Ruby - Bug #13167

Dir.glob is 25x slower since Ruby 2.2

01/30/2017 10:11 AM - ahorek (Pavel Rosický)

Status:	Closed		
Priority:	Normal		
Assignee:	h.shirosaki (Hiroshi Shirosaki)		
Target version:			
ruby -v:	2.4.0	**Backport:**	2.2: UNKNOWN, 2.3: UNKNOWN, 2.4: UNKNOWN

Description

Hello,
we've found a huge speed regression in our Rails app. After some digging the reason is in Dir.glob method which is much slower since Ruby 2.2.6. This is probably Windows only!

This code is used heavily in Rails for partial lookups:

```
Dir.glob(
'c:/test/myapp/app/views/common/_menu_stats{.en,}{.html,}{}{.erb,.builder,.raw,.ruby,.jbuilder,.coffee,}')
```

Comparsion (x64):
jruby 9.1.7.0 2540 i/s
ruby 2.1.5 2568 i/s
ruby 2.1.9 2569 i/s
ruby 2.2.6 99 i/s 25 times slower!
ruby 2.3.3 102 i/s
ruby 2.4.0 103 i/s

I would like to help, but I don't know much about Ruby C internals. Please let me know if you need any additional info. Now we're stuck at 2.1.9 because this issue makes the development on more recent versions unusable.

Related issues:

Related to Ruby - Bug #10015: Performance regression in Dir#[]	**Closed**
Related to Ruby - Feature #13873: Optimize Dir.glob with FNM_EXTGLOB	**Closed**
Related to Ruby - Bug #19042: Bug: Dir.glob ignores subdirectories in alterna...	**Closed**

Associated revisions

Revision 2a119042145973fd66504c70ba0cd86d2571d014 - 09/22/2018 01:11 AM - h.shirosaki (Hiroshi Shirosaki)

dir.c: performance fix with braces

Braces were expended before ruby_glob0(). This caused to call
replace_real_basename() for same plain patterns repeatedly.
Move blace expansion into glob_helper() in ruby_glob0() to reduce
replace_real_basename() call.
This fix changes the order of glob results.
[Feature #13167] [Fix GH-1864]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@64810 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 2a119042145973fd66504c70ba0cd86d2571d014 - 09/22/2018 01:11 AM - h.shirosaki (Hiroshi Shirosaki)

dir.c: performance fix with braces

Braces were expended before ruby_glob0(). This caused to call
replace_real_basename() for same plain patterns repeatedly.
Move blace expansion into glob_helper() in ruby_glob0() to reduce
replace_real_basename() call.
This fix changes the order of glob results.
[Feature #13167] [Fix GH-1864]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@64810 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 2a119042 - 09/22/2018 01:11 AM - h.shirosaki (Hiroshi Shirosaki)

dir.c: performance fix with braces

Braces were expended before ruby_glob0(). This caused to call
replace_real_basename() for same plain patterns repeatedly.
Move blace expansion into glob_helper() in ruby_glob0() to reduce
replace_real_basename() call.
This fix changes the order of glob results.
[Feature #13167] [Fix GH-1864]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@64810 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision b14325443a1e868100c3a5fa4365223b61a77f08 - 09/25/2018 03:31 PM - h.shirosaki (Hiroshi Shirosaki)

dir.c: fix memory leak of glob with braces

join_path uses malloc. So free is required.
[Feature #13167]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@64835 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision b14325443a1e868100c3a5fa4365223b61a77f08 - 09/25/2018 03:31 PM - h.shirosaki (Hiroshi Shirosaki)

dir.c: fix memory leak of glob with braces

join_path uses malloc. So free is required.
[Feature #13167]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@64835 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision b1432544 - 09/25/2018 03:31 PM - h.shirosaki (Hiroshi Shirosaki)

dir.c: fix memory leak of glob with braces

join_path uses malloc. So free is required.
[Feature #13167]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@64835 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision f73d504c6d23059df79d1eea8380c88e3b3b7d1d - 09/25/2018 03:31 PM - h.shirosaki (Hiroshi Shirosaki)

dir.c: fix glob with recursive and brace

Fixed bug that glob with recursive and braces (**/{a,b}) pattern
fails.
[Feature #13167]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@64836 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision f73d504c6d23059df79d1eea8380c88e3b3b7d1d - 09/25/2018 03:31 PM - h.shirosaki (Hiroshi Shirosaki)

dir.c: fix glob with recursive and brace

Fixed bug that glob with recursive and braces (**/{a,b}) pattern
fails.
[Feature #13167]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@64836 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision f73d504c - 09/25/2018 03:31 PM - h.shirosaki (Hiroshi Shirosaki)

dir.c: fix glob with recursive and brace

Fixed bug that glob with recursive and braces (**/{a,b}) pattern
fails.
[Feature #13167]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@64836 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 01/30/2017 11:25 AM - nobu (Nobuyoshi Nakada)

- Related to Bug #10015: Performance regression in Dir#[] added

#2 - 01/30/2017 11:33 AM - nobu (Nobuyoshi Nakada)

- Description updated

#3 - 07/24/2017 07:42 PM - ahorek (Pavel Rosický)

- File logruby24.txt added

- File logruby21.txt added

I used Procmon.exe https://live.sysinternals.com to monitor system calls
and it looks like ruby 2.4.1 is traversing the whole directory tree over and over again for each {} matcher. This should be definitely avoided!

take a look, the same single call for a Dir.glob takes
30 sys-calls on Ruby 2.1.9 but **2086** sys-calls on Ruby 2.4.1!

