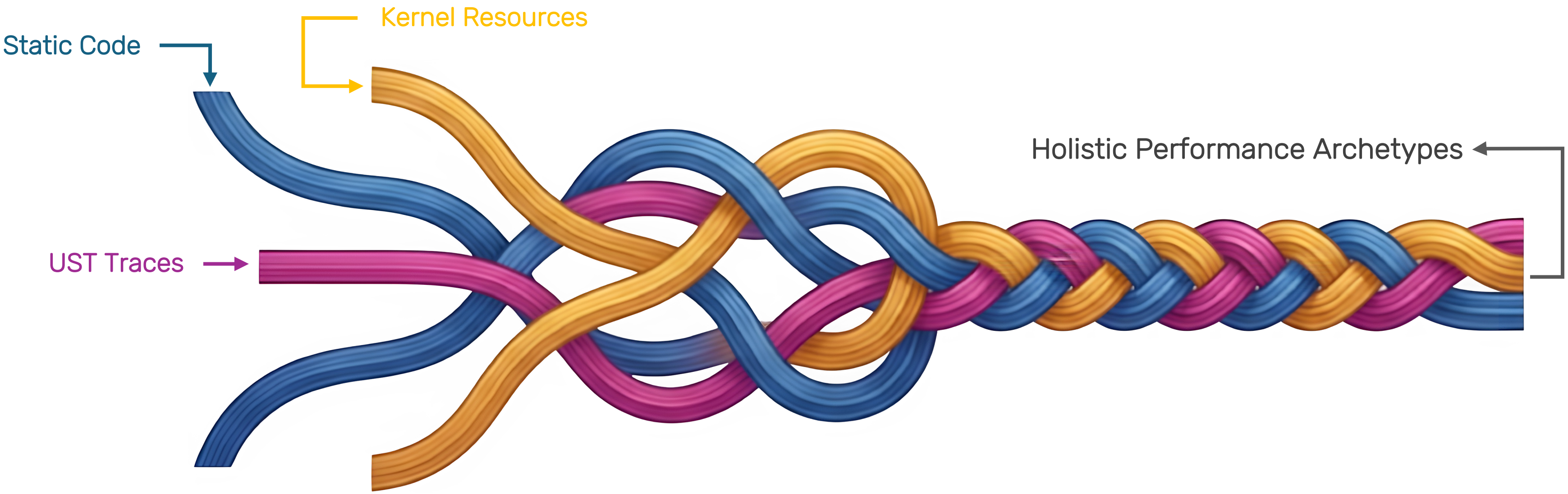
Performance Archetypes

Multi-Execution Critical Path Pattern Analysis



Static Code

Kernel Resources

UST Traces

Holistic Performance Archetypes

Kaveh Shahedi, Heng Li

Dorsal Progress Report Meeting – February 2026

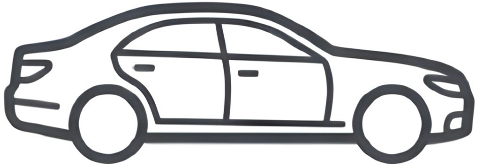
POLYTECHNIQUE MONTRÉAL

DISCLAIMER

The term “Critical Path” does NOT mean the usual thing in this work

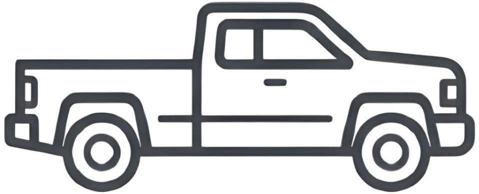
(e.g., from Trace Compass)

The Universal ”Morning Commute”



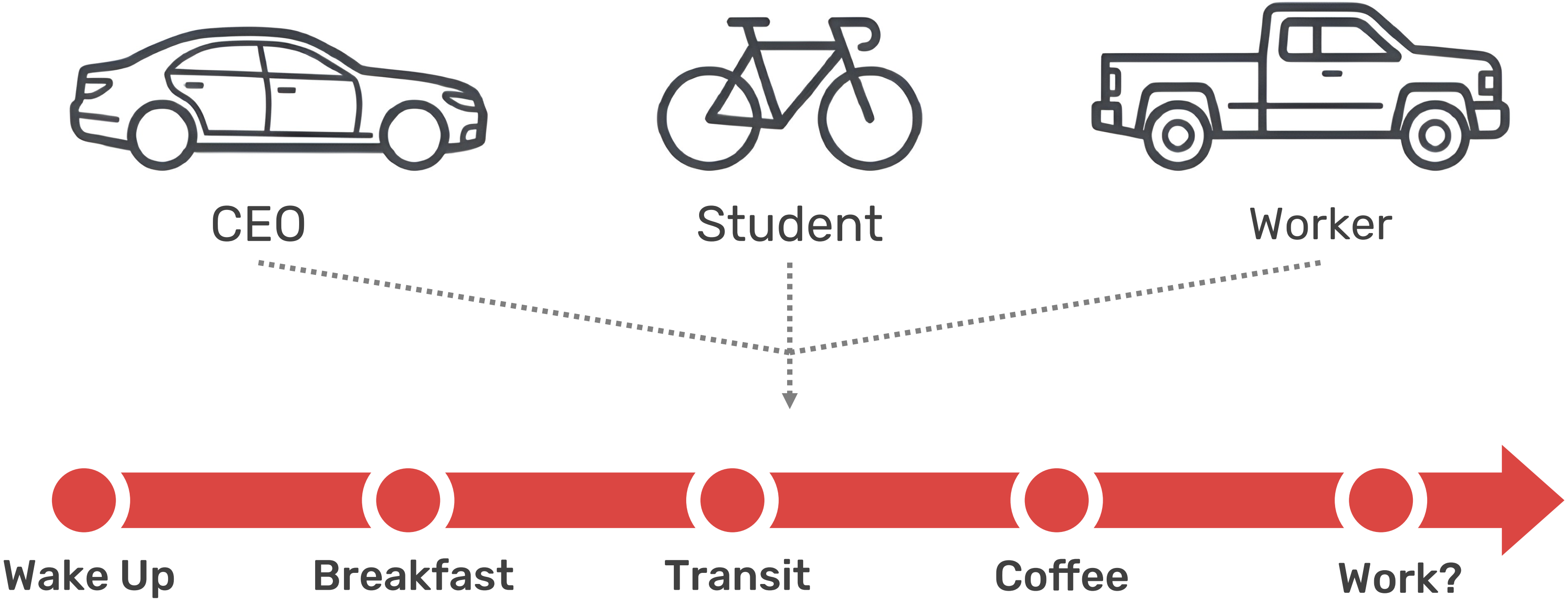
CEO

Student

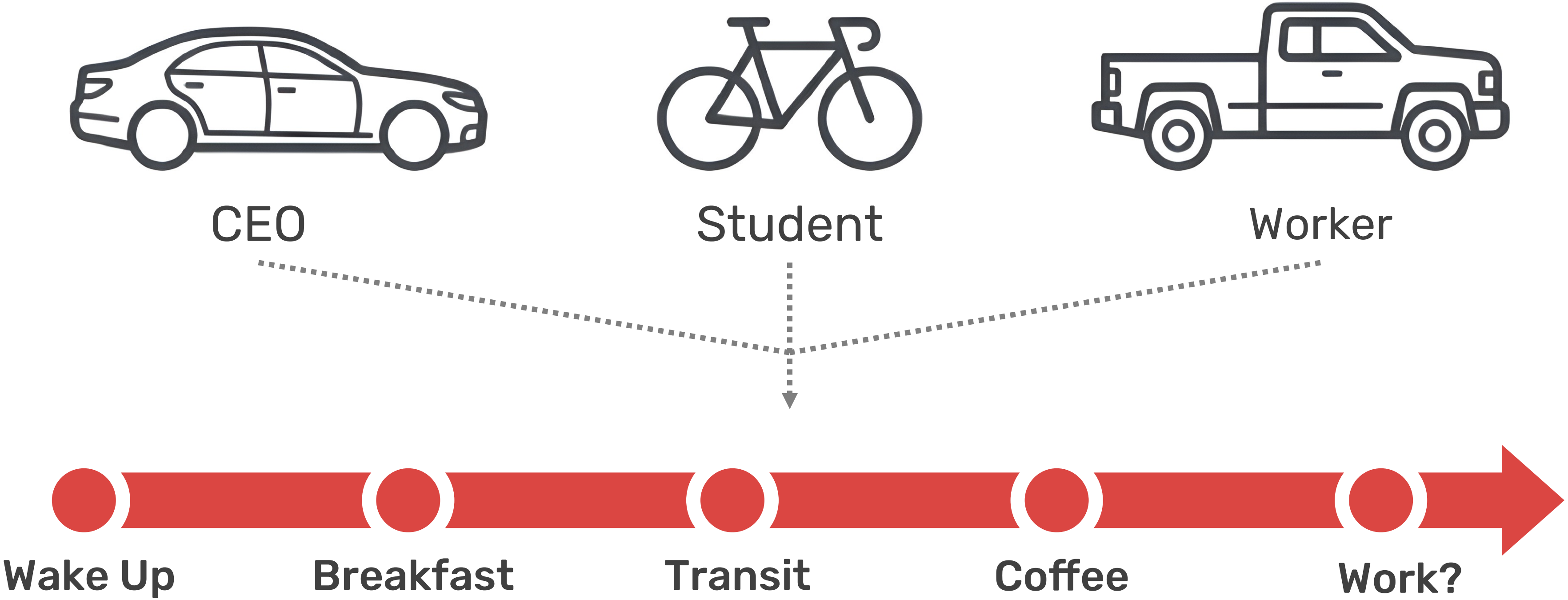


Worker

The Universal "Morning Commute"



The Universal "Morning Commute"



