Learn to Build Automated Software Analysis Tools with Graph Paradigm and Interactive Visual Framework

GIAN, September 12-16, 2016

Suresh C. Kothari
Richardson Professor
Department of Electrical and Computer Engineering

Ben Holland, Iowa State University

Module 0: Introduction

Acknowledgement: **Team members at Iowa State University and EnSoft, DARPA contracts FA8750-12-2-0126 & FA8750-15-2-0080**

Day 1- Overview

- Lecture Topics:
  - Software Engineering – the Big Picture
  - When Programs Fail – a one semester course in two lectures
  - When it took 9 months to debug – what did we do?
  - An introduction to the Atlas graph query language with experiments to analyze Java programs
- Lab:
  - DDMIN Problems with an interesting research direction
  - Analysis experiments using the Atlas graph query language for Java programs

A big inversion

Need: *Automation* – reduce human labor and produce software economically.

Figure 1. IBM System/360 Model 40 Data Processing System

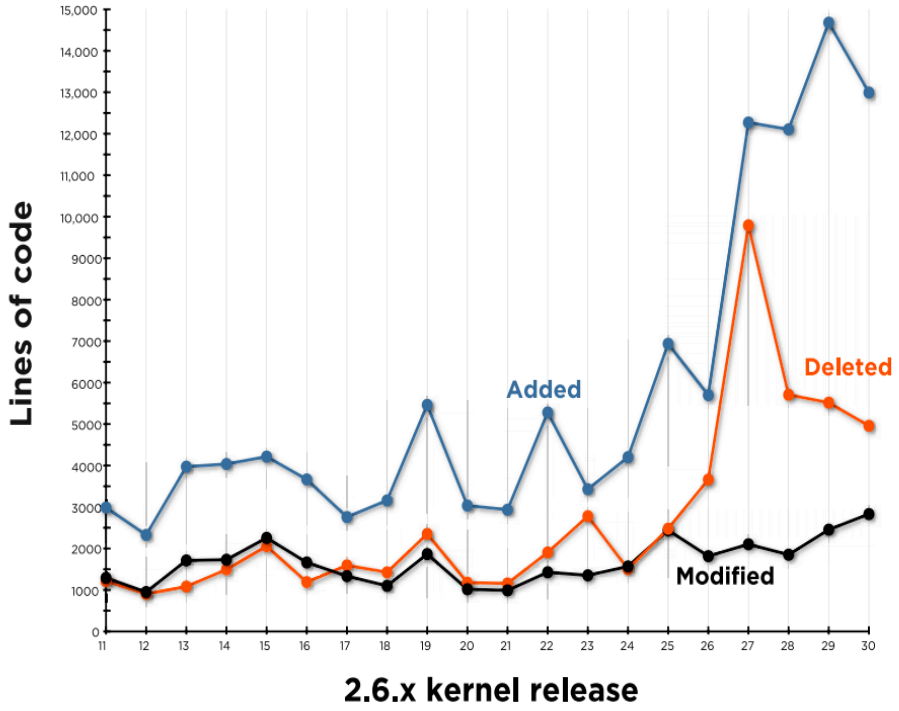
Today

1960: a computer's cost 25 million dollars, and a programmer's annual salary six thousand dollars.

Today: a computer's cost one thousand dollars, and a programmer's annual salary sixty thousand dollars.

Enormous size and continuous change

Need: *Automation* – manage the enormous size and complexity of software.



Linux 1.0.0 – 176,250 LOC
Linux 4.0 – 13 M LOC

A printed version would be 136 feet stack of paper!

Added LOC/day	Deleted LOC/day	Modified LOC/ day
12993	4958	2830

Catastrophic consequences of software failure

Need: *Automation* – affordable analysis and monitoring of software for safety and security.

An *unintentional software error* caused a 500 million dollar rocket to explode.

A *malicious software hack* caused Ukraine power outage

Impact of software automation

- Growing software use: From F4 to F35 fighter jet, more than 80% functionality has moved from hardware to software. F35 software has 24 million lines of code.
- With software everywhere, expect *software automation* to have huge impact.

