Seminar *Theoretical Foundations of Programming Languages*

Introduction
Summer Semester 2016; 13 April, 2016
C. Dehnert, T. Lange, C. Matheja, T. Noll, M. Volk, H. Wu
Software Modeling and Verification Group
RWTH Aachen University

https://moves.rwth-aachen.de/teaching/ss-16/tfopl/
Overview

Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

T. Noll, C. Matheja: Analysis of Pointer Programs

T. Lange: Software Model Checking

C. Dehnert, M. Volk, H. Wu: Analysis of Probabilistic Systems

Final Hints
Theoretical Foundations of Programming Languages

- Seminar addresses several aspects of programming languages and software systems (in a broad sense)
- Emphasis: formal foundations and principles underpinning practical applications
### Overview

#### Theoretical Foundations of Programming Languages

- Seminar addresses several aspects of programming languages and software systems (in a broad sense)
- Emphasis: formal foundations and principles underpinning practical applications

### Aspects

- **Analysis of Pointer Programs**
  - *Static Program Analysis* (WS 2014/15)
  - *Semantics and Verification of Software* (SS 2015)
- **Software Model Checking**
  - *Introduction to Model Checking* (SS 2015, now)
  - *Advanced Model Checking* (SS 2014)
- **Analysis of Probabilistic Systems**
  - *Modelling and Verification of Probabilistic Systems* (SS 2014)
Analysis of Pointer Programs

Pointer-related software errors

- Dereferencing null (or disposed) pointers
- Creation of memory leaks
- Accidental invalidation of data structures
- Deadlocks
- Data races, ...
Overview

Software Model Checking

Programmable Logic Controller (PLC) Code

- Tailored for automation processes
- Run ad infinitum
- Executed in a cyclic manner
- Terminate within predefined cycle time

Challenges

- Application domains have high safety requirements
- Violations entail high economical costs
- Currently checked by extensive testing

Example

Motor must move in safe range, otherwise worker gets injured.
Overview

Analysis of Probabilistic Systems

“Jungle” of models
- Discrete vs. continuous time
- Deterministic vs. non-deterministic
- ...

Interesting questions
- Reachability properties (of “bad” states)
- Transient probability distributions
- Model checking
- Analysis of failures (fault trees)
- ...

6 of 34 Seminar Theoretical Foundations of Programming Languages T. Noll et al. Summer Semester 2016; 13 April, 2016
Aims of this Seminar

Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

T. Noll, C. Matheja: Analysis of Pointer Programs

T. Lange: Software Model Checking

C. Dehnert, M. Volk, H. Wu: Analysis of Probabilistic Systems

Final Hints
### Aims of this Seminar

#### Goals

<table>
<thead>
<tr>
<th>Aims of this seminar</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Independent understanding of a scientific topic</td>
</tr>
<tr>
<td>• Acquiring, reading and understanding scientific literature</td>
</tr>
<tr>
<td>• Writing of your own report on this topic</td>
</tr>
<tr>
<td>• Oral presentation of your results</td>
</tr>
</tbody>
</table>
Aims of this Seminar

Requirements on Report

Your report

- Independent writing of a report of $\approx 15$ 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 “standard” page layout
- Language: German or English
- We expect the correct usage of spelling and grammar
  - $\geq 10$ errors per page $\implies$ abortion of correction
- Report template will be made available on seminar web page
Aims of this Seminar

Requirements on Talk

Your talk

- Talk of about 45 (= 40 + 5) 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
Aims of this Seminar

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
Important Dates

Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

T. Noll, C. Matheja: Analysis of Pointer Programs

T. Lange: Software Model Checking

C. Dehnert, M. Volk, H. Wu: Analysis of Probabilistic Systems

Final Hints
## Important Dates

### Deadlines

- 09.05.2016: Detailed outline of report due
- 13.06.2016: Report due
- 04.07.2016: Slides due
- 18./19.07.2016 (???): Seminar
Important Dates

Deadlines

- 09.05.2016: Detailed outline of report due
- 13.06.2016: Report due
- 04.07.2016: Slides due
- 18./19.07.2016 (???): Seminar

Missing a deadline causes immediate exclusion from the seminar
Seminar Topics

Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

T. Noll, C. Matheja: Analysis of Pointer Programs

T. Lange: Software Model Checking

C. Dehnert, M. Volk, H. Wu: Analysis of Probabilistic Systems

Final Hints
Seminar Topics

Selecting Your Topic

Procedure

- You obtain(ed) a list of topics of this seminar.
- Classified according to BSc/MSc level (or both).
- Indicate the preference of your topics (first, second, third).
- Return sheet by Friday (15 April) via e-mail/to secretary.
- We do our best to find an adequate topic-student assignment.
  - disclaimer: no guarantee for an optimal solution
- Assignment will be published on website by **18 April**.
- Please give language preference.
  - unsure $\Rightarrow$ German
Seminar Topics