CEO

Student

Worker

Wake Up

Breakfast

Transit

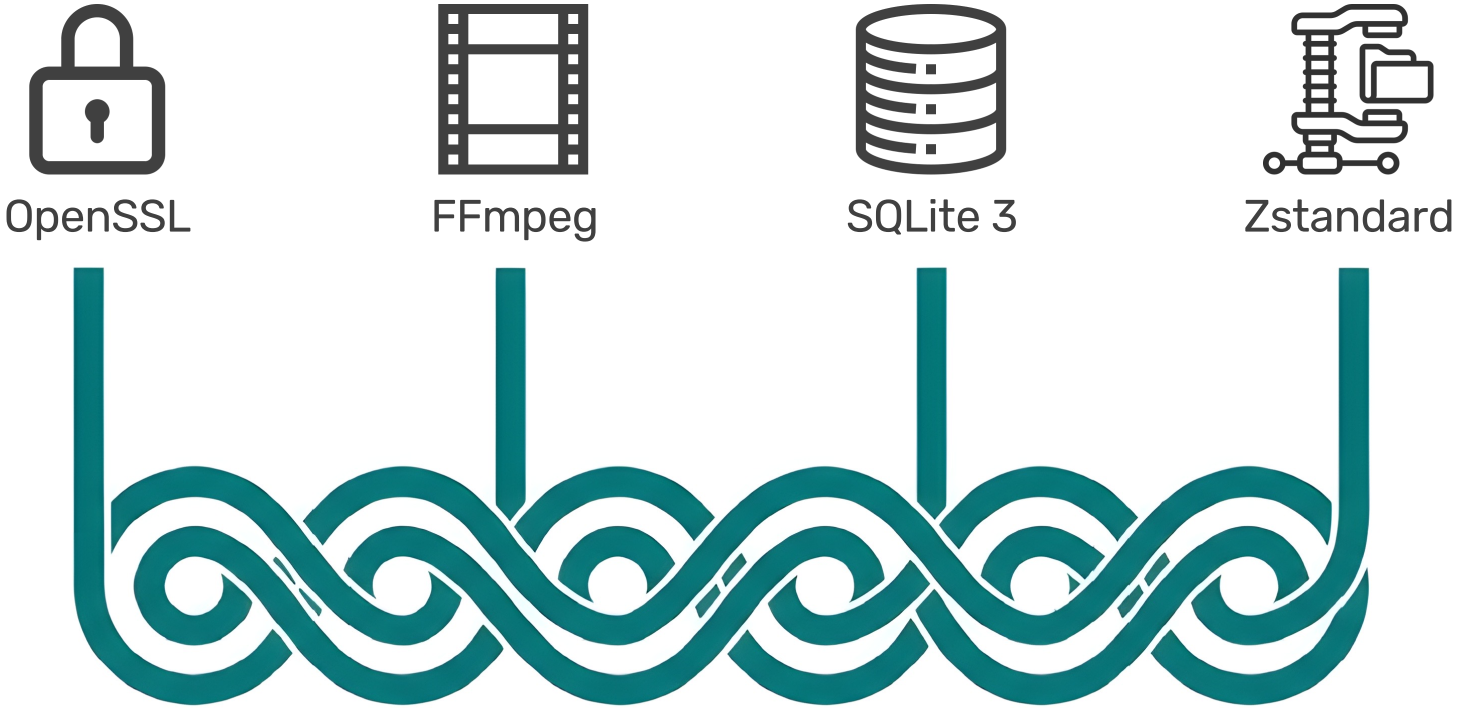
Coffee

Work?

different inputs, different individuals, almost identical behavioral archetypes

Thousands of Movies, Yet All Sharing Similar Patterns

Do Applications Have Personalities?



