Seminar *Principles of Programming Languages*

Introduction

Summer Semester 2015; 9 April 2015

Thomas Noll

Software Modeling and Verification Group

RWTH Aachen University

`http://moves.rwth-aachen.de/teaching/ss-15/popl/`

Software Modeling and Verification Chair

RWTH AACHEN UNIVERSITY

Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

Program Analysis

Model Checking

Probabilistic Systems

Concurrent Systems

Separation Logic

Final Hints

Overview

Principles of Programming Languages

Principles of Programming Languages

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

Principles of Programming Languages

Principles of Programming Languages

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

Aspects

- Program Analysis
  - Static Program Analysis (WS 2014/15)
  - Semantics and Verification of Software (SS 2013/2015)
- Model Checking
  - Introduction to Model Checking (WS 2013/14, SS 2015)
  - Advanced Model Checking (SS 2014)
- Probabilistic Systems
  - Modeling and Verification of Probabilistic Systems (SS 2014)
- Concurrent Systems
  - Concurrency Theory (WS 2013/14)
- Separation Logic

Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

Program Analysis

Model Checking

Probabilistic Systems

Concurrent Systems

Separation Logic

Final Hints

Software Modeling and Verification Chair
RWTH AACHEN UNIVERSITY

Aims of this Seminar

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 $\approx$ 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 $\Longrightarrow$ abortion of correction
- Report template will be made available on seminar web page

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

Overview

Aims of this Seminar

Important Dates

Seminar Topics

Program Analysis

Model Checking

Probabilistic Systems

Concurrent Systems

Separation Logic

Final Hints

Important Dates

Deadlines

- 18.05.2015: Detailed outline due
- 01.06.2015: Report due
- 15.06.2015: Final version of report due
- 29.06.2015: Slides due
- 06.07.2015: Final version of slides due
- 13./14.07.2015: Seminar

Important Dates

Deadlines

- 18.05.2015: Detailed outline due
- 01.06.2015: Report due
- 15.06.2015: Final version of report due
- 29.06.2015: Slides due
- 06.07.2015: Final version of slides due
- 13./14.07.2015: Seminar

Missing a deadline causes immediate exclusion from the seminar

Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

Program Analysis

Model Checking

Probabilistic Systems

Concurrent Systems

Separation Logic

Final Hints

Selecting Your Topic

Procedure

- You obtain(ed) a list of topics of this seminar.
- Indicate the preference of your topics (first, second, third).
- Return sheet by next Monday (13 April) via e-mail/to secretary.
- We do our best to find an adequate topic-student distribution.
- Disclaimer: no guarantee for an optimal solution.
- Assignment will be published on website by 15 April.
- 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).
- Return sheet by next Monday (13 April) via e-mail/to secretary.
- We do our best to find an adequate topic-student distribution.
- Disclaimer: no guarantee for an optimal solution.
- Assignment will be published on website by 15 April.
- 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

Overview

Aims of this Seminar

Important Dates

Seminar Topics

Program Analysis

Model Checking

Probabilistic Systems

Concurrent Systems

Separation Logic

Final Hints

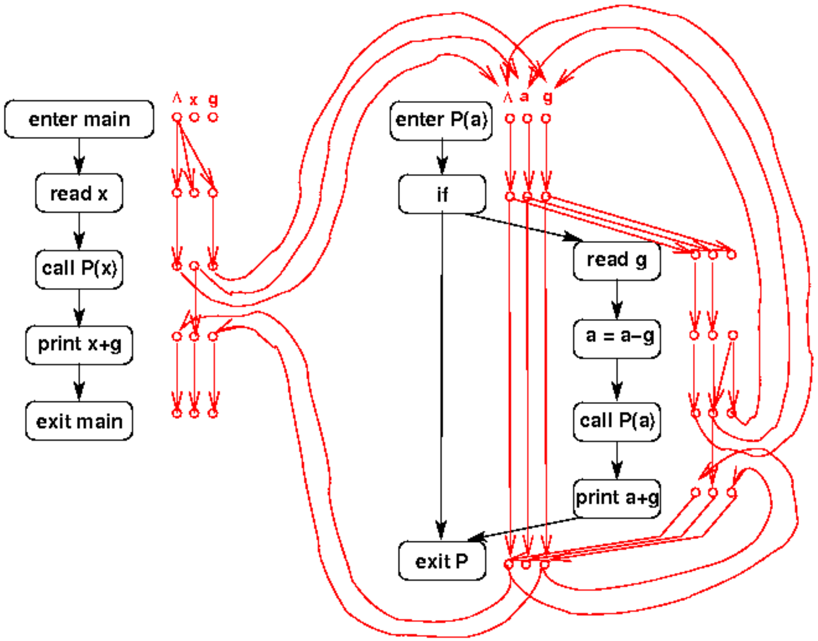
1: Quantitative Interprocedural Analysis