Ruby 2.1.9 just tries to open all combinations without checking the directory structure

```
c:/test/myapp/app/views/common/_menu_stats.en.html.erb open
c:/test/myapp/app/views/common/_menu_stats.en.html.builder open
...
```

but Ruby 2.4.1 behaves like this

```
c:/ open
c:/ stats
c:/ close
c:/test open
c:/test stats
c:/test close
c:/test/myapp open
c:/test/myapp stats
c:/test/myapp close
...
c:/test/myapp/app/views/common/_menu_stats.en.html.erb open
c:/test/myapp/app/views/common/_menu_stats.en.html.erb stats
c:/test/myapp/app/views/common/_menu_stats.en.html.erb close
*** AND AGAIN ***
c:/ open
c:/ stats
c:/ close
c:/test open
c:/test stats
c:/test close
c:/test/myapp open
c:/test/myapp stats
c:/test/myapp close
...
c:/test/myapp/app/views/common/_menu_stats.en.html.builder open
c:/test/myapp/app/views/common/_menu_stats.en.html.builder stats
c:/test/myapp/app/views/common/_menu_stats.en.html.builder close
*** AND AGAIN ***
c:/ open
c:/ stats
c:/ close
c:/test open
c:/test stats
c:/test close
etc ...
```

#4 - 07/24/2017 08:32 PM - normalperson (Eric Wong)

pdahorek@seznam.cz wrote:

> Bug #13167: Dir.glob is 25x slower since Ruby 2.2
> https://bugs.ruby-lang.org/issues/13167#change-65905

I didn't see a difference in Linux between 2.1 and trunk;
but this seems wrong on Linux and could be optimized:

$ strace -c -e getdents ruby --disable=gems -e 'Dir.glob("/.{flac}")'
=> 935 getdents calls

$ strace -c -e getdents ruby --disable=gems -e 'Dir.glob("/.{flac,ogg}")'

=> 1870 getdents calls

$ strace -c -e getdents ruby --disable=gems -e 'Dir.glob("/.{flac,ogg,mp3}")'
=> 2805 getdents calls

Investigating...

#5 - 07/24/2017 09:41 PM - normalperson (Eric Wong)

Eric Wong normalperson@yhbt.net wrote:

> $ strace -c -e getdents ruby --disable=gems -e 'Dir.glob("/.{flac}")'
> => 935 getdents calls
>
> $ strace -c -e getdents ruby --disable=gems -e 'Dir.glob("/.{flac,ogg}")'
> => 1870 getdents calls
>
> $ strace -c -e getdents ruby --disable=gems -e 'Dir.glob("/.{flac,ogg,mp3}")'
> => 2805 getdents calls

ksh93, zsh, bash all exhibit the same behavior, even.
And it appears a major refactoring of dir.c is necessary to
support optimizing away redundant readdir (getdents on Linux) calls.

#6 - 07/24/2017 10:40 PM - ahorek (Pavel Rosický)

There isn't noticable difference on Linux, it's even slightly faster.

Linux

```
2.1.9 77991 i/s
2.4.1 78497 i/s
```

Windows

```
2.1.9 1143000 i/s
2.4.1   39829 i/s
```

https://github.com/ruby/ruby/blob/trunk/dir.c

#7 - 07/25/2017 02:51 AM - normalperson (Eric Wong)

pdahorek@seznam.cz wrote:

> There isn't noticable difference on Linux, it's even slightly faster.

The problem isn't the noticeability in Linux. I suspect the
problem here is Linux hides performance problems with fast syscalls:

> Linux
>
> ```
> 2.1.9 77991 i/s
> 2.4.1 78497 i/s
> ```
>
> Windows
>
> ```
> 2.1.9 1143000 i/s
> 2.4.1   39829 i/s
> ```

Are those numbers on the same hardware? If so, it's because
our glob performance on Linux always sucked :)

So, I suspect the performance on 2.1.9 was good because Ruby
used Win32-specific APIs; but when the code path changed to
use work the same on both systems, it got silly slow.

I've been having a tough time figuring out what changes in the
2.1..2.2 era did what over time, especially on a platform
I don't run...

Can you run "git bisect" to narrow down the performance problem
to a particular commit?

Thanks.

#8 - 07/25/2017 11:22 AM - ahorek (Pavel Rosický)

yes, it's on the same hardware and also with the same file path. I used Bash on Windows which could be slower then the native Windows app.
So I also compared it on a native Ubuntu and 2.4.1 is faster on it

```
2.1.9 695000 i/s
2.4.1 766827 i/s
```

after some digging I found out that this change introduced the problem
https://bugs.ruby-lang.org/issues/5994

around this commit
https://github.com/ruby/ruby/commit/5b92c0bea3dc23b0c2be356bedafdd4e7f9110d7

#9 - 07/25/2017 11:37 AM - ahorek (Pavel Rosický)

https://github.com/ruby/ruby/pull/1669

```
2.1.9        1143000 i/s
2.4.1          39829 i/s
2.5.0          40730 i/s
2.5.0 + patch 936338 i/s
```

this patch is probably wrong, but it's a good place to start

@normalperson (Eric Wong) - could you take a look?