OpenSSL

FFmpeg

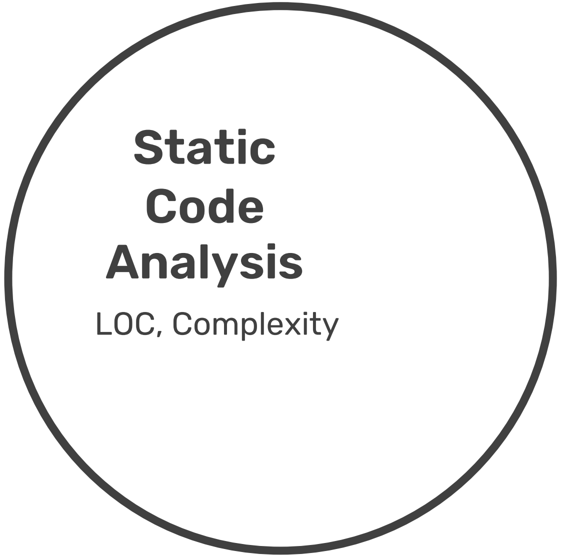
SQLite 3

Zstandard

distinct codebases, universal execution language

Triangulating the Personalities

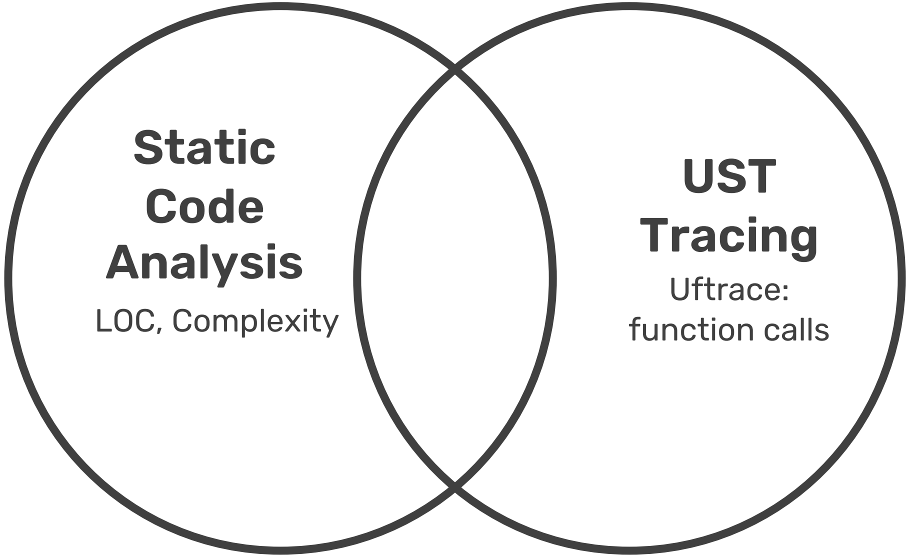
Triangulating the Personalities



Static Code Analysis

LOC, Complexity

Triangulating the Personalities



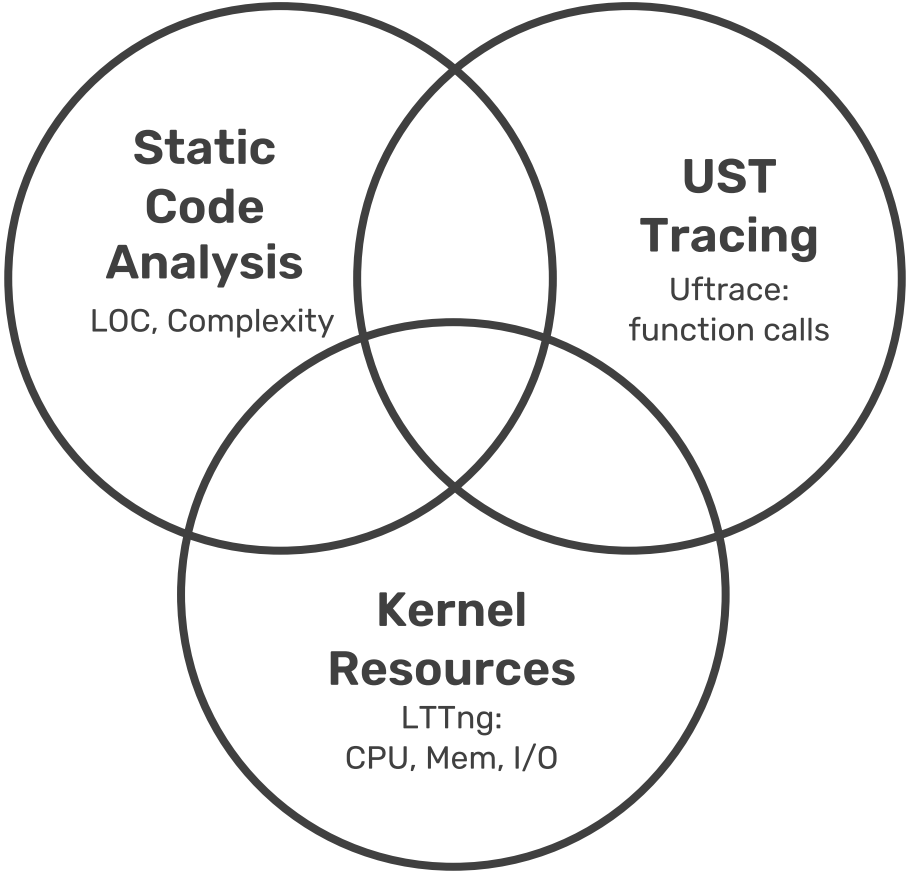
Static Code Analysis

LOC, Complexity

UST Tracing

Uftrace: function calls

Triangulating the Personalities



Static Code Analysis

LOC, Complexity

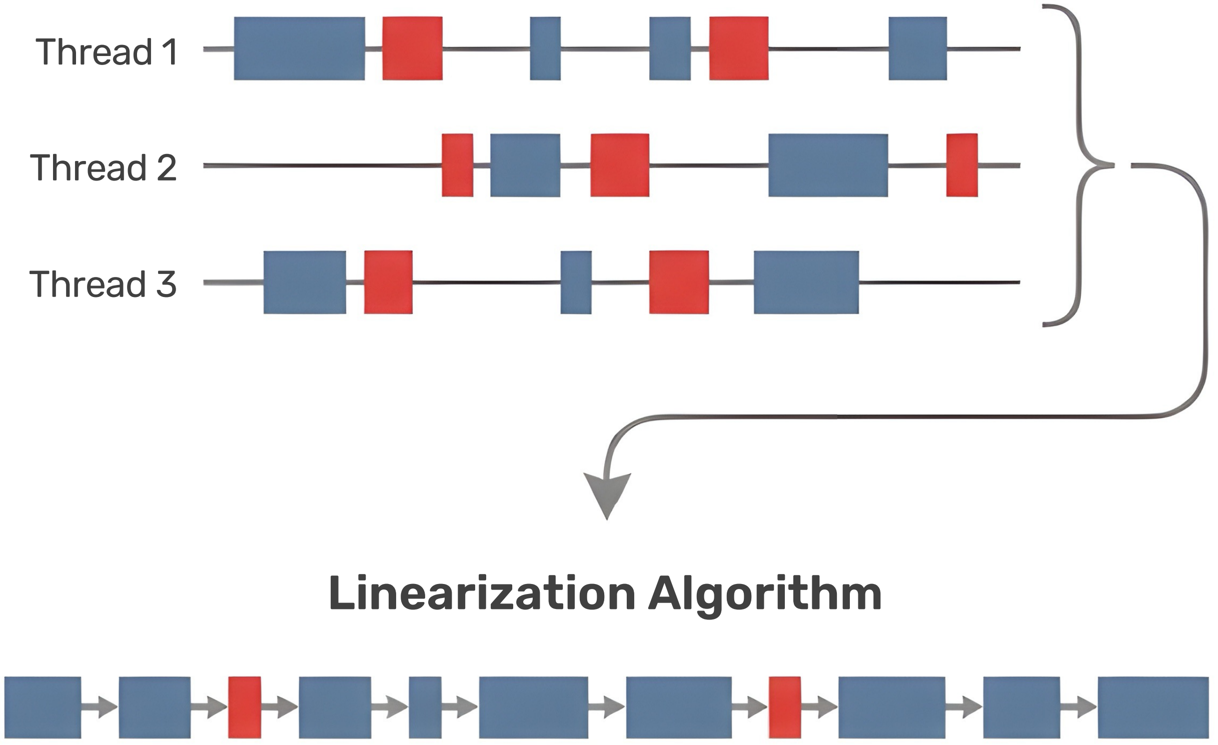
UST Tracing

Uftrace: function calls

Kernel Resources

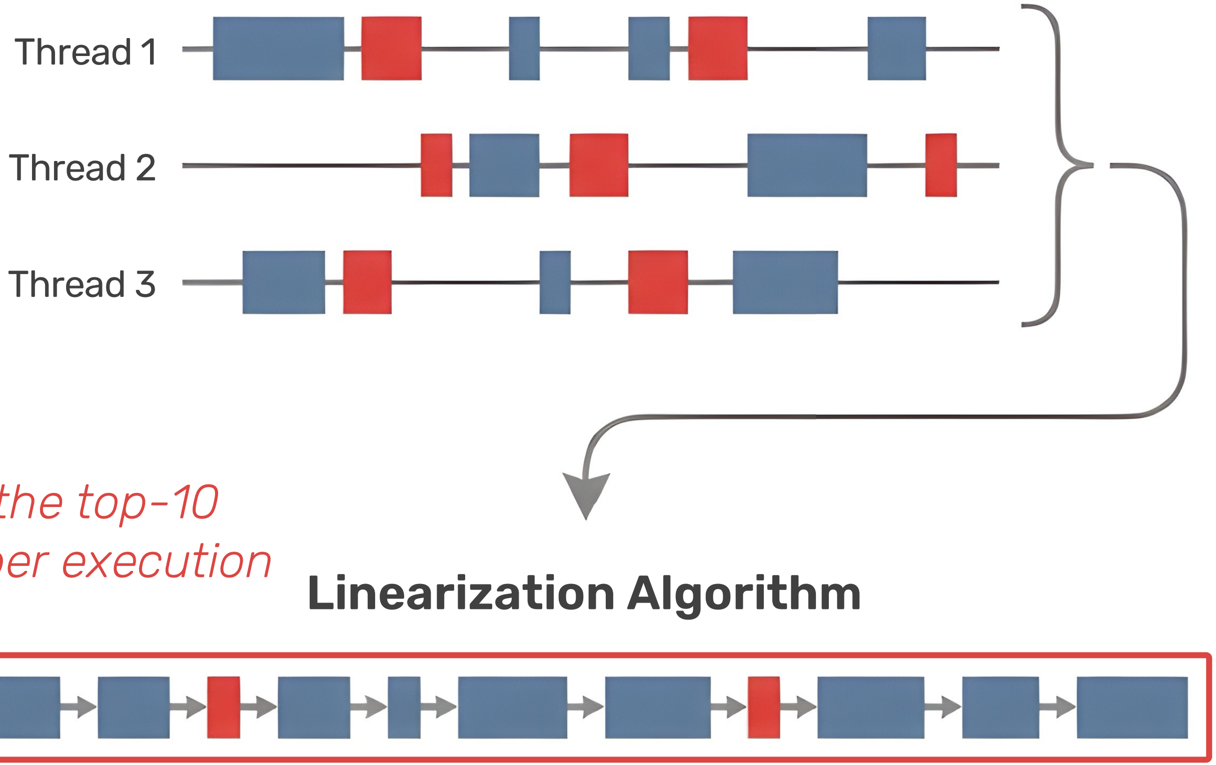
LTTng: CPU, Mem, I/O

Extracting Function Call Stack Critical Paths



Unified Critical Path

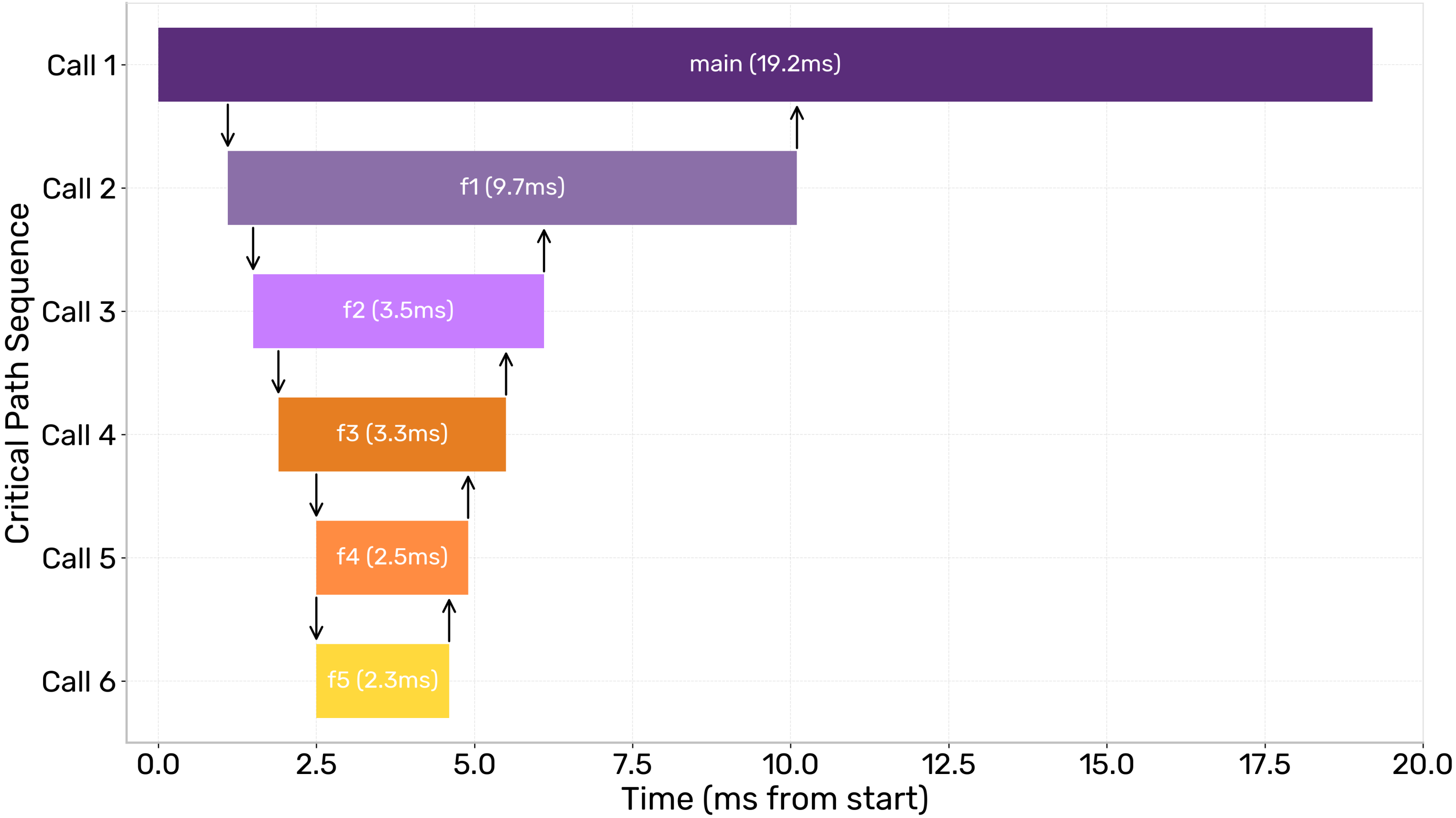
Extracting Function Call Stack Critical Paths



We do this for the top-10 critical paths per execution

Unified Critical Path

Extracting Function Call Stack Critical Paths



The Dataset

SQLite 3
(I/O & Compute Mixed)

FFmpeg
(I/O Streaming)

OpenSSL
(CPU Heavy)

Zstandard
(Memory & CPU)

Inputs
500 per app

↓

Iterations
3 per input

↓

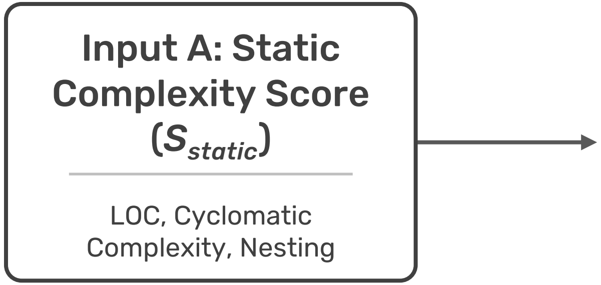
Total Paths
~50,000

RQ1: The Static-Dynamic Paradox?

Input A: Static Complexity Score (S_{static})

LOC, Cyclomatic Complexity, Nesting

RQ1: The Static-Dynamic Paradox?

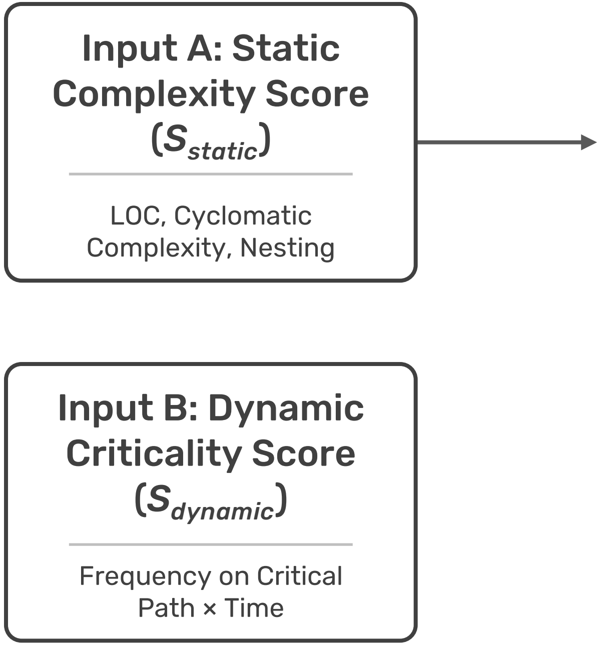


Input A: Static Complexity Score (S_{static})

LOC, Cyclomatic Complexity, Nesting

1. Function A: 0.95 (very complex)
2. Function B: 0.88
3. Function C: 0.59
4. Function D: 0.30
5. Function E: 0.12 (very simple)

RQ1: The Static-Dynamic Paradox?



Input A: Static Complexity Score (S_{static})

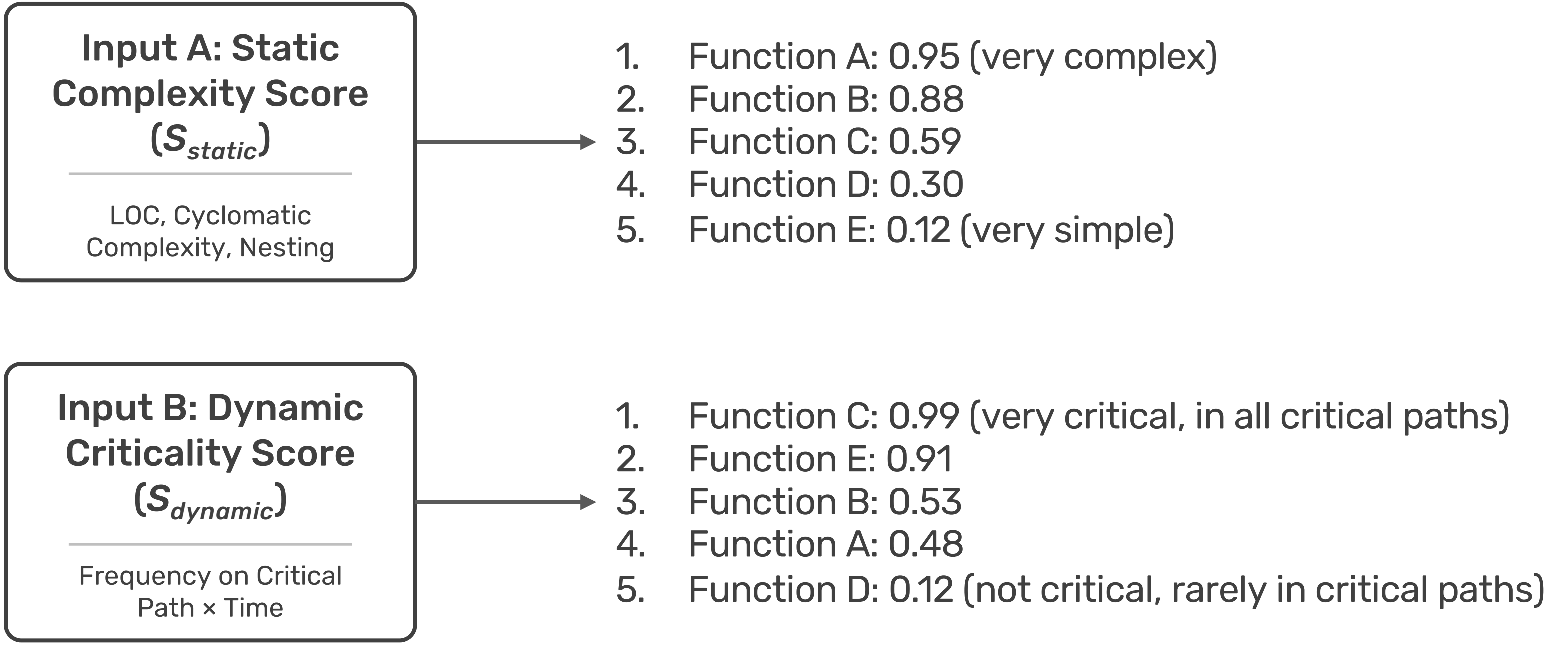
LOC, Cyclomatic Complexity, Nesting

Input B: Dynamic Criticality Score ($S_{dynamic}$)

Frequency on Critical Path × Time

1. Function A: 0.95 (very complex)
2. Function B: 0.88
3. Function C: 0.59
4. Function D: 0.30
5. Function E: 0.12 (very simple)

RQ1: The Static-Dynamic Paradox?



