LG Electronics

uftrace: A function graph tracer for userspace programs

https://github.com/namhyung/uftrace

Namhyung Kim(namhyung.kim@lge.com), Honggyu Kim(hong.gyu.kim@lge.com)

Introduction

The ftrace framework in Linux kernel utilizes function instrumentation techniques from compilers to provide deeper understanding of kernel execution behavior and performance characteristics. The same thing can be provided to userspace programs written in C/C++ with uftrace.

Technical Description

- uftrace uses compiler assist to trace each function calls.
  - same as Linux kernel ftrace function graph does
- '-pg' option inserts mcount function calls at the entries of each function.
- uftrace hooks these mcount calls and sets return trampolines.
  - those hooks record function call and return information during execution.
- It also supports '-finstrument-functions'
  - to set hook functions at both entries and exits of each function.
- It does NOT require source code modification.

Features

- It shows how target program actually executes internally
  - by showing function call and return relations
- It shows execution time of each function call
- also traces library calls and Linux kernel functions
- provides various filtering to selectively show or hide trace output
  - depth filter, function filter, and time filter
- able to show its trace output in chrome browser for GUI friendly-users
  - does NOT require additional plugin installation

Limitations

- It can only trace a native C/C++ application compiled with -pg option.
- It cannot trace already running process.
- It cannot be used for system-wide tracing.

uftrace Commands

- **record** - Run a command and record its trace data
- **replay** - Print recorded function trace
- **report** - Print statistics and summary for trace data
- **live** - Trace functions in a command lively
- **dump** - Print raw tracing data in the data files
- **info** - Print tracing information for trace data
- **recv** - Receive tracing data from socket and save it to files
- **graph** – Show function call graph

Simple Usage

```
$ cat abc.c
void c() {}
void b() { c(); }
void a() { b(); }
int main() { a(); }

$ gcc -pg -o abc abc.c
$ uftrace ./abc
# DURATION    TID     FUNCTION
   0.470 us [ 9767] | __monstartup();
   0.348 us [ 9767] | __cxa_atexit();
            [ 9767] | main() {
            [ 9767] |   a() {
            [ 9767] |     b() {
   0.065 us [ 9767] |       c();
   0.382 us [ 9767] |     } /* b */
   0.559 us [ 9767] |   } /* a */
   0.707 us [ 9767] | } /* main */

$ uftrace record -F b ./abc
$ uftrace replay
# DURATION    TID     FUNCTION
            [ 9900] | b() {
   0.388 us [ 9900] |   c();
   2.667 us [ 9900] | } /* b */

$ uftrace record -N b ./abc
$ uftrace replay
# DURATION    TID     FUNCTION
   2.207 us [ 9946] | __monstartup();
   1.536 us [ 9946] | __cxa_atexit();
            [ 9946] | main() {
   1.766 us [ 9946] |   a();
   3.257 us [ 9946] | } /* main */
```

Notable Options

- **-D DEPTH, --depth=DEPTH**
  Set global trace limit in nesting level.
- **-F FUNC, --filter=FUNC**
  Set filter to trace selected functions only.
- **-N FUNC, --notrace=FUNC**
  Set filter not trace selected functions only.
- **-T TRG, --trigger=TRG**
  Set trigger on selected functions.
- **-t TIME, --time-filter=TIME**
  Do not show small functions under the time threshold.
- **-A SPEC, --argument=SPEC**
  Record function arguments.
- **-R SPEC, --retval=SPEC**
  Record function return value.

Arguments/Return Value Display

```
$ gcc -pg -o fibonacci fibonacci.c
$ uftrace -A fib@arg1 -R fib@retval ./fibonacci 5
# DURATION    TID     FUNCTION
   1.624 us [ 6780] | __monstartup();
   0.997 us [ 6780] | __cxa_atexit();
            [ 6780] | main() {
   1.336 us [ 6780] |   atoi();
            [ 6780] |   fib(5) {
            [ 6780] |     fib(4) {
            [ 6780] |       fib(3) {
   3.398 us [ 6780] |         fib(2) = 1;
   0.231 us [ 6780] |         fib(1) = 1;
   5.679 us [ 6780] |       } = 2; /* fib */
   0.226 us [ 6780] |       fib(2) = 1;
   6.924 us [ 6780] |     } = 3; /* fib */
            [ 6780] |     fib(3) {
   0.219 us [ 6780] |       fib(2) = 1;
   0.177 us [ 6780] |       fib(1) = 1;
   1.504 us [ 6780] |     } = 2; /* fib */
  10.006 us [ 6780] |   } = 5; /* fib */
   9.899 us [ 6780] |   printf();
  23.382 us [ 6780] | } /* main */
```