#10 - 07/25/2017 04:33 PM - normalperson (Eric Wong)

pdahorek@seznam.cz wrote:

> Issue #13167 has been updated by ahorek (Pavel Rosický).
>
> https://github.com/ruby/ruby/pull/1669
>
> ```
> 2.1.9        1143000 i/s
> 2.4.1          39829 i/s
> 2.5.0          40730 i/s
> 2.5.0 + patch 936338 i/s
> ```
>
> Thanks.
>
> this patch is probably wrong, but it's a good place to start
>
> @normalperson (Eric Wong) - could you take a look?

This is @nobu's job, since he made the original change
and knows far more about case-insensitive FSes than I do.

I think the performance on Linux is a separate problem.
The 766827 i/s you got on Ubuntu is still worse than Win32;
so I think that could be improved, possibly on all platforms.

#11 - 07/25/2017 07:15 PM - ahorek (Pavel Rosický)

Sure, faster glob could make a big difference in overall performance. It's a very good candidate for optimalization.

for Windows and maybe other case-insensitive FS that shares the same codepath we should
avoid (or cache) recurring tree-stats for each magic {.txt} which are very expensive (explained here https://bugs.ruby-lang.org/issues/13167#note-3)

#12 - 07/31/2017 02:22 PM - ahorek (Pavel Rosický)

there's a good article about this
https://research.swtch.com/glob
https://perl5.git.perl.org/perl.git/commitdiff/33252c318625f3c6c89b816ee88481940e3e6f95?hp=57ab6c610267dba697199c8256f4258af7d391c1

take a look at the python's implementation
https://github.com/python/cpython/commits/3.6/Lib/glob.py

Ruby has tons of ifs, gotos and recursions for many special cases, it's not very readable and I have a tough time to understand what's happening
For instance this Windows problem is solved, Python has different approach, because results of the glob will be the same even with the previous

Ruby 2.1 implementation, you just need to normalize the output according to realpaths (I expect that I can't create two files or directories with a same name like "test.txt" and "Test.txt", am I right?)

simplier example

```
Dir.glob("c:/test/myapp")
```

Python
CreateFile
QueryInformationVolume
QueryAllInformationFile
CloseFile

Ruby 2.1
CreateFile
QueryNetworkOpenInformationFile
CloseFile

Ruby 2.4.1
CreateFile
QueryNetworkOpenInformationFile
CloseFile
CreateFile
QueryNetworkOpenInformationFile
CloseFile
CreateFile
QueryNetworkOpenInformationFile
CloseFile
CreateFile
QueryNetworkOpenInformationFile
CloseFile
QueryFirectory
CloseFile
CreateFile
QueryNetworkOpenInformationFile
CloseFile
CreateFile
QueryDirectory
CloseFile

I think that Python has fully compatible syntax, even with {} expansion and also works fast on Windows (case sensitive)

#13 - 08/27/2017 04:22 PM - ahorek (Pavel Rosický)

https://github.com/ruby/ruby/pull/1685

I reverted nobu's change and instead of recursion for simple patterns I want to call "replace_real_basename" only for results. There's no need to call it for each directory because the result will always be same. It's not final and I'll be really glad if someone more experienced can help me with it.
Also other parts like path normalization could be called only once.

What do you think about optimizing most common use cases like
Dir.glob('/test/file.{html,erb}')
Dir.glob('/test/*')
?

Ruby 2.4.1

```
               plain:     1089.3 i/s
                   *:      324.9 i/s
              braces:       37.7 i/s
                 * 2:        8.6 i/s
                  **:        3.1 i/s
```

Trunk (2.5)

```
               plain:     1013.7 i/s
                   *:      569.6 i/s
              braces:       34.7 i/s
                 * 2:       23.3 i/s
                  **:        2.8 i/s
```

Trunk (2.5) + patch

```
               plain:    18020.3 i/s
                   *:     1432.5 i/s
```

```
              braces:      917.7 i/s
                 * 2:       25.4 i/s
                  **:        3.1 i/s
```

Ruby 2.1.9

```
               plain:    20519.1 i/s
                   *:     1905.2 i/s
              braces:     1094.0 i/s
                 * 2:       46.4 i/s
                  **:        6.7 i/s
```

btw Python's performace is even faster then Ruby 2.1.9 (20x), this is a huge difference.

#14 - 08/28/2017 12:58 AM - nobu (Nobuyoshi Nakada)

ahorek (Pavel Rosický) wrote:

> There's no need to call it for each directory because the result will always be same.

It is not same.
Path components in middle also should be replaced.

#15 - 09/11/2017 02:28 PM - h.shirosaki (Hiroshi Shirosaki)

- File 0001-dir.c-performance-fix-with-braces.patch added

- File bench_dir_glob.rb added

- File 0001-dir.c-performance-fix-with-braces-using-cache.patch added

replace_real_basename() is called for same head plain paths because braces are expanded early before ruby_glob0().

Moving braces expansion to later phase in glob_helper() is a way to reduce replace_real_basename().
The idea is same as #13873.

Another idea is caching real name of each directory and use the cache.

I attached a patch and benchmark script.

Here is my benchmark result.