Input A: Static Complexity Score (S_{static})

LOC, Cyclomatic Complexity, Nesting

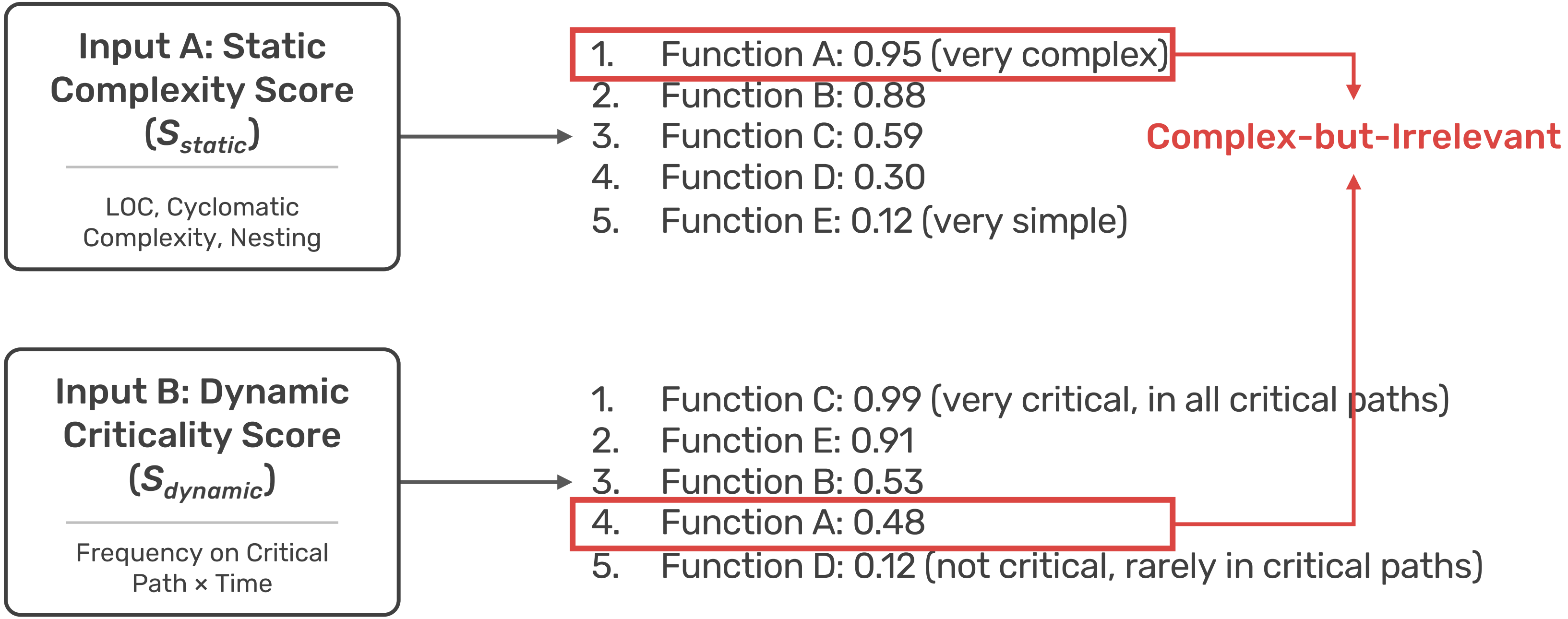
1. Function A: 0.95 (very complex)
2. Function B: 0.88
3. Function C: 0.59
4. Function D: 0.30
5. Function E: 0.12 (very simple)

Input B: Dynamic Criticality Score ($S_{dynamic}$)

Frequency on Critical Path × Time

1. Function C: 0.99 (very critical, in all critical paths)
2. Function E: 0.91
3. Function B: 0.53
4. Function A: 0.48
5. Function D: 0.12 (not critical, rarely in critical paths)

RQ1: The Static-Dynamic Paradox?



Input A: Static Complexity Score (S_{static})

LOC, Cyclomatic Complexity, Nesting

1. Function A: 0.95 (very complex)
2. Function B: 0.88
3. Function C: 0.59
4. Function D: 0.30
5. Function E: 0.12 (very simple)

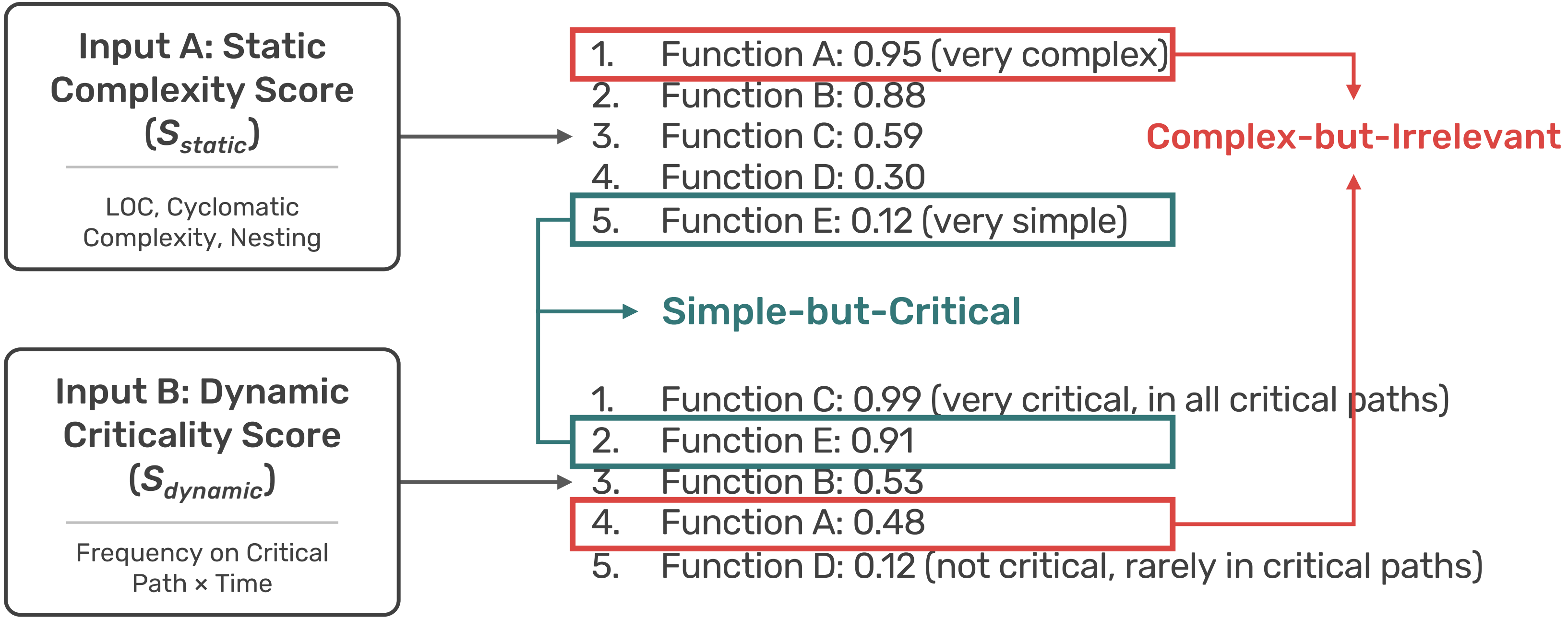
Complex-but-Irrelevant

Input B: Dynamic Criticality Score ($S_{dynamic}$)

Frequency on Critical Path × Time

1. Function C: 0.99 (very critical, in all critical paths)
2. Function E: 0.91
3. Function B: 0.53
4. Function A: 0.48
5. Function D: 0.12 (not critical, rarely in critical paths)

RQ1: The Static-Dynamic Paradox?



Input A: Static Complexity Score (S_{static})

LOC, Cyclomatic Complexity, Nesting

1. Function A: 0.95 (very complex)
2. Function B: 0.88
3. Function C: 0.59
4. Function D: 0.30
5. Function E: 0.12 (very simple)

Complex-but-Irrelevant

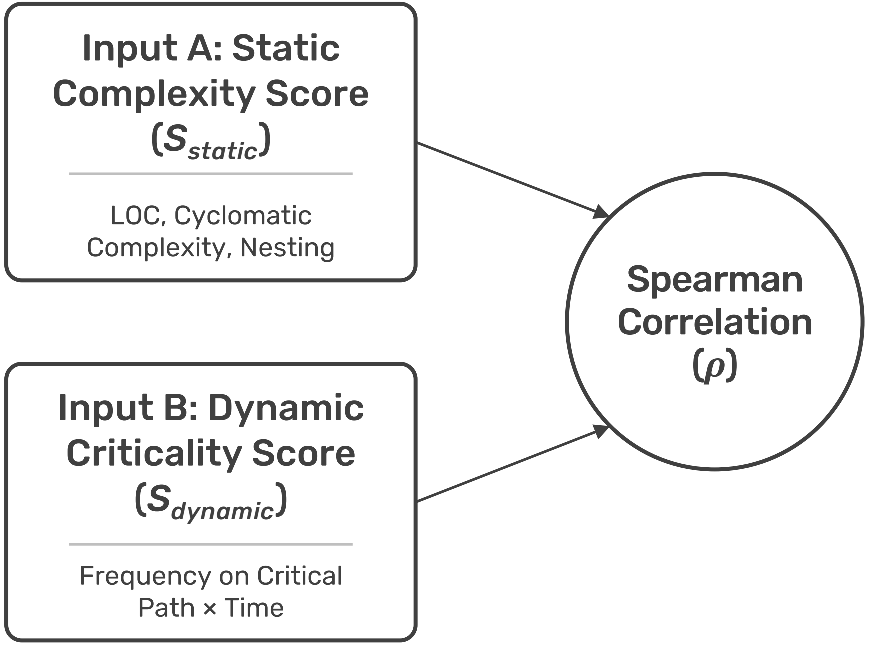
Simple-but-Critical

Input B: Dynamic Criticality Score ($S_{dynamic}$)

Frequency on Critical Path × Time

1. Function C: 0.99 (very critical, in all critical paths)
2. Function E: 0.91
3. Function B: 0.53
4. Function A: 0.48
5. Function D: 0.12 (not critical, rarely in critical paths)

RQ1: The Static-Dynamic Paradox?



