Two ways to use AI for assurance of critical software

JULY 15, 2024

Bjorn Andersson

Carnegie Mellon University
Software Engineering Institute

Carnegie Mellon University Software Engineering Institute

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM24-0796

Agenda

- My background (and how it relates to space)
- Using Large-Language Model (LLM) for Hazard Analysis
- Using Artificial Intelligence for Worst-Case Execution Time Analysis

My background (and how it relates to space)

Carnegie Mellon University Software Engineering Institute

Apollo program:

We need to schedule real-time tasks on a single processor.

1969

time

Apollo program:
Fixed-priority preemptive scheduling is a good idea [1].

1969 time

[1] Liu, C. L. Scheduling algorithms for hard-real-time multiprogramming of a single processor. JPL Space Programs Summary 37,60, Vol. II, Jet Propulsion Lab., Calif. Inst. of Tech., Pasadena, Calif., Nov. 1969

Apollo program:

Assign priorities to process according to rate-monotonic (short period yields high priority) [1].

1969 time

[1] Liu, C. L. Scheduling algorithms for hard-real-time multiprogramming of a single processor. JPL Space Programs Summary 37,60, Vol. II, Jet Propulsion Lab., Calif. Inst. of Tech., Pasadena, Calif., Nov. 1969

Apollo program: Single processor system: Rate-Monotonic has utilization bound 69% [1].

1969 time

[1] Liu, C. L. Scheduling algorithms for hard-real-time multiprogramming of a single processor. JPL Space Programs Summary 37,60, Vol. II, Jet Propulsion Lab., Calif. Inst. of Tech., Pasadena, Calif., Nov. 1969

Apollo program: Single processor system: Rate-Monotonic has utilization bound 69% [1].

A rich research literature and practice was developed for Rate-Monotonic.

1969 — time

[1] Liu, C. L. Scheduling algorithms for hard-real-time multiprogramming of a single processor. JPL Space Programs Summary 37,60, Vol. II, Jet Propulsion Lab., Calif. Inst. of Tech., Pasadena, Calif., Nov. 1969

Apollo program: Multi-processor system: Rate-Monotonic has utilization bound approaching 0% [2].

1969 time

[2] C. Liu, "Scheduling algorithms for multiprocessors in a hard real-time environment," in JPL Space Programs Summary, vol. 37-60. JPL, Pasadena, CA, 28–31, 1969.

Apollo program:

Multi-processor system: Rate-Monotonic has utilization bound approaching 0% [2].

Multi-processor system: There is another way of assigning priorities; this yields utilization bound 33% [3].

1969

2001

time

[3] B. Andersson, S. Baruah, and J. Jonsson, "Static-priority scheduling on multiprocessors," IEEE RTSS, 2001.

Two ways to use AI for Assurance

Using Large-Language Model (LLM) for Hazard Analysis

Carnegie Mellon University Software Engineering Institute

Hazard Analysis is about Safety

Environment (that we can't control)

Technical system (that we design and can control)

A mishap occurs when (i) a certain condition is true about the technical system, and (ii) a certain condition is true about the environment.

Hazard Analysis is about Safety

Environment (that we can't control)

Technical system (that we design and can control)

Since we can't control the environment, let us focus on what we can control; that is, let us focus on the technical system.

Hazard Analysis is about Safety

Environment (that we can't control)

Technical system (that we design and can control)

A hazard of a technical systems is a condition such that if this condition is true, and the environment is in a bad state, then a mishap occurs.

Hazard Analysis is about Safety

Environment (that we can't control)

Technical system (that we design and can control)

If we could find all hazards and eliminate them, then we would eliminate all mishaps.

Hazard Analysis is about Safety

Environment (that we can’t control)

Technical system (that we design and can control)

In practice, we can’t find all hazards but we can find many of them and we can try to eliminate them.

Hazard Analysis is about Safety

Environment (that we can't control)

Technical system (that we design and can control)

Hazard analysis is about finding hazards.

Hazard Analysis is about Safety

Carnegie Mellon University Software Engineering Institute

Hazard analysis is about looking at documentation to find hazards.

Hazard Analysis is about Safety

Environment (that we can't control)

Technical system (that we design and can control)

Hazard analysis is about looking at documentation from various perspectives to find hazards. One perspective yields one hazard analysis. Another perspective yields another hazard analysis.

Hazard Analysis is about Safety

Environment (that we can't control)

Technical system (that we design and can control)

Hazard analysis is not about proving correctness properties. It is about discovering issues.

Hazard Analysis is Laborious and Expensive

Environment (that we can’t control)

Technical system (that we design and can control)

It takes a lot of time and money for humans to read and analyze documents.

Hazard Analysis is Hard to Automate

Environment (that we can't control)

Technical system (that we design and can control)

It requires common-sense reasoning, contextual knowledge about the technical system and its environment, and background knowledge (that most humans have).

