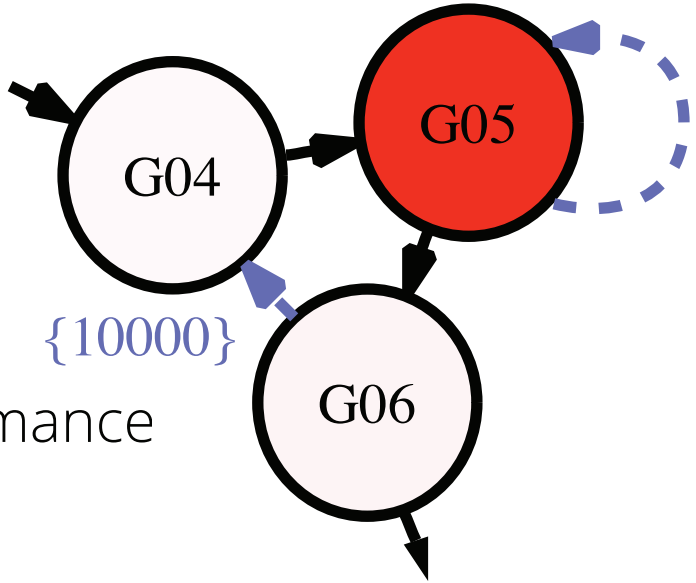
SaSe™
Business Solutions

EPSRC

UNIVERSITY OF CAMBRIDGE

THE UNIVERSITY OF WAIKATO
Te Whare Wānanga o Waikato

Imperial College London



OpenPAT | The Open Performance Analysis Toolkit

Analysing Programs the Easy Way

Dr Simon Spacey

B.Sc. Hons. (York), M.Sc. (Lancaster), M.B.A. (Cass), J.L.P. (Keio),
D.E.A. (Montpellier), D.CSC. (Cambridge), D.I.U. (Imperial), Ph.D. (Imperial)

What is OpenPAT?

- A Toolkit to Analyse Programs to Identify:
  - Hotspot and Performance Bottlenecks
  - Dynamic Execution Trace Information
  - Code Structure
- How can this Help Me?
  - Increase Performance on Existing Hardware
  - Quantify Performance Potentials for New Approaches
  - Understand and Verify Large Programs

OpenPAT Advantage

Approach	SUIF [3]	GILK [4]	Valgrind [5]	Pin [6]	OpenPAT [1,2]
Instrumentation	Static	Dynamic	Dynamic	Dynamic	Static
Analysis	Static	Dynamic	Dynamic	Dynamic	Dynamic
Target	Source	Binary	Binary	Binary	Assembly

The OpenPAT Approach Compared with other Toolkits

Contribution	SUIF [3]	GILK [4]	Valgrind [5]	Pin [6]	OpenPAT [1,2]
Efficient	✔	✗	✗	✗	✔
Accurate	✗	✔	✔	✔	✔
Simple	✗	✗	✗	✗	✔

