

A09

Visual Performance Analyzer

Luc Smolders

**IBM SYSTEM p, AIX 5L
and LINUX TECHNICAL
UNIVERSITY
Sept 11 - 15, 2006**

Las Vegas, NV

charts to be available at:

<http://www.ibm.com/developerworks/blogs/page/Systemptechuniv>

Agenda

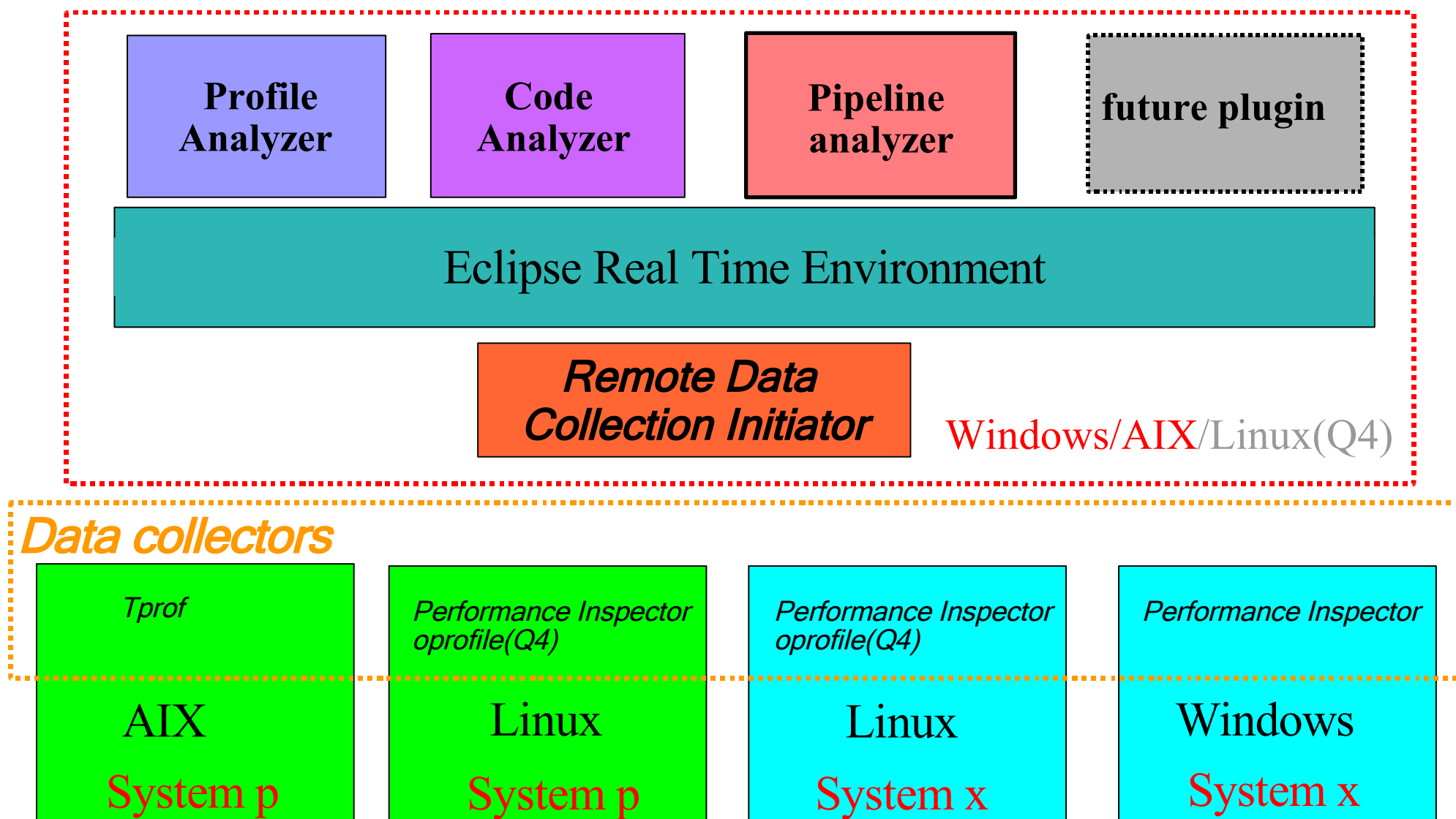
- Visual Performance analyzer
 - goals and components
- Profile Analyzer overview
 - components, selected features and sample views
- Code Analyzer overview
 - motivations, features and sample views
- Pipeline Analyzer overview
 - features and sample views
- Code availability
 - Alphaworks and Source Forge
- Future plans
- Recorded demo

Visual Performance Analyzer - goals

- Provide platform independent easy to use integrated set of graphical application performance analysis tools
- Target audience includes
 - performance analysts
 - application developers
 - compiler developers
- Leverage existing platform specific non-GUI performance analysis tools to provide comprehensive set of features
 - Performance Inspector on Windows and Linux
 - tprof on AIX
 - oProfile on Linux
 - FDPR on Linux and AIX
- Create consistent set of integrated tools to provide platform independent drill down performance analysis experience

Visual Performance Analyzer - components

- The Visual Performance Analyzer is an Eclipse based tool set, initially including three plug-in applications working collaboratively: Profile Analyzer, Code Analyzer and Pipeline Analyzer



Visual Performance Analyzer - current look

The screenshot displays the Visual Performance Analyzer (VPA) interface. The main window shows a hierarchical tree of process threads and modules. A yellow callout bubble points to the 'Tools' menu, which contains three items: Code Analyzer, Profile Analyzer, and Pipeline Analyzer. Another yellow callout bubble points to the 'Custom appearance' option in the 'Tools' menu. A third yellow callout bubble points to the 'Samples Distribut...' window, which shows a bar chart of samples distribution for various modules. The chart shows the following data:

Module	Percentage
.mmipExecuteJava	17.64%
.get4bytes	11.76%
.alloc	5.88%
.boo	5.88%
.creat	5.88%
.ens	5.88%
.findUTF8Entr	5.88%
Other	41.17%

The main window also shows a table of ticks and symbols for the selected module:

Ticks	%	Symbol
3	17.65	.mmipExecuteJava
2	11.76	.get4bytes
1	5.88	.utf2Unicode
1	5.88	.quickFieldAccess
1	5.88	.loadConstantPool
1	5.88	.lkMonitorExit
1	5.88	.hashUTF8
1	5.88	.getUTF8String
1	5.88	.get2bytes
1	5.88	.findUTF8Entry
1	5.88	.ensureLoaded
1	5.88	.createJavaFrame
1	5.88	.bootstrapSystemClasses
1	5.88	.allocMiddlewareContextObject

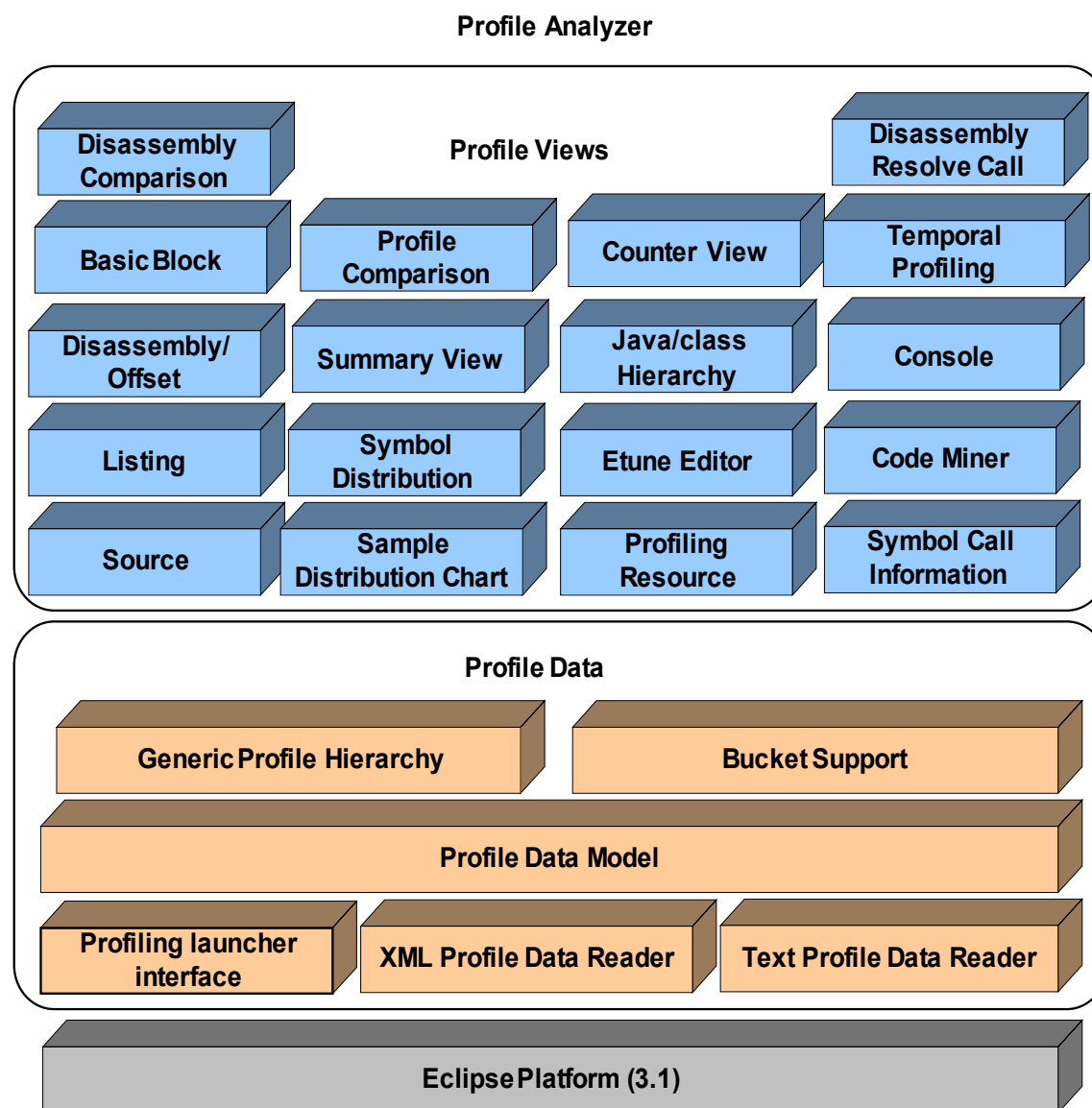
The bottom window shows the disassembly/offsets for the selected module:

Offset	Bytes	Disassembl
0x00000000	7c0802a6	mfspr r0,L
0x00000004	7d800026	mfcrr r12
0x00000008	48006a15	bl 0x6a1c
0x0000000c	39000000	li r8,+0
0x00000010	80c2004c	lwz r6,+76(r
0x00000014	3ca04330	lis r5,0x4330
0x00000018	bda1ff84	stmw r13,-124(r1
0x0000001c	607e0000	ori r30,r3,0x0

1. Windows, AIX, Linux(Q4)
2. DB2 and HSQLDB support
3. Java profiling via JVMPI
4. Single tool framework
5. Common data model
6. Tool integration
7. RSE for remote execution of test case and collection of data

Profile Analyzer - components

- Uses data from numerous platform specific profiling tools, including:
 - AIX tprof XML output data
 - Linux and Windows profiles produced by Performance Inspector's tprof
 - IBM JRE Java profile data
 - Linux profiles produced by oprofile via XML output option
 - ▶ coming in the next version(Q4)
 - profiles from additional IBM systems platforms
 - ▶ internal version only for now
- Provides many analysis views and features to help identify CPU performance bottlenecks



Profile Analyzer - selected features

- Customizable profile hierarchies
 - by default uses generic hierarchies
 - includes JAVA class hierarchy view
 - allows user-defined hierarchical views
 - ▶ includes bucketized view
- Built-in multi-platform disassembler
 - supports PowerPC, IA32 and AMD64
 - provides annotated instruction view
- Source annotation
 - synchronized with disassembly view
- Basic blocks view
 - automatically splits instructions into basic blocks and annotate them with profiling data
- Listing files annotation
 - limited to .lst files for now
- Temporal profiles
 - shows evolution of samples distribution over time
 - works on process, threads, objects/modules and on symbols

Profile Analyzer - selected features(*cont*)

- Profile comparison and merging
 - includes normalization
 - ▶ on transaction rate
 - ▶ cpu busyness
 - ▶ number of cpus
- Remote capture of profiles
 - uses RSE plugin to
 - ▶ start remote test execution
 - ▶ capture profile data
- Hardware events support
- Code Miner
 - wizard populates SQL database with profile information
 - ▶ supports DB2 and HSQLDB
 - queries allow detection of performance patterns hard to find with traditional profilers, e.g.
 - ▶ find hottest pairs of sequential instructions
 - ▶ find all symbols that contain a particular instruction sequence
 - ▶ all symbols that are hotter than a certain threshold and that have a certain pattern in their name
 - ideal for analysis of flat profile
 - ▶ where no single symbol uses more than a fractional percentage of total ticks

Profile Analyzer - custom hierarchy and Java support

Custom profile hierarchy with JAVA bucket

Profiling resource

Symbols distribution table

Sample Distribution Chart

Java source code with hotness bar

Ticks	%	Module	Symbol
331	13.03	JITCODE	Hello.foo(I)V
264	10.39	D:\...\j9jit23.dll	returnFromJIT0
239	9.41	D:\...\j9thr23.dll	j9thread_spinlock_sw
198	7.79	D:\...\j9jit23.dll	i2jTransition
141	5.55	D:\...\j9vm23.dll	JBinvokestatic
116	4.57	D:\...\j9vm23.dll	JBload2
114	4.49	D:\...\j9vm23.dll	JBreturn
94	3.70	D:\...\j9vm23.dll	JBldc
87	3.42	D:\...\j9vm23.dll	JBiinc
80	3.15	D:\...\j9jit23.dll	TR_IProfiler::bcHash
75	2.95	D:\...\j9vm23.dll	JBbipush
74	2.91	D:\...\j9vm23.dll	JBsipush
62	2.44	D:\...\j9jit23.dll	vaint
Total: ticks (2541), time share (64.15%)			

Line	Source	%	Ticks
13	}		
14			
15	public static void foo(int i)		
16	{		
17	int foo = bar(i);	84.59	280
18			
19	if (foo == 0)	12.08	40
20	{		
21	flag = !flag;	0.60	1
22	if (flag)		
23	System.err.print('.');	0.91	1
24	}		
25	}	5.44	9
26			
27	public static void main(String[] argv)		
28	{		
29			
30	int numRuns = 1;		

Profile Analyzer - symbol distribution and disassembly

The screenshot displays the Profile Analyzer interface with the following components:

- Profiling Resource:** A tree view on the left showing the local host and sample directories.
- Profile Hierarchy:** A tree view in the center showing the process hierarchy, including Java (3961 ticks/78.90%) and JIT (2413 ticks/94.96%).
- Symbols Distribution Table:** A table on the right showing the distribution of symbols across modules.

