Mutation Testing for Physical Computing

Qianqian Zhu
Ph.D. student

Co-author: Andy Zaidman

Software Engineering Research Group,
Delft University of Technology, Netherlands

QRS 2018, July 17, 2018

Physical Computing

Wearable

Smart Home

Robotics

Accidents in Physical Computing

Tesla Says Autopilot Was Engaged in Fatal Crash Under Investigation in California

Vehicle's system shows driver had hands off the wheel for six seconds before striking highway divider

A Tesla equipped with automated-control system crashed last week near Mountain View, Calif. **PHOTO:** PUBLISHED CREDIT: KTVU FOX 2/REUTERS.

Most Popular Videos

1. The Last Cowboy at Pine Creek Ranch
2. Finding the Perfect $1,000 Windows Laptop
3. 'Ready Player One': Can Spielberg's Film Speed Up Adoption of VR?
4. How to Delete Your Facebook Account (or Take Less Drastic Measures)
5. Former Australian Cricket Captain Breaks Down at Press Conference

Most Popular Articles

1. Iowa's Employment Problem: Too Many Jobs, Not Enough People
2. Dominant Box Office

Accidents in Physical Computing

Worker killed by robot in welding accident at car parts factory in India

The man was reportedly stabbed by a metal arm and electrocuted

Lizzie Dearden | @lizziedearden | Thursday 13 August 2015 15:02 | 1 comment

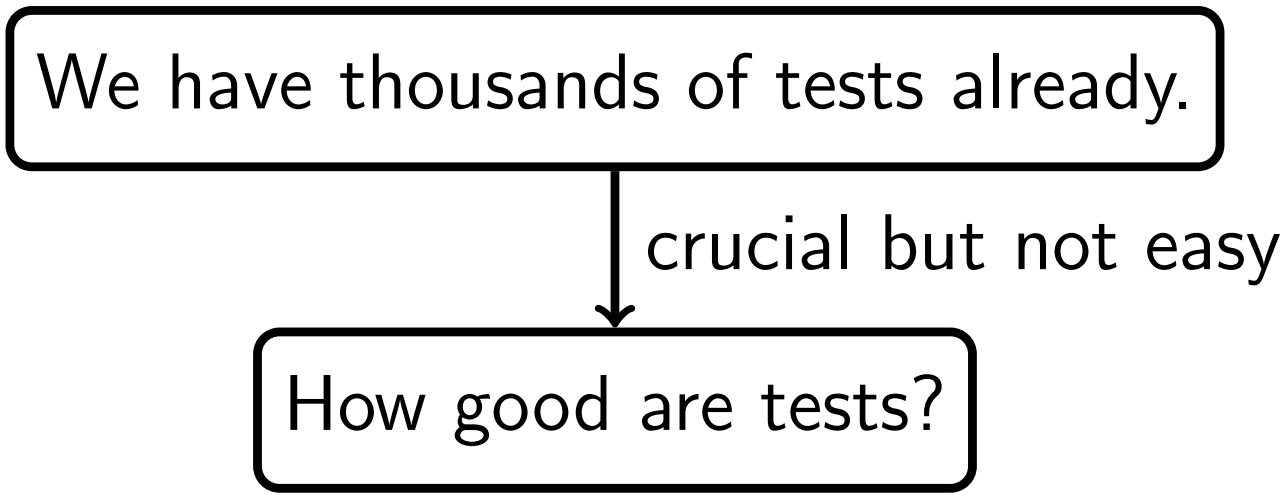
Like Click to follow The Independent Online

Challenges in Testing Cyber-Physical Systems

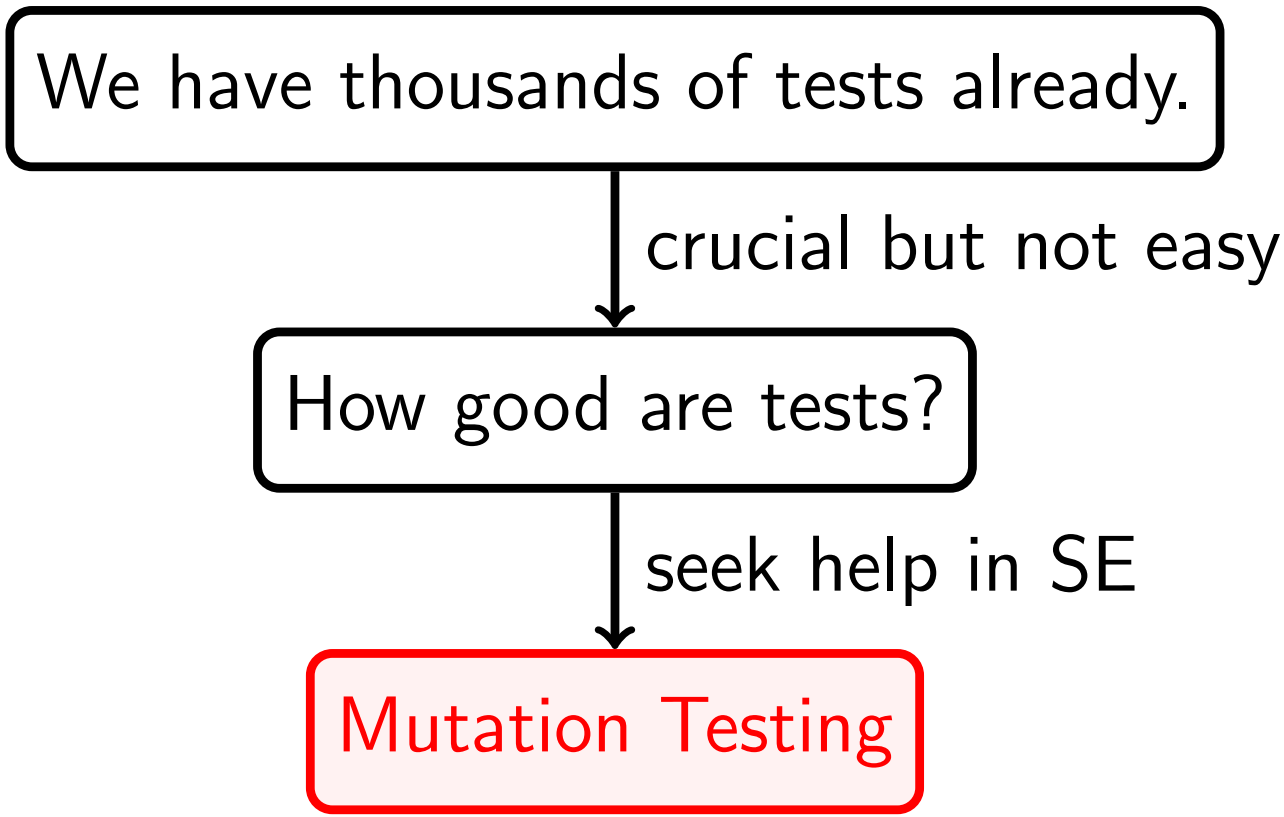
Challenges in Testing Cyber-Physical Systems

We have thousands of tests already.

Challenges in Testing Cyber-Physical Systems



Challenges in Testing Cyber-Physical Systems



Mutation Testing

Mutation Testing

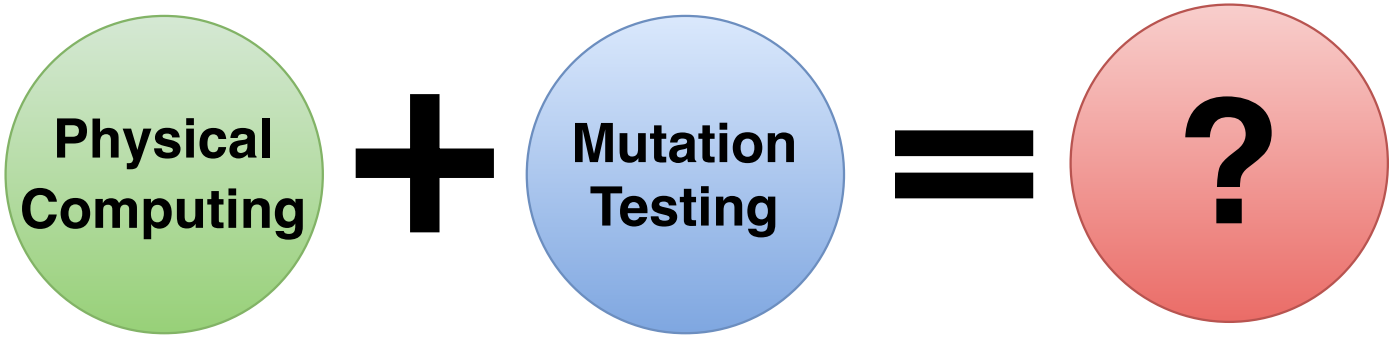
An actively investigated field since 1970s

- **Main idea:** small syntactic changes $\rightarrow$ test suite quality
- **Benefit:**
  - better fault exposing capability
  - a good alternative to real faults

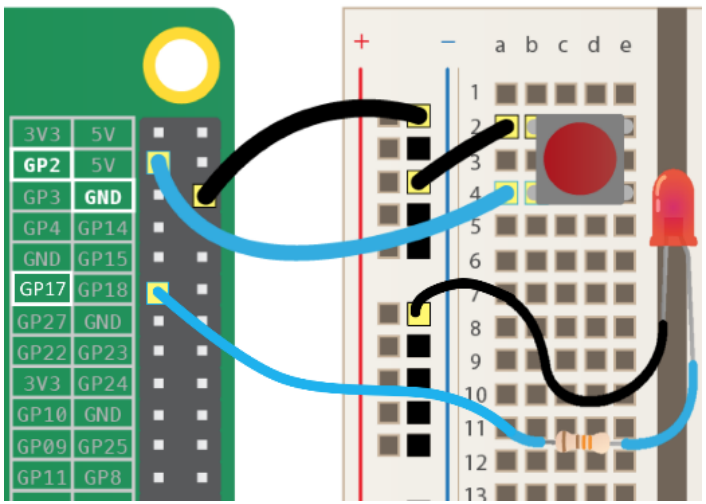
Mutation Testing

An actively investigated field since 1970s

- **Main idea:** small syntactic changes $\rightarrow$ test suite quality
- **Benefit:**
  - better fault exposing capability
  - a good alternative to real faults

