ECE.750 T29

Automated Program Verification (APV)
Fall 2018

Prof. Arie Gurfinkel

UNIVERSITY OF WATERLOO

Course Time and Location

Date: Friday

Location: EIT-3151

Time: 8:30 – 11:20 AM

No Lecture September 28, 2018

Instructor and TA

Instructor

- Prof. Arie Gurfinkel

Teaching Assistant

- None

Course Web Page

- https://ece.uwaterloo.ca/~agurfink/ece750t29
- LEARN: https://learn.uwaterloo.ca
- SLACK: uw-apv.slack.com
- GitHub: https://github.com/uw-apv

Topics: Automated Program Analysis

Introduction to Model Checking

SAT and SAT-based Bounded Model Checking

Unbounded SAT-based MC: k-induction and interpolation

Property Directed Reachability

Constrained Horn Clauses: From Hardware to Software

Solving Constrained Horn Clauses over Arithmetic

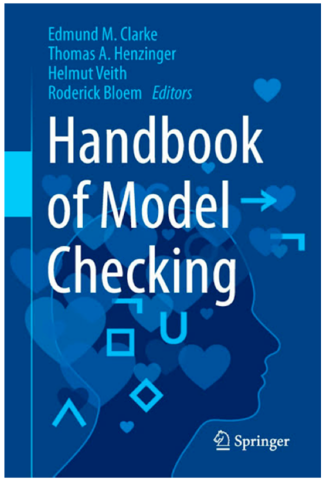
Safety Verification of Push-Down Systems

Machine Learning-based Algorithms for Program Analysis

PLUS

Additional topics based on project ideas

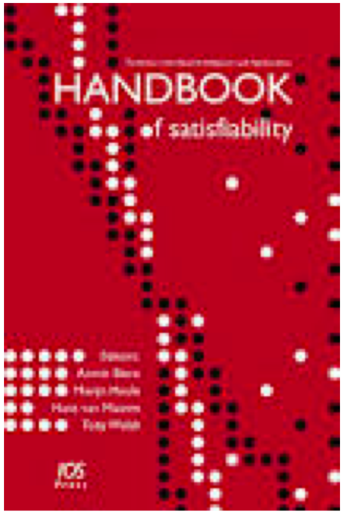
Textbooks



No textbooks are required

Material will be based on:

- Handbook of Model Checking
  - https://link.springer.com/book/10.1007%2F978-3-319-10575-8



- Handbook of Satisfiability
- Research papers
  - expect to read 2-3 papers for each class

Course “style”

Seminar-style course

In class presentations of basic foundations

In class discussion (read required papers!)

Homework assignments

Research Project (50% of the final grade)

Project

Goals

- improve research skills (understanding, synthesizing, creating, explaining)
- develop a deeper understanding of an area in Automated Verification

Project Types

- Review
  - critical overview of a topic. At least 3-4 papers. Must have “value added”. Not just a summary of the papers
- Application
  - apply an existing automated verification tool to an interesting problem domain (e.g., information flow or ML algorithm)
- Implementation
  - implement an existing algorithm and compare/reproduce results (e.g., mini-PDR, interpolation for SMT)
- Creative
  - Propose new theory / algorithm / technique. Prove and/or prototype.

Automated Verification Conferences

Look at recently published papers for project ideas

Computer Aided Verification (CAV)

Formal Methods for Computer-Aided Design (FMCAD)

Principles of Programming Languages (POPL)

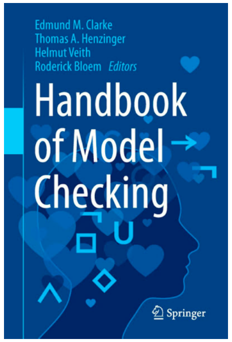
Verification, Model Checking, and Abstract Interpretation (VMCAI)

Tools and Algorithms for Construction and Analysis of Systems (TACAS)

Automated Technology for Verification and Analysis (ATVA)

…

and Handbook of Model Checking



Project: Logistics

Project proposal to be approved by the instructor (~700-1000 words)

Presentation on project background (towards end of lectures)

