IntroPerf: Transparent Context-Sensitive Multi-layer Performance Inference using System Stack Traces

Chung Hwan Kim, Junghwan Rhee*, Hui Zhang*, Nipun Arora*, Guofei Jiang*, Xiangyu Zhang, Dongyan Xu
Purdue University and CERIAS, NEC Laboratories America*

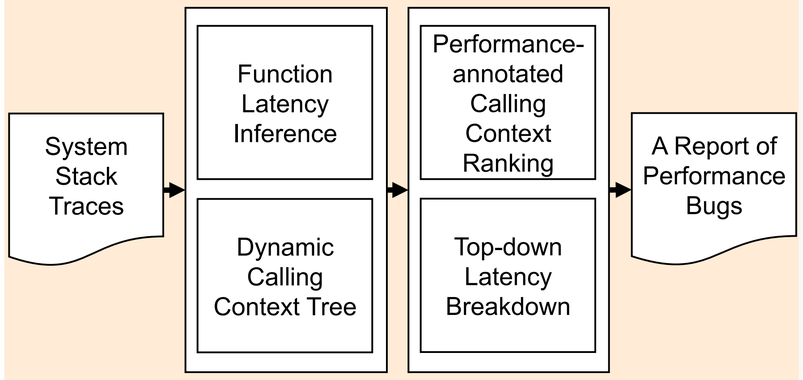
Motivation

- Performance bugs are frequently observed in commodity software.
- Performance bugs may escape the development stage, and incur problems in a post-development setting.
- Commodity software consists of many inter-dependent components across multiple system layers.
- Software is often deployed in a binary format which lacks source level semantics.

Approach

- Transparent performance diagnosis with low overhead in the post-development stage.
- All components in the vertical software layers are analyzed with a system-wide scope.
- OS tracers are commonly used in modern operating systems for troubleshooting and advanced OS tracers provide system-wide stack traces.
- **IntroPerf** infers context-sensitive application performance and analyzes performance bugs by leveraging stack traces from OS tracers.

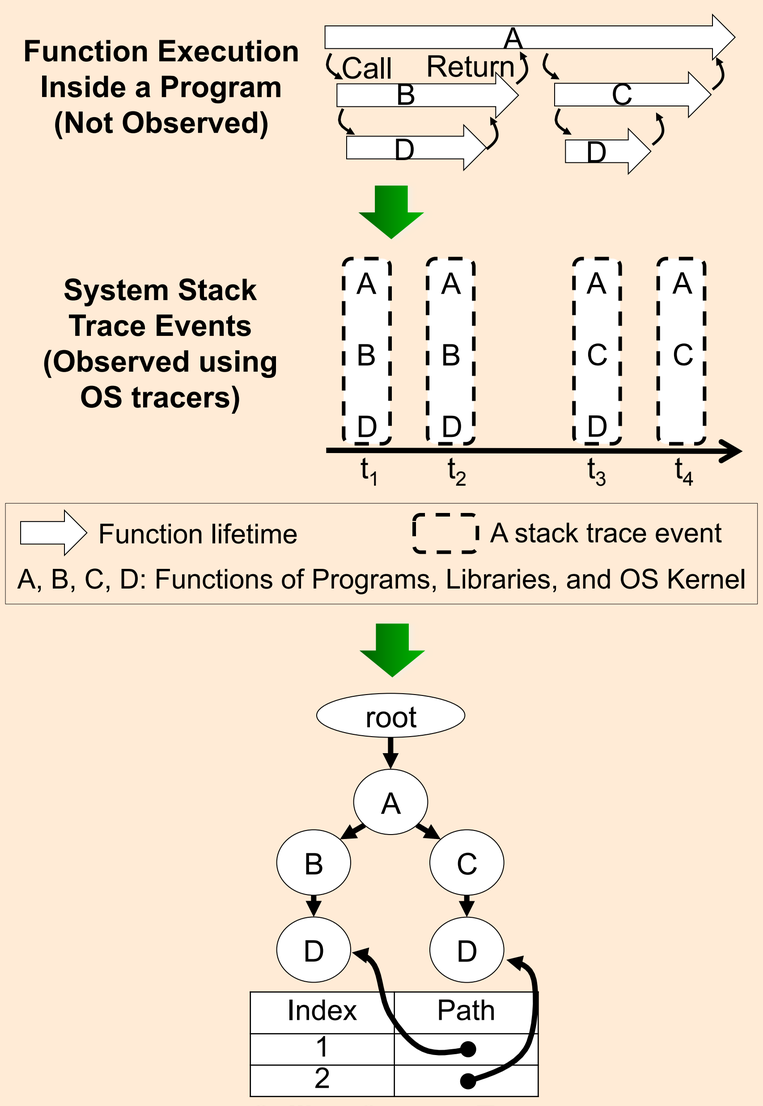
IntroPerf Architecture



- IntroPerf converts system stack traces to a set of function latencies.
- Performance bug candidate functions are ranked regarding dynamic calling contexts.