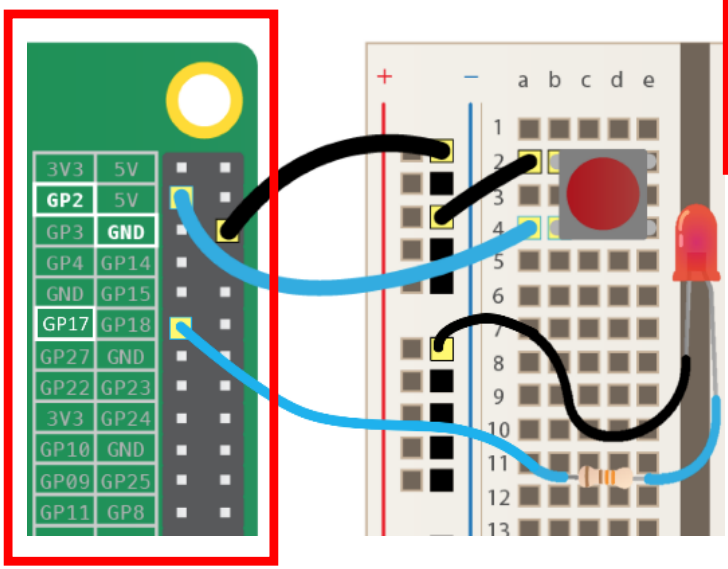


LED example



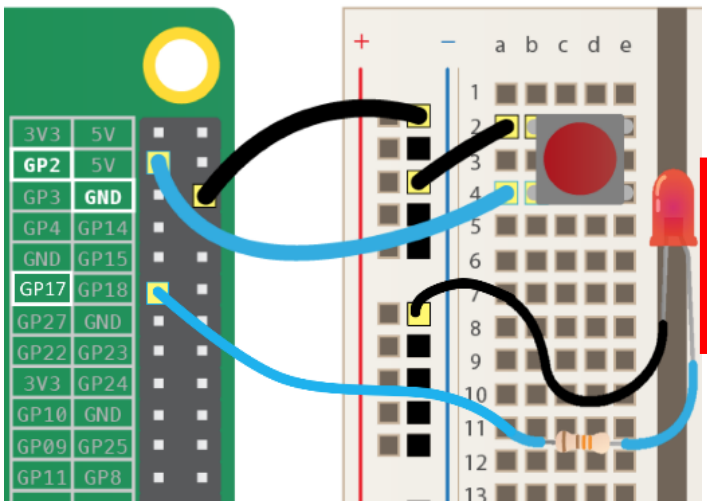

```
from RPi import GPIO
from time import sleep

pbutton = 2
pled = 17

def setup():
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(pled,GPIO.OUT,initial=GPIO.LOW)
    GPIO.setup(pbutton,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
    GPIO.add_event_detect(pbutton, GPIO.RISING, bouncetime=200)

def on():
    GPIO.output(pled,1)

def off():
    GPIO.output(pled,0)

# main
setup()
while (True):
    if GPIO.event_detected(pbutton):  # Check to see if button has been pushed
        activate = True
        while (activate is True):    # Execute this code until the button is pushed again
            on() # Turn LED on
            if GPIO.event_detected(pbutton):   # Check for a 2nd button push
                activate = False
    else:
        off() # Turn LED off
```

LED example



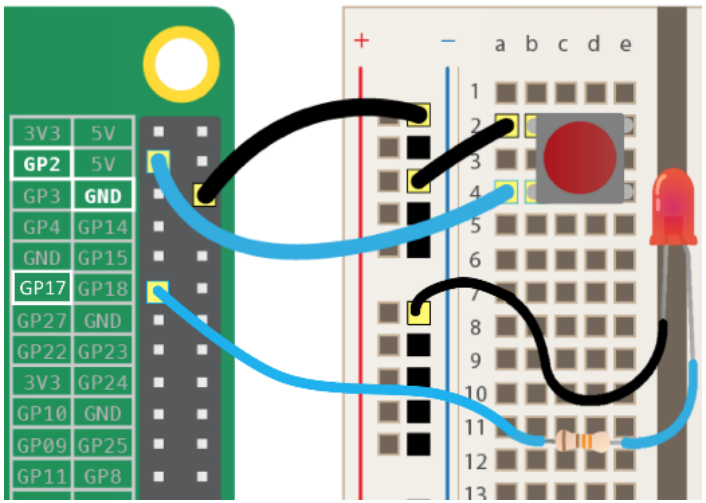

```
from RPi import GPIO
from time import sleep

pbutton = 2
pled = 17

def setup():
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(pled,GPIO.OUT,initial=GPIO.LOW)
    GPIO.setup(pbutton,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
    GPIO.add_event_detect(pbutton, GPIO.RISING, bouncetime=200)

def on():
    GPIO.output(pled,1)

def off():
    GPIO.output(pled,0)

# main
setup()
while (True):
    if GPIO.event_detected(pbutton):  # Check to see if button has been pushed
        activate = True
        while (activate is True):    # Execute this code until the button is pushed again
            on() # Turn LED on
            if GPIO.event_detected(pbutton):   # Check for a 2nd button push
                activate = False
    else:
        off() # Turn LED off
```

LED example




```
from RPi import GPIO
from time import sleep

pbutton = 2
pled = 17

def setup():
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(pled,GPIO.OUT,initial=GPIO.LOW)
    GPIO.setup(pbutton,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
    GPIO.add_event_detect(pbutton, GPIO.RISING, bouncetime=200)

def on():
    GPIO.output(pled,1)

def off():
    GPIO.output(pled,0)

# main
setup()
while (True):
    if GPIO.event_detected(pbutton):  # Check to see if button has been pushed
        activate = True
        while (activate is True):    # Execute this code until the button is pushed again
            on() # Turn LED on
            if GPIO.event_detected(pbutton):   # Check for a 2nd button push
                activate = False
    else:
        off() # Turn LED off
```

LED example with one mistake




```
from RPi import GPIO
from time import sleep

pbutton = 2
pled = 17

def setup():
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(pled,GPIO.OUT,initial=GPIO.LOW)
    GPIO.setup(pbutton,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
    GPIO.add_event_detect(pbutton, GPIO.RISING, bouncetime=200)

def on():
    GPIO.output(pled,0)

def off():
    GPIO.output(pled,0)

# main
setup()
while (True):
    if GPIO.event_detected(pbutton):  # Check to see if button has been pushed
        activate = True
        while (activate is True):    # Execute this code until the button is pushed again
            on() # Turn LED on
            if GPIO.event_detected(pbutton):   # Check for a 2nd button push
                activate = False
    else:
        off() # Turn LED off
```

1 **output value errors:** *OutputValueReplacement (OVR)*

Mutation Operators for Physical Computing

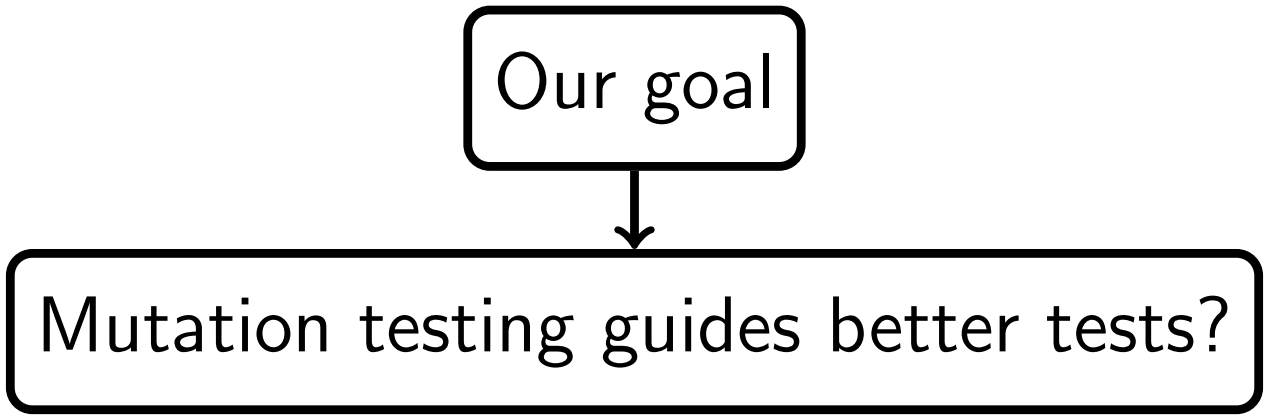
1 output value error	1 OutputValueReplacement (OVR)
2 output setting omissions	2 OutputValueReplacement (OVR)
3 pin number errors	3 PinNumberReplacement (PNR)
4 input value errors	4 InputValueReplacement (IVR)
	5 EdgeDetectionReplacement (EDR)
5 I/O pin mode errors	6 I/OModeReplacement (IOMR)
6 initial setup value errors	7 SetupInputReplacement (SIR)
	8 SetupOutputReplacement (SOR)
	9 SetupValueRemoval (SVR)
6 common mistakes	**9 mutation operators**