Input A: Static Complexity Score (S_{static})

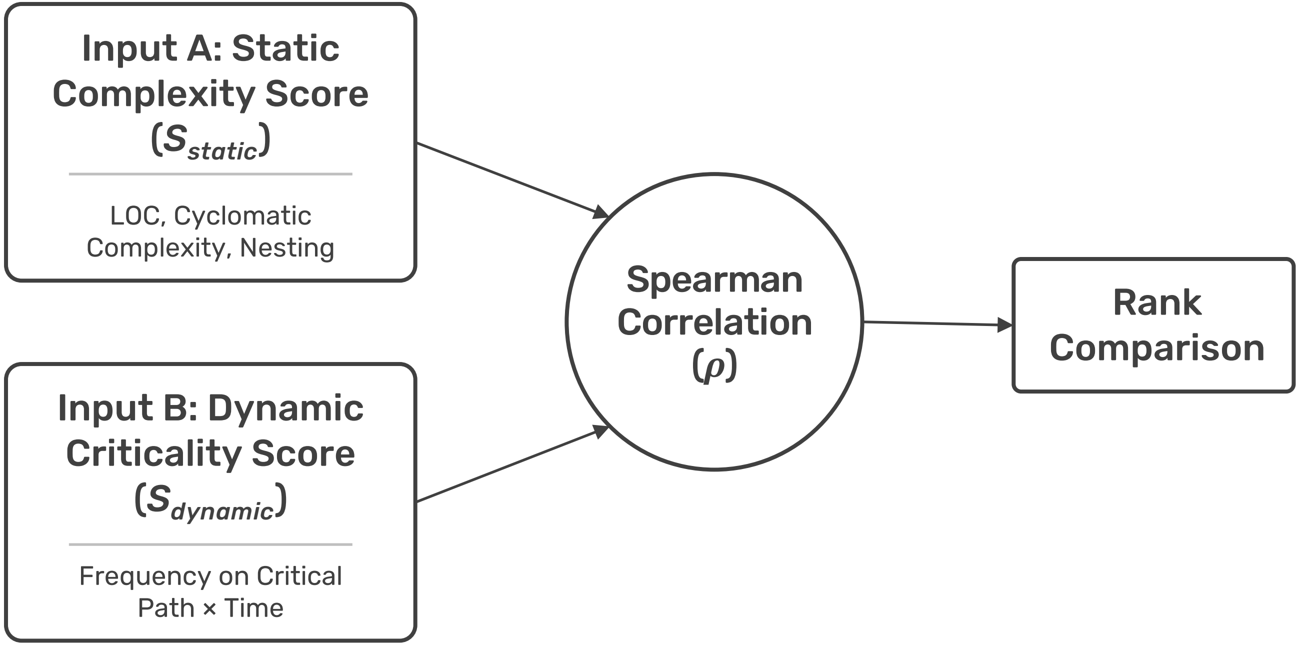
LOC, Cyclomatic Complexity, Nesting

Input B: Dynamic Criticality Score ($S_{dynamic}$)

Frequency on Critical Path × Time

Spearman Correlation (ρ)

RQ1: The Static-Dynamic Paradox?



Input A: Static Complexity Score (S_{static})

LOC, Cyclomatic Complexity, Nesting

Input B: Dynamic Criticality Score ($S_{dynamic}$)

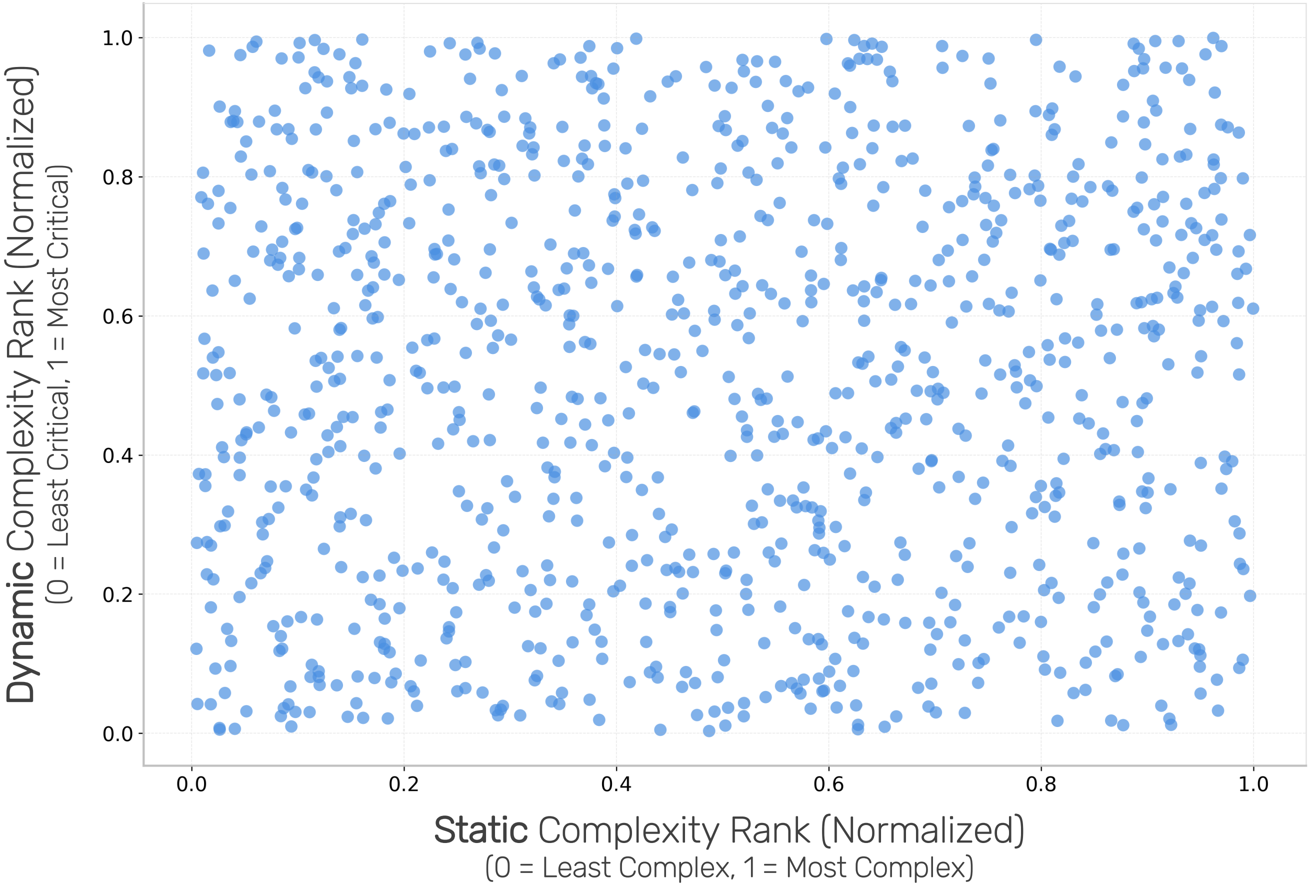
Frequency on Critical Path × Time

Spearman Correlation (ρ)

Rank Comparison

11.0%

Static metrics explain only 11% of runtime performance variance



11.0%

Average variance explained by the static metrics

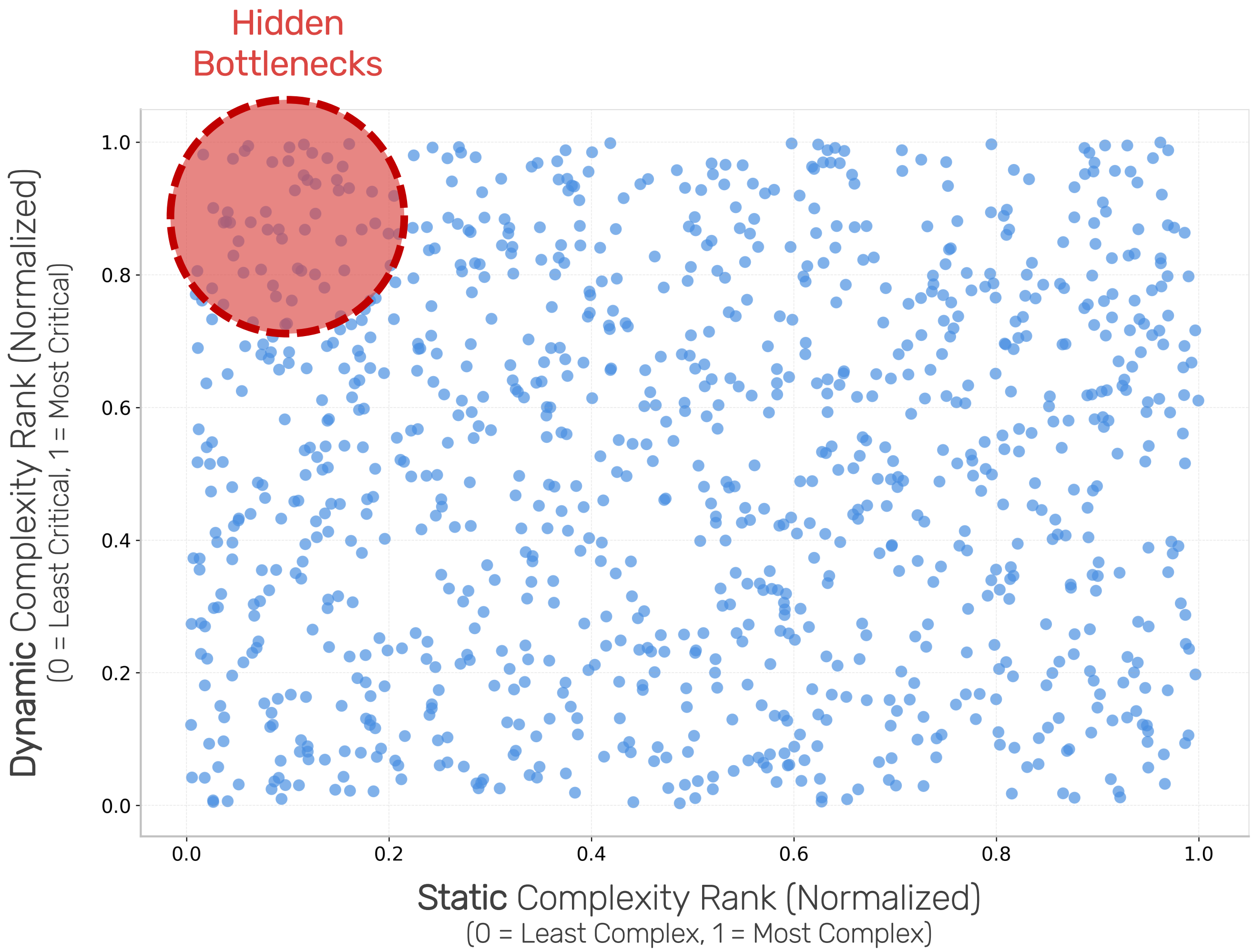
Spearman Correlations

SQLite 3: -0.041 (zero)

