

Driving The Gooda Visualizer

point browser at gooda-visualizer/index.html

The screenshot shows a web browser window displaying the Generic Optimization Data Analyzer (GOODA) interface. The browser's address bar shows the file path: `file:///home/levinth/demo/gooda-visualizer/index.html#`. The page title is "Generic Optimization Data Analyzer GOODA".

On the left side, there is a "Reports" sidebar with a tree view containing the following items:

- Sample
- atlas_cg
- calchain
- malloc_test
- geant
- atlas_reco_pile
- atlas_reco

The main content area features a "Welcome" message and a large hierarchical diagram illustrating the decomposition of Total Cycles:

- Total Cycles
 - Halted Cycles
 - Unhalted Cycles
 - Unstalled Cycles
 - Port Saturation
 - Function Call Overhead
 - Instruction Serialization
 - Exception Handling
 - Stalled Cycles
 - Load Latency
 - Bandwidth Saturation
 - Instruction Starvation
 - Instruction Latency
 - Store Resource Saturation
 - Multithread Collisions
 - Branch Misprediction

Below the diagram, a paragraph explains the tool's purpose:

To address this issue, a generic decomposition of how a microprocessor is using the consumed cycles allows code developers to quickly understand which of the myriad of microarchitectural complexities they are battling, without requiring a detailed knowledge of the microarchitecture. When this approach is intrinsically integrated into a performance data analysis tool, it enables software developers to take advantage of the microarchitectural methodology that has only been available to experts. The Generic Optimization Data Analyzer (GOODA) project integrates this expertise into a profiling tool in order to lower the required expertise of the user and, being designed from the ground up with large-scale object-oriented applications in mind, it will be particularly useful for large HENP codebases.

The bottom of the browser window shows a taskbar with several open applications: "RecoTRFW...tar.bz2", "RecoTRFW...tar.bz2", and "ParallelG4.tar.bz2". A "Show all downloads" button is visible in the bottom right corner.

expand the Load_latency metric to display its components

The screenshot displays the Generic Optimization Data Analyzer GUI. The top section shows a summary table for the 'cc1' process, with columns for various metrics such as 'unmalted_core_cycles', 'wups_retired:stall_cycles', 'instruction_retired:unmalted_core_cycles', 'load_latency', 'wups_uncore...cache_hit', 'wups_uncore...emote_dram', 'wups_load_f...shared_bit', 'wups_uncore...r_ill_miss', 'wups_uncore...emote_hit', 'wups_load_retired:l2_hit', 'wups_load_retired:hit_l1f', 'wups_load_hit_pre', 'instruction_starvation', 'bandwidth_saturated', 'branch_misprediction', 'store_resources_saturated', and 'instruction'. The 'load_latency' column is highlighted in yellow.

The bottom section provides a detailed view of the 'Load_latency' metric, showing a table of function names, offsets, lengths, and modules. The 'Load_latency' column is expanded to show its components: 'unmalted_core_cycles', 'wups_retired:stall_cycles', 'wups_uncore...cache_hit', 'wups_uncore...emote_dram', 'wups_load_f...shared_bit', 'wups_uncore...r_ill_miss', 'wups_uncore...emote_hit', 'wups_load_retired:l2_hit', 'wups_load_retired:hit_l1f', 'wups_load_hit_pre', 'instruction_starvation', 'bandwidth_saturated', 'branch_misprediction', 'store_resources_saturated', 'instruction_latency', and 'exception_handling'. The 'Load_latency' column is highlighted in yellow.

Function Name	Offset	Length	Module	Process	unmalted_core_cycles	wups_retired:stall_cycles	wups_uncore...cache_hit	wups_uncore...emote_dram	wups_load_f...shared_bit	wups_uncore...r_ill_miss	wups_uncore...emote_hit	wups_load_retired:l2_hit	wups_load_retired:hit_l1f	wups_load_hit_pre	instruction_starvation	bandwidth_saturated	branch_misprediction	store_resources_saturated	instruction_latency	exception_handling		
_cpp_lex_direct	0xef09f1	0x1151	cc1	cc1	12508 (100%)	9693 (77%)	9093	11021	1103	9%	3085	24%	21	0%	926	7%	62	0%	41	0%		
record_reg_classes	0x7458ac	0x1f7a	cc1	cc1	10387 (100%)	6352 (61%)	9323	13577	298	2%	442	4%			370	3%		10	0%	10	0%	
ggc_internal_alloc_stat	0x530df8	0x31c	cc1	cc1	8052 (100%)	5914 (73%)	6457	7923	1861	23%	2129	26%			411	5%	51	0%		10	0%	
ht_lookup_with_hash	0xefce74	0x48a	cc1	cc1	9091 (100%)	8959 (98%)	2083	4835	8371	92%	2100	23%	10	0%	309	3%	10	0%		31	0%	
lex_identifier	0xeeec01	0x295	cc1	cc1	6059 (100%)	4301 (69%)	7157	8652	216	3%	823	11%	21	0%	206	3%				21	0%	
acc_char_cmp	0xeeed9f	0x59	cc1	cc1	5656 (100%)	2596 (45%)	6645	10009			62	1%			10	0%	154	2%	10	0%	10	0%
find_reloads	0x80d60e	0x63c5	cc1	cc1	6222 (100%)	4589 (73%)	4837	6999	93	1%	1913	30%	10	0%	247	3%				21	0%	
cpp_get_token_1	0xef82cf	0x446	cc1	cc1	5802 (100%)	4644 (78%)	4717	5723	584	8%	1584	26%	10	0%	452	7%	226	3%				
extract_insn	0x7ea58f	0x4e1	cc1	cc1	5327 (100%)	3034 (56%)	5476	6429	566	10%	987	18%			298	5%	31	0%		21	0%	
preprocess_constraints	0x7ea770	0x575	cc1	cc1	4905 (100%)	2793 (56%)	3753	5871	51	1%	113	2%			62	1%				41	0%	
search_line_acc_char	0xeed446	0x130	cc1	cc1	4052 (100%)	2234 (55%)	3873	7934	41	1%	93	2%			10	0%	154	3%		21	0%	
grogdeclarator	0x496bbe	0x31e2	cc1	cc1	4432 (100%)	3965 (89%)	1630	2611	247	5%	4391	99%	165	3%	545	12%	10	0%		10	0%	
c_parser_peek_token	0x4c6339	0x3c	cc1	cc1	3589 (100%)	2311 (64%)	2653	4286	72	2%	1429	39%	10	0%	309	8%	51	1%		10	0%	
df_ref_create_structure	0x55ee5	0x291	cc1	cc1	3108 (100%)	1884 (59%)	3156	5016	62	1%	41	1%					206	6%				
get_attr_enabled	0x8db61b	0x1750	cc1	cc1	3239 (100%)	1599 (49%)	4052	6133			154	4%			10	0%				21	0%	
linemap_position_for_co	0xef377e	0x07	cc1	cc1	2921 (100%)	1522 (52%)	3702	4902	82	2%	566	19%			31	1%	31	1%	10	0%	21	0%
ggc_round_alloc_size_1	0x530df1	0x80	cc1	cc1	3303 (100%)	2201 (65%)	2192	4252	617	18%	915	27%			41	1%	103	3%		21	0%	
c_lex_one_token	0x4c556f	0x3ca	cc1	cc1	3311 (100%)	2377 (71%)	2354	3295	401	12%	1360	41%			301	11%				21	0%	
bitmap_find_bit	0x539f57	0x12b	cc1	cc1	2355 (100%)	1300 (58%)	3327	4620	31	1%	144	6%			21	0%	10	0%				
ix86_decompose_address	0xaa9103	0x759	cc1	cc1	2540 (100%)	1073 (73%)	2243	3340	113	4%	1142	44%	41	1%	93	3%	62	2%	10	0%	10	0%
_cpp_lex_token	0xef96c9	0x21d	cc1	cc1	2869 (100%)	1804 (65%)	2098	2804	82	2%	967	33%			278	9%				10	0%	
reserv_sets_are_interse	0x406c77	0x23e	genautomata	genautomata	2201 (100%)	1205 (54%)	2678	3944	31	1%	10	0%			62	2%				10	0%	
_cpp_clean_line	0xeed07c	0x4c9	cc1	cc1	2736 (100%)	2048 (74%)	1817	2827			751	27%	10	0%	72	2%	31	1%				
htab_find_slot_with_hash	0xf28869	0x230	cc1	cc1	2972 (100%)	2453 (82%)	1280	2166	1409	47%	720	24%			247	8%	10	0%		10	0%	
find_costs_and_classes	0x7490bd	0x10f4	cc1	cc1	2232 (100%)	1435 (64%)	2405	2975	123	5%	51	2%			62	2%				10	0%	
pool_alloc	0x530ff1	0x15f	cc1	cc1	2448 (100%)	1719 (70%)	2090	2725	51	2%	165	6%					82	3%				
c_lex_with_flags	0x516c24	0x6c2	cc1	cc1	2797 (100%)	1960 (70%)	1254	2417	298	10%	617	22%			216	7%	247	8%		21	0%	