+ patch: 0001-dir.c-performance-fix-with-braces.patch
+ cache: 0001-dir.c-performance-fix-with-braces-using-cache.patch

braces:
Dir["v:/test/myapp/app/views/common/_menu_stats{.en,}{.html,}{}{.erb,.builder,.raw,.ruby,.jbuilder,.coffee,}"]

recursive:
Dir["v:/test/myapp/app/views/**/_menu_stats{.en,}{.html,}{}{.erb,.builder,.raw,.ruby,.jbuilder,.coffee,}"]

On Windows 10
ruby 2.5.0dev (2017-09-11 trunk 59831) [x64-mingw32]

```
              braces    148.111  (± 3.4%) i/s -    742.000  in   5.015963s
+ patch       braces      1.809k (± 3.8%) i/s -      9.078k in   5.027256s  => 12x faster
+ cache       braces    480.215  (± 5.4%) i/s -      2.397k in   5.005954s  =>  3x faster

           recursive     71.280  (± 4.2%) i/s -    357.000  in   5.014841s
+ patch    recursive    111.464  (± 2.7%) i/s -    561.000  in   5.037387s
+ cache    recursive     94.775  (± 3.2%) i/s -    477.000  in   5.037445s
```

On Linux(Ubuntu 16.04)
ruby 2.5.0dev (2017-09-11 trunk 59831) [x86_64-linux]

```
              braces      6.171k (± 1.0%) i/s -     31.408k in   5.090401s
+ patch       braces     11.241k (± 0.6%) i/s -     57.252k in   5.093467s

           recursive    720.448  (± 1.4%) i/s -      3.640k in   5.053382s
+ patch    recursive    730.159  (± 0.5%) i/s -      3.723k in   5.099068s
```

#16 - 09/26/2017 07:24 PM - naruse (Yui NARUSE)

- Related to Feature #13873: Optimize Dir.glob with FNM_EXTGLOB added

#17 - 02/05/2018 04:00 PM - sfcgeorge (Simon George)

Is there any progress on this, I see feature #13873 is related, but it looks like that got reverted again? https://bugs.ruby-lang.org/issues/13873

I ran into this issue with Rails; when a request doesn't specify the format Rails uses this Glob and it takes 10x longer to respond. In our real-world app that means collection partials that normally take 40ms each now take 300ms, thus the whole page takes 5 or more seconds! There's an issue I opened with Rails but it seems this is the root cause https://github.com/rails/rails/issues/30502

#18 - 02/06/2018 02:08 AM - h.shirosaki (Hiroshi Shirosaki)

- File deleted (0001-dir.c-performance-fix-with-braces.patch)

#19 - 02/06/2018 02:22 AM - h.shirosaki (Hiroshi Shirosaki)

- File 0001-dir.c-performance-fix-with-braces.patch added

#13873 seems reverted in order to avoid test changes (incompabitility of the order).
My patch (0001-dir.c-performance-fix-with-braces.patch) passes test-all and test-rubyspec without test changes.
It would be more similar to trunk behavior than #13873 implementation although not 100% compatible.
I rebased a patch for latest trunk and did some format fix.

#20 - 08/05/2018 04:46 PM - ahorek (Pavel Rosický)

- File linux_braces.png added

- File linux_list.png added

- File linux_recursive.png added

- File windows_braces.png added

- File windows_list.png added

- File windows_recursive.png added

- File bench_dir_glob2.rb added

#21 - 08/05/2018 05:40 PM - ahorek (Pavel Rosický)

@h.shirosaki (Hiroshi Shirosaki), thanks for your work on this. I tested your patch 0001-dir.c-performance-fix-with-braces.patch (ruby head + braces) based on the current trunk https://github.com/ruby/ruby/pull/1864

environment:

```
Samsung 850 Pro 250GB
AMD 8350FX 8C
Windows 10 and Ubuntu
16GB DDR3

ruby 2.6.0dev (2018-08-05 trunk 64192) [x86_64-linux]
jruby 9.2.1.0-SNAPSHOT (2.5.0) 2018-08-02 5aa064b Java HotSpot(TM) 64-Bit Server VM 10.0.1+10 on 10.0.1+10 +ji
t [linux-x86_64]
```

ratio (faster than trunk)

```
linux braces 1.26x
linux recursive 0.99x
windows braces 10.75x
windows recursive 1.66x
```

I think the patch fixes the main problem I originaly reported. Especially "windows braces" is almost 11-times faster, almost as fast as ruby 2.1.9 was.

I also tested it with my rspec suite and it runs 2.14x faster, this is a huge perf difference. It passes all tests.