Empirical Evaluation

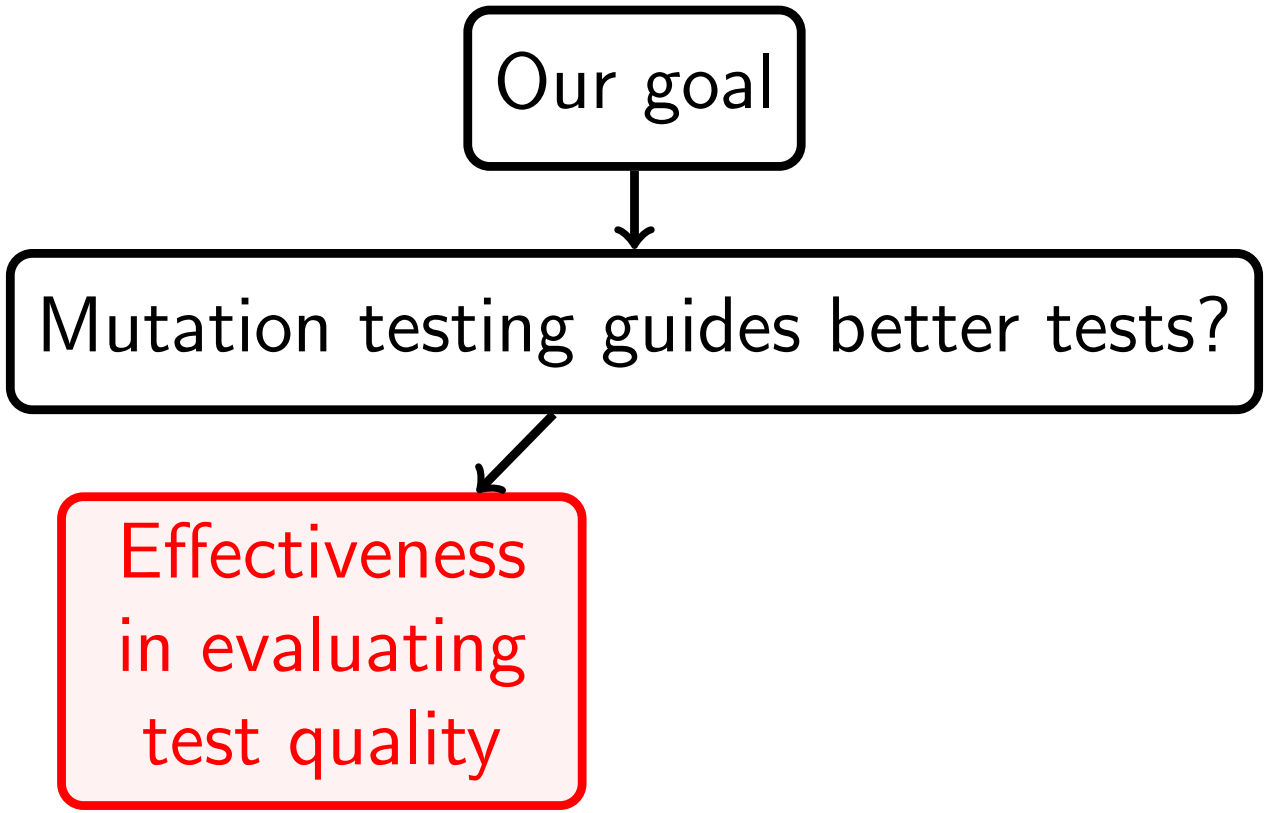
Empirical Evaluation

Our goal

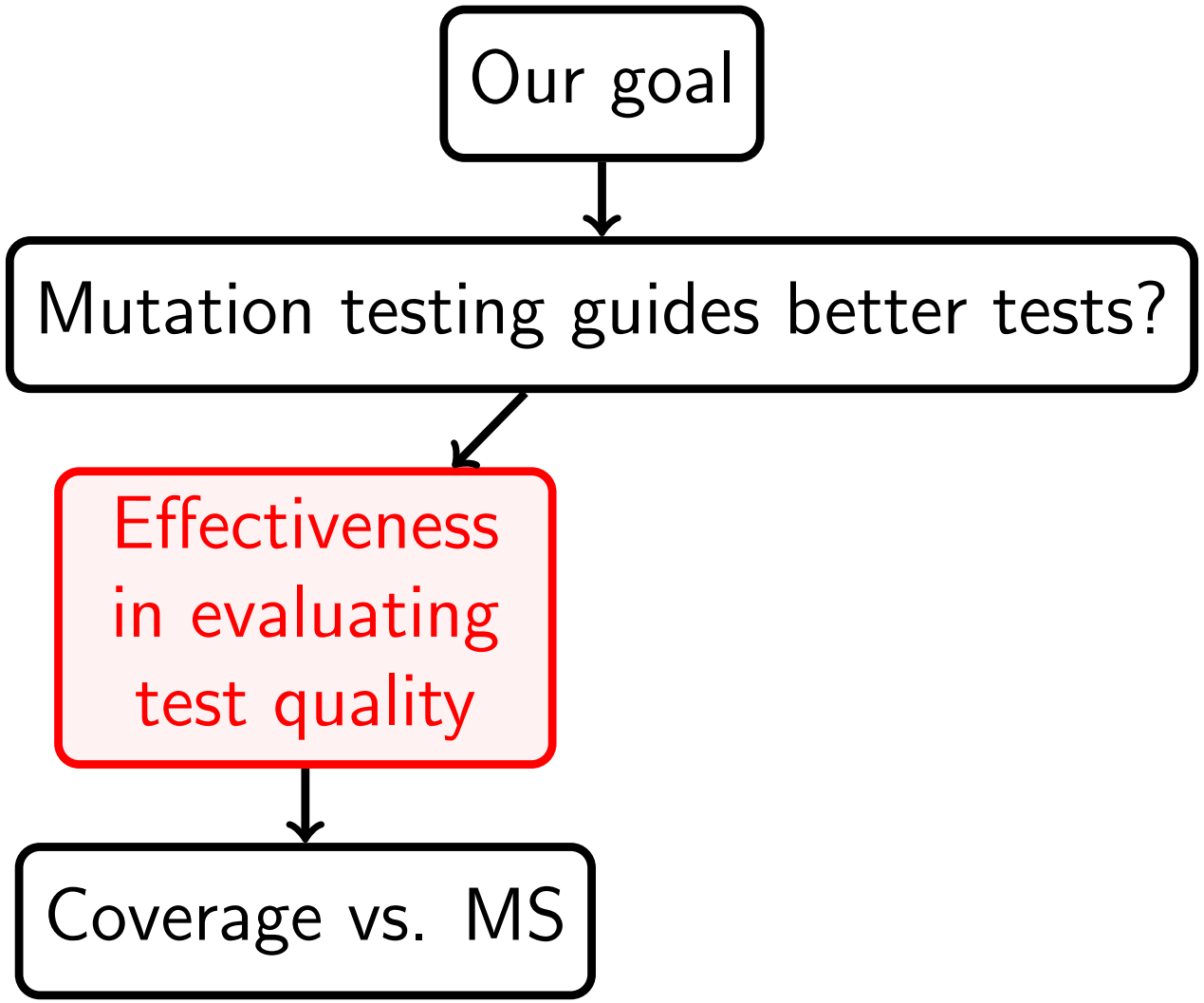
Empirical Evaluation



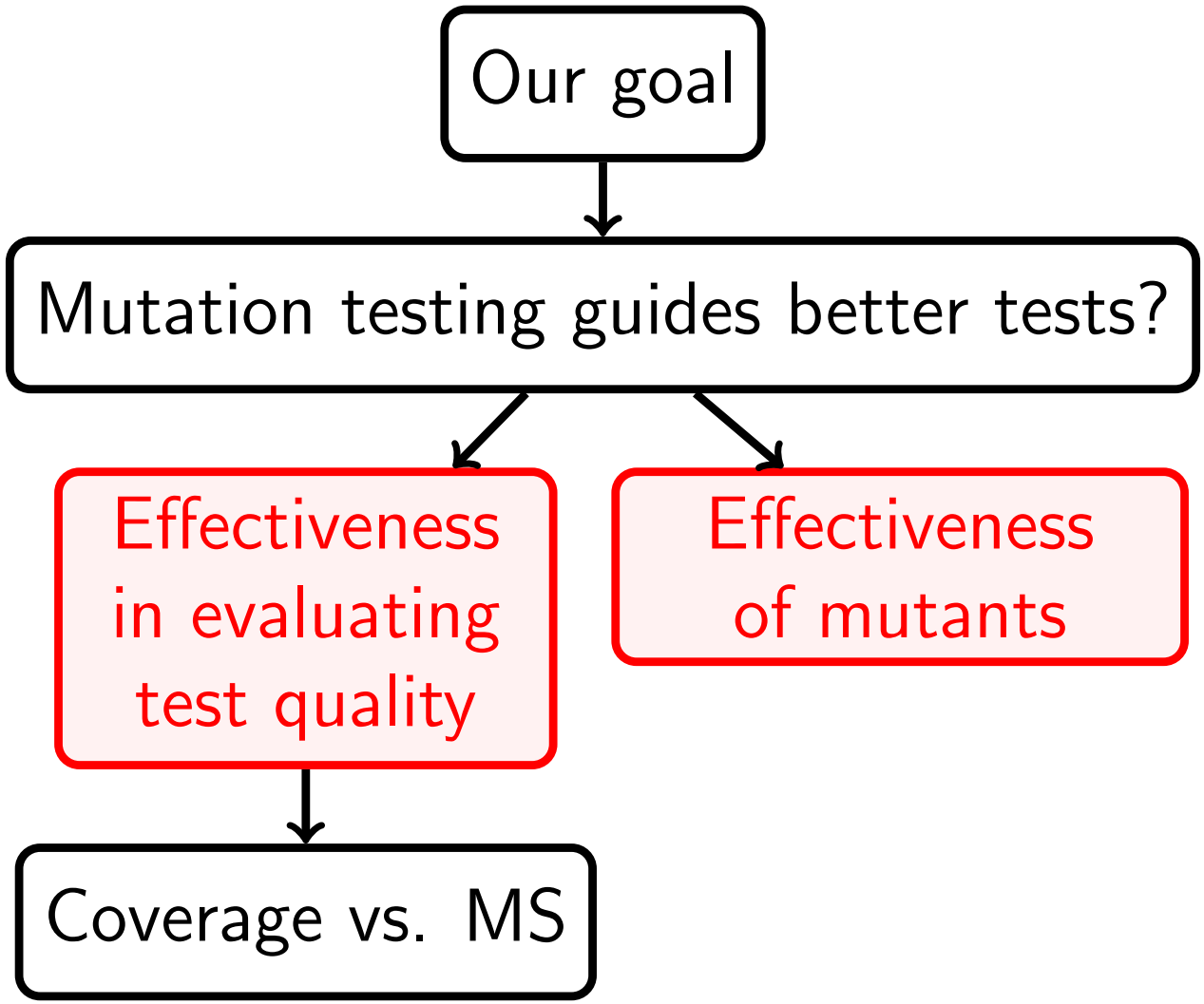
Empirical Evaluation



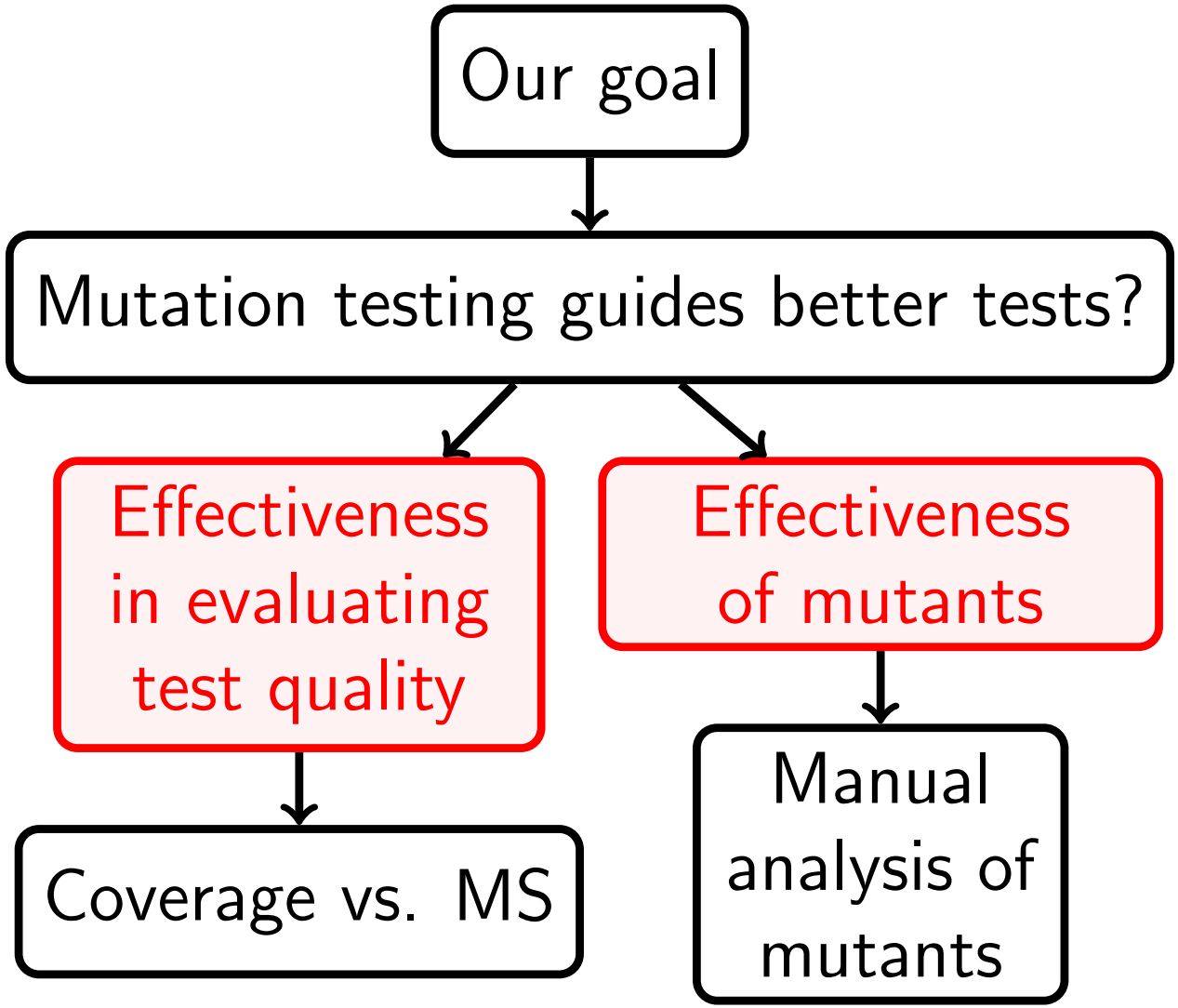
Empirical Evaluation



Empirical Evaluation



Empirical Evaluation

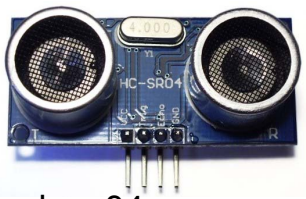


Empirical Evaluation

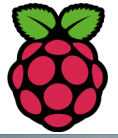
- **Subjects:** 9 cyber-physical systems

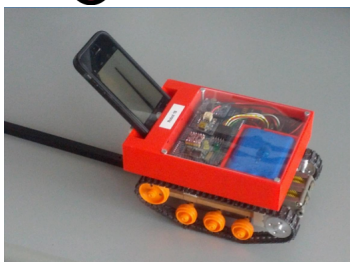

jean-pierre

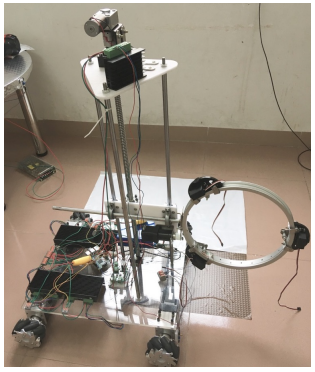
RPLCD



hcsr04sensor


gpiozero


line-follower (4 projects)

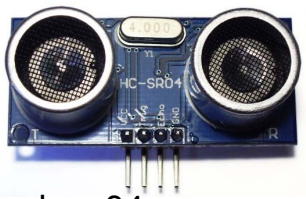

four-wheel robot

Empirical Evaluation

- **Subjects:** 9 cyber-physical systems

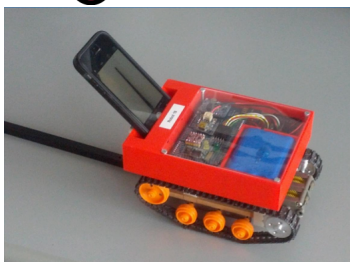

jean-pierre

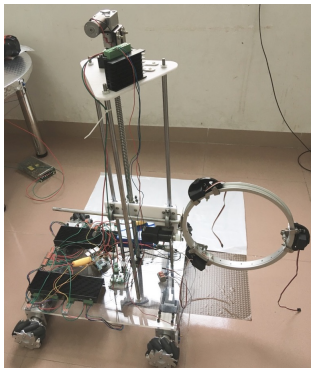
RPLCD



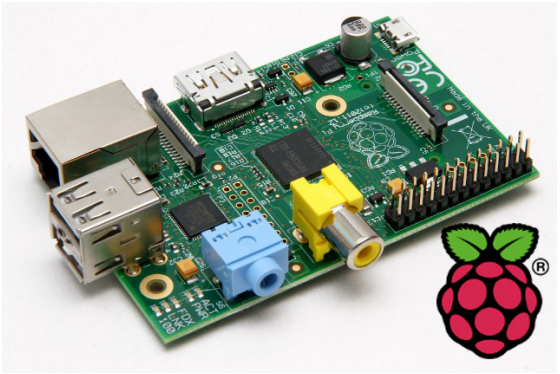
hcsr04sensor

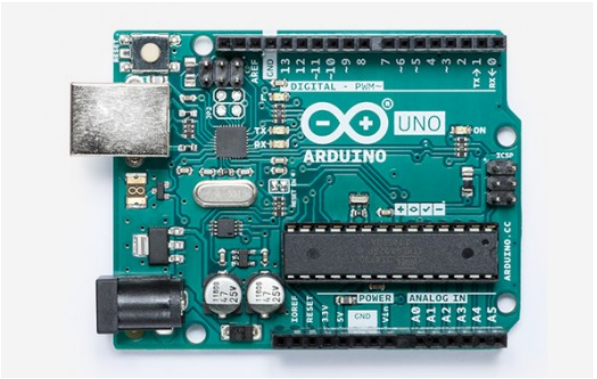
gpiozero


line-follower (4 projects)


four-wheel robot

- **Platforms:** Raspberry Pi & Arduino



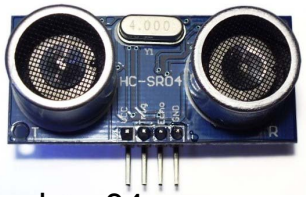


Empirical Evaluation

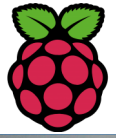
- **Subjects:** 9 cyber-physical systems