FFmpeg: 0.175 (Weak)

Zstandard: 0.336 (Weak)

OpenSSL: 0.543 (Moderate)



11.0%

Average variance explained by the static metrics

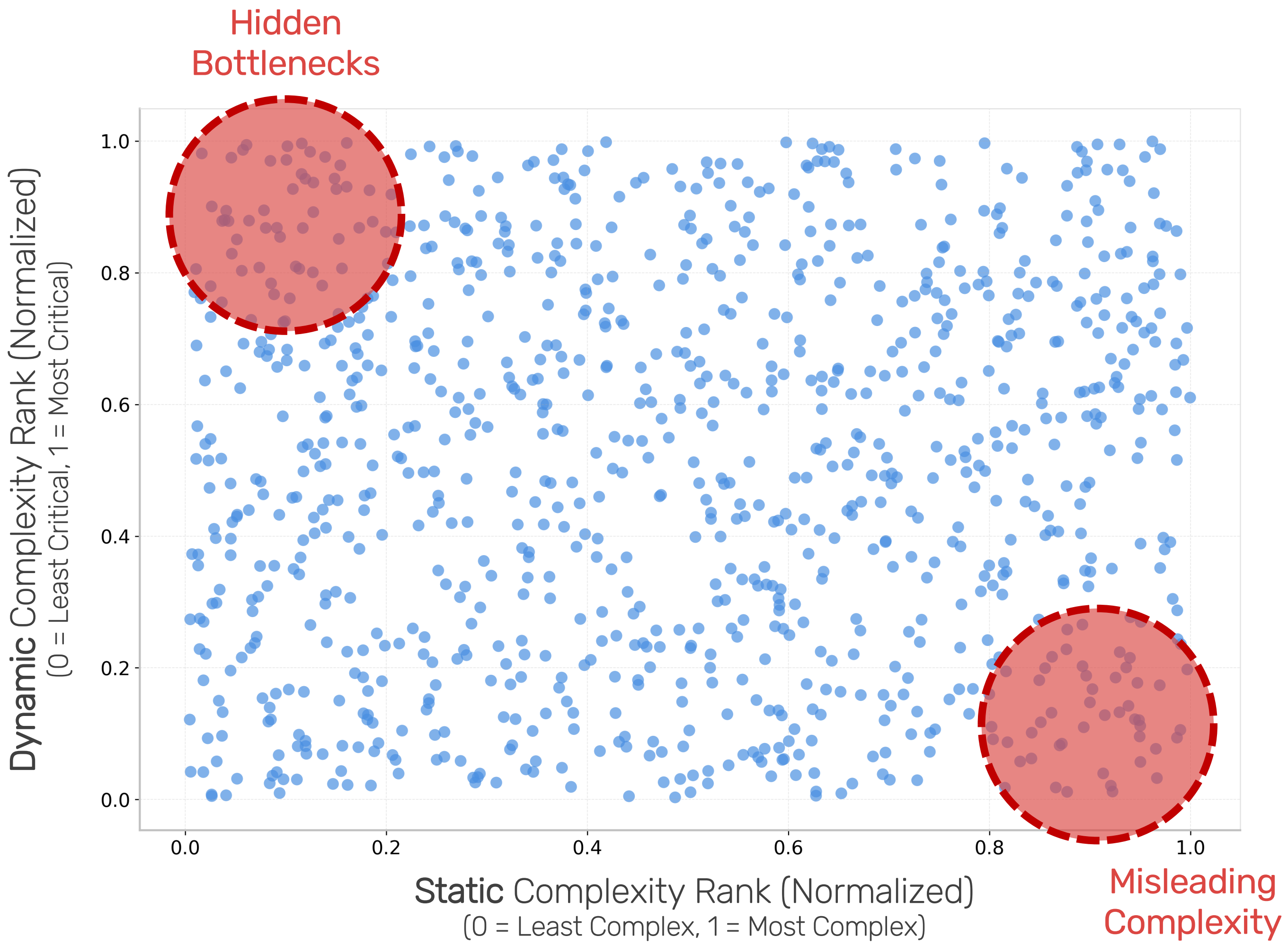
Spearman Correlations

SQLite 3: -0.041 (zero)

FFmpeg: 0.175 (Weak)

Zstandard: 0.336 (Weak)

OpenSSL: 0.543 (Moderate)



11.0%

Average variance explained by the static metrics

Spearman Correlations

SQLite 3: -0.041 (zero)

FFmpeg: 0.175 (Weak)

Zstandard: 0.336 (Weak)

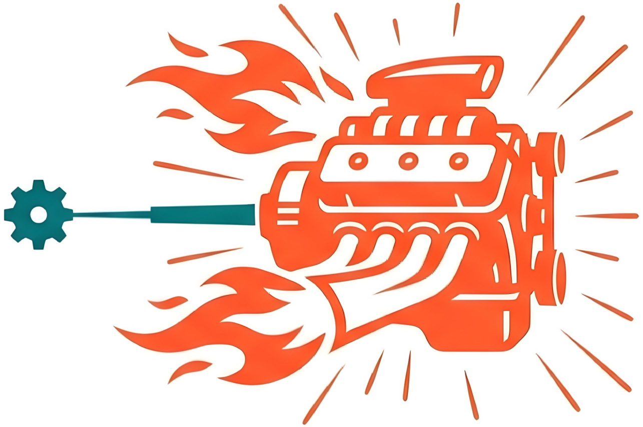
OpenSSL: 0.543 (Moderate)

The Paradox Functions

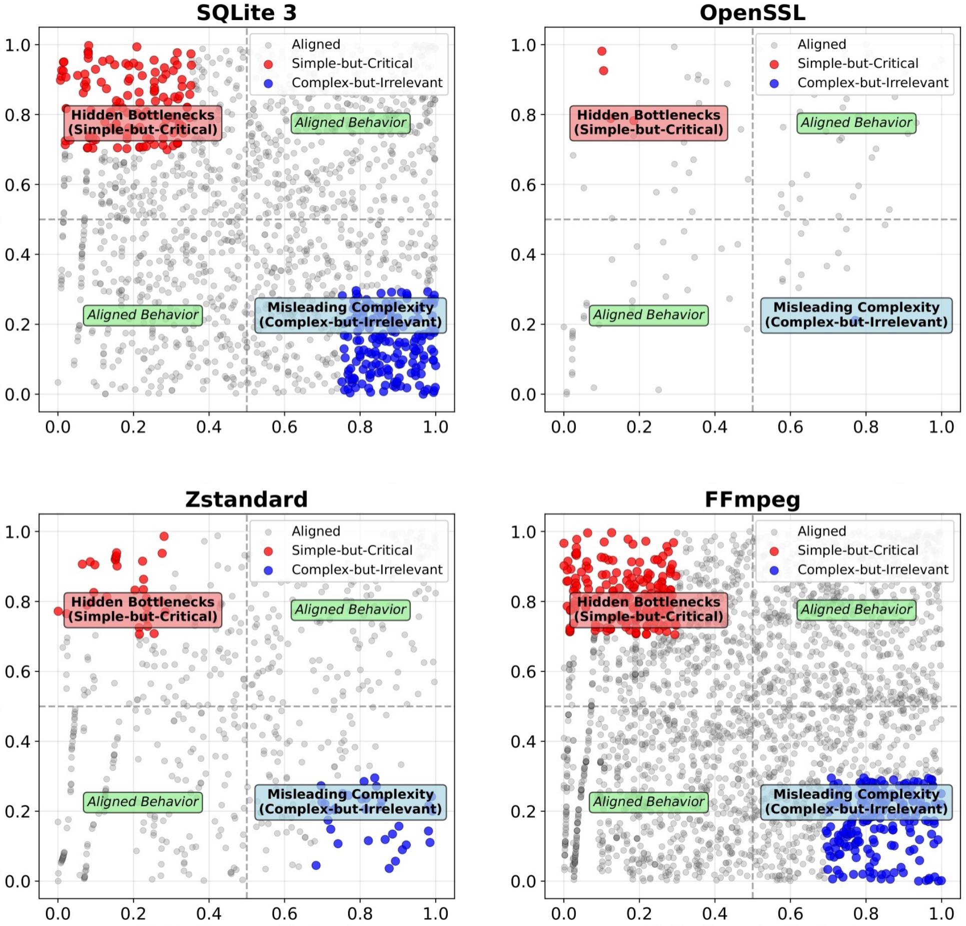
The Imposter

High Complexity, but Low Impact **(8.2%)**

The Assassin



Low Complexity, but High Impact **(7.3%)**



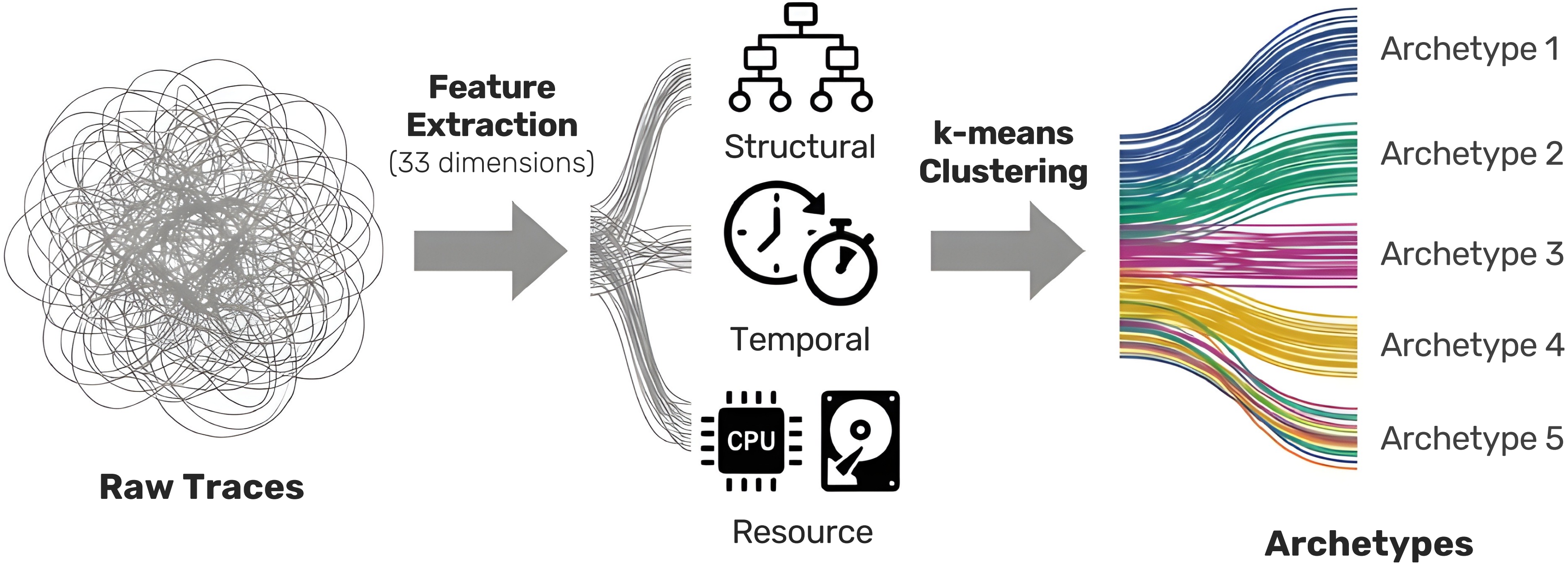
Dynamic Complexity Rank (Normalized)

(0 = Least Critical, 1 = Most Critical)

Static Complexity Rank (Normalized)

(0 = Least Complex, 1 = Most Complex)

RQ2: Universal Performance Archetypes?