Expand the sub-component: Dtlb_latency

Generic Optimization Data Analyzer GUI

Reports

- Sample
 - atlas_cg
 - calchain
 - malloc_test
 - geant
 - atlas_reco_jl
 - atlas_reco

Sample Hotspots

Cycles Samples

process path	module path	unmalted_core_cycles	wups_retired:stall_cycles	instruction_retired	wups_retired:any	load_latency	mem_uncore...cache_hit	mem_uncore...emote_dram	dtlb_load...shared_hit	dtlb_latency	dtlb_load...s:stlb_hit	dtlb_load...completed	dtlb_load...dlk_cycles	mem_uncore...r:llc_miss	emote_hit	mem_load_retired:l3_hit	mem_load_retired:l1d	mem_load_retired:l1g_hit	instruction_starvation	bandwidth
cc1		512719 (100%)	371649 (72%)	369217 534338	64010 (16%)	21667 (4%)	3759 (0%)	32227 (6%)	21967 (3%)	9699 (1%)	1468 (0%)	10833 (1%)	2646 (0%)	20 (0%)	6967 (1%)	1057 173243 3835	216550 (37%)	9646 (1%)		
	/data/home/vittorio/install...	463747 (100%)	331638 (71%)	342867 496753	68605 (14%)	18958 (4%)	2837 (0%)	25714 (5%)	15447 (3%)	7822 (1%)	865 (0%)	6772 (1%)	1314 (0%)		4374 (0%)	732 146248	177450 (38%)	6345 (1%)		
	/lib64/libc-2.12.so	26759 (100%)	20218 (75%)	18596 23037	7415 (27%)	1083 (4%)	162 (0%)	3601 (13%)	1954 (7%)	886 (3%)	108 (0%)	976 (3%)	155 (0%)		474 (1%)	70 4511 167	12084 (45%)	586 (2%)		
	/vmlinux	19375 (100%)	17688 (91%)	5911 11696	7353 (37%)	1726 (8%)	760 (3%)	2628 (13%)	1913 (9%)	109 (0%)	216 (1%)	1603 (8%)	70 (0%)	20 (0%)	270 (1%)	46 1784 2553	13225 (68%)	915 (4%)		
	/lib64/libc-2.12.so	2499 (100%)	1709 (68%)	1740 1881	381 (15%)	67 (2%)		228 (9%)	72 (2%)	42 (1%)					36 (1%)			740 (29%)		
	/lib/modules/3.3.0-rc7/ker...	195 (100%)	186 (95%)	60 114	93 (47%)	13 (6%)									13 (6%)			237 (121%)	21 (10%)	
	/lib/modules/3.3.0-rc7/ker...	51 (100%)	77 (150%)	26 23														113 (221%)		
	/lib64/ld-2.12.so	93 (100%)	120 (129%)	17 34	31 (33%)	26 (27%)														
	/lib/modules/3.3.0-rc7/ker...			11																
	/lib64/ld-2.12.so			11																
	/data/home/vittorio/install...																			