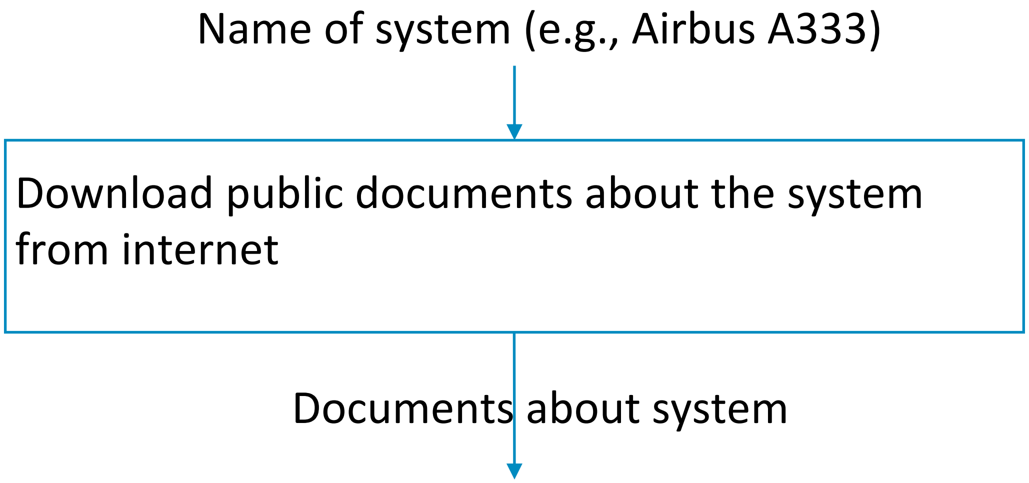
Hazard Analysis is Hard to Automate

Environment (that we can't control)

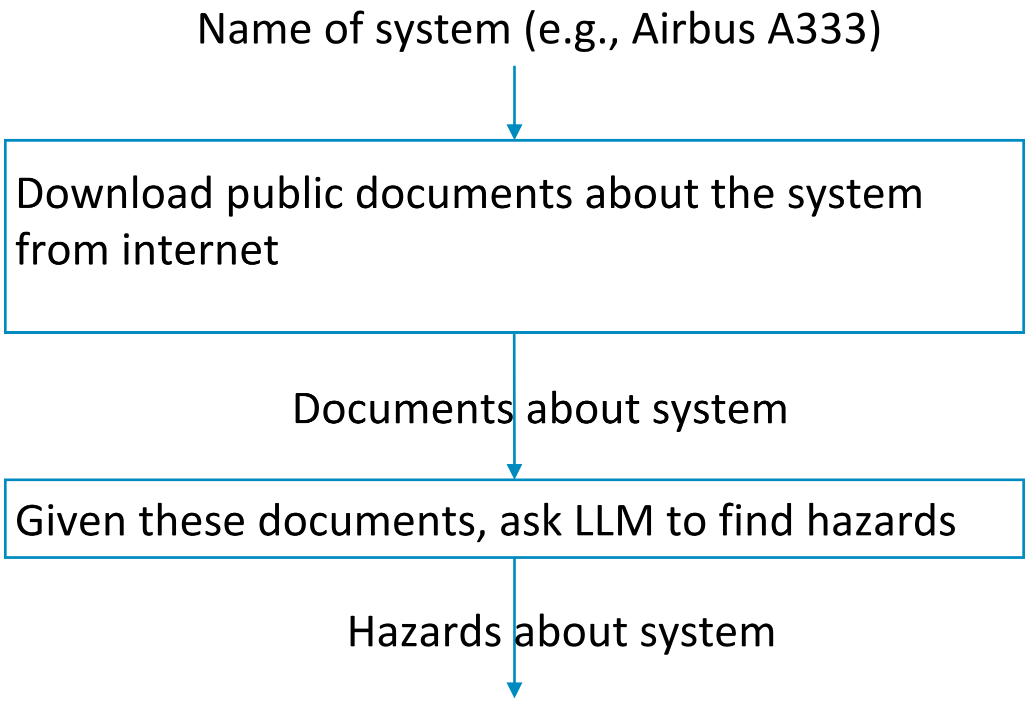
Technical system (that we design and can control)

Idea: An LLM can "simulate" human thinking. Hence, using an LLM for hazard analysis seems worthwhile.

Our Tool for Hazard Analysis using LLM



Our Tool for Hazard Analysis using LLM



Our Tool for Hazard Analysis using LLM

Name of system (e.g., Airbus A333)

↓

Download public documents about the system from internet

↓ Documents about system

Given these documents, ask LLM to find hazards

↓

"The Airbus A330 carries various hazards related to operational, maintenance, and environmental factors, including its electrical, fuel, engine, hydraulic, and wastewater systems. ….."

The Quality of the Output from Hazard Analysis Depends on the Quality of Input

Carnegie Mellon University Software Engineering Institute

Name of system (e.g., Airbus A333)

Download public documents about the system from internet

Documents about system

Given these documents, ask LLM to find hazards

"The Airbus A330 carries various hazards related to operational, maintenance, and environmental factors, including its electrical, fuel, engine, hydraulic, and wastewater systems. ….."

Hazard Analysis based on Detailed Documents (Proprietary) yields better output than Superficial Documents (Publicly Available)

Name of system (e.g., Airbus A333)