Raw Traces

Feature Extraction
(33 dimensions)

Structural

Temporal

Resource

k-means Clustering

Archetype 1

Archetype 2

Archetype 3

Archetype 4

Archetype 5

Archetypes

10 Universal Performance Archetypes

A0
CPU Burst

18.4% of paths.
Found in 2/4 apps.

A1
Universal Transaction

23.2% of paths.
Found in 4/4 apps.

A2
Fast Initialization

6.3% of paths.
Found in 4/4 apps.

A3
Deep SQL Processing

4.7% of paths.
Found in 1/4 apps.

A4
Intensive Compression

12.0% of paths.
Found in 3/4 apps.

A5
I/O Sync

1.6% of paths.
Found in 3/4 apps.

A6
Crypto Primitives

8.8% of paths.
Found in 3/4 apps.

A7
Memory Management

7.7% of paths.
Found in 4/4 apps.

A8
Growing Mem. Pattern

2.2% of paths.
Found in 2/4 apps.

A9
Long Shallow Computation

15.2% of paths.
Found in 2/4 apps.

10 Universal Performance Archetypes

A0
CPU Burst

18.4% of paths.
Found in 2/4 apps.

A1
Universal Transaction

23.2% of paths.
Found in 4/4 apps.

A2
Fast Initialization

6.3% of paths.
Found in 4/4 apps.

A3
Deep SQL Processing

4.7% of paths.
Found in 1/4 apps.

A4
Intensive Compression

12.0% of paths.
Found in 3/4 apps.

A5
I/O Sync

1.6% of paths.
Found in 3/4 apps.

A6
Crypto Primitives

8.8% of paths.
Found in 3/4 apps.

A7
Memory Management

7.7% of paths.
Found in 4/4 apps.

A8
Growing Mem. Pattern

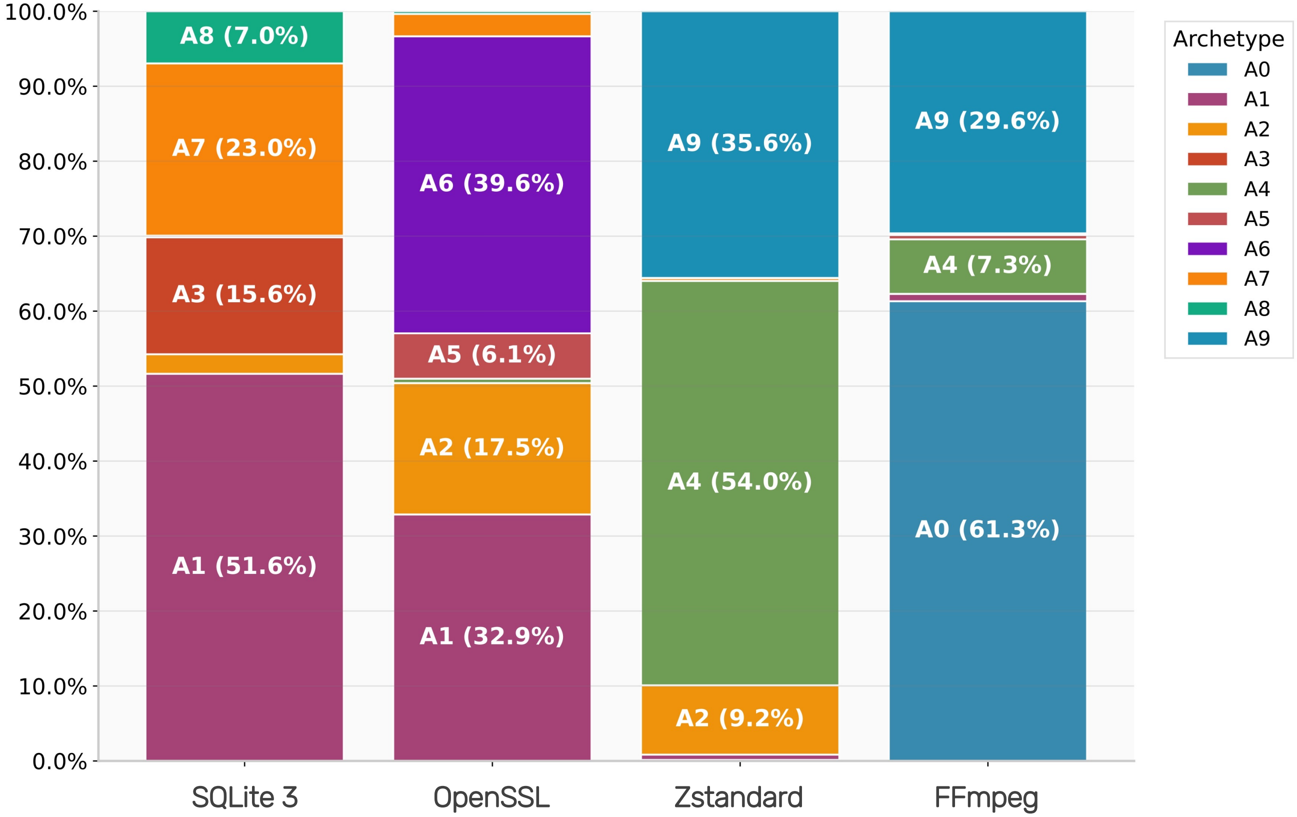
2.2% of paths.
Found in 2/4 apps.

A9
Long Shallow Computation

15.2% of paths.
Found in 2/4 apps.

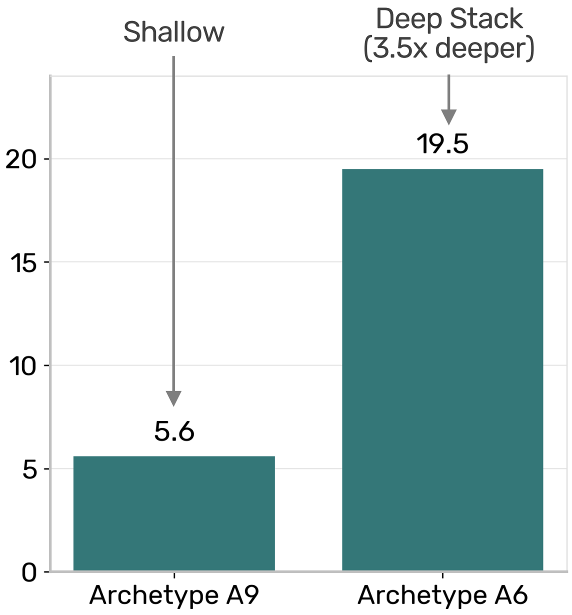
These 3 archetypes appear in every application studied.
They covered 37.2% of all critical paths.

Architectural DNA: **Application Signatures**



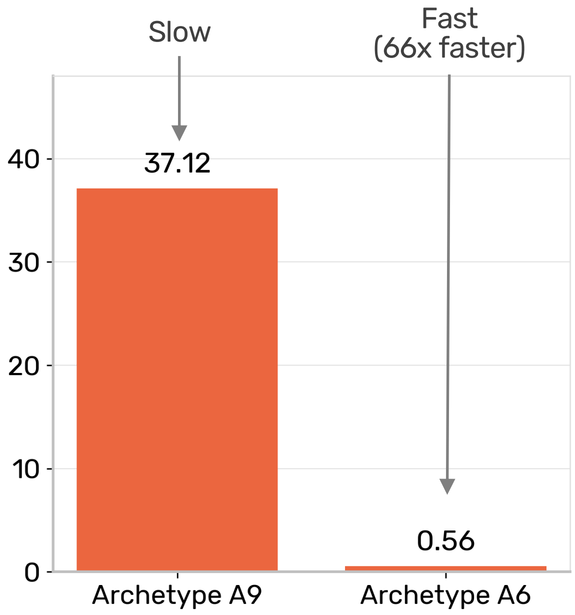
Myth-Busting: **Depth ≠ Duration**

Stack Depth



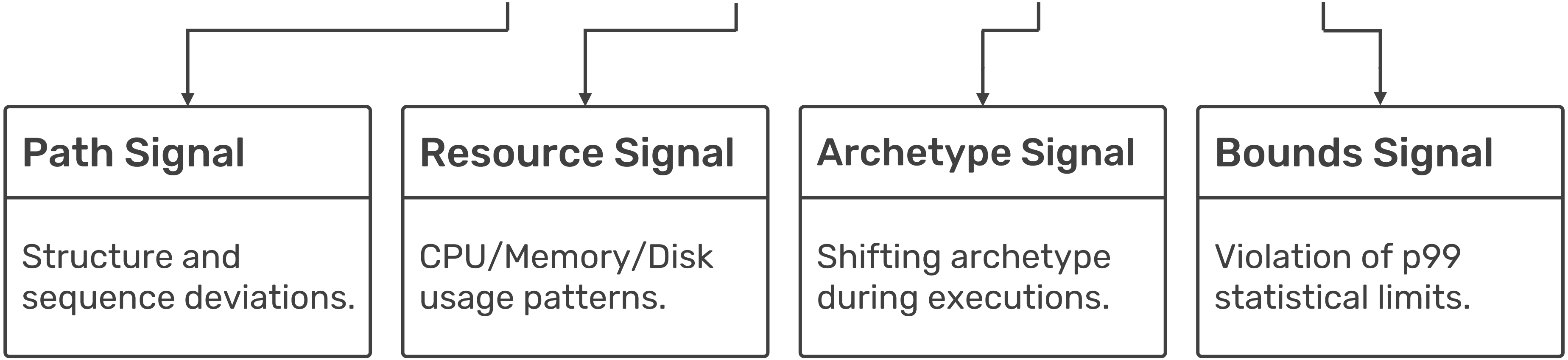
Performance Contradiction

Duration



RQ3: Regression Detection?

$$S_{anomaly} = S_{path} + S_{resource} + S_{archetype} + S_{bounds}$$



Path Signal

Structure and sequence deviations.

Resource Signal

CPU/Memory/Disk usage patterns.

