

# Measuring the User Debugging Experience

Greg Bedwell

Sony Interactive Entertainment



introducing  
**DExTer**

introducing

# Debugging **EX**perience **T**ester

introducing

# Debugging **EX**perience **T**ester

**(currently in internal review in preparation for open sourcing)**

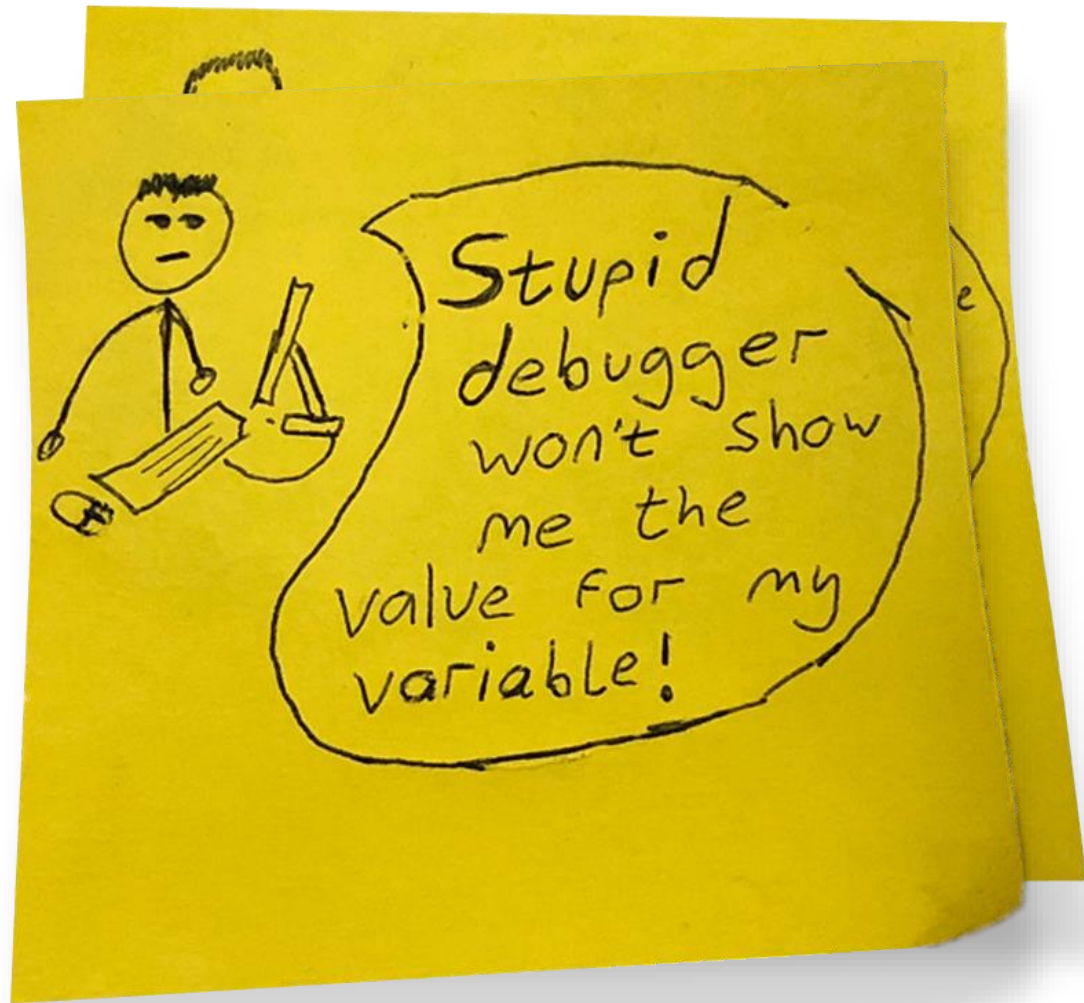
A suite of tools for viewing and measuring the quality of the user debugging experience