Cycles Samples

function name	offset	length	module	process	unmalted_core_cycles	wups_retired:stall_cycles	instruction_retired	wups_retired:any	load_latency	instruction_starvation	bandwidth_saturated	branch_misprediction	store_resources_saturated	instruction_latency	exception_handling
@_cpp_lex_direct	0xf09f1	0x1151	cc1	cc1	12588 (100%)	9693 (77%)	9993 11821	1183 (9%)	3085 (24%)	21 (0%)	926 (7%)	62 (0%)		41 (0%)	
@record_reg_classes	0x7458c	0x117a	cc1	cc1	10387 (100%)	632 (61%)	9323 13577	298 (2%)	442 (4%)		378 (3%)		10 (0%)	10 (0%)	
@ggc_internal_alloc_stat	0x5380f8	0x31c	cc1	cc1	8052 (100%)	5914 (73%)	6457 7923	1861 (23%)	2129 (26%)		411 (5%)	51 (0%)		10 (0%)	
@ht_lookup_with_hash	0xfce74	0x40a	cc1	cc1	9091 (100%)	8959 (98%)	2883 4035	8371 (92%)	2180 (23%)	10 (0%)	309 (3%)	10 (0%)		31 (0%)	
@lex_identifier	0xeec01	0x295	cc1	cc1	6859 (100%)	4381 (63%)	7157 8652	216 (3%)	823 (11%)	21 (0%)	206 (3%)			21 (0%)	
@acc_char_cmp	0xeed09f	0x59	cc1	cc1	5656 (100%)	2596 (45%)	6645 10009		62 (1%)		10 (0%)	154 (2%)	10 (0%)	10 (0%)	
@find_reloads	0x80609e	0x63c5	cc1	cc1	6222 (100%)	4589 (73%)	4037 6999	93 (1%)	1913 (30%)	10 (0%)	247 (3%)			21 (0%)	
@cpp_get_token_l	0xf82cf	0x446	cc1	cc1	5882 (100%)	4644 (78%)	4717 5723	584 (8%)	1584 (26%)	10 (0%)	452 (7%)	226 (3%)		21 (0%)	
@extract_insn	0x7ea5f	0x4e1	cc1	cc1	5327 (100%)	3034 (56%)	5476 6429	566 (10%)	987 (18%)		298 (5%)	31 (0%)		21 (0%)	
@preprocess_constraints	0x7ea70	0x575	cc1	cc1	4905 (100%)	2793 (56%)	3753 5871	51 (1%)	113 (2%)		62 (1%)			41 (0%)	
@search_line_acc_char	0xeed446	0x130	cc1	cc1	4052 (100%)	2234 (55%)	3073 7934	41 (1%)	93 (2%)		10 (0%)	154 (3%)		21 (0%)	
@gproclerator	0x496b0e	0x31e2	cc1	cc1	4432 (100%)	3965 (89%)	1638 2611	247 (5%)	4391 (98%)	165 (3%)	545 (12%)	10 (0%)		10 (0%)	
@c_parser_peek_token	0x4c6339	0x3c	cc1	cc1	3589 (100%)	2311 (64%)	2653 4286	72 (2%)	1429 (39%)	10 (0%)	309 (8%)	51 (1%)		10 (0%)	
@df_ref_create_structure	0x5d5ee5	0x291	cc1	cc1	3188 (100%)	1804 (56%)	3156 5016	62 (1%)	41 (1%)			206 (6%)			
@get_attr_enabled	0xb0b61b	0x1750	cc1	cc1	3239 (100%)	1599 (49%)	4052 6133		154 (4%)		10 (0%)			21 (0%)	
@linemap_position_for_co...	0xf377e	0x07	cc1	cc1	2921 (100%)	1522 (52%)	3782 4902	82 (2%)	566 (19%)		31 (1%)	31 (1%)	10 (0%)	21 (0%)	
@ggc_round_alloc_size_l	0x5380f1	0x80d	cc1	cc1	3383 (100%)	2201 (65%)	2192 4252	617 (18%)	915 (27%)		41 (1%)	103 (3%)		21 (0%)	
@c_lex_one_token	0x4c5f6f	0x3ca	cc1	cc1	3311 (100%)	2377 (71%)	2354 3295	401 (12%)	1368 (41%)		381 (11%)			21 (0%)	
@bitmap_find_bit	0x539f57	0x12b	cc1	cc1	2355 (100%)	1380 (58%)	3327 4628	31 (1%)	144 (6%)		21 (0%)	10 (0%)			
@x86_decompose_address	0xaa9103	0x759	cc1	cc1	2540 (100%)	1073 (42%)	2243 3340	113 (4%)	1142 (44%)	41 (1%)	93 (3%)	62 (2%)	10 (0%)	10 (0%)	
@_cpp_lex_token	0xf096c9	0x21d	cc1	cc1	2869 (100%)	1804 (63%)	2098 2804	82 (2%)	967 (33%)		278 (9%)			10 (0%)	
@reserv_sets_are_interse...	0x406c77	0x23e	genautomata	genautomata	2201 (100%)	1205 (54%)	2678 3944	31 (1%)	10 (0%)		62 (2%)			10 (0%)	
@_cpp_clean_line	0xeed07c	0x4c9	cc1	cc1	2736 (100%)	2048 (74%)	1817 2827		751 (27%)	10 (0%)	72 (2%)	31 (1%)			
@htab_find_slot_with_hash	0xf28869	0x230	cc1	cc1	2972 (100%)	2453 (82%)	1280 2166	1489 (47%)	720 (24%)		247 (8%)	10 (0%)		10 (0%)	
@find_costs_and_classes	0x749b0d	0x10f4	cc1	cc1	2232 (100%)	1435 (64%)	2405 2975	123 (5%)	51 (2%)		62 (2%)			10 (0%)	
@pool_alloc	0x5380f1	0x15f	cc1	cc1	2448 (100%)	1719 (70%)	2099 2725	51 (2%)	165 (6%)			82 (3%)			
@c_lex_with_flags	0x516c24	0x6c2	cc1	cc1	2797 (100%)	1960 (70%)	1254 2417	108 (10%)	617 (22%)		216 (7%)	247 (8%)		21 (0%)	

Help

RecoTRFW...tar.bz2 | RecoTRFW...tar.bz2 | ParallelG4.tar.bz2

Show all downloads...

Double click on Function name to go to source view

The screenshot displays the Generic Optimization Data Analyzer GUI. The top section shows a table of performance metrics for various processes and modules. The bottom section shows a table of performance metrics for various functions. The 'cc1' function is highlighted in red, and a red circle is drawn around its name in the first row of the function table. A mouse cursor is hovering over the function name 'cc1'.

