ECE.650

Methods & Tools for Software Engineering (MTSE)
Fall 2017

Prof. Arie Gurfinkel

UNIVERSITY OF
WATERLOO

Course Time and Location

Date: Friday

Location: RCH 211

Time: 8:30 – 11:20 AM

No Lecture Next Friday

September 15, 2017

- there will be a makeup lecture, to be announced later

Instructor and TA

Instructor

- Prof. Arie Gurfinkel

Teaching Assistant

- Jakub Kurderski

Course Web Page

- https://ece.uwaterloo.ca/~agurfink/ece650
- LEARN: https://learn.uwaterloo.ca

Topics

Software systems (~40%)

- systems programming and operating systems, scripting, system calls, libraries, compilers, ...

Mathematical logic (~15%)

- propositional logic, syntax, semantics, entailment, deduction and the use of logic in software

Algorithms and Data structures (~45%)

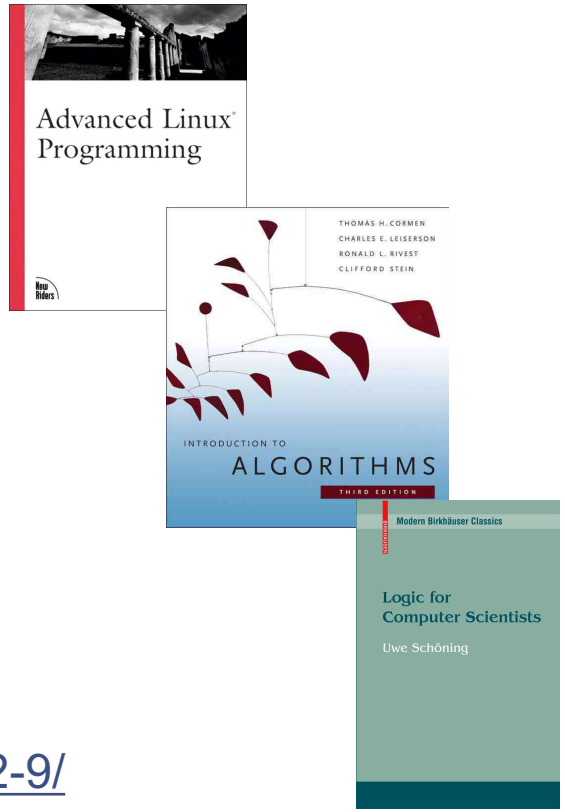
- stacks, heaps, trees, and graphs, … algorithms to manipulate them

Textbooks

No textbooks are required

Material will be based on:

- Advanced Linux Programming
  http://www.advancedlinuxprogramming.com/
- Introduction to Algorithms, Cormen et al., 2nd edition
  http://lib.uwaterloo.ca/
- Logic for Computer Scientists
  http://www.springerlink.com/content/978-0-8176-4762-9/



Course “style”

Hands-on, on-screen, discussions

Reading material

Programming assignments that develop one coherent project

Work outside of the lectures is required!

What is this course “not”

Not a course on programming

- but there is a large, complex, multi-part programming assignment / project

Not a course on operating systems

- but the assignments require and help develop intimate understanding of processes, threads, inter-process communication, and systems’ programming

Not a course on mathematical logic

- but the assignments requires modeling a problem in logic and using a decision procedure (SAT) to solve it

Not a course in data-structures and algorithms

- but the assignment will require using and understanding common algorithms (sorting, searching, parsing) and data structures (graph, list, map)

Assesment

Assignments: 40%

Project: 10%

Final Exam: 50%

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

4 Assignments + Final Project

- Programming assignments leading towards a final project
- Mix of Python and C++ programming
  - one assignment is purely in Python
  - one assignment is purely in C / C++
  - the rest of assignments and a project require using both languages

Course Website & LEARN

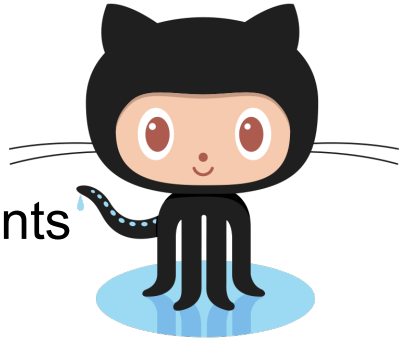
The course website is the definitive source

- When in doubt, consult the web page

YOUR responsibility to check for updates!

