# Ruby master - Bug #13917

## Comparable#clamp is slower than using Array#min,max.

09/19/2017 02:26 AM - kei-s (Kei Shiratsuchi)

<table>
<thead>
<tr>
<th>Status:</th>
<th>Rejected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td>nobu (Nobuyoshi Nakada)</td>
</tr>
<tr>
<td>Target version:</td>
<td>ruby -v: Backport: 2.2: UNKNOWN, 2.3: UNKNOWN, 2.4: UNKNOWN</td>
</tr>
</tbody>
</table>

**Description**

Comparable#clamp is slower than using Array#min,max.
(I noticed it by @onk's tweet. [https://twitter.com/onk/status/907856892604461056](https://twitter.com/onk/status/907856892604461056))

**Performance**

<table>
<thead>
<tr>
<th></th>
<th>user</th>
<th>system</th>
<th>total</th>
<th>real</th>
</tr>
</thead>
<tbody>
<tr>
<td>minmax</td>
<td>0.740000</td>
<td>0.000000</td>
<td>0.740000 (0.732744)</td>
<td></td>
</tr>
<tr>
<td>clamp</td>
<td>2.060000</td>
<td>0.010000</td>
<td>2.070000 (2.072794)</td>
<td></td>
</tr>
</tbody>
</table>

**Test Code**

```ruby
require 'benchmark'
Benchmark.bmbm do |x|
  v = Random.rand(-10..110)
  x.report "minmax:" do
    10000000.times { [99, [0, v].max].min }
  end
  x.report "clamp:" do
    10000000.times { v.clamp(0, 99) }
  end
end
```

**Patch**

I made patch for it. But I'm not sure this is good way.
[https://gist.github.com/kei-s/b303aca105df5c26be9c98f833db80f7#file-compar-diff](https://gist.github.com/kei-s/b303aca105df5c26be9c98f833db80f7#file-compar-diff)

**After**

<table>
<thead>
<tr>
<th></th>
<th>user</th>
<th>system</th>
<th>total</th>
<th>real</th>
</tr>
</thead>
<tbody>
<tr>
<td>minmax</td>
<td>0.820000</td>
<td>0.000000</td>
<td>0.820000 (0.822517)</td>
<td></td>
</tr>
<tr>
<td>clamp</td>
<td>1.090000</td>
<td>0.000000</td>
<td>1.090000 (1.087491)</td>
<td></td>
</tr>
</tbody>
</table>

Other benchmark for this patch is here.
[https://gist.github.com/kei-s/0c34cbe4e21a499601e8247077629082](https://gist.github.com/kei-s/0c34cbe4e21a499601e8247077629082)

**Questions**

1. Should clamp version be faster than Array#min/max version?
   Array#min/max version would have overhead of array creation.

2. Is OPTIMIZED_CMP in cmpint best way?
   Some method doesn't pass cmpint (e.g. Integer#>). But OPTIMIZED_CMP checks Integer.

09/02/2022
i can explain why Array#min/max isn't much slower, because it was optimized to not create Array overhead WHEN using variables (interesting it isn't optimized when only using literals)

RubyVM::InstructionSequence.compile("[4,5].max").disasm
#0002 duparray [4, 5]
#0004 opt_send_without_block <callinfo!mid:max, argc:0, ARGS_SIMPLE>, <callcache>

and this:

puts RubyVM::InstructionSequence.compile("x=4;[x,5].max").disasm
#0002 putobject 4
#0004 setlocal_OP__WC__0 3
#0006 getlocal_OP__WC__0 3
#0008 putobject 5
#0010 opt_newarray_max 2

clamp itself doesn't seem to be optimized like that

Thank you, Hanmac. I understand why Array#min/max is so fast.
I guess clamp would be implemented in numeric.c and so on to be as fast as Array#min/max.

Hanmac (Hans Mackowiak) wrote:

i can explain why Array#min/max isn't much slower, because it was optimized to not create Array overhead WHEN using variables (interesting it isn't optimized when only using literals)

It's expected behavior.
doc/NEWS-2.4.0 says

* In some condition, `x, y].max` and `x, y].min` are optimized
so that a temporal array is not created. The concrete condition is
an implementation detail: currently, the array literal must have no
splat, must have at least one expression but literal, the length must
be <= 0x100, and Array#max and min must not be redefined. It will work
in most casual and real-life use case where it is written with intent
to `Math.max(x, y)`.

Array#[min,max] are optimized because they are used frequently. I don't think clamp is used that much.

The main topic "Comparable#clamp is slower than using Array#min,max." is valid.
@nobu (Nobuyoshi Nakada) Could you check the patch?

It was same or slower with gcc 7.2.0.
<table>
<thead>
<tr>
<th>name</th>
<th>ruby 2.5.0dev (2017-10-08 trunk 60140) [x86_64-linux]</th>
<th>built-ruby</th>
</tr>
</thead>
<tbody>
<tr>
<td>comparable_object_between</td>
<td>2.600</td>
<td>2.736</td>
</tr>
<tr>
<td>comparable_object_clamp</td>
<td>3.590</td>
<td>3.579</td>
</tr>
<tr>
<td>comparable_object_less</td>
<td>1.632</td>
<td>1.679</td>
</tr>
</tbody>
</table>

Speedup ratio: compare with the result of `ruby 2.5.0dev (2017-10-08 trunk 60140) [x86_64-linux]` (greater is better)

<table>
<thead>
<tr>
<th>name</th>
<th>built-ruby</th>
</tr>
</thead>
<tbody>
<tr>
<td>comparable_object_between</td>
<td>0.950</td>
</tr>
<tr>
<td>comparable_object_clamp</td>
<td>1.003</td>
</tr>
<tr>
<td>comparable_object_less</td>
<td>0.972</td>
</tr>
</tbody>
</table>

#7 - 08/25/2019 08:16 PM - jeremyevans0 (Jeremy Evans)
- Status changed from Assigned to Rejected