Function Name	Offset	Length	Module	Process	unmated_core_cycles	uops_retired:stall_cycles	instruction_retired	uops_retired:any	load_latency	instruction_starvation	bandwidth_saturated	branch_misprediction	store_resources_saturated	instruction_latency	exception_handling
cc1				cc1	578999 (99%)	434163 (74%)	423477	618659	108639 (17%)	216550 (37%)	8646 (1%)	30707 (6%)	13853 (2%)	411 (0%)	2931 (0%)
cc1				cc1	12588 (100%)	9693 (77%)	9993	11821	1183 (9%)	3085 (24%)	21 (0%)	926 (7%)	62 (0%)		41 (0%)
cc1				cc1	18387 (100%)	6352 (61%)	9323	13577	298 (2%)	442 (4%)		378 (3%)		10 (0%)	10 (0%)
cc1				cc1	8052 (100%)	5914 (73%)	6457	7923	1861 (23%)	2129 (26%)		411 (5%)	51 (0%)		10 (0%)
cc1				cc1	9091 (100%)	8959 (98%)	2883	4835	8371 (92%)	2188 (23%)	10 (0%)	309 (3%)	10 (0%)		31 (0%)
cc1				cc1	6859 (100%)	4381 (63%)	7157	8652	216 (3%)	823 (11%)	21 (0%)	206 (3%)			21 (0%)
cc1				cc1	5656 (100%)	2596 (45%)	6645	10809		62 (1%)		10 (0%)	154 (2%)	10 (0%)	10 (0%)
cc1				cc1	6222 (100%)	4589 (73%)	4837	6999	93 (1%)	1913 (30%)	10 (0%)	247 (3%)			21 (0%)
cc1				cc1	5882 (100%)	4644 (78%)	4717	5723	584 (8%)	1584 (26%)	10 (0%)	452 (7%)	226 (3%)		
cc1				cc1	5327 (100%)	3034 (56%)	5476	6429	566 (10%)	987 (18%)		298 (5%)	31 (0%)		21 (0%)
cc1				cc1	4985 (100%)	2793 (56%)	3753	5871	51 (1%)	113 (2%)		62 (1%)			41 (0%)
cc1				cc1	4052 (100%)	2234 (55%)	3873	7934	41 (1%)	93 (2%)		10 (0%)	154 (3%)		21 (0%)
cc1				cc1	4432 (100%)	3965 (89%)	1638	2611	247 (5%)	4391 (99%)	165 (3%)	545 (12%)	10 (0%)		10 (0%)
cc1				cc1	3589 (100%)	2311 (64%)	2653	4286	72 (2%)	1429 (39%)	10 (0%)	309 (8%)	51 (1%)		10 (0%)
cc1				cc1	3188 (100%)	1884 (59%)	3156	5816	62 (1%)	41 (1%)			206 (6%)		
cc1				cc1	3239 (100%)	1599 (49%)	4652	6133		154 (4%)		10 (0%)			21 (0%)
cc1				cc1	2921 (100%)	1522 (52%)	3782	4982	82 (2%)	566 (19%)		31 (1%)	31 (1%)	10 (0%)	21 (0%)
cc1				cc1	3383 (100%)	2201 (65%)	2192	4252	617 (18%)	915 (27%)		41 (1%)	103 (3%)		21 (0%)
cc1				cc1	3311 (100%)	2377 (71%)	2354	3295	401 (12%)	1368 (41%)		381 (11%)			21 (0%)
cc1				cc1	2355 (100%)	1388 (58%)	3327	4628	31 (1%)	144 (6%)		21 (0%)	10 (0%)		
cc1				cc1	2548 (100%)	1873 (73%)	2243	3348	113 (4%)	1142 (44%)	41 (1%)	93 (3%)	62 (2%)	10 (0%)	10 (0%)
cc1				cc1	2869 (100%)	1884 (65%)	2898	2884	82 (2%)	967 (33%)		278 (9%)			10 (0%)
cc1				cc1	2201 (100%)	1295 (54%)	2678	3944	31 (1%)	10 (0%)		62 (2%)			10 (0%)
cc1				cc1	2736 (100%)	2848 (74%)	1817	2827		751 (27%)	10 (0%)	72 (2%)	31 (1%)		
cc1				cc1	2972 (100%)	2453 (82%)	1286	2166	1489 (47%)	728 (24%)		247 (8%)	10 (0%)		10 (0%)
cc1				cc1	2232 (100%)	1435 (64%)	2485	2975	123 (5%)	51 (2%)		62 (2%)			10 (0%)
cc1				cc1	2448 (100%)	1719 (70%)	2898	2725	51 (2%)	165 (6%)			82 (3%)		
cc1				cc1	2797 (100%)	1968 (70%)	1254	2417	298 (10%)	617 (22%)		216 (7%)	247 (8%)		21 (0%)

Three pane source view

The screenshot displays the Generic Optimization Data Analyzer GUI, which is divided into three main panes for source code analysis.

- Top Left Pane (Assembly):** Shows assembly code for the function `_cpp_lex_direct`. It includes columns for address, principal instruction, disassembly, and various performance metrics such as `unmated_core_cycles`, `uops_retired`, `stall_cycles`, `instruction_retired`, `uops_retired_any`, `load_latency`, `instruction_starvation`, `bandwidth_saturated`, `branch_misprediction`, and `store_resources_except`.
- Top Right Pane (Source Code):** Displays the corresponding C++ source code. The code includes conditional logic like `if (buffer->need_line)` and `if (!_cpp_get_fresh_`. It also features performance metrics similar to the assembly pane, such as `unmated_core_cycles`, `uops_retired`, `stall_cycles`, `instruction_retired`, `uops_retired_any`, `load_latency`, `instruction_starvation`, `bandwidth_saturated`, `branch_misprediction`, `store_resources_except`, and `store_resources_except`.
- Bottom Pane (Control Flow Graph):** A graph showing the flow of execution between basic blocks. Nodes are labeled "Basic Block 0" through "Basic Block 85". A red box highlights a specific area of the graph, likely corresponding to the assembly and source code shown above.

The interface also includes a "Reports" sidebar on the left, a search bar at the top, and a taskbar at the bottom with open applications like "RecoTRFWe...tar.bz2" and "ParallelG4.tar.bz2".

Collapse the basic blocks

The screenshot displays the Generic Optimization Data Analyzer GUI. The interface is divided into several sections:

- Reports:** A sidebar on the left lists various reports, with 'Sample' selected.
- Sample Hotspots:** A tab titled '_cpp_lex_direct' is active, showing a table of performance metrics for 27 basic blocks.
- Code View:** A window on the right shows the source code for the selected basic blocks, with line numbers and source text.
- Control Flow Graph (CFG):** A graph at the bottom illustrates the relationships between basic blocks, with a red box highlighting a specific area of interest.

