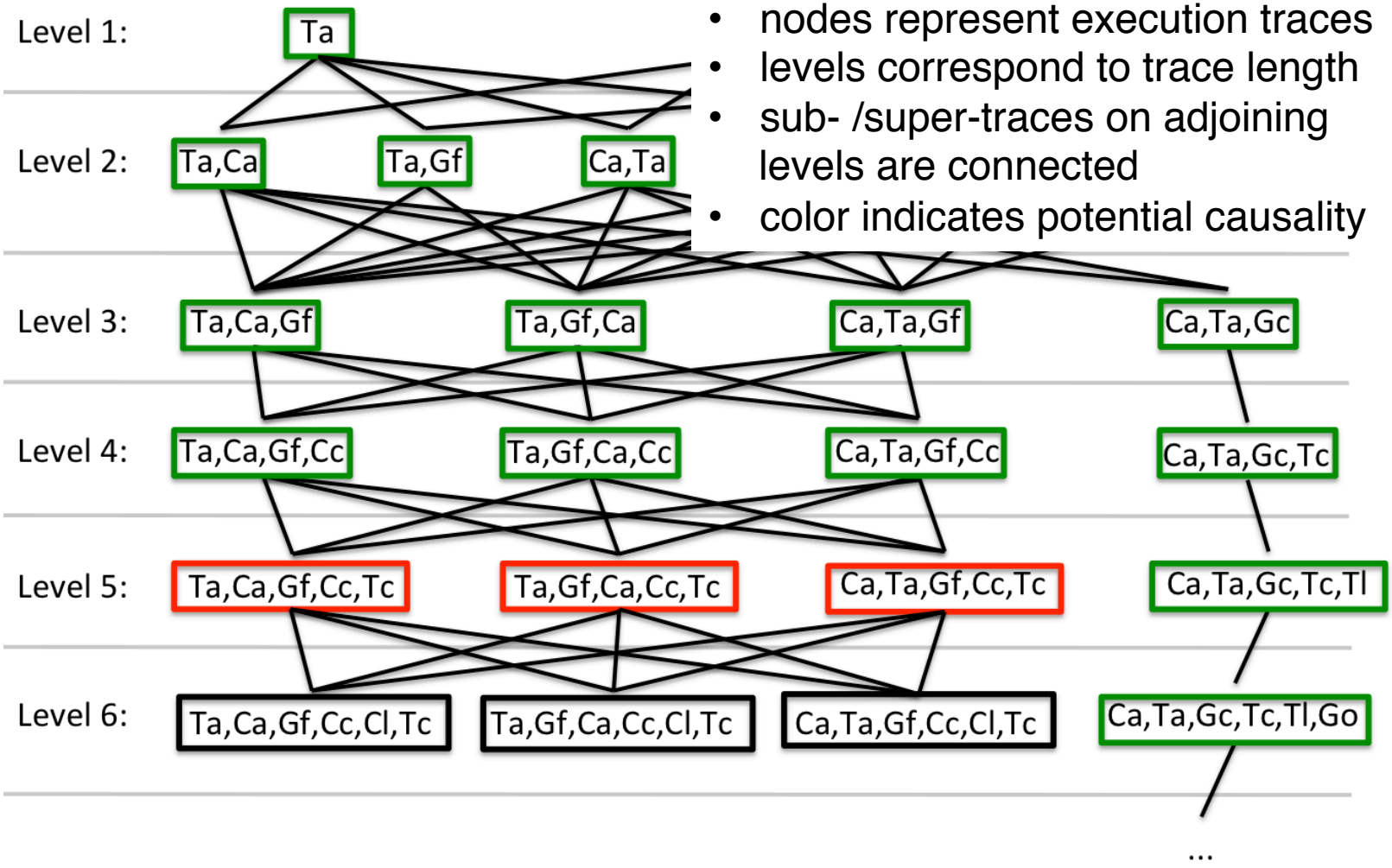
Σ_G: set of "good" traces

Algorithmics

- **Sub-Executions**
  - reduce checks for AC1-AC3 and OC1 to sub-execution tests
    - ordered and unordered sub-execution operators
  - proofs in the paper

- **Implementation Variants**
  - Off-line Enumeration
    - enumerate traces
    - store Σ_B and Σ_G
    - perform sub-trace computations
  - On-the-fly
    - use DFS / BFS on the state space
      - store paths in an adequate data structure as you obtain them
        - subset graph