Download public documents about the system from internet

Documents about system

Given these documents, ask LLM to find hazards

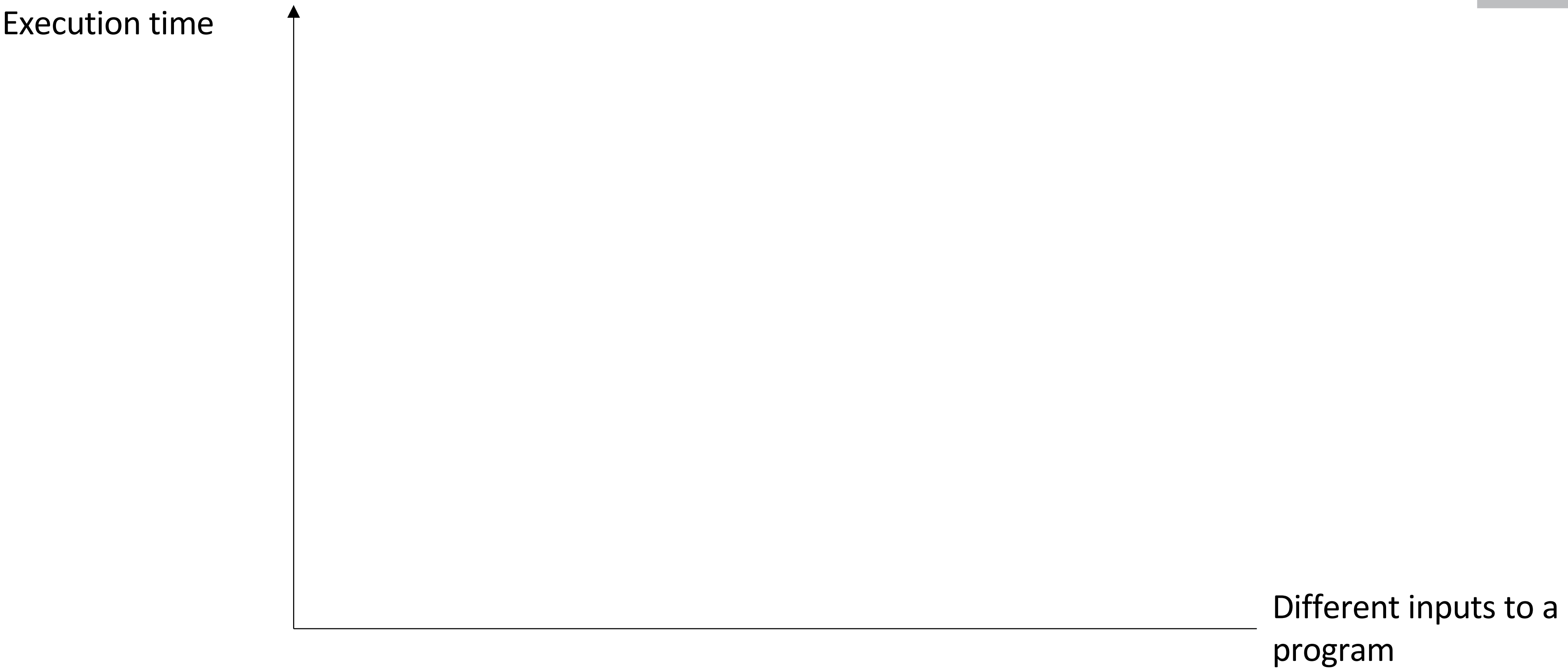
“The Airbus A330 carries various hazards related to operational, maintenance, and environmental factors, including its electrical, fuel, engine, hydraulic, and wastewater systems. …..”

Two ways to use AI for assurance

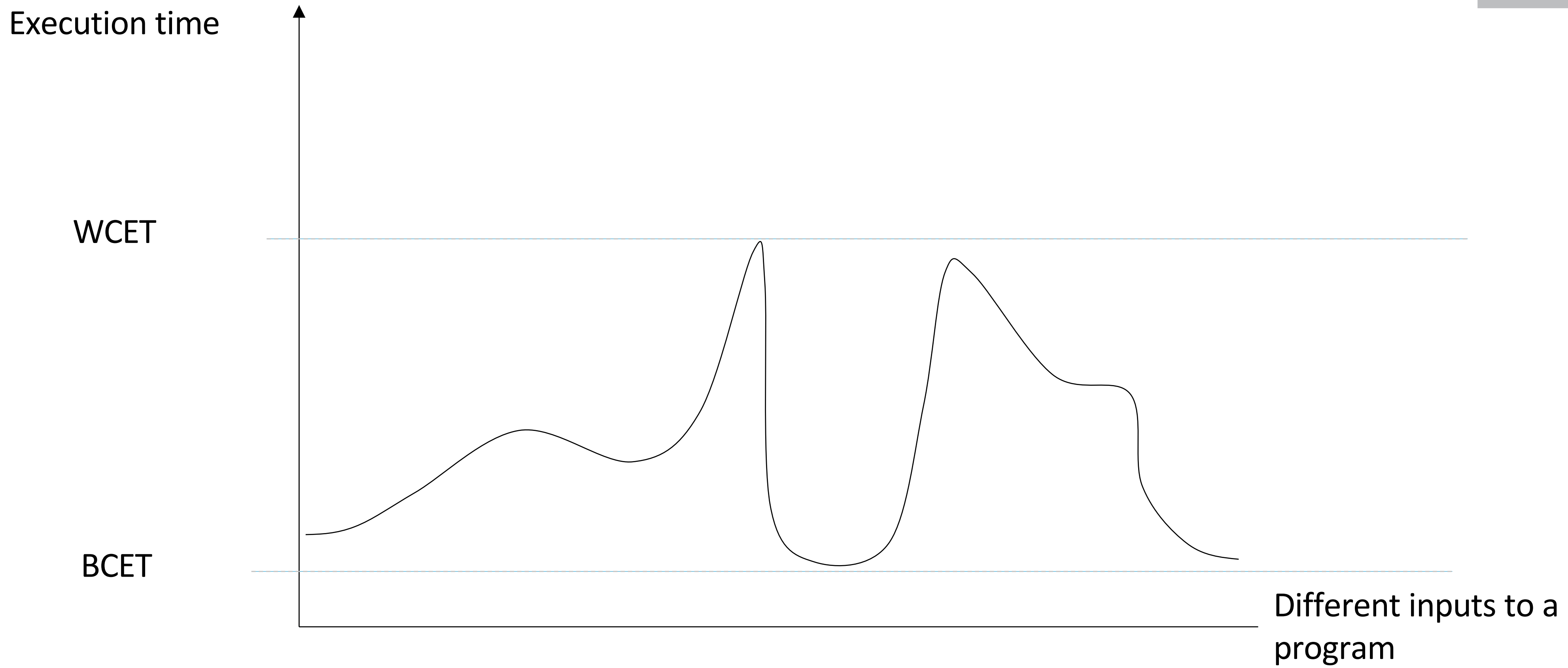
Using Artificial Intelligence for Worst-Case Execution Time Analysis

