Ruby master - Feature #11786

[PATCH] micro-optimize case dispatch even harder

12/08/2015 03:36 AM - normalperson (Eric Wong)

Status:	Open
Priority:	Normal
Assignee:	
Target version:	

Description

I noticed these optimizations while working on r52931.

By using a bare hash table, we avoid the overhead of rb_hash_*
functions as well as the cost of translating FIX2INT for jump
labels.

This also reduces GC overhead, as the iseq mark array no longer
carries redundant objects nor the Hash object.

Summary speedup:

loop_whileloop2 1.000
vm2_case* 1.225
vm2_case_lit* 1.000
vm2_case_small* 1.162

Passes all tests and specs, so probably safe to commit soon...

Full results:

2015-12-08 03:26:19 +0000
target 0: a (ruby 2.3.0dev (2015-12-08 trunk 52932) [x86_64-linux]) at "/home/ew/rrrr/b/ruby"
target 1: b (ruby 2.3.0dev (2015-12-08 master 52932) [x86_64-linux]
last_commit=micro-optimize case dispatch even harder) at "/home/ew/ruby/b/ruby"

---

loop_whileloop2

```
i = 0
while i< 6_000_000 # benchmark loop 2
i += 1
end
```

a 0.10256672604009509
a 0.10279557690955698
a 0.10286601004190743
a 0.10278227413073182
a 0.10278794192709029
b 0.10288732498884201
b 0.10327051300555468
b 0.1026718900538981
b 0.10263488302007318
b 0.10256404406391084

---

vm2_case

```
i = 0
while i<6_000_000 # while loop 2
case :foo
when :bar
raise
when :baz
```

```
raise
when :boo
raise
when :foo
i += 1
end
end

a  0.18099936190992594
a  0.1836838680319488
a  0.1825075231026858
a  0.18087006197310984
a  0.18375545786693692
b  0.16652655508369207
b  0.16673082998022437
b  0.16754083498381078
b  0.16704767406918108
b  0.16648077592253685
```

```
vm2_case_lit

i = 0
@ret = [ "foo", true, false, :sym, 6, nil, 0.1, 0xffffffffffffffff ]
def foo(i)
@ret[i % @ret.size]
end

while i<6_000_000 # while loop 2
case foo(i)
when "foo" then :foo
when true then true
when false then false
when :sym then :sym
when 6 then :fix
when nil then nil
when 0.1 then :float
when 0xffffffffffffffff then :big
end
i += 1
end

a  0.670805990928784
a  0.6559841448906809
a  0.6560367799829692
a  0.655022048857063
a  0.671947613125667
b  0.659776204964146
b  0.6549702121410519
b  0.6788892599288374
b  0.6793588208965957
b  0.6878500080201775
```

```
vm2_case_small

i = 0
while i<6_000_000 # while loop 2
case :foo
when :foo
i += 1
else
raise
end
end

a  0.1667630800511688
```

a 0.16669297800399363
a 0.1665362489875406
a 0.16644068900495768
a 0.16646360605955124
b 0.15790433110669255
b 0.15774213010445237
b 0.15753646893426776
b 0.15781028987839818
b 0.15771835204213858

Elapsed time: 11.069307098 (sec)

benchmark results:
minimum results in each 5 measurements.
Execution time (sec)
name a b
loop_whileloop2 0.103 0.103
vm2_case* 0.078 0.064
vm2_case_lit* 0.552 0.552
vm2_case_small* 0.064 0.055

History

#1 - 12/08/2015 03:48 AM - normalperson (Eric Wong)

Oops, I forgot to free the table when iseq is destroyed :x
Update coming...

#2 - 12/08/2015 03:58 AM - ko1 (Koichi Sasada)

On 2015/12/08 12:43, Eric Wong wrote:

> Oops, I forgot to free the table when iseq is destroyed :x
> Update coming...

how to mark literal objects?

BTW, I'm writing new iseq dumper and loader.
it is some tough timing such a format change Xp

--
// SASADA Koichi at atdot dot net

#3 - 12/08/2015 04:58 AM - normalperson (Eric Wong)

SASADA Koichi ko1@atdot.net wrote:

> On 2015/12/08 12:43, Eric Wong wrote:
>
> > Oops, I forgot to free the table when iseq is destroyed :x
> > Update coming...
>
> how to mark literal objects?

Take ideas from nobu (r52708 :)

http://80x24.org/spew/20151208044815.1843-1-e%4080x24.org/raw

#4 - 12/08/2015 05:18 AM - normalperson (Eric Wong)

Full v2 patch including leak fix with wrapper object:

http://80x24.org/spew/20151208050811.6803-1-e%4080x24.org/t/raw

Slightly smaller improvements with leak fix, but it could be noise
for the small cases.