Subset Graph



- nodes represent execution traces
- levels correspond to trace length
- sub- /super-traces on adjoining levels are connected
- color indicates potential causality

Inference From Subset Graph

- **Theorems for Adopted SEM Conditions**
  - eol formula ψ_σ derived from a red node σ fulfills AC1 and AC2(1) and AC3
    - for BFS: AC3 fulfilled immediately
    - for DFS: when search terminates

- **Construction of Subset Graph**
  - on the fly during state space search
  - once state space search complete, perform tests for AC2(2) and OC1

Complexity (Preliminary)

- **Caveat**
  - *even for an SEM with only binary variables, computing causal relationships between variables is NP-complete*

    Eiter and Lukasziewicz, 2002

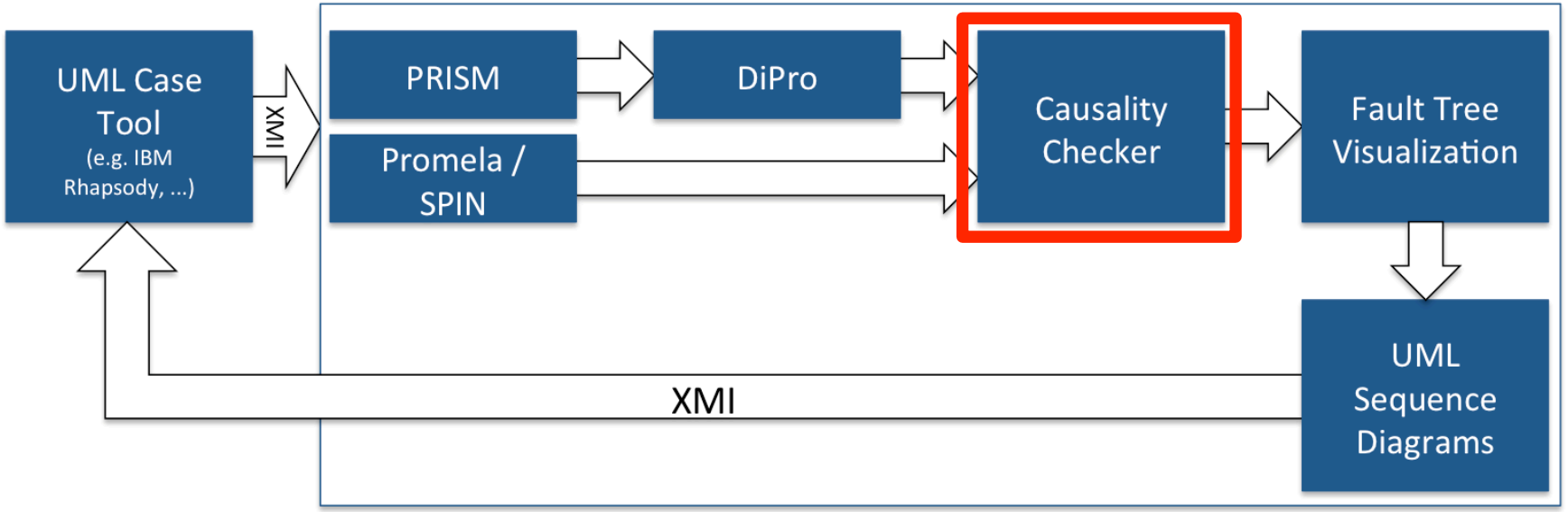
- **However**
  - Eiter and Lukasziewicz, 2006: for cycle-free causal dependencies computing causal relationships can be done polynomially
  - naturally, this is only part of the complexity analysis…

Outline

- Models of Causation
- An Adopted Structural Equation Model
- Causality Checking
- **Experimental Evaluation**
- Conclusion

