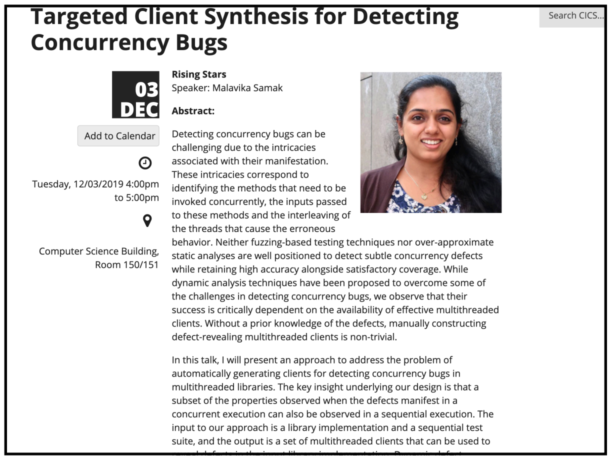
CS 520

Theory and Practice of Software Engineering
Fall 2019

Collaboration and pair programming

December 3, 2019

1



Targeted Client Synthesis for Detecting Concurrency Bugs

Rising Stars
Speaker: Malavika Samak

Tuesday, 12/03/2019 4:00pm to 5:00pm

Computer Science Building, Room 150/151

Abstract:

Detecting concurrency bugs can be challenging due to the intricacies associated with their manifestation. These intricacies correspond to identifying the methods that need to be invoked concurrently, the inputs passed to these methods and the interleaving of the threads that cause the erroneous behavior. Neither fuzzing-based testing techniques nor over-approximate static analyses are well positioned to detect subtle concurrency defects while retaining high accuracy alongside satisfactory coverage. While dynamic analysis techniques have been proposed to overcome some of the challenges in detecting concurrency bugs, we observe that their success is critically dependent on the availability of effective multithreaded clients. Without a prior knowledge of the defects, manually constructing defect-revealing multithreaded clients is non-trivial.

In this talk, I will present an approach to address the problem of automatically generating clients for detecting concurrency bugs in multithreaded libraries. The key insight underlying our design is that a subset of the properties observed when the defects manifest in a concurrent execution can also be observed in a sequential execution. The input to our approach is a library implementation and a sequential test suite, and the output is a set of multithreaded clients that can be used to

2

Final projects

Presentation

In class, next Tuesday, Dec 10

You will have 4 minutes with the professor.
Use them how you want. Demo. Poster. Chat. Presentation on laptop.

Everyone will set up their presentations around the room. I and the TA will go around. Do your thing to us! Also, walk around the room and see what others did.

Submission

Depends on your project.
For research projects, submit a paper describing what you did. For replication studies too.

For implementation projects, submit shorter written-up report that mostly points to code, an automated demo, a video, etc. Make it easy for me to see what you want me to evaluate.

3

Agile development

- Fast paced
- Frequent releases
- Developer centered
  – do we need managers?

4

Scrum

- A very popular flavor of agile
- Three pillars:
  – transparency
  – inspection
  – adaptation

5

Three roles

- Product owner
  – represents the customer
- Development team
  – performs sprints
  – delivers software product
- Scrum master
  – Buffer between team and outside world
  – Prevents distractions, barriers

6

Many aspects of Scrum

- Sprints
- Scrums
- Stand-up meetings
  - what did I do yesterday?
  - what will I do today?
  - do I see any impediment from our goal?
- Reviews

7

Pair programming

- Coding, testing, designing, etc.
- Pair-work facilitates
  - transparency
  - no single point of failure
  - decision making
  - focus

8

Everything's better in pairs

9

Collaboration Exercise

- An exercise game for learning about collaboration
- Developed by
  Laurie Williams and Lucas Layman at NCSU

©Williams and Layman 2007

NC STATE UNIVERSITY

10

http://www.youtube.com/watch?v=rG_U12uqRhE

11

Design a transportation device

- Each person, individually, designs a transportation device that can do all of the following:
  - transport people between 1 and 10 miles per hour
  - stop on demand
  - carry at least one person
  - restrain at least one person (so they don't fall out)
  - look nice
- Draw your transportation device. Work alone and don't look at others' papers. No talking.
  You have 5 minutes.