Carnegie Mellon University Software Engineering Institute

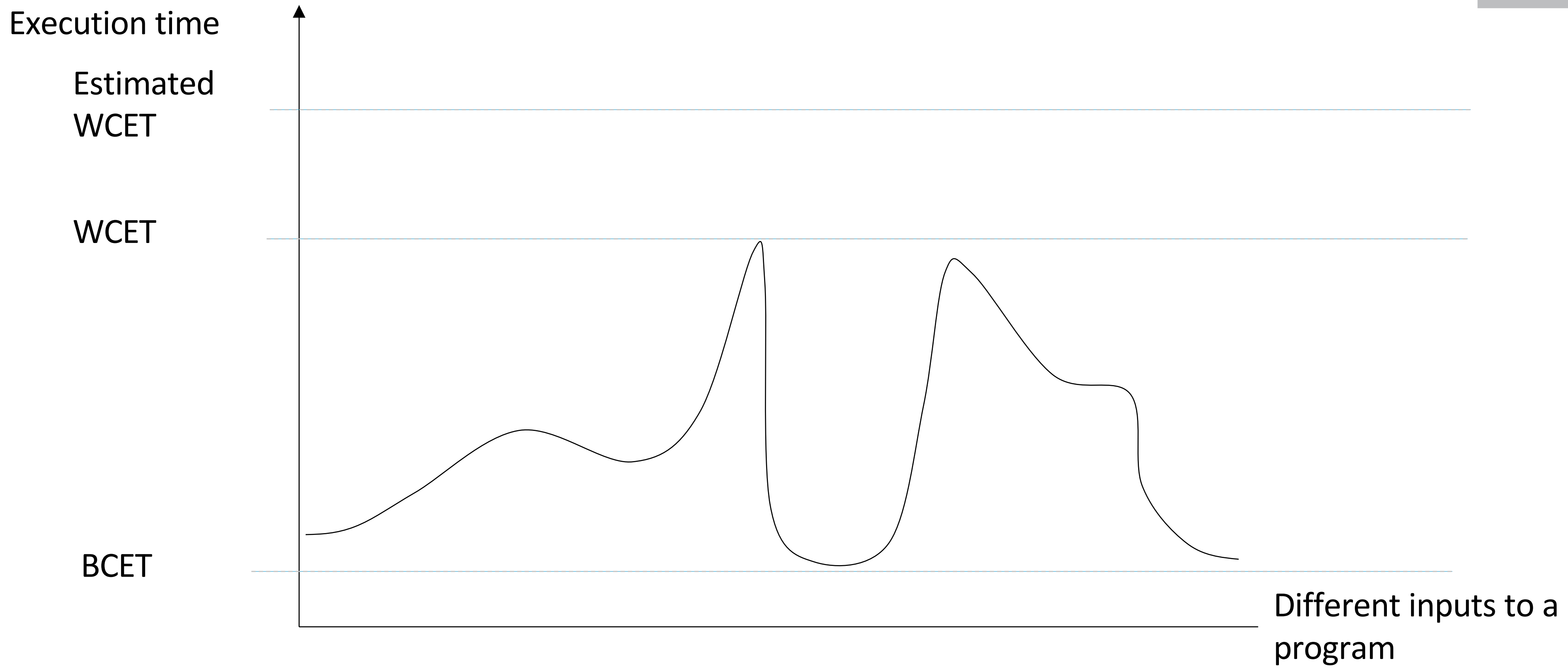
What is Worst-Case Execution Time?