Analyzing Smart Pointer Resource Management in STL

```
$ cat shared_ptr.cpp
#include <memory>
int main() {
  std::shared_ptr<int> s1(new int);
  {
    std::shared_ptr<int> s2 = s1;
  }
}

$ g++ -pg -std=c++14 -o shared_ptr shared_ptr.cpp
$ uftrace -D 4 \
        -F main -F "operator .*" -F "std::shared_ptr::.*" \
        -A "operator new"@arg1 -R "operator new"@retval \
        -A "operator delete"@arg1 \
        shared_ptr

# DURATION    TID     FUNCTION
            [10471] | main() {
   2.335 us [10471] |   operator new(4) = 0x1209910;
            [10471] |   std::shared_ptr::shared_ptr() {
            [10471] |     std::__shared_ptr::__shared_ptr() {
            [10471] |       std::__shared_count::__shared_count() {
   2.860 us [10471] |         operator new(24) = 0x122d630;
   0.456 us [10471] |         std::_Sp_counted_ptr::_Sp_counted_ptr();
   4.907 us [10471] |       } /* std::__shared_count::__shared_count */
   0.163 us [10471] |       std::__enable_shared_from_this_helper();
   5.982 us [10471] |     } /* std::__shared_ptr::__shared_ptr */
   6.450 us [10471] |   } /* std::shared_ptr::shared_ptr */
            [10471] |   std::shared_ptr::shared_ptr() {
            [10471] |     std::__shared_ptr::__shared_ptr() {
            [10471] |       std::__shared_count::__shared_count() {
   0.649 us [10471] |         std::_Sp_counted_base::_M_add_ref_copy();
   1.313 us [10471] |       } /* std::__shared_count::__shared_count */
   1.735 us [10471] |     } /* std::__shared_ptr::__shared_ptr */
   2.177 us [10471] |   } /* std::shared_ptr::shared_ptr */
            [10471] |   std::shared_ptr::~shared_ptr() {
            [10471] |     std::__shared_ptr::~__shared_ptr() {
            [10471] |       std::__shared_count::~__shared_count() {
   0.518 us [10471] |         std::_Sp_counted_base::_M_release();
   1.104 us [10471] |       } /* std::__shared_count::~__shared_count */
   1.532 us [10471] |     } /* std::__shared_ptr::~__shared_ptr */
   2.029 us [10471] |   } /* std::shared_ptr::~shared_ptr */
            [10471] |   std::shared_ptr::~shared_ptr() {
            [10471] |     std::__shared_ptr::~__shared_ptr() {
            [10471] |       std::__shared_count::~__shared_count() {
            [10471] |         std::_Sp_counted_base::_M_release() {
   3.493 us [10471] |           operator delete(0x1209910);
   0.349 us [10471] |           operator delete(0x122d630);
   7.118 us [10471] |         } /* std::_Sp_counted_base::_M_release */
   7.524 us [10471] |       } /* std::__shared_count::~__shared_count */
   7.888 us [10471] |     } /* std::__shared_ptr::~__shared_ptr */
   8.250 us [10471] |   } /* std::shared_ptr::~shared_ptr */
  24.897 us [10471] | } /* main */
```

Report Command (Summary)

```
$ uftrace report
 Total time   Self time  Nr. called  Function
  ==========  ==========  ==========  ====================================
   24.897 us    3.656 us           1  main
   10.279 us    0.859 us           2  std::shared_ptr::~shared_ptr
    9.420 us    0.792 us           2  std::__shared_ptr::~__shared_ptr
    8.628 us    0.992 us           2  std::__shared_count::~__shared_count
    7.636 us    3.794 us           2  std::_Sp_counted_base::_M_release
    6.450 us    0.468 us           1  std::shared_ptr::shared_ptr
    5.982 us    0.912 us           1  std::__shared_ptr::__shared_ptr
    5.195 us    5.195 us           2  operator new
    4.907 us    1.591 us           1  std::__shared_count::__shared_count
    3.842 us    3.842 us           2  operator delete
    2.177 us    0.442 us           1  std::shared_ptr::shared_ptr
    1.735 us    0.422 us           1  std::__shared_ptr::__shared_ptr
    1.313 us    0.664 us           1  std::__shared_count::__shared_count
    0.649 us    0.649 us           1  std::_Sp_counted_base::_M_add_ref_copy
    0.456 us    0.456 us           1  std::_Sp_counted_ptr::_Sp_co
```