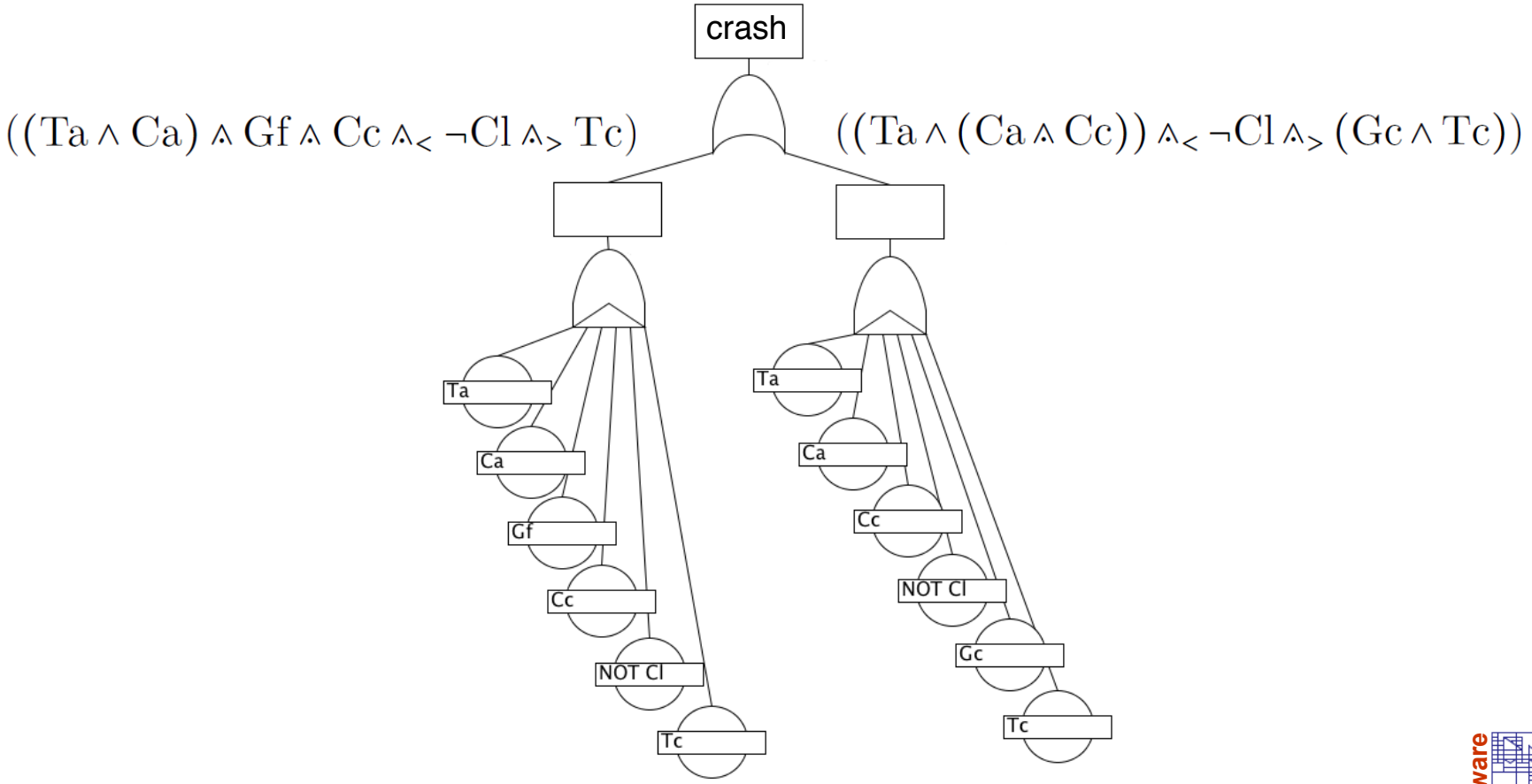
Tool Environment

- **QuantUM Tool Architecture**
  - prototypical implementation: SpinCause



Experiments

- **Railroad Crossing**
  - Promela model with 133 states and 137 transitions
  - represented as Dynamic Fault Tree



Experiments

- **Railway Crossing**

- **Airbag System**

- **Observations**
  - BFS outperforms DFS
    - rely on minimality of length of bad traces found
      - requires less good traces to be stored
  - on-the-fly outperforms off-line enumeration
    - on-line: only store red and black traces
    - off-line: store all traces

Outline

- Models of Causation
- An Adopted Structural Equation Model
- Causality Checking
- Experimental Evaluation
- **Conclusion**

Conclusion

- **Causality Checking**
  - technique complementing model checking
    - aim: algorithmic support for the debugging of models
  - defined / adopted causality model
  - proposed implementation
  - applicability to non-trivial case studies

- **Future Work**
  - causality checking at the limits of scalability
    - dealing with incomplete information
  - causality checking in a symbolic environment
  - on-line causality checking for probabilistic models
  - specific adaptions to functional safety analysis
    - minimal cut sets
    - root, common and cascading causes