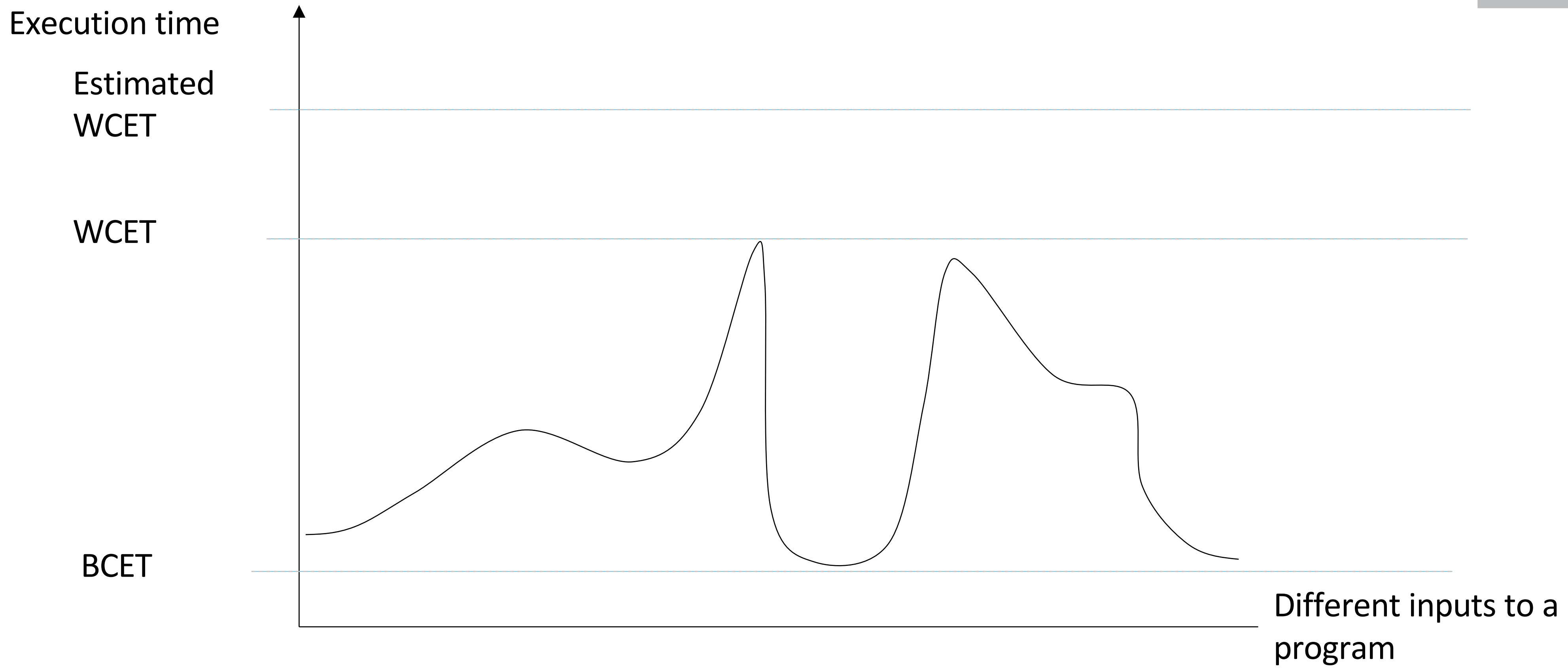
What is Worst-Case Execution Time?



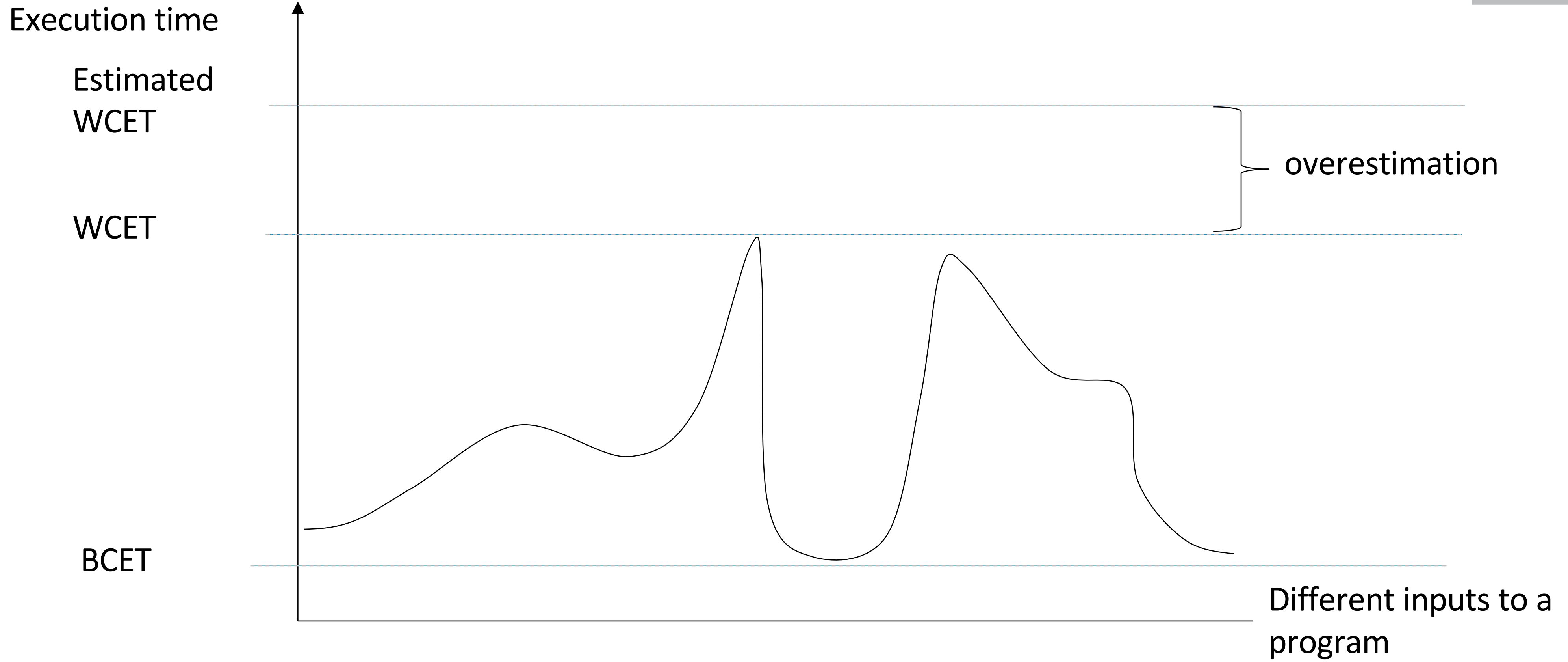
What is Worst-Case Execution Time Estimate?



What is Worst-Case Execution Time Estimate?
Find WCET estimate



What is Worst-Case Execution Time Estimate? Find WCET estimate with small overestimation



Why is Worst-Case Execution Time Analysis Challenging?