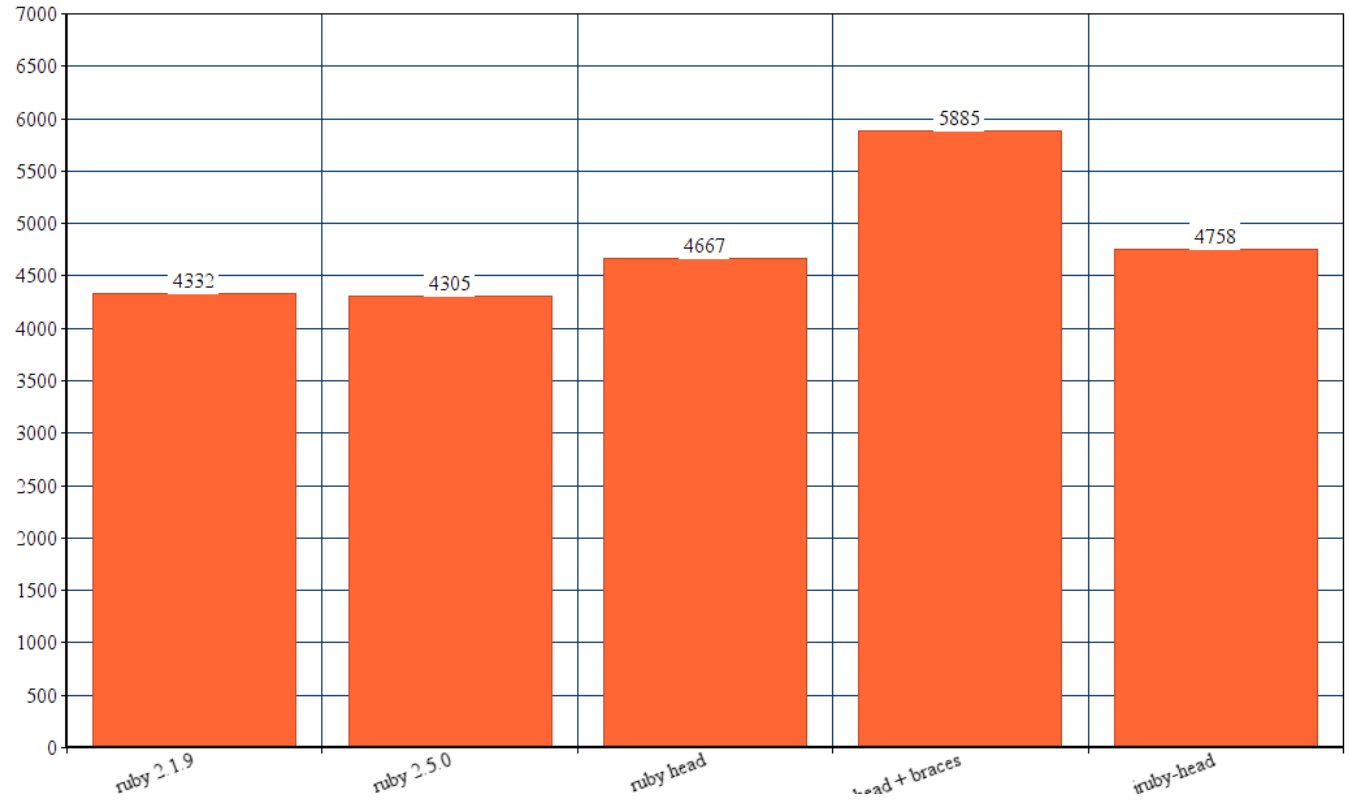
```
ruby trunk
22 minutes 46 seconds
ruby trunk + patch
10 minutes 5 seconds
```

cc @nobu (Nobuyoshi Nakada) if you have time, could you please review it?

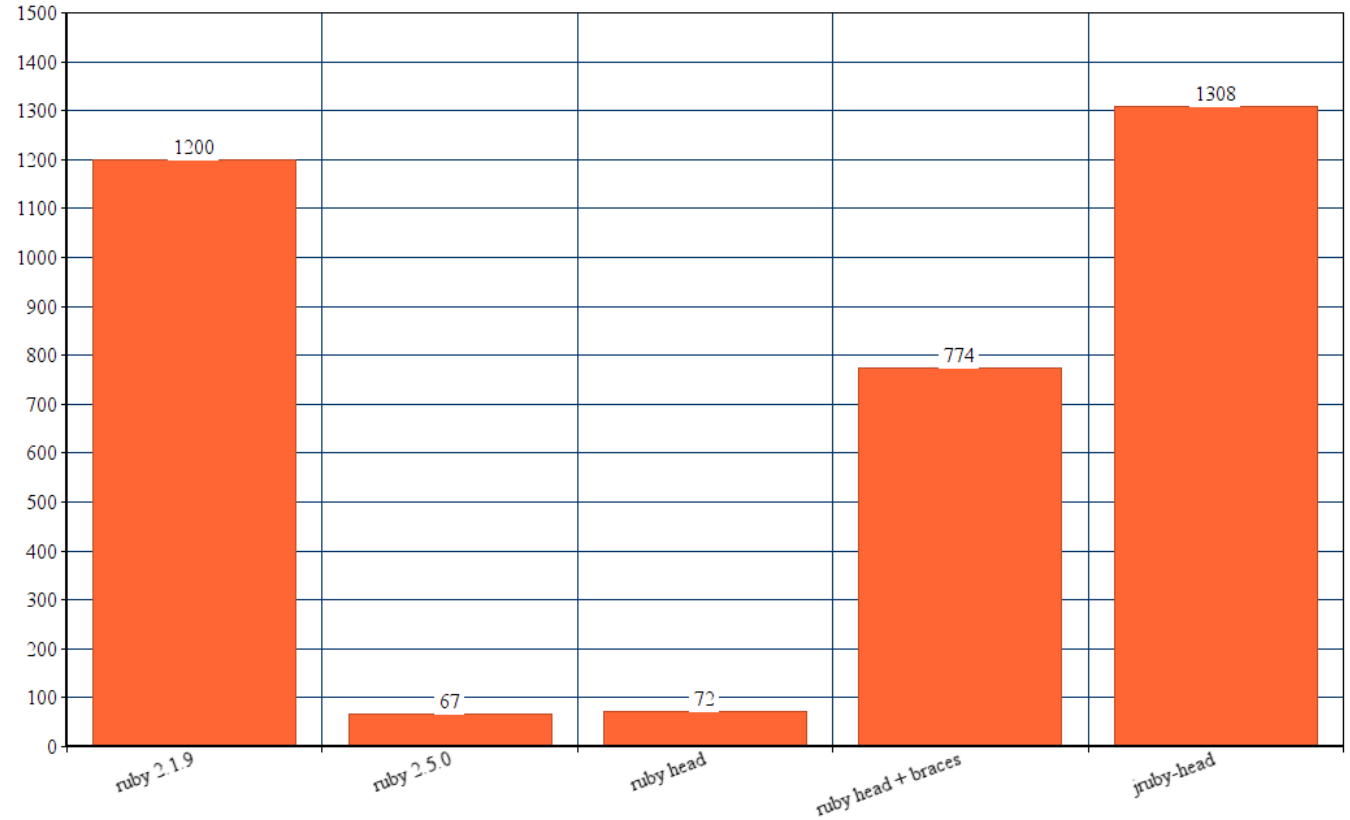
```
Linux
ruby 2.1.9
                 list     12.627k (± 1.6%) i/s -     63.232k in   5.008885s
              braces      4.332k (± 1.9%) i/s -     21.889k in   5.054435s
           recursive     81.603  (± 1.2%) i/s -    413.000  in   5.062313s

ruby 2.5.0
```