Developed by Greg Bedwell & Stephen Wilks

Python v2.7 and v3.5+



This  
DW\_TAG\_variable  
is missing a  
DW\_AT\_location  
after SROA runs



```
1 // =====
2 NOINLINE int foo(int x) {
3     return x * 2; // DexWatch('x')
4 }
5
6 int main(int argc, char**) {
7     return foo(argc);
8 }
9 // =====
```



```
1 // =====
2 NOINLINE int foo(int x) {
3     return x * 2; // DexWatch('x')
4 }
5
6 int main(int argc, char**) {
7     return foo(argc);
8 }
9 // =====
```

```
$ python dexter.py view example.json
## BEGIN ##
[1, "main", "/home/greg/dexter/tests/example/test.cpp", 7, 14, "BREAKPOINT", "FUNC", {}]
. [2, "foo(int)", "/home/greg/dexter/tests/example/test.cpp", 3, 10, "BREAKPOINT", "FUNC", {"x": "1"}]
[3, "main", "/home/greg/dexter/tests/example/test.cpp", 7, 3, "STEP", "FUNC", {}]
## END (3 steps) ##
```



```
1 // =====
2 NOINLINE int foo(int x) {
3     return x * 2; // DexWatch('x')
4 }
5
6 int main(int argc, char**) {
7     return foo(argc);
8 }
9 // =====
```

```
$ python dexter.py view example.json
## BEGIN ##
[1, "main", "/home/greg/dexter/tests/example/test.cpp", 7, 14, "BREAKPOINT", "FUNC", {}]
. [2, "foo(int)", "/home/greg/dexter/tests/example/test.cpp", 3, 10, "BREAKPOINT", "FUNC", {"x": "1"}]
[3, "main", "/home/greg/dexter/tests/example/test.cpp", 7, 3, "STEP", "FUNC", {}]
## END (3 steps) ##
```

```
1 // =====
2 NOINLINE int foo(int x) {
3     return x * 2; // DexWatch('x')
4 }
5
6 int main(int argc, char**) {
7     return foo(argc);
8 }
9 // =====
```

```
{
  "step_index": 2,
  "step_kind": "FUNC",
  "stop_reason": "BREAKPOINT",
  "frames": [
    {
      "function": "foo(int)",
      "is_inlined": false,
      "loc": {
        "path": "/home/greg/dexter/tests/example/test.cpp",
        "lineno": 3,
        "column": 10
      }
    },
    {
      "function": "main",
      "is_inlined": false,
      "loc": {
        "path": "/home/greg/dexter/tests/example/test.cpp",
        "lineno": 7,
        "column": 10
      }
    }
  ],
  "watches": {
    "x": {
      "expression": "x",
      "value": "1",
      "type": "int",
      "could_evaluate": true,
      "is_optimized_away": false,
      "could_retrieve_data": true
    }
  }
}
```

```
1 // =====
2 NOINLINE int foo(int x) {
3     return x * 2; // DexWatch('x')
4 }
5
6 int main(int argc, char**) {
7     return foo(argc);
8 }
9
10 // DexExpectWatchValue('x', '1', on_line=3)
11
12 // DexExpectStepKind('FUNC_EXTERNAL', 0)
13 // DexExpectStepKind('SAME', 0)
14 // DexExpectStepKind('BACKWARD', 0)
15 // =====
```

```
1 // =====
2 NOINLINE int foo(int x) {
3     return x * 2; // DexWatch('x')
4 }
5
6 int main(int argc, char**) {
7     return foo(argc);
8 }
9
10 // DexExpectWatchValue('x')
11
12 // DexExpectStepKind('FUNC')
13 // DexExpectStepKind('SAME')
14 // DexExpectStepKind('BACKWARD')
15 // =====
```

-00 -g

```
Greg's Linux Machine
$ python dexter.py view example-00g.json
## BEGIN ##
[1, "main", "/home/greg/dexter/tests/example/test.cpp", 7, 14, "BREAKPOINT", "FUNC", {}]
. [2, "foo(int)", "/home/greg/dexter/tests/example/test.cpp", 3, 10, "BREAKPOINT", "FUNC", {"x": "1"}]
[3, "main", "/home/greg/dexter/tests/example/test.cpp", 7, 3, "STEP", "FUNC", {}]
## END (3 steps) ##
$ python dexter.py calc -v example-00g.json
example-00g.json = 0/10 (0.0000)

step kind differences [0/3]
  BACKWARD:
    0

  FUNC_EXTERNAL:
    0

  SAME:
    0

test.cpp:3 [x] [0/7]
  expected encountered values:
    1
```

```
1 // =====
2 NOINLINE int foo(int x) {
3     return x * 2; // DexWatch('x')
4 }
5
6 int main(int argc, char**) {
7     return foo(argc);
8 }
9
10 // DexExpectWatchValue('x')
11
12 // DexExpectStepKind('FUNC')
13 // DexExpectStepKind('SAME')
14 // DexExpectStepKind('BACKWARD')
15 // =====
```

