Foundations of Probabilistic Programming/Program Synthesis

Introduction to seminar
Summer Semester 2019; April 3, 2019
Joost-Pieter Katoen, Thomas Noll et al.
Software Modeling and Verification Group
RWTH Aachen University

https://moves.rwth-aachen.de/teaching/ss-19/fpp/
https://moves.rwth-aachen.de/teaching/ss-19/pgmsyn/
Overview

Outline

Overview

Aims of this Seminar

Important Dates

The *Foundations of Probabilistic Programming* Topics

The *Program Synthesis* Topics

Final Hints
Overview

Foundations of Probabilistic Programming

Probabilistic Programs

Probabilistic programs extend sequential programs with \textbf{random sampling} and (hard and soft) \textbf{conditioning}. They are used in machine learning, robotics, computer vision, etc. Almost every programming language (C, Python, Prolog, etc.) has a probabilistic extension.
Overview

Foundations of Probabilistic Programs

• What do probabilistic programs exactly mean? Semantics.
• How to reason about probabilistic programs? Correctness.
• When does a probabilistic program terminate? Termination.
• How long does a probabilistic program run? Complexity.
• When do two programs behave (almost) the same? Equivalence.
Program Synthesis

Program synthesis is the task of automatically finding a program in the underlying programming language that satisfies the user intent expressed in the form of some specification.
Overview

Dream of Program Synthesis

Programming language → Synthesiser → User intent

Challenges

- Intractability of program space
- Diversity of user intent

Program

Foundations of Probabilistic Programming/Program Synthesis
Joost-Pieter Katoen
Summer Semester 2019; April 3, 2019
Overview

Dream of Program Synthesis

Programming language → Synthesiser ← User intent

Challenges

- Intractability of program space
- Diversity of user intent
Overview

Applications of Program Synthesis

Data wrangling: cleaning, transforming, and/or preparing data for analysis and presentation
  • often using programming-by-example (e.g., john.doe@rwth.de → John Doe)

Graphics: synthesis of programs for constructing graphical objects to enable interactive editing experiences and efficient animations

Code Repair: given buggy program $P$ that violates specification $\varphi$, find new $P'$ that satisfies $\varphi$

Code Suggestions: advanced form of “autocompletion”

Probabilistic modelling: learning of probabilistic programs that explain empirical evidence

Superoptimisation: synthesising an optimal sequence of instructions
  • e.g., computing average of unsigned integers $x$ and $y$ without conditionals: $(x + y)/2$ can overflow, but $(x \mid y) - ((x \oplus y) \gg 1)$ works

Concurrent Programming: determine placement of minimal synchronisation constructs to avoid data races
Aims of this Seminar

Outline

Overview

Aims of this Seminar

Important Dates

The *Foundations of Probabilistic Programming* Topics

The *Program Synthesis* Topics

Final Hints
Aims of this Seminar

Goals

Aims of this seminar

- **Independent understanding** of a scientific topic
- Acquiring, reading and understanding **scientific literature**
  - given references sufficient in most cases
- Writing of your **own report** on this topic
  - far more that just a translation/rewording
  - usually an "**extended subset**" of paper
    - "subset": present core ideas and omit too specific details
    - "extended": more extensive explanations, examples, ...
    - discuss with supervisor!
- **Oral presentation** of your results
  - can be "proper subset" of report
  - generally: less (detailed) definitions/proofs and more examples
Aims of this Seminar

Requirements on Report

Your report

- Independent writing of a report of **10–15 pages**
- First milestone: **detailed outline**
  - but: overview of structure (section headers, main definitions/theorems) and initial part of main section (one page)
- **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
- **\LaTeX template** will be made available on seminar web page
Aims of this Seminar

Requirements on Talk

Your talk

- Talk of **30 minutes**
- Available: projector, presenter, [laptop]
- Focus your talk on the **audience**
- **Descriptive** slides:
  - \( \leq 15 \) lines of text
  - use (base) colors in a useful manner
  - number your slides
- **Language**: German or English
- No spelling mistakes please!
- Finish **in time**. Overtime is bad
- Ask for **questions**
- Have **backup slides** ready for expected questions
- **\LaTeX/beam**er template** will be made available on seminar web page
Important Dates

Outline

Overview

Aims of this Seminar

Immediate Dates

The Foundations of Probabilistic Programming Topics

The Program Synthesis Topics

Final Hints
# Important Dates

## Deadlines

- 6 May: Detailed outline of report due
- 3 June: Full report due
- 1 July: Presentation slides due
- 11 July (afternoon): Seminar

Missing a deadline causes immediate exclusion from the seminar.
Important Dates

Deadlines

- 6 May: Detailed outline of report due
- 3 June: Full report due
- 1 July: Presentation slides due
- 11 July (afternoon): Seminar

Missing a deadline causes immediate exclusion from the seminar
Important Dates

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 here or **by Friday (5 April)** via e-mail (noll@cs.rwth-aachen.de) or 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 web site next week.
Important Dates

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 here or by Friday (5 April) via e-mail (noll@cs.rwth-aachen.de) or 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 web site next week.