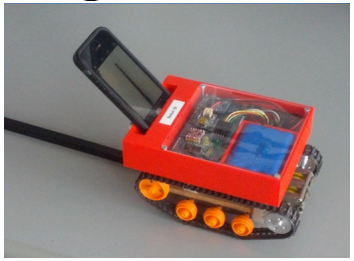

jean-pierre

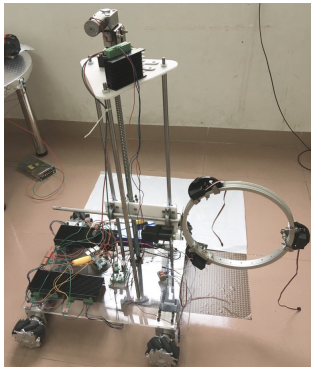
RPLCD



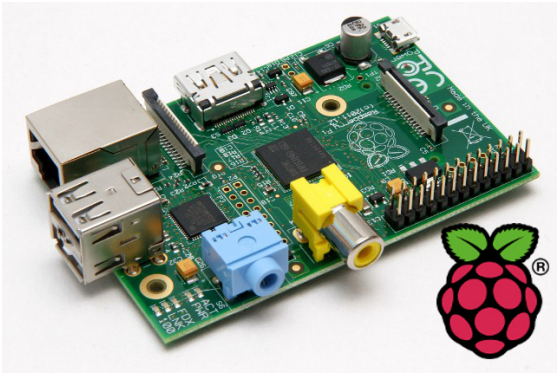
hcsr04sensor

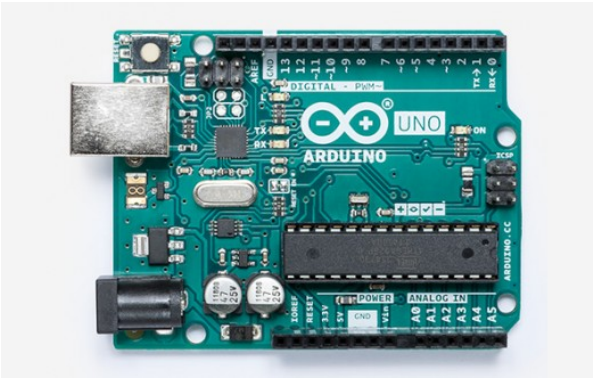

gpiozero


line-follower (4 projects)