-00 -g

```
Greg's Linux Machine
$ python dexter.py view example-00g.json
## BEGIN ##
[1, "main", "/home/greg/dexter/tests/example/test.cpp", 7, 14, "BREAKPOINT", "FUNC", {}]
. [2, "foo(int)", "/home/greg/dexter/tests/example/test.cpp", 3, 10, "BREAKPOINT", "FUNC", {"x": "1"}]
[3, "main", "/home/greg/dexter/tests/example/test.cpp", 7, 3, "STEP", "FUNC", {}]
## END (3 steps) ##
$ python dexter.py calc -v example-00g.json
example-00g.json = 0/10 (0.0000)

step kind differences [0/3]
  BACKWARD:
    0

  FUNC_EXTERNAL:
    0

  SAME:
    0

test.cpp:3 [x] [0/7]
  expected encountered values:
    1
```

-O2 -g

```
1 // =====
2 NOINLINE int foo(int x) {
3     return x * 2; // DexWatch('x')
4 }
5
6 int main(int argc, char**) {
7     return foo(argc);
8 }
9
10 // DexExpectWatchValue('x')
11
12 // DexExpectStepKind('FUNC')
13 // DexExpectStepKind('SAME')
14 // DexExpectStepKind('BACKWARD')
15 // =====
```

```
Greg's Linux Machine
$ python dexter.py view example-O2g.json
## BEGIN ##
[1, "main", "/home/greg/dexter/tests/example/test.cpp", 7, 10, "BREAKPOINT", "FUNC", {}]
[2, "foo(int)", "/home/greg/dexter/tests/example/test.cpp", 3, 12, "BREAKPOINT", "FUNC", {"x": "1"}]
## END (2 steps) ##
$ python dexter.py calc -v example-O2g.json
example-O2g.json = 0/10 (0.0000)

step kind differences [0/3]
  BACKWARD:
    0

  FUNC_EXTERNAL:
    0

  SAME:
    0

test.cpp:3 [x] [0/7]
  expected encountered values:
    1
```

-O2 -gmlt

```
1 // =====
2 NOINLINE int foo(int x) {
3     return x * 2; // DexWatch('x')
4 }
5
6 int main(int argc, char**) {
7     return foo(argc);
8 }
9
10 // DexExpectWatchValue('x')
11
12 // DexExpectStepKind('FUNC')
13 // DexExpectStepKind('SAME')
14 // DexExpectStepKind('BACKWARD')
15 // =====
```

```
Greg's Linux Machine
$ python dexter.py view example-O2gmlt.json
## BEGIN ##
[1, "main", "/home/greg/dexter/tests/example/test.cpp", 7, 10, "BREAKPOINT", "FUNC", {}]
[2, "foo(int)", "/home/greg/dexter/tests/example/test.cpp", 3, 12, "BREAKPOINT", "FUNC", {"x": null}]
## END (2 steps) ##
$ python dexter.py calc -v example-O2gmlt.json
example-O2gmlt.json = 5/10 (0.5000)

step kind differences [0/3]
  BACKWARD:
    0

  FUNC_EXTERNAL:
    0

  SAME:
    0

test.cpp:3 [x] [5/7]
could not evaluate:
  step 2 [+5]
```

-02 -gmlt

```
1 // =====
2 NOINLINE int foo(int x) {
3     return x * 2; // DexWatch('x')
4 }
5
6 int main(int argc, char**) {
7     return foo(argc);
8 }
9
10 // DexExpectWatchValue('x')
11
12 // DexExpectStepKind('FUNC')
13 // DexExpectStepKind('SAME')
14 // DexExpectStepKind('BACKWARD')
15 // =====
```

```
Greg's Linux Machine
$ python dexter.py view example-02gmlt.json
## BEGIN ##
[1, "main", "/home/greg/dexter/tests/example/test.cpp", 7, 10, "BREAKPOINT", "FUNC", {}]
[2, "foo(int)", "/home/greg/dexter/tests/example/test.cpp", 3, 12, "BREAKPOINT", "FUNC", {"x": null}]
## END (2 steps) ##
$ python dexter.py calc -v example-02gmlt.json
example-02gmlt.json = 5/10 (0.5000)

step kind differences [0/3]
  BACKWARD:
    0

  FUNC_EXTERNAL:
    0

  SAME:
    0

test.cpp:3 [x] [5/7]
could not evaluate:
step 2 [+5]
```



```
Command Prompt
$ py -2 dexter.py list-debuggers

lldb [lldb] NO (Module use of python36.dll conflicts with this version of Python.)
vs2015 [Visual Studio 2015] YES (14.0)
vs2017 [Visual Studio 2017] YES (15.0)

$ py -3 dexter.py list-debuggers

lldb [lldb] YES (lldb version 7.0.0 (https://git.llvm.org/git/lldb.git/ revision bf66622dc962721edc30f4c67c7cc3db7a78684e))
vs2015 [Visual Studio 2015] YES (14.0)
vs2017 [Visual Studio 2017] YES (15.0)

$
```

```
Greg's Linux Machine
$ python2 dexter.py list-debuggers

lldb [lldb] YES (lldb version 7.0.0 (https://git.llvm.org/git/lldb.git/ revision d0c2df1fedf5dad964c54a954a309472795e886f))
vs2015 [Visual Studio 2015] NO (No module named win32com.client)
vs2017 [Visual Studio 2017] NO (No module named win32com.client)

$ python3 dexter.py list-debuggers

lldb [lldb] NO (dynamic module does not define module export function (PyInit__lldb))
vs2015 [Visual Studio 2015] NO (No module named 'win32com')
vs2017 [Visual Studio 2017] NO (No module named 'win32com')

$ █
```