Ticks	%	Module	Symbol
331	13.72	JITCODE	Hello.foo(I)V
264	10.94	D:\...\j9jit23.dll	returnFromJIT0
239	9.90	D:\...\j9thr23.dll	j9thread_spinlock_sw
198	8.21	D:\...\j9jit23.dll	i2jTransition
141	5.84	D:\...\j9vm23.dll	JBinvokestatic
116	4.81	D:\...\j9vm23.dll	JBload2
114	4.72	D:\...\j9jit23.dll	TR_IProfiler::addSam
94	3.90	D:\...\j9vm23.dll	JBldc
87	3.61	D:\...\j9vm23.dll	JBiinc
80	3.32	D:\...\j9jit23.dll	TR_IProfiler::bcHash
75	3.11	D:\...\j9vm23.dll	JBbipush
74	3.07	D:\...\j9vm23.dll	JBsipush
62	2.57	D:\...\j9jit23.dll	vaint
Total: ticks (2413), time share (94.96%)			
- Basic Block:** A flow diagram on the bottom left showing basic blocks BB2 (289 ticks), BB3 (18 ticks), BB4 (1 tick), and BB5 (0 ticks).
- Disassembly/Offset View:** A table on the bottom right showing the disassembly of the selected symbol.

Offset	Bytes	Disassembly	%	Ticks	Remarks
0x00000016	f7ef	IMUL EDI		17	
0x00000018	c1fa06	SAR EDX,6	14.80	49	17
0x0000001b	8bda	MOV EBX,EDX	9.06	30	17
0x0000001d	c1eblf	SHR EBX,31	8.46	28	17
0x00000020	03d3	ADD EDX,EBX	3.32	11	17
0x00000022	69d2e8030000	IMUL EDX,EDX,3e8H	9.97	23	17
0x00000028	2bfa	SUB EDI,EDX	24.77	62	17
0x0000002a	85ff	TEST EDI,EDI	9.37	31	19
0x0000002c	7407	JE 35H			19
0x0000002e	5b	POP EBX	2.72	9	19
0x0000002f	83c408	ADD ESP,8H	1.21	4	25
0x00000032	c20400	RET 4H	1.51	5	25
0x00000035	8b0590e1eb00	MOV EAX,DWORD PTR [0ebe190H]			25
0x0000003b	85c0	TEST EAX,EAX	0.30	1	21
0x0000003d	7537	JNE 76H			21
0x0000003f	bb01000000	MOV EBX,1H			21

Profiling resource

Profile hierarchy

Symbols distribution table

Basic block

Disassembly/Offset view

Profile Analyzer - temporal profiling

The screenshot displays the Profile Analyzer interface within the Eclipse SDK. The main window shows a profile hierarchy for 'Hello_java.out', listing various DLLs and their tick counts. A table on the right provides a detailed view of the symbols distribution, including ticks, percentage, and symbol names. The bottom section features a temporal profiling chart showing ticks over time, with a list of events and their counts below it.

Profile hierarchy

Symbols distribution

Ticks	%	Symbol
264	26.91	returnFromJIT0
198	20.18	i2jTransition
114	11.62	TR_IPProfiler::addSampleData
80	8.15	TR_IPProfiler::bcHash
62	6.32	paint
48	4.89	TR_IPProfiler::searchForSample
42	4.28	parseBuffer
40	4.08	TR_IPProfiler::profilingSample_**DUP**_0x0000B
19	1.94	TR_IPBCDataCompact::getData
16	1.63	TR_IPProfiler::releaseHashTableWriteLock
11	1.12	TR_IPProfiler::profilingSample_**DUP**_0x0000B
11	1.12	TR_IPProfiler::canFitDataInOneByte
8	0.82	TR_IPProfiler::acquireHashTableWriteLock

Counters information

Counter	Count
Ticks	981

Temporal profiling chart for modules and symbols

Module D:\...\j9jit23.dll, 979/981 events found

- returnFromJIT0 (264 events)
- i2jTransition (198 events)
- TR_IPProfiler::addSampleData (114 events)
- TR_IPProfiler::bcHash (80 events)
- paint (62 events)
- TR_IPProfiler::searchForSample (48 events)

Code analyzer – motivations

- Architectures are becoming more and more complex
- Using only hardware simulators to detect information about potential instruction level performance bottlenecks in a given program is very hard
- There is a need for performance tools that can statically analyze and visualize programs structure and characteristics for a specific platform design
- Target audience includes
 - hardware architects
 - compiler writers
 - application developers

Code Analyzer - overview

- Code Analyzer is an eclipse plugin which performs a comprehensive static analysis on executable files and libraries/DLLs
 - relies on the FDPR-Pro tool for the analysis phase
- Displays the analyzed information together with optional profiling data collected by
 - tprof (AIX only)
 - FDPR-Pro (AIX and Linux PPC) phase 2 output
 - ▶ `fdpr -12 -p binary -x binary binary_options`
- The code is then colored according to:
 - frequency counters
 - ▶ gathered by FDPR-Pro
 - can also collect value profiles
 - hardware event ticks
 - ▶ gathered by tprof

Code Analyzer – overview(*cont*)

- Provides several views of the input binary
 - assembly instructions
 - basic blocks
 - procedures
 - CSECT modules
 - control flow graph
 - hot loops
 - call graph
 - annotated source code
 - dispatch group formation
 - pipeline slots and functional units

The screenshot displays the Perf Visual interface within the Eclipse Platform. The main window shows the 'Instructions Table' view, which lists assembly instructions with their addresses, opcodes, mnemonics, comments, frequencies, and ticks. The instructions are color-coded by frequency, with red indicating higher frequency and blue indicating lower frequency. The table is as follows:

Address	Opcode	Mnemonic	Comment	Freq.	Ticks	Graph
0x100040e4	0x7c0802a6	mflr r0	.livecdr	973686	1 (0.023%)	
0x100040e8	0x93e1fffc	stw r31,-4(r1)		973686	---	
0x100040ec	0x90010008	stw r0,8(r1)		973686	---	
0x100040f0	0x9421ffb0	stwu r1,-80(...)		973686	---	
0x100040f4	0x83e20070	lhz r31,112(...)	-> 0x20007...	973686	---	
0x100040f8	0x90610068	stw r3,104(r1)		973686	---	
0x100040fc	0x80610068	lhz r3,104(r1)		973686	4 (0.090%)	
0x10004100	0x88630000	lbz r3,0(r3)		973686	---	
0x10004104	0x90610044	stw r3,68(r1)		973686	---	
0x10004108	0x3083ffff	addc r4,r3,-1		973686	---	
0x1000410c	0x28040001	cmpli cr0,0x0...		973686	3 (0.068%)	
0x10004110	0x40810020	ble cr0,0x10...		973686	---	
0x10004114	0x3083fffd	addc r4,r3,-3		943990	2 (0.045%)	
0x10004118	0x28040001	cmpli cr0,0x0...		943990	---	
0x1000411c	0x4081001c	ble cr0,0x10...		943990	---	
0x10004120	0x3063fffb	addc r3,r3,-5		64864	---	
0x10004124	0x28030005	cmpli cr0,0x0...		64864	---	
0x10004128	0x40810008	ble cr0,0x10...		64864	---	

The interface also includes a 'Program Tree' on the left showing the file structure, an 'Error Log' at the bottom left, and an 'Instruction Properties' window at the bottom right showing branch and value profiles.

Code Analyzer - features

- Analysis features provided
 - easy code navigation
 - ▶ basic block, function, hot loops, ...
 - architectural comments generation
 - ▶ based on static analysis of code structure
 - performance comments generation
 - ▶ when profile information is available
 - dispatch group and pipeline functional units mapping
 - source code mapping of executable
 - statistical views
- Binary types currently supported
 - AIX XCOFF
 - Linux ELF
 - JITed code
 - ▶ embedded in AIX tprof XML output file and
 - ▶ J2N files produced by Performance Inspector's jprof

Code Analyzer - views

The screenshot displays the Code Analyzer application interface. The main window is titled "CodeAnalyzer - D:\vpn_testing_data\Samples\CodeAnalyzer\li32 - Eclipse Platform". The interface includes a menu bar (File, Edit, Refactor, Navigate, Search, Project, Run, Window, Help), a toolbar, and several panes:

- Program Tree:** Located on the left, it shows a hierarchical view of the executable file "xldmem.c" and its various sections, including ".evform", ".xlapply", ".xlxeval", ".xllevel", ".xlsym.c", ".xlsinit", ".xlfndprop", ".xliemprop", ".xlputprop", ".xlgetprop", ".xlsetvalue", ".xlygetvalue", ".xlxgetvalue", ".xlgetvalue", ".xlbind", ".xlframe", ".xlmakesym", ".xlsenter", ".xlenter", "xldmem.c", ".xlminit", ".stats", ".livecdr", ".livecar", and ".addseg".
- Instructions Table:** The central pane displays a table of instructions for the selected function ".addseg" at address 0x100036dc. The table has columns for Address, Opcode, Mnemonic, Comment, Freq., and Graph. The instructions are:

Address	Opcode	Mnemonic	Comment	Freq.	Graph
0x100036cc	0x80410014	lhz r2, 20(r1)		0	
0x100036d0	0x30210050	addic r1, r1, 80		973686	
0x100036d4	0x83e1ffff	lhz r31, -4(r1)		973686	
0x100036d8	0x4e800020	bclr 0x14, cr...		973686	
FUNCTION ...					
0x100036dc	0x7c0802a6	mflr r0	.addseg	9	
0x100036e0	0x93e1ffff	stw r31, -4(r1)		9	
0x100036e4	0x90010008	stw r0, 8(r1)		9	
0x100036e8	0x9421ffb0	stwu r1, -80(...)		9	
0x100036ec	0x80620074	lhz r3, 116(r2)	-> anodes...	9	
0x100036f0	0x80630000	lhz r3, 0(r3)		9	
0x100036f4	0x2c030000	cmpi cr0, 0x0...		9	
0x100036f8	0x4082000c	bne cr0, 0x1...		9	
0x100036fc	0x38600000	li r3, 0		0	
0x10003700	0x48000130	b 0x10003830		0	
0x10003704	0x83e20074	lhz r31, 116(...)	-> anodes...	9	
0x10003708	0x807f0000	lhz r3, 0(r31)		9	
0x1000370c	0x3063ffff	addic r3, r3, -1		9	
0x10003710	0x1e63000c	mulli r3, r3, 12		9	
- Graph:** A vertical bar on the right side of the instructions table, showing the frequency of each instruction. The frequency values are 0, 973686, 973686, 973686, 9, 9, 9, 9, 9, 9, 9, 9, 0, 0, 9, 9, 9, 9.
- Console:** Located at the bottom left, it displays the current instruction: "BB: 715 inst: 7 Address: 0x100036f8 Mnemonic: bne cr0".
- Instruction Properties:** Located at the bottom right, it shows the "Branch Profile" tab with a table for "To Address" and "Count".

Program tree of an executable file

Detailed instruction information

Code Analyzer - architecture comments

- Three types of comments
 - Power5
 - Power6
 - general
- Examples of comments
 - pipeline stalls (bubbles) for the Power6 architecture
 - unresolved dependencies
 - unaligned code
 - unreachable code and non-used data
 - architectural hazards, for instance
 - ▶ load after store

Code Analyzer - performance comments

- Profile-based comments

- non-variant instructions within Hot loops
- hot function calls preceeded by overwriting non-volatile registers
- hot saves and restores of registers which could be relocated to cold spill areas
- hot instructions that could be scheduled to colder areas in the code
- removable hot branches
- dead code and non-used identification
- hot direct unconditional branches
- hot direct conditional branches that are taken, which have a colder fallthru
- hot call sites that are appropriate candidates for function inlining
- hot call sites that are appropriate for function specialization
- hot loops that are appropriate for loop unrolling
- performance bottlenecks identified with collected hardware events
- hot TOC load instructions that can be replaced by immediate add instructions

Pipeline Analyzer - features

- Pipeline visualization tool for POWER series processor
 - pipeline data comes from processor timing models which is fed an instruction trace of a program execution
- Provides views in two modes:
 - The “**scroll**” mode shows how instructions are executed in the pipeline
 - ▶ various pipeline events are represented by different symbols in different colors
 - The “**resource**” mode shows how resources are utilized during instruction execution
 - ▶ in this view, a continuous bar in a color means during this cycle period a particular resource is being utilized.
- Provides graphical views of pipeline data, easy data navigation and customization for GUI elements.
- Very detailed hardware performance data is provided
 - useful for hardware performance design or critical library or module analysis.