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.
The Foundations of Probabilistic Programming Topics

Outline

Overview

Aims of this Seminar

Important Dates

The Foundations of Probabilistic Programming Topics

The Program Synthesis Topics

Final Hints
The Foundations of Probabilistic Programming Topics

Logical essentials of Bayesian reasoning

Book chapter: B. Jacobs and F. Zanasi: *Logical essentials of Bayesian reasoning*, 2019

Goal: understanding the essence of Bayesian inference

Programs: a uniform, structured and expressive language for describing Bayesian phenomena in terms of familiar programming concepts, like channel, predicate transformation and state transformation

Approach: channel-based approach to Bayesian probability theory

Example:
Probabilistic couplings

Book chapter: J. Hsu and G. Barthe: *Probabilistic Couplings from Program Logics*, 2019

Goal: Prove whether two probabilistic programs behave (almost) the same.

Programs: Probabilistic programs without conditioning


Example:

\[
k \leftarrow \text{Flip}; \\
r \leftarrow k \oplus s \\]

\[
k \leftarrow \text{Flip}; \\
r \leftarrow k \\]

If the distribution of \( r \) is the same in both programs, then the XOR cipher is secure.
Termination analysis with martingales

Book chapter: K. Chatterjee et al.: *Termination Analysis of Probabilistic Programs with Martingales*, 2019

Goal: How to check whether:
- A probabilistic program terminates with probability one?
- A probabilistic program terminates in finitely many expected number of steps?

Programs: finite/infinite probabilistic choice, non-determinism

Approach: martingale theory, partial automation (algorithms)

Example:

```
1: x := 100;
2: while x ≥ 0 do
3:    if prob(0.5) then
4:       x := x + 1
5:     else
6:       x := x - 1
7:    fi;
8: od
```
The Foundations of Probabilistic Programming

Improving program testing by probabilistic programming


Goal: How to efficiently generate test cases for random testing of programs?

Programs: the probabilistic domain-specific language Luck in which test generators are conveniently expressed. Predicates with lightweight annotations to control the distribution of generated values and the amount of constraint solving needed.

Approach: • formal semantics to the language Luck
  • soundness and completeness of random generation w.r.t. a standard predicate semantics
  • case studies showing effectiveness of bug-finding in programs

Example:
The Foundations of Probabilistic Programming Topics

Concentration measures

Book chapter: S. Sankaranarayanan: Quantitative Analysis of Programs with Probabilities and Concentration of Measure Inequalities, 2019

Goal: Quantitative analysis of probabilistic programs, in particular concentration of measure inequalities to reason about bounds on the probabilities of assertions.

Technique: measure inequalities, i.e., a set of mathematical ideas that characterize how functions of random variables deviate from their expected value.

Example:

```
1 angles = [10, 60, 110, 160, 140, ... 100, 60, 20, 10, 0]
2 x := TransGaussian(0, 0.05, -0.5, 0.5)
3 y := TransGaussian(0, 0.1, -0.6, 0.5)
4 for reps in range(0, 100):
5   #Iterate through angles
6   for theta in angles:
7     # Distance travelled variation
8     d = Uniform(0.95, 1.02)
9     # Steering angle variation
10    t = deg2rad(theta) * (1 + ... )
11    TransGaussian(0, 0.01, -0.05, 0.05)
12    # More distance d with angle t
13   x = x + d * cos(t)
14   y = y + d * sin(t)
15 #Probability that we went too far?
16  assert(x >= 277)
```
The *Program Synthesis* Topics

Outline

Overview

Aims of this Seminar

Important Dates

The *Foundations of Probabilistic Programming* Topics

The *Program Synthesis* Topics

Final Hints
The Program Synthesis Topics

Syntax-guided synthesis


Goal: synthesizing implementations given by grammar template from declarative specifications

Specification: background theory, semantic correctness specification given by logical formula, syntactic set of candidate implementations given by grammar

Output: implementation satisfying the specification in the theory

Technique: counter-example-guided-inductive-synthesis (CEGIS)

Example: specification

\[
\begin{align*}
\max(x, y) \geq x & \land \max(x, y) \geq y \land (\max(x, y) = x \lor \max(x, y) = y) \\
\end{align*}
\]

with candidate grammar

\[
E \rightarrow E + E \mid E - E \mid E \leq E \mid E?E : E \mid 0 \mid 1
\]

(interpreted over integers) yields implementation

\[
x \leq y ? y : x
\]
The Program Synthesis Topics

Synthesis of pointer programs

**Paper:** N. Polikarpova, I. Sergey: *Structuring the synthesis of heap-manipulating programs*, POPL 2019

**Goal:** synthesizing imperative programs with pointers from declarative specifications

**Programming language:** simple pointer-manipulating language

**Specification:** pair of assertions \((\varphi, \psi)\) (pre- and postcondition) from Separation Logic (= Hoare Logic + pointers)

**Output:** program that transforms any heap state satisfying \(\varphi\) into one satisfying \(\psi\)

**Technique:** deductive search

**Example:** specification

\[
\{ x \mapsto a \; y \mapsto b \} P \{ x \mapsto b \; y \mapsto a \}
\]

yields program

\[
a1 = \ast x; \text{let } b1 = \ast y; \ast y = a1; \ast x = b1
\]
The Program Synthesis Topics

Synthesis of probabilistic programs


Goal: synthesising probabilistic programs from real-world datasets

Programming language: imperative programs with sampling and conditioning

Specification: data set and “sketch” of probabilistic program with “holes”

Output: replacement of holes with program fragments such that execution is consistent with data

Technique: Markov Chain Monte Carlo based synthesis

Example: sketch

\[ c = ??(a, b) \]

may be replaced by

\[ c = (a > b) \]
Final Hints

Outline

Overview

Aims of this Seminar

Important Dates

The *Foundations of Probabilistic Programming* Topics

The *Program Synthesis* Topics

Final Hints
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.
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!