

What is claimed is:

1. A method comprising:
processing, using a processor coupled to a memory, an unstructured binary code
region that is single-threaded;
5 determining a slice criterion for the region;
determining a call edge, a return edge, and a fallthrough pseudo-edge for the region
based on analysis of the region at a binary level;
generating a context-sensitive code slice based on the call edge, the return edge,
the fallthrough pseudo-edge, and the slice criterion; and
10 saving the context-sensitive code slice in the memory.
2. The method of claim 1 including generating the context-sensitive code slice
by traversing the return edge.
- 15 3. The method of claim 2 including determining a recursive subslice for the
region based on the return edge.
4. The method of claim 3, wherein the determination of the recursive subslice
is not based on metadata for another call edge.
20
5. The method of claim 3 including traversing the fallthrough pseudo-edge
after determining the recursive subslice.
6. The method of claim 1 including:
25 determining a recursive function call; and
generating a context-insensitive code slice for the recursive function call but not
attempting to generate a context-sensitive code slice for the recursive function call.
7. The method of claim 1 including generating, based on a time limit
30 expiration, a context-insensitive code slice for a second call edge by treating the second
call edge as a normal control flow edge.
8. The method of claim 6 including generating the context-insensitive code

slice by treating another return edge included in the region as a normal control flow edge and ignoring another fallthrough pseudo-edge included in the region.

9. The method of claim 1 including parallelizing the region, based on the
5 context-sensitive code slice, to determine first and second threads from the single-threaded region; wherein the first and second threads may be executed in parallel.

10. The method of claim 1 including generating the context-sensitive code slice
10 at runtime.

11. The method of claim 10, wherein the region is a hot region.

12. The method of claim 10 including determining the call edge, the return
15 edge, and the fallthrough pseudo-edge based on a control flow graph for the region.

13. The method of claim 1, wherein the region is completely included in a
micro-operation thread formed at runtime.

14. The method of claim 1, wherein the determination of the slice is not based
20 on metadata that includes debug information.

15. An article comprising a medium storing instructions that enable a
processor-based system to:

25 process, using a processor of the system, an unstructured binary code region that is single-threaded;

determine a return edge for the region based on analysis of the region at a binary level;

generate a context-sensitive code slice based on the return edge; and

store, in a memory of the system, the context-sensitive code slice.

30 16. The article of claim 15, further storing instructions that enable the processor based-system to:

determine a call edge and a fallthrough pseudo-edge for the region based on

analysis of the region at the binary level; and

generate the slice based on the call edge and the fallthrough pseudo-edge.

17. The article of claim 15, further storing instructions that enable the
5 processor based-system to:

generate the slice by traversing the return edge; and

generate, based on a time limit, a context-insensitive slice by treating another call
edge included in the region as a normal control flow edge.

18. An apparatus comprising:

a memory to receive an unstructured binary code region that is single-threaded;

a processor, coupled to the memory, to:

determine a return edge for the region based on analysis of the region at a binary
level; and

15 generate a context-sensitive code slice based on the return edge.

19. The apparatus of claim 18, wherein the processor is to:

determine a call edge and a fallthrough pseudo-edge for the region based on
analysis of the region at the binary level; and

20 generate the slice based on the call edge and the fallthrough pseudo-edge.

20. The apparatus of claim 18, wherein the processor is to:

generate the slice by traversing the return edge; and

25 generate, based on a time limit, a context-insensitive code slice by treating another
call edge included in the region as a normal control flow edge.

What is claimed is:

1. A method comprising:

processing, using a processor coupled to a memory, an unstructured binary code region that is single-threaded;

determining a slice criterion for the region;

determining a call edge, a return edge, and a fallthrough pseudo-edge for the region based on analysis of the region at a binary level;

traversing the fallthrough pseudo-edge, which immediately follows, in binary order, a call instruction corresponding to the call edge;

generating a context-sensitive code slice based on the call edge, the return edge, the fallthrough pseudo-edge, and the slice criterion; and

saving the context-sensitive code slice in the memory.

2. The method of claim 1 including generating the context-sensitive code slice by traversing the return edge.

3. The method of claim 2 including determining a recursive subslice for the region based on the return edge and ignoring one of the call edge and another call edge.

4. The method of claim 3, wherein the determination of the recursive subslice is not based on metadata for the another call edge.

5. The method of claim 3 including traversing the fallthrough pseudo-edge after determining the recursive subslice.

6. The method of claim 1 including:

determining a recursive function call; and

generating a context-insensitive code slice for the recursive function call but not attempting to generate a context-sensitive code slice for the recursive function call.

7. The method of claim 1 including generating, based on a time limit expiration, a context-insensitive code slice for a second call edge by treating the second call edge as a normal control flow edge.

8. The method of claim 6 including generating the context-insensitive code slice by treating another return edge included in the region as a normal control flow edge and ignoring another fallthrough pseudo-edge included in the region.

9. The method of claim 1 including parallelizing the region, based on the context-sensitive code slice, to determine first and second threads from the single-threaded region; wherein the first and second threads may be executed in parallel.

10. The method of claim 1 including generating the context-sensitive code slice at runtime.

11. The method of claim 10, wherein the region is a hot region.

12. The method of claim 10 including determining the call edge, the return edge, and the fallthrough pseudo-edge based on a control flow graph for the region.

13. The method of claim 1, wherein the region is completely included in a micro-operation thread formed at runtime.

14. The method of claim 1, wherein the determination of the slice is not based on metadata that includes debug information.

15. An article comprising a medium storing instructions that enable a processor-based system to:

process, using a processor of the system, an unstructured binary code region that is single-threaded;

determine a return edge, a call edge, and a fallthrough pseudo-edge for the region based on analysis of the region at a binary level;

traverse the fallthrough pseudo-edge, which immediately follows, in binary order, a call instruction corresponding to the call edge;

generate a context-sensitive code slice based on the return edge, the call edge, and the fallthrough pseudo-edge; and

store, in a memory of the system, the context-sensitive code slice.

16. The article of claim 15, further storing instructions that enable the processor based-system to:

generate the slice by traversing the return edge; and

generate, based on a time limit, a context-insensitive slice by treating another call edge included in the region as a normal control flow edge.

17. An apparatus comprising:

at least one memory to receive an unstructured binary code region that is single-threaded;

a processor, coupled to the at least one memory, configured to:

determine a return edge, a call edge, and a fallthrough pseudo-edge for the region based on analysis of the region at a binary level;

traverse the fallthrough pseudo-edge, which immediately follows, in binary order, a call instruction corresponding to the call edge;

generate a context-sensitive code slice based on the return edge, the call edge, and the fallthrough pseudo-edge; and

save the context-sensitive code slice in the at least one memory.

18. The apparatus of claim 17, wherein the processor is configured to:

determine another call edge for the region based on analysis of the region at the binary level; and

generate the slice based on the another call edge.

19. The apparatus of claim 17, wherein the processor is configured to generate, based on a time limit, a context-insensitive code slice by treating another call edge included in the region as a normal control flow edge.