Seminar
Trends in Computer-Aided Verification
Introduction

Harold Bruintjes Christian Dehnert Christina Jansen Tim Lange
Viet Yen Nguyen Thomas Noll Kevin van der Pol
Sabrina von Styp

Software Modeling and Verification Group

RWTH AACHEN UNIVERSITY

noll@cs.rwth-aachen.de

Winter Semester 2013/14; 9 October, 2013

Outline

Formal Verification Methods

Formal verification methods

- Rigorous, mathematically based techniques for the specification, development and verification of software and hardware systems
- Aim at improving correctness, reliability and robustness of such systems

Formal Verification Methods

Formal verification methods

- Rigorous, mathematically based techniques for the specification, development and verification of software and hardware systems
- Aim at improving correctness, reliability and robustness of such systems

Classifications

- According to design phase
  - specification, implementation, testing, ...
- According to specification formalism
  - process algebras, timed automata, Markov chains, ...
- According to underlying mathematical theories
  - model checking, theorem proving, static analysis, ...

Outline

Goals

Aims of this seminar

- Independent understanding of a scientific topic
- Acquiring, reading and understanding scientific literature
- Writing of your own report on this topic
- Oral presentation of your results

Requirements on Report

Your report

- Independent writing of a report of 15–20 pages
- Complete set of references to all consulted literature
- Correct citation of important literature
- Plagiarism: taking text blocks (from literature or web) without source indication causes immediate exclusion from this seminar
- Font size 12pt with "normal" page layout
- Language: German or English
- We expect the correct usage of spelling and grammar
  - $\geq$ 10 errors per page $\Longrightarrow$ abortion of correction

Requirements on Talk

Your talk

- Talk of about 45 minutes
- Focus your talk on the audience
- Descriptive slides:
  - $\leq$ 15 lines of text
  - use (base) colors in a useful manner
- Language: German or English
- No spelling mistakes please!
- Finish in time. Overtime is bad
- Ask for questions

Final Preparations

Preparation of your talk

- Setup laptop and projector ahead of time
- Use a (laser) pointer
- Number your slides
- Multiple copies: laptop, USB, web
- Have backup slides ready for expected questions

Outline

Important Dates

Talks

The seminar will be held as a weekly double feature on Mondays at 15:30 (?) starting end of November

- see `http://www-i2.informatik.rwth-aachen.de/i2/cav13/`

Important Dates

Talks

The seminar will be held as a weekly double feature on Mondays at 15:30 (?) starting end of November

- see `http://www-i2.informatik.rwth-aachen.de/i2/cav13/`

Deadlines

You are requested to adhere to the following firm deadlines:

- immediately: obtain the required literature from the web or library
- eight weeks before your talk: present a table of contents
- six weeks before your talk: preliminary version of your report
- four weeks before your talk: final version of your report
- two weeks before your talk: preliminary version of your slides
- one week before your talk: final version of your slides

Important Dates

Talks

The seminar will be held as a weekly double feature on Mondays at 15:30 (?) starting end of November

- see `http://www-i2.informatik.rwth-aachen.de/i2/cav13/`

Deadlines

You are requested to adhere to the following firm deadlines:

- immediately: obtain the required literature from the web or library
- eight weeks before your talk: present a table of contents
- six weeks before your talk: preliminary version of your report
- four weeks before your talk: final version of your report
- two weeks before your talk: preliminary version of your slides
- one week before your talk: final version of your slides

Missing a deadline causes immediate exclusion from the seminar

Outline

Selecting Your Topic

Procedure

- You obtain(ed) a list of topics of this seminar.
- Indicate the preference of your topics (first, second, third).
- We do our best to find an adequate topic-student distribution.
- Disclaimer: no guarantee for an optimal solution.
- Your topic will be published on our website by 11 October.
- Please give language preference
  - unsure $\Longrightarrow$ German

Selecting Your Topic

Procedure

- You obtain(ed) a list of topics of this seminar.
- Indicate the preference of your topics (first, second, third).
- We do our best to find an adequate topic-student distribution.
- Disclaimer: no guarantee for an optimal solution.
- Your topic will be published on our website by 11 October.
- Please give language preference
  - unsure $\Longrightarrow$ German