Analyzing Clang/LLVM

```
$ uftrace -t 2ms -F cc1_main ./clang fibonacci.c
# DURATION    TID     FUNCTION
            [ 9045] | cc1_main() {
            [ 9045] |   clang::CompilerInvocation::CreateFromArgs() {
   2.270 ms [ 9045] |     ParseCodeGenArgs();
   8.653 ms [ 9045] |   } /* clang::CompilerInvocation::CreateFromArgs */
            [ 9045] |   clang::ExecuteCompilerInvocation() {
            [ 9045] |     clang::CompilerInstance::ExecuteAction() {
   2.185 ms [ 9045] |       clang::FrontendAction::BeginSourceFile();
            [ 9045] |       clang::FrontendAction::Execute() {
            [ 9045] |         clang::CodeGenAction::ExecuteAction() {
            [ 9045] |           clang::ASTFrontendAction::ExecuteAction() {
            [ 9045] |             clang::ParseAST() {
            [ 9045] |               clang::Parser::Initialize() {
   3.841 ms [ 9045] |                 clang::Preprocessor::Lex();
   3.887 ms [ 9045] |               } /* clang::Parser::Initialize */
            [ 9045] |               clang::BackendConsumer::HandleTranslationUnit() {
            [ 9045] |                 clang::EmitBackendOutput() {
            [ 9045] |                   llvm::LLVMTargetMachine::addPassesToEmitFile() {
   2.044 ms [ 9045] |                     addPassesToGenerateCode();
   2.068 ms [ 9045] |                   } /* llvm::LLVMTargetMachine::addPassesToEmitFile */
            [ 9045] |                   llvm::legacy::PassManager::run() {
   2.196 ms [ 9045] |                     llvm::legacy::PassManagerImpl::run();
   2.196 ms [ 9045] |                   } /* llvm::legacy::PassManager::run */
   5.053 ms [ 9045] |                 } /* clang::EmitBackendOutput */
   5.076 ms [ 9045] |               } /* clang::BackendConsumer::HandleTranslationUnit */
  23.361 ms [ 9045] |             } /* clang::ParseAST */
  23.385 ms [ 9045] |           } /* clang::ASTFrontendAction::ExecuteAction */
  23.385 ms [ 9045] |         } /* clang::CodeGenAction::ExecuteAction */
  23.386 ms [ 9045] |       } /* clang::FrontendAction::Execute */
  25.651 ms [ 9045] |     } /* clang::CompilerInstance::ExecuteAction */
  25.667 ms [ 9045] |   } /* clang::ExecuteCompilerInvocation */
  34.368 ms [ 9045] | } /* cc1_main */
```