12

Now, let's integrate

- write down what time you woke up this morning
- together, draw a transportation device that integrates:
  - first riser's braking system
  - second riser's restraint system
  - third riser's propulsion system
  - fourth riser's device appearance

13

What does this exercise teach us?

- Everyone drew a different solution to the same spec ➔ hard to integrate
- Same thing can happen in teams if there is no communication
- Working alone was boring (at least for me)
- Was working together more fun?

14

Movie script

- Break up into pairs (named pair 1 and pair 2)
- Each pair writes a script that must have:
  - a love interest between well-known movie stars
  - attraction for the 18-45 age bracket
  - explosions… lots of explosions
  - a significant plot twist
- Integrate pair 1's stars and explosions with pair 2's romantic storyline and plot twist

15

What does this exercise teach us?

- It's easier to integrate 2 parts than 4 parts
- The result likely fits together better than the 4-part meshed transport
- Were the pair-written scripts or individual-drawn transports more creative?

16

Robotic classroom assistant

- We are going to design a classroom robot responsibilities:
  - person 1: monitor the number of people in the room
  - person 2: mechanism the instructor can use to get students' attention
  - person 3: mechanism for communication between instructor and students
  - person 4: a marketable, interesting name

Spend three minutes 1 working with 2, 3 with 4

Spend three minutes 1 working with 3, 2 with 4

Spend three minutes 1 working with 4, 2 with 3

17

Draw the assistant design

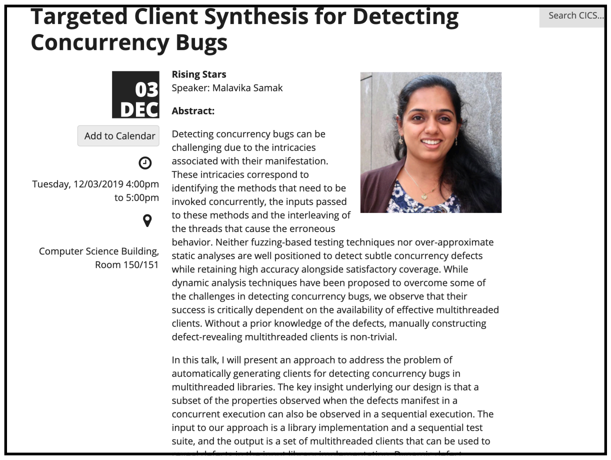
- Each person now draws the design of the robotic assistant

18

What does this exercise teach us?

- Rotation improved everyone's understanding of the product as a whole.
- Risk management: If a person were to drop out, the team could recover more easily (everyone has a partial understanding)
- Each person got more input, leading to more creative, better solutions

19



Targeted Client Synthesis for Detecting Concurrency Bugs

03 DEC

Tuesday, 12/03/2019 4:00pm to 5:00pm

Computer Science Building, Room 150/151

Rising Stars
Speaker: Malavika Samak

Abstract:

Detecting concurrency bugs can be challenging due to the intricacies associated with their manifestation. These intricacies correspond to identifying the methods that need to be invoked concurrently, the inputs passed to these methods and the interleaving of the threads that cause the erroneous behavior. Neither fuzzing-based testing techniques nor over-approximate static analyses are well positioned to detect subtle concurrency defects while retaining high accuracy alongside satisfactory coverage. While dynamic analysis techniques have been proposed to overcome some of the challenges in detecting concurrency bugs, we observe that their success is critically dependent on the availability of effective multithreaded clients. Without a prior knowledge of the defects, manually constructing defect-revealing multithreaded clients is non-trivial.

In this talk, I will present an approach to address the problem of automatically generating clients for detecting concurrency bugs in multithreaded libraries. The key insight underlying our design is that a subset of the properties observed when the defects manifest in a concurrent execution can also be observed in a sequential execution. The input to our approach is a library implementation and a sequential test suite, and the output is a set of multithreaded clients that can be used to

20