Pipeline Analyzer – view in scroll mode

The screenshot shows the Pipeline Analyzer interface in scroll mode. The top window, titled "Pipeline Analyzer - scroll-mode - Eclipse SDK", contains a "Pipeline View" with a "General" panel on the left and a "Pipeline Overall View" on the right. The "General" panel displays: Cycles: 1355, Lines: 2357, Divider: 1, Offset Base: -1:-1, and Cursor: 608:1148. The "Pipeline Overall View" shows a graph of pipeline activity. Below this is a "scroll-mode" window showing a detailed pipeline scroll view. This view includes a main instruction stream with columns for Data, Inst, Arch, Mnemo, Top Id, Mnemo, Read, and Write. A tooltip is visible over the instruction at address 1149, showing details for the "stw" instruction: Data Addr: 2ff3f2a8, Inst Addr: d0092aa8, Arch Id: 886, Mnemonic: stw R3, 24 (R1), Top Id: 1149, and Read/Write registers: gpr3.

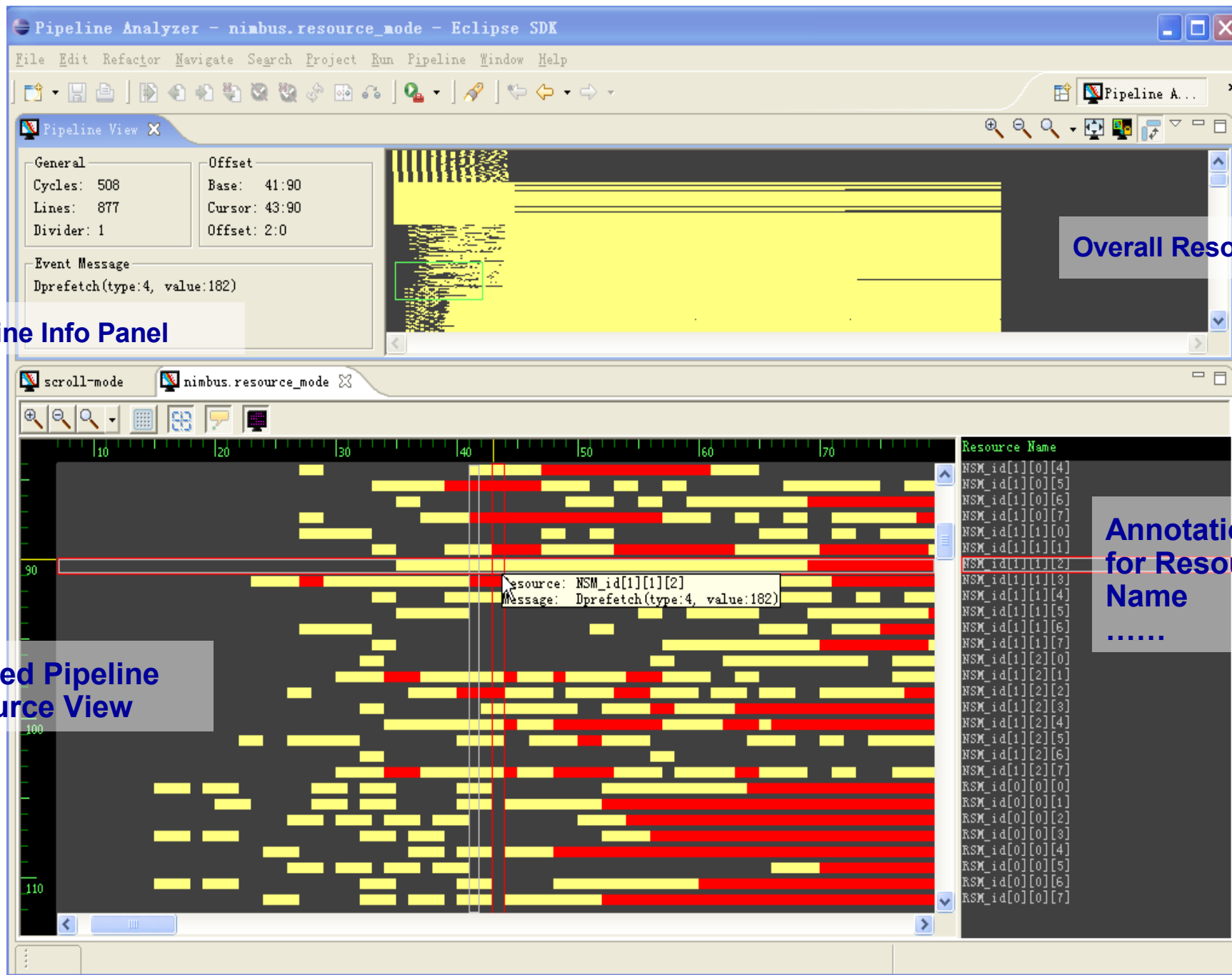
Pipeline Info Panel

Pipeline Overall View

Annotation Bars for Instruction, I_Address, d_Address, ...

Detailed Pipeline Scroll View

Pipeline Analyzer – view in resource mode



Pipeline Info Panel

Overall Resource View

Annotation Bars for Resource Name

Detailed Pipeline Resource View

Code Availability

- AIX Profile Analyzer support
 - tprof -X option(undocumented) produces XML file loadable into VPA
 - ▶ included with AIX 5.3 TL5
- Alphaworks (www.alphaworks.ibm.com)
 - Virtual Performance Analyzer
 - ▶ Windows and AIX binaries
 - Performance Inspector for Windows
 - ▶ profiler for Windows
 - Post-Link Optimization for Linux on Power
 - ▶ FDPR for Linux PowerPC
 - Full-System Simulator for IBM PowerPC 970
 - ▶ can generate instruction traces to be used as input for processor timing models
 - IBM Performance Simulator for Linux on POWER
 - ▶ timing models for most recent PowerPC processors
 - output feeds into Pipeline Analyzer
 - ▶ also includes older standalone version of Pipeline Analyzer
- Sourceforge
 - Linux Performance Inspector
 - ▶ support Intel, AMD and PPC64
 - ▶ includes itrace tool which feeds into processor timing models
 - <http://perfinsp.sourceforge.net/>

Future plans

- Better platform coverage
 - Linux PPC and Intel VPA binaries
 - ▶ should be available in the next release scheduled for Q4
 - goal is to cover all of IBM's server platforms and Operating Systems
 - ▶ internal version already provides wider coverage
 - full static and dynamic code coverage
- Better plugin interaction
 - goal is to provide true point and click drill down experience
 - fully functional remote data collector
- Future plugins
 - PM data postprocessing
 - Trace visualization
 - built-in instruction trace collection and processor models
- Feedback from alphaworks

Recorded demo

Profile Analyzer - initial view of a profile

Profile Analyzer - C:/vpa3.0.0rc/workspace/Profile Analyzer/78434532.out - Visual Performance Analyzer

File Edit Search Tools Window Help

Navigator Database Connections »1

C:/vpa3.0.0rc/workspace/Profile Analyzer/78434532.out

78434532.out (5020.0 ticks) [default counter: Ticks]

- Process > Thread > Module
- Process > Module
- Modules
- My Hierarchy

Ticks	%	Process	Thread
604	12.03	IdleProcess [PID 0]	tid_0000 [TID 0]
331	6.59	java.exe [PID 3356]	tid_main_0c94 [
280	5.58	javaw.exe [PID 2908]	tid_0680 [TID 1]
264	5.26	java.exe [PID 3356]	tid_main_0c94 [
238	4.74	java.exe [PID 3356]	tid_main_0c94 [
219	4.36	javaw.exe [PID 2908]	tid_0c8c [TID 3]
198	3.94	java.exe [PID 3356]	tid_main_0c94 [
176	3.51	javaw.exe [PID 2908]	tid_0680 [TID 1]
169	3.37	javaw.exe [PID 2908]	tid_0c8c [TID 3]
141	2.81	java.exe [PID 3356]	tid_main_0c94 [
116	2.31	java.exe [PID 3356]	tid_main_0c94 [
114	2.27	java.exe [PID 3356]	tid_main_0c94 [
94	1.87	java.exe [PID 3356]	tid_main_0c94 [
88	1.75	javaw.exe [PID 2908]	tid_0680 [TID 1]
87	1.73	java.exe [PID 3356]	tid_main_0c94 [
82	1.63	javaw.exe [PID 2908]	tid_0c8c [TID 3]

Process > Thread > Module (first 100 rows)

Samples Distribut... Counters »3

root: Process > Thread > Module - Total ticks: 5020, time shar
 root: All process under Process > Thread > Module

Disassembly/Offsets Compiler Listing Profile Details Java/Hierarchy Profile Comparison Temporal Profiling »5

Offsets for: Hello.foo(I)V (JITCODE)

Offset	Bytes	Disassembly	%	Ticks	Remarks
0x00000000	83ec08	SUB ESP,8H			
0x00000003	3b6518	CMP ESP,DWORD PTR [EBP+18H]	6.65	22	0
0x00000006	0f8694000000	JBE 0a0H			0
0x0000000c	53	PUSH EBX			0
0x0000000d	8b7c2410	MOV EDI,DWORD PTR [ESP+10H]	0.60	2	0
0x00000011	b8d34d6210	MOV EAX,10624dd3H	6.95	23	
0x00000016	f7ef	IMUL EDI			0
0x00000018	c1fa06	SAR EDX,6	14.80	49	
0x0000001b	8bda	MOV EBX,EDX	9.06	30	
0x0000001d	c1eblf	SHR EBX,31	8.46	28	
0x00000020	03d3	ADD EDX,EBX	3.32	11	0

Profile Analyzer - module view

Profile Analyzer - C:/vpa3.0.0rc/workspace/Profile Analyzer/Sample2.etm - Visual Performance Analyzer

File Edit Search Tools Window Help

Profiling Resources Navigator

Sample2.etm (374567.0 ticks) [default counter: Ticks]

Process -> Module

Modules

Ticks	%	Module	Symbol/Functions
178023	47.53	JITCODE	spec/benchmarks/_201_compres
63636	16.99	JITCODE	spec/benchmarks/_201_compres
43320	11.57	JITCODE	spec/benchmarks/_201_compres
38403	10.25	JITCODE	spec/benchmarks/_201_compres
6837	1.83	NoModule	NoSymbol
5491	1.47	NoModule	NoSymbol
2671	0.71	lib9vm23	JBALOAD0GETFIELD
2614	0.70	lib9vm23	JBILOAD
2319	0.62	GFUAPFS1	GFUGRWNE
2289	0.61	lib9vm23	JBGETFIELD
1893	0.51	lib9gc23	J9AllocateIndexableObject
1822	0.49	CEEPLPKA	NoSymbol
1301	0.35	lib9vm23	JBISTORE
1291	0.34	lib9vm23	JBILOAD2
1154	0.31	NoModule	NoSymbol
1127	0.30	lib9jit23	J9JITTRANSITION

Modules (first 100 rows)

Disassembly/Offsets Compiler Listing Profile Details Java/Hierarchy Profile Comparison Temporal Profiling

Offsets for: spec/benchmarks/_201_compress/Code_Table.of(I)I[warm] (JITCODE)

Offset	Bytes	Disassembly	%	Ticks	CTRL
0x0000000E	E3505FF4FF71	LAY GPR5, -12(,GPR5)	7.43	11	6
0x00000014	5550D018	CL GPR5, 24(,GPR13)	1.35	2	2
0x00000018	C044003FAE2C	BRCL 4, *+8346712	7.43	11	11
0x0000001E	1832	LR GPR3, GPR2			
0x00000020	BFEF100C	ICM GPR14, B'1111', 12(GPR1)	1.35	2	2
0x00000024	C084003FAE34	BRCL 8, *+8346728	2.70	4	4
0x0000002A	5800E00C	L GPRO, 12(,GPR14)	8.11	12	2
0x0000002E	1503	CLR GPRO, GPR3			
0x00000030	C0C4003FAE2B	BRCL 12, *+8346710	1.35	2	1
0x00000036	41333000	LA GPR3, 0(GPR3, GPR3)	3.38	5	5
0x0000003A	E32E30100091	LLGH GPR2, 16(GPR14, GPR3)	22.30	33	13

Samples Distribut...