Analyzing V8 JavaScript Engine

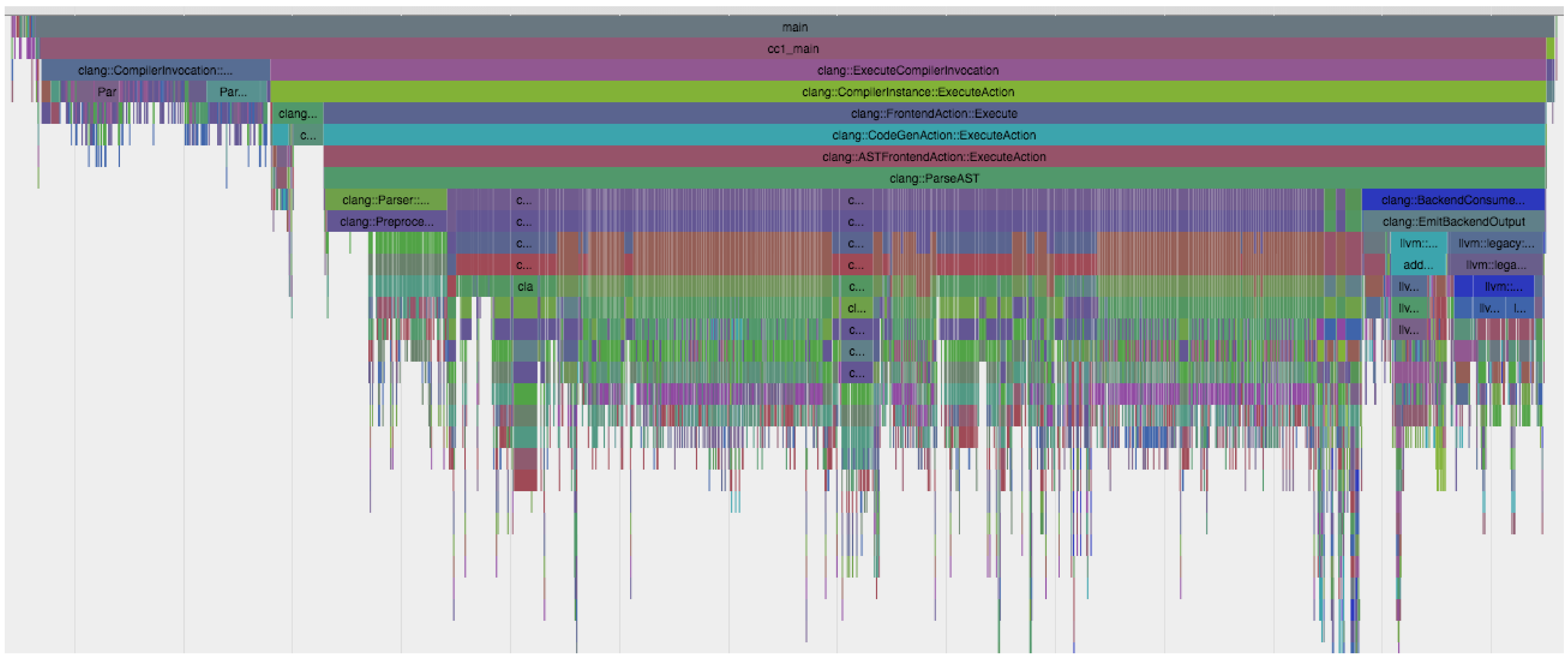
```
$ uftrace -F v8::Shell::Main -t 50ms ./d8 fibonacci.js
# DURATION    TID     FUNCTION
            [13090] | v8::Shell::Main() {
            [13090] |   v8::Isolate::New() {
            [13090] |     v8::internal::Isolate::Init() {
            [13090] |       v8::internal::Heap::CreateHeapObjects() {
            [13090] |         v8::internal::Heap::CreateInitialObjects() {
            [13090] |           v8::internal::Heap::CreateFixedStubs() {
  57.433 ms [13090] |             v8::internal::CodeStub::GenerateStubsAheadOfTime();
  57.472 ms [13090] |           } /* v8::internal::Heap::CreateFixedStubs */
  60.804 ms [13090] |         } /* v8::internal::Heap::CreateInitialObjects */
  60.902 ms [13090] |       } /* v8::internal::Heap::CreateHeapObjects */
 100.935 ms [13090] |     } /* v8::internal::Isolate::Init */
 100.992 ms [13090] |   } /* v8::Isolate::New */
            [13090] |   v8::Shell::RunMain() {
            [13090] |     v8::Shell::CreateEvaluationContext() {
            [13090] |       v8::Context::New() {
            [13090] |         v8::NewContext() {
            [13090] |           v8::internal::Bootstrapper::CreateEnvironment() {
            [13090] |             v8::internal::Genesis::Genesis() {
 191.952 ms [13090] |               v8::internal::Genesis::InstallNatives();
 203.549 ms [13090] |             } /* v8::internal::Genesis::Genesis */
 203.569 ms [13090] |           } /* v8::internal::Bootstrapper::CreateEnvironment */
 203.575 ms [13090] |         } /* v8::NewContext */
 203.575 ms [13090] |       } /* v8::Context::New */
 203.721 ms [13090] |     } /* v8::Shell::CreateEvaluationContext */
            [13090] |     v8::SourceGroup::Execute() {
            [13090] |       v8::Shell::ExecuteString() {
            [13090] |         v8::Script::Run() {
  55.348 ms [13090] |           v8::internal::Execution::Call();
  55.350 ms [13090] |         } /* v8::Script::Run */
  55.910 ms [13090] |       } /* v8::Shell::ExecuteString */
  55.940 ms [13090] |     } /* v8::SourceGroup::Execute */
 259.667 ms [13090] |   } /* v8::Shell::RunMain */
 361.898 ms [13090] | } /* v8::Shell::Main */
```