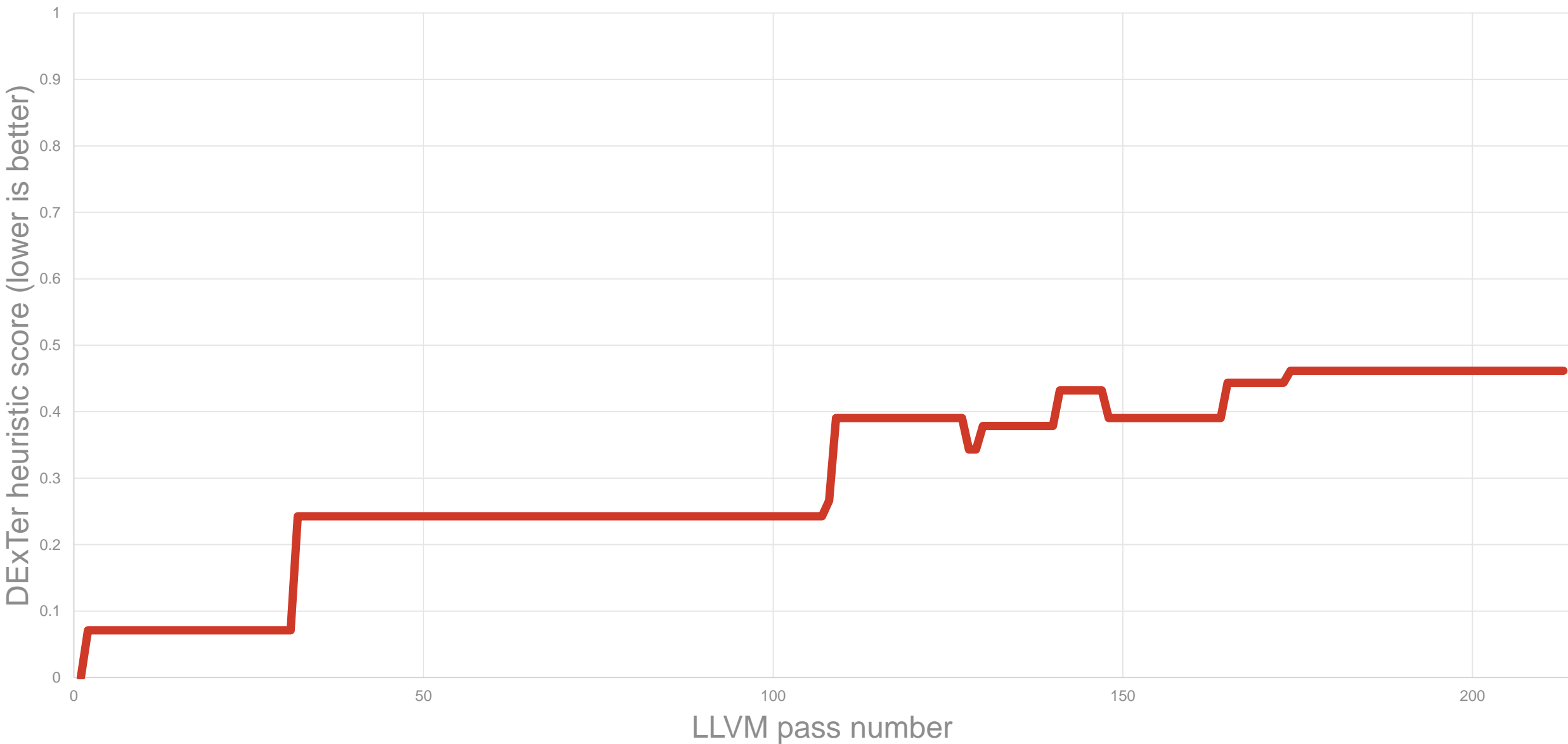
Clang trunk over time (“Fibonacci” test: -O2 -g)