86.45% 5.33% 4.17% 4.05%

root: Modules - Total ticks: 374567, time share: 100.00%

root: All module under Modules

Profile Analyzer – context-sensitive module graph

Profile Analyzer - C:/vpa3.0.0rc/workspace/Profile Analyzer/Sample2.etm - Visual Performance Analyzer

File Edit Search Tools Window Help

Profiling Resources Navigator

- Profile Analyzer
 - Code Analyzer
 - current
 - db
 - pipeline
 - Sample
 - .project
 - 58343103.log
 - 66294918.etm
 - 78434532.out
 - 84338450.etm
 - pi_config.cfg
 - Sample1.etm
 - Sample2.etm

C:/vpa3.0.0rc/workspace/Profile Analyzer/Sample2.etm

- Sample2.etm (374567.0 ticks) [default counter: Ticks]
 - Process > Module
 - Modules
 - JITCODE (323806 ticks/86.45%)**
 - lib9vm23 (19951 ticks/5.93%)
 - NoModule (15636 ticks/4.17%)
 - lib9jit23 (6366 ticks/1.70%)
 - GFUAPFS1 (2410 ticks/0.64%)
 - lib9gc23 (2379 ticks/0.64%)
 - CEEPLPKA (1824 ticks/0.49%)
 - IRASAMFC (387 ticks/0.10%)
 - lib9jtd23 (267 ticks/0.07%)
 - BPXINPVT (258 ticks/0.07%)
 - IEAVEDAT (213 ticks/0.06%)
 - lib9a2e (165 ticks/0.04%)
 - lib9dyn23 (125 ticks/0.03%)
 - CELHV003 (103 ticks/0.03%)
 - IRXINIT (99 ticks/0.03%)

Ticks	%	Symbol/Functions
178023	54.98	spec/benchmarks/_201_compress/Compress...
63636	19.65	spec/benchmarks/_201_compress/Decompre...
43320	13.38	spec/benchmarks/_201_compress/Compress...
38403	11.86	spec/benchmarks/_201_compress/Decompre...
148	0.05	spec/benchmarks/_201_compress/Code_Tab...
95	0.03	spec/benchmarks/_201_compress/Input_Bufl...
73	0.02	spec/benchmarks/_201_compress/Output_Bu...
26	0.01	spec/benchmarks/_201_compress/Code_Tab...
15	0.00	spec/benchmarks/_201_compress/Compress...
8	0.00	java/lang/StringBuffer.append(C)Ljava/lang/...
5	0.00	java/io/FileInputStream.read()I[warm]
5	0.00	java/lang/String.hashCode()I[warm]
4	0.00	spec/io/FileInputStream.read()I[warm]
3	0.00	spec/io/ValidityCheckOutputStream.strip1(Lj...
3	0.00	java/lang/StringCoding.encode(Ljava/lang/St...
2	0.00	java/util/Properties.getProperty(Ljava/lang/S...

Total: ticks (323806), time share (86.45%) (43 rows)

Samples Distribut...

54.98% 19.65% 13.38% 11.86% 0.13%

module: JITCODE - Total ticks: 323806, time share: 86.45%

module: All Symbol under JITCODE

Disassembly/Offsets Compiler Listing Profile Details Java/Hierarchy Profile Comparison Temporal Profiling

Offsets for: spec/benchmarks/_201_compress/Code_Table.of(I)I[warm] (JITCODE)

Offset	Bytes	Disassembly	%	Ticks	CTR1
0x0000000E	E3505FF4FF71	LAY GPR5, -12(, GPR5)	7.43	11	6
0x00000014	5550D018	CL GPR5, 24(, GPR13)	1.35	2	2
0x00000018	C044003FAE2C	BRCL 4, *+8346712	7.43	11	11
0x0000001E	1832	LR GPR3, GPR2			
0x00000020	BFEF100C	ICM GPR14, B'1111', 12(GPR1)	1.35	2	2
0x00000024	C084003FAE34	BRCL 8, *+8346728	2.70	4	4
0x0000002A	5800E00C	L GPRO, 12(, GPR14)	8.11	12	2
0x0000002E	1503	CLR GPRO, GPR3			
0x00000030	C0C4003FAE2B	BRCL 12, *+8346710	1.35	2	1
0x00000036	41333000	LA GPR3, 0(GPR3, GPR3)	3.38	5	5
0x0000003A	E32E30100091	LLGH GPR2, 16(GPR14, GPR3)	22.30	33	13

CLR - Complate Logical RR variant CLR - Compare Logical RR

Profile Analyzer - Java class hierarchy view

The screenshot displays the Profile Analyzer application window with the following components:

- Navigator:** Shows the project structure including Profile Analyzer, Code Analyzer, current, db, pipeline, and Sample folders.
- Tree View:** Displays the loaded modules for Sample2.etm, including JITCODE (86.45%), libj9vm23 (5.33%), NoModule (4.17%), and various other JVM libraries.
- Table:** A table listing the top modules with columns for Ticks, %, Module, and Symbol/Functions.
- 3D Bar Chart:** Visualizes the distribution of ticks across modules: JITCODE (86.45%), libj9vm23 (5.33%), NoModule (4.17%), and Other (4.05%).
- Class Hierarchy:** Shows the hierarchy for the selected class `java/lang/String`, listing methods like `hashCode()`, `equals()`, `charAt()`, and `replace()`.

Ticks	%	Module	Symbol/Functions
178023	47.53	JITCODE	spec/benchmarks/_201_compres
63636	16.99	JITCODE	spec/benchmarks/_201_compres
43320	11.57	JITCODE	spec/benchmarks/_201_compres
38403	10.25	JITCODE	spec/benchmarks/_201_compres
6837	1.83	NoModule	NoSymbol
5491	1.47	NoModule	NoSymbol
2671	0.71	libj9vm23	JBALOAD0GETFIELD
2614	0.70	libj9vm23	JBILOAD
2319	0.62	GFUAPFS1	GFUGRWNE
2289	0.61	libj9vm23	JBGETFIELD
1893	0.51	libj9gc23	J9AllocateIndexableObject
1822	0.49	CEEPLPKA	NoSymbol
1301	0.35	libj9vm23	JBISTORE
1291	0.34	libj9vm23	JBILOAD2
1154	0.31	NoModule	NoSymbol

Ticks	%	% R...	Method/Function Name
5	0.00	45.45	java/lang/String.hashCode()I[warm]
2	0.00	18.18	java/lang/String.equals(Ljava/lang/Ob
2	0.00	18.18	java/lang/String.charAt(I)C[warm]
1	0.00	9.09	java/lang/String.<init>();([CII)V[wa
1	0.00	9.09	java/lang/String.replace(CC)Ljava/lang

Profile Analyzer - temporal profiling: main process

Profile Analyzer - C:/vpa3.0.0rc/workspace/Profile Analyzer/78434532.out - Visual Performance Analyzer

File Edit Search Tools Window Help

Navigator

- Code Analyzer
- current
- db
- pipeline
- Sample
- .project
- 58343103.log
- 66294918.etm
- 78434532.out
- 84338450.etm
- pi_config.cfg
- Sample1.etm
- Sample2.etm

Resolved Calls

Method Hello.foo(I)V
Module: JITCODE
Process: java.exe

Known callers (calls to here; ticks in caller)

Symbol

Known descendants (calls to descendant from)

Symbol

C:/vpa3.0.0rc/workspace/Profile Analyzer/78434532.out

Process > Thread > Module	Ticks	%	Thread	Module
java.exe [PID 3356] (2541 ticks/50.62%)	331	13.03	tid_main_0c94 [TID 3220]	JITCODE
tid_main_0c94 [TID 3220] (2355 ticks/92.68%)	264	10.39	tid_main_0c94 [TID 3220]	D:\tr_jit\latest.dev\jre\bin
tid_0b40 [TID 2880] (142 ticks/5.59%)	228	9.02	tid_main_0c94 [TID 3220]	D:\tr_jit\latest.dev\jre\bin
tid_0c94 [TID 3220] (40 ticks/1.57%)	40	1.57	tid_main_0c94 [TID 3220]	D:\tr_jit\latest.dev\jre\bin

Total: ticks (2541), time share (50.62%) (first 100 rows)

Disassembly/Offsets Compiler Listing Profile Details Profile Comparison **Temporal Profiling** Query Tree CodeMiner Queries

process java.exe, 2379/2541 events found

25 intervals, 4.00 secs/interval

Zoom children vertically 100%

- tid_main_0c94 (2355.00 events)
- tid_0b40 (142.00 events)
- tid_0c94 (40.00 events)
- tid_0c64 (4.00 events)

Profile Analyzer - main process with children threads

Profile Analyzer - C:/vpa3.0.0rc/workspace/Profile Analyzer/78434532.out - Visual Performance Analyzer

File Edit Search Tools Window Help

Navigator

- Code Analyzer
- current
- db
- pipeline
- Sample
 - .project
 - 58343103.log
 - 66294918.etm
 - 78434532.out
 - 84338450.etm
 - pi_config.cfg
 - Sample1.etm
 - Sample2.etm

Resolved Calls

Method Hello.foo(I)V
Module: JITCODE
Process: java.exe

Known callers (calls to here; ticks in caller)

Symbol

Known descendants (calls to descendant from)

Symbol

C:/vpa3.0.0rc/workspace/Profile Analyzer/78434532.out

Process > Thread > Module	Ticks	%	Thread	Module
java.exe [PID 3356] (2541 ticks/50.62%)	331	13.03	tid_main_0c94 [TID 3220]	JITCODE
tid_main_0c94 [TID 3220] (2355 ticks/92.68%)	264	10.39	tid_main_0c94 [TID 3220]	D:\tr_jit\latest.dev\jre\bin
tid_0b40 [TID 2880] (142 ticks/5.59%)	228	9.02	tid_main_0c94 [TID 3220]	D:\tr_jit\latest.dev\jre\bin
tid_0c94 [TID 3220] (40 ticks/1.57%)	40	1.57	tid_main_0c94 [TID 3220]	D:\tr_jit\latest.dev\jre\bin

Total: ticks (2541), time share (50.62%) (first 100 rows)

Disassembly/Offsets Compiler Listing Profile Details Profile Comparison Temporal Profiling Query Tree CodeMiner Queries

process java.exe, 2379/2541 events found

25 intervals, 4.00 secs/interval

Zoom children vertically 100%

- tid_main_0c94 (2355.00 events)
- tid_0b40 (142.00 events)
- tid_0c94 (40.00 events)
- tid_0c64 (4.00 events)

Entire profile

Profile Analyzer - top thread with children modules

Profile Analyzer - C:/vpa3.0.0rc/workspace/Profile Analyzer/78434532.out - Visual Performance Analyzer

File Edit Search Tools Window Help

Navigator

- Code Analyzer
- current
- db
- pipeline
- Sample
- .project
- 58343103.log
- 66294918.etm
- 78434532.out
- 84338450.etm
- pi_config.cfg
- Sample1.etm
- Sample2.etm

Resolved Calls

Method Hello.foo(I)V
Module: JITCODE
Process: java.exe

Known callers (calls to here; ticks in caller)

Symbol

Known descendants (calls to descendant from)

Symbol

C:/vpa3.0.0rc/workspace/Profile Analyzer/78434532.out

Process > Thread > Module

Process	Thread	Module	Ticks	%	Module	Symb
java.exe	[PID 3356]		2541	50.62%	JITCODE	Hello...
	tid_main_0c94	[TID 3220]	2355	92.68%	D:\tr_jit\latest.dev\jre\bin\j9jit23.dll	return...
	tid_0b40	[TID 2880]	142	5.59%	D:\tr_jit\latest.dev\jre\bin\j9jit23.dll	return...
	tid_0c94	[TID 3220]	40	1.57%		

Total: ticks (2355), time share (46.91%) (73 rows)

Disassembly/Offsets Compiler Listing Profile Details Profile Comparison Temporal Profiling Query Tree CodeMiner Queries

Ticks

Seconds

thread tid_main_0c94, 2197/2355 events found

25 intervals, 4.00 secs/interval

Zoom children vertically 100%

- D:\tr_jit\latest.dev\jre\bin\j9jit23.dll
- D:\tr_jit\latest.dev\jre\bin\j9vm23.dll
- D:\tr_jit\latest.dev\jre\bin\j9thr23.dll
- JITCODE (337.00 events)
- C:\WINNT\system32\MSVCRT.dll (35.00 events)
- C:\WINNT\System32\ntoskrnl.exe (25.00 events)

Entire profile

Profile Analyzer - hide lower-hit children

Profile Analyzer - C:/vpa3.0.0rc/workspace/Profile Analyzer/78434532.out - Visual Performance Analyzer

File Edit Search Tools Window Help

Navigator

- Code Analyzer
- current
- db
- pipeline
- Sample
 - .project
 - 58343103.log
 - 66294918.etm
 - 78434532.out
 - 84338450.etm
 - pi_config.cfg
 - Sample1.etm
 - Sample2.etm

Resolved Calls

Method Hello.foo(I)V
Module: JITCODE
Process: java.exe

Known callers (calls to here; ticks in caller)

Symbol

Known descendants (calls to descendant from)

Symbol

C:/vpa3.0.0rc/workspace/Profile Analyzer/78434532.out

Process > Thread > Module	Ticks	%	Module	Symb
java.exe [PID 3356] (2541 ticks/50.62%)	331	14.06	JITCODE	Hello...
tid_main_0c94 [TID 3220] (2355 ticks/92.68%)	264	11.21	D:\tr_jit\latest.dev\jre\bin\j9jit23.dll	recurr...
tid_0b40 [TID 2880] (142 ticks/5.59%)
tid_0c94 [TID 3220] (40 ticks/1.57%)

Total: ticks (2355), time share (46.91%) (73 rows)

Disassembly/Offsets Compiler Listing Profile Details Profile Comparison Temporal Profiling Query Tree CodeMiner Queries

thread tid_main_0c94, 2197/2355 events found

25 intervals, 4.00 secs/interval

Zoom children vertically 100%

- D:\tr_jit\latest.dev\jre\bin\j9jit23.dll
- D:\tr_jit\latest.dev\jre\bin\j9vm23.dll
- D:\tr_jit\latest.dev\jre\bin\j9thr23.dll
- JITCODE (337.00 events)
- C:\WINNT\system32\MSVCRT.dll (35.00 events)
- C:\WINNT\System32\ntoskrnl.exe (25.00 events)

Profile Analyzer - symbols in a module

Profile Analyzer - C:/vpa3.0.0rc/workspace/Profile Analyzer/78434532.out - Visual Performance Analyzer

File Edit Search Tools Window Help

Navigator

- Code Analyzer
- current
- db
- pipeline
- Sample
 - .project
 - 58343103.log
 - 66294918.etm
 - 78434532.out
 - 84338450.etm
 - pi_config.cfg
 - Sample1.etm
 - Sample2.etm

Resolved Calls

Symbol i2jTransition
Module: D:\tr_jit\latest.dev\jre\bin\j9jit23.dll
Process: java.exe