The Contribution for Users of Each Approach Dimension

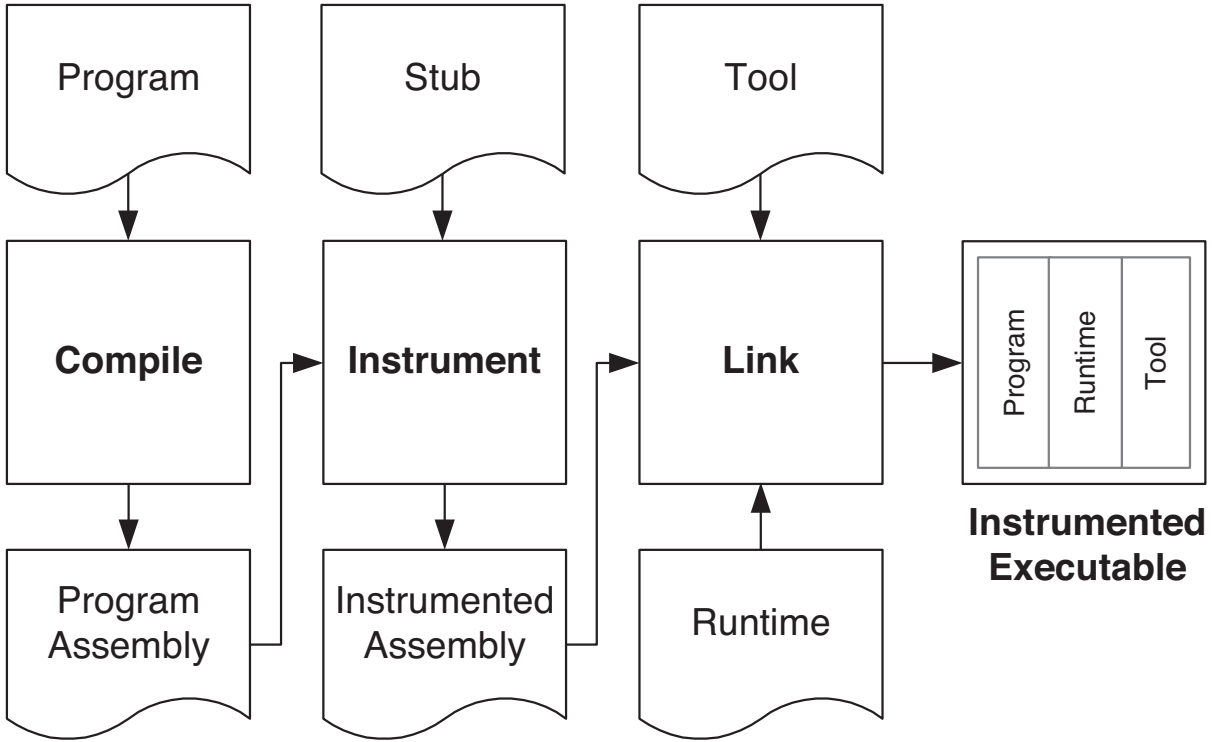
[1] S. Spacey, 3S: Program Characterisation, Imperial Tech. Paper (2006).
[2] S. Spacey, 3S Quick Start Guide, Imperial Tech. Manual (2009).
[3] G. Aigner et al., An Overview of the SUIF2 Compiler Infrastructure (2000).
[4] D.J. Pearce et al., GILK: A Dynamic Instrumentation Tool (2002).
[5] N. Nethercote et al., Valgrind: A Program Supervision Framework (2003).
[6] C. Luk et al., Pin: Building Customized Program Analysis Tools with Dyn. Instr. (2005).

OpenPAT Key Features

Feature	v0	v1	v2	v3
	3S (04/06 – 05/06)			OpenPAT
Block Traces	✔	✔	✔	✔
Block Tools		✔	✔	✔
CPU Timestamp Timings		✔	✔	✔
Instruction Level DFG Tools (Integer)			✔	✔
Instruction Level DFG Tools (x87 FPU)				✔
Static Memory Instrumentation				✔
Assembly Aspect-Oriented Programming				✔
Ranged Instrumentations				✔
Granularity and Threading Features				✔
Retargetable (CPU, OS, Language, Compiler)				✔
Community Tool Support & Site				✔
Main Toolkit Languages		Python/C		Perl/C

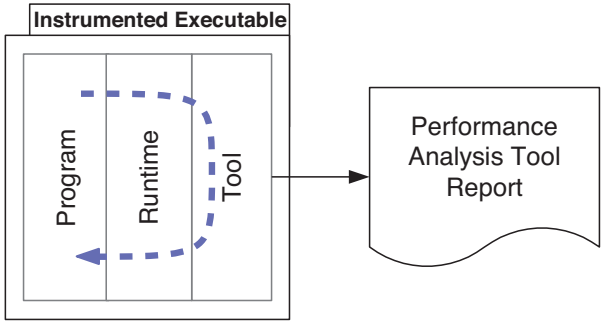
OpenPAT Version History and Key Features