Using Google Chrome Tracing Facility

The recorded data by uftrace can be dumped as JSON style output that can be loaded by Google chrome browser.
The output JSON file can also be converted into HTML file that can be easily shared as a web link.

Compilation Procedure Study of Clang/LLVM

Internal function trace records of Clang/LLVM that compiles a target source code

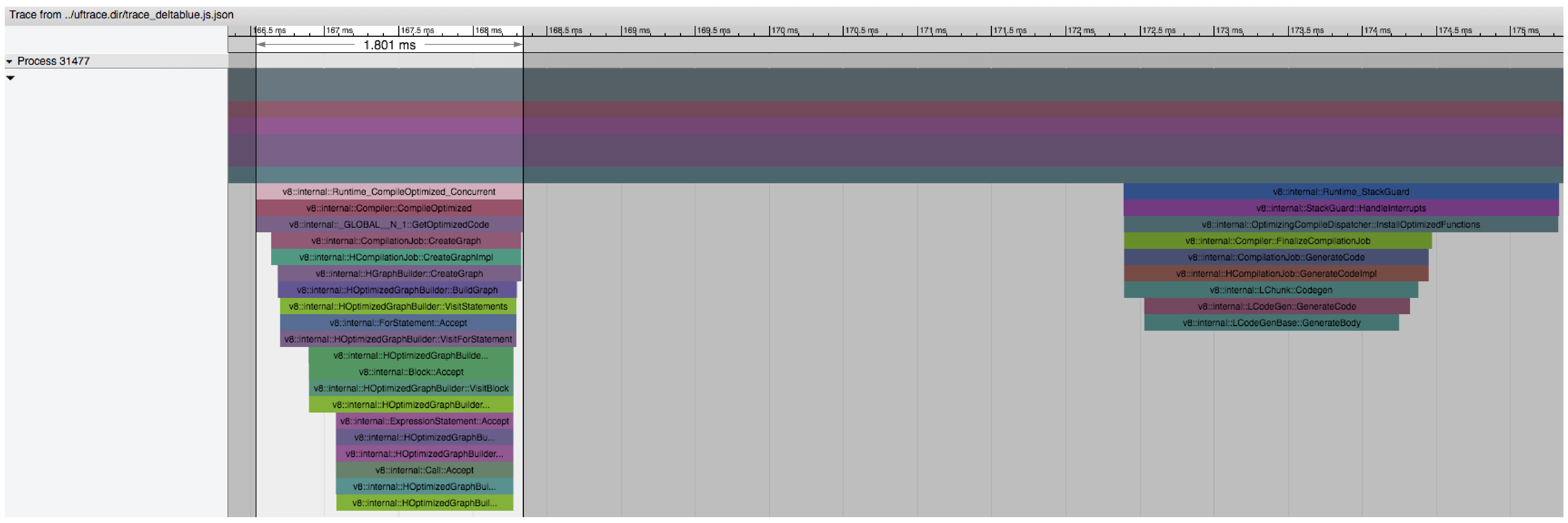

main
cc1_main
clang::CompilerInvocation::...
clang::ExecuteCompilerInvocation
Par
Par...
clang::CompilerInstance::ExecuteAction
clang...
clang::FrontendAction::Execute
c...
clang::CodeGenAction::ExecuteAction
clang::ASTFrontendAction::ExecuteAction
clang::ParseAST
clang::Parser::...
clang::Preproce...
clang::BackendConsume...
clang::EmitBackendOutput
llvm::...
llvm::legacy...
add...
llvm::lega...
llvm::...
cla
cl...