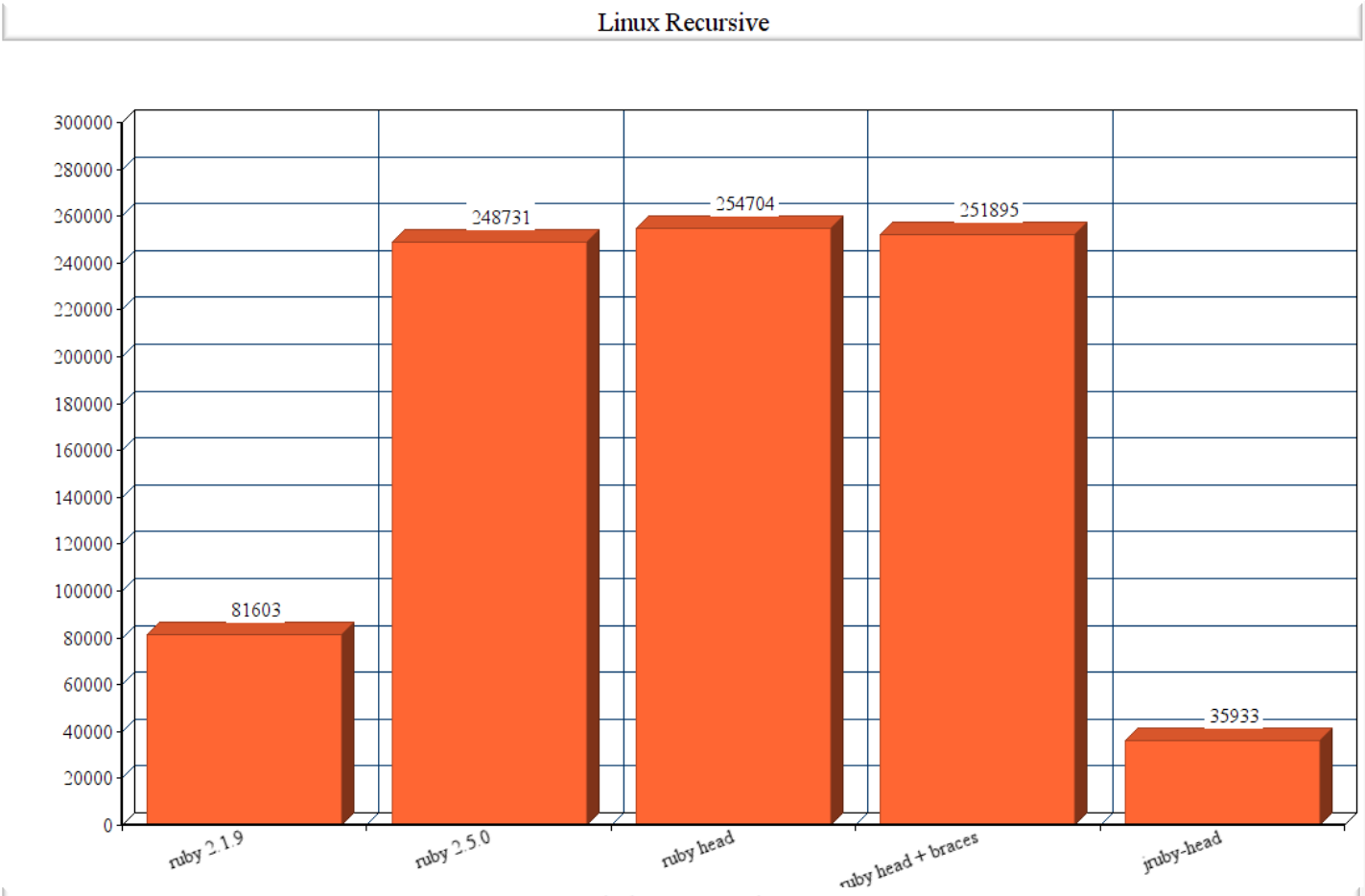
```
                list     11.752k (± 1.3%) i/s -     59.176k in   5.036229s
             braces      4.305k (± 2.0%) i/s -     21.600k in   5.019530s
          recursive    248.731  (± 1.6%) i/s -      1.248k in   5.018503s

ruby head
                list     12.128k (± 2.4%) i/s -     60.840k in   5.019484s
             braces      4.667k (± 3.1%) i/s -     23.613k in   5.064703s
          recursive    254.704  (± 2.0%) i/s -      1.275k in   5.007455s

ruby head + braces
                list     12.123k (± 3.3%) i/s -     61.048k in   5.041848s
             braces      5.885k (± 2.2%) i/s -     29.784k in   5.063815s
          recursive    251.895  (± 2.0%) i/s -      1.275k in   5.063459s

jruby-head
                list      9.931k (± 2.4%) i/s -     49.764k in   5.014070s
             braces      4.758k (± 1.7%) i/s -     23.940k in   5.032956s
          recursive     35.933  (± 5.6%) i/s -    180.000  in   5.022796s



Windows
ruby 2.1.9
                list      2.683k (± 5.9%) i/s -     13.566k in   5.077196s
             braces      1.200k (± 3.2%) i/s -      6.000k in   5.005971s
          recursive    111.844  (± 0.9%) i/s -    561.000  in   5.016557s

ruby 2.5.0
                list    945.309  (± 3.0%) i/s -      4.794k in   5.076069s
             braces     67.879  (± 2.9%) i/s -    342.000  in   5.041694s
          recursive     33.314  (± 3.0%) i/s -    168.000  in   5.046526s

ruby head
                list      1.001k (± 1.8%) i/s -      5.049k in   5.047494s
             braces     72.145  (± 1.4%) i/s -    364.000  in   5.046341s
          recursive     34.943  (± 2.9%) i/s -    177.000  in   5.068275s

ruby head + braces
                list      1.001k (± 1.3%) i/s -      5.049k in   5.044865s
             braces    773.822  (± 0.9%) i/s -      3.927k in   5.075205s
          recursive     58.596  (± 1.7%) i/s -    295.000  in   5.034900s

jruby-head
                list      5.121k (? 1.3%) i/s -     25.935k in   5.064926s
             braces      1.308k (? 2.1%) i/s -      6.625k in   5.066130s
          recursive      9.987  (? 0.0%) i/s -     50.000  in   5.008338s
```

