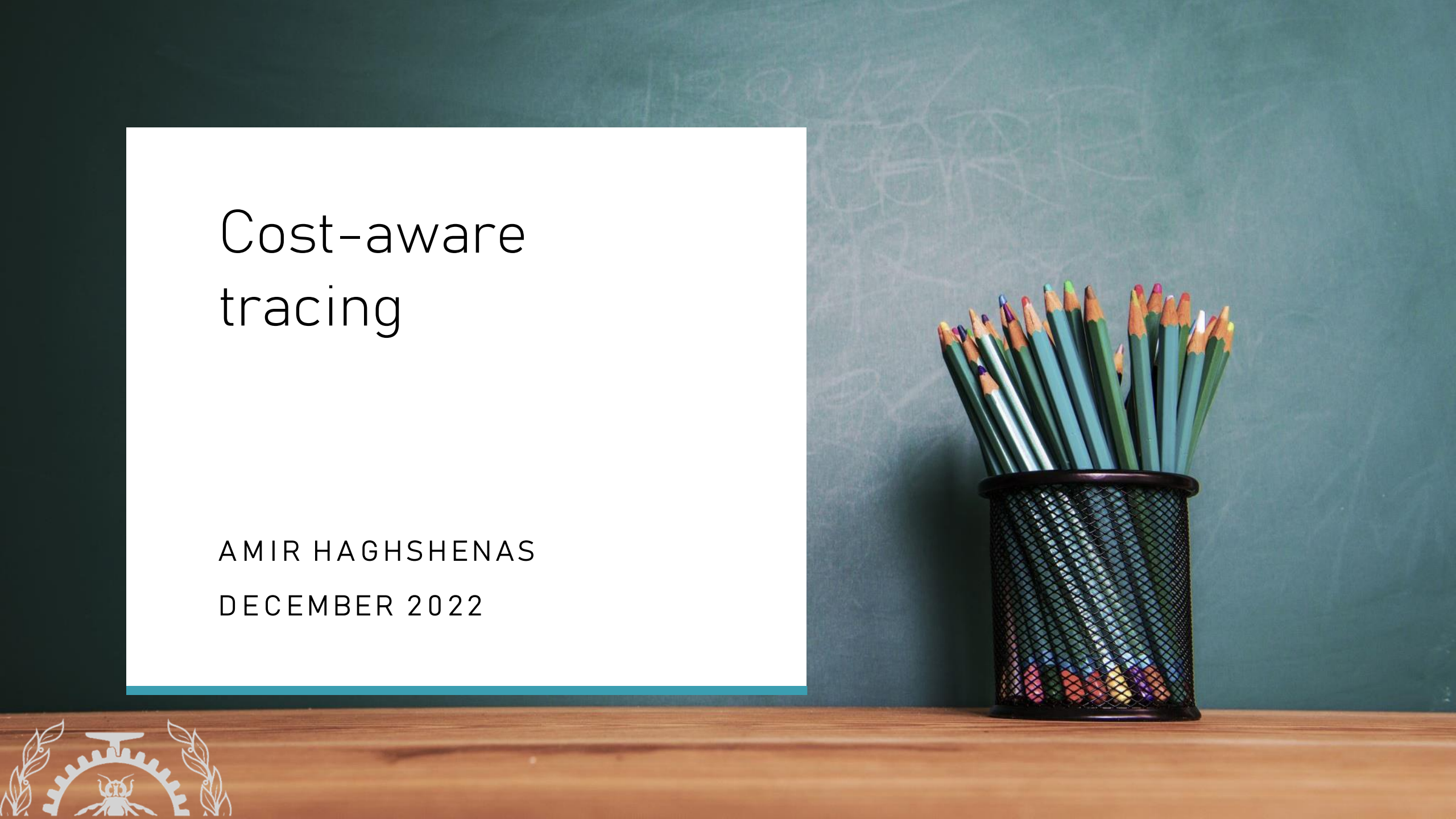
Cost-aware tracing

AMIR HAGHSHENAS

DECEMBER 2022



Content

- Tracing cost definition
- Cost functions
- Tracing objective
- Cost calculation and application call graph analysis
- Next steps
- challenges

Tracing cost definition

- Tracing generates large amount of data in short time
- It is not always possible to trace and save everything

  IoT devices

  Embedded systems
- A solution is required to choose how to collect as much as possible

  systems with limited resources

  Systems with large number of functions

Cost functions

Cost Function	Definition
Execution time overhead	CPU cycles added for collecting trace data
Concurrent uniformity	the effect of tracing on concurrent behavior of a system
Memory volume overhead	the amount of memory required to collect and save tracing data
Code size	The added code size due to injection of tracepoint data
Detection delay	The delay between when a variable is defined and when it is captured
Bandwidth overhead	The overhead of transmitting the trace data over the network to processing units

Tracing objective

- Performance monitoring
- Calculate cost of adding each tracepoint
- Extract the most performance influencing code sections (functions) for monitoring
- Maximize the number of functions to be traced, respect the available tracing budget

Tracing cost calculation

A micro benchmark was developed to calculate the execution time of calling a tracepoint

We run the benchmark in both kernel and user spaces.