Concurrent Compilation Study of V8 JavaScript Engine

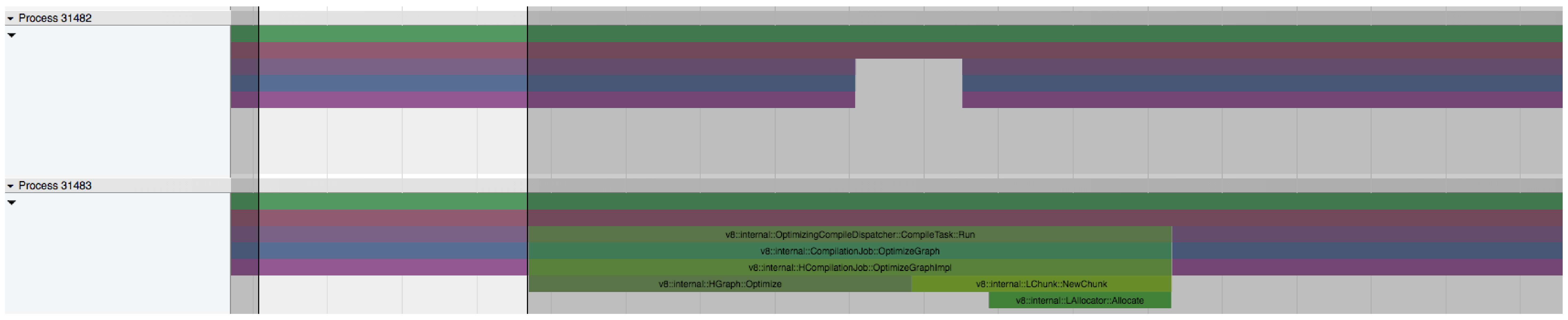
V8 optimizes hot functions following 3 steps as below:

1. V8 maintains worker threads to help master thread. (v8::internal::Runtime_CompileOptimized_Concurrent)
2. Master thread sometimes send optimization requests to worker threads to hot JavaScript functions. (v8::internal::OptimizingCompileDispatcher::CompileTask::Run)
3. If the optimization is done by worker threads, master thread installs the newly optimized code. (v8::internal::OptimizingCompileDispatcher::InstallOptimizedFunctions)

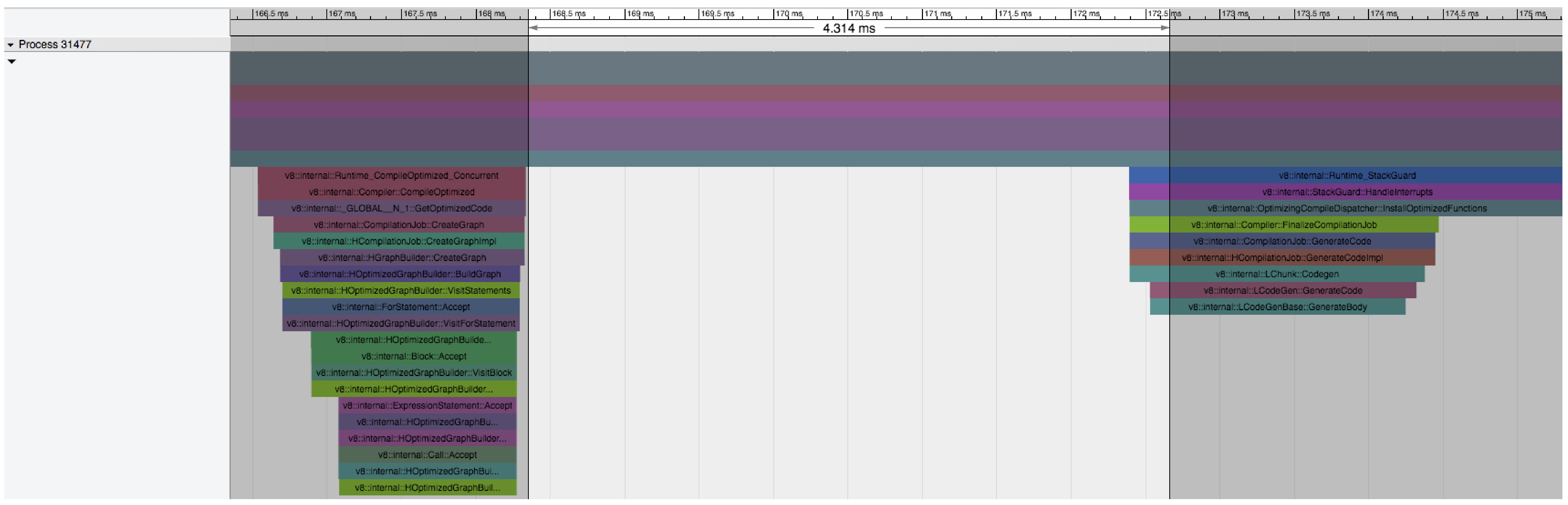
Step 1. v8::internal::Runtime_CompileOptimized_Concurrent() by **master thread**