Interprocedural analysis: analyses data and control flow across procedure/method boundaries

- constant propagation
- index values
- ...

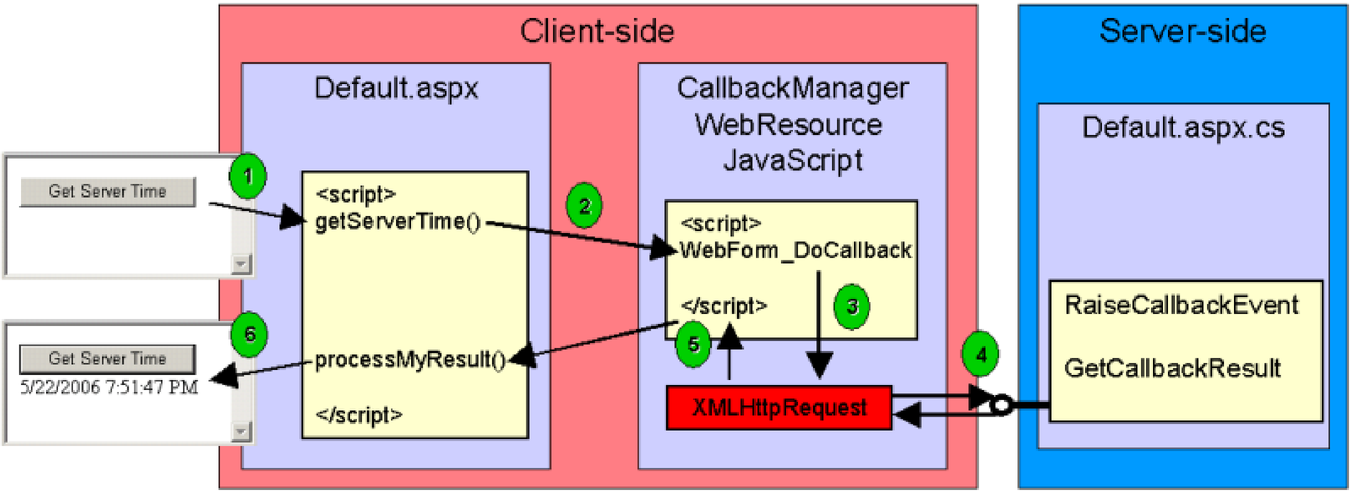
Here: extension by edge weights

- worst-case execution time
- average energy consumption
- ...



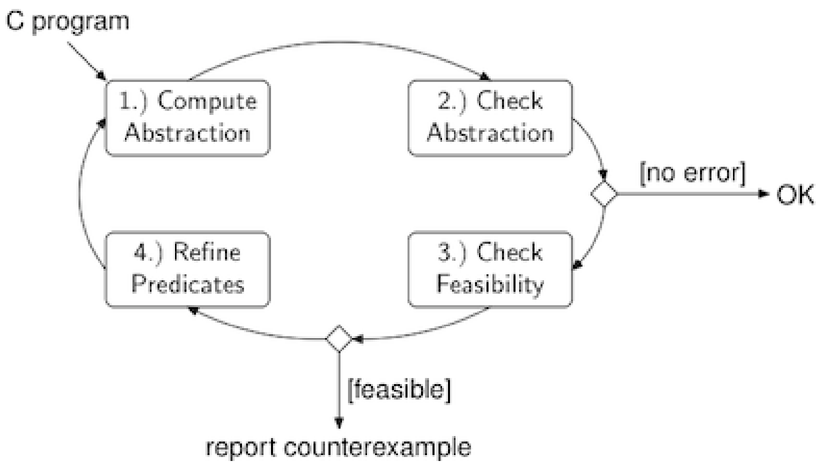
2: Interprocedural Analysis with Callbacks