We constructed a map of execution time for different tracepoint payloads.

The mapping will be used in predicting the tracing overhead.

Benchmark results

Tracepoint size	Time in user space (ns)	Time in kernel space(ns)
4 B	143.46	77.93
16 B	215.24	86.09
64 B	301.83	92.32
256 B	392.41	97.87
512 B	517.52	123.95
1 KB	895.18	177.65
2 KB	1468.36	276.45

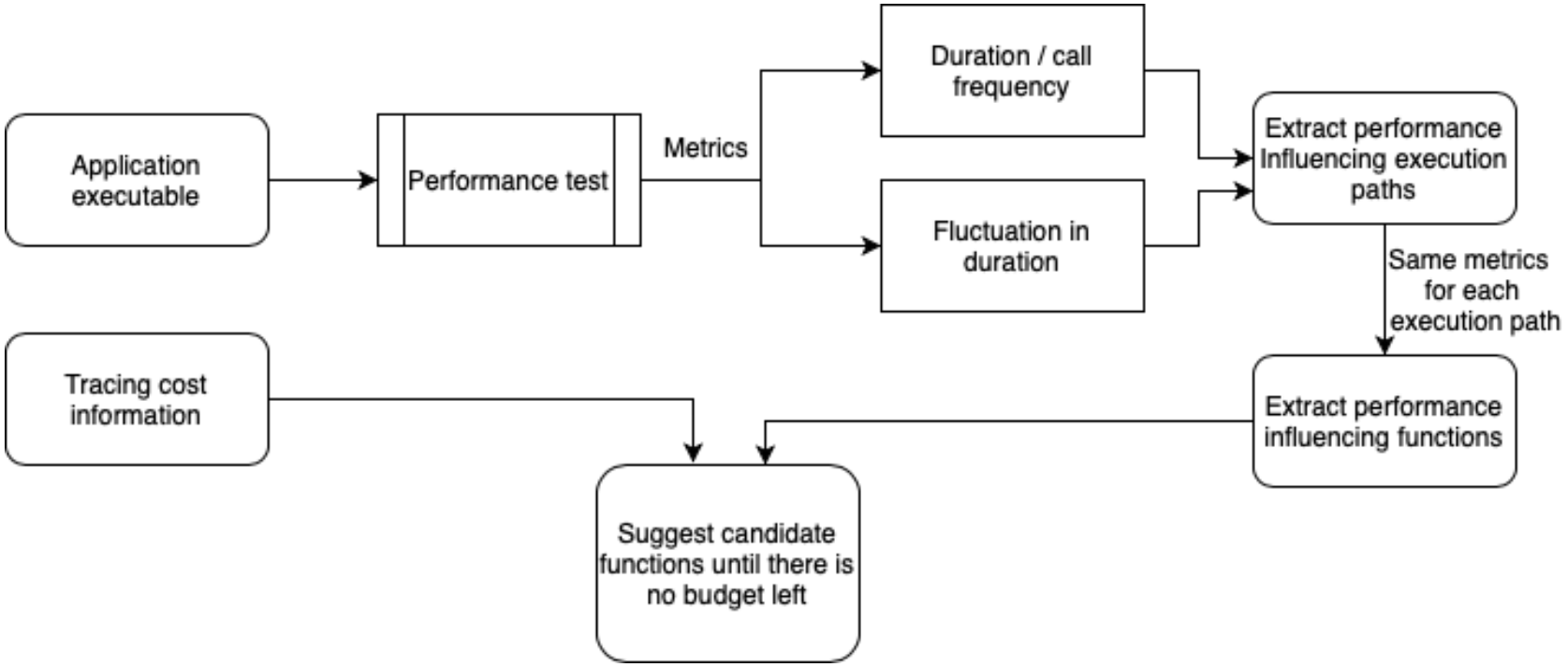
Application call graph analysis

- Extract control flow and profiling information using gprof profiler in testing environments
- Metrics for identifying performance influencing methods

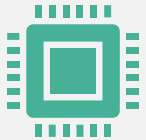
  Function duration divided by call frequency

  Fluctuation in function duration in different test scenarios
- Rank functions in each execution path based on our metrics
- Select functions for tracing until there is no budget available (tracing configuration)

Proposed framework

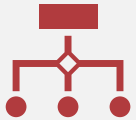


Next step: real-time adjustment



Monitor the application in production phase using existing phase detection algorithms.

Extract tracepoint metrics in addition to phase change detection



Update the tracing configuration and suggest new functions for tracing with respect to budget.

Challenges

- Using profilers on large applications take long to complete.

  We apply this method in testing phase

- Phase change detection algorithm should not introduce huge overhead to the application

  We will benchmark different existing algorithm to apply the best algorithm

Thank you

AMIR.HAGHSHENAS@POLYMTL.CA