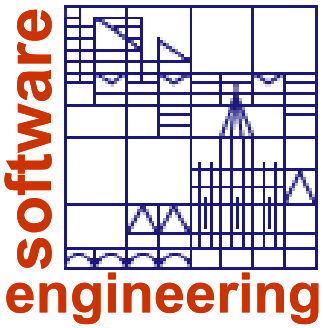
Recent Advances in Causality Checking

Florian Leitner-Fischer

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



Joint work with

Stefan Leue

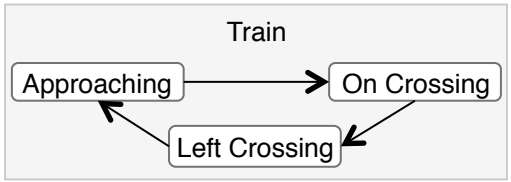
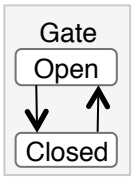
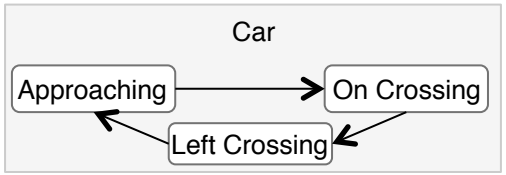
Chair for Software Engineering
Department of Computer and Information Science
University of Konstanz
Germany

Analysis of Complex Systems

- **A Railroad Crossing**

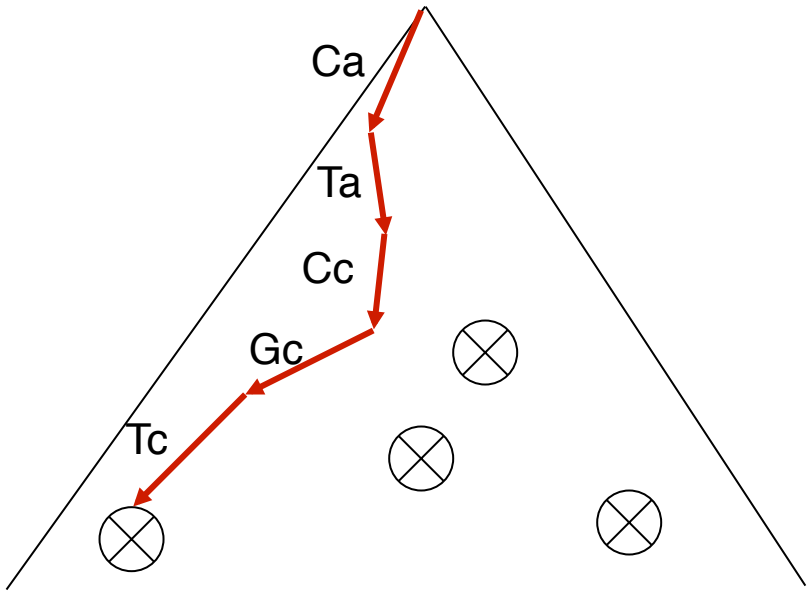
Model Checking

$M \models S$

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* $\varphi = \Box\neg(Tc \wedge Cc)$
		
		