Known callers (calls to here; ticks in caller)

Symbol

Known descendants (calls to descendant from)

Symbol

C:/vpa3.0.0rc/workspace/Profile Analyzer/78434532.out

Process > Thread > Module

- java.exe [PID 3356] (2541 ticks/50.62%)
 - tid_main_0r94 [TID 3220] (2355 ticks/92.68%)
 - D:\...j9jit23.dll (867 ticks/36.82%)
 - D:\...j9vm23.dll (692 ticks/29.38%)

Ticks	%	Symbol/Functions
264	30.45	returnFromJIT0
198	22.84	i2jTransition
114	13.15	TR_IProfiler::addSampleData

Total: ticks (867), time share (17.27%) (24 rows)

Disassembly/Offsets Compiler Listing Profile Details Profile Comparison Temporal Profiling Query Tree CodeMiner Queries

Ticks

Seconds

module D:\tr_jit\latest.dev\jre\bin\j9jit23.dll, 821/867 events found

25 intervals, 4.00 secs/interval

Zoom children vertically 100%

- returnFromJIT0 (264.00 events)
- i2jTransition (198.00 events)
- TR_IProfiler::addSampleData (114.00 events)
- TR_IProfiler::bcHash (80.00 events)
- TR_IProfiler::searchForSample (47.00 events)
- parseBuffer (42.00 events)

Entire profile

Profile Analyzer - source and disassembly view

Profile Analyzer - C:/vpa3.0.0rc/workspace/Profile Analyzer/78434532.out - Visual Performance Analyzer

File Edit Search Tools Window Help

Navigator

- Code Analyzer
- current
- db
- pipeline
- Sample
- .project
- 58343103.log
- 66294918.etm
- 78434532.out
- 84338450.etm
- pi_config.cfg
- Sample1.etm
- Sample2.etm

Resolved Calls

Method: Hello.foo(I)V
Module: JITCODE
Process: java.exe

Known callers (calls to here; ticks in caller)

Symbol

Known descendants (calls to descendant from)

Symbol

C:/vpa3.0.0rc/workspace/Profile Analyzer/78434532.out

78434532.out (5020.0 ticks) [default counter: Ticks]

- Process > Thread > Module
- java.exe [PID 3356] (2541 ticks/50.62%)
 - tid_main_0c94 [TID 3220] (2355 ticks/92.68%)
 - tid_0b40 [TID 2880] (142 ticks/5.59%)
 - tid_0c94 [TID 3220] (40 ticks/1.57%)

Ticks	%	Module	Symb
331	14.06	JITCODE	Hello.
264	11.21	D:\tr_jit\latest.dev\jre\bin\j9jit23.dll	return
238	10.11	D:\tr_jit\latest.dev\jre\bin\j9thr23.dll	j9thre

Total: ticks (2355), time share (46.91%) (73 rows)

Disassembly/Offsets Compiler Listing Profile Details Profile Comparison Temporal Profiling Query Tree CodeMiner Queries

Offsets for: Hello.foo(I)V (JITCODE)

Offset	Bytes	Disassembly	%	Ticks	Remarks
0x00000000	83EC08	SUB ESP,8H			
0x00000003	3B6518	CMP ESP,DWORD PTR [BBP+18H]	6.65	22	0
0x00000006	0F8694000000	JBE 0A0H			0
0x0000000C	53	PUSH EBX			0
0x0000000D	8B7C2410	MOV EDI,DWORD PTR [ESP+10H]	0.60	2	0
0x00000011	B8D34D6210	MOV EAX,10624DD3H	6.95	23	
0x00000016	F7EF	IMUL EDI			0
0x00000018	C1FA06	SAR EDX,6	14.80	49	
0x0000001B	8BDA	MOV EBX,EDX	9.06	30	
0x0000001D	C1FB1F	SHD EBX,31	8.46	28	

Source Code

Source for: Hello.foo(I)V (JITCODE)

Line	Source	%	Ticks
24	}		
25	}		
26			
27	public static void main(String[] argv)		
28	{		
29			
30	int numRuns = 1;		
31			
32	if (argv.length >= 1)		
33	{		
34	try		

Profile Analyzer – compiler listing view

Profile Analyzer - C:/vpa3.0.0rc/workspace/RemoteSystemsTempFiles/9.3.104.80/home/tscripts/sema/cfiles/onefile/out.etm - Visual Performance ...

File Edit Search Tools Window Help

Profiling Resources Navigator »1

LOCALHOST (Local)
9.3.104.80 (AIX)
9.3.104.45 (AIX)
9.3.104.25 (AIX)

C:/vpa3.0.0rc/workspace/RemoteSystemsTempFiles/9.3.104.80/home/tscripts/sema/cfiles/onefile/out.etm »2

out.etm (1009.0 ticks) [default counter: Ticks]

- Process > Thread > Module
 - test [PID 22784] (502 ticks/49.75%)
 - test TID 37685 [TID 37685] (502 ticks/100.00%)
 - /unix (476 ticks/94.82%)
 - ./test (26 ticks/5.18%)
 - wait [PID 516] (500 ticks/49.55%)
 - tprof [PID 23458] (3 ticks/0.30%)
 - brcton [PID 22786] (1 ticks/0.10%)

Ticks % Symbol/Functions

19	73.08	.semctl
7	26.92	.badperf

Total: ticks (26), time share (2.58%) (2 rows)

Disassembly/Offsets Compiler Listing Profile Comparison Temporal Profiling Remote Shell Remote Systems »3

Listing for: .badperf (./test)

Line#	%	Ticks	Source#	Offset	Opcode	Bytes	Latency	XIL
347			49	000220	bl	4BFFFDE1	0	CALL
348			49	000224	ori	60000000	1	
349			49	000228	b	4800005C	0	B
350			49					CL.9:
351			49					CL.11:
352			54	00022C	lwz	8061004C	1	L4A
353			54	000230	addi	38800000	1	LI
354			54	000234	addi	38A00003	1	LI
355			54	000238	addi	38C00000	1	LI
356			54	00023C	bl	4BFFFDC5	0	CALL
357			54	000240	ori	60000000	1	
358			54	000244	cmpwi	2C030001	1	C4
359			54	000248	bc	40800024	1	BF
360			54					CL.13:
361			54	00024C	lwz	8061004C	1	L4A
362	14.29	1	54	000250	addi	38800000	1	LI
363			54	000254	addi	38A00003	1	LI
364			54	000258	addi	38C00000	1	LI
365			54	00025C	bl	4BFFFDA5	0	CALL
366	71.43	5	54	000260	ori	60000000	1	
367	14.29	1	54	000264	cmpwi	2C030001	1	C4
368			54	000268	bc	41800004	1	RT

Samples Distribut... »4

73.08% 26.92%

module: ./test - Total ticks: 26, time share: 2.58%

module: All Symbol under ./test

Profile Analyzer - normalization factors for comparisons

Profile Comparison Wizard

Enter the normalization factors for each profile.

Each profile will be normalized for direct comparison using the following formulas:
 CPU% = the number of ticks in the specific function / total ticks in the profile
 ITR = Transaction rate / CPU utilization
 Total Microseconds per transaction = (1,000,000 / ITR) * Number of CPUs
 Ms/Tx = Total Microseconds per transaction * CPU%

Profile 1

Transaction rate: 1000000.0
 CPU utilization (%): 100.0
 Number of CPUs: 1

Profile 2

Transaction rate: 1000000.0
 CPU utilization (%): 100.0
 Number of CPUs: 1

< Back Next > Finish Cancel

Background Interface:

Profile Analyzer - C:/vpa3.0.0rc/workspace/Profile Analyzer/84338450.etm - Visual Performance Analyzer

File Edit Search Tools Window Help

Profiling Resources Navigator Database Conn

Code Analyzer
 current
 db
 pipeline
 Sample
 .project
 58343103.log
 66294918.etm
 78434532.out
 84338450.etm
 pi_config.cfg
 Sample1.etm
 Sample2.etm

Samples Distribut... Resolved Calls Symbol D

Symbol .rsemctl
 Module /unix
 Process: test

Known callers (calls to here; ticks in caller)

Symbol

Known descendants (calls to descendant from all sources; ticks in descendant)

Symbol Calls

IBMinvscout
 LicenseUseManagement
 lost+found
 perl
 rseserver
 Tivoli
 vpatprof
 out.etm
 run.tprof_xml.sh
 run.tprof_xml.sh.mod

PM_CYC	%	Process	Thread
234	15.99	java [PID 368826]	java TID 77
195	13.33	wait [PID 49176]	wait TID 57
148	10.12	wait [PID 8196]	wait TID 81
146	9.98	wait [PID 8196]	wait TID 81
65	4.44	test [PID 327920]	test TID 11
64	4.37	java [PID 368826]	java TID 77
41	2.80	test [PID 327920]	test TID 11
33	2.26	test [PID 327920]	test TID 11
32	2.19	test [PID 327920]	test TID 11
31	2.12	test [PID 327920]	test TID 11
30	2.05	wait [PID 49176]	wait TID 57
30	2.05	wait [PID 49176]	wait TID 57
27	1.85	wait [PID 8196]	wait TID 81
26	1.78	test [PID 327920]	test TID 11
26	1.78	test [PID 327920]	test TID 11
24	1.64	test [PID 327920]	test TID 11
24	1.64	test [PID 327920]	test TID 11
20	1.37	test [PID 327920]	test TID 11

Profile Analyzer - profile comparison chart

The screenshot shows the IBM Profile Analyzer interface. The main window displays a comparison between two profile files: `C:/vpa3.0.0rc/workspace/Profile Analyzer/84338450.etm` and `C:/vpa3.0.0rc/workspace/Profile Analyzer/66294918.etm`. The interface includes a Navigator on the left, a Resolved Calls pane, and a main comparison table.

The comparison table shows the following data:

%CPU_1	%CPU_2	us/Tx_1	us/Tx_2	us/Tx_delta	%change	%total	%accum	Module	Symbol
[60.01]	[74.68]	[0.60]	[0.75]	[0.15]	[24.45]			[Totals]	
0.62	16.43	0.01	0.16	0.16	2571.57	107.81	107.81	NoModule	NoSymbol
2.19	1.27	0.02	0.01	-0.01	-41.97	-6.26	101.55	/unix	.simple_lock
4.37	3.54	0.04	0.04	-0.01	-19.07	-5.68	95.87	/.../libjvm.a	.JVM_IsThreadAlive
1.78	1.00	0.02	0.01	-0.01	-43.62	-5.28	90.58	/unix	.getgidx
2.12	2.74	0.02	0.03	0.01	29.26	4.23	94.81	/unix	.trchook64
0.48	1.07	0.00	0.01	0.01	123.41	4.02	98.83	/unix	sc_msr_2_point
1.09	0.60	0.01	0.01	0.00	-45.03	-3.36	95.48	/unix	.mtrchook1
0.34		0.00		0.00		-2.33	93.15	/unix	sc_debug_point
2.80	3.14	0.03	0.03	0.00	12.03	2.30	95.45	/unix	.getuidx
	0.33		0.00	0.00		2.28	97.72	/.../test	.badperf
13.33	13.63	0.13	0.14	0.00	2.24	2.03	99.76	/unix	.waitproc
1.03	0.73	0.01	0.01	0.00	-28.33	-1.98	97.78	/unix	.svc_instr
0.96	0.67	0.01	0.01	0.00	-30.19	-1.97	95.81	/unix	.priv_req
1.30	1.54	0.01	0.02	0.00	18.30	1.62	97.43	/unix	sc_ctrace_start_point
2.26	2.07	0.02	0.02	0.00	-8.19	-1.26	96.17	/unix	.sc_ctrace_start_fixup
1.23	1.40	0.01	0.01	0.00	14.02	1.18	97.35	/unix	.ipcaccess
2.05	2.20	0.02	0.02	0.00	7.51	1.05	98.39	/unix	.fetch_and_nop
0.75	0.60	0.01	0.01	0.00	-20.03	-1.03	97.37	/unix	sc_msr_2_fixup
0.62	0.47	0.01	0.00	0.00	-23.98	-1.01	96.36	/unix	.sc_trace_start_fixup
0.14		0.00		0.00		-0.93	95.43	/unix	mtrchook2_syst
0.14		0.00		0.00		-0.93	94.50	/unix	sc_trace_end_point
0.07	0.20	0.00	0.00	0.00	193.41	0.90	95.40	/unix	.semctl
0.07	0.20	0.00	0.00	0.00	193.41	0.90	96.30	/unix	svc_call_ret

Code Analyzer - main view

Perf Visual - Eclipse Platform

File Edit Navigate Search Project Perf Visual Run Tests Run Window Help

Program Tree Navigator

Label View BB: 1017 inst: 4 Address: 0x10004724 Mnemonic: rlwinm r3,r3,0x0,0x18,0x1f Opcode: 0x5463063e Remark:

Instructions Table

Address	Opcode	Mnemonic	Comment	Freq.	Graph
0x10004714	0x80810040	lwz r4,64(r1)		973686	
0x10004718	0x9081004c	stw r4,76(r1)		973686	
0x1000471c	0x88640001	lbz r3,1(r4)		973686	
0x10004720	0x60630001	ori r3,r3,0x1		973686	
0x10004724	0x5463063e	rlwinm r3,r3,0x...		973686	
0x10004728	0x98640001	stb r3,1(r4)		973686	
0x1000472c	0x80610040	lwz r3,64(r1)		973686	
0x10004730	0x4bfffa85	bl 0x100041b4	livecar	973686	
0x10004734	0x2c030000	cmpl cr0,0x0,r3,0		973686	
0x10004738	0x41820044	beq cr0, 0x100...		973686	
0x1000473c	0x80810040	lwz r4,64(r1)		868467	
0x10004740	0x90810050	stw r4,80(r1)		868467	
0x10004744	0x88640001	lbz r3,1(r4)		868467	
0x10004748	0x60630002	ori r3,r3,0x2		868467	
0x1000474c	0x5463063e	rlwinm r3,r3,0x...		868467	
0x10004750	0x98640001	stb r3,1(r4)		868467	
0x10004754	0x80610044	lwz r3,68(r1)		868467	
0x10004758	0x90610048	stw r3,72(r1)		868467	
0x1000475c	0x80610040	lwz r3,64(r1)		868467	
0x10004760	0x00610044	stw r3,68(r1)		868467	