Selecting Your Topic

Procedure

- You obtain(ed) a list of topics of this seminar.
- Classified according to BSc/MSc level (or both).
- Indicate the preference of your topics (first, second, third).
- Return sheet by Friday (15 April) via e-mail/to secretary.
- We do our best to find an adequate topic-student assignment.
  – disclaimer: no guarantee for an optimal solution
- Assignment will be published on website by 18 April.
- Please give language preference.
  – unsure ⇒ 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.
Analysis of Pointer Programs

Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

T. Noll, C. Matheja: Analysis of Pointer Programs

T. Lange: Software Model Checking

C. Dehnert, M. Volk, H. Wu: Analysis of Probabilistic Systems

Final Hints
Analysis of Pointer Programs

1. B: Introduction to Separation Logic [Noll]

- Logic for reasoning about programs that manipulate pointer data structures
- Extension of Hoare logic (correctness properties and proof rules)
- SL formula represents set of heap states
- Symbolic execution of programs on SL formulae
2. M: Separation Logic with Permissions [Noll]

Idea
- Threads acquire/release read and write permissions
- Read permission for shared read access
- Write permissions for exclusive write access

Observations
- Permission not available $\implies$ potential data race
- Permissions can always be acquired $\implies$ data-race freedom
Analysis of Pointer Programs

3. M: Concurrent Separation Logic [Noll]

\[
\begin{align*}
\{P_1\}C_1\{Q_1\} & \quad \{P_2\}C_2\{Q_2\} \\
\{P_1 \ast P_1\}C_1 & \parallel C_2\{Q_1 \ast Q_2\}
\end{align*}
\]

Concurrent Separation Logic (CSL)

- Extension of SL that allows independent reasoning about threads that access separate storage
- Proving soundness of CSL is a difficult problem
- Earlier approaches are based on non-standard semantics or are purely syntactic
- Paper presents new soundness proof for CSL in terms of standard operational semantics
Compositional Shape Analysis by Means of Bi-Abduction [Matheja]

- Compositional analysis: each procedure is analyzed independently of its callers
- Shape analysis: static analysis to discover and verify properties of heap manipulating programs
- Abduction: identify part ? of a formula to make implication \( \varphi \ast ? \rightarrow \psi \) valid
- Approach of this paper:
  - Heuristic to solve abduction problem of separation logic
  - Use abduction to obtain a compositional shape analysis generating pre/post-conditions for each procedure
  - Apply analysis to real-world programs: Linux Kernel, GIMP, Emacs, Sendmail...

This paper provides the theoretical foundations of a static analyzer developed and used at Facebook called Infer
Analysis of Pointer Programs

5. **M: Verification of Pointer Programs with Data by Forest Automata** [Matheja]

- **Setting:** C-like programs with dynamic data structures and integer data (e.g. binary search trees)
- **Goal:** Verify that a program successfully sorts a list, traverses a search tree...
- **Forest automata:** Extension of tree automata to accept graph-like structures
- **Approach of this paper:**
  - Extend forest automata to handle data structures with data
  - Develop a shape analysis based on forest automata
  - Apply analysis to several simple algorithms (e.g. binary search)
Software Model Checking

Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

T. Noll, C. Matheja: Analysis of Pointer Programs

T. Lange: Software Model Checking

C. Dehnert, M. Volk, H. Wu: Analysis of Probabilistic Systems

Final Hints
6. B: Abstraction in SMT-Based Unbounded Software Model Checking [Lange]

- Abstraction over data domains very successful
- Simple programs hard to verify
- Abstracted version of program easy to verify
- Combine abstraction of program and data
7. M: Configurable Software Verification [Lange]

Configurable SW Verification:
- Static Analysis (SA) and Verification reducible to each other
- SA knows generic algorithm for decades
- Won Goedel medal "for their contributions to the development of efficient verification methods and algorithms"

Adjustable Block Encoding:
- CEGAR hampered by large programs, especially sequences
- Simplify program by folding sequences [Beyer et al. 2009]
- Folding until minimality sometimes not very efficient, follow spirit of CPA and make it adjustable

- Invariants are at the heart of software verification
- Abduction: Inference of missing hypotheses
- Given known facts \( \Gamma \) and desired outcome \( \phi \), abductive inference finds "simple" explanatory hypothesis \( \psi \) such that

\[
\Gamma \land \psi \models \phi \text{ and } \text{SAT}(\Gamma \land \psi)
\]

- i.e. given invalid formula \( \Gamma \Rightarrow \phi \), find a "simple" formula \( \psi \) such that \( \Gamma \land \psi \Rightarrow \phi \) is valid and \( \psi \) does not contradict \( \Gamma \)
Analysis of Probabilistic Systems

Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

T. Noll, C. Matheja: Analysis of Pointer Programs

T. Lange: Software Model Checking

C. Dehnert, M. Volk, H. Wu: Analysis of Probabilistic Systems

Final Hints
9. B/M: Verification of MDPs Using Learning Algorithms [Dehnert]

**observation:** MDPs (probabilities + nondeterminism) used in various areas:

- randomized algorithms: leader election, mutual exclusion, ...  
- protocols: zeroconf, wlan, firewire, bluetooth, ...  
- (partially) unknown environments: planning (robots ‘n stuff), power management