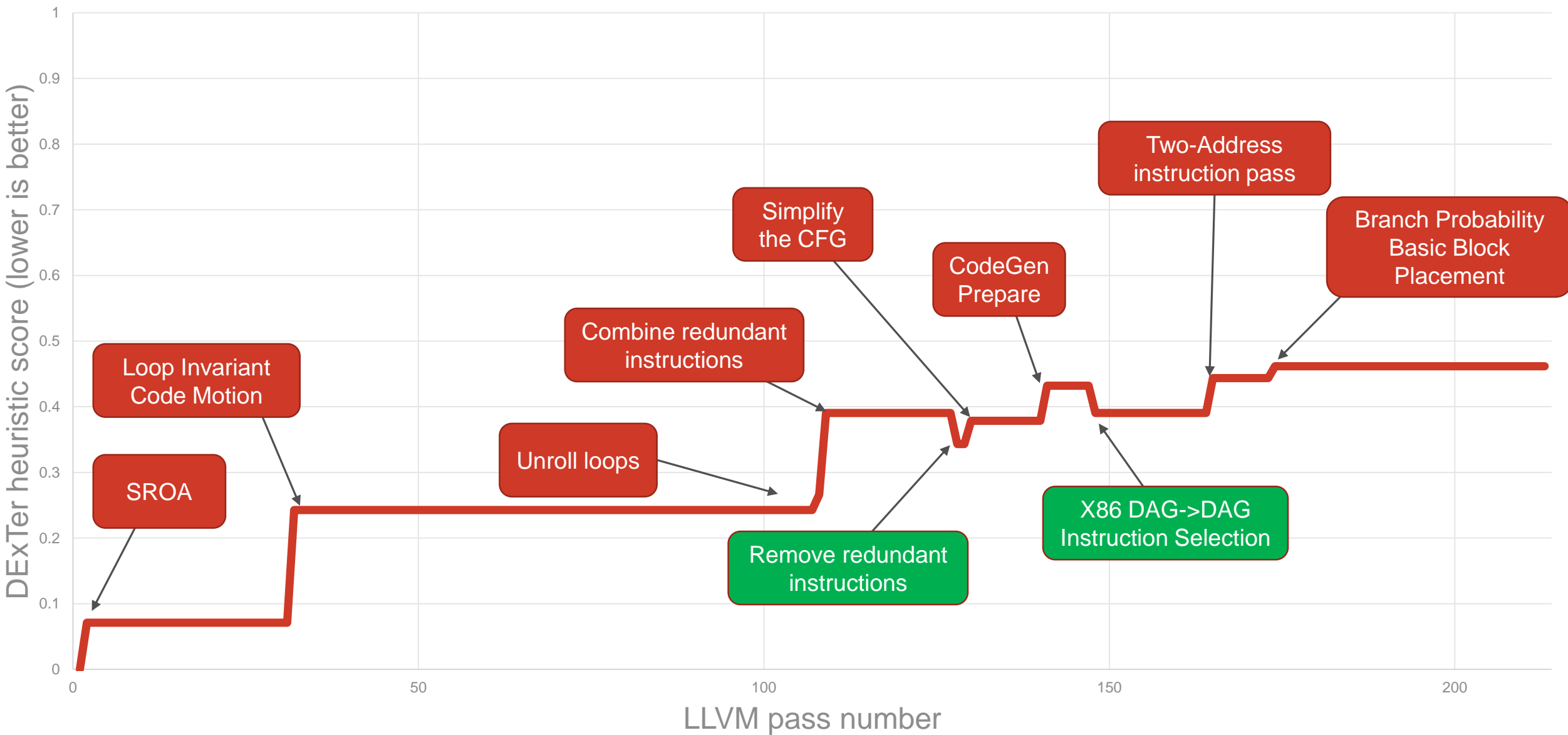
```
$ python dexter.py clang-opt-bisect --builder clang --debugger lldb --cflags="-O2 -g" --tests-directory=tests/fibonacci
```

```
pass 1/213 = 0/169 (0.0000, 0.0000) [Simplify the CFG on function (_Z9FibonacciRi)]
pass 2/213 = 12/169 (0.0710, 0.0710) [SROA on function (_Z9FibonacciRi)]
pass 3/213 = 12/169 (0.0710, 0.0000) [Early CSE on function (_Z9FibonacciRi)]
pass 4/213 = 12/169 (0.0710, 0.0000) [Simplify the CFG on function (main)]
pass 5/213 = 12/169 (0.0710, 0.0000) [SROA on function (main)]
pass 6/213 = 12/169 (0.0710, 0.0000) [Early CSE on function (main)]
pass 7/213 = 12/169 (0.0710, 0.0000) [Infer set function attributes on module (/home/greg/dexter/tests/fibonacci/test.cpp)]
pass 8/213 = 12/169 (0.0710, 0.0000) [Interprocedural Sparse Conditional Constant Propagation on module (/home/greg/dexter/tests/fibonacci/test.cpp)]
pass 9/213 = 12/169 (0.0710, 0.0000) [Called Value Propagation on module (/home/greg/dexter/tests/fibonacci/test.cpp)]
pass 10/213 = 12/169 (0.0710, 0.0000) [Global Variable Optimizer on module (/home/greg/dexter/tests/fibonacci/test.cpp)]
pass 11/213 = 12/169 (0.0710, 0.0000) [Promote Memory to Register on function (_Z9FibonacciRi)]
pass 12/213 = 12/169 (0.0710, 0.0000) [Promote Memory to Register on function (main)]
pass 13/213 = 12/169 (0.0710, 0.0000) [Dead Argument Elimination on module (/home/greg/dexter/tests/fibonacci/test.cpp)]
pass 14/213 = 12/169 (0.0710, 0.0000) [Combine redundant instructions on function (_Z9FibonacciRi)]
pass 15/213 = 12/169 (0.0710, 0.0000) [Simplify the CFG on function (_Z9FibonacciRi)]
pass 16/213 = 12/169 (0.0710, 0.0000) [Combine redundant instructions on function (main)]