Trace from ../uftrace.dir/trace_deltablue.js.json
1.801 ms
Process 31477
v8::internal::Runtime_CompileOptimized_Concurrent
v8::internal::Compiler::CompileOptimized
v8::internal::_GLOBAL__N_1::GetOptimizedCode
v8::internal::CompilationJob::CreateGraph
v8::internal::HCompilationJob::CreateGraphImpl
v8::internal::HGraphBuilder::CreateGraph
v8::internal::HOptimizedGraphBuilder::BuildGraph
v8::internal::HOptimizedGraphBuilder::VisitStatements
v8::internal::ForStatement::Accept
v8::internal::HOptimizedGraphBuilder::VisitForStatement
v8::internal::HOptimizedGraphBuilde...
v8::internal::Block::Accept
v8::internal::HOptimizedGraphBuilder::VisitBlock
v8::internal::HOptimizedGraphBuilder...
v8::internal::ExpressionStatement::Accept
v8::internal::HOptimizedGraphBu...
v8::internal::HOptimizedGraphBuilder...
v8::internal::Call::Accept
v8::internal::HOptimizedGraphBui...
v8::internal::HOptimizedGraphBuil...
v8::internal::Runtime_StackGuard
v8::internal::StackGuard::HandleInterrupts
v8::internal::OptimizingCompileDispatcher::InstallOptimizedFunctions
v8::internal::Compiler::FinalizeCompilationJob
v8::internal::CompilationJob::GenerateCode
v8::internal::HCompilationJob::GenerateCodeImpl
v8::internal::LChunk::Codegen
v8::internal::LCodeGen::GenerateCode
v8::internal::LCodeGenBase::GenerateBody

Step 2. v8::internal::OptimizingCompileDispatcher::CompileTask::Run() by **background worker thread** (pid 31483)


Process 31482
Process 31483
v8::internal::OptimizingCompileDispatcher::CompileTask::Run
v8::internal::CompilationJob::OptimizeGraph
v8::internal::HCompilationJob::OptimizeGraphImpl
v8::internal::HGraph::Optimize
v8::internal::LChunk::NewChunk
v8::internal::LAllocator::Allocate

Step 3. v8::internal::OptimizingCompileDispatcher::InstallOptimizedFunctions() by **master thread**


4.314 ms
Process 31477
v8::internal::Runtime_CompileOptimized_Concurrent
v8::internal::Compiler::CompileOptimized
v8::internal::_GLOBAL__N_1::GetOptimizedCode
v8::internal::CompilationJob::CreateGraph
v8::internal::HCompilationJob::CreateGraphImpl
v8::internal::HGraphBuilder::CreateGraph
v8::internal::HOptimizedGraphBuilder::BuildGraph
v8::internal::HOptimizedGraphBuilder::VisitStatements
v8::internal::ForStatement::Accept
v8::internal::HOptimizedGraphBuilder::VisitForStatement
v8::internal::HOptimizedGraphBuilde...
v8::internal::Block::Accept
v8::internal::HOptimizedGraphBuilder::VisitBlock
v8::internal::HOptimizedGraphBuilder...
v8::internal::ExpressionStatement::Accept
v8::internal::HOptimizedGraphBu...
v8::internal::HOptimizedGraphBuilder...
v8::internal::Call::Accept
v8::internal::HOptimizedGraphBui...
v8::internal::HOptimizedGraphBuil...
v8::internal::Runtime_StackGuard
v8::internal::StackGuard::HandleInterrupts
v8::internal::OptimizingCompileDispatcher::InstallOptimizedFunctions
v8::internal::Compiler::FinalizeCompilationJob
v8::internal::CompilationJob::GenerateCode
v8::internal::HCompilationJob::GenerateCodeImpl
v8::internal::LChunk::Codegen
v8::internal::LCodeGen::GenerateCode
v8::internal::LCodeGenBase::GenerateBody