Instructions Source Code

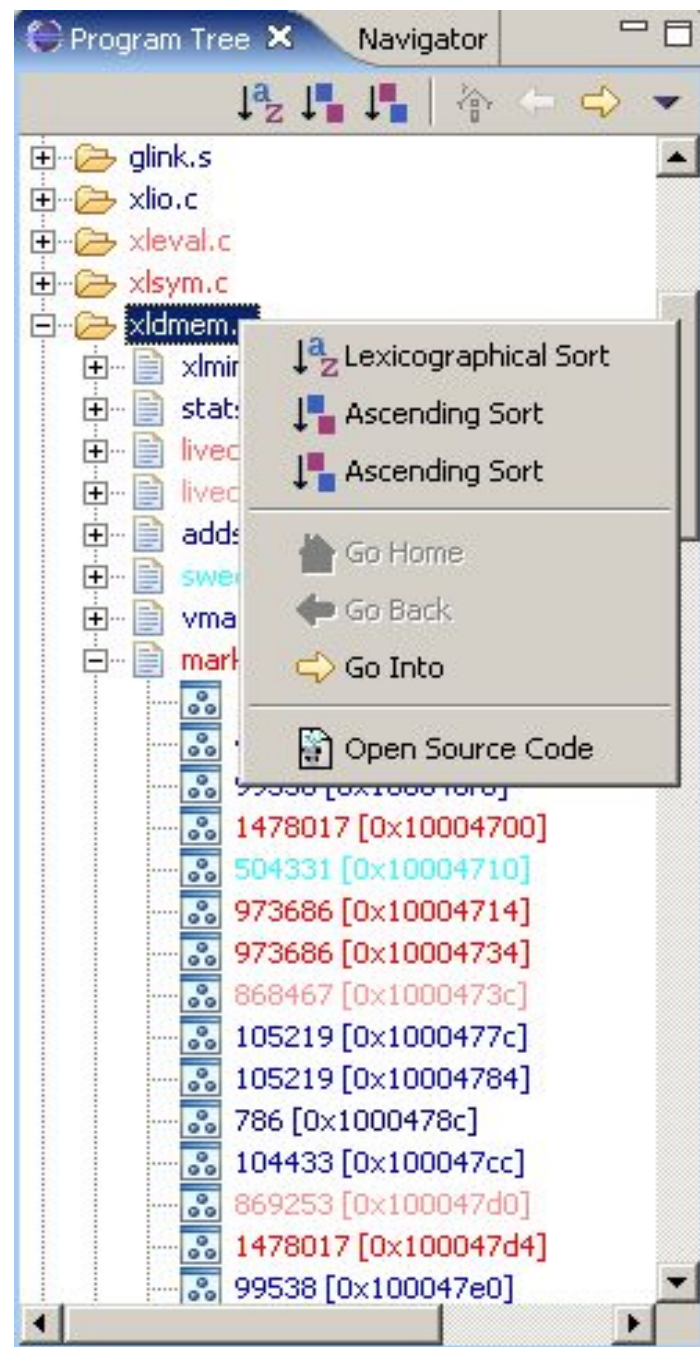
Error Log Console

Message	Plug-in	Date

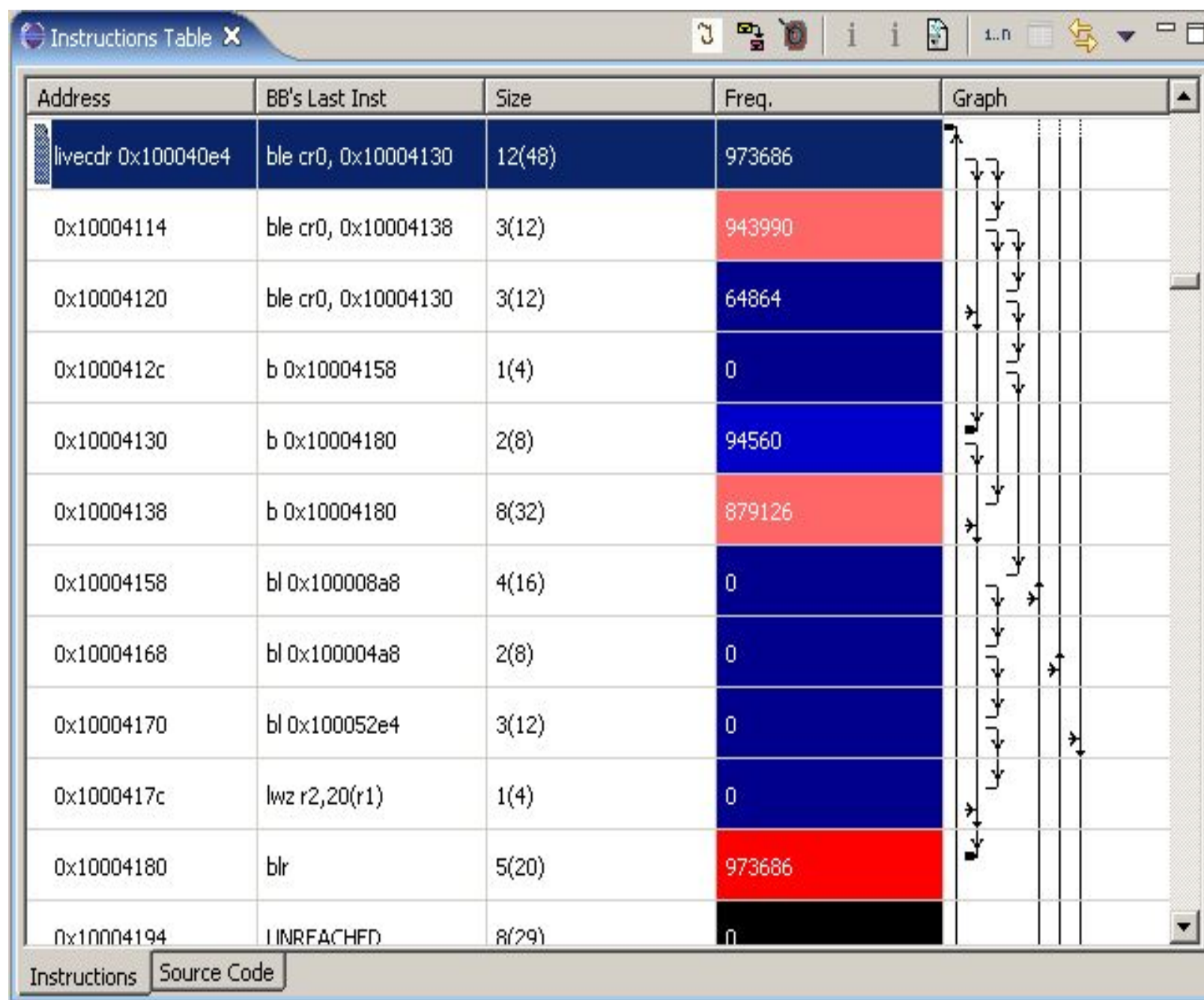
Instruction Properties

Branch Profile		Value Profile	
To Address	Count		

Code Analyzer - tree view



Code Analyzer - basic block view



Code Analyzer - branch profiling

Instructions Table

Address	Opcode	Mnemonic	Symbol	Freq.	Graph
	FUNCTION li...				
0x100040e4	0x7c0802a6	mflr r0	.livecdr	973686	
0x100040e8		lw r31,-4(r1)		973686	
0x100040ec		w r0,8(r1)		973686	
0x100040f0		wu r1,-80(r1)		973686	
0x100040f4		z r31,112(r2)	-> 0x20007f84	973686	
0x100040f8		w r3,104(r1)		973686	
0x100040fc	0x80610068	lwz r3,104(r1)		973686	
0x10004100	0x88630000	lbz r3,0(r3)		973686	
0x10004104	0x90610044	stw r3,68(r1)		973686	
0x10004108	0x3083ffff	addic r4,r3,-1		973686	
0x1000410c	0x28040001	cmpli cr0,0x0,r4...		973686	
0x10004110	0x40810020	ble cr0, 0x1000...		973686	
0x10004114	0x3083ffffd	addic r4,r3,-3		943990	
0x10004118	0x28040001	cmpli cr0,0x0,r4...		943990	
0x1000411c	0x4081001c	ble cr0, 0x1000...		943990	
0x10004120	0x3063ffffb	addic r3,r3,-5		64864	
0x10004124	0x28030005	cmpli cr0,0x0,r3...		64864	
0x10004128	0x40810008	ble cr0, 0x1000...		64864	

Instructions Source Code

Instruction Properties

Branch Profile Value Profile Link With Table

To Address	Count
0x10004114 (Fallthru)	943990
0x10004130	29696

Code Analyzer - frequency and tprof profiling view

Perf Visual - Eclipse Platform

File Edit Navigate Search Project Perf Visual Run Tests Run Window Help

Program Tree Navigator

Label View BB: 937 inst: 8 Address: 0x10004104 Mnemonic: stw r3,68(r1) Opcode: 0x90610044 Remark:

Instructions Table

Address	Opcode	Mnemonic	Comment	Freq.	Ticks	Graph
0x100040e4	0x7c0802a6	mflr r0	.livecdr	973686	1 (0.023%)	
0x100040e8	0x93e1fffc	stw r31,-4(r1)		973686	---	
0x100040ec	0x90010008	stw r0,8(r1)		973686	---	
0x100040f0	0x9421ffb0	stwu r1,-80(...		973686	---	
0x100040f4	0x83e20070	lwz r31,112(...	-> 0x20007...	973686	---	
0x100040f8	0x90610068	stw r3,104(r1)		973686	---	
0x100040fc	0x80610068	lwz r3,104(r1)		973686	4 (0.090%)	
0x10004100	0x88630000	lbz r3,0(r3)		973686	---	
0x10004104	0x90610044	stw r3,68(r1)		973686	---	
0x10004108	0x3083ffff	addic r4,r3,-1		973686	---	
0x1000410c	0x28040001	cmpli cr0,0x0...		973686	3 (0.068%)	
0x10004110	0x40810020	ble cr0, 0x10...		973686	---	
0x10004114	0x3083fffd	addic r4,r3,-3		943990	2 (0.045%)	
0x10004118	0x28040001	cmpli cr0,0x0...		943990	---	
0x1000411c	0x4081001c	ble cr0, 0x10...		943990	---	
0x10004120	0x3063ffff	addic r3,r3,-5		64864	---	
0x10004124	0x28030005	cmpli cr0,0x0...		64864	---	
0x10004128	0x40810008	ble cr0, 0x10...		64864	---	

Program Tree

- xdsym.c
- xddmem.c
 - xlminit
 - stats
 - livecdr
 - 973686 [0x100040e4]
 - 943990 [0x10004114]
 - 64864 [0x10004120]
 - 0 [0x1000412c]
 - 94560 [0x10004130]
 - 879126 [0x10004138]
 - 0 [0x10004158]
 - 0 [0x10004168]
 - 0 [0x10004170]
 - 0 [0x1000417c]
 - 973686 [0x10004180]
 - livecar
 - addseg
 - sweep
 - vmark
 - mark
 - 142849 [0x100046d0]
 - 43311 [0x100046ec]
 - 99538 [0x100046f0]
 - 1478017 [0x10004700]
 - 504331 [0x10004710]
 - 973686 [0x10004714]
 - 973686 [0x10004734]

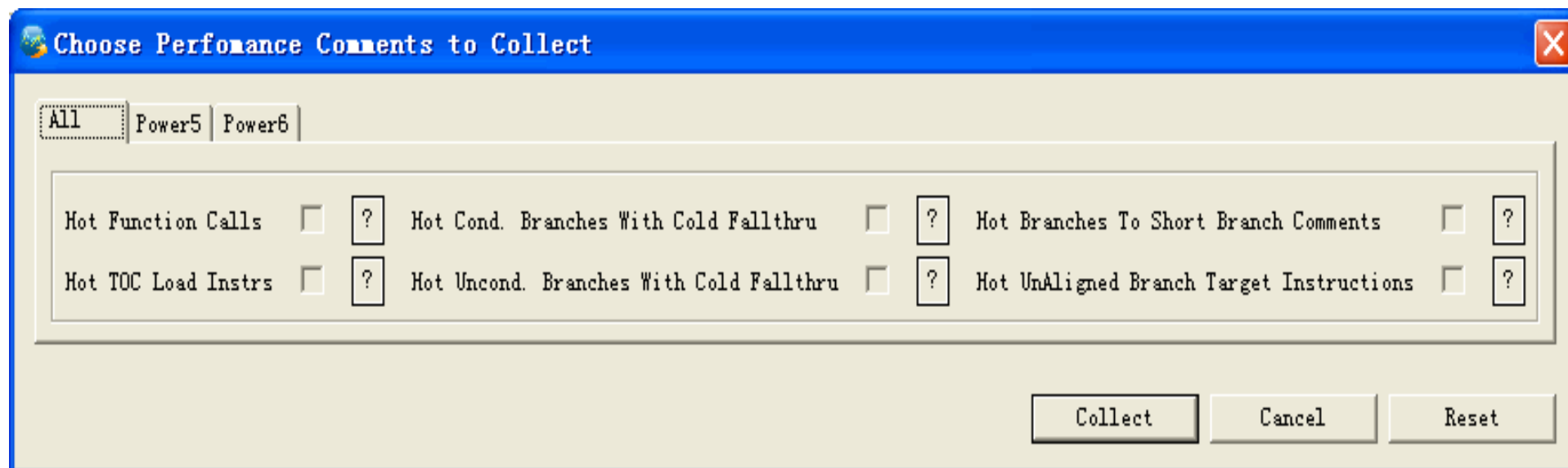
Error Log Console

Message	Plug-in	Date

Instruction Properties

Branch Profile		Value Profile	
To Address	Count		

Code Analyzer – collecting hazard information



Code Analyzer – comment and instruction grouping view

The screenshot displays the Code Analyzer interface for the function `.checkneg` in `xlmath.c`. The main window shows a table of instructions with columns for Address, Opcode, Mnemonic, Comment, Freq., and Graph. A tooltip is visible over the instruction at address `0x1000edc4`, stating: "Load instruction accesses the same address and may fall into the same dispatch group of the store instruction in 0x1000edc4".