Table 1: Basic Block Performance Metrics

address	princ_id	disassembly	umalted_core_cycles	uops_retired	stall_cycles	instruction_retired	uops_retired_any	load_latency	instruction_starvation	bandwidth_saturated	branch_misprediction	store
0xfef09f1	1988	Basic Block 1 <0xfef0a2...	12588 (100%)	9693 (77%)	9093	11821	1183 (9%)	3085 (24%)	21 (0%)	826 (7%)	62 (0%)	41 (0%)
0xfef0a28	1993	Basic Block 2 <0xfef0b7...	740 (100%)	350 (47%)	648	798	134 (18%)	10 (1%)	31			
0xfef0a50	1995	Basic Block 3 <0xfef0aa...		11		11						
0xfef0a5f	1997	Basic Block 4 <0xfef0a9...										
0xfef0a81	2000	Basic Block 5 <0xfef0a9...										
0xfef0a99	2001	Basic Block 6 <0xfef0b3...										
0xfef0aa2	2003	Basic Block 7 <0xfef0be...	21 (100%)	22 (104%)	9	11						
0xfef0ab1	2003	Basic Block 8 <0xfef0af...			22	17	23		10			
0xfef0ab5	2006	Basic Block 9 <0xfef0ae...	31 (100%)	11 (35%)	34	57						
0xfef0acc	2009	Basic Block 10 <0xfef0a...										
0xfef0aea	2012	Basic Block 11 <0xfef1b...		22	17	11						
0xfef0af3	2014	Basic Block 12 <0xfef0b...	82 (100%)	11 (13%)	77	125		10 (12%)				
0xfef0b04	2018	Basic Block 13 <0xfef0b...	185 (100%)	55 (29%)	162	194						
0xfef0b48	2021	Basic Block 14 <0xfef0b...	82 (100%)	99 (120%)	51	103	31 (37%)					
0xfef0b61	2022	Basic Block 15 <0xfef0b...	10 (100%)									
0xfef0b76	2024	Basic Block 16 <0xfef0b...	154 (100%)	110 (71%)	60	91		31 (20%)	62 (40%)			
0xfef0b84	2026	Basic Block 17 <0xfef0b...	360 (100%)	383 (106%)	179	422	103 (28%)	165 (45%)	31 (8%)	10		
0xfef0b98	2029	Basic Block 18 <0xfef0b...	1265 (100%)	953 (75%)	913	1072	144 (11%)	123 (9%)				
0xfef0bc0	2030	Basic Block 19 <0xfef0b...			9							
0xfef0b00	2032	Basic Block 20 <0xeeeb...										
0xfef0be4	2033	Basic Block 21 <0xfef0b...		11	9	11						
0xfef0bf8	2035	Basic Block 22 <0xfef0c...	720 (100%)	584 (78%)	870	570	82 (11%)	82 (11%)	10 (1%)			
0xfef0c26	2038	Basic Block 23 <0xfef0c...	10 (100%)									
0xfef0c3e	2040	Basic Block 24 <0xfef37...	740 (100%)	570 (77%)	964	809	31 (4%)	51 (6%)				
0xfef0c66	2040	Basic Block 25 <0xfef0c...	280 (100%)	230 (79%)	299	353		21 (7%)				
0xfef0c6c	2043	Basic Block 26 <0xfef1a...	566 (100%)	350 (61%)	452	616		72 (12%)				
0xfef0c76	2043	Basic Block 27 <0xdead...	946 (100%)	778 (82%)	614	1129	432 (45%)	288 (30%)	10 (1%)	31 (3%)		

Table 2: Source Code Snippets

```
1993 if (buffer->need_line)
1994 {
1995     if (pfile->state.in_
1996     {
1997         result->type = C_
1998         pfile->state.in_
1999         if (lpfile->stat_
2000         pfile->state.p_
2001         return result;
2002     }
2003     if (!_cpp_get_fresh_
2004     {
2005         result->type = C_
2006         if (lpfile->stat_
2007         {
2008             /* Tell the _
2009             result->src_
2010             result->flag_
2011         }
2012         return result;
2013     }
2014     if (lpfile->keep_tok_
2015     {
2016         pfile->cur_run = _
2017         result = pfile->_
2018         pfile->cur_token_
2019     }
```

Sort by cycles

The screenshot displays the Generic Optimization Data Analyzer GUI. The main window is titled "Generic Optimization Data Analyzer GUI" and shows a "Sample Hotspots" report for the file "_cpp_lex_direct". The report is sorted by "Cycles" and displays a table of hotspots with columns for address, principal ID, assembly, and various performance metrics.

address	princ_id	assembly	unmated_core_cycles	uops_retired_stall_cycles	instruction_retired_uops_retired	load_latency	instruction_starvation	bandwidth_saturated	branch_misprediction	store		
0xef0a28	1993	Basic Block 2 <0xef0b7...	12588 (100%)	9693 (77%)	9093	11821	1183 (9%)	8085 (24%)	21 (0%)	826 (7%)	62 (0%)	41 (0%)
0xef0b98	2029	Basic Block 18 <0xef0b...	1265 (100%)	953 (75%)	913	1072	144 (11%)	123 (9%)				
0xef0c76	2043	Basic Block 27 <0xdead...	946 (100%)	778 (82%)	614	1129	432 (45%)	288 (30%)	10 (1%)	31 (3%)		
0xef0ecc	2184	Basic Block 55 <0xeecc...	823 (100%)	668 (81%)	264	388		113 (13%)	10 (1%)	267 (32%)		
0xef09f1	1988	Basic Block 1 <0xef0a2...	740 (100%)	350 (47%)	640	798		134 (18%)		10 (1%)	31 (0%)	
0xef0c3e	2040	Basic Block 24 <0xef37...	740 (100%)	570 (77%)	964	809	31 (4%)	51 (6%)				
0xef0bf8	2035	Basic Block 22 <0xef0c...	720 (100%)	504 (70%)	870	570	82 (11%)	82 (11%)		10 (1%)		
0xef0c83	2046	Basic Block 28 <0xeeee6...	576 (100%)	471 (81%)	256	467	10 (1%)	41 (7%)		72 (12%)		
0xef0c6c	2043	Basic Block 26 <0xef1a...	566 (100%)	350 (61%)	452	616		72 (12%)				
0xef0cc9	2052	Basic Block 31 <0xef34...	514 (100%)	449 (87%)	239	274	31 (6%)	288 (56%)		10 (1%)		
0xef0f31	2113	Basic Block 57 <0xef1a...	381 (100%)	153 (40%)	418	650		82 (21%)				
0xef0bb4	2026	Basic Block 17 <0xef0b...	360 (100%)	383 (106%)	179	422	103 (28%)	165 (45%)		31 (8%)	10 (0%)	
0xef0c66	2040	Basic Block 25 <0xef0c...	280 (100%)	230 (79%)	299	353		21 (7%)				
0xef1995	2317	Basic Block 167 <0xef1...	257 (100%)	175 (68%)	60	80		10 (3%)				
0xef0cb1	2051	Basic Block 30 <0xef0d...	237 (100%)	285 (120%)	60	91		21 (8%)		62 (26%)		
0xef0bb4	2018	Basic Block 13 <0xef0b...	185 (100%)	55 (29%)	162	194						
0xef1b38	2350	Basic Block 209 <0xdea...	175 (100%)	186 (106%)	68	171	10 (5%)	103 (58%)				
0xef0b76	2024	Basic Block 16 <0xef0b...	154 (100%)	110 (71%)	60	91		31 (20%)		62 (40%)		
0xef1988	2316	Basic Block 166 <0xef1...	154 (100%)	131 (85%)	60	103		21 (13%)				
0xef19a2	2318	Basic Block 168 <0xef1...	134 (100%)	230 (171%)	43	137		21 (15%)				
0xef0d4c	2054	Basic Block 32 <0xef0a...	123 (100%)	110 (89%)	145	125		10 (8%)				
0xef1aeb	2118	Basic Block 183 <0xef1...	123 (100%)	77 (62%)	77	148		51 (41%)				
0xef192e	2312	Basic Block 161 <0xef1...	113 (100%)	110 (97%)	17	11		41 (36%)				
0xef19e3	2323	Basic Block 173 <0xef1...	113 (100%)	88 (77%)	9	11		82 (72%)				
0xef0f0f	2109	Basic Block 56 <0xeeee7...	103 (100%)	131 (127%)	179	160	21 (20%)	62 (60%)				
0xef0a73	2014	Basic Block 12 <0xef0b...	82 (100%)	11 (13%)	77	125		10 (12%)				
0xef0b48	2021	Basic Block 14 <0xef0b...	82 (100%)	99 (120%)	51	103	31 (37%)					