2015-12-08 05:06:17 +0000
target 0: a (ruby 2.3.0dev (2015-12-08 trunk 52938) [x86_64-linux]) at "/home/ew/rrrr/b/ruby"

target 1: b (ruby 2.3.0dev (2015-12-08 master 52938) [x86_64-linux]
last_commit=compile.c: wrap literal object) at "/home/ew/ruby/b/ruby"

benchmark results:

minimum results in each 5 measurements.
Execution time (sec)
name a b
loop_whileloop2 0.103 0.103
vm2_case* 0.075 0.064
vm2_case_lit* 0.560 0.561
vm2_case_small* 0.061 0.055

Speedup ratio: compare with the result of `a' (greater is better)
name b
loop_whileloop2 1.000
vm2_case* 1.169
vm2_case_lit* 1.000
vm2_case_small* 1.119

#5 - 12/08/2015 05:48 AM - ko1 (Koichi Sasada)

On 2015/12/08 13:53, Eric Wong wrote:

> how to mark literal objects?
> Take ideas from nobu (r52708 :)

I meant marking, not for freeing a table.

For example, bignum objects should mark from this table.

```
case nil
when [bignum literal here]
end
```

(Sorry if I overlooked marking)

I'm not sure it is worth or not. IMO keeping simple with Hash is better on this case with your measured improvements.

--
// SASADA Koichi at atdot dot net

#6 - 12/08/2015 08:28 AM - normalperson (Eric Wong)

SASADA Koichi ko1@atdot.net wrote:

> On 2015/12/08 13:53, Eric Wong wrote:
>
> > how to mark literal objects?
> > Take ideas from nobu (r52708 :)
>
> I meant marking, not for freeing a table.

Oops :x I added rb_mark_set to the Data_Wrap_Struct calls

http://80x24.org/spew/20151208080840.24265-1-e@80x24.org/raw

benchmark results:
minimum results in each 10 measurements.
Execution time (sec)
name a b
loop_whileloop2 0.104 0.104
vm2_case* 0.072 0.062
vm2_case_lit* 0.540 0.539
vm2_case_small* 0.060 0.053

Speedup ratio: compare with the result of `a' (greater is better)

name b
loop_whileloop2 0.994
vm2_case* 1.153
vm2_case_lit* 1.002
vm2_case_small* 1.136

> I'm not sure it is worth or not. IMO keeping simple with Hash is better
> on this case with your measured improvements.

Yeah, it's pretty minor. I'm not even sure if opt_case_dispatch
makes any difference at all for most real code. st_table still
has a lot of space overhead and id_table is not usable here.

#7 - 12/09/2015 09:28 AM - normalperson (Eric Wong)

SASADA Koichi ko1@atdot.net wrote:

> On 2015/12/08 13:53, Eric Wong wrote:
>
> > > how to mark literal objects?
> > > Take ideas from nobu (r52708 :)
>
> I meant marking, not for freeing a table.
>
> For example, bignum objects should mark from this table.
>
> ```
> case nil
> when [bignum literal here]
> end
> ```
>
> (Sorry if I overlooked marking)

Wait, putobject calls (and TS_VALUE operands) do not go away when
opt_case_dispatch insn is emitted, so marking keys in literal hash is
redundant. since iseq_set_sequence adds marks for all TS_VALUE.
(but maybe I'm misreading compile.c, too)

> I'm not sure it is worth or not. IMO keeping simple with Hash is better
> on this case with your measured improvements.

Yes, we can revisit after 2.3 release. Maybe this is appropriate for
sorted array + bsearch since the table is frozen at compile time and
st uses too much memory.

#8 - 12/09/2015 09:58 AM - ko1 (Koichi Sasada)

On 2015/12/09 18:25, Eric Wong wrote:

> > (Sorry if I overlooked marking)
> > Wait, putobject calls (and TS_VALUE operands) do not go away when
> > opt_case_dispatch insn is emitted, so marking keys in literal hash is
> > redundant. since iseq_set_sequence adds marks for all TS_VALUE.
> > (but maybe I'm misreading compile.c, too)

OMG. You are correct.
(I forget that the following instructions)

> > I'm not sure it is worth or not. IMO keeping simple with Hash is better
> > on this case with your measured improvements.
> > Yes, we can revisit after 2.3 release. Maybe this is appropriate for
> > sorted array + bsearch since the table is frozen at compile time and
> > st uses too much memory.

I agree. But I'm not sure we can do bsearch (can we compare with different type values?).

--
// SASADA Koichi at atdot dot net

Files

0001-micro-optimize-case-dispatch-even-harder.patch	9.34 KB	12/08/2015	normalperson (Eric Wong)