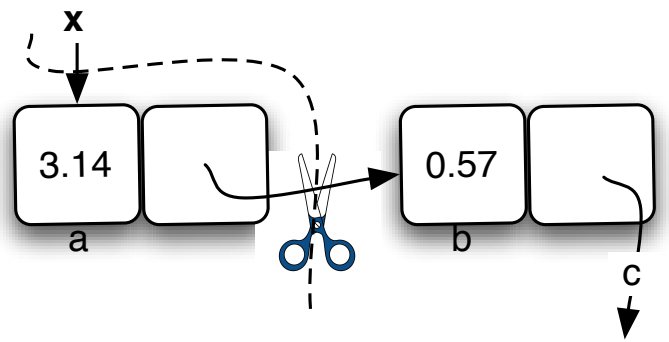
Withdrawal

- You have up to three weeks to refrain from participating in this seminar.
- Later cancellation (by you or by us) causes a not passed for this seminar and reduces your (three) possibilities by one.

Outline

1: Interprocedural Shape Analysis with Separated Heap Abstractions

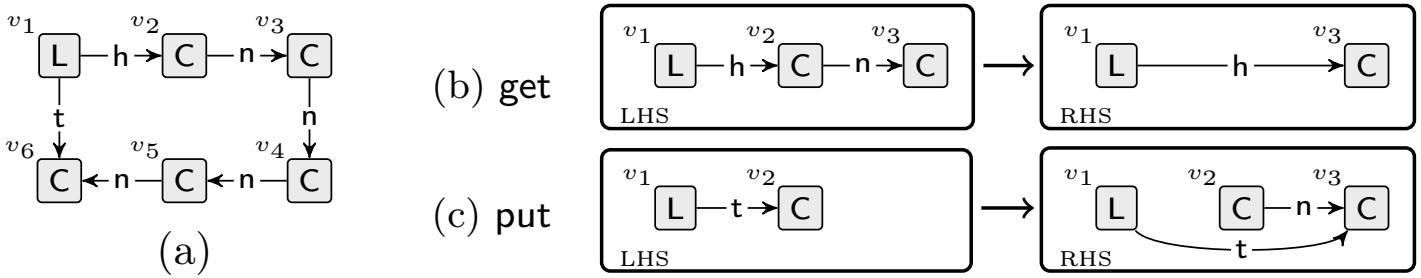
Microsoft Research



- interprocedural analysis of heap-manipulating programs
- goal: proving shape properties
- heap described by (Separation) Logic formulae
- Separation Logic: extension of Hoare Logic