Carnegie Mellon University Software Engineering Institute

Software Complexity

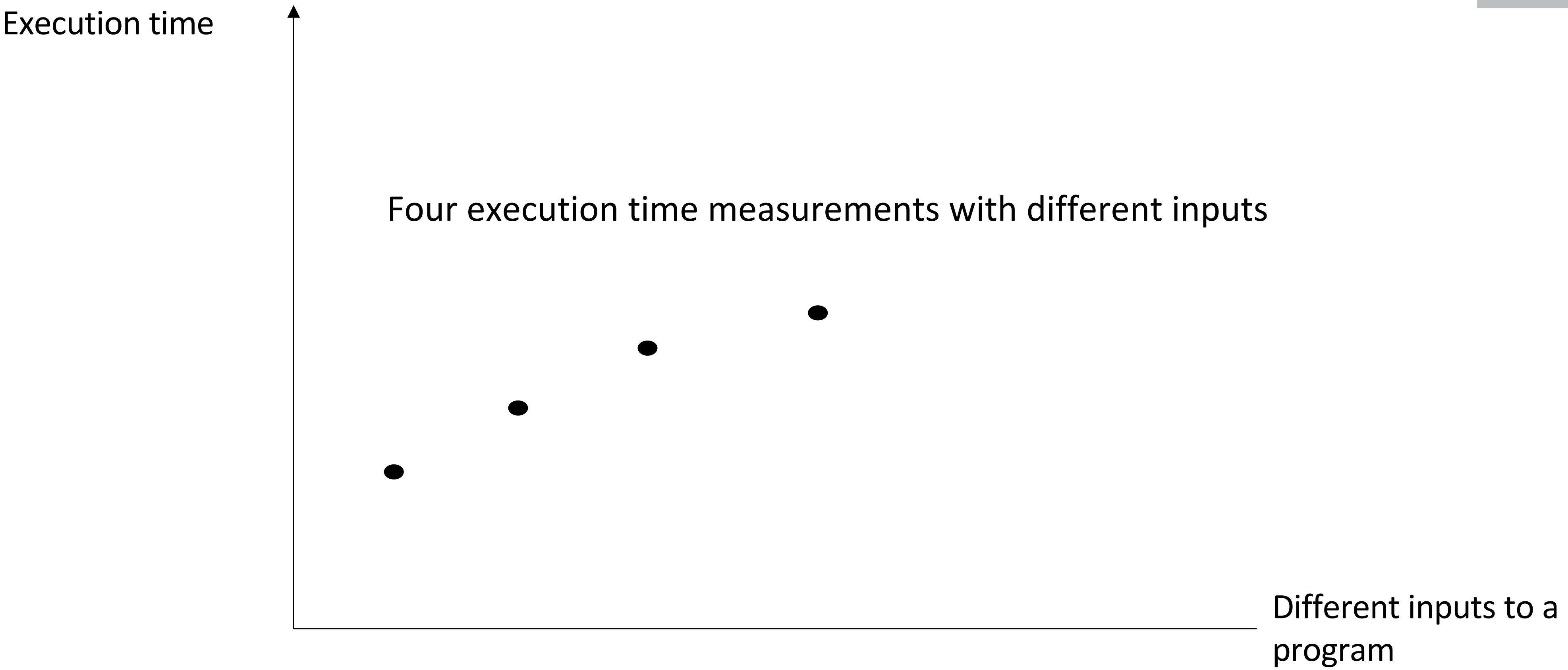
- The number of execution paths in a program tends to be very large and input dependent. We cannot explicitly enumerate all of them.

Hardware Complexity

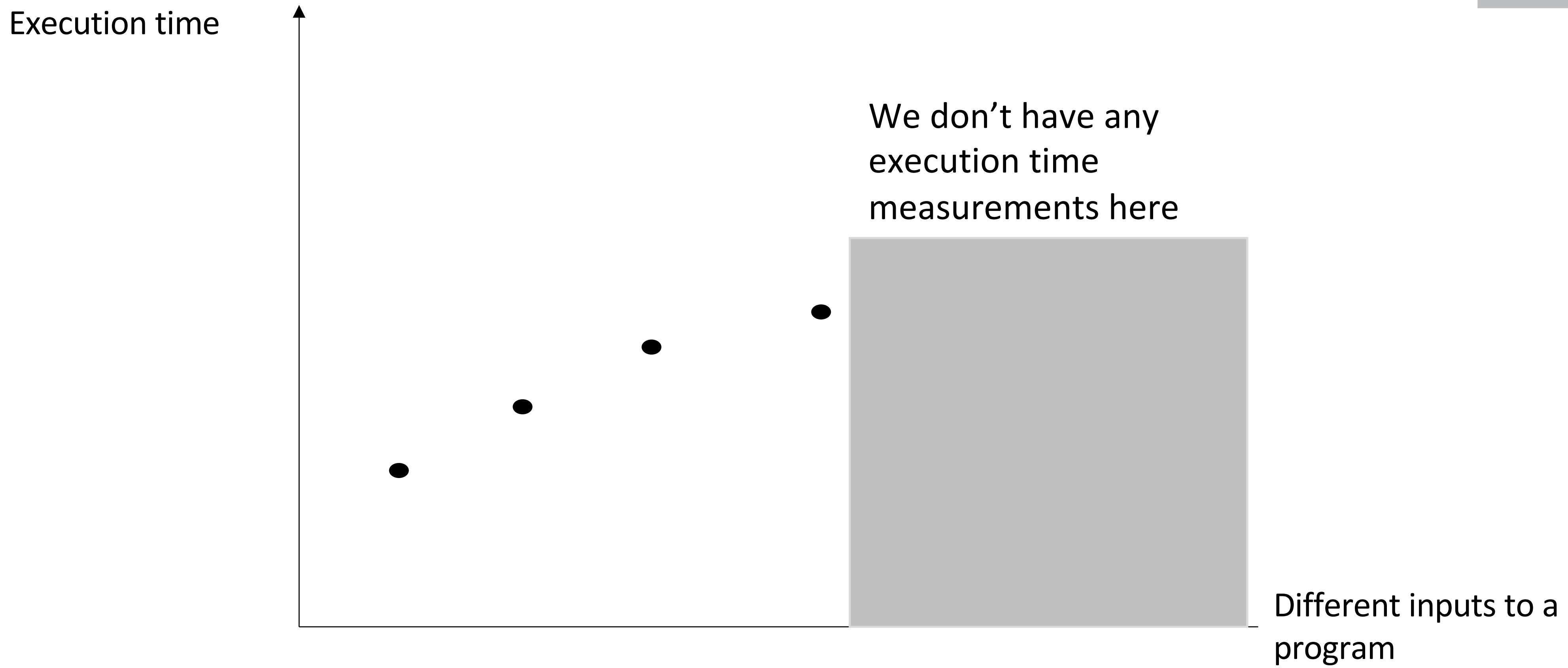
- Even for a single path in a program, the execution time depends on (i) initial state (variable initialization), (ii) state of the hardware (dirty cache blocks initially), (iii)behavior of hardware (cache, pipelining, etc).
- On a multicore, it gets even more complex because the execution time of a program depends on co-runners.