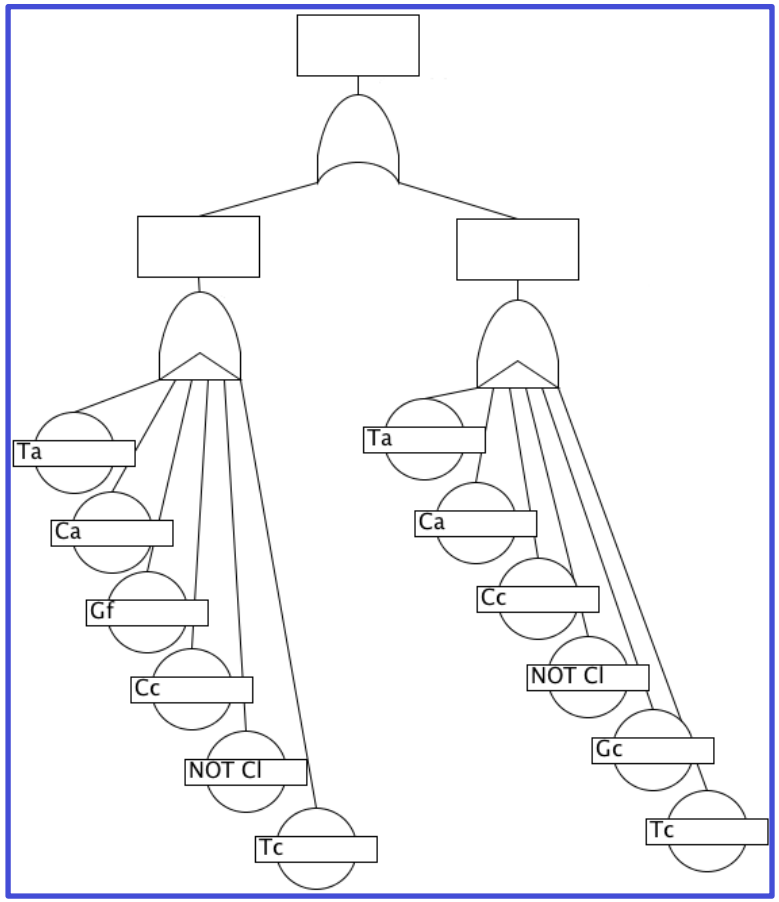
Model Checking

- **Model Checking Result**
  - the path into a property violating state
    - 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**

- **Models of Causality**
- **Causality Computation**
- **Probability Computation for Causal Events**
- **Evaluation**
- **Conclusion**

- **Models of Causality**
- Causality Computation
- Probability Computation for Causal Events
- 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

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)

- Models of Causality
- **Causality Computation**
- Probability Computation for Causal Events
- Evaluation
- Conclusion

Causality Computation

- **Main Goal:**
  Computation of Causal Events for a Property Violation
  - Consider event order as causal factor
  - Make Structural Equation Model applicable to transition systems

Florian Leitner-Fischer and Stefan Leue:
Probabilistic Fault Tree Synthesis using Causality Computation,
accepted for publication in International Journal of Critical Computer-Based Systems, 2013.

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**

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

$$\psi = \mathrm{Ta} \barwedge \mathrm{Ca} \barwedge \mathrm{Gf} \barwedge \mathrm{Cc} \barwedge \mathrm{Tc}$$

- **Representation of Ordering Constraints**

$$(\mathrm{Ta} \wedge \mathrm{Ca}) \barwedge \mathrm{Gf}$$

$$\mathrm{Cc} \barwedge_{<} \neg\mathrm{Cl} \barwedge_{>} \mathrm{Tc}$$

Causality Computation

- **Probabilistic Causality Computation**
  - Probabilistic counterexample and good paths are computed
  - Causality computation performed as post-processing step
  - Benefit
    - Probability for combination of events causing a hazard
  - Disadvantage
    - Probability computation for each bad trace necessary

- **Causality Checking**
  - Integrated into the state space search algorithms used for model checking
  - Benefit
    - Enables on-the-fly causality computation
  - Disadvantage
    - No probabilities available