four-wheel robot

- **Platforms:** Raspberry Pi & Arduino

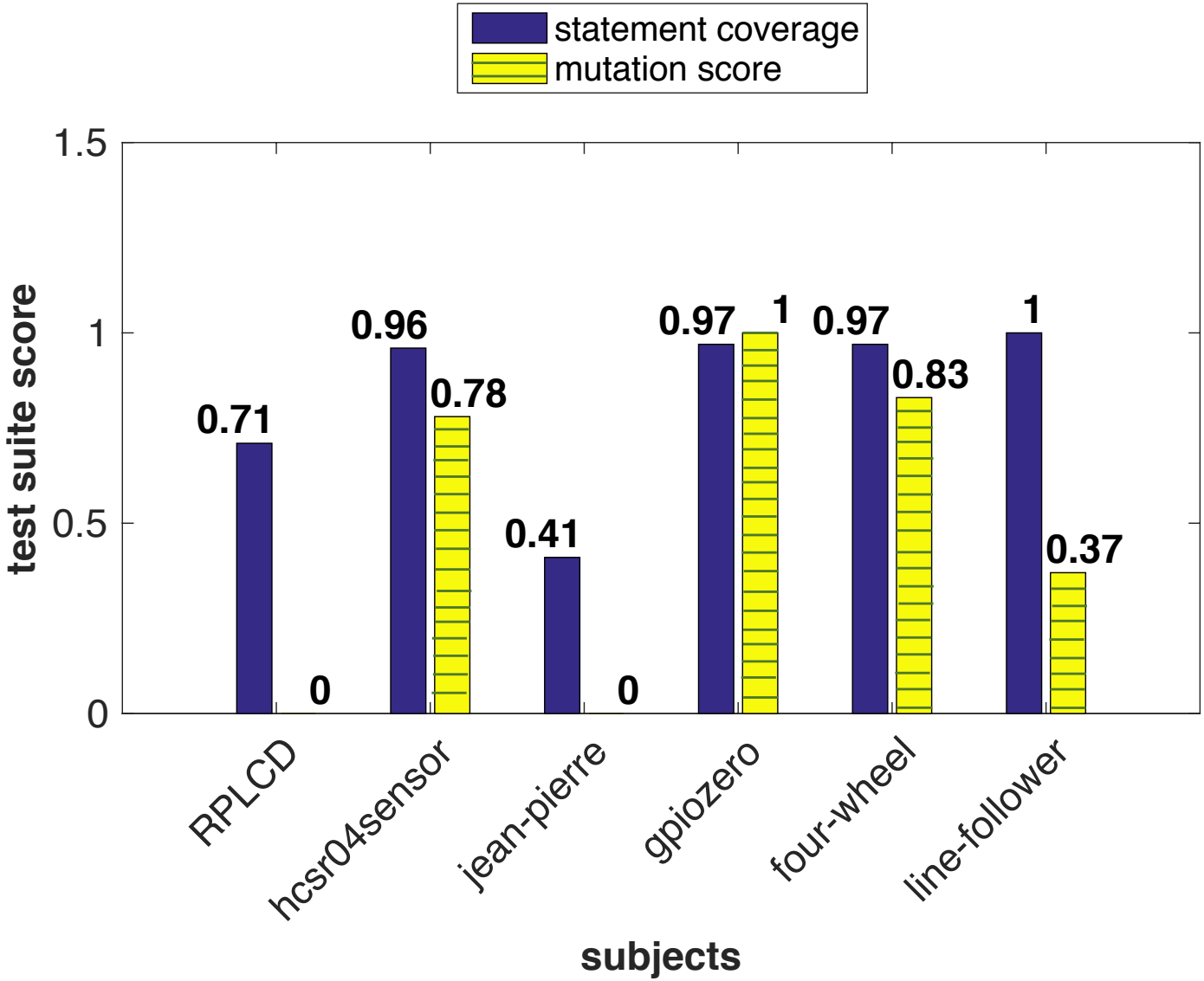




- **Languages:** Python & C/C++

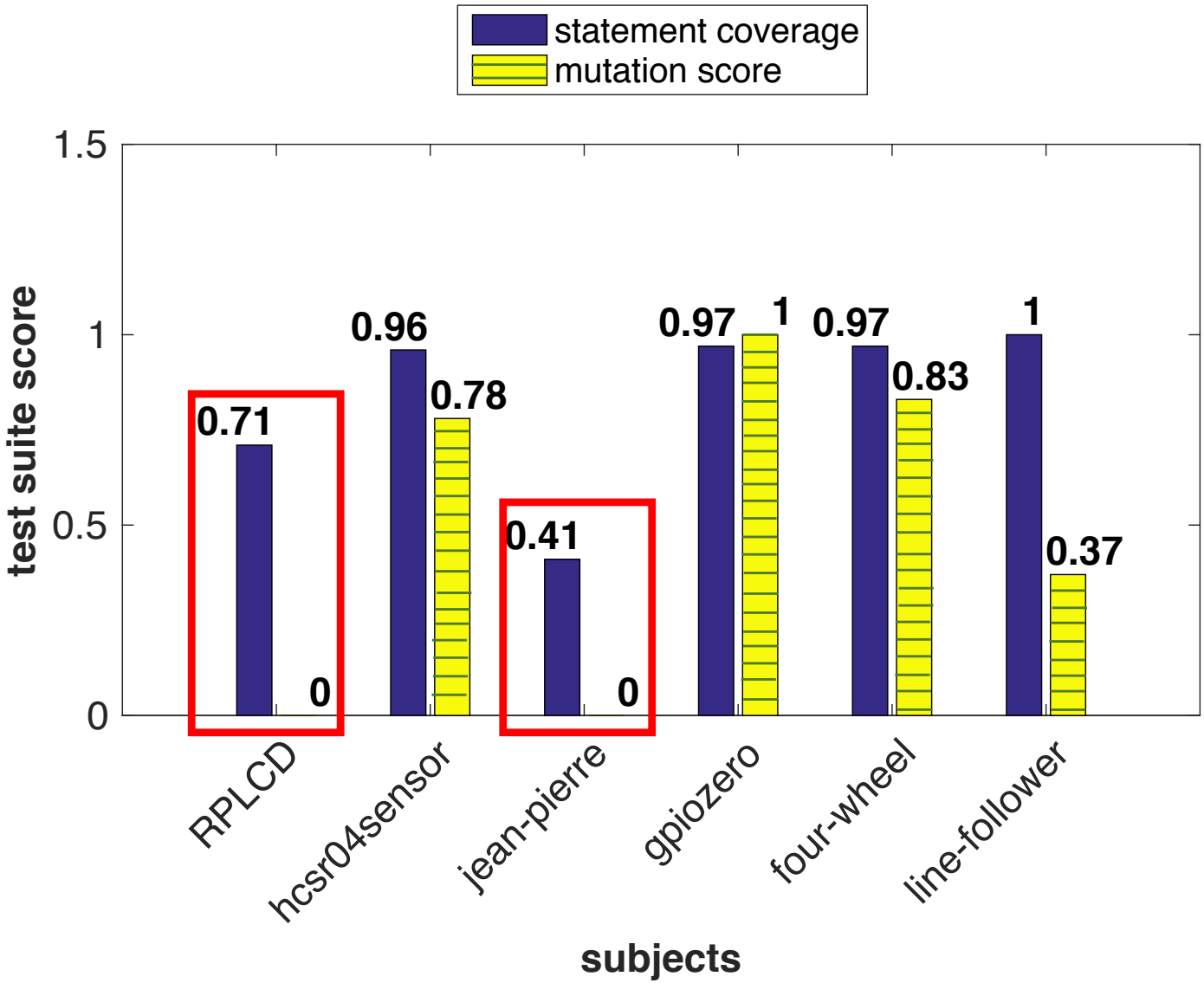
Results

- **Effectiveness in evaluating test suites**



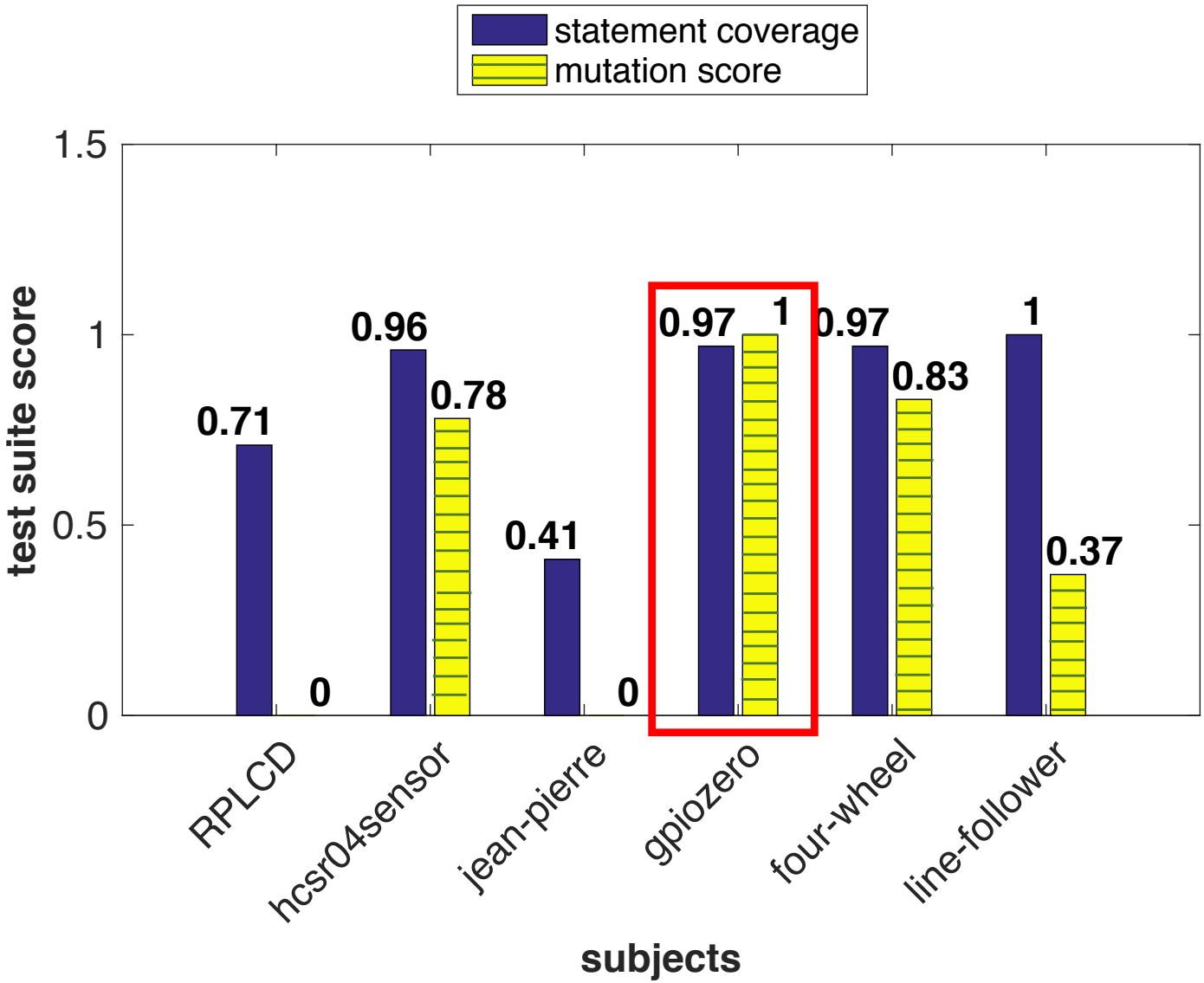
