

Ruby trunk - Feature #14330

Speedup `block.call` where `block` is passed block parameter.

01/07/2018 04:58 PM - ko1 (Koichi Sasada)

Status:	Closed
Priority:	Normal
Assignee:	ko1 (Koichi Sasada)
Target version:	2.6

Description

abstract

Speedup block.call where block is passed block parameter with a special object named "proxyblock" which responds to call and invoke passed block.

background

Ruby 2.5 improved performance of passing a passed block parameter by lazy Proc creation ([Feature #14045]). However, block.call (block is a passed block parameter) is not optimized and need to create a Proc object immediately.

proposal

We need to make Proc creation lazily for performance. There are several way to achieve it, but I propose to use special object named "blockproxy" object.

This is a pseudo code to use it:

```
# block is given block parameter

block.call(1, 2, 3)

#=> translate at compile time

if block is not modified and
  block is ISeq/IFunc block
  tmp = blockproxy
else
  tmp = block # create Proc and so on
end
tmp.call(1, 2, 3)
```

blockproxy.call invoke given block if Proc#call is not redefined, otherwise make a Proc and call Proc#call as usual.

Advantage of this method is we can also use this technique with the safe navigation operator (block&.call). If block is not given, then tmp will be nil, and no method dispatched with &..

Note that this technique depends on the assumption "we can't access to the evaluated receiver just before method dispatch". We don't/can't access blockproxy object at method dispatch, and no compatibility issue.

evaluation

Using https://github.com/k0kubun/benchmark_driver

```
prelude: |
  def block_yield
    yield
  end
  def bp_yield &b
    yield
  end
  def bp_call &b
```

```

    b.call
  end
  def bp_safe_call &b
    b&.call
  end
benchmark:
- block_yield{}
- bp_yield{}
- bp_call{}
- bp_safe_call{}
- bp_safe_call

```

Result:

```

Warming up -----
  block_yield{}    1.298M i/100ms
  bp_yield{}      1.177M i/100ms
  bp_call{}       447.723k i/100ms
  bp_safe_call{}  413.261k i/100ms
  bp_safe_call    2.955M i/100ms
Calculating -----
           trunk      modified
  block_yield{}  20.672M    20.808M i/s -   51.933M in 2.512265s 2.495806s
  bp_yield{}    16.294M    16.220M i/s -   47.099M in 2.890459s 2.903797s
  bp_call{}     5.626M    14.752M i/s -   17.909M in 3.182966s 1.213976s # x2.62
  bp_safe_call{} 5.555M    14.557M i/s -   16.530M in 2.975892s 1.135586s # x2.62
  bp_safe_call  31.339M    23.561M i/s -  118.184M in 3.771157s 5.016200s

```

The patch is here:

<https://gist.github.com/ko1/d8a1a9d92075b27a8e95ca528cc57fd2>

Related issues:

Related to Ruby trunk - Bug #14335: block.call should respect redefinition of...

Closed

Associated revisions

Revision 7fd11834 - 01/07/2018 07:18 PM - ko1 (Koichi Sasada)

Speedup block.call [Feature #14330]

- insns.def (getblockparamproxy): introduce new instruction to return the rb_block_param_proxy object if possible. This object responds to call method and invoke given block (completely similar to yield).
- method.h (OPTIMIZED_METHOD_TYPE_BLOCK_CALL): add new optimized call type which is for rb_block_param_proxy.cal.
- vm_inshelper.c (vm_call_method_each_type): ditto.
- vm_inshelper.c (vm_call_opt_block_call): ditto.
- vm_core.h (BOP_CALL, PROC_REDEFINED_OP_FLAG): add check for Proc#call redefinition.
- compile.c (iseq_compile_each0): compile to use new insn getblockparamproxy for method call.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@61659 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 61659 - 01/07/2018 07:18 PM - ko1 (Koichi Sasada)

Speedup block.call [Feature #14330]

- insns.def (getblockparamproxy): introduce new instruction to return the rb_block_param_proxy object if possible. This object responds to call method and invoke given block (completely similar to yield).
- method.h (OPTIMIZED_METHOD_TYPE_BLOCK_CALL): add new optimized call type which is for rb_block_param_proxy.cal.

- vm_inshelper.c (vm_call_method_each_type): ditto.
- vm_inshelper.c (vm_call_opt_block_call): ditto.
- vm_core.h (BOP_CALL, PROC_REDEFINED_OP_FLAG): add check for Proc#call redefinition.
- compile.c (iseq_compile_each0): compile to use new insn getblockparamproxy for method call.

Revision 61659 - 01/07/2018 07:18 PM - ko1 (Koichi Sasada)

Speedup block.call [Feature #14330]

- insns.def (getblockparamproxy): introduce new instruction to return the rb_block_param_proxy object if possible. This object responds to call method and invoke given block (completely similar to yield).
- method.h (OPTIMIZED_METHOD_TYPE_BLOCK_CALL): add new optimized call type which is for rb_block_param_proxy.cal.
- vm_inshelper.c (vm_call_method_each_type): ditto.
- vm_inshelper.c (vm_call_opt_block_call): ditto.
- vm_core.h (BOP_CALL, PROC_REDEFINED_OP_FLAG): add check for Proc#call redefinition.
- compile.c (iseq_compile_each0): compile to use new insn getblockparamproxy for method call.

Revision 633b4638 - 02/21/2018 08:14 AM - ko1 (Koichi Sasada)

add NEWS entries about [Feature #14318] and [Feature #14330].

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@62514 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 62514 - 02/21/2018 08:14 AM - ko1 (Koichi Sasada)

add NEWS entries about [Feature #14318] and [Feature #14330].

Revision 62514 - 02/21/2018 08:14 AM - ko1 (Koichi Sasada)

add NEWS entries about [Feature #14318] and [Feature #14330].

History

#1 - 01/07/2018 07:18 PM - ko1 (Koichi Sasada)

- Status changed from Open to Closed

Applied in changeset [trunk|r61659](#).

Speedup block.call [Feature [#14330](#)]

- insns.def (getblockparamproxy): introduce new instruction to return the rb_block_param_proxy object if possible. This object responds to call method and invoke given block (completely similar to yield).
- method.h (OPTIMIZED_METHOD_TYPE_BLOCK_CALL): add new optimized call type which is for rb_block_param_proxy.cal.
- vm_inshelper.c (vm_call_method_each_type): ditto.
- vm_inshelper.c (vm_call_opt_block_call): ditto.
- vm_core.h (BOP_CALL, PROC_REDEFINED_OP_FLAG): add check for Proc#call redefinition.
- compile.c (iseq_compile_each0): compile to use new insn

getblockparamproxy for method call.

#2 - 01/08/2018 09:00 AM - nobu (Nobuyoshi Nakada)

- Related to Bug #14335: *block.call* should respect redefinition of *Proc#call* added