Algorithmics

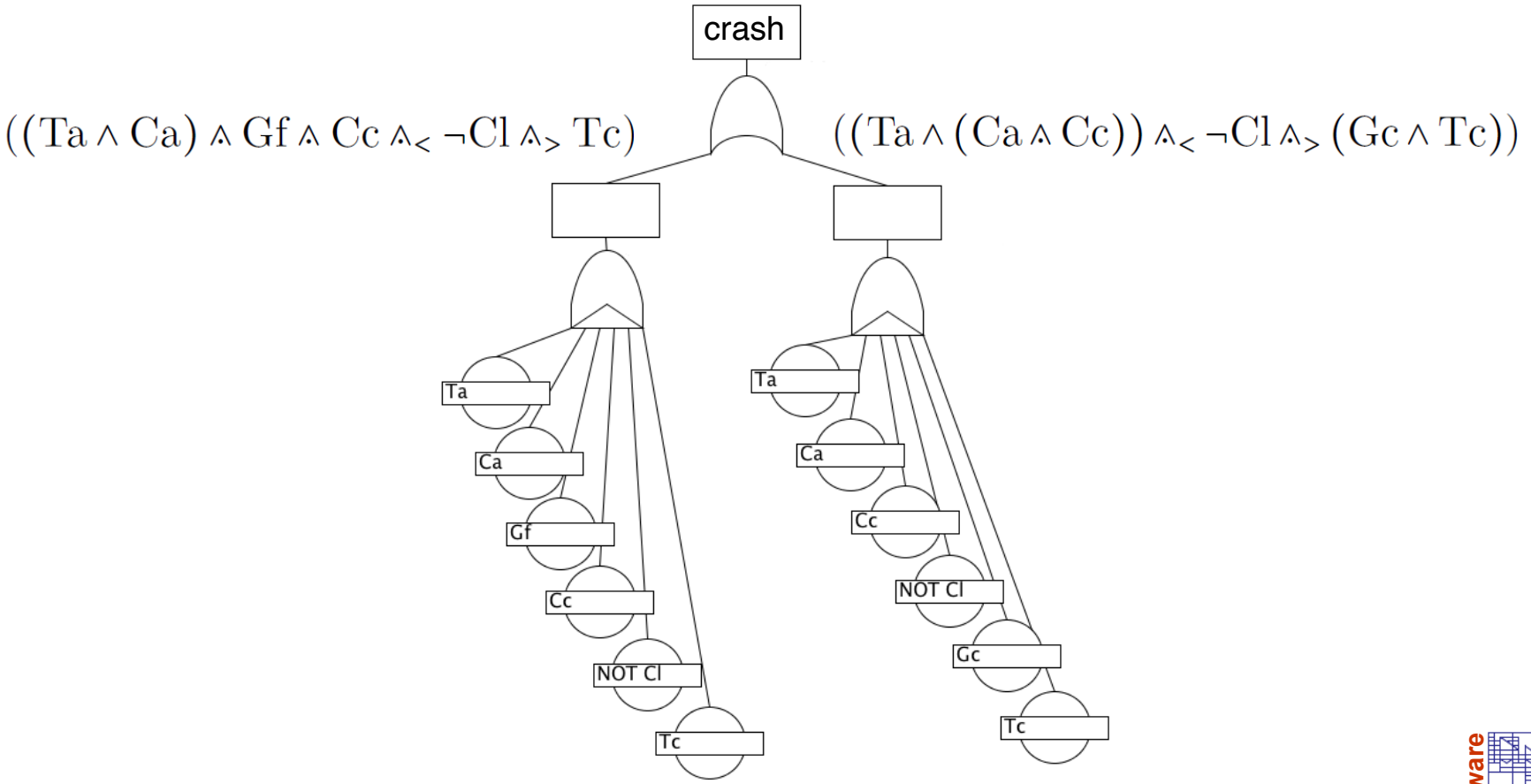
- **Sub-Executions**
  - reduce checks for AC1-AC3 and OC1 to sub-execution tests
    - ordered and unordered sub-execution operators
  - proofs in the paper

- **Implementation Variants**
  - Off-line Enumeration
    - enumerate traces
    - store counterexamples and good traces
    - 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

Florian Leitner-Fischer and Stefan Leue:
Causality Checking for Complex System Models,
In Proceedings of 14th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI2013), LNCS, Volume 7737, Springer Verlag, 2013.

Result of Causality Checking

- **Railroad Crossing**
  - represented as Dynamic Fault Tree



- Models of Causality
- Causality Computation
- **Probability Computation for Causal Events**
- Evaluation
- Conclusion