- Callbacks interrupt standard information flow
- Reachability relationships between procedure nodes cannot be fully determined during procedure analysis
- Procedure code may have to be inspected again during analysis of client code
- Here: formal model (tree-adjoining-language reachability)



3: Abstraction Refinement

- Goal: model checking large programs
- Successful approach: counterexample-guided abstraction refinement (CEGAR) based on predicates over program variables
- Challenge: identification of parsimonious abstractions
- Here: Craig interpolation to efficiently construct relevant predicates



Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

Program Analysis

Model Checking

Probabilistic Systems

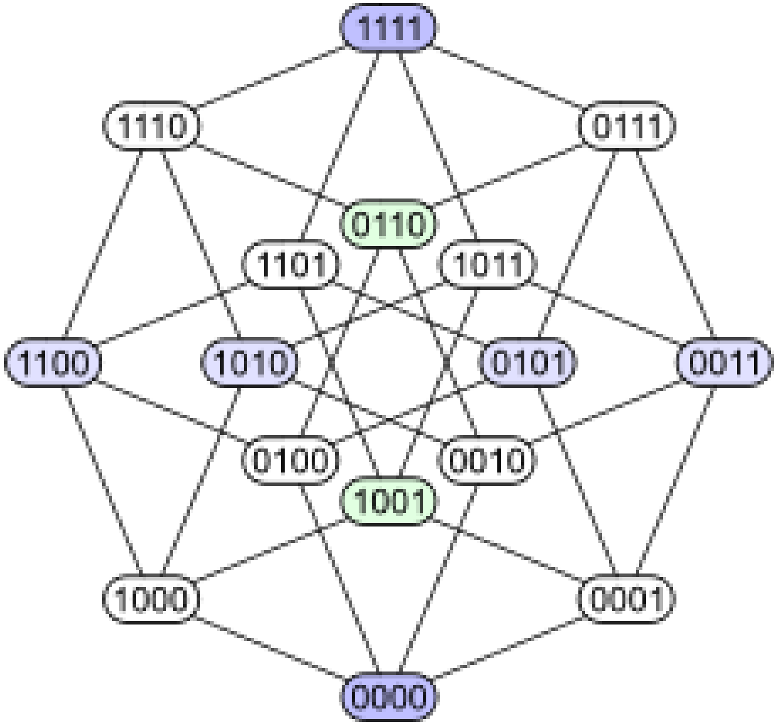
Concurrent Systems

Separation Logic

Final Hints

4: Partial Order Reduction

- Setting: program verification by stateless model checking
- Suffers from exponential growth in number of explored executions
- Goal: reducing this number while maintaining complete coverage
- Approach: Dynamic Partial Order Reduction (DPOR)



5: Bounded Model Checking

- Bounded verification: proving correctness of program behaviour up to certain execution length (usually incorrect)
- Most prominent technique: bounded model checking based on automata
  - language generated by system is disjoint from language of bad traces
- Doable if language closed under Boolean operations and emptiness problem decidable (“perfect”)
- Here: languages accepted by multi-head pushdown automata are perfect modulo bounded languages
- Can encode well-established system models:
  - recursive multi-threaded programs
  - recursive counter machines
  - communicating finite-state machines

6: Software Model Checking via IC3

- Goal: establish invariants of software programs
- IC3: new verification technique for analysis of sequential circuits
- Incrementally over-approximates state space, refuting potential violations to the property
- Here: first adaptation to software verification
  - generalisation SAT $\rightarrow$ SMT to support symbolic transition systems
  - tree-like search on control-flow graph of program
  - lazy abstraction with interpolants

Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

Program Analysis

Model Checking

Probabilistic Systems

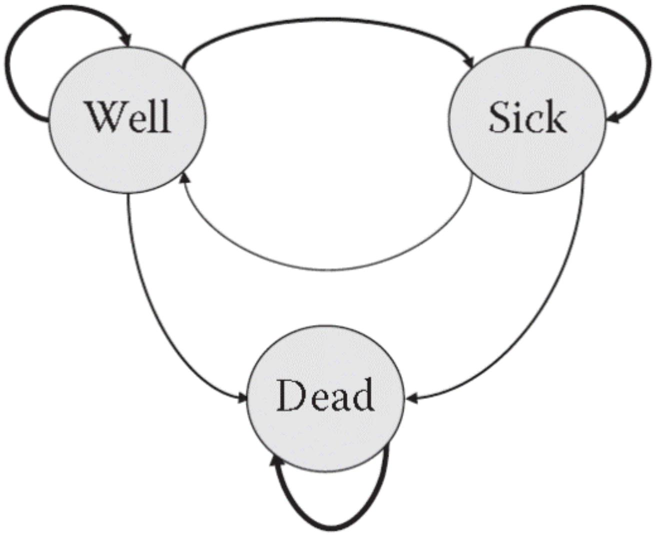
Concurrent Systems

Separation Logic

Final Hints

7: Probabilistic Termination

- Framework to prove “almost sure” termination for probabilistic programs with real-valued variables
- Based on ranking supermartingales (probabilistic analogous to ranking functions on non-probabilistic programs)
- Proven sound and complete for programs involving randomisation and bounded nondeterminism



8: Abstract Semantics of Probabilistic Programs

- Goal: analysis of programs like

```
let x := flip 0.5; y := flip (if x = heads then 0.5 else 0.3) in <x, y>
```

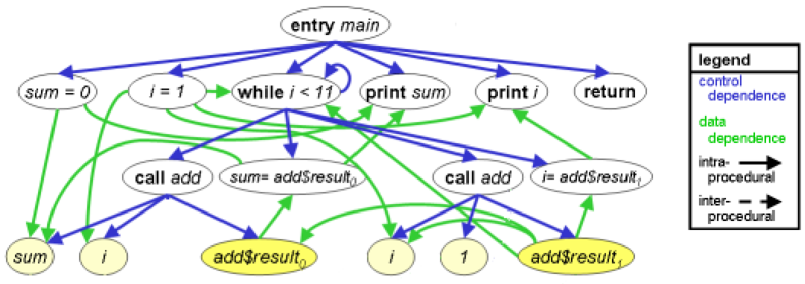
- Meaning: probability mass function

$$p := [\langle heads, heads\rangle \mapsto 0.25, \langle heads, tails\rangle \mapsto 0.25,$$
$$\langle tails, heads\rangle \mapsto 0.15, \langle tails, tails\rangle \mapsto 0.35]$$

- Here: measure-theoretic semantics for a probabilistic language with recursion
- Abstract semantics to ensure computability
- Applications:
  - Bayesian inference
  - stochastic ray tracing (rare event simulation)
  - probabilistic verification of floating-point error bounds

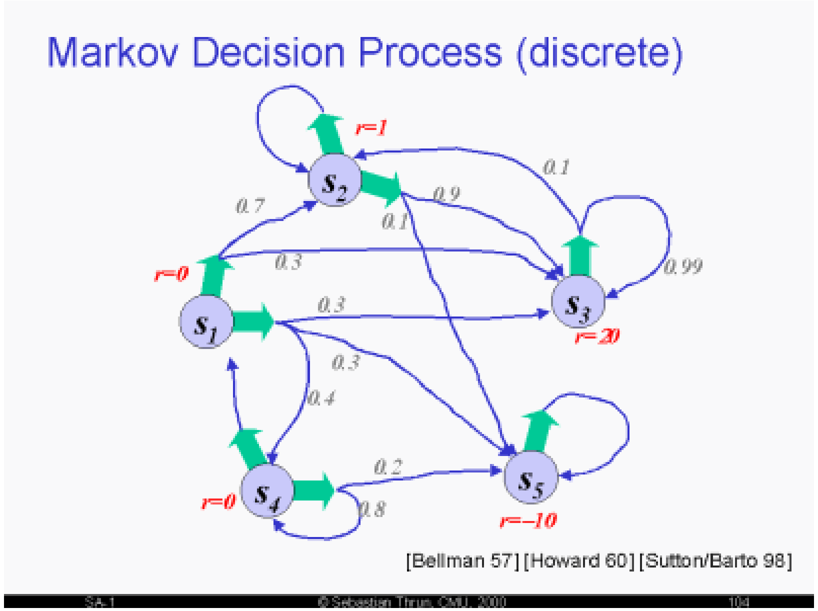
9: Slicing Probabilistic Programs

- Slicing: which outputs depend on which inputs?
  - interesting output values define slicing criterion
  - backward analysis of information flow based on program dependence graph
- Applications:
  - Debugging
  - Testing
  - Model checking
- Here: adaptation to probabilistic programs
  - usual notions of control dependence and data dependence not sufficient
  - introduces new observe dependence