pass 17/213 = 12/169 (0.0710, 0.0000) [Simplify the CFG on function (main)]
pass 18/213 = 12/169 (0.0710, 0.0000) [Remove unused exception handling info on SCC (_Z9FibonacciRi)]
pass 19/213 = 12/169 (0.0710, 0.0000) [Function Integration/Inlining on SCC (_Z9FibonacciRi)]
pass 20/213 = 12/169 (0.0710, 0.0000) [Deduce function attributes on SCC (_Z9FibonacciRi)]
pass 21/213 = 12/169 (0.0710, 0.0000) [SROA on function (_Z9FibonacciRi)]
pass 22/213 = 12/169 (0.0710, 0.0000) [Early CSE w/ MemorySSA on function (_Z9FibonacciRi)]
pass 23/213 = 12/169 (0.0710, 0.0000) [Speculatively execute instructions if target has divergent branches on function (_Z9FibonacciRi)]
pass 24/213 = 12/169 (0.0710, 0.0000) [Jump Threading on function (_Z9FibonacciRi)]
pass 25/213 = 12/169 (0.0710, 0.0000) [Value Propagation on function (_Z9FibonacciRi)]
pass 26/213 = 12/169 (0.0710, 0.0000) [Simplify the CFG on function (_Z9FibonacciRi)]
pass 27/213 = 12/169 (0.0710, 0.0000) [Combine redundant instructions on function (_Z9FibonacciRi)]
pass 28/213 = 12/169 (0.0710, 0.0000) [Tail Call Elimination on function (_Z9FibonacciRi)]
pass 29/213 = 12/169 (0.0710, 0.0000) [Simplify the CFG on function (_Z9FibonacciRi)]
pass 30/213 = 12/169 (0.0710, 0.0000) [Reassociate expressions on function (_Z9FibonacciRi)]
pass 31/213 = 12/169 (0.0710, 0.0000) [Rotate Loops on loop]
pass 32/213 = 41/169 (0.2426, 0.1716) [Loop Invariant Code Motion on loop]
pass 33/213 = 41/169 (0.2426, 0.0000) [Unswitch loops on loop]
pass 34/213 = 41/169 (0.2426, 0.0000) [Simplify the CFG on function (_Z9FibonacciRi)]
```

Effect of LLVM passes ("Fibonacci" test: -O2 -g)



# Effect of LLVM passes ("Fibonacci" test: -O2 -g)



# Overall effect of LLVM passes across multiple DExTer tests (-O2 -g)