SCADA Systems

Source: Laing O'Rourke

Source: Dept. of Energy

Medical Devices

Source: www.seekinglpha.com

Source: www.medtechbusiness.com

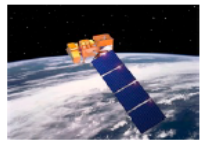
Computer Peripherals

Source: HP

Source: www.buy.com

Source: www.bagitech.com

Communication Devices



Source: NASA

Source: www.engadget.com

Source: GD C4S

Vehicles

Source: www.militaryaerospace.com

Source: www.naval-technology.com

Source: www.motortrend.com

Examples of how the struggle with software continues

- The F-35 fighter jet project overspent by $400 billion dollars and behind schedule – the root cause of the problem, 24 million lines of software in the F35 jet, often called the “flying computer.”
- For the last 15 years, the automotive and and avionics industry continues to struggle to migrate their C software to model-based software development.
- Cybersecurity protection continues to be reactive with software patches after the problem occurs, and no technology defend against the zero-day catastrophic attacks – this has led to cybersecurity DARPA programs with the motto “ *We created the Internet, now we will the create the technology to defend Internet*.”

The critical bottleneck

- *Inefficiency*, *incorrectness*, *ineffective management* and *unpredictability* of many software tasks are rooted in the human difficulty to understand and reason about large software.
- Automated program analysis and software verification continues to suffer from the problems of accuracy, scalability, and usability.

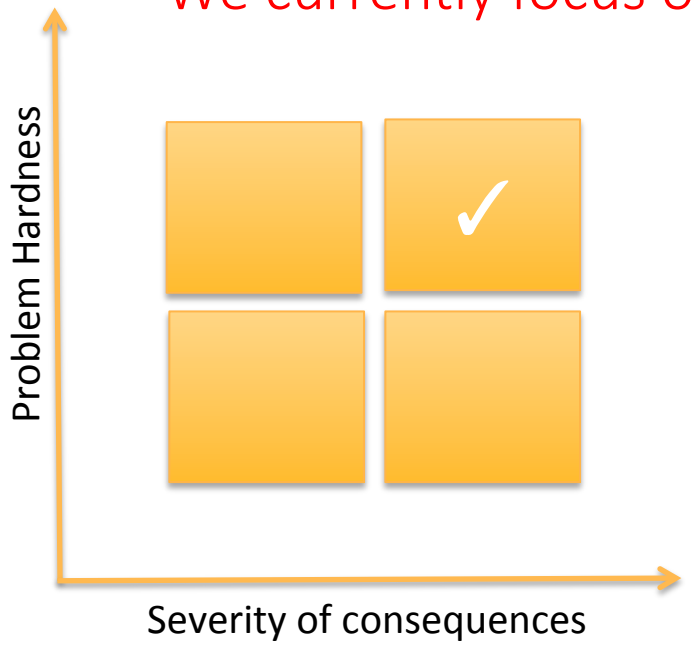
"*If indeed our objective is to build computer systems that solve very challenging problems, my thesis is that IA > AI, that is, that intelligence amplifying systems can, at any given level of available systems technology, beat AI systems. That is, a machine and a mind can beat a mind-imitating machine working by itself.*" – Frederick Brooks, US Nat

Our approach to Intelligence Amplifying (IA) Technology

- Leverages the powerful mathematical abstraction of graphs to manage the complexity of software.
- Our last 20 years of research on solving complex software problems led to the Atlas platform developed over the last ten years with three DARPA grants.
- Atlas provides an unprecedented capability to advance software engineering by *performing experiments* to develop novel solutions to complex software problems.
- Major capabilities of the Atlas platform: a database of program graphs, a powerful query language, and interactive visualization - to build tools to analyze and verify large software.

Current focus of our IA technology

We currently focus on the top-quadrant.



- We are funded by the US Defense Advanced Research Project Agency (DARPA) to develop automated software analysis tools for cybersecurity.
- We have been the *top performing team* on the DARPA Automated Program Analysis for Cybersecurity (APAC) Program – competitors include MIT, Stanford, and some defense companies.
- Commercialization: EnSoft's automation tools for safety-critical control systems software are used worldwide by all major avionics and automobile companies.

Big opportunities

- Careers for personal gain and social impact – from SAP to Genetics.
- Entrepreneurship – from Apps to the graph algorithm patent worth $360 million.
- Unparalleled opportunities for contributions with software engineering research in its infancy - 35 metrics to measure temperature when physics was in its infancy, today we have more than software 1600 metrics.

Experiments lead to innovative concepts and techniques

- We believe the research on automated software engineering has had limited impact because it is not empowered by innovative concepts and techniques enabled by mathematical abstractions.
- Software engineering needs the experimental apparatus and mindset to evolve new concepts and techniques to innovate software engineering

Importance of experimentation: Gauss declared that his way of arriving at mathematical truths was through systematic experimentation — his notebooks attest to it. As mathematicians do, we need to perform experiments; experiments that motivate future development of theory and automation technology.