Our Method

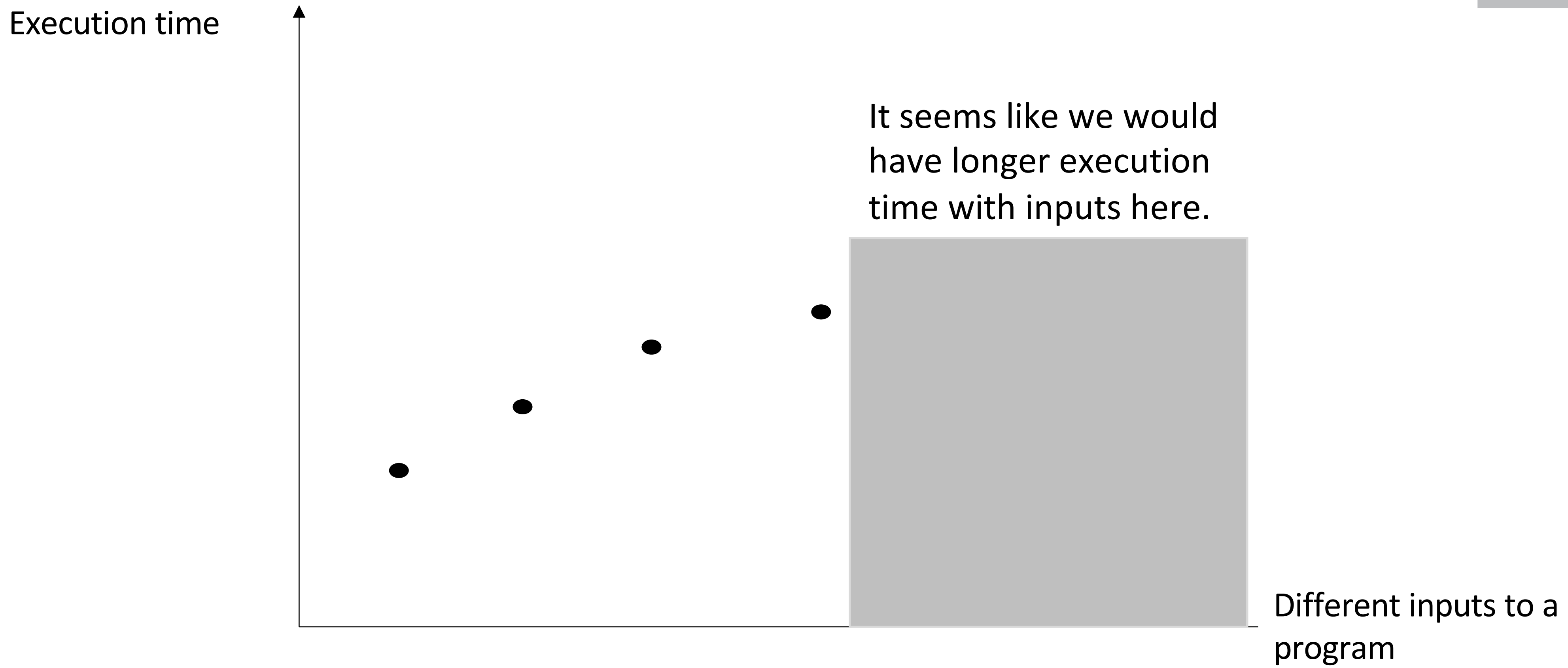
A Highly Simplified View of Our Method



A Highly Simplified View of Our Method

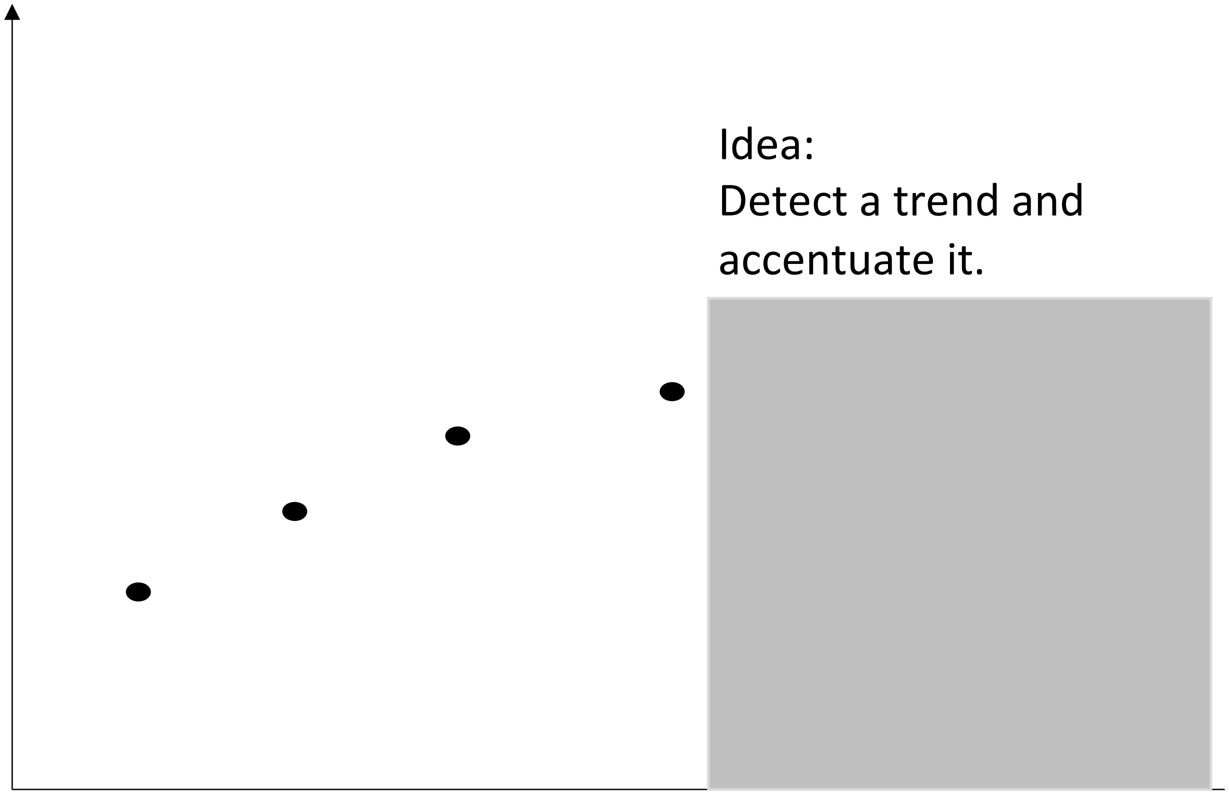


A Highly Simplified View of Our Method



A Highly Simplified View of Our Method

Execution time



Idea:
Detect a trend and accentuate it.

Different inputs to a program

Our Method

Carnegie Mellon University Software Engineering Institute

Step 1: Generate random inputs

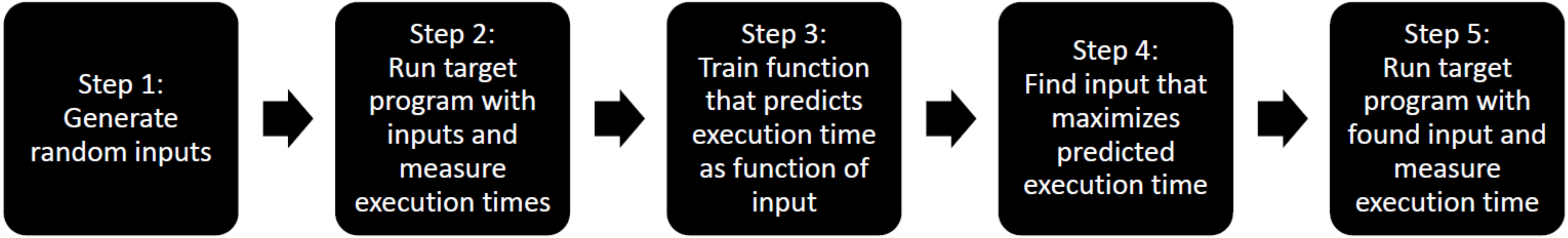
→ Step 2: Run target program with inputs and measure execution times

→ Step 3: Train function that predicts execution time as function of input

→ Step 4: Find input that maximizes predicted execution time

→ Step 5: Run target program with found input and measure execution time

Our Method



Our method can find a WCET estimate even if (i) we do not have source code of program, and (ii) we do not have documentation of hardware.

Our Method

1. **Step 1:** Generate random inputs
2. **Step 2:** Run target program with inputs and measure execution times
3. **Step 3:** Train function that predicts execution time as function of input
4. **Step 4:** Find input that maximizes predicted execution time
5. **Step 5:** Run target program with found input and measure execution time

Our method can find counter-intuitive effects that humans can’t find and other WCET analysis tools can’t find.

Our Method

1. Step 1: Generate random inputs
2. Step 2: Run target program with inputs and measure execution times
3. Step 3: Train function that predicts execution time as function of input
4. Step 4: Find input that maximizes predicted execution time
5. Step 5: Run target program with found input and measure execution time

Our method can find counter-intuitive effects that humans can’t find and other WCET analysis tools can’t find.

Details at:
DOT/FAA/TC-23/06 Assessing the Use of Machine Learning to Find the Worst-Case Execution Time of Avionics Software
https://www.faa.gov/aircraft/air_cert/design_approvals/air_software/research/TC2306

Bjorn Andersson
Principal Researcher

Telephone: +1 412.268.9243
Email: baandersson@sei.cmu.edu