# Ruby - Bug #17774

## Quantified empty group causes regex to fail

04/04/2021 08:08 AM - Davidebyzero (David Ellsworth)

| | | | |
|---|---|---|---|
| **Status:** | Open | | |
| **Priority:** | Normal | | |
| **Assignee:** | | | |
| **Target version:** | | | |
| **ruby -v:** | ruby 2.7.2p137 (2020-10-01 revision 5445e04352) [x86_64-msys] | **Backport:** | 2.5: UNKNOWN, 2.6: UNKNOWN, 2.7: UNKNOWN, 3.0: UNKNOWN |

**Description**

The regex ^((x*)(?=\2$))*x$ matches powers of 2 in unary, expressed as strings of x characters whose length is the number.

Adding an empty group () in the middle of it should have no effect on its operation, and indeed it does not. ^((x*)()(?=\2$))*x$ still matches powers of 2 just fine.
Quantifying that empty group, (){4}, should still have no effect. And indeed, ^((x*)(){4}(?=\2$))*x$ still matches powers of 2. But quantify that to (){5}, and suddenly it fails.

The following command line should print 1, but instead prints nothing:

```
ruby -e 'print 1 if "x"*32 =~ /^((x*)(){5}(?=\2$))*x$/'
```

However this one does print 1:

```
ruby -e 'print 1 if "x"*32 =~ /^((x*)(){4}(?=\2$))*x$/'
```

Bug found to occur on Try It Online: ruby 2.5.5p157 (2019-03-15 revision 67260) [x86_64-linux]
Bug confirmed to happen on my own machine: ruby 2.7.2p137 (2020-10-01 revision 5445e04352) [x86_64-msys]

Solving the challenge Is that number a Two Bit Number™? on Code Golf Stack Exchange is what led me to discover this bug.

### History

#### #1 - 04/05/2021 06:16 AM - mame (Yusuke Endoh)

Thank you, I can reproduce the issue.

The issue is in the code from onigmo, so it would be helpful if you could report this issue to the upstream.

By a quick investigation, an optimization expands (){4}, and does not expand (){5}, which makes the difference of the behavior.
Enabling debug output suggests that the bug is caused by USE_MONOMANIAC_CHECK_CAPTURES_IN_ENDLESS_REPEAT option. The (){5} case works great by the following change that disables the option, but I'm unsure the performance impact.

```
diff --git a/regint.h b/regint.h
index 0740429688..968ea6cde8 100644
--- a/regint.h
+++ b/regint.h
@@ -71,7 +71,6 @@
 #define USE_PERL_SUBEXP_CALL
 #define USE_CAPITAL_P_NAMED_GROUP
 #define USE_BACKREF_WITH_LEVEL        /* \k<name+n>, \k<name-n> */
-#define USE_MONOMANIAC_CHECK_CAPTURES_IN_ENDLESS_REPEAT  /* /(?:()|())*\2/ */
 #define USE_NEWLINE_AT_END_OF_STRING_HAS_EMPTY_LINE     /* /\n$/ =~ "\n" */
 #define USE_WARNING_REDUNDANT_NESTED_REPEAT_OPERATOR
 /* !!! moved to regenc.h. */ /* #define USE_CRNL_AS_LINE_TERMINATOR */
```

#### #2 - 05/01/2021 11:06 AM - wanabe (_ wanabe)

The reproduction example could be a bit shorter.

```
$ ruby -ve 'p "xxxx" =~ /(?:x(){5})*$/, "xxxx" =~ /(?:x(){4})*$/'
ruby 3.1.0dev (2021-05-01T02:04:17Z origin/master 121fa24a34) [x86_64-linux]
3
0
```

This problem has already been fixed in Oniguruma, a derivative of Onigmo.

https://github.com/kkos/oniguruma/commit/ca64663ca8bb34ca7dc219d18ec6e475cca9dec8

```
$ (git checkout ca64663ca8bb34ca7dc219d18ec6e475cca9dec8~ && autoreconf -vfi && ./configure && make -j6 && sed
 -i sample/simple.c -e 's/\(pattern *= [^"]*\)"[^"]*"/\1"(?:x(){5})*$"/' -e 's/\(str *= [^"]*\)"[^"]*"/\1"xxxx
"/' && (cd sample; make simple)) > build.log 2>&1 && ./sample/simple
match at 3
0: (3-4)
1: (4-4)
```

```
$ (git checkout ca64663ca8bb34ca7dc219d18ec6e475cca9dec8 && autoreconf -vfi && ./configure && make -j6 && sed
-i sample/simple.c -e 's/\(pattern *= [^"]*\)"[^"]*"/\1"(?:x(){5})*$"/' -e 's/\(str *= [^"]*\)"[^"]*"/\1"xxxx"
/' && (cd sample; make simple)) > build.log 2>&1 && ./sample/simple
match at 0
0: (0-4)
1: (4-4)
```

I think that introducing a mechanism that exists in Oniguruma 6.x, such as empty_status_mem and set_empty_status_check_trav, may solve the problem.

#### #3 - 05/01/2021 12:50 PM - sawa (Tsuyoshi Sawada)

wanabe (_ wanabe) wrote in #note-2:

> ... Oniguruma, a derivative of Onigmo

I believe it is the other way around.

#### #4 - 05/01/2021 08:06 PM - wanabe (_ wanabe)

sawa (Tsuyoshi Sawada) wrote in #note-3:

> wanabe (_ wanabe) wrote in #note-2:
>
> > ... Oniguruma, a derivative of Onigmo
>
> I believe it is the other way around.

Oh I'm very sorry, I wrote it wrong.
I was aware of it, but I simply used the wrong word.

#### #5 - 10/13/2021 04:43 PM - jeremyevans0 (Jeremy Evans)

I looked into fixing this by removing the define of USE_MONOMANIAC_CHECK_CAPTURES_IN_ENDLESS_REPEAT, as @mame (Yusuke Endoh) indicated: https://github.com/ruby/ruby/commit/018922ba15eb7aea86957789d7defae9ffc43688

It ends up breaking a few specs.  For example, it changes the behavior of:

```
/(a|\2b|())*/.match("aaabbb").to_a

# Before:
# => ["aaabbb", "", ""]

# After:
# => ["aaa", "", ""]
```

For this example, Ruby 1.8 returns ["aaa", "a", nil].  The equivalent in Perl returns ["aaa", "", ""]. The equivalent in Python 2 and 3 returns ["aaabbb", "", ""].  I think the ["aaabbb", "", ""] result seems best for a greedy match since it matches the most characters. However, I can also see where an implementation would return one of the other results if a scan terminates when no forward progress is made during an iteration.

Anyway, if we are OK with this behavior change for empty capture groups, I can submit the commit as a pull request.  However, I think it would be better to wait for a fix in Onigmo.