2: Pattern-Based Graph Abstraction

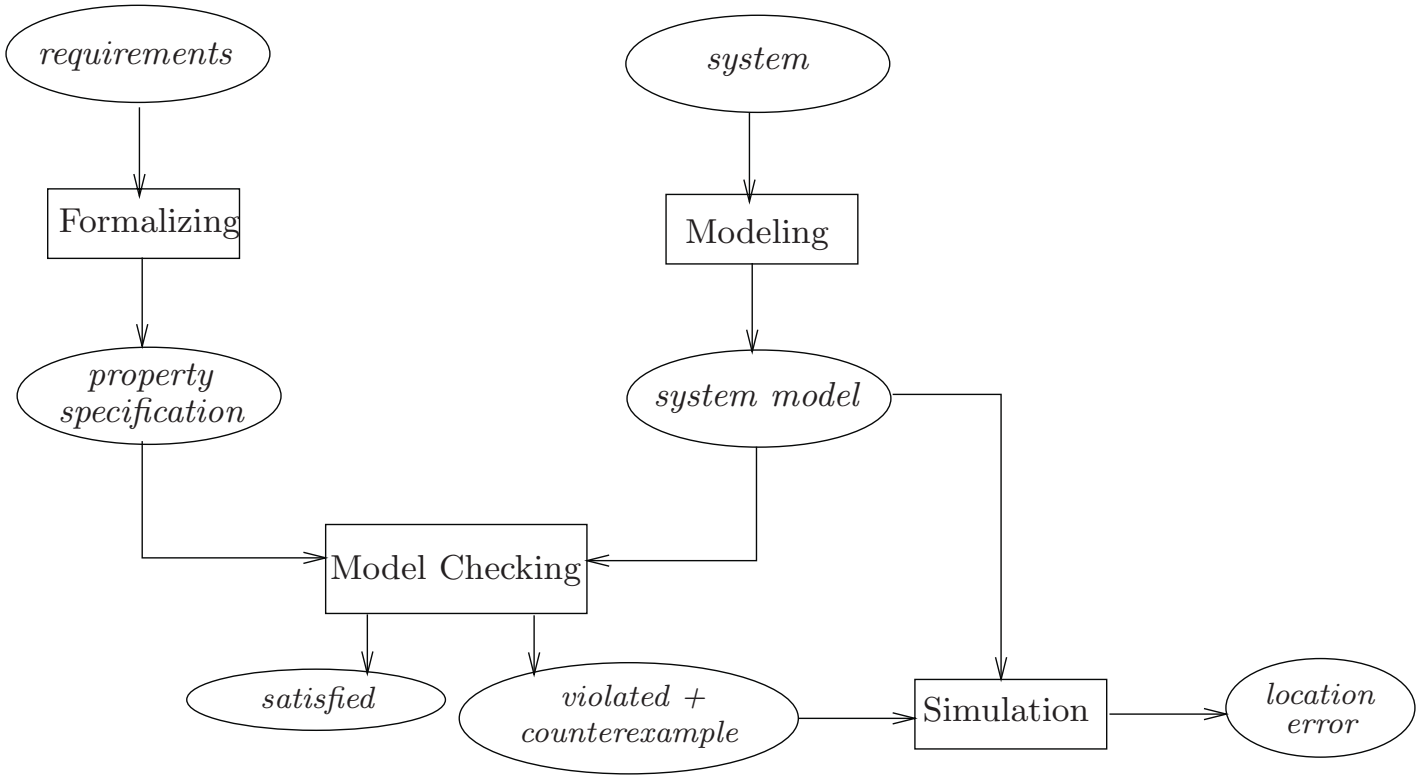
University of Twente
The Netherlands



- (abstract) state space of heap-manipulating programs
- program statements: graph transformations
- abstraction: composition of pattern graphs
- tool support: GROOVE

Outline

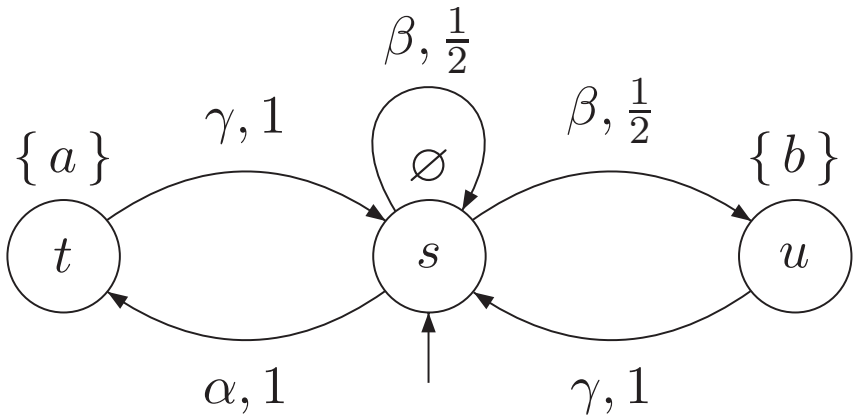
Model Checking



Extension: Probabilistic Models and Temporal Logics

3: Faster Algorithms for MDPs with Low Treewidth

Model: Markov Decision Processes



Problem: verification time consuming

Idea: exploit typical shape of models to speed-up verification

4: Assume-Guarantee Verification for Probabilistic Systems

Model: Markov Decision Processes

$$\begin{array}{ccccccccc} \mathcal{M} & = & \mathcal{M}_1 & \| & \mathcal{M}_2 & \| & \ldots & \| & \mathcal{M}_n \\ \downarrow & & \downarrow & & \downarrow & & & & \downarrow \\ \text{huge} & & \text{small} & & \text{small} & & & & \text{small} \end{array}$$

Problem: verification of full system time and memory consuming

Idea: try to verify subsystems instead

5: Pareto Curves for Probabilistic Model Checking

Model: Markov Decision Processes

$$\overrightarrow{\varphi} = \Big(\underset{\downarrow}{\varphi_1} , \underset{\downarrow}{\varphi_2} , \cdots , \underset{\downarrow}{\varphi_n} \Big)$$