Causality Classes

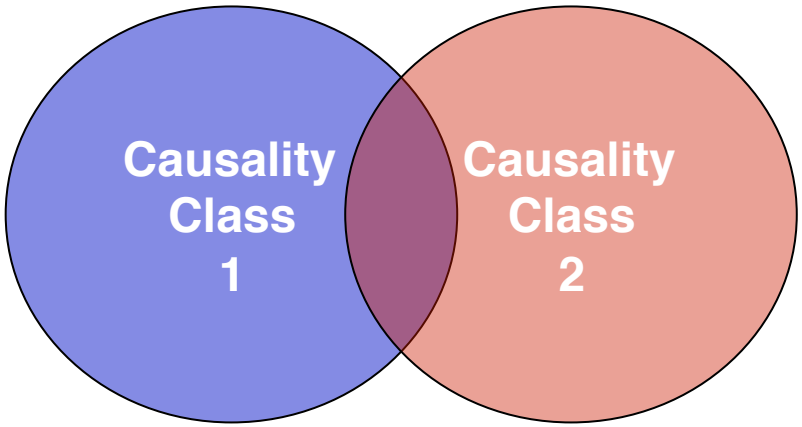
- **Disjuncts of the EOL formula represent „causality classes“**
  - Causality Classes: Represent a class of execution paths where the same events in the same order cause an effect or hazard

$$((\mathrm{Ta} \wedge \mathrm{Ca}) \mathbin{\mathbb{A}} \mathrm{Gf} \mathbin{\mathbb{A}} \mathrm{Cc} \mathbin{\mathbb{A}}_{<} \neg\mathrm{Cl} \mathbin{\mathbb{A}}_{>} \mathrm{Tc})$$

Ta, Ca, Gf, Cc, Tc
Ca, Ta, Gf, Cc, Tc
…

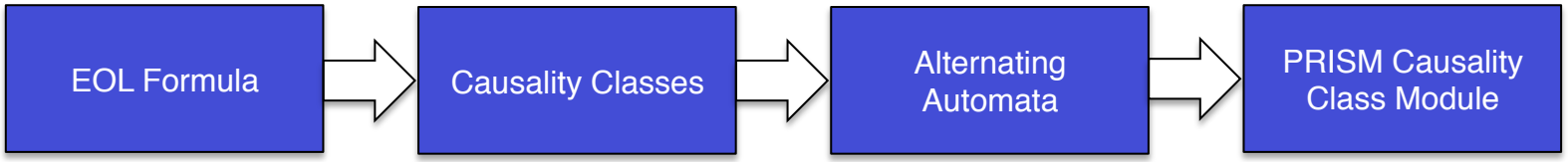
$$((\mathrm{Ta} \wedge (\mathrm{Ca} \mathbin{\mathbb{A}} \mathrm{Cc})) \mathbin{\mathbb{A}}_{<} \neg\mathrm{Cl} \mathbin{\mathbb{A}}_{>} (\mathrm{Gc} \wedge \mathrm{Tc}))$$

Ta, Ca, Gc, Gc, Tc
Ca, Ta, Cc, Gc, Tc
…



Probability Computation for Causality Classes

- **Key Idea:**
  1. Compute causal events using causality checking
  2. Compute probabilities of the paths represented by a causality class



- Causality classes are represented by alternating automata
- Alternating automata are translated to PRISM Causality Class Modules
- PRISM Causality Class Modules are synchronized with the PRISM model

Florian Leitner-Fischer and Stefan Leue:
On the Synergy of Probabilistic Causality Computation and Causality Checking,
In Proceedings of International SPIN Symposium on Model Checking of Software, Stony Brook, NY, USA, 2013 (to appear).

- Models of Causality
- Causality Computation
- Probability Computation for Causal Events
- **Evaluation**
- Conclusion

Experimental Evaluation

	Combined Approach		Probabilistic Causality Comp.	
	Run time (sec.)	Memory (MB)	Run time (sec.)	Memory (MB)
Embedded: States: 6,013 Transitions: 25,340				
T=10	3.06	19.27	2,003.00	409
T=3600	4.79	19.29	2,102.00	409
Airbag: States: 2,952 Transitions: 14,049				
T=10	10.88	52.44	682.00	154
T=1000	33.63	52.44	874.00	154
Train Odometer Controller: States: 117,222 Transitions: 66,262				
T=10	91.37	195.29	16,191.00	1,886
T=1000	2,572.74	195.29	44,356.00	1,886

- **Combined Approach:**
  - Causality Checking + Probability Computation for Causality Classes
- **Combined Apporach outperforms Probabilistic Causality Computation**

- Models of Causality
- Causality Computation
- Probability Computation for Causal Events
- Evaluation
- **Conclusion**

Conclusion

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

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