Ruby master - Feature #1218
New method needed to set and get the current recursion limit

02/27/2009 12:13 AM - Anonymous

Status: Rejected
Priority: Normal
Assignee: ko1 (Koichi Sasada)
Target version: 2.0.0

Description

=begin
Hi, I would like to propose the addition of two methods for setting and getting the recursion limit:

recursion_limit(limit)
recursion_limit => returns an Integer value

At this time, I have been experimenting with recursion to compute the nth fibonacci number using ruby 1.9.1 ( i.e ruby 1.9.1p0 (2009-01-30 revision 21907) ). These tests are being ran on a Apple PowerMac G5 2.5 Ghz Quad 4 GB RAM and a MacBook Pro 2.8 GHz Intel Core 2 Duo 4 GB RAM. Also, I was able to successfully produce a result using an iteration which I'm attaching to verify the correctness of the other two algorithms:

action: ./fibonacci-recursive.rb 10000

result:

```
./fibonacci-recursive.rb:10:in fibonacci': stack level too deep (SystemStackError)
  from ./fibonacci-recursive.rb:10:in fibonacci'
  from ./fibonacci-recursive.rb:10:in fibonacci'
  from ./fibonacci-recursive.rb:10:in fibonacci'
  from ./fibonacci-recursive.rb:10:in fibonacci'
  from ./fibonacci-recursive.rb:10:in fibonacci'
  from ./fibonacci-recursive.rb:10:in fibonacci'
  from ./fibonacci-recursive.rb:10:in fibonacci'
  ... 8174 levels...
  from ./fibonacci-recursive.rb:10:in fibonacci'
  from ./fibonacci-recursive.rb:10:in fibonacci'
  from ./fibonacci-recursive.rb:10:in fibonacci'
  from ./fibonacci-recursive.rb:10:in fibonacci'
  from ./fibonacci-recursive.rb:18:in `'
```

action: ./fibonacci-tail-recursive.rb 10000

result:

```
./fibonacci-tail-recursive.rb:10:in fibonacci_helper': stack level too deep (SystemStackError)
  from ./fibonacci-tail-recursive.rb:10:in fibonacci_helper'
  from ./fibonacci-tail-recursive.rb:10:in fibonacci_helper'
  from ./fibonacci-tail-recursive.rb:10:in fibonacci_helper'
  from ./fibonacci-tail-recursive.rb:10:in fibonacci_helper'
  from ./fibonacci-tail-recursive.rb:10:in fibonacci_helper'
  from ./fibonacci-tail-recursive.rb:10:in fibonacci_helper'
  from ./fibonacci-tail-recursive.rb:10:in fibonacci_helper'
  ... 7265 levels...
  from ./fibonacci-tail-recursive.rb:10:in fibonacci_helper'
  from ./fibonacci-tail-recursive.rb:10:in fibonacci_helper'
  from ./fibonacci-tail-recursive.rb:17:in fibonacci'
  from ./fibonacci-tail-recursive.rb:24:in `'
```

action: ./fibonacci-iterative.rb 10000

result:

```
33644764876431783266621612005107543310302148460680063906564769974680081442166623681555955136337340255820
```

03/17/2020
#1 - 02/27/2009 12:33 AM - daz (Dave B)

=begin
See if this post helps:
http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-talk/89931
=end

#2 - 02/27/2009 03:32 AM - jredville (Jim Deville)

=begin
-----Original Message-----
From: Dave B [mailto:redmine@ruby-lang.org]
Sent: Thursday, February 26, 2009 7:33 AM
To: ruby-core@ruby-lang.org
Subject: [ruby-core:22545] [Feature #1218] New method needed to set and get the current recursion limit

Issue #1218 has been updated by Dave B.

See if this post helps:
http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-talk/89931

That wouldn't help for Ruby on Windows. It would also be nice to have something tied to the code, not the system.
JD
=end

#3 - 02/28/2009 09:27 PM - Anonymous

=begin
Hi Dave, thanks for responding to the post. However, this doesn't resolve the issue that I'm seeing after increasing the stack size. There should be some class methods for obtaining and setting the recursion limit.

-Conrad
=end

#4 - 03/01/2009 11:23 PM - austin (Austin Ziegler)

=begin
On Sat, Feb 28, 2009 at 7:26 AM, Conrad Taylor redmine@ruby-lang.org wrote:

Issue #1218 has been updated by Conrad Taylor.

=end
Hi Dave, thanks for responding to the post. However, this does resolve the issue that I'm seeing after increasing the stack size. There should be some class methods for obtaining and setting the recursion limit.

You can't change the recursion limit from within the program on all operating systems.

-austin

Austin Ziegler * halostatue@gmail.com * http://www.halostatue.ca/
* austin@halostatue.ca * http://www.halostatue.ca/feed/
* austin@zieglers.ca

=begin

#5 - 03/02/2009 05:42 PM - brent (Brent Roman)

=`begin

setrlimit(RLIMIT_STACK, size_in_bytes)

should do what you want under a POSIX compliant OS.