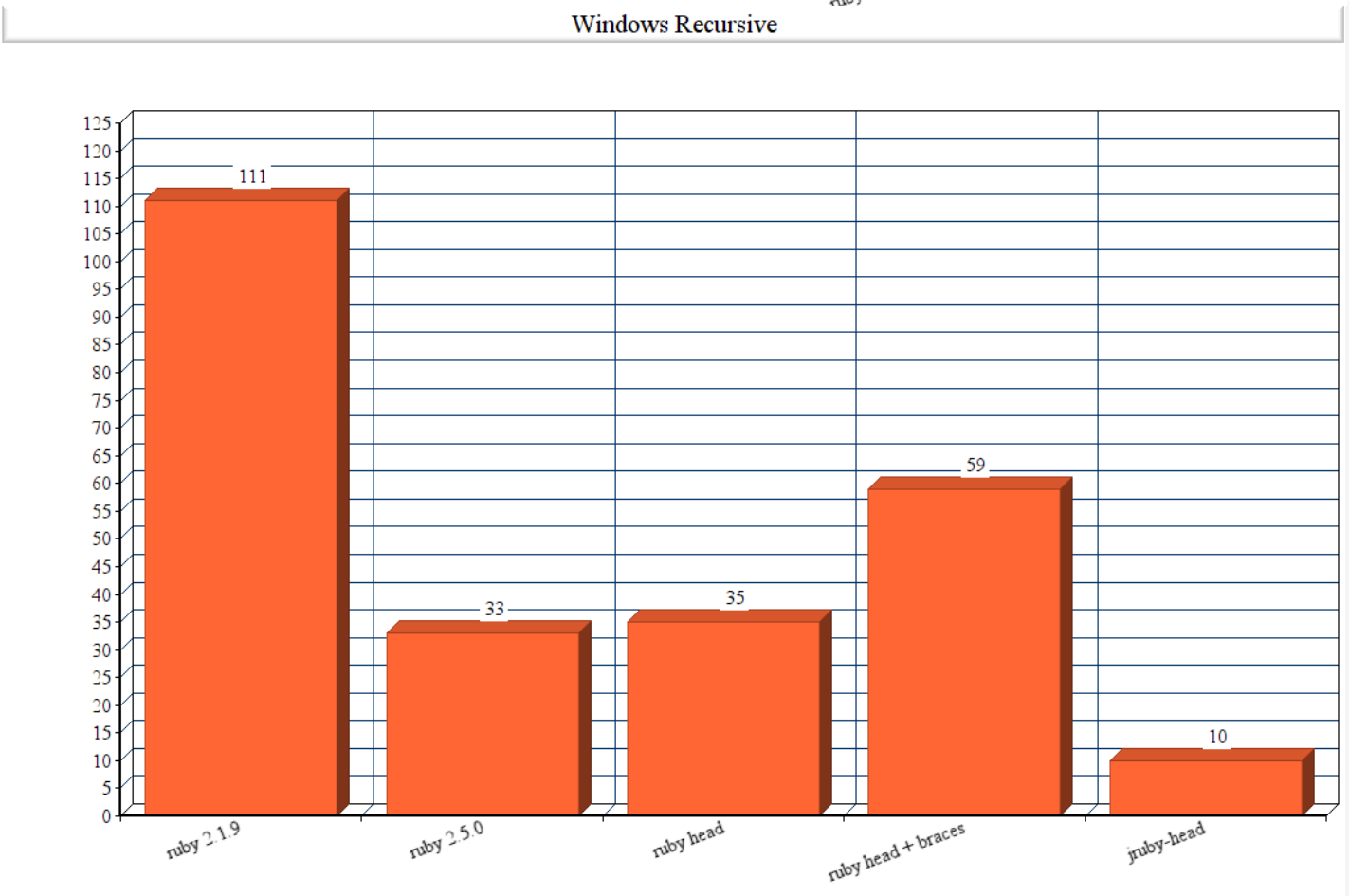
Linux Braces



Windows Braces







#22 - 08/09/2018 07:55 AM - h.shirosaki (Hiroshi Shirosaki)