Address	Opcode	Mnemonic	Comment	Freq.	Graph
0x1000edac	FUNCTION .check...	mflr r0	.checkneg	0	
0x1000edb0	0x93e1fffc	stwu r31,-4(r1)		0	
0x1000edb4	0x90010008	stwu r0,8(r1)		0	
0x1000edb8	0x9421ffb0	stwu r1,-80(r1)		0	
0x1000edbc	0x83e20204	lwz r31,516(r2)	-> 0x20008118	0	
0x1000edc0	0xfc200818	frsp f1,f1		0	
0x1000edc4	0xd0210068	stfs f1,104(r1)		0	
0x1000edc8	0xc0210068	lfs f1,104(r1)		0	
0x1000edcc	0xc05f0064	lfs f2,100(r31)		0	
0x1000edd0	0xfc0	Comments			
0x1000edd4	0x408	Load instruction accesses the same address and may fall into the same dispatch group of the store instruction in 0x1000edc4			
BASIC BLOCK					
0x1000edd8	0x307f00d4	addic r3,r31,212		0	
0x1000edd4	0x4bff2155	bl 0x10000f30	.xlfail	0	
BASIC BLOCK					
0x1000ede0	0x60000000	ori r0,r0,0x0		0	
BASIC BLOCK					
0x1000ede4	0x80010058	lwz r0,88(r1)		0	
0x1000ede8	0x7c0803a6	mtlr r0		0	
0x1000edec	0x30210050	addic r1,r1,80		0	
0x1000edf0	0x83e1fffc	lwz r31,-4(r1)		0	
0x1000edf4	0x4e800020	bclr 0x14,cr0,0x0...		0	
TRACE BACK TABLE					
0x1000edf8	0x00000000		0	

The CAConsole window shows several debug warnings related to profiling and function descriptors. The Instruction Properties window displays a Branch Profile table:

Slot 0	Slot 1	Slot 2	Slot 3	Slot 4
CRLogical_odd	CRLogical_e...			
FXU_0_odd	FXU_1_odd	FXU_1_even	FXU_0_even	
LSU_0_odd	LSU_1_odd	LSU_1_even	LSU_0_even	
FPU_0_odd	FPU_1_odd	FPU_1_even	FPU_0_even	Branch_any

The bottom status bar shows execution statistics: 0 1E0 5.6E4 13E4 23E4 36E4 46E4 51E4 87E4 91E4.

Code Analyzer - comments and instructions properties

The screenshot displays the Code Analyzer application window, titled "CodeAnalyzer - E:\TOOLS\CodeAnalyzer\example\li32\li32 - Visual Performance Analyzer". The interface is divided into several panes:

- Program Tree:** Shows a hierarchical view of the code being analyzed, including files like `.putoct`, `.putfloat`, `.putdec`, `.putatm`, `.putstring`, `.xlputstr`, `.xlterpri`, `.xlprint`, `glink.s`, `.sprintf`, `xlio.c`, `.xlflush`, `.xlputc`, `.xlpeek`, `.xlgetc`, `xlevel.c`, `.xlsave`, `.iskeyword`, `.xlbind`, `.evfun`, `.xlbounbound`, `.xlevlist`, `.evalhook`, `.evform`, `.xlapply`, `.xlkeval`, and `.xlevel`.
- Instruction Table:** Displays a table of instructions with columns for Address, Opcode, Mnemonic, Comment, Freq., and Graph. The current instruction is at address `0x10003318` with opcode `0x80820118` and mnemonic `lwz r4,28...`. The frequency is `365822`. The comment is `-> xltrac...`. The table shows a sequence of instructions, including `TRACE BACK...` and `UNREACHED` sections.
- Static Color...**: A window showing a frequency profile graph with a hand cursor pointing to a specific instruction.
- Comments:** A table listing comments with columns for Description, File, Function, and Address. The comments are related to load instructions accessing the same address in various files.
- Instruction Properties:** A window showing properties for the selected instruction, including Branch Profile, Value Profile, and Dispatch Info.

The bottom status bar shows the current instruction address `0` and its frequency `1E0 5.6E4 13E4 23E4 36E4 46E4 51E4 87E4 91E4`.

Code Analyzer - statistics view

CodeAnalyzer - G:\fdpr_gui\TestPrograms\li32 - Eclipse Platform

File Edit Navigate Search Project Run Tests Run Window Help

Program Tree Navigator

File: xlsym.c Function: .xlsinit

- .xlsinit
- .hash
- .findprop
- .xlreprop
- .xlputprop
- .xlgetprop
- .xlsetvalue
- .xlygetvalue
- .xlxgetvalue
- .xlgetvalue
- .xlbnd
- .xlframe
- .xlmakesym
- .xlsenter
- .xlenter
- xldmem.c
 - .xlinit
 - .stats
 - .livecdr
 - .livecar
 - .addseg
 - .sweep
 - .vmark
 - .mark

Instructions Table Browser **Statistics**

Graph Options
 Average Method: Simple Average Filter functions under (%): 10.0 Refresh

Function Heat Graph

Function Name	Execute Count (Heat)
.xlsave	~340,000
.xlevlist	~120,000
.xleval	~240,000
.xlygetvalue	~450,000
.xlxgetvalue	~210,000
.xlgetvalue	~120,000
.livecdr	~350,000
.livecar	~330,000
.sweep	~200,000
.mak	~550,000
.newnode	~100,000
.consa	~120,000

File Heat Graph Function Heat Graph **Instruction Mix Graph**

Error Log Console

Message	Plug-in	Date
Unable to find Action Set: com.ibm.hrl.analyze.actionSet1	org.eclipse.ui	09:36:25.47 2005, 20 11
Unable to find Action Set: com.ibm.hrl.analyze.actionSet2	org.eclipse.ui	09:36:25.31 2005, 20 11

Instruction Properties

Branch Profile	Value Profile	Dispatch Info
Slot 0	Slot 1	Slot 2
Slot 3	Slot 4	
CRLogical...	CRLogic...	
mitspr_0_...		
FXU_0_odd	FXU_1_...	FXU_1_...
LSU_0_odd	LSU_1_odd	LSU_1_e...
FPU_0_odd	FPU_1_...	FPU_1_...
		FPU_0_...
		Branch_...

0 8.5E4 14E4 23E4 37E4 46E4 87E4 88E4 97E4

Start | Gad Haber ... | D:\fdpr_foils | DOCUMENT... | CodeAnal... | CodeAnaly... | wincvs - [G... | 10:01

Code Analyzer – more statistics views

Statistics

Graph Options

Display Mode: Show Count Filter values under (%): 2 Show Executions Refresh

Instruction Count Graph

Instruction Opcode/Name	Execute Count
bc	1300
cmpi	600
cmpli	500
ori	1000
b	2000
stw	2900
stwu	500
lwz	4200
addi	1200
addic	1100
mtspr	500
bcl	600

File Heat Graph | Function Heat Graph | **Instruction Mix Graph** | Comments Graph

Instruction Table Statistics

Graph Options

Show graph for comment Show graph for function Filter values under (%): 0.0 Refresh

Load After Store

Functions For Comment

Functions Names	Comment count
.checkfneg	1
.checkzero	1
.tan	2
.exp	3
.expinner2	4

File Heat Graph | Function Heat Graph | Instruction Mix Graph | **Comments Graph**

Display Mode: Show Count Filter values under (%): 2 Show Count Refresh

Instruction Executions Graph

Instruction Opcode/Name	Execute Count
bc	3,200,000
cmpi	1,000,000
cmpli	1,800,000
b	1,600,000
rlwinm	1,000,000
stw	6,000,000
lbz	1,400,000
lwz	8,500,000
addi	1,000,000
addic	2,000,000

File Heat Graph | Function Heat Graph | Instruction Mix Graph | **Comments Graph**

Code Analyzer - PowerPC instructions help view

CodeAnalyzer - G:\fdpr_gui\TestPrograms\i32 - Eclipse Platform

File Edit Navigate Search Project Run Tests Run Window Help

Program Tree Navigator

File: **xlsym.c** Function: **.xlsinit**

- .xlsinit
- .hash
- .findprop
- .xlreprop
- .xlputprop
- .xlgetprop
- .xlsetvalue
- .xlygetvalue
- .lxgetvalue
- .xlgetvalue
- .xlbind
- .xlframe
- .xlmakesym
- .xlsenter
- .xlenter
- xldmem.c
- .xlinit
- .stats
- .livecdr
- .livecar
- .addseg
- .sweep
- .vmark
- .mark

Instructions Table | Browser | Statistics

Back | Forward | Stop | Refresh | Go

stw or st (Store) Instruction

Purpose

Stores a word of data from a general-purpose register into a specified location in memory.

Syntax

Bits	Value
0-5	36
6-10	RS
11-15	RA
16-31	D

PowerPC

Done

Instruction Properties

Slot 0	Slot 1	Slot 2	Slot 3	Slot 4
CRLogical...	CRLogic...			
FXU_0_odd	FXU_divi...			
LSU_0_odd	LSU_1_...	FXU_1_...	FXU_0_...	
FPU_0_odd	FPU_1_...	LSU_1_e...	LSU_0_e...	

Error Log Console

Message	Plug-in	Date
Unable to find Action Set: com.ibm.hrl.analyze.actionSet1	org.eclipse.ui	09:36:25.47 2005, 20 31
Unable to find Action Set: com.ibm.hrl.analyze.actionSet2	org.eclipse.ui	09:36:25.31 2005, 20 31

0 8.5E4 14E4 23E4 37E4 46E4 87E4 88E4 97E4

Code Analyzer - instructions, source & comments view

The screenshot displays the CodeAnalyzer application within the Eclipse IDE. The interface is divided into several panes:

- Program Tree:** Shows the project structure for `xlsvm.c`, including files like `.xlsave`, `.xlkeyword`, `.xlbind`, `.xlfun`, `.xlunbound`, `.xlolist`, `.evalhook`, `.evform`, `.xlapply`, `.xlxeval`, `.xlval`, `.xlinit`, `.hash`, `.findprop`, `.xlremprop`, `.xlputprop`, `.xlgetprop`, `.xlsetvalue`, and `.xlygetvalue`.
- Instructions Table:** A table showing assembly instructions for the selected function `.xlgetvalue`. The table has columns for Address, Opcode, Mnemonic, Comment, F..., and G... The instruction at address `0x100039f4` is highlighted: `bl 0x100039f4`.
- Source Code:** Displays the C source code for `xlsvm.c`. The function `xlgetvalue` is visible, showing a loop that calls `xlunbound` and returns a value. Comments describe the function's purpose: `/* xlgetvalue - get the value of a NODE *xlgetvalue (NODE *sym) */`.
- CAConsole:** A table listing system events and warnings. The table has columns for Description, File, Function, and Address.

Description	File	Function	Address
Hot TOC Load Instruction: 0x10003948	xlsvm.c	.xlygetvalue	0x10003948
Hot TOC Load Instruction: 0x10003a04	xlsvm.c	.xlxgetvalue	0x10003a04
Hot Function Call to: 0x10006548	xlsvm.c	.xlxgetvalue	0x10003a20
Hot TOC Load Instruction: 0x10003a3c	xlsvm.c	.xlxgetvalue	0x10003a3c
Hot Function Call to: 0x100039f4	xlsvm.c	.xlgetvalue	0x10003b04
Hot TOC Load Instruction: 0x10003b0c	xlsvm.c	.xlgetvalue	0x10003b0c
Hot TOC Load Instruction: 0x100040f4	xldmem.c	.livecdr	0x100040f4
Hot TOC Load Instruction: 0x100041c4	xldmem.c	.livecar	0x100041c4
- Instruction Properties:** A pane for analyzing the selected instruction, with tabs for Branch Profile, Value Profile, and Dispatch Info. It includes a table for To Address and Count.

The bottom of the screen shows the Windows taskbar with the Start button, system tray icons, and the time 16:47.

Invoking Code Analyzer from Profile Analyzer

The screenshot shows the Visual Performance Navigator interface with the following components:

- Navigator:** Shows LOCALHOST (Local).
- Profiling Resources:** Lists modules with their ticks and time share percentages. A context menu is open over the 'SpecDb' module, with 'Open in CodeAnalyzer' selected.
- Samples Distribut...:** A 3D bar chart showing the distribution of ticks across modules. 'SHELLSRT' is the largest bar at 68.89%. Other modules include EQDB, GETMAI, INITE, NEWEN, SETINE, ELEMEN, INITO, EBC2AS, and Other.
- Disassembly/Offse...:** A table showing disassembly instructions for the 'SHELLSRT' module. The selected instruction is at offset 0x00000252.