Below the table is a control flow graph (CFG) showing the relationships between basic blocks. The graph consists of nodes labeled "Basic Block 0" through "Basic Block 85" and "Addr 0" through "Addr 19". The blocks are connected by arrows indicating control flow. A red shaded area highlights a cluster of blocks, including Basic Block 1, Basic Block 32, and Basic Block 33.

Move all 3 panes in unison

The screenshot displays the Generic Optimization Data Analyzer GUI, which is divided into three main panes. The top-left pane, titled 'Reports', shows a list of sample hotspots for the function '_cpp_lex_direct'. The top-right pane, titled 'Cycles Samples', displays a detailed table of performance metrics for each hotspot, including address, PC, assembly, and various cycle and latency counts. The bottom pane shows a control flow graph (CFG) with nodes representing basic blocks and addresses, connected by flow edges. A red box highlights 'Basic Block 18' in the CFG, which corresponds to the selected row in the 'Cycles Samples' table.

address	pc	atassembly	umalted_core_cycles	uops_retired	stall_cycles	instruction_retired	uops_retired: any	load_latency	instruction_starvation	bandwidth_saturated	branch_misprediction	store
0xef8a28	1993	Basic Block 2 <0xef8b7...	12588 (100%)	9693 (77%)	9093	11821	1183 (9%)	3085 (24%)	21 (0%)	926 (7%)	62 (0%)	41 (2%)
0xef8b98	2029	Basic Block 18 <0xef8b...	1265 (100%)	953 (75%)	913	1072	144 (11%)	123 (9%)				
0xef8c76	2043	Basic Block 27 <0xdead...	946 (100%)	778 (82%)	614	1129	432 (45%)	288 (30%)	10 (1%)	31 (3%)		
0xef8ecc	2104	Basic Block 35 <0xeeec...	823 (100%)	668 (81%)	264	388		113 (13%)	10 (1%)	267 (32%)		
0xef891f	1988	Basic Block 1 <0xef8a2...	740 (100%)	350 (47%)	640	798		134 (18%)		10 (1%)	31 (4%)	
0xef8c3e	2040	Basic Block 24 <0xef83...	740 (100%)	570 (77%)	964	809	31 (4%)	51 (6%)				
0xef8b8f	2035	Basic Block 22 <0xef8c...	720 (100%)	504 (70%)	870	570	82 (11%)	82 (11%)		10 (1%)		
0xef8c83	2046	Basic Block 28 <0xeeee...	576 (100%)	471 (81%)	256	467	10 (1%)	41 (7%)		72 (12%)		
0xef8c6c	2043	Basic Block 26 <0xef1...	566 (100%)	350 (61%)	452	616		72 (12%)				
0xef8cc9	2052	Basic Block 31 <0xef34...	514 (100%)	449 (87%)	239	274	31 (6%)	208 (56%)		10 (1%)		
0xef8f31	2113	Basic Block 37 <0xef1a...	381 (100%)	153 (40%)	418	650		82 (21%)				
0xef8b04	2026	Basic Block 17 <0xef8b...	360 (100%)	303 (84%)	179	422	103 (28%)	165 (45%)		31 (8%)	10 (3%)	
0xef8c66	2040	Basic Block 25 <0xef8c...	288 (100%)	230 (79%)	299	353		21 (7%)				
0xef1995	2317	Basic Block 167 <0xef1...	257 (100%)	175 (68%)	60	80		10 (3%)				
0xef8cbl	2051	Basic Block 30 <0xef8d...	237 (100%)	285 (120%)	60	91		21 (8%)		62 (26%)		
0xef8b04	2018	Basic Block 13 <0xef8b...	185 (100%)	55 (29%)	162	194						
0xef1b38	2350	Basic Block 209 <0xdea...	175 (100%)	106 (106%)	60	171	10 (5%)	103 (58%)				
0xef8b76	2024	Basic Block 16 <0xef8b...	154 (100%)	110 (71%)	60	91		31 (20%)		62 (40%)		
0xef1908	2316	Basic Block 166 <0xef1...	154 (100%)	131 (85%)	60	103		21 (13%)				
0xef19a2	2318	Basic Block 168 <0xef1...	134 (100%)	230 (171%)	43	137		21 (15%)				
0xef8d4c	2054	Basic Block 32 <0xef8a...	123 (100%)	110 (89%)	145	125		10 (8%)				
0xef1aeb	2118	Basic Block 183 <0xef1...	123 (100%)	77 (62%)	77	148		51 (41%)				
0xef192e	2312	Basic Block 161 <0xef1...	113 (100%)	110 (97%)	17	11		41 (36%)				
0xef19e3	2323	Basic Block 173 <0xef1...	113 (100%)	88 (77%)	9	11		82 (72%)				
0xef8f0f	2109	Basic Block 36 <0xeeee...	103 (100%)	131 (127%)	179	160	21 (20%)	62 (60%)				
0xef8af3	2014	Basic Block 12 <0xef8b...	82 (100%)	11 (13%)	77	125		10 (12%)				
0xef8b48	2021	Basic Block 14 <0xef8b...	82 (100%)	99 (120%)	51	103	31 (37%)					