- ~30 minutes
- present background / area / papers on which the project is based

Project Report

- ~15-20 pages in LNCS format
- https://www.springer.com/gp/computer-science/lncs/conference-proceedings-guidelines

Project presentation

- ~20 minutes
- We will have a conference day after end of classes for project presentations

Grades

Assignments: 30%

Class participation (questions & discussions): 10%

Background Presentation: 10%

Course Project: 50%

Grades may be curved or adjusted at the Instructor’s discretion

Course Website & LEARN

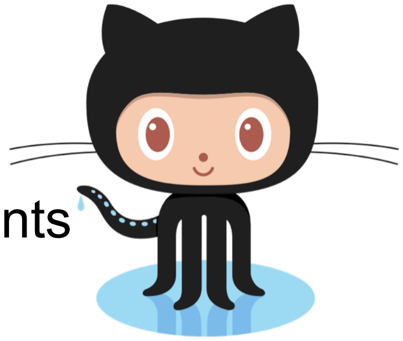
The course website is the definitive source

- When in doubt, consult the web page

YOUR responsibility to check for updates!

- Course website: https://ece.uwaterloo.ca/~agurfink/ece750t29/
- LEARN (http://learn.uwaterloo.ca)

GitHub and Slack



We will use **GitHub** for managing and submitting assignments

- This requires a free GitHub account

We will use **Slack** for communication (don’t use EMAIL if you can!)

- https://uw-apv.slack.com/
- there are slack apps for Win/Mac/Linux/iPhone/Android – use them!
- Signup link is available on LEARN (or use your @uwaterloo.ca email)
- monitor #announcements for course announcements
- ask questions about assignments in #assignments
- invite @prof-arie to a channel for a private question
- Share cool slack features that you find helpful with the rest of the class

Independent Work

All work turned in must be of that individual student unless stated otherwise.

Violations will result in zero credit to all students concerned. University of Waterloo Policy 71 will be followed for any discovered cases of plagiarism.

Policy on Late Assignments

You have 2 days of lateness for assignments that you can use throughout the term

- These are TWO days for the term. Not for each assignment!

Each day the assignment is late consumes one day of lateness

For example,

- You can be 2 days late on assignment A1, or
- One day late on A1, and one day late on A3, or
- You can hand all of the assignments on time ☺

Contact

Office Hours

- by appointment
- best time is after lectures

Use Slack to communicate

- but, if you don't get a reply, send an email

Email (email address on the course web page)

- https://ece.uwaterloo.ca/~agurfink/ece750t29
- Identify yourself
  - Originated from your uwaterloo email address, or
  - Signed with your full name and student ID
- Start **Subject** of email with **[ECE750t29]**

My Expectations

Attend lectures

- talk to classmates if you are away!

Participate

- during discussions and activities

Be professional

- questions in class, slack, email, discussion on LEARN, interacting with TA, …

A little about me

2007, PhD University of Toronto

2006-2016, Principal Researcher at Software Engineering Institute, Carnegie Mellon University