See the manpage for setrlimit and getrlimit

This could easily be made accessible to Ruby code via a ‘C’ extension.

I have found that 1.8 Ruby is very inefficient in its use of stack-space and made some patches that improve this.

With ulimit -s 8192, the attached little script on gives up after:

```
1.6.8-mbari8B:   14374 levels
1.8.6-p287:     7198 levels
1.8.7-p72:      7197 levels
1.8.7-p72-mbari8B: 14750 levels
1.8.6-p287-mbari8B: 14750 levels
1.9.0:       7704 levels (but, see note below)
```

Ruby 1.9 seems completely unaffected by stack size configured via setrlimit (via ulimit -s from the bash shell)

Perhaps 1.9 is allocating the stack on the heap.
If this is the case, you will need some support in the 1.9 core for changing the size of this allocation.

   • brent

Usaku NAKAMURA wrote:

   Issue #1218 has been updated by Conrad Taylor.

   Hi Dave, thanks for responding to the post. However, this does resolve the issue that I'm seeing after increasing the stack size. There should be some class methods for obtaining and setting the recursion limit.

   -Conrad

http://redmine.ruby-lang.org/issues/show/1218

http://redmine.ruby-lang.org

--

View this message in context:
Sent from the ruby-core mailing list archive at Nabble.com.

=end

#6 - 03/02/2009 06:01 PM - nobu (Nobuyoshi Nakada)
Hi,

At Mon, 2 Mar 2009 17:41:03 +0900, Brent Roman wrote in [ruby-core:22619]:

Ruby 1.9 seems completely unaffected by stack size configured via setrlimit
(via ulimit -s from the bash shell)

Perhaps 1.9 is allocating the stack on the heap.

1.9 uses the default of pthread.

--
Nobu Nakada

#7 - 03/02/2009 10:08 PM - Anonymous

austin (Austin Ziegler) Ziegler - I was able to change these limits within Python using FreeBSD, Linux, and Mac OS X operating systems. The first two are production machines and the last is a development machine. Otherwise, the Python examples would have crapped out as did the Ruby recursive examples. Thus, I'm sure that it's possible on most modern platforms to change these limits within the Ruby language.

#8 - 03/03/2009 01:20 AM - brent (Brent Roman)

I don't do much pthreads programming. The call you want seems to be:

int pthread_attr_setstacksize(pthread_attr_t *attr, size_t stacksize)
Set the stack size attribute in a thread attributes object.
(The prospective 'C' extension would have to get at the relevant pthread_attr_t)

Here's the little test script I forgot to attach yesterday:

```ruby
$a=0

def x
  $a += 1
  x
end

begin
  x
rescue SystemStackError
  puts "After #{$a} levels"
  puts $!
end
```

Conrad Taylor-3 wrote:

Issue #1218 has been updated by Conrad Taylor.

austin (Austin Ziegler) Ziegler - I was able to change these limits within Python using FreeBSD, Linux, and Mac OS X operating systems. The first two are production machines and the last is a development machine. Otherwise, the Python examples would have crapped out as did the Ruby recursive examples. Thus, I'm sure that it's possible on most modern platforms to change these limits within the Ruby language.

http://redmine.ruby-lang.org/issues/show/1218
#9 - 03/04/2009 01:44 AM - austin (Austin Ziegler)

> Did you try this on Windows?
>
> I'm not sure it's possible on most modern platforms (and despite what people think and my own personal prejudices, Windows is a modern platform that Rubyists can't afford to ignore).

-austin

---

#10 - 03/05/2009 07:12 PM - Anonymous

> Hi, I do not have access to a Windows machine because I do all my development using Unix environments.

-Conrad

---

#11 - 09/14/2010 04:37 PM - shyouhei (Shyouhei Urabe)

- Status changed from Open to Assigned

---

#12 - 02/13/2012 11:38 PM - mame (Yusuke Endoh)

- Status changed from Assigned to Rejected

> I'm rejecting this feature ticket because no progress has been made for a long time. See [ruby-core:42391].

> I think it is impossible to implement recursion_limit exactly because we cannot know how much stack each C function consumes.

> I expected Python's recursion_limit to fail conservatively (i.e., it will raise an exception for big limit even when it is actually possible). But it was worse than I expected. Python caused SEGV when big limit is specified:

```python
import sys
sys.setrecursionlimit(1000000)
```
def foo():
    foo()
foo() #=> SEGV

I think that this is not a correct way, at least, in Ruby.

--
Yusuke Endoh mame@tsq.ne.jp

<table>
<thead>
<tr>
<th>Files</th>
<th>Size</th>
<th>Date</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>fibonacci-iterative.rb</td>
<td>373 B</td>
<td>02/27/2009</td>
<td>Anonymous</td>
</tr>
<tr>
<td>fibonacci-recursive.rb</td>
<td>408 B</td>
<td>02/27/2009</td>
<td>Anonymous</td>
</tr>
<tr>
<td>fibonacci-tail-recursive.rb</td>
<td>478 B</td>
<td>02/27/2009</td>
<td>Anonymous</td>
</tr>
</tbody>
</table>