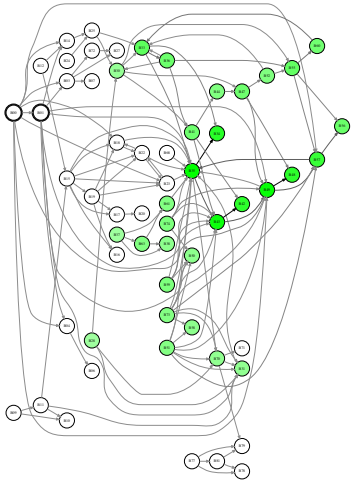
OpenPAT Approach

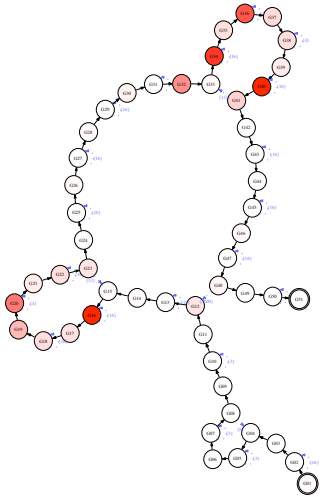


The OpenPAT Static Instrumentation and Dynamic Analysis Approach

OpenPAT Tools







```
#### OPEN PERFORMANCE ANALYSIS TOOLKIT
(OpenPAT) REGEX 2.9.1 REPORT FOR RUN
STARTED Thu Jan 24 00:16:58 2013
B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13
B14 (B15 (B16 B17 B18 B19 B20 B21 B22 B23
B24 (B25){12} B26 B27){100} B28){24} B15
(B16 B17 B18 B19 B20 B21 B22 B23 B24 (B25)
{12} B26 B27){90} B16 B17 B18 B19 B20 B21
B22 B23 B24 (B25){11} (B26 B27 B16 B17 B18
B19 B20 B21 B22 B23 B24 (B25){12} (B26 B27
B28 B15 (B16 B17 B18 B19 B20 B21 B22 B23
B24 (B25){12} B26 B27){100} B28){75} ){9}
B29 B30

* The above regular expression is written
in compressed x format with {}
representing loop counts.

#### SYMBOL TABLE
B1,main.0.0.0,15
B2,main.0.1.0,6
B3,main.0.1.call sscanf,1
B4,main.0.2.0,2
B5,main.L4.0.0,3
B6,main.L4.0.call printf,1
B7,main.L4.1.0,6
B8,main.L4.1.call calloc,1
B9,main.L4.2.0,3
B10,main.L4.2.call calloc,1
B11,main.L4.3.0,3
...
B22,main.L41.0.0,8
B23,main.L40.0.0,10
B24,main.L40.1.0,0
B25,main.L8.0.0,58
B26,main.L8.1.0,0
B27,main.L39.0.0,5
B28,main.L39.1.0,4
B29,main.L39.2.0,0
B30,main.L5.0.0,1
B31,main.L5.0.call exit,1
```

Example Tools
trace
hotspot
memory
dfg
cfg
ilp
regex
loopgraph_d
profile
cache_flow
override
simulator

Selected Tools

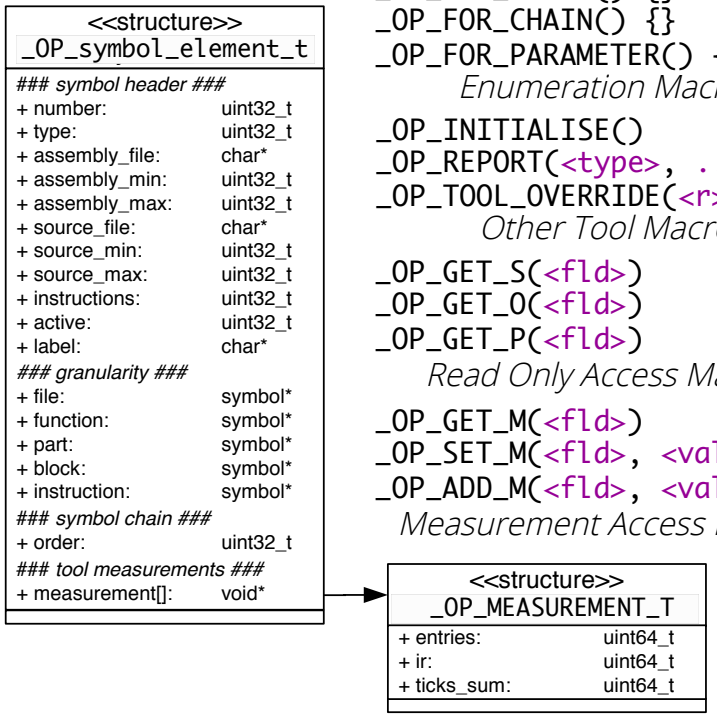
OpenPAT Tool Infrastructure

```
#define _OP_STUB        symbols_ticks
#include "../../../../_/runtime/OpenPAT.h"

typedef struct {} _OP_MEASUREMENT_T;

/* Called once before the main program starts */
void _OP_TOOL_INITIALISE(void) {}

/* Called with _OP set as code sections run */
void _OP_TOOL_ANALYSE(_OP_t *_OP) {}

/* Called once after main program exits */
void _OP_TOOL_REPORT(void) {}
```