Results

- **Effectiveness in evaluating test suites**



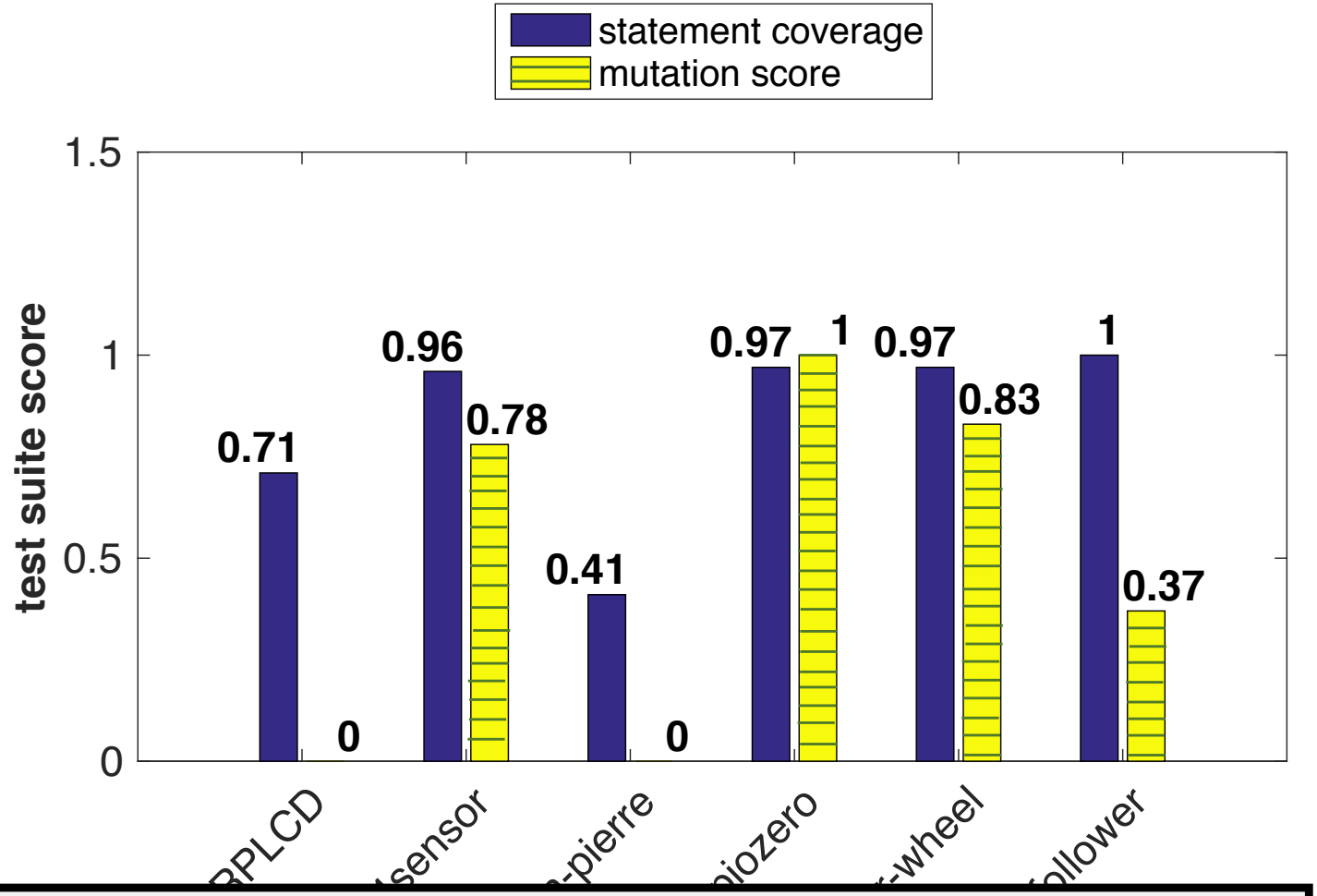
Results

- **Effectiveness in evaluating test suites**



Results

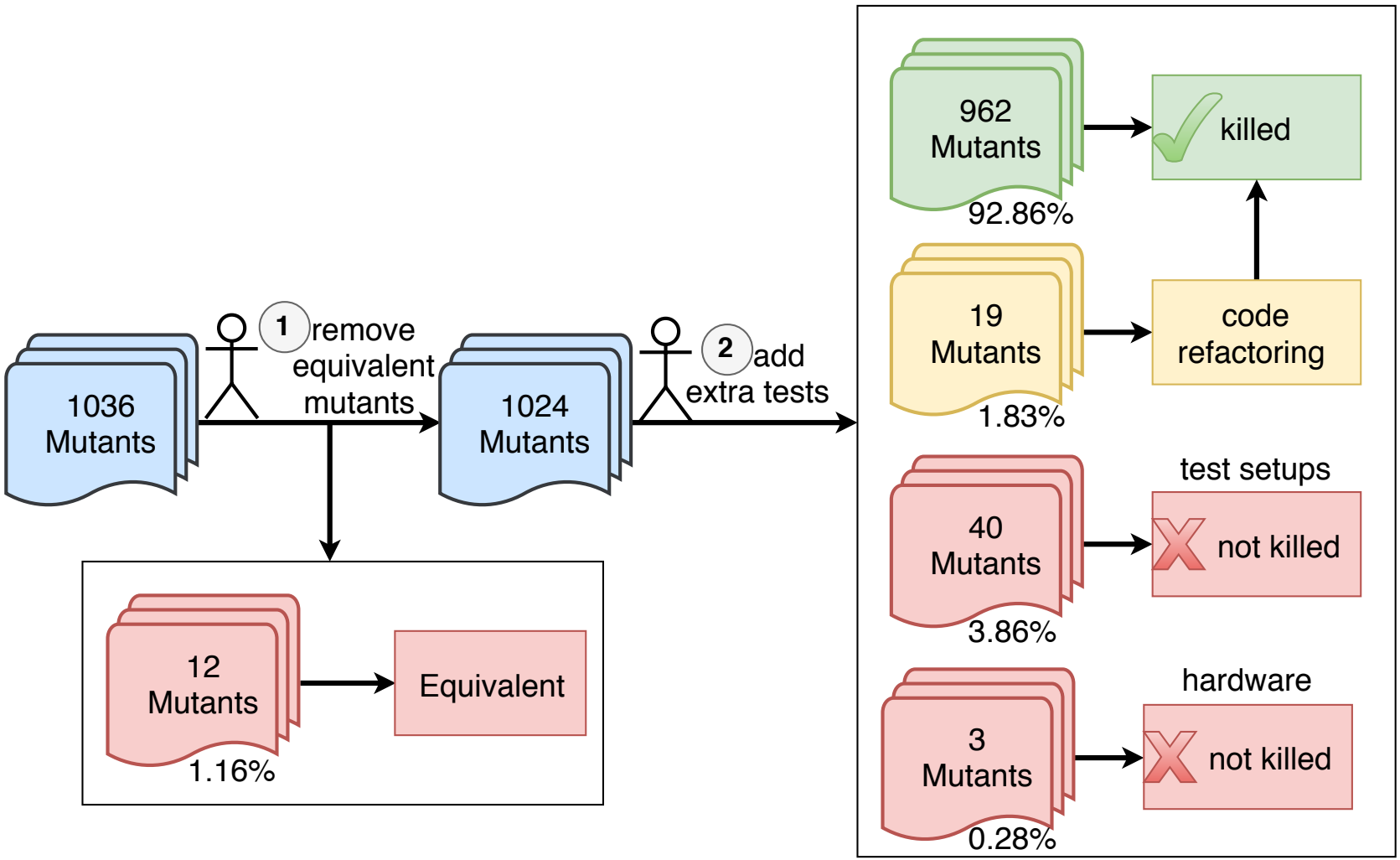
- **Effectiveness in evaluating test suites**