Expand 1 basic block

Generic Optimization Data Analyzer GUI

Sample Hotspots: `_cpp_lex_direct`

address	princ_id	disassembly	unmated_core_cycles	uops_retired	instruction_stall_cycles	uops_retired	load_latency	instruction_starvation	bandwidth_saturated	branch_misprediction	store
0x0fb28	1993	Basic Block 2 <0x0fb07...	12588 (100%)	9693 (77%)	9893	11821	1183 (9%)	3085 (24%)	21 (0%)	826 (7%)	62 (0%)
0x0fb98	2029	Basic Block 10 <0x0fb0...	1265 (100%)	953 (75%)	913	1072	144 (11%)	123 (9%)			
0x0fb98	2030	mov -0x48(%rbp),%rax	72 (100%)	88 (122%)	51	114					
0x0fb9c	2030	mov (%rax),%rdx	10 (100%)	11 (110%)	17	23	21 (210%)				
0x0fb9f	2029	mov -0x48(%rbp),%rax	93 (100%)	131 (140%)	128	125		10 (10%)			
0x0fba3	2029	mov 0x28(%rax),%rcx	10 (100%)		11						
0x0fba7	2029	mov -0x48(%rbp),%rax	216 (100%)	142 (65%)	85	160	10 (4%)				
0x0fbab	2029	mov 0x30(%rax),%eax									
0x0fbac	2029	mov %eax,%eax	72 (100%)	66 (91%)	43	80					
0x0fbb0	2029	shl \$0x4,%rax	21 (100%)	22 (104%)	17	11					
0x0fbb4	2029	lea (%rcx,%rax,1),%r...	103 (100%)	77 (74%)	51	57					
0x0fbb8	2030	mov (%rax),%rax	51 (100%)	22 (43%)	43	23	10 (19%)				
0x0fbbb	2029	cmp %rax,%rdx	617 (100%)	394 (63%)	478	467	72 (11%)	72 (11%)			
0x0fbbe	2029	jb ef0bf8									
0x0fc76	2043	Basic Block 27 <0xdead...	946 (100%)	778 (82%)	614	1129	432 (45%)	288 (30%)	10 (1%)	31 (3%)	
0x0fc8c	2104	Basic Block 55 <0xeeec...	823 (100%)	668 (81%)	264	388		113 (13%)	10 (1%)	267 (32%)	
0x0f09f1	1988	Basic Block 1 <0x0fa02...	740 (100%)	350 (47%)	640	798		134 (18%)		10 (1%)	31 (4%)
0x0fc3e	2040	Basic Block 24 <0x0ef37...	740 (100%)	570 (77%)	964	809	31 (4%)	51 (6%)			
0x0fbf8	2035	Basic Block 22 <0x0ef0c...	720 (100%)	504 (70%)	870	570	82 (11%)	82 (11%)		10 (1%)	
0x0fc03	2046	Basic Block 20 <0xeeeb...	576 (100%)	471 (81%)	256	467	10 (1%)	41 (7%)	72 (12%)		
0x0fc6c	2043	Basic Block 26 <0x0ef1a...	566 (100%)	350 (61%)	452	616		72 (12%)			
0x0fc09	2052	Basic Block 31 <0x0ef34...	514 (100%)	449 (87%)	239	274	31 (6%)	288 (56%)		10 (1%)	
0x0f031	2113	Basic Block 57 <0x0ef1a...	381 (100%)	153 (40%)	418	650		82 (21%)			
0x0fb84	2026	Basic Block 17 <0x0fb0b...	360 (100%)	383 (106%)	179	422	103 (28%)	165 (45%)		31 (8%)	10 (3%)
0x0fc66	2040	Basic Block 25 <0x0ef0c...	288 (100%)	230 (79%)	299	353		21 (7%)			
0x0f195	2317	Basic Block 167 <0x0ef1...	257 (100%)	175 (68%)	60	80		10 (3%)			
0x0f0c1	2051	Basic Block 30 <0x0ef0d...	237 (100%)	285 (120%)	60	91		21 (8%)	62 (26%)		

Line number | source | unmated_core_cycles | uops_retired | instruction_stall_cycles | uops_retired | load_latency | instruction_starvation | bandwidth_saturated | branch_misprediction | store_resources_sa | except

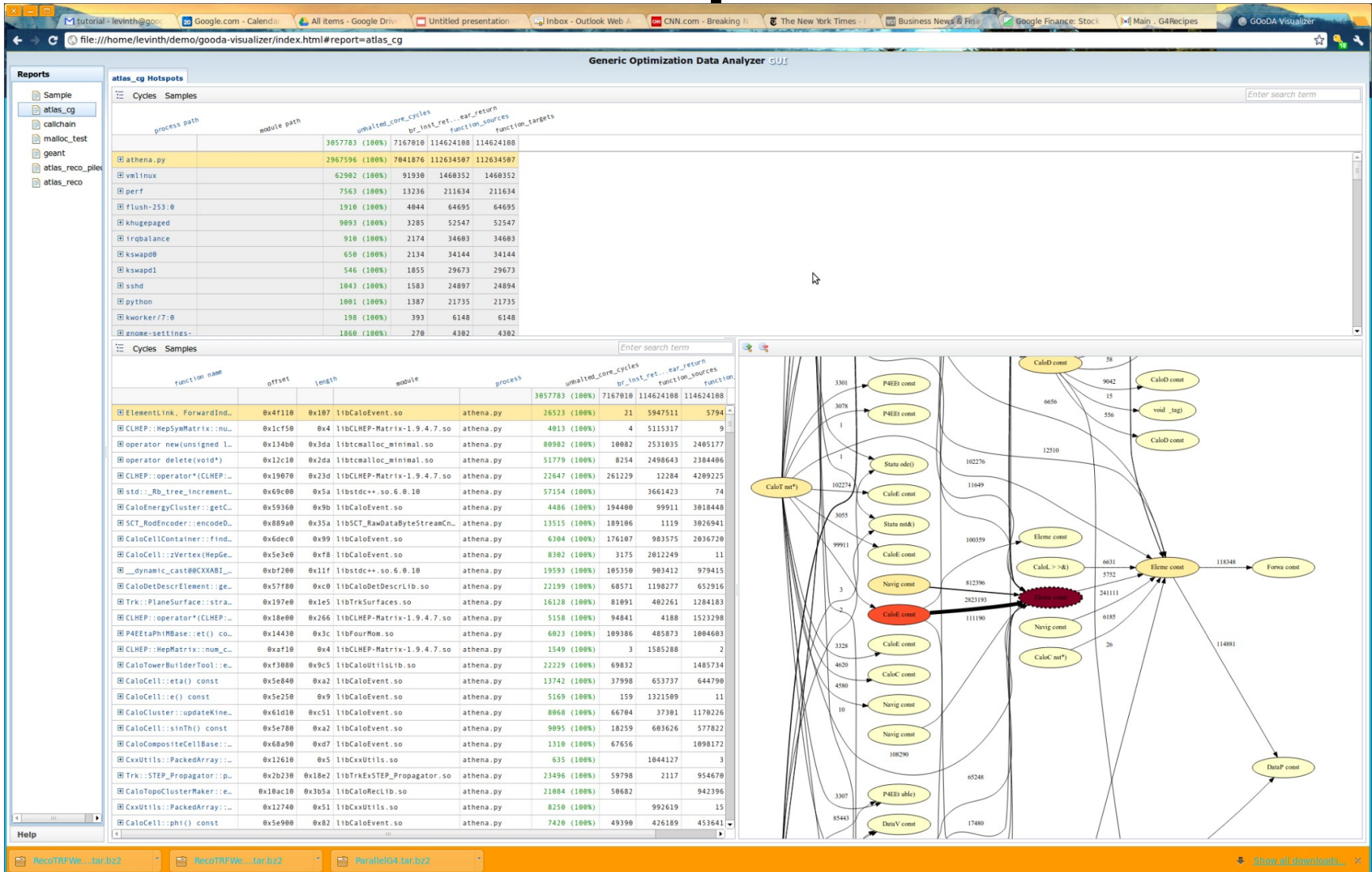
```
2029 if (buffer->cur >= buffe...
2030 && !pfile->overlaid_...
2031 {
2032     _cpp_process_line_no...
2033     result->src_loc = pf...
2034 }
2035 c = *buffer->cur++;
2036
2037 if (pfile->forced_token_...
2038     result->src_loc = "pf...
2039 else
2040     result->src_loc = line...
2041     123 (100%) 153 (124%) 179 57 10 (8%) 21 (17%)
2042
2043 switch (c)
2044 {
2045     case ' ': case '\t': c...
2046     result->flags |= PRE...
2047     skip_whitespace (pf...
2048     goto skipped_white;
2049
2050 case '\n':
2051     if (buffer->cur < bu...
2052         CPP_INCREMENT_LINE...
2053     buffer->need_line = ...
2054     goto fresh_line;
2055
```