- Status changed from Open to Assigned

- Assignee set to nobu (Nobuyoshi Nakada)

#23 - 09/13/2018 07:53 AM - nobu (Nobuyoshi Nakada)

- Assignee changed from nobu (Nobuyoshi Nakada) to h.shirosaki (Hiroshi Shirosaki)

Thank you for the patch, let's try, please commit the patch for braces.

#24 - 09/13/2018 07:53 AM - naruse (Yui NARUSE)

> 0001-dir.c-performance-fix-with-braces.patch
> It would be more similar to trunk behavior than #13873 implementation although not 100% compatible.

I'm wondering whether this incompatibility is critical or not.
Anyway the easiest way is just merge it and wait the feedback from Rails.

#25 - 09/22/2018 01:11 AM - Anonymous

- Status changed from Assigned to Closed

Applied in changeset trunk|r64810.

---

dir.c: performance fix with braces

Braces were expended before ruby_glob0(). This caused to call
replace_real_basename() for same plain patterns repeatedly.
Move blace expansion into glob_helper() in ruby_glob0() to reduce
replace_real_basename() call.
This fix changes the order of glob results.
[Feature #13167] [Fix GH-1864]

#26 - 09/23/2018 02:35 AM - k0kubun (Takashi Kokubun)

After this commit is merged, some CIs that has -DVM_CHECK_MODE=2 and continue to test latest revision started to randomly crash "TestGem#test_load_plugins":
http://ci.rvm.jp/results/trunk-asserts@silicon-docker
http://ci.rvm.jp/results/trunk-vm-asserts@silicon-docker

Their logs will be lost after 3 days, so I attach persisted failed logs too:
https://gist.github.com/ko1/2c905ef9194b727001bea1fa5cb22f70
https://gist.github.com/ko1/f4f9afb4ea2e48600467ca0a75decd58
https://gist.github.com/ko1/ba7cc479072764cb46482f112811d4b6
... and more

There may be a possibility that rubygems will become unstable by this (but currently it's reproductive only when -DVM_CHECK_MODE=2 is used), and I'm writing here since CI notifies the failure too often.

#27 - 09/26/2018 04:09 AM - h.shirosaki (Hiroshi Shirosaki)

k0kubun (Takashi Kokubun) wrote:

> After this commit is merged, some CIs that has -DVM_CHECK_MODE=2 and continue to test latest revision started to randomly crash "TestGem#test_load_plugins":
> http://ci.rvm.jp/results/trunk-asserts@silicon-docker
> http://ci.rvm.jp/results/trunk-vm-asserts@silicon-docker
>
> Their logs will be lost after 3 days, so I attach persisted failed logs too:
> https://gist.github.com/ko1/2c905ef9194b727001bea1fa5cb22f70
> https://gist.github.com/ko1/f4f9afb4ea2e48600467ca0a75decd58
> https://gist.github.com/ko1/ba7cc479072764cb46482f112811d4b6
> ... and more

Is that fixed by r64849? Thanks for the patch.

#28 - 09/26/2018 06:10 AM - k0kubun (Takashi Kokubun)

I hope so too (not confident enough since it was a random failure). Thank you for your attention to these CIs.

#29 - 10/10/2022 02:49 PM - nobu (Nobuyoshi Nakada)

- Related to Bug #19042: Bug: Dir.glob ignores subdirectories in alternation when alternation is preceded by recursive directory pattern added

Files

logruby24.txt	484 KB	07/24/2017	ahorek (Pavel Rosický)
logruby21.txt	10.8 KB	07/24/2017	ahorek (Pavel Rosický)
bench_dir_glob.rb	880 Bytes	09/11/2017	h.shirosaki (Hiroshi Shirosaki)
0001-dir.c-performance-fix-with-braces-using-cache.patch	5.84 KB	09/11/2017	h.shirosaki (Hiroshi Shirosaki)
0001-dir.c-performance-fix-with-braces.patch	8.64 KB	02/06/2018	h.shirosaki (Hiroshi Shirosaki)
linux_braces.png	24.1 KB	08/05/2018	ahorek (Pavel Rosický)
linux_list.png	23.5 KB	08/05/2018	ahorek (Pavel Rosický)
linux_recursive.png	26.6 KB	08/05/2018	ahorek (Pavel Rosický)
windows_braces.png	23.6 KB	08/05/2018	ahorek (Pavel Rosický)
windows_list.png	23.1 KB	08/05/2018	ahorek (Pavel Rosický)
windows_recursive.png	28 KB	08/05/2018	ahorek (Pavel Rosický)
bench_dir_glob2.rb	982 Bytes	08/05/2018	ahorek (Pavel Rosický)