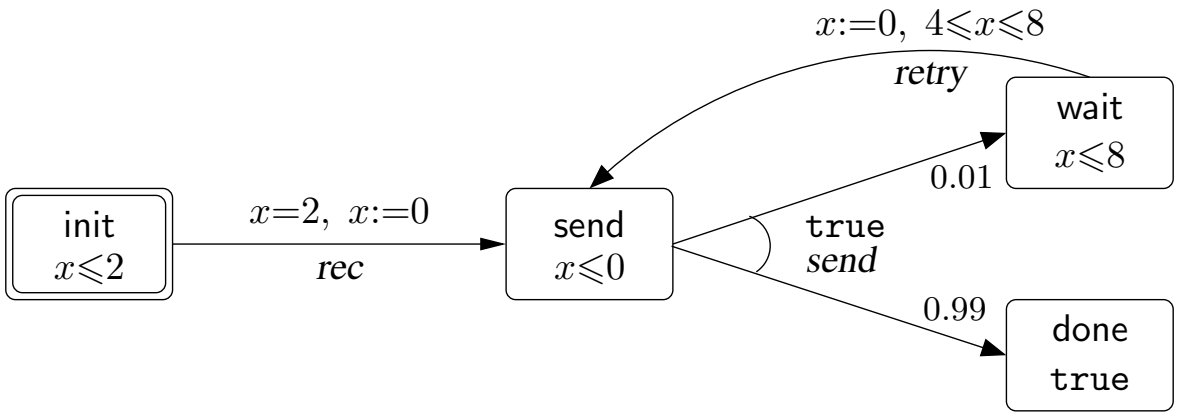
$$p(\rightsquigarrow e_1) \bowtie p_1 \qquad p(\rightsquigarrow e_2) \bowtie p_2 \qquad p(\rightsquigarrow e_1) \bowtie p_1$$

Problem: how to verify (possibly conflicting) $\overrightarrow{\varphi}$

Idea: successively approximate Pareto curve of MDP

6: Stochastic Games for Verification of PTA

Model: Probabilistic Timed Automata



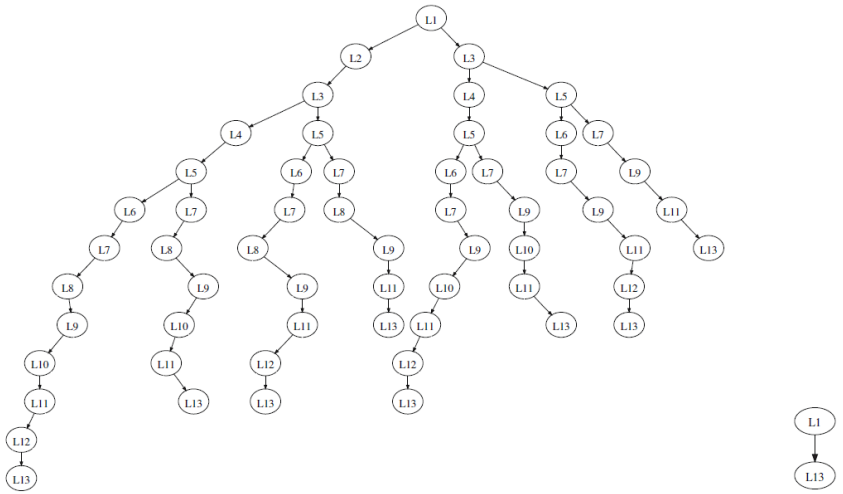
Problem: how to verify timed properties

Idea: reduce model to two-player game

Outline

7: Software Model Checking via Large-Block Encoding

Model: Abstract Reachability Tree



Problem: State space explosion

Idea: Minimize ART by minimizing control flow graph

8: Checking Safety by Inductive Generalization of Counterexamples to Induction

First completely new MC approach since McMillan's interpolation.

Base:	Initial states satisfy property
Induction:	If in property state, no transition to non-property state

Problem: unreachable states violating induction

Outline

9: Action Refinement in Conformance Testing

- In model-based testing actions are automatically generated from a formal model
- The formal model is often more abstract than the actual implementation
- Rewriting the model is error-prone and time consuming
- Solution: Action refinement, relating action from the abstract model to the concrete action of the implementation

10: Testable Properties

- Generation test-cases from the model can lead to large test suits
- One possibility to improve the practical feasibility is to use properties for deriving test-cases
- Advantage: Properties allow for selecting certain test-cases

