Ruby - Bug #1650

Time range === is slow

06/18/2009 09:48 AM - kmeaw (Dmitry Bilunov)

Status:	Rejected		
Priority:	Normal		
Assignee:	akr (Akira Tanaka)		
Target version:	1.9.3		
ruby -v:	ruby 1.9.1p129 (2009-05-12 revision 23412) [x86_64-linux]	**Backport:**	

Description

=begin
The following program runs N times slower in ruby19 compared to ruby18.
N depends on the input range size.

dev@rails ~ $ time ruby19 -rtime -e '(Time.now - 1000000 .. Time.now) === (Time.now - 3)'

real 0m0.723s
user 0m0.709s
sys 0m0.013s
dev@rails ~ $ time ruby18 -rtime -e '(Time.now - 1000000 .. Time.now) === (Time.now - 3)'

real 0m0.043s
user 0m0.039s
sys 0m0.005s
=end

History

#1 - 06/19/2009 03:06 AM - drbrain (Eric Hodel)

=begin
Adding Time#to_int allows Range#include? to use its fast path, but I don't know if it is appropriate. There's a Process::Status#to_int so I suppose it would be valid for Time too.
=end

#2 - 06/20/2009 02:16 AM - matz (Yukihiro Matsumoto)

=begin
Hi,

In message "Re: [ruby-core:23908] [Bug #1650] Time range === is slow"
on Thu, 18 Jun 2009 09:48:30 +0900, Dmitry Bilunov redmine@ruby-lang.org writes:

|The following program runs N times slower in ruby19 compared to ruby18.
|N depends on the input range size.

1.9 Range#=== now checks according to enumeration (unless both ends
are integers). It might be improved in the future, but until then, I
recommend you to use t3.between?(t1, t2) instead of (t1 .. t2)===t3.

```
matz.
```

=end

#3 - 07/13/2009 09:44 PM - yugui (Yuki Sonoda)

- Status changed from Open to Rejected

=begin

=end

#4 - 12/29/2010 05:45 AM - cdunn2001 (Christopher Dunn)

=begin

That's not what the docs say. They are ambiguous. http://ruby-doc.org/ruby-1.9/classes/Range.html:

- rng.cover?(val) => true or false
  Returns true if obj is between beg and end, i.e beg <= obj <= end (or end exclusive when exclude_end? is true).

- rng.include?(val) => true or false
  Returns true if obj is an element of rng, false otherwise. If beg and end are numeric, comparison is done according magnitude of values.

- rng === obj => true or false
  Returns true if obj is an element of rng, false otherwise. Conveniently, === is the comparison operator used by case statements.

If Range#=== works exactly the same as Range#include?, the docs should say that. If they differ when beg/end are numeric, the docs should say that. I really do not know precisely how Range#=== works from the docs, the web, or this discussion. I have to test it myself. The distinction is critical because it may dramatically impact performance of 'case' statements in the move from 1.8 to 1.9.

Please update the docs with clarification.
=end

#5 - 12/29/2010 06:42 AM - marcandre (Marc-Andre Lafortune)

- Category changed from lib to doc

- Status changed from Rejected to Open

=begin

=end

#6 - 06/26/2011 01:57 PM - naruse (Yui NARUSE)

- Status changed from Open to Assigned

- Assignee set to akr (Akira Tanaka)

#7 - 06/26/2011 04:43 PM - nahi (Hiroshi Nakamura)

- Target version set to 1.9.3

#8 - 06/27/2011 01:59 AM - neleai (Ondrej Bilka)

Ah bug I pointed out in ruby-core:8609
currently there is following error.
-:1:in each': can't iterate from Time (TypeError) from -:1:in include?'
from -:1:in include?' from -:1:in ==='
from -:1:in `'

Anyway what is worse (Time.now - 1000000 .. Time.now) === (Time.now - 3)
returns false as it enumerates Times and third time is usualy few
microseconds off.

To me === expected behaviour is that (a..b)===c should call c.between?(a,b).

Problem at this time was that matz wanted in 1.9 ("a1".."a11") === "a9"
return true (which in 1.8 returns false.
This also could be slow for example for ("a1".."a1000000000") === "ab"

My proposal was call between and redefine String#between to do desired
comparison as in patch attached at ruby-core:8609
BTW patch is more general than one-char comparsion as is in
range_include

On Sun, Jun 26, 2011 at 04:43:06PM +0900, Hiroshi NAKAMURA wrote:

> Issue #1650 has been updated by Hiroshi NAKAMURA.
>
> Target version set to 1.9.3
>
> ---
>
> Bug #1650: Time range === is slow
> http://redmine.ruby-lang.org/issues/1650
>
> Author: Dmitry Bilunov
> Status: Assigned
> Priority: Normal
> Assignee: Akira Tanaka

Category: DOC
Target version: 1.9.3
ruby -v: ruby 1.9.1p129 (2009-05-12 revision 23412) [x86_64-linux]

=begin
The following program runs N times slower in ruby19 compared to ruby18.
N depends on the input range size.

dev@rails ~ $ time ruby19 -rtime -e '(Time.now - 1000000 .. Time.now) === (Time.now - 3)'

real   0m0.723s
user   0m0.709s
sys   0m0.013s
dev@rails ~ $ time ruby18 -rtime -e '(Time.now - 1000000 .. Time.now) === (Time.now - 3)'

real   0m0.043s
user   0m0.039s
sys   0m0.005s
=end

--
http://redmine.ruby-lang.org

--

Network failure - call NBC

#9 - 07/12/2011 08:33 PM - mame (Yusuke Endoh)

- Status changed from Assigned to Rejected

Hello,

> If Range#=== works exactly the same as Range#include?, the docs should say that. If they differ when beg/end are numeric, the docs should say that.

That is an implementation detail, I think. Not a bug. So I'm closing
the ticket.
If you want to clarify it as a spec, please register another ticket
into feature tracker.

Thank you,

--
Yusuke Endoh mame@tsg.ne.jp