Archetype Signal

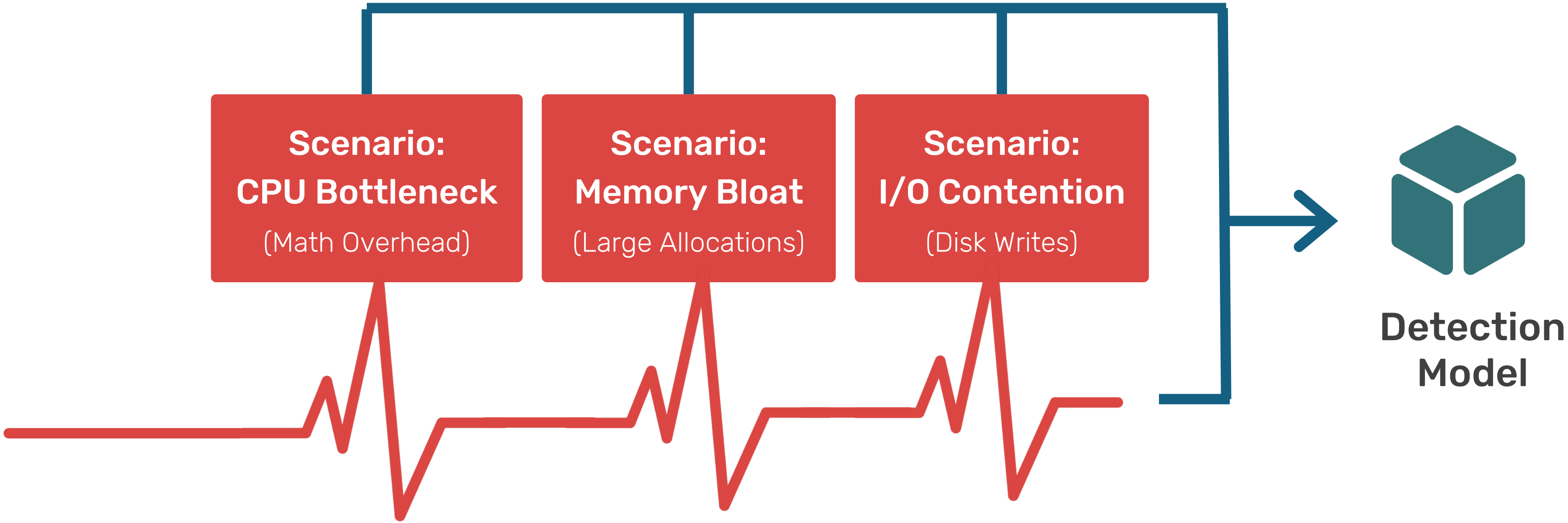
Shifting archetype during executions.

Bounds Signal

Violation of p99 statistical limits.

All signals have equal weights (i.e., 0.25), so the model ***avoids the false positives*** *of single-metric monitoring.*

Regression Injection & Detection



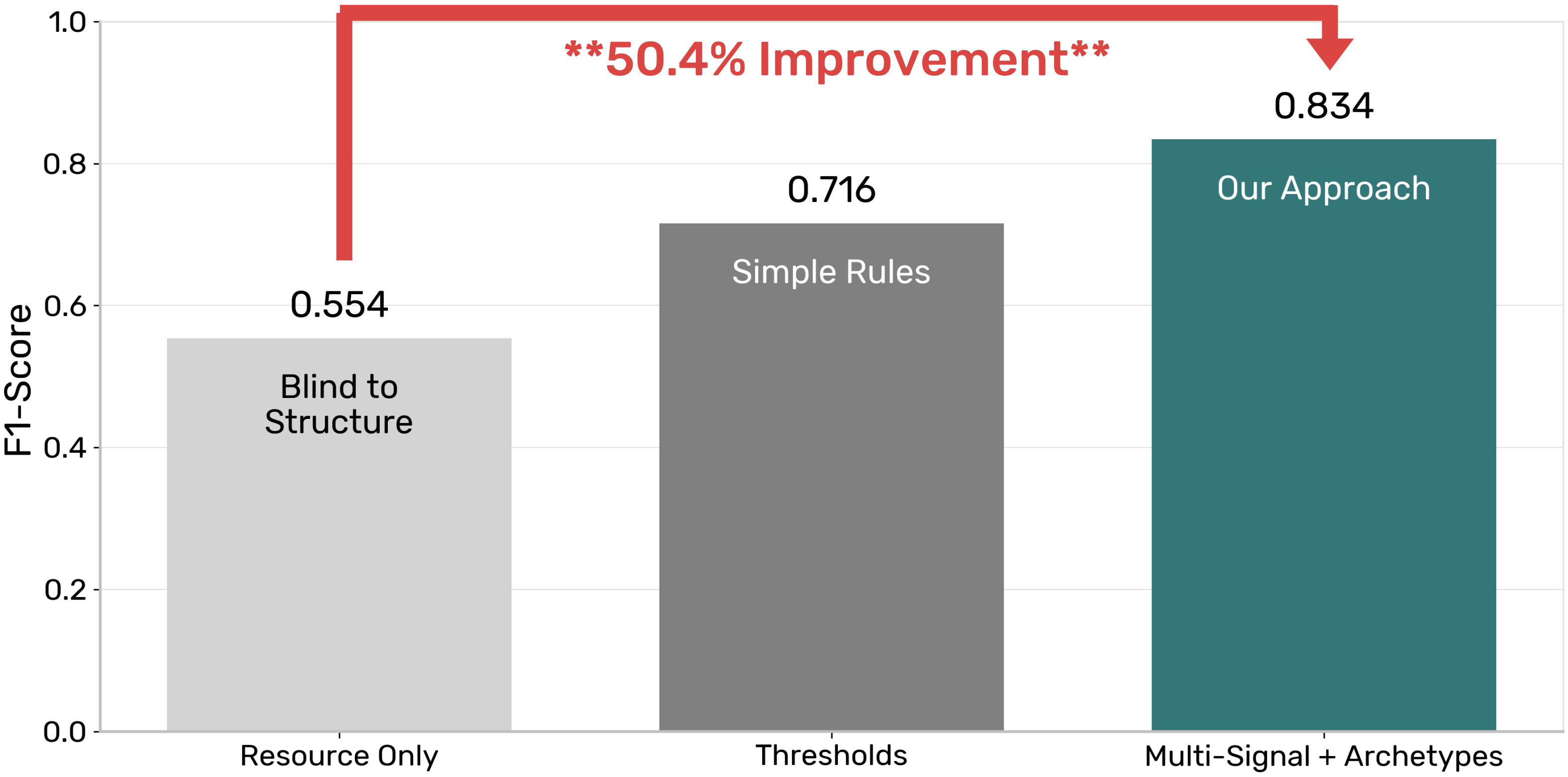
Scenario:
CPU Bottleneck
(Math Overhead)

Scenario:
Memory Bloat
(Large Allocations)

Scenario:
I/O Contention
(Disk Writes)

Detection Model

Context Matters!



Precision

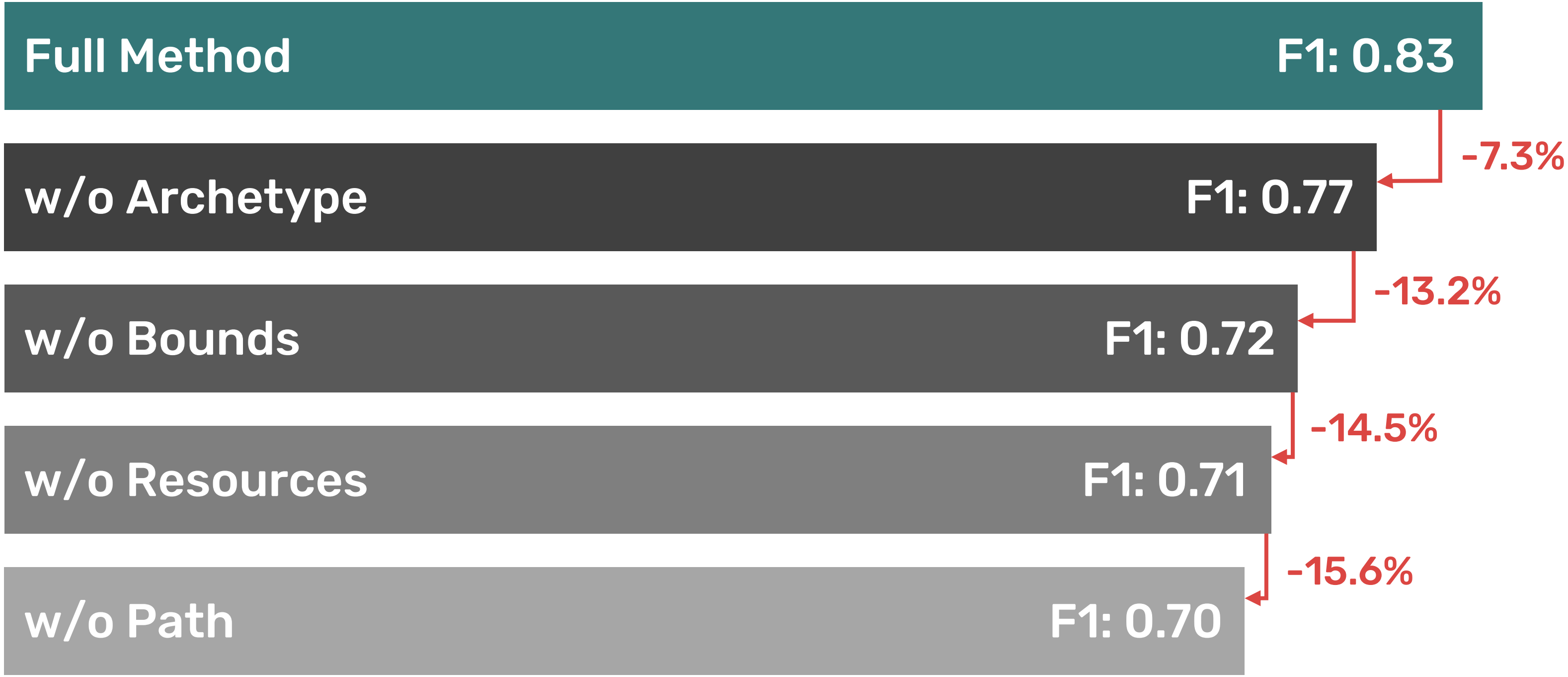
0.906

Recall

0.773

Every Signal Matters!

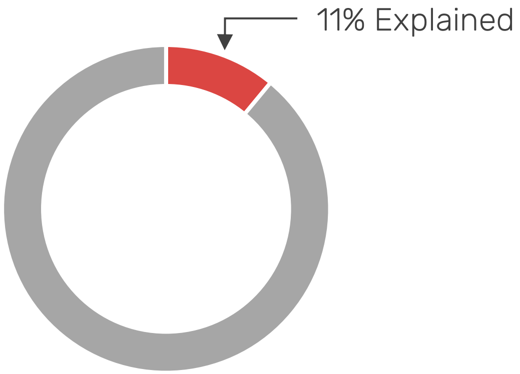
Ablation Study



Executive Summary

1. The Paradox

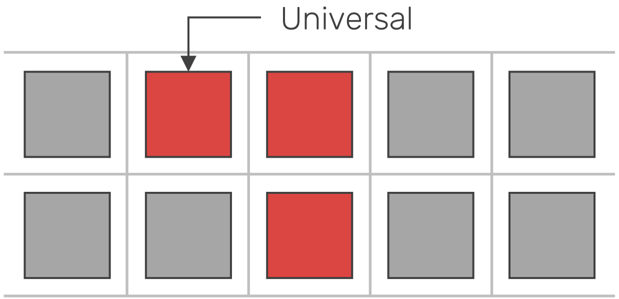
Traditional static metrics like LOC or Cyclomatic Complexity explain only 11.0% of runtime performance variance.



~16.0% of functions are **“Paradox Functions”,** which are either hidden bottlenecks or deceptive complexity

2. The DNAs

Critical paths are not random. They may cluster into **10 Universal Performance Archetypes** (A0-A9).



Patterns like **“Transaction”** and **“Memory”** appear in all tested apps, showing a shared architectural DNA.

3. The Application

Multi-signal regression detection outperforms resource-only monitoring.

0.554 ↦ 0.834

Resource-Only F1 Multi Signal F1

Driven by **context-awareness** of the **Critical Paths**.