11: Integrating Formal Verification and Conformance Testing for Reactive Systems

- Verification of formal models can be undecidable or too complex, especially for symbolic, infinite-state transition systems
- There is no guarantee that verified properties are tested at all
- This paper describes the link between verifying properties on the specification and testing them on the implementation
- Testing may detect violations of properties on the implementation that could not fully be verified on the specification

Outline

12: Zeno analysis of UPPAAL models

Zeno analysis of UPPAAL models

- Rinast, J., & Schupp, S. (2012). Static detection of zeno runs in UPPAAL networks based on synchronization matrices and two data-variable heuristics. Formal Modeling and Analysis of Timed Systems, 220-235.
- Timed automata may contain erroneous behavior: Zeno.
- Paper describes method for static analyis of such models (UPPAAL) to detect this behavior.
- Heuristics to deal with data variables.
- Algorithm implemented in tool (ZenoTool) and tested on various models, compared to existing detection methods.

13: Expected time and long-run average in IMCs

Expected time and long-run average in IMCs

- Guck, D., Han, T., Katoen, J., & Neuhäußer, M. (2012). Quantitative timed analysis of interactive Markov chains. NASA Formal Methods, 1-15.
- IMC: Model containing both non-determinism and continuous time probabilistic transitions.
- Paper describes analysis of long-run average and expected time properties of such models.
- Reduction to SSP problem for both properties.
- Implemented in the IMCA tool, expiriments with two case studies
  - Google file system
  - Workstation cluster

Outline

14: Interpolation-Based Model Checking

Interpolation (Craig, 1957)

Simple & important result from mathematical logic. Craig proved:

- Given $A \implies B$, where A and B are first-order logic formulae,
- Then $\exists C :: A \implies C \wedge C \implies B$, where $var(C) = var(A) \cap var(B)$.

Interpolant C *abstracts* A to variables shared between A and B.

Applications to Model Checking (McMillan, 2003)

McMillan showed the power of Craig's result to model checking:

- Abstracting successor-states for state space exploration.
- Abstracting the transition relation itself.

Also applications to program verification, compositional reasoning, etc.

Supervisor

Viet Yen Nguyen (nguyen@cs.rwth-aachen.de)

Outline

15: Imperative Program Synthesis

- *Andrew Ireland and Jamie Stark*, Combining Proof Plans with Partial Order Planning for Imperative Program Synthesis, 2006
- Problem: *construct* program P s.t. $\{Q\}P\{R\}$ for given Q, R
- Based on structured programming
- When to apply which rules?
- Planning heuristics on two levels
  - Proof plans
  - Partial order planning

Outline

/Users/thomasnoll/Research/D-MILS/Meetings/05_Aachen_Oct_13/W

16: Using Architecture to Reason About Information Security

- **Goal:** derivation of information flow security properties
- **Inputs:**
  - architectural abstraction of system
    - set of security domains
    - information flow relation
  - logical specification of distribution of knowledge
- **Procedure:** model checking of system architecture w.r.t. logical spec
- Possible extension of architecture by filter functions to restrict information flow
- Implementation by access control policies

17: Hyperproperties

- "Classical" properties: satisfied by traces
  - safety: "nothing bad happens" (e.g., mutual exclusion)
  - liveness: "eventually something good happens" (e.g., every server request will be answered)
- Every property representable as intersection of safety and liveness property
- Some security properties expressible as properties:
  - "system may not write to network after reading from file" (safety)
- But not all, e.g., noninterference:
  - "commands issued by users holding high clearances can be removed without affecting observations of users holding low clearances"
  - formally: whenever system contains trace t, it also contains t' without high inputs but with same low events as t

  $\Longrightarrow$ property satisfied by set of traces ("hyperproperty")
- Hypersafety, hyperliveness, representation theorem

Outline

Some Final Hints

Hints

- Take your time to understand your literature.
- Be proactive! Look for additional literature and information.
- Discuss the content of your report with other students.
- Be proactive! Contact your supervisor on time.
- Prepare the meeting(s) with your supervisor.
- Forget the idea that you can prepare a talk in a day or two.

Some Final Hints

Hints

- Take your time to understand your literature.
- Be proactive! Look for additional literature and information.
- Discuss the content of your report with other students.
- Be proactive! Contact your supervisor on time.
- Prepare the meeting(s) with your supervisor.
- Forget the idea that you can prepare a talk in a day or two.

We wish you success and look forward to an enjoyable and high-quality seminar!