Ruby - Bug #9114

InstructionSequence.compile w/tailcall_optimization: true, trace_instruction: false not working as expected

11/16/2013 02:38 AM - garysweaver (Gary Weaver)

Status:	Rejected		
Priority:	Normal		
Assignee:	ko1 (Koichi Sasada)		
Target version:			
ruby -v:	ruby 2.0.0p247 (2013-06-27 revision 41674) [x86_64-darwin11.4.2]	**Backport:**	1.9.3: UNKNOWN, 2.0.0: UNKNOWN

Description

Code to reproduce is a recursive sort I wrote; I was trying to compile RubyVM::InstructionSequence with tailcall_optimization: true, trace_instruction: false, which has worked before, but not for this case.

```
method_string = <<RUBY
def recursively_sort(obj, in_sort_by=false)
if obj.respond_to?(:sort_by)
obj.sort_by{|*args|args.map{|v|recursively_sort(v, true)}}
end rescue nil
in_sort_by ? (obj.to_s rescue obj.inspect rescue nil) : obj
end
RUBY

RubyVM::InstructionSequence.new(method_string, nil, nil, nil, tailcall_optimization: true, trace_instruction: false).eval

a = eval '[{b:['*1000 + '2,1' + '],a:1}]'*1000

recursively_sort a
```

results in error for latest releases of Ruby 1.9.3 and 2.0.0:

```
1.9.3-p448 :014 >   recursively_sort a
SystemStackError: stack level too deep
from /Users/gary/.rvm/rubies/ruby-1.9.3-p448/lib/ruby/1.9.1/irb/workspace.rb:80
Maybe IRB bug!

2.0.0p247 :014 >   recursively_sort a
SystemStackError: stack level too deep
from /Users/gary/.rvm/rubies/ruby-2.0.0-p247/lib/ruby/2.0.0/irb/workspace.rb:86
Maybe IRB bug!
```

I had assumed that since the blocks were defined within the tail call optimized method that referenced the method compiled with TCO, it would still be tail call optimized.

Did I do something wrong/is there a suggested workaround?

Thanks for looking at this!

History

#1 - 11/16/2013 02:55 AM - garysweaver (Gary Weaver)

btw- that method is incorrect, and when I wrote a better/working method, now it doesn't have that error in 2.0.0p247 but still fails with 1.9.3-p448. Not sure why:

```
method_string = <<RUBY
def recursively_sort(obj)
case obj
when Array
obj.map!{|v| recursively_sort(v)}.sort_by!{|v| (v.to_s rescue nil) }
when Hash
obj = Hash[Hash[obj.map{|k,v| [recursively_sort(k),recursively_sort(v)]}].sort_by{|k,v| [(k.to_s rescue nil), (v.to_s rescue nil)]}]
else
```

obj
end
end
RUBY

RubyVM::InstructionSequence.new(method_string, nil, nil, nil, tailcall_optimization: true, trace_instruction: false).eval

a = eval '[{b:['*1000 + '2,1' + '],a:1}]'*1000

recursively_sort a

So, there is something about the "bad" method in the original post that keeps TCO from working even in Ruby 2. Thanks.

#2 - 11/16/2013 06:54 AM - drbrain (Eric Hodel)

Which call is in tail position for this method? I'm not seeing it.

#3 - 11/16/2013 09:47 AM - drbrain (Eric Hodel)

Here's an update where a call is in tail position, but from the output you'll see only the initial call uses the tail call optimization.

```
source = <<-SOURCE
def fact n, acc = 1
  if n.zero?
    acc
  else
    fact n - 1, acc * n
  end
end

fact 10000
SOURCE

i_seq = RubyVM::InstructionSequence.new source, nil, nil, nil,
  tailcall_optimization: false

puts i_seq.disasm

begin
  value = i_seq.eval

  p value
rescue SystemStackError => e
  puts e
end

i_seq = RubyVM::InstructionSequence.new source, nil, nil, nil,
  tailcall_optimization: true

puts i_seq.disasm

begin
  value = i_seq.eval

  p value
rescue SystemStackError => e
  puts e
end
```

#4 - 11/19/2013 06:10 AM - drbrain (Eric Hodel)

- Status changed from Open to Assigned

- Assignee set to ko1 (Koichi Sasada)

#5 - 11/28/2013 07:43 AM - garysweaver (Gary Weaver)

Eric,

My apologizes as I probably wasted your time with that. The problem with the code you posted is that for TCO you still have to specify trace_instruction: false. If you execute the following, it is fine in Ruby 2.0.0p247 at least, unless it doesn't work in core:

source = <<-SOURCE
def fact n, acc = 1
if n.zero?
acc

```
else
fact n - 1, acc * n
end
end

fact 10000
SOURCE

i_seq = RubyVM::InstructionSequence.new source, nil, nil, nil,
tailcall_optimization: true, trace_instruction: false

puts i_seq.disasm

begin
value = i_seq.eval

p value
rescue SystemStackError => e
puts e
end
```

I wasn't doing a tail call, which was my problem I think :(, so can close.

#6 - 12/02/2013 01:43 PM - drbrain (Eric Hodel)

- Status changed from Assigned to Rejected

No time was wasted, I wanted to be sure your use-case was understood.

As you requested, this issue is now closed.