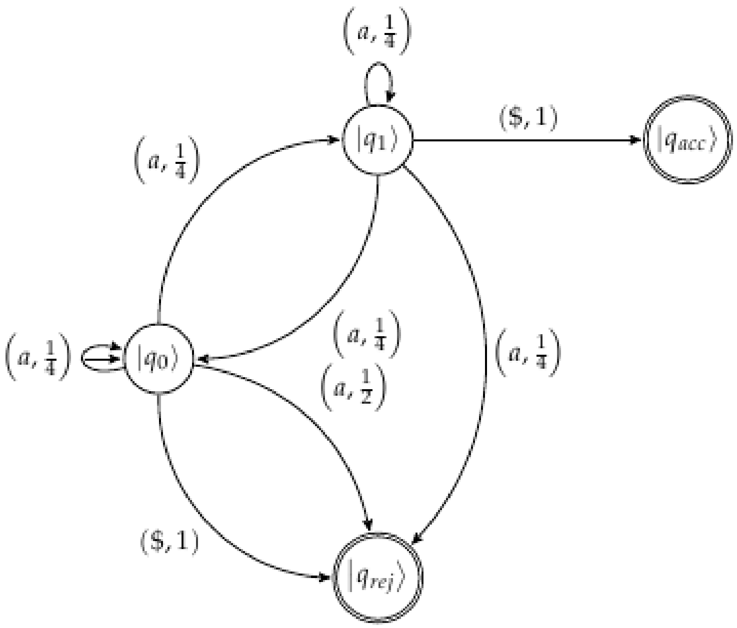
10: Analysis of Markov Decision Processes

- Markov Decision Processes (MDP): formal model to integrate non-deterministic and probabilistic choices
- Interesting property: minimal/maximal probabilities to reach set of target states (with respect to policy resolving non-determinism)
- Algorithm: value iteration (iteratively finding the probabilities of paths of increasing length)
- Problems:
  - definition of stopping criterion to ensure bound on approximation
  - analysis of convergence rate
  - estimate required number of iterations
- Here: interval iteration algorithm based on graph analysis and transformation of MDPs



11: Probabilistic Automata on Finite Words

- Classic automaton: state and input symbol determine next state, string accepted in final state
- Probabilistic automaton: state and input symbol give distribution over next states
- Question: given $\lambda \in [0, 1]$, what strings are accepted with probability $\geq \lambda$?
- Here: problem generally undecidable, but decidable for new subclass of sharp acyclic automata



Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

Program Analysis

Model Checking

Probabilistic Systems

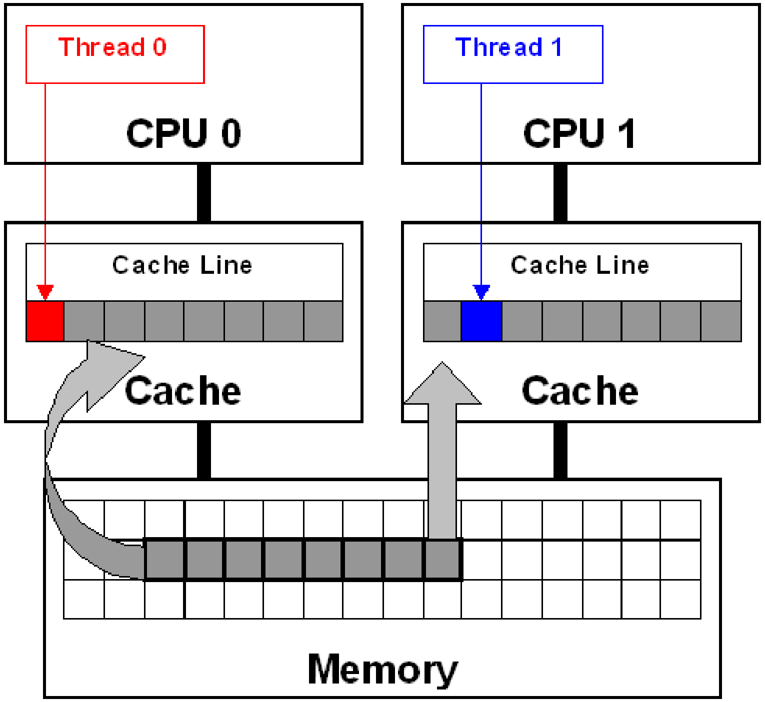
Concurrent Systems

Separation Logic

Final Hints

12: Verification of Multithreaded Programs

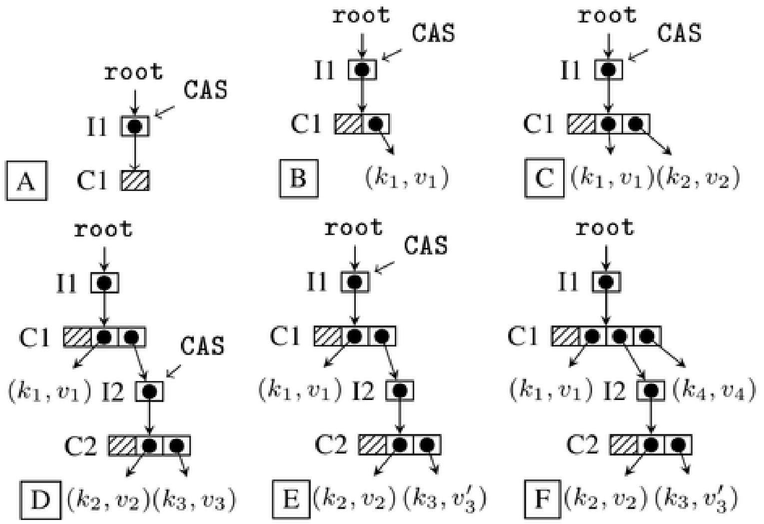
- Programming model: threads with recursive procedures, shared counters and finite variables
- Unrestricted verification undecidable in this setting
- Here: restriction to program executions that follow a given pattern
  - pattern = regular expression over program actions
  - actions = reads and writes to shared storage
- Multiparameter analysis of problem complexity depending on
  - number of threads/counters/variables
  - maximal size of threads
  - size of the pattern
  - maximal number of procedures per thread
  - ...



Software Modeling and Verification Chair
RWTH AACHEN UNIVERSITY

13: Linearisability of Concurrent Data Structures

- Goal: efficient implementation of data structures (queues, stacks, hash tables) supporting concurrent access
- Standard correctness criterion: linearisability
  - concurrent object is linearisable if its operations appear to occur at some instant between their invocation and return ("linearisation point")
- Proof techniques usually tailored to specific data structures
- Here: universal approach based on (backward) simulation techniques
- Soundness and completeness



14: Quantitative Quiescent Consistency

- Observation: linearisability imposes performance penalty which scales linearly in number of contending threads
- Solution: relaxation of consistency requirements
- Example: quiescent consistency
  - allows effect to become visible after return from operation
  - operations separated by a period of quiescence appear to take effect in real-time order
  - but says very little about executions that have any contention
- Here: quantitative quiescent consistency (QQC)
  - another relaxation of linearisability
  - degree of relaxation is proportional to degree of contention

Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

Program Analysis

Model Checking

Probabilistic Systems

Concurrent Systems

Separation Logic

Final Hints

15: Complexity of Deciding Entailment

- Separation Logic:
  - logic for reasoning about programs that manipulate pointer data structures
  - extension of Hoare logic (proof triples and proof rules)
- Symbolic execution of programs on SL formulae representing sets of program states
- State-space exploration requires checking subset relationships $\implies$ entailment
- Here: decidability and computational complexity of entailment in certain fragments of SL

$$\frac{\{P\}\ C\ \{Q\}}{\{P * R\}\ C\ \{Q * R\}}\ \mathsf{mod}(C) \cap \mathsf{fv}(R) = \emptyset$$

16: Deciding Satisfiability

- Formula is satisfiable iff it is not contradictory
- Thus: important criterion for validating logical specifications
- Here: decidability and computational complexity of satisfiability of certain fragments of SL

Outline

Overview

Aims of this Seminar

Important Dates

Seminar Topics

Program Analysis

Model Checking

Probabilistic Systems

Concurrent Systems

Separation Logic

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.

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!