- Course website: https://ece.uwaterloo.ca/~agurfink/ece650/
- LEARN (http://learn.uwaterloo.ca)

GitHub and Slack



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

- This requires a free GitHub account
- Follow the link in Assignment 0 to get started
- Let me know if there are any problems!!!

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

- https://uw-ece650.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)
- Create #channels for group discussion and DM for private conversations
- monitor #announcements for course announcements
- ask questions about assignments in #assignments
- invite @arie and @jakub 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/ece650
- Identify yourself
  - Originated from your uwaterloo email address, or
  - Signed with your full name and student ID
- Start **Subject** of email with **[ECE650]**

My Expectations

Attend lectures

- talk to classmates if you are away!

Participate

- during discussions and activities

Be professional

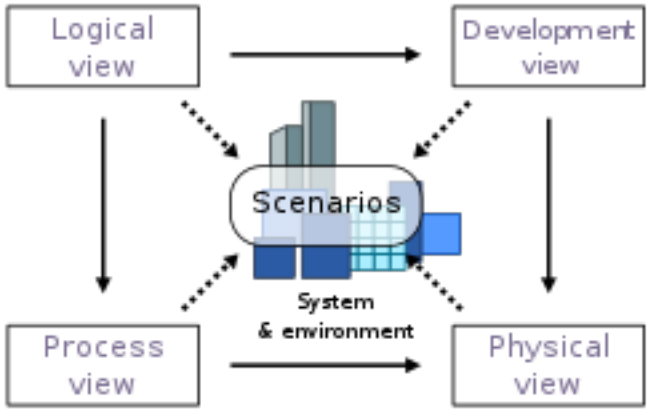
- questions in class, slack, email, discussion on LEARN, interacting with TA, …

ECE 650: Methods and Tools for Software Engineering

Software Systems - Systems programming and operating systems, scripting, system calls, libraries, compilers and interpreters. Mathematical logic - propositional & predicate logic, and some higher-order logics, syntax, semantics, entailment, deduction, use of logic in software. Data structures - lists, stacks, queues, heaps, trees, graphs, and algorithms to manipulate such data structures.

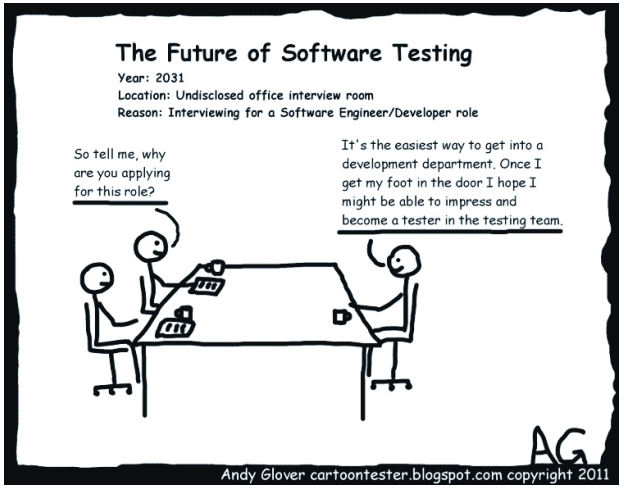
ECE 651: Foundations of Software Engineering

Fundamentals of software requirement analysis, software development as an engineering activity, basic process models, software specifications, modularity, cohesion, coupling, encapsulation, information hiding, principles of object oriented design, software project management, quality assurance and control. Principles of Software Architecture: Fundamental software architecture styles, ... UML ...



ECE 653: Software Testing, Quality Assurance, and Maintenance

Introduces students to systematic testing of software systems. Software verification, reviews, metrics, quality assurance, and prediction of software reliability and availability. Students are expected to have programming experience with reading and writing code for large projects.



ECE 654: Software Reliability Engineering

The course consists of two related parts.

The first part deals with the engineering of reliable software. It introduces basic software reliability concepts, describes relevant models and discusses processes for engineering of reliable software, including schemes and patterns for the design of reliable and fault tolerant software.

The second part addresses development of secure software. It presents key software security concepts, techniques and models, overviews major software security vulnerabilities and their exploitation, and considers processes for development of secure software.

