[PATCH] Let debuggers know when a tail call happens

11/16/2018 08:10 PM - alanwu (Alan Wu)

Description

Background
The popular debugger "byebug" relies on tracepoint events to implement a few core functionality such as "next". The "next" command needs to pause execution when the VM reaches the next line on the same stack frame. As there is no direct way, using the public API, to tell the execution progress of a frame, Byebug keeps track of the currently executing frame by maintaining a counter using the "call" and "return" tracepoint event.

https://github.com/deivid-rodriguez/byebug/blob/58ee5114aa856ec49483532a86fd159a877dd6ab/ext/byebug/byebug.c#L162-L170

Byebug's counter becomes incorrect when the interpreter performs a tail call, since after a tail call, the "return" event doesn't fire for the caller. #15303

This causes the "next" command to misbehave when stepping over code that performs tail call.

Proposed solution

I implemented a new method in TracePoint that lets Byebug, or any other debugger to differentiate between a normal method call and a tail call. Using this API, Byebug can maintain the counter properly in the event of a tail call.

Here are some other solutions

- Some sort of public api that exposes execution progress of control frames. This might take a lot of effort.
- Some sort of public api that give ids to stack frames. After a tail call, debuggers can use this to tell that a frame is different, even though it's on the same location on the stack.
- Turning off tail call optimization in Forwardable. This side steps the problem but the core issue still remain. Third party libraries could still use tco and byebug wouldn't be able to work with them.

Here are some downsides to my solution:

- This is a pretty niche feature that only debuggers want
- I'm adding a vm frame flag. While we have a few more bits to play with, it is a limited resource.

History

#1 - 11/19/2018 04:22 PM - alanwu (Alan Wu)
- Description updated

#2 - 11/21/2018 07:05 PM - ko1 (Koichi Sasada)
- Assignee set to ko1 (Koichi Sasada)

fix #15303 (invoke return event) doesn't solve it?

BTW, if you provide an API, sample code will help us to understand.

#3 - 11/21/2018 11:58 PM - alanwu (Alan Wu)
Yes if the return events can fire as usual even when tail calls happen it would be perfect and would fix #15303.
I don't see a way to fire the return events in the normal order without extra allocations, though. Maybe it's possible?

We could fire the return event before the tailcall happens, #return_value method would be confusing though.

BTW, if you provide an API, sample code will help us to understand.

Thank you for letting me know! I will remember to also post examples on the tracker next time.

The following demonstrates how this API could be used:

```ruby
require 'forwardable'

trace = TracePoint.new(:call) do |tp|
  puts if tp.tailcall?
    "a tail call happened!"
  else
    "a normal call happened"
  end
end

trace.enable do
  # call some methods
  printer = String.new
  printer.extend Forwardable
  printer.def_delegator "STDOUT", "puts"
  printer.puts "Howdy!"
end
```

---

#4 - 11/22/2018 05:49 PM - alanwu (Alan Wu)
- File deleted (0001-Add-a-tail-call-predicate-for-TracePoint.patch)
- File 0001-Add-a-tail-call-predicate-for-TracePoint.patch added

I noticed that the original patch doesn't compile with GCC. `-Werror=declaration-after-statement` doesn't seem to have any effect on Clang: [https://godbolt.org/z/8Qykk4](https://godbolt.org/z/8Qykk4) (switch to gcc to see it fail to compile. It compiles on Clang) seems to be a known issue: [https://bugs.llvm.org/show_bug.cgi?id=27493](https://bugs.llvm.org/show_bug.cgi?id=27493)

#5 - 12/05/2018 04:59 AM - alanwu (Alan Wu)
Could I ask for some feedback on this?

#6 - 12/05/2018 07:05 AM - ko1 (Koichi Sasada)
how about [https://bugs.ruby-lang.org/issues/15303#note-2](https://bugs.ruby-lang.org/issues/15303#note-2)?

#7 - 12/29/2019 10:33 AM - hsbt (Hiroshi SHIBATA)
- Tags set to patch, tracepoint

#8 - 08/31/2020 08:17 PM - jeremyevans0 (Jeremy Evans)
- Status changed from Open to Closed

As tailcall optimization was removed in 241dced625f9ba8a4071954579778a0940e75179, I think this can be closed.

Files

<table>
<thead>
<tr>
<th>Patch File</th>
<th>Size</th>
<th>Date</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001-Add-a-tail-call-predicate-for-TracePoint.patch</td>
<td>18.1 KB</td>
<td>11/22/2018</td>
<td>alanwu (Alan Wu)</td>
</tr>
</tbody>
</table>