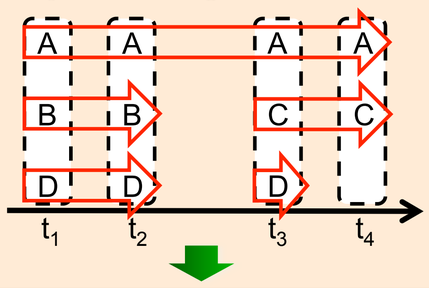
System Stack Traces & Calling Context Tree

- A sequence of system stack traces is converted to a calling context tree.
- Each call path is indexed using the leaf node for quick retrieval and computation.

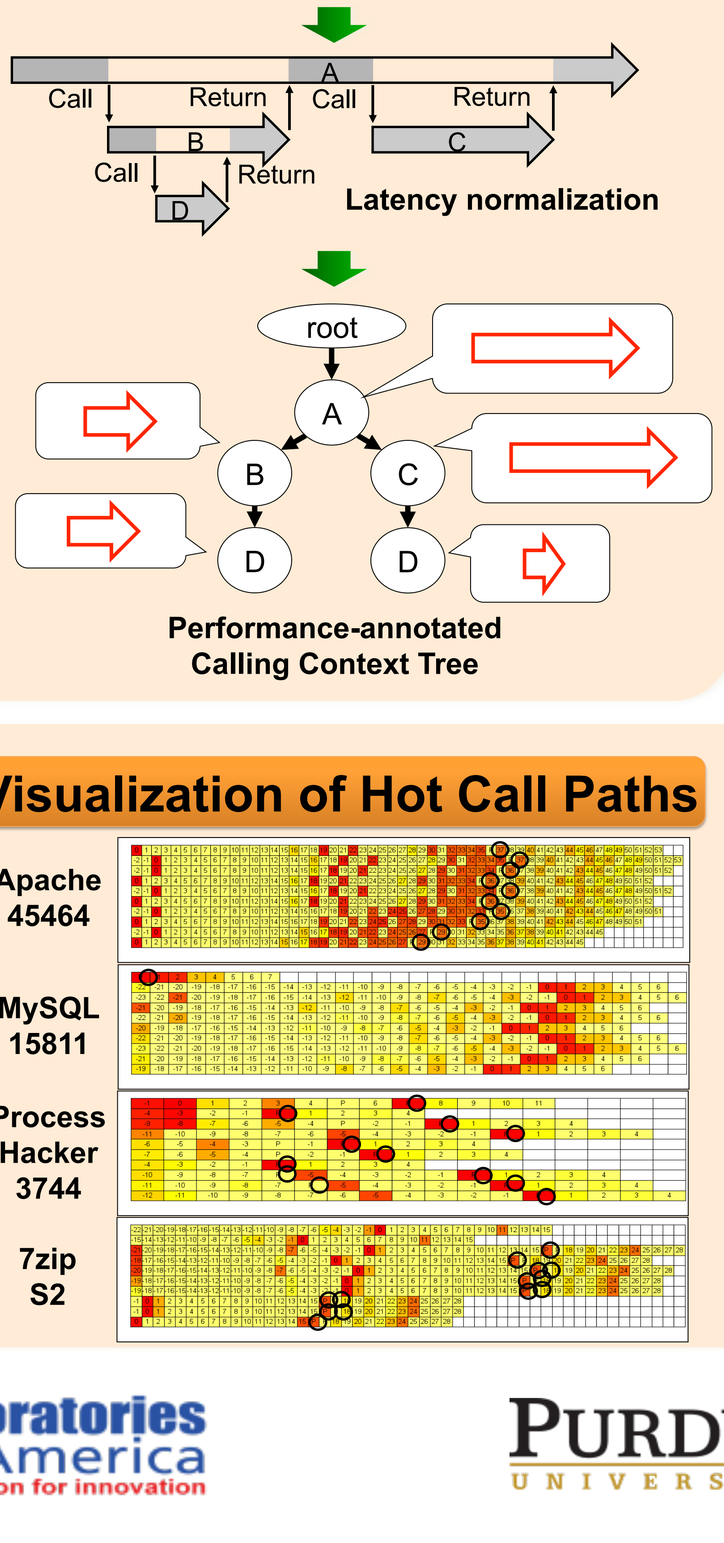


Function Latency Inference & Performance-annotated CCT

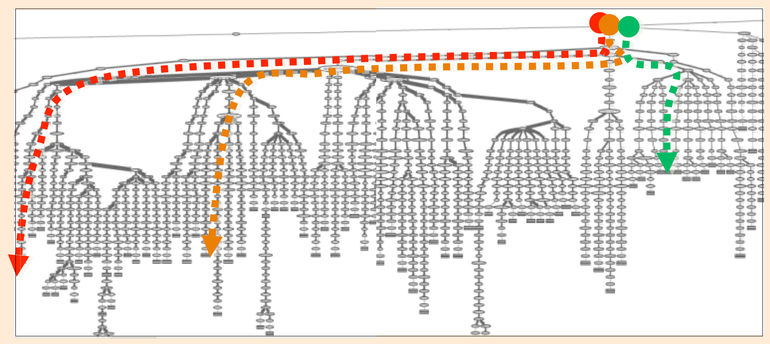
- Function latencies are inferred based on **the continuity of calling context**.



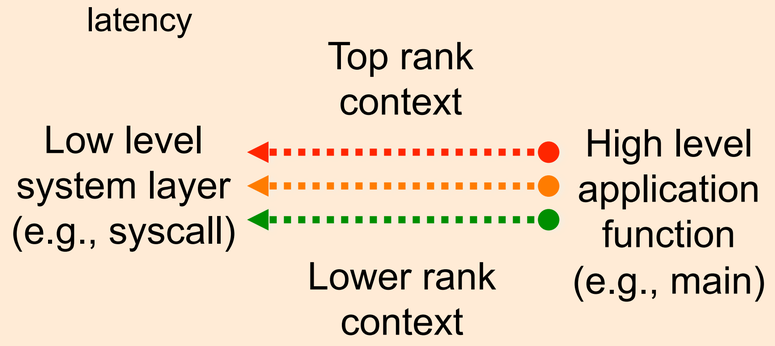
Function instance	Calling context	Inferred latency
A	**A**	
B	A→**B**	
C	A→**C**	
D	A→B→**D**	
D	A→C→**D**	



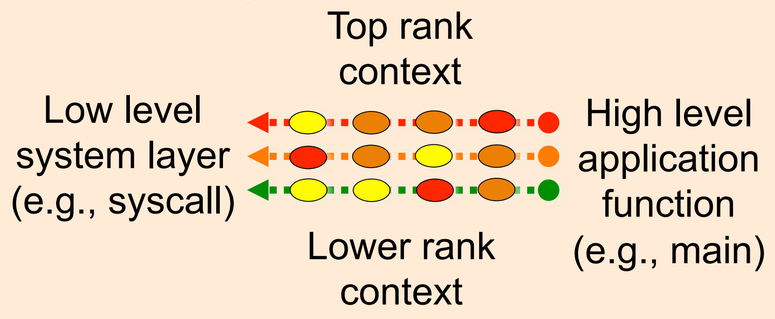
Context-sensitive Performance Analysis



- Ranking of dynamic calling context with latency



- Ranking functions within calling context with latency



Performance Bug Detection

Program	Bug ID	p_{min}	f_{min}	Root Cause Function
Apache	45464	**1**	**36**	libapr-1.dll!apr_stat
MySQL	15811	**1**	**0**	mysql.exe!strlen
MySQL	49491	**3**	**5**	mysqld.exe! Item_func_sha::val_str
Process Hacker	3744	**1**	**0**	ProcessHacker.exe! PhSearchMemoryString
7zip	S1	**11**	**16**	7zFM.exe! CPanel::RefreshListCtrl
7zip	S2	**3**	**16**	7zFM.exe! CPanel::RefreshListCtrl
…	…	…	…	…

Visualization of Hot Call Paths

Coverage of Program States

- The experiments with Apache, MySQL, 7zip show that stack traces generally cover 5.3~49.4% of dynamic calling contexts and 0.6~31.2% of function instances
- However, the coverage of calling contexts and instances for top 1% slowest functions are respectively 34.7~100% and 16.6~100% depending on applications.
- IntroPerf focuses on the functions with large latencies for performance diagnosis.

NEC Laboratories America
Relentless passion for innovation

PURDUE UNIVERSITY

Author Paper