Sep 2016, Associate Professor, University of Waterloo

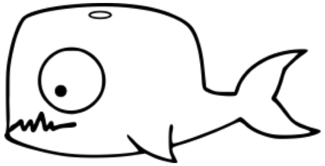
వింత

UFO

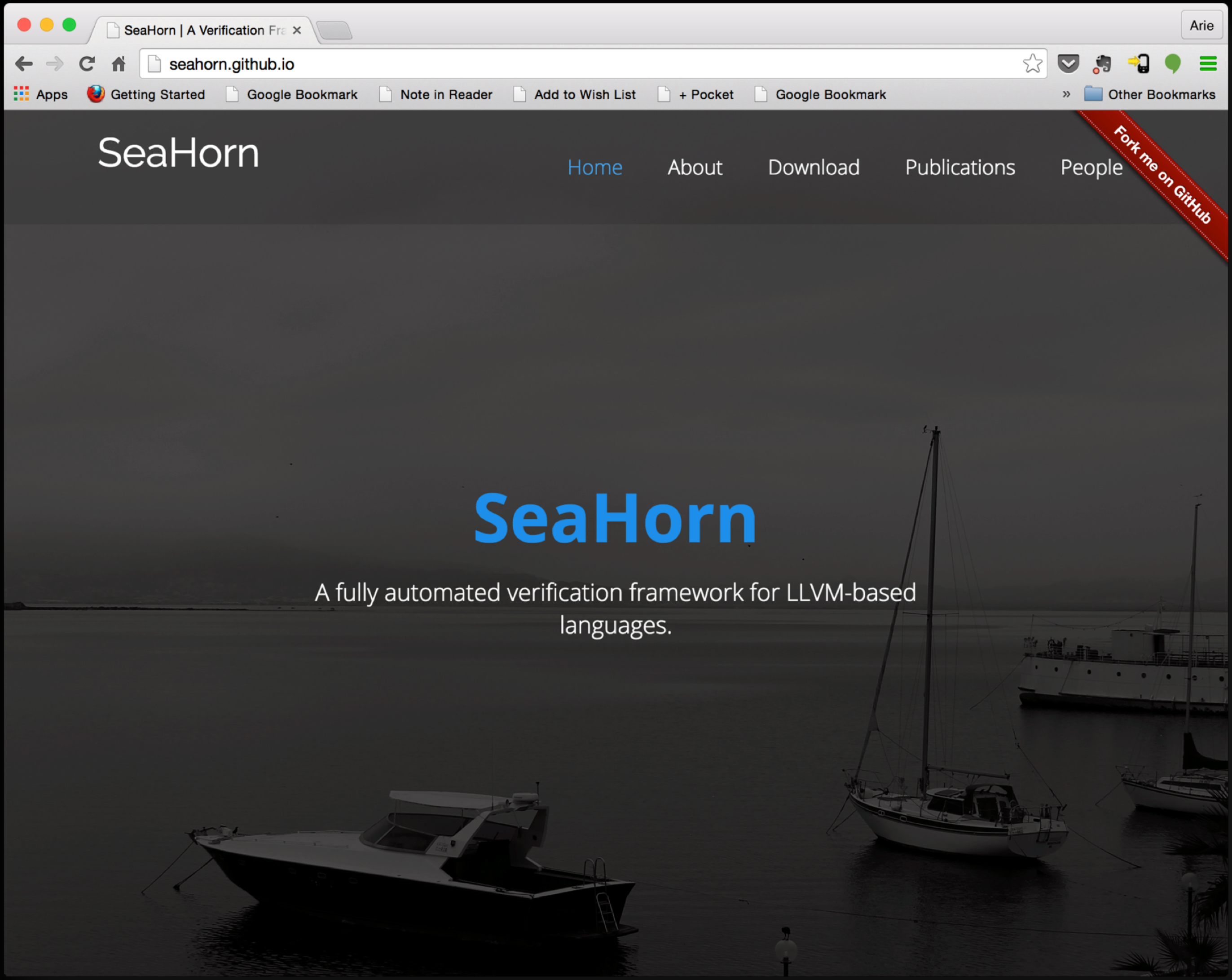
FrankenBit

SPACER

Avy



SeaHorn



SeaHorn

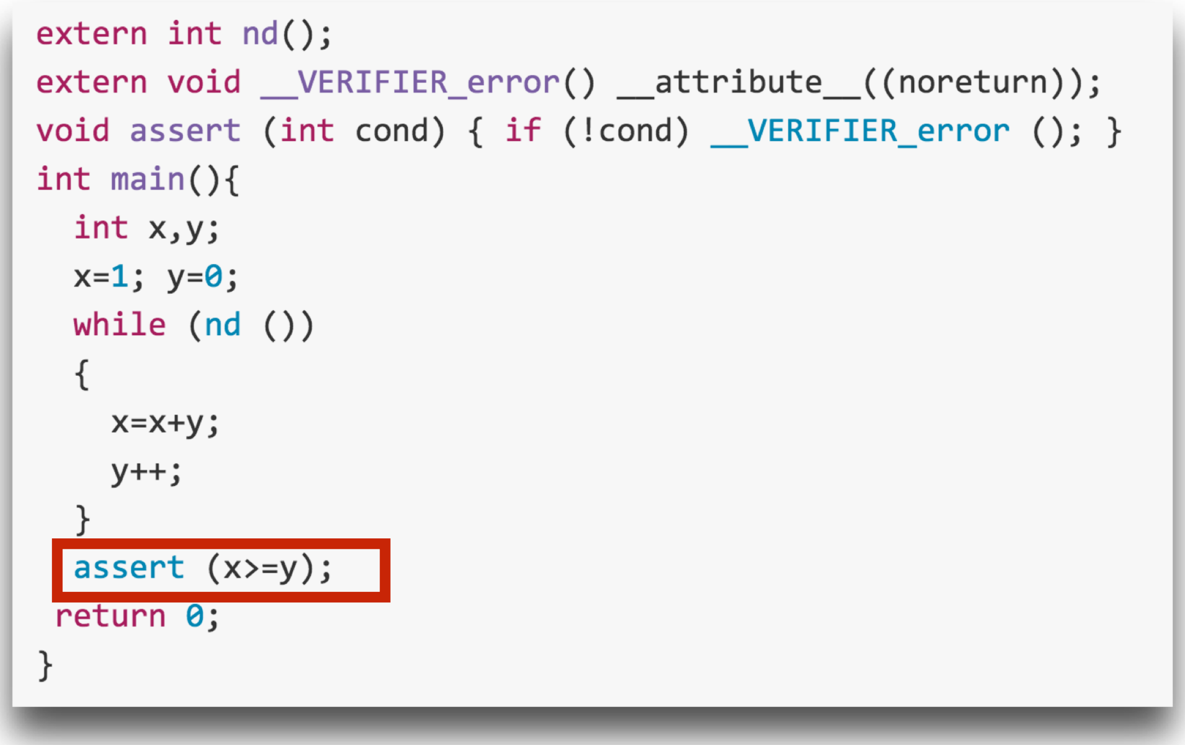
A fully automated verification framework for LLVM-based languages.

http://seahorn.github.io

SeaHorn Usage

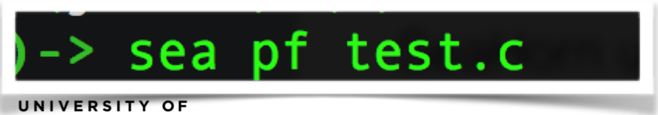
Example: in test.c, check that x is always greater than or equal to y

test.c



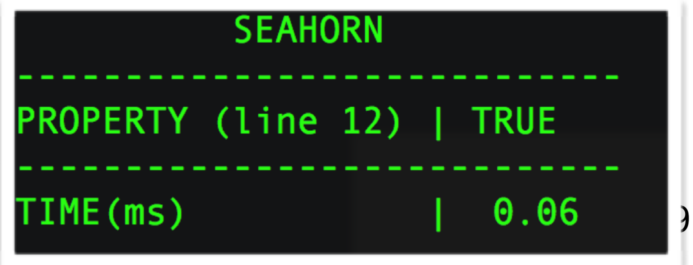
```
extern int nd();
extern void __VERIFIER_error() __attribute__((noreturn));
void assert (int cond) { if (!cond) __VERIFIER_error (); }
int main(){
  int x,y;
  x=1; y=0;
  while (nd ())
  {
    x=x+y;
    y++;
  }
  assert (x>=y);
 return 0;
}
```

SeaHorn command:

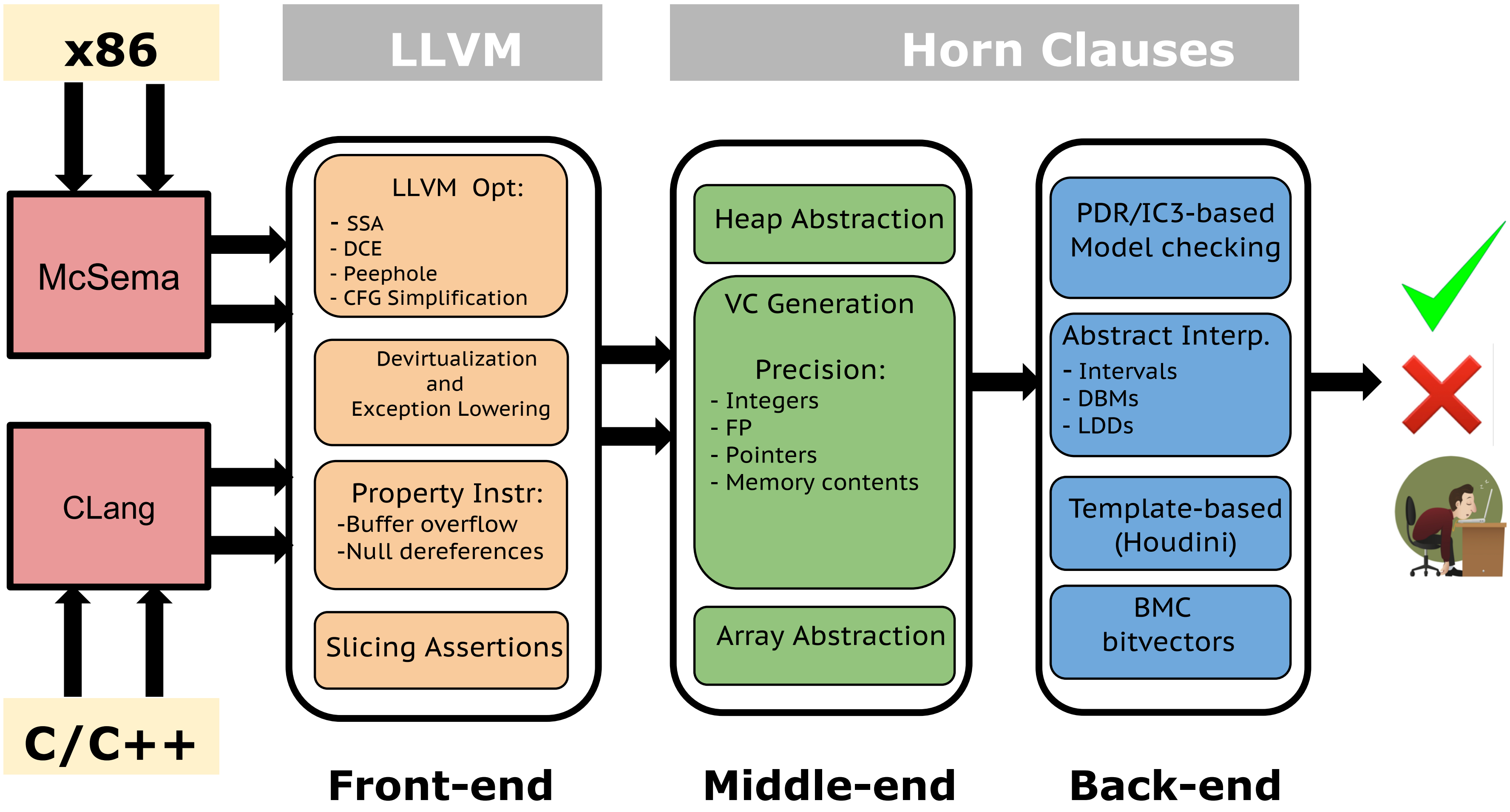


```
)-> sea pf test.c
```

SeaHorn result:



```
            SEAHORN
----------------------------
PROPERTY (line 12) | TRUE
----------------------------
TIME(ms)           |  0.06
```



x86

C/C++

McSema

CLang

LLVM

Horn Clauses

LLVM Opt:
- SSA
- DCE
- Peephole
- CFG Simplification

Devirtualization and Exception Lowering

Property Instr:
-Buffer overflow
-Null dereferences

Slicing Assertions

Heap Abstraction

VC Generation

Precision:
- Integers
- FP
- Pointers
- Memory contents

Array Abstraction

PDR/IC3-based Model checking

Abstract Interp.
- Intervals
- DBMs
- LDDs

Template-based (Houdini)

BMC bitvectors

Front-end

Middle-end

Back-end