Course Introduction

17-654/17-765

Analysis of Software Artifacts

Jonathan Aldrich

Introductions

- Instructor
  - Jonathan Aldrich
    aldrich+ at cs.cmu.edu
- TAs
  - Nicholas Sherman
    nds at cs.cmu.edu
  - Dean Sutherland
    dfsuther at cs.cmu.edu
- Students
  - What would you like to learn from this course?

What is Analysis?

My definition:

The systematic examination of an artifact to determine its properties

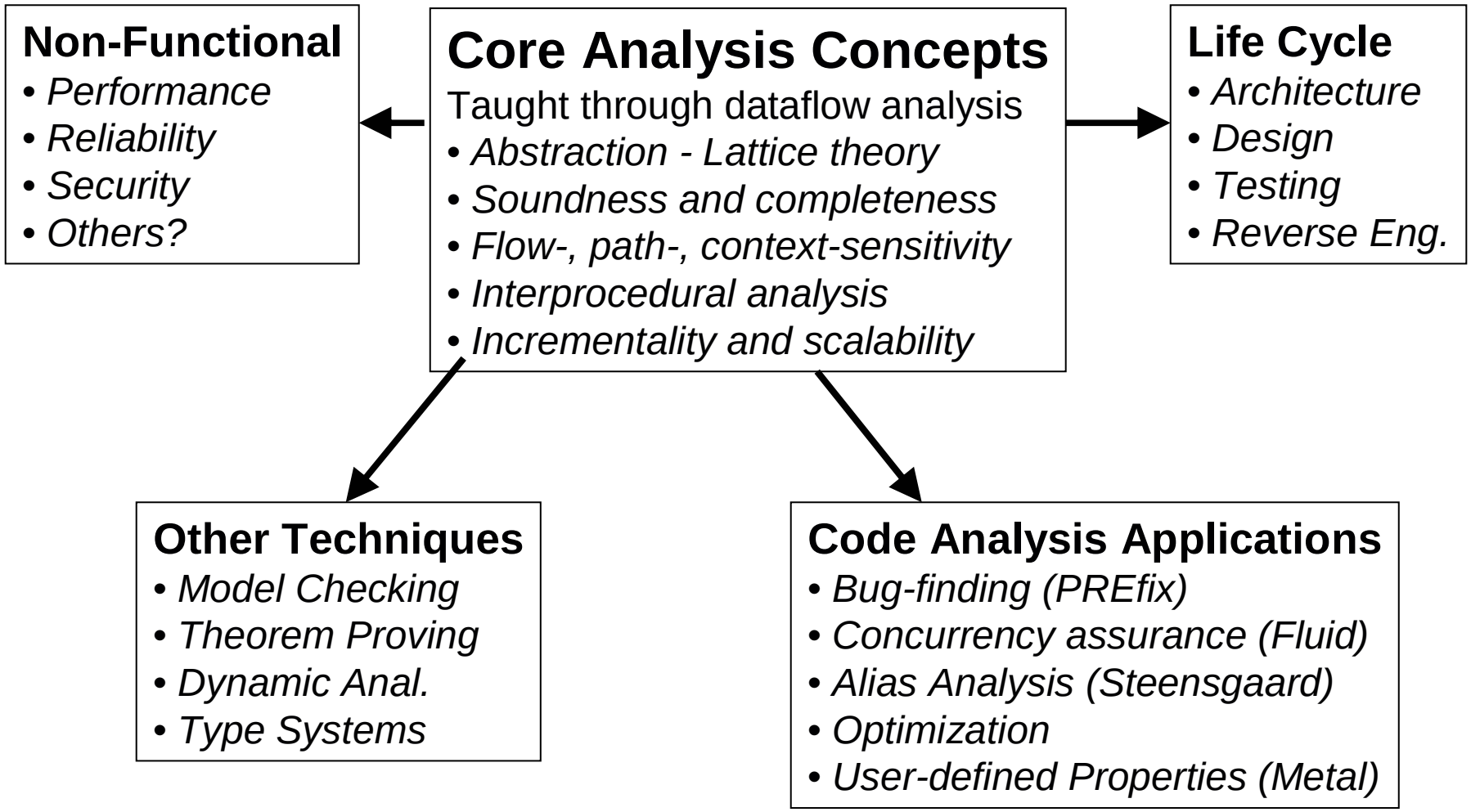
What to Analyze?