■ integration part: mutation score > statement coverage

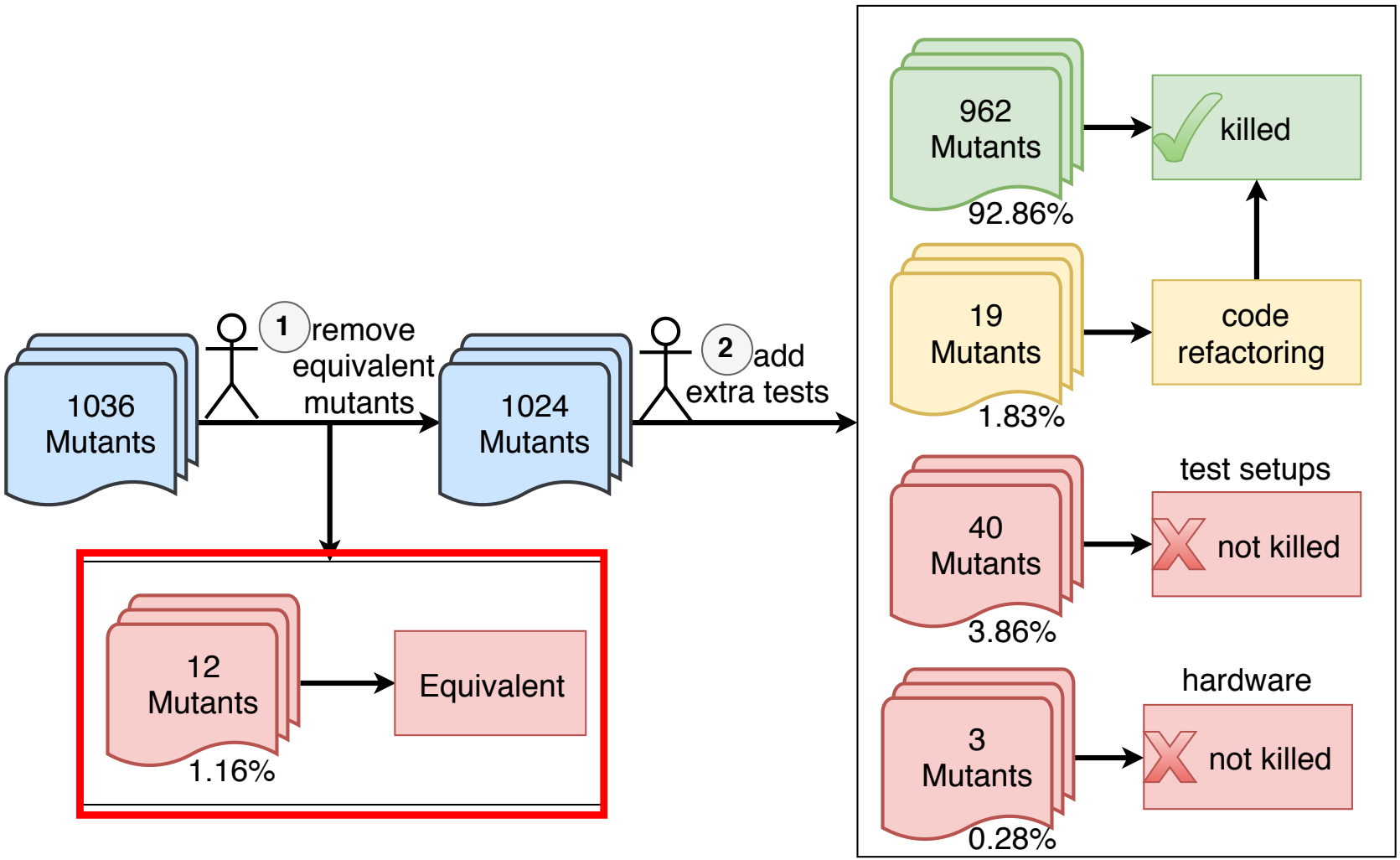
Results

- **Effectiveness of mutants**



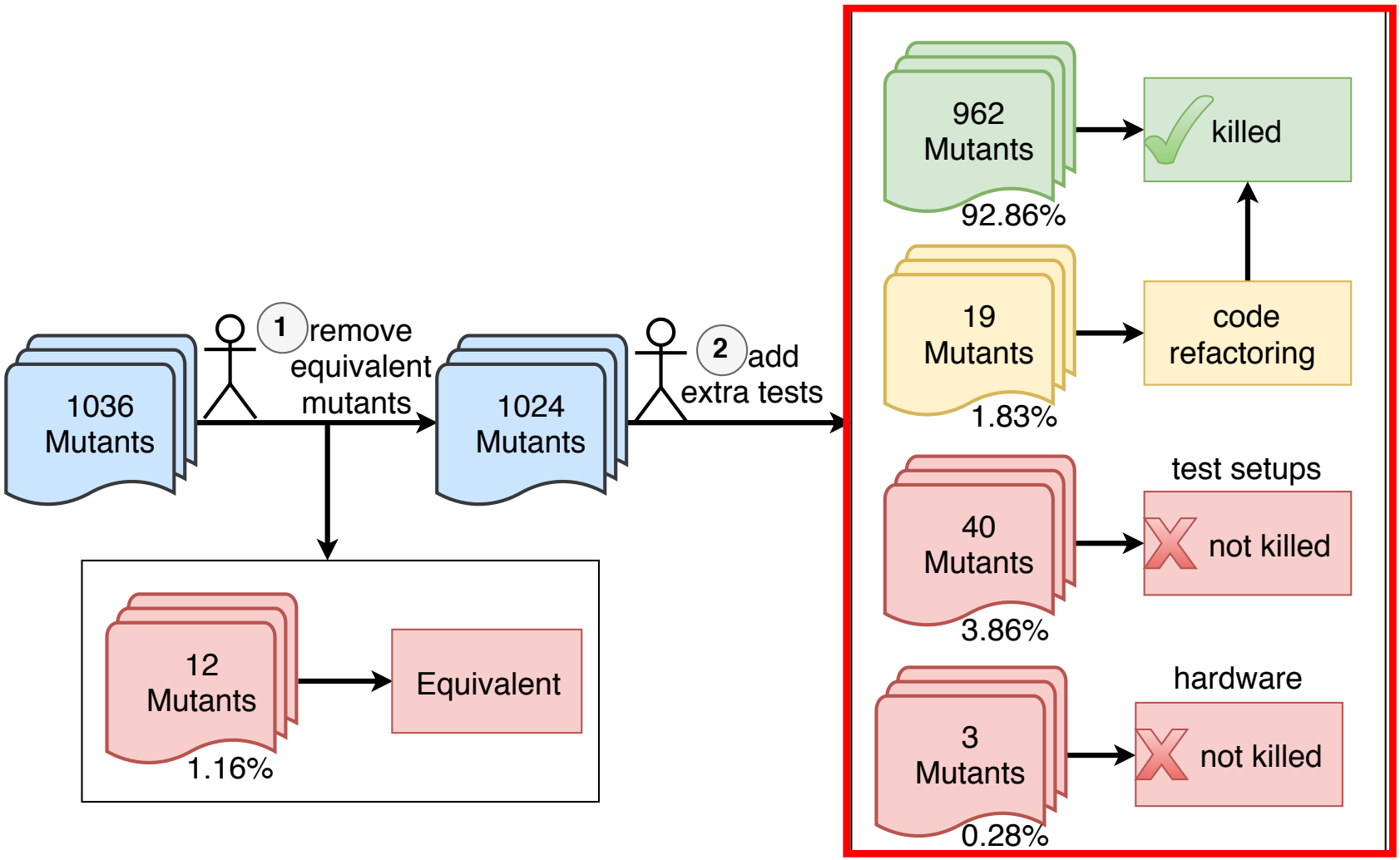
Results

- **Effectiveness of mutants**



Results

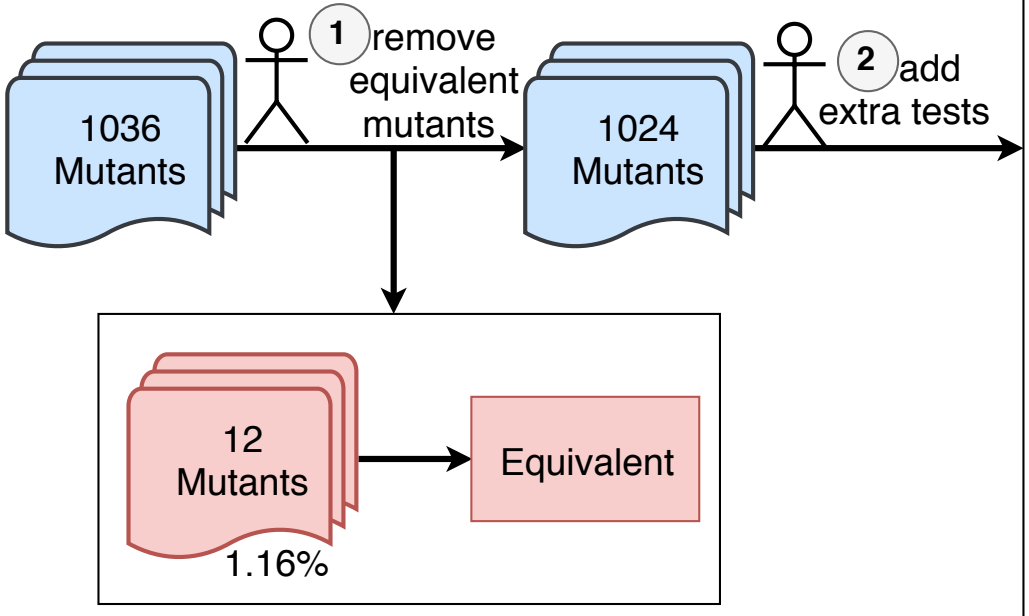
- **Effectiveness of mutants**

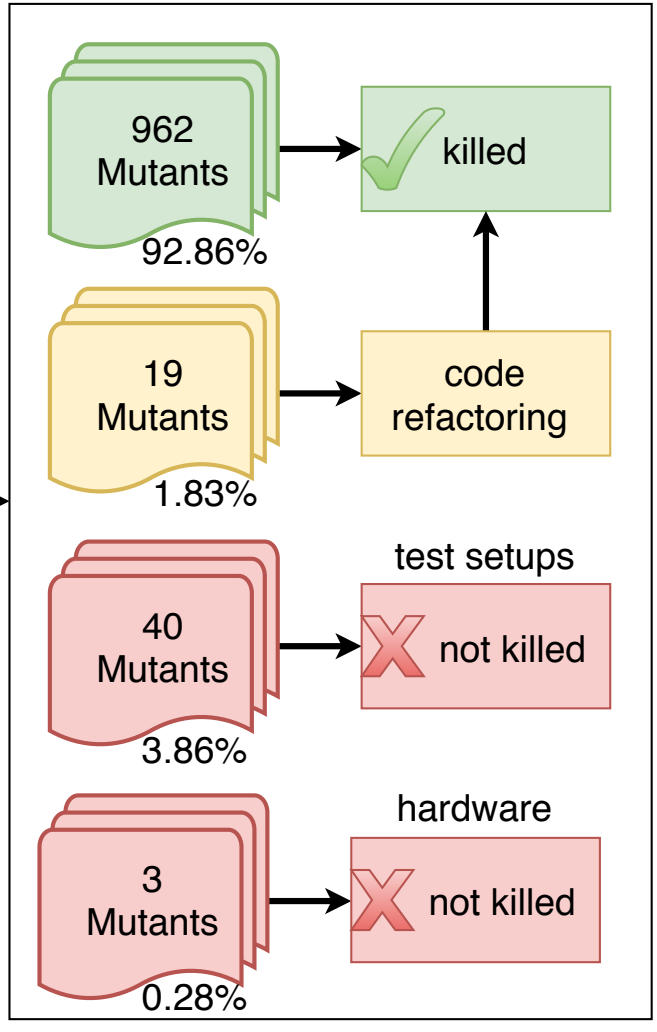


Results

- **Effectiveness of mutants**

- 94.69% mutants → effective
- mutation score ≠ test quality
- non-killable ⊆ equivalent?



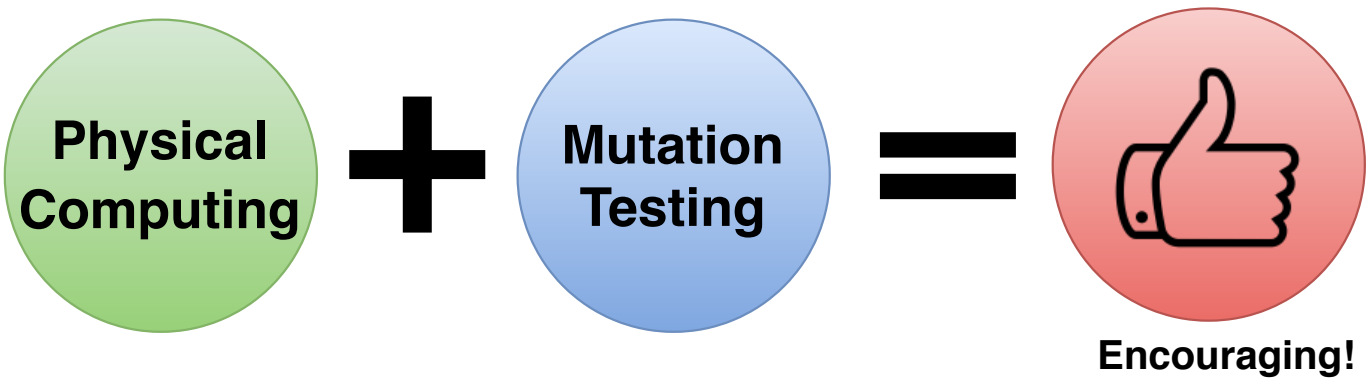


Summary

- **What we have done:**
  - 9 mutation operators for physical computing
  - a preliminary experiment on 9 systems

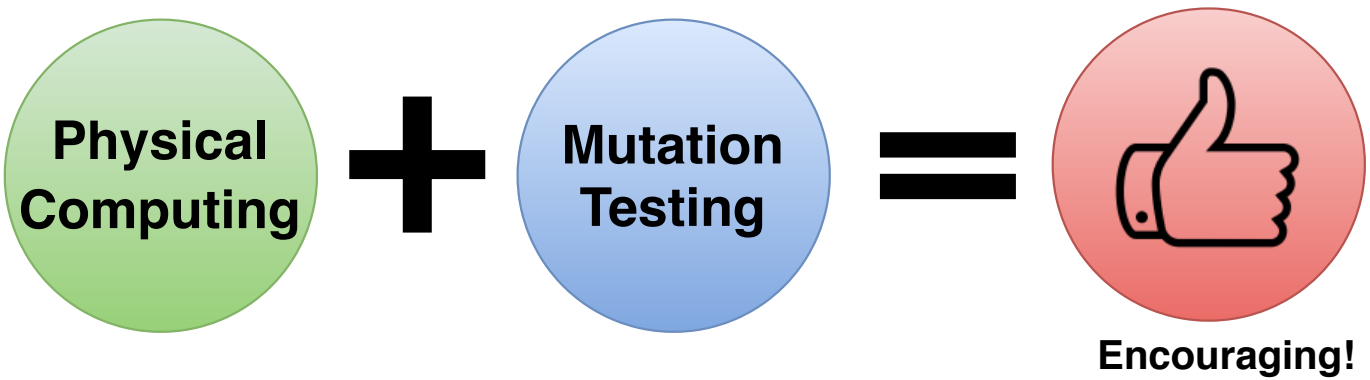
Summary

- **What we have done:**
  - 9 mutation operators for physical computing
  - a preliminary experiment on 9 systems
