Ruby - Bug #13343

Improve Hash#merge performance

03/21/2017 12:49 AM - watson1978 (Shizuo Fujita)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:		**Backport:**	2.2: UNKNOWN, 2.3: UNKNOWN, 2.4: UNKNOWN

Description

Hash#merge will be faster around 60%.

Before

```
                user     system      total        real
Hash#merge   0.160000   0.020000   0.180000 (  0.182357)
```

After

```
                user     system      total        real
Hash#merge   0.110000   0.010000   0.120000 (  0.114404)
```

Test code

```
require 'benchmark'

Benchmark.bmbm do |x|
  hash1 = {}
  100.times { |i| hash1[i.to_s] = i }
  hash2 = {}
  100.times { |i| hash2[(i*2).to_s] = i*2 }

  x.report "Hash#merge" do
    10000.times do
      hash1.merge(hash2)
    end
  end
end
```

Patch

The patch is in https://github.com/ruby/ruby/pull/1533

Associated revisions

Revision 9cd66d7022aa2b8aff719a26c594efc9c3797ec1 - 05/20/2017 09:23 AM - watson1978 (Shizuo Fujita)

Improve Hash#merge performance

- hash.c (rb_hash_merge): use rb_hash_dup() instead of rb_obj_dup() to duplicate
  Hash object. rb_hash_dup() is faster duplicating function for Hash object
  which got rid of Hash#initialize_dup method calling.

  Hash#merge will be faster around 60%.
  [ruby-dev:50026] [Bug #13343] [Fix GH-1533]

Before

```
            user     system      total        real
```

Hash#merge 0.160000 0.020000 0.180000 (0.182357)

After

```
                user     system      total        real
```

Hash#merge 0.110000 0.010000 0.120000 (0.114404)

Test code

require 'benchmark'

Benchmark.bmbm do |x|
hash1 = {}
100.times { |i| hash1[i.to_s] = i }
hash2 = {}
100.times { |i| hash2[(i*2).to_s] = i*2 }

x.report "Hash#merge" do
10000.times do
hash1.merge(hash2)
end
end
end

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@58811 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 9cd66d70 - 05/20/2017 09:23 AM - watson1978 (Shizuo Fujita)

Improve Hash#merge performance

- hash.c (rb_hash_merge): use rb_hash_dup() instead of rb_obj_dup() to duplicate
  Hash object. rb_hash_dup() is faster duplicating function for Hash object
  which got rid of Hash#initialize_dup method calling.

  Hash#merge will be faster around 60%.
  [ruby-dev:50026] [Bug #13343] [Fix GH-1533]

Before

```
                user     system      total        real
```

Hash#merge 0.160000 0.020000 0.180000 (0.182357)

After

```
                user     system      total        real
```

Hash#merge 0.110000 0.010000 0.120000 (0.114404)

Test code

require 'benchmark'

Benchmark.bmbm do |x|
hash1 = {}
100.times { |i| hash1[i.to_s] = i }
hash2 = {}
100.times { |i| hash2[(i*2).to_s] = i*2 }

x.report "Hash#merge" do
10000.times do
hash1.merge(hash2)
end
end
end

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@58811 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 03/27/2017 05:21 AM - normalperson (Eric Wong)

watson1978@gmail.com wrote:

> https://bugs.ruby-lang.org/issues/13343
>
> Hash#merge will be faster around 60%.

+Cc ruby-core, since your post was English (and I don't read Japanese)

This is promising!

> The patch is in https://github.com/ruby/ruby/pull/1533

We need to check for redefinition of initialize_dup and
initialize_copy methods in Hash for this to be correct.

Unfortunately for people optimizing Ruby, corner-case
redefinition checks are probably necessary :<

Also, I wonder if we can improve rb_funcall to better support
inline caching. rb_funcall API is also bad since it cannot use
inline cache for method lookup. Maybe a better C API can be
introduced for faster function calls from C.

Note: I checked commit c5d74afdb4cfea2a4c9ff432d9da82f0649a1e67
by having a "fetch = +refs/pull/*:refs/remotes/ruby/pull/*"
line in a "remote" section of my .git/config. I did not
use any proprietary API or JavaScript to view your changes.

#2 - 03/27/2017 05:21 AM - normalperson (Eric Wong)

watson1978@gmail.com wrote:

> https://bugs.ruby-lang.org/issues/13343
>
> Hash#merge will be faster around 60%.

+Cc ruby-core, since your post was English (and I don't read Japanese)

This is promising!

> The patch is in https://github.com/ruby/ruby/pull/1533

We need to check for redefinition of initialize_dup and
initialize_copy methods in Hash for this to be correct.

Unfortunately for people optimizing Ruby, corner-case
redefinition checks are probably necessary :<

Also, I wonder if we can improve rb_funcall to better support
inline caching. rb_funcall API is also bad since it cannot use
inline cache for method lookup. Maybe a better C API can be
introduced for faster function calls from C.

Note: I checked commit c5d74afdb4cfea2a4c9ff432d9da82f0649a1e67
by having a "fetch = +refs/pull/*:refs/remotes/ruby/pull/*"
line in a "remote" section of my .git/config. I did not
use any proprietary API or JavaScript to view your changes.

#3 - 03/27/2017 05:52 AM - watson1978 (Shizuo Fujita)

I followed the behavior of Array's methods such as

```
VALUE
rb_ary_sort(VALUE ary)
{
    ary = rb_ary_dup(ary);
```

It does not check whether initialize_dup/initialize_copy were overridden.

#4 - 05/20/2017 09:23 AM - watson1978 (Shizuo Fujita)

- Status changed from Open to Closed

Applied in changeset trunk|r58811.

Improve Hash#merge performance

- hash.c (rb_hash_merge): use rb_hash_dup() instead of rb_obj_dup() to duplicate Hash object. rb_hash_dup() is faster duplicating function for Hash object which got rid of Hash#initialize_dup method calling.

  Hash#merge will be faster around 60%.
  [ruby-dev:50026] [Bug #13343] [Fix GH-1533]

Before

```
                user     system      total        real
```

Hash#merge 0.160000 0.020000 0.180000 (0.182357)

After

```
                user     system      total        real
```

Hash#merge 0.110000 0.010000 0.120000 (0.114404)

Test code

require 'benchmark'

Benchmark.bmbm do |x|
hash1 = {}
100.times { |i| hash1[i.to_s] = i }
hash2 = {}
100.times { |i| hash2[(i*2).to_s] = i*2 }

x.report "Hash#merge" do
10000.times do
hash1.merge(hash2)
end
end
end