Offset	Disassembly	Ticks	AGI...	OSC	CachMs
0x00000234	ICM GPR5,B'0011',0(...	5245	2771	19.0	87.0
0x00000238	LG GPR14,16(,GPR14)	18740	5506	33.0	424
0x0000023E	LGR GPR7,GPR14	2005	6		1
0x00000242	XGR GPR3,GPR3	650	2		
0x00000246	LA GPR2,2(,GPR7)	12496	2	4	
0x0000024A	ICM GPR3,B'0011',0(...	3517	1	3	41.0
0x0000024E	CLCLE GPR2,GPR4,0	11915	1	2	
0x00000252	BERC *+124	177118	11924	13329	108
0x00000256	LTGR GPR3,GPR3	4703	2	444	
0x0000025A	BNPRC *+36	1985	2	5	
0x0000025E	LTGR GPR5,GPR5	470			
0x00000262	BNPRC *+28	2086	594	10.0	
0x00000266	LLGC CPR7,0(,CPR2)	66	3	5	

Code Analyzer – combined view with disassembly

CodeAnalyzer - E:\TOOLS\ppc\sort - Eclipse Platform

File Edit Navigate Search Project Run Window Help

CodeAnalyzer
Profile An...
Java

Instructions Table

File: sort.c Function: .xxxxsort Index: 57 BB: addr:

Address	Opcode	Mnemonic	Comment	Freq.	Graph
0x100...	0x9061...	stw r3...		0	
0x100...	0x3860...	li r3,0	.bf	0	
0x100...	0x9061...	stw r3...		0	
0x100...	0x8061...	lwz r3...		0	
0x100...	0x2c03...	cmpi c...		0	
0x100...	0x4080...	bge cr...		0	
0x100...	0x3860...	li r3,0		0	
0x100...	0x9061...	stw r3...		0	
0x100...	0x8061...	lwz r3...		0	
0x100...	0x2c03...	cmpi c...		0	
0x100...	0x4080...	bge cr...		0	
0x100...	0x8061...	lwz r3...		0	
0x100...	0x8081...	lwz r4...		0	
0x100...	0x5484...	rlwinm...		0	
0x100...	0x7c63...	lwzx r...		0	
0x100...	0x80a1...	lwz r5...		0	
0x100...	0x80c1...	lwz r6...		0	
0x100...	0x38c6...	addi r...		0	
0x100...	0x54c6...	rlwinm...		0	
0x100...	0x7ca5...	lwzx r...		0	
0x100...	0x7c03...	cmp cr...		0	
0x100...	0x4081...	ble cr...		0	

Disassembly/Offsets

Offsets for: .xxxxsort (/sort)

Offset	Bytes	Disassembly
0x00000000	9421ffb0	stwu r1,-80(r1)
0x00000004	90610068	stw r3,+104(r1)
0x00000008	38600000	li r3,+0
0x0000000c	90610044	stw r3,+68(r1)
0x00000010	80610044	lwz r3,+68(r1)
0x00000014	2c03000a	cmpi cr0,0,r3,+10
0x00000018	4080010c	bc B0_IF_NOT,CRO_L
0x0000001c	38600000	li r3,+0
0x00000020	90610040	stw r3,+64(r1)
0x00000024	80610040	lwz r3,+64(r1)
0x00000028	2c03000a	cmpi cr0,0,r3,+10
0x0000002c	40800100	bc B0_IF_NOT,CRO_L
0x00000030	80610068	lwz r3,+104(r1)
0x00000034	80810040	lwz r4,+64(r1)
0x00000038	5484103a	rlwinm r4,r4,2,0,2
0x0000003c	7c63202e	lwzx r3,r3,r4
0x00000040	80a10068	lwz r5,+104(r1)
0x00000044	80c10040	lwz r6,+64(r1)
0x00000048	38c60001	addi r6,r6,+1
0x0000004c	54c6103a	rlwinm r6,r6,2,0,2
0x00000050	7ca5302e	lwzx r5,r5,r6
0x00000054	7c032800	cmp cr0,0,r3,r5
0x00000058	408100bc	bc B0_IF_NOT,CRO_G
0x0000005c	80610068	lwz r3,+104(r1)
0x00000060	7c63202e	lwzx r3,r3,r4
0x00000064	90610048	stw r3,+72(r1)

cmpi - Compare Immediate D-form

Pipeline Analyzer – scroll mode

Pipeline Perspective - scroll-mode - Eclipse SDE

File Edit Navigate Search Project Run Pipeline Window Help

Pipe View X

General

Cycles: 1354
Lines: 2357
Divider: 1

Offset

Base: -1:-1
Cursor: 7:1
Offset: 8:2

Event Message

D:instruction in one stage of decode

scroll-mode

0 10 20

```

F V B D D D D D M I 2 E f C D
F V B D D D D D M I 0 f C
F V B D D D D D M I 3 E f C
F V B D D D D D M I 2 E f C D
F V B D D D D D M I 0 f C
F V B D D D D D M I 1 E f C
F V B D D D D D M u I 1 E f C
F V B D D D D D M u s I 0 E f C
F V B D D D D D M I 0 E f C
F V B D D D D D M s s s s I 6 f C
F V B D D D D D M s s s I 2 E f C
F V B D D D D D M s s s I 3 E f C
F V B D D D D D M s s s s I 1 E f C
F V B D D D D D M u s s s I 0 E f C
F V B D D D D D M u I 2 E f
F V B D D D D D M s s s u s I 0 f
F V B D D D D D M s s s s I 6 f
F V B D D D D D M u I 2 E f
F V B D D D D D M s s s I 3 E f
F V B D D D D D M I 2 E f
F V B D D D D D M u I 3 E f
F V B D D D D D M u u I 0 E f
F V B D D D D D M I 6 f
F V B D D D D D M s s s I 2
F V B D D D D D M s I 1 E f
F V B D D D D D M s u s I 1
F V B D D D D D M u u I 0 E
  
```

image

```

stw R0, 8(R1)
stw R0, 8(R1)
lwz R0, 0(R2)
stwu R1, -80(R1)
stwu R1, -80(R1)
stwu R1, -80(R1)
ori R30, R3, 0x0
cmpi CR0, 0, R0, 0
addi R31, R0, 0
bc 4, 2, .+76
lwz R5, 0(R30)
lwz R4, 4(R30)
addic. R4, R4, -1
addi R0, R5, 1
stw R4, 4(R30)
stw R4, 4(R30)
bc 4, 0, .+32
stw R0, 0(R30)
stw R0, 0(R30)
lbz R3, 0(R5)
law R30, 72(R1)
law R30, 72(R1)
addi R1, R1, 80
blr
lwz R2, 20(R1)
ori R4, R30, 0x0
addi R5, R1, 64
cmpi CR0, 0, R29, 0
  
```


Pipeline Analyzer – scroll mode navigation

The screenshot displays the Pipeline Analyzer interface in scroll mode. The top window, titled "Pipeline Perspective - scroll-mode - Eclipse SDK", shows a "Pipe View" with the following statistics:

- General: Cycles: 1355, Lines: 2357, Divider: 1
- Offset: Base: -1:-1, Cursor: 5:4, Offset: 6:5

The "Event Message" section displays: "B:branch prediction occurred". A red arrow points from this message to the "scroll-mode" window below.

The "scroll-mode" window shows a pipeline view with instructions and their addresses. The instruction at address d005e880 is highlighted in red, corresponding to the branch prediction event.

Inst Addr	Data Addr
d005e878	2ff3fb48
d005e878	d005e878
d005e87c	f0076088
d005e880	d005e880
d005e880	2ff3faf0
d005e880	d005e880
d005e884	d005e884
d005e888	d005e888
d005e88c	d005e88c
d005e890	d005e890
d005e894	f00b70d0

Pipeline Analyzer – scroll mode navigation

The screenshot shows a window titled "scroll-mode" with a toolbar at the top. The toolbar contains several icons: a magnifying glass with a plus sign, a magnifying glass with a minus sign, a magnifying glass with a downward arrow, a grid icon, a scroll mode icon (highlighted with a red circle), and a refresh icon. The main display area shows a list of instructions in a scrollable view. The instructions are displayed in a grid format with columns for instruction type (F, V, B, D, M, I, E, f, C, D, S) and address. A red horizontal line highlights the instruction at address 0005e884: `ori R30, R3, 0x0`. A tooltip is displayed over this instruction, showing the following details:

- read_regs: gpr3
- id_address:
- write_regs: gpr30
- mnemonic:
- hi_address: d005e884
- image: ori R30, R3, 0x0
- arch_number: 4
- hiop_number: 7

The scroll mode icon in the toolbar is highlighted with a red circle, indicating the current navigation mode. The scroll mode icon is a speech bubble with a downward arrow.

Pipeline Analyzer – scroll mode navigation

Preferences

- Trace
- Event
- Display
- Side Bar**

Side Bar

```

arch_number (Arch Id)
image (Mnemonic)
iop_number (Iop Id)
mnemonic (Mnemonic)
read_regs (Read Regs)
write_regs (Write Regs)
    
```

scroll-mode X

	Data Addr	Inst Addr
0		d005e878
	2ff3fb48	d005e878
	f0076088	d005e87c
		d005e880
	2ff3faf0	d005e880
		d005e880
		d005e884
		d005e888
		d005e88c
		d005e890
	f00b70d0	d005e894

Pipeline Analyzer – resource editor

The screenshot displays the Eclipse IDE interface for the Pipeline Analyzer resource editor. The main window is titled "Pipeline Perspective - nimbus.resource_mode - Eclipse SDK". The menu bar includes File, Edit, Refactor, Navigate, Search, Project, Pipeline, Run, Window, and Help. The toolbar contains various icons for file operations and navigation.

The interface is divided into several panes:

- Pipe View (Left Pane):** This pane is highlighted with a red border. It contains a "General" section with the following data:
 - Cycles: 508
 - Lines: 877
 - Divider: 1The "Offset" section shows:
 - Base: 96:56
 - Cursor: 69:71
 - Offset: -27:15Below these sections is an "Event Message" field. At the bottom of the pane is a detailed pipeline diagram showing resource usage over time, with a mouse cursor pointing to a specific resource block.
- Main Editor (Right Pane):** This pane displays a resource usage diagram for "nimbus.resource_mode". The diagram shows horizontal bars representing resource usage, with a vertical red line indicating the current cursor position. The diagram is color-coded with red and pink bars. The vertical axis is labeled with values 80, 90, and 100. The horizontal axis is labeled with values 80 and 90.
- Resource Name List (Far Right):** A list of resource names is displayed, including:
 - NSM_id[1][0][
 - NSM_id[1][1][
 - NSM_id[1][2][

Pipeline Analyzer – resource editor

The screenshot displays the Eclipse IDE interface for the Pipeline Analyzer resource editor. The main window is titled "Pipeline Perspective - nimbus_resource_mode - Eclipse SDK". The menu bar includes File, Edit, Refactor, Navigate, Search, Project, Pipeline, Run, Window, and Help. The toolbar contains various icons for file operations and navigation.

The "Pipe View" is the primary display area, showing a pipeline graph with a large pink rectangular region. To the left of the graph, there are two panels:

- General:** Cycles: 508, Lines: 877, Divider: 1
- Offset:** Base: 56:387, Cursor: 63:57, Offset: 7:-330

Below these panels is the "Event Message" section, which displays:

```
Dprefetch(type:4, value:336)
Dprefetch(type:4, value:404)
```

The bottom window, titled "nimbus_resource_mode", shows a resource usage graph. The x-axis represents time, with markers at 50, 60, and 70. The y-axis represents resource usage, with a marker at 60. The graph consists of horizontal bars in red and pink, representing different resource usage patterns over time. A tooltip is visible over the graph, showing:

```
source: NSM_id[0][0][1]
message: Dprefetch(type:4, value:336)
Dprefetch(type:4, value:404)
```

To the right of the graph is a "Resource Name" list, which includes:

- DPQ_id[1][6]
- DPQ_id[1][7]
- NSM_id[0][0][0]
- NSM_id[0][0][1]
- NSM_id[0][0][2]
- NSM_id[0][0][3]
- NSM_id[0][0][4]
- NSM_id[0][0][5]
- NSM_id[0][0][6]
- NSM_id[0][0][7]
- NSM_id[0][1][0]

Pipeline Analyzer – tie cycle control

The screenshot displays the Eclipse SDK interface for the Pipeline Analyzer. The main window is titled "Pipeline Perspective - scroll-mode - Eclipse SDK". The menu bar includes File, Edit, Refactor, Navigate, Search, Project, Pipeline, Run, Window, and Help. The toolbar contains various icons for file operations and navigation.

The "Pipe View" panel shows the following statistics:

- General: Cycles: 1355, Lines: 2357
- Offset: Base: 8:29, Cursor: 10:32

The "scroll-mode" view shows a pipeline diagram with instructions and their stages. The instructions are:

- Instruction 1: F B D D u O
- Instruction 2: F B D D M I f
- Instruction 3: F B D D M I
- Instruction 4: F V D D M u
- Instruction 5: F V D D M u

The "nimbus.resource_mode" view shows a resource usage diagram with yellow bars representing resource usage over time. The resource names are:

- DPT_id[0] [3]
- DPT_id[0] [4]
- DPT_id[0] [5]
- DPT_id[0] [6]
- DPT_id[0] [7]

The "Data Addr" and "Inst Addr" columns in the scroll-mode view show the following addresses:

Data Addr	Inst Addr
	10003694
	10003698
	[1000369c
	100036a0
	100036a4

Thank you

charts to be available at:

<http://www.ibm.com/developerworks/blogs/page/Systemptechuniv>