- **What we have learned:**



Summary

- **What we have done:**
  - 9 mutation operators for physical computing
  - a preliminary experiment on 9 systems
- **What we have learned:**



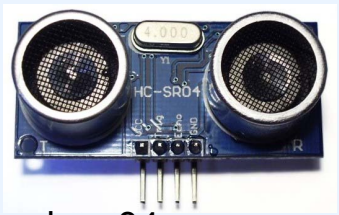
- **What we plan to do:**
  - more case studies
  - traditional mutation operators **vs.** ours
  - fault-finding ability **vs.** mutation score

Case studies with Raspberry Pi

- **Test environment:**
  Raspberry Pi → one-chip computer, mutation testing tool
- **Subjects:**

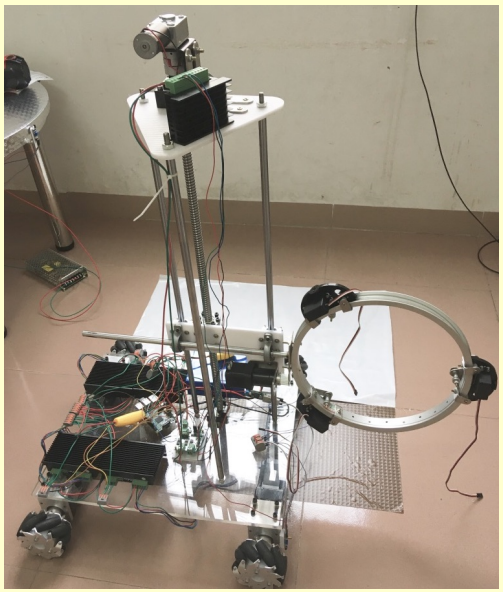

jean-pierre

gpiozero



hcsr04sensor

RPLCD

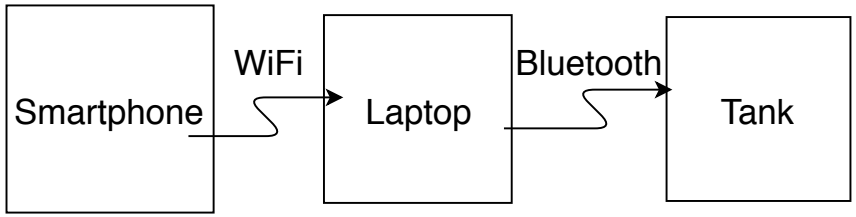

four-wheel robot

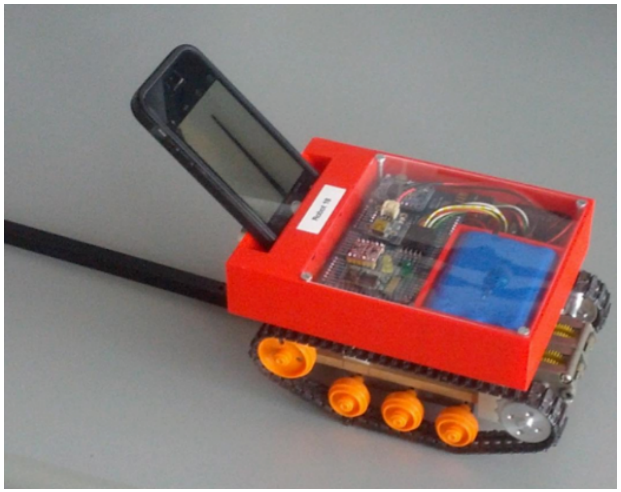
GitHub

Kompline
康湃蓝电子

Case studies with Arduino

- **Subject:** line-follower robot





- **Test environment:** hardware monitor