Control flow graph showing basic blocks and their connections. Basic Block 17 is highlighted in red.

Help

RecoTRFW...tar.bz2 | RecoTRFW...tar.bz2 | ParallelG4.tar.bz2 | Show all downloads...

The Call Count Graph



Shrink the graph

Generic Optimization Data Analyzer GUI

atlas_cg Hotspots

Cycles Samples

process path	module path	umathd_core_cycles	br_inst_ret...	ear_return	function_sources	function_targets
athena.py		3857783 (100%)	7167810	114624108	114624108	
vm1linux		2967596 (100%)	7841876	112634507	112634507	
perf		62902 (100%)	91930	1468352	1468352	
flush-253:0		7563 (100%)	13236	211634	211634	
khugepaged		1910 (100%)	4044	64695	64695	
irqbalance		9893 (100%)	3285	52547	52547	
kswapd0		918 (100%)	2174	34603	34603	
kswapd1		650 (100%)	2134	34144	34144	
sshd		546 (100%)	1855	29673	29673	
python		1043 (100%)	1583	24897	24894	
worker/7:0		1001 (100%)	1387	21735	21735	
znome-settings-		198 (100%)	393	6148	6148	
znome-settings-		1868 (100%)	278	4382	4382	

function name offset length module process umathd_core_cycles br_inst_ret... ear_return function_sources function_targets

ElementLink, ForwardInd...	0x4f110	0x107	libCaloEvent.so	athena.py	3857783 (100%)	7167810	114624108	114624108	
CaloEnergyCluster::getC...	0x59398		libCaloEvent.so		26523 (100%)	21	5947511	5794	
CaloCompositeCellBase::...	0x68a18		libCaloEvent.so						2823193
Navigable, ForwardIndex...	0x5a158		libCaloEvent.so						996446
CaloCompositeCellBase::...	0x685c8		libCaloEvent.so						812396
Navigable, ForwardIndex...	0x4d9bd		libCaloEvent.so						689170
CaloTopoTowerBuilderToo...	0x8d9af		libCaloUtilsLib.so						111190
softeTopoBuilder::AddTo...	0x1bd998		libgammaReclib.so						108290
Navigable, ForwardIndex...	0x4db35		libCaloEvent.so						75818
Navigable, ForwardIndex...	0x4e4dd		libCaloEvent.so						66891
CaloCompositeCellBase::...	0x68a4f		libCaloEvent.so						65248
ElementLink, ForwardInd...	0x4f07a		libCaloEvent.so						17783
	0x50b129		vm1linux						5752
CLHEP::HepSymMatrix::nu...	0x1c158	0x4	libCLHEP-Matrix-1.9.4.7.so	athena.py	4813 (100%)	4	5115317	9	42
operator new(unsigned l...	0x1340b	0x3da	libtcmalloc_minimal.so	athena.py	80982 (100%)	10882	2531035	2485177	
operator delete(void*)	0x12c18	0x2da	libtcmalloc_minimal.so	athena.py	51779 (100%)	8254	2498643	2384486	
CLHEP::operator*(CLHEP...	0x19070	0x23d	libCLHEP-Matrix-1.9.4.7.so	athena.py	22647 (100%)	261229	12284	4289225	
std::_Rb_tree_increment...	0x69c08	0x5a	libstdc++.so.6.0.10	athena.py	57154 (100%)		3661423	74	
CaloEnergyCluster::getC...	0x59360	0x9b	libCaloEvent.so	athena.py	4486 (100%)	194480	99911	3018448	
SCT_RodEncoder::encode...	0x889a8	0x35a	libSCT_RawDataByteStreamCh...	athena.py	13515 (100%)	189186	1119	3826941	
CaloCellContainer::find...	0x6dec8	0x99	libCaloEvent.so	athena.py	6384 (100%)	176187	983575	2836728	
CaloCell::Vertex(HepGe...	0x5e3e8	0xf8	libCaloEvent.so	athena.py	8302 (100%)	3175	2012249	11	
_dynamic_cast@GCCXABI...	0xf1208	0x11f	libstdc++.so.6.0.10	athena.py	19593 (100%)	185358	98412	979415	
CaloDetDescrElement::ge...	0x57f88	0xc0	libCaloDetDescrLib.so	athena.py	22199 (100%)	68571	1198277	652916	
Trk::PlaneSurface::stra...	0x197e8	0x1e5	libTrkSurfaces.so	athena.py	16128 (100%)	81891	402261	1284183	
CLHEP::operator*(CLHEP...	0x18e08	0x266	libCLHEP-Matrix-1.9.4.7.so	athena.py	5158 (100%)	94841	4188	1532398	
P4EEtaPhiBase::et() co...	0x14438	0x3c	libFourMom.so	athena.py	6823 (100%)	189386	485873	1084683	