A little about me

2007, PhD University of Toronto

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

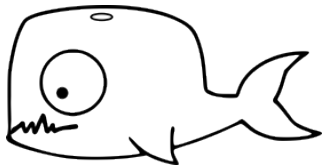
Sep 2016, Associate Professor, University of Waterloo

UFO

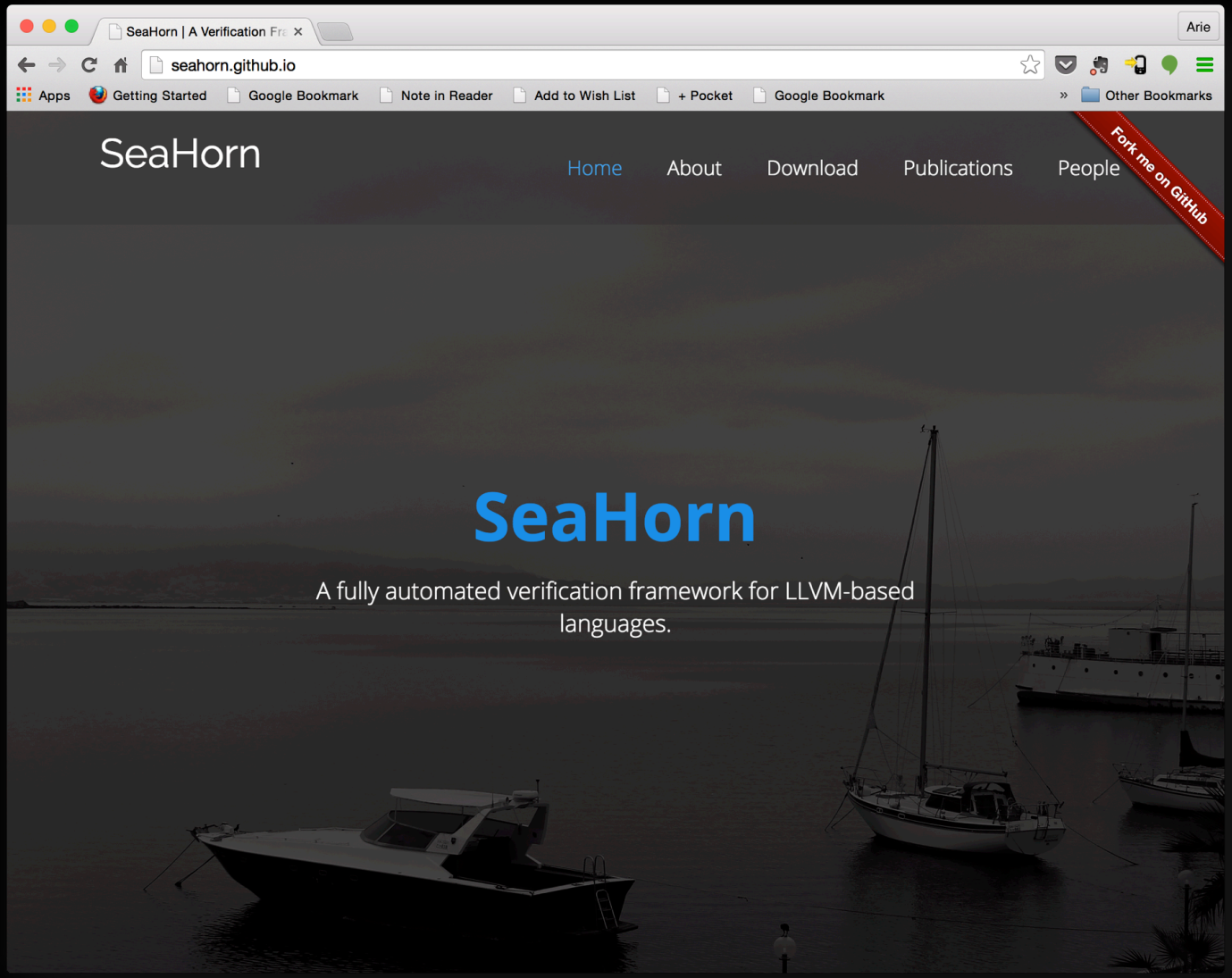
FrankenBit

SPACER

Avy



SeaHorn



http://seahorn.github.io

SeaHorn Usage

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

test.c

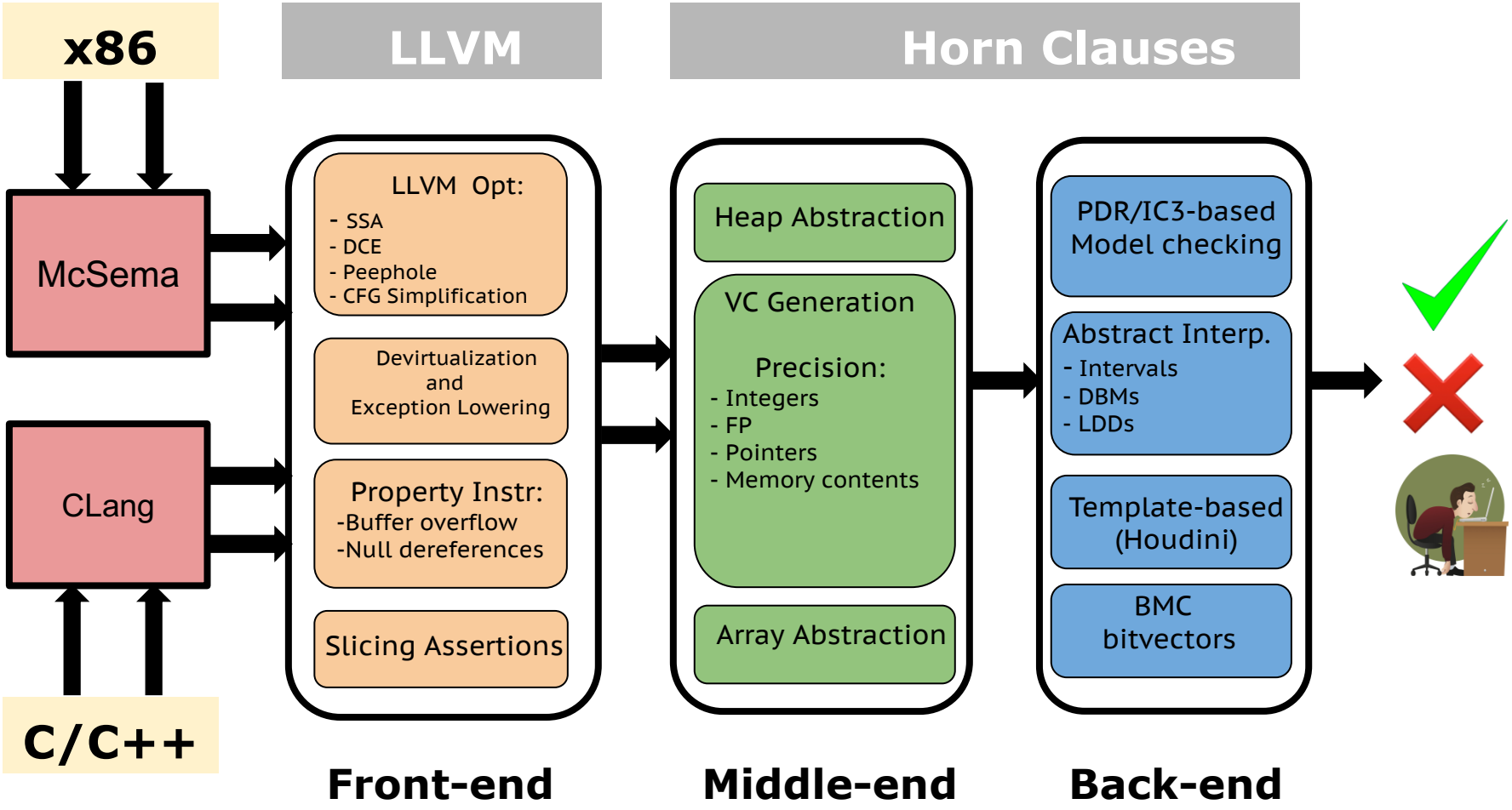
```
extern int nd();
extern void __VERIFIER_error() __attribute__((noreturn));
void assert (int cond) { if (!cond) __VERIFIER_error (); }
int main(){
  int x,y;
  x=1; y=0;
  while (nd ())
  {
    x=x+y;
    y++;
  }
  assert (x>=y);
 return 0;
}
```

SeaHorn command:

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

SeaHorn result:

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


x86
LLVM
Horn Clauses
McSema
CLang
C/C++
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