A Runnable OpenPAT Tool

```
typedef struct {
  _OP_thread_id_t          thread_id;
  _OP_symbol_element_t*    last_symbol;
  _OP_symbol_element_t*    next_symbol;
  _OP_ticks_t              last_start;
  _OP_ticks_t              last_end;
  _OP_parameter_t          last_parameters[];
} _OP_t;
```

Analysis Information Supplied by Runtime/Stubs



OpenPAT Symbol Table and Measurement Structures

```
_OP_FOR_TABLE() {}
_OP_FOR_CHAIN() {}
_OP_FOR_PARAMETER() {}
```

Enumeration Macros

```
_OP_INITIALISE()
_OP_REPORT(<type>, ...)
_OP_TOOL_OVERRIDE(<r>, <fn>)
```

Other Tool Macros

```
_OP_GET_S(<fld>)
_OP_GET_O(<fld>)
_OP_GET_P(<fld>)
```

Read Only Access Macros

```
_OP_GET_M(<fld>)
_OP_SET_M(<fld>, <val>)
_OP_ADD_M(<fld>, <val>)
```

Measurement Access Macros

OpenPAT Summary

- The Open Performance Analysis Toolkit (OpenPAT)
  - Provides: Hotspot, Trace and Structural Information
  - Useful For: Performance Optimisation, Solution Evaluation and QA
- The OpenPAT Approach is:
  - Efficient: Statically Instruments Programs
  - Accurate: Dynamically Analyses Exact Program Execution Traces
  - Simple: Leaves (De-)Compilation to (De-)Compilers
- OpenPAT has Lots of Unique Features and Tools

More Information

[1] S. Spacey, 3S: Program Characterisation Framework, Imperial Tech. Paper (2006).

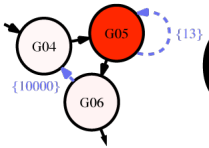
[2] S. Spacey, 3S Quick Start Guide, Imperial Tech. Manual (2009).

[3] G. Aigner, A. Diwan, D. Heine, M. Lam, D. Moore, B. Murphy, C. Sapuntzakis, An Overview of the SUIF2 Compiler Infrastructure. Stanford Tech. Paper (2000).

[4] D.J. Pearce, P.H.J. Kelly, T. Field, U. Harder, GILK: A Dynamic Instrumentation Tool for the Linux Kernel. Proceedings of the 12th International Conference on Computer Performance Evaluation, Modelling Techniques and Tools 37, pp. 220–226 (2002).

[5] N. Nethercote, J. Seward, Valgrind: A Program Supervision Framework. Electronic Notes in Theoretical Computer Science, 89(2) pp. 44-66 (2003).

[6] C-K. Luk, R. Cohn, R. Muth, H. Patil, A. Klauser, G. Lowney, S. Wallace, V.J. Reddi, K. Hazelwood, Pin: Building Customized Program Analysis Tools with Dynamic Instrumentation. Proceedings of the 2005 ACM SIGPLAN Conference on Programming Language Design and Implementation, pp. 190-200 (2005).



OpenPAT.org