**problem:** state space explosion

**idea:** apply techniques from AI to compute reachability probabilities

**approach:** modify Q-learning to work with unbounded, undiscounted probs.
10. B: Parametric Probabilistic Reachability [Volk]

- Given: DTMC
- Goal: compute probability to reach target state
10. B: Parametric Probabilistic Reachability [Volk]

- Given: DTMC
- Goal: compute probability to reach target state
- Use parameters instead of concrete values
10. B: Parametric Probabilistic Reachability [Volk]

- Given: DTMC
- Goal: compute probability to reach target state
- Use parameters instead of concrete values
- Perform state elimination

\[
\begin{align*}
\text{s}_0 & \rightarrow s_1 \\
1 - p & \rightarrow s_2 \\
p & \rightarrow s_1 \\
0.5 & \rightarrow s_3
\end{align*}
\]
10. B: Parametric Probabilistic Reachability [Volk]

- Given: DTMC
- Goal: compute probability to reach target state
- Use parameters instead of concrete values
- Perform state elimination

![Diagram with states and transitions](image)
11. M: Fault Tree Analysis [Volk]

- Dynamic Fault Trees (DFT) model system failures
11. M: Fault Tree Analysis [Volk]

- Dynamic Fault Trees (DFT) model system failures
- Analyse DFTs by I/O-IMCs:
  - Convert each element into corresponding MC
  - Apply parallel composition
12. M: Multi-Objective Model Checking [Volk]

- Given: MDP
- Goal: compute strategy to fulfill each property $\varphi_i$ with probability $\geq p_i$
12. M: Multi-Objective Model Checking [Volk]

- Given: MDP
- Goal: compute strategy to fulfill each property $\varphi_i$ with probability $\geq p_i$

$$\mathbb{P}(\diamond s_1) \geq 0.8$$
12. M: Multi-Objective Model Checking [Volk]

- Given: MDP
- Goal: compute strategy to fulfill each property $\varphi_i$ with probability $\geq p_i$

\[
P(\Diamond s_2) \geq 0.9
\]
12. M: Multi-Objective Model Checking [Volk]

- Given: MDP
- Goal: compute strategy to fulfill each property $\varphi_i$ with probability $\geq p_i$

\[
\mathbb{P}(\Diamond s_1) \geq 0.3 \land \mathbb{P}(\Diamond s_2) \geq 0.3
\]
12. M: Multi-Objective Model Checking [Volk]

- Given: MDP
- Goal: compute strategy to fulfill each property $\varphi_i$ with probability $\geq p_i$

$$\mathbb{P}$$ (♦ $s_1$) $\geq 0.4 \land \mathbb{P}$$ (♦ $s_2$) $\geq 0.4$$

An *interactive Markov chain* is a tuple $I = (S, Act, \rightarrow, \rightarrow, s_0)$, where

- $S$ is a nonempty set of states with *initial state* $s_0 \in S$,
- $Act$ is a set of actions,
- $\rightarrow \subseteq S \times Act \times S$ is a set of *interactive* transitions, and
- $\rightarrow \subseteq S \times \mathbb{R}_{>0} \times S$ is a set of *Markovian* transitions.

The operators are defined on the IMCs such as:

- parallel composition $I_1 \parallel_A I \in w.r.t$ to a synchronization set $A \in Act$,
- hiding $I \setminus H$ w.r.t to a hiding set $H \in Act$.

The interesting questions are:

- How to analysis the IMC?
- How make the IMC smaller?
- etc.
A Markov automaton (MA) is a tuple $\mathcal{M} = (S, s_0, Act, \rightarrow, \Rightarrow)$, where

- $S$ is a countable set of states with initial state $s_0 \in S$,
- $Act$ is a countable set of actions,
- $\rightarrow \subseteq S \times Act \times Distr(S)$ is the interactive probabilistic transition relation,
- $\Rightarrow \subseteq S \times \mathbb{R}_{>0} \times S$ is the Markovian transition relation.

How we can compute the following properties on the MA?

- The expected time to reach a set of target states
- The long-run average time spend in a set of target states
- The time-bounded reachability to reach a set of target states within a given time interval
Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

T. Noll, C. Matheja: Analysis of Pointer Programs

T. Lange: Software Model Checking

C. Dehnert, M. Volk, H. Wu: Analysis of Probabilistic Systems
## Final Hints

### Some Final Hints

<table>
<thead>
<tr>
<th>Hints</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Take your time to <strong>understand</strong> your literature.</td>
</tr>
<tr>
<td>• Be <strong>proactive</strong>! Look for <strong>additional</strong> literature and information.</td>
</tr>
<tr>
<td>• Discuss the content of your report with other students.</td>
</tr>
<tr>
<td>• Be <strong>proactive</strong>! Contact your supervisor <strong>on time</strong>.</td>
</tr>
<tr>
<td>• Prepare the meeting(s) with your supervisor.</td>
</tr>
<tr>
<td>• Forget the idea that you can prepare a talk in a day or two.</td>
</tr>
</tbody>
</table>
Final Hints

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!