- Software engineering degree
  - $\Rightarrow$ Analyze software artifacts
- Product primacy
  - $\Rightarrow$ Focus on analysis of code
  - $\Rightarrow$ Also consider analysis of designs, tests, etc.
- Properties
  - Functional: code correctness
  - Quality attributes: performance, reliability, security

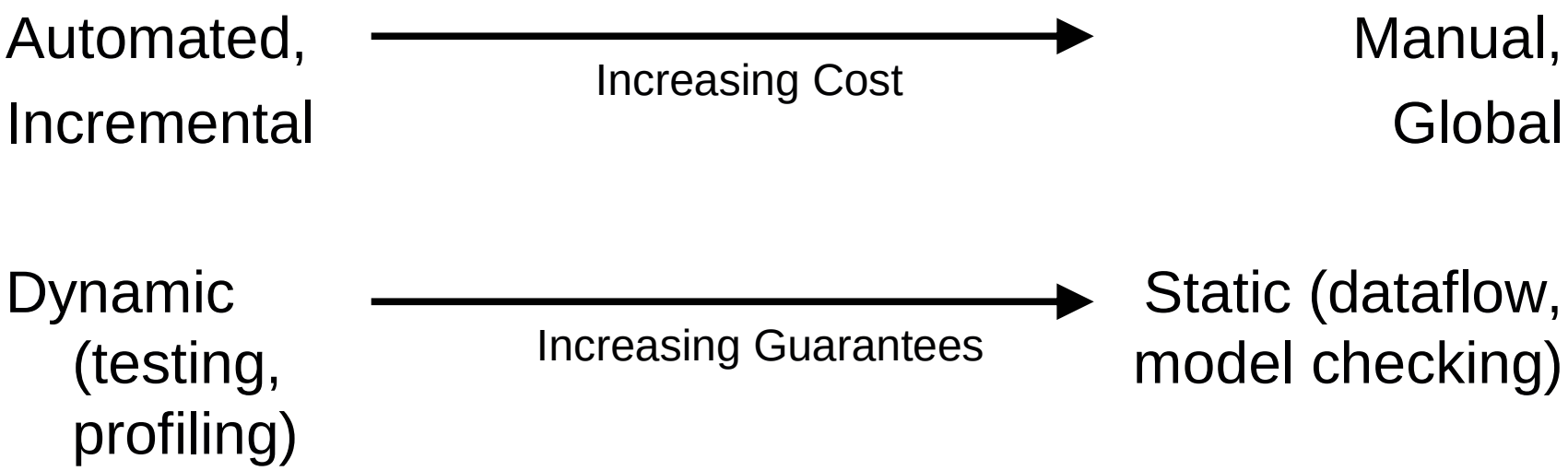
Course Goals

- Understanding
  - Where different analyses are appropriate
  - Tradeoffs between analysis techniques
  - Theory sufficient to evaluate new analyses
- Experience
  - Writing simple analyses
  - Applying analysis to software artifacts

Course Structure



Analysis Tradeoffs



Common Theme: *engineering tradeoffs between different analysis techniques*

Evaluation

- Class participation (~10%)
  - Discussion and presentations
- Homework (30%)
  - Basic understanding of analysis techniques
  - Engineering tradeoffs
- Mini-projects (30%)
  - Evaluate analysis tools on studio or other project
    - Written reports and in-class presentations
  - Write and apply custom analyses
- Midterm and final exams (10% and 20%)
  - Theory and engineering

Ph.D. Projects

- Possible topics
  - Literature survey
    - Study techniques, put into framework, identify open problems
  - Comparative evaluation
    - Your experience with multiple techniques or tools
    - Higher standard than mini tool evals
  - Development of a new analysis technique
  - Application of an analysis technique to a new problem domain
- Requirements
  - Written report
    - Length depends on nature of project
  - Class presentation
- Details to be arranged with instructor

Readings

- Textbook
  - Principles of Program Analysis by Neilson, Neilson, and Hankin
  - Won't be in the bookstore until end of January
    - Badly timed re-printing
    - Will do much of the reading ***before*** then
    - Try to get it online—a link to bookstores is on the course home page
    - Share if you can
    - I'll do my best to make the lectures self-contained
- Papers from the analysis literature
  - Will be provided in class and on the web

Course Emphasis

- Differs slightly from textbook
  - Broader: we will analyze non-code artifacts and consider techniques not in the book
  - Shallower: we will not cover all the theory and techniques in the book
  - Motivation: we focus on engineering rather than optimization
- The text will still be a very useful reference

Free Advice

- Slides will be provided on the web
  - Focus on asking, answering questions
  - Write down what's *NOT* in the slides
- Come to class
  - Participation is required and graded
  - Exercises worked in class will help you
  - The book and papers are terse and incomplete