Ruby trunk - Bug #9227

use opt_aset ?

12/08/2013 11:01 AM - tmm1 (Aman Gupta)

Status:	Closed		
Priority:	Normal		
Assignee:	ko1 (Koichi Sasada)		
Target version:	2.1.0		
ruby -v:	trunk	**Backport:**	1.9.3: UNKNOWN, 2.0.0: UNKNOWN

Description

I noticed we have an opt_aset instruction, but nothing is using it. Is there some reason?

```
diff --git a/compile.c b/compile.c
index 812f692..9d9f14f 100644
--- a/compile.c
+++ b/compile.c
@@ -1955,6 +1955,11 @@ iseq_specialized_instruction(rb_iseq_t *iseq, INSN *iobj)
case idAREF:   SP_INSN(aref);   return COMPILE_OK;
}
break;
+           case 2:
+             switch (ci->mid) {
+                case idASET: SP_INSN(aset);   return COMPILE_OK;
+             }
+             break;
}
}
if (ci->flag & VM_CALL_ARGS_SKIP_SETUP) {
```

Associated revisions

Revision 44136 - 12/11/2013 06:38 AM - tmm1 (Aman Gupta)

compile.c: add opt_aset instruction for faster Hash#[]= and Array#[]=

- compile.c (iseq_specialized_instruction): emit opt_aset instruction to optimize Hash#[]= and Array#[]= when called with Fixnum argument. [Bug #9227] [ruby-core:58956]

Revision 44136 - 12/11/2013 06:38 AM - tmm1 (Aman Gupta)

compile.c: add opt_aset instruction for faster Hash#[]= and Array#[]=

- compile.c (iseq_specialized_instruction): emit opt_aset instruction to optimize Hash#[]= and Array#[]= when called with Fixnum argument. [Bug #9227] [ruby-core:58956]

Revision 44136 - 12/11/2013 06:38 AM - tmm1 (Aman Gupta)

compile.c: add opt_aset instruction for faster Hash#[]= and Array#[]=

- compile.c (iseq_specialized_instruction): emit opt_aset instruction to optimize Hash#[]= and Array#[]= when called with Fixnum argument. [Bug #9227] [ruby-core:58956]

Revision 44136 - 12/11/2013 06:38 AM - tmm1 (Aman Gupta)

compile.c: add opt_aset instruction for faster Hash#[]= and Array#[]=

- compile.c (iseq_specialized_instruction): emit opt_aset instruction to optimize Hash#[]= and Array#[]= when called with Fixnum argument. [Bug #9227] [ruby-core:58956]

History

#1 - 12/08/2013 11:10 AM - tmm1 (Aman Gupta)

Maybe some minor improvement on microbenchmarks.

Before patch:

```
./miniruby -I. benchmark/bm_so_k_nucleotide.rb > /dev/null  1.54s user 0.02s system 99% cpu 1.572 total
./miniruby -I. benchmark/bm_so_k_nucleotide.rb > /dev/null  1.58s user 0.02s system 99% cpu 1.608 total
./miniruby -I. benchmark/bm_so_k_nucleotide.rb > /dev/null  1.54s user 0.02s system 99% cpu 1.565 total
./miniruby -I. benchmark/bm_so_k_nucleotide.rb > /dev/null  1.57s user 0.02s system 99% cpu 1.605 total
./miniruby -I. benchmark/bm_so_k_nucleotide.rb > /dev/null  1.54s user 0.02s system 99% cpu 1.569 total
```

After patch:

```
./miniruby -I. benchmark/bm_so_k_nucleotide.rb > /dev/null  1.51s user 0.02s system 99% cpu 1.535 total
./miniruby -I. benchmark/bm_so_k_nucleotide.rb > /dev/null  1.52s user 0.02s system 99% cpu 1.545 total
./miniruby -I. benchmark/bm_so_k_nucleotide.rb > /dev/null  1.49s user 0.02s system 99% cpu 1.515 total
./miniruby -I. benchmark/bm_so_k_nucleotide.rb > /dev/null  1.54s user 0.02s system 99% cpu 1.567 total
./miniruby -I. benchmark/bm_so_k_nucleotide.rb > /dev/null  1.50s user 0.02s system 99% cpu 1.530 total
```

#2 - 12/09/2013 07:30 PM - ko1 (Koichi Sasada)

- Assignee set to ko1 (Koichi Sasada)

I forget why I remove it.
Maybe I can't measure improvement (and possible to have trouble with this insn).

If there are no regression, I don't against to use it.

#3 - 12/11/2013 12:57 PM - tmm1 (Aman Gupta)

I did some more microbenchmarks. In the Array#[Fixnum]= case, the new instruction makes a huge difference.

Before:

```
array aset      1.430000   0.000000   1.430000 (  1.430370)
array aset      1.430000   0.000000   1.430000 (  1.429422)
array aset      1.520000   0.000000   1.520000 (  1.544912)
array aset      1.440000   0.000000   1.440000 (  1.444454)
```

After:

```
array aset      0.910000   0.000000   0.910000 (  0.908866)
array aset      0.910000   0.000000   0.910000 (  0.910356)
array aset      0.890000   0.000000   0.890000 (  0.891267)
array aset      0.920000   0.000000   0.920000 (  0.921012)
```

Benchmark:

```
require 'benchmark'
```

```
Benchmark.bmbm(20) do |b|
ary = [1,2,3,4]
b.report('array aset') do
10_000_000.times do
ary[2] = 2
ary[3] = 3
end
end
end
```

Here's a copy of the patch that applies on trunk:

```
diff --git a/compile.c b/compile.c
index af11baf..25e3652 100644
--- a/compile.c
+++ b/compile.c
@@ -1955,6 +1955,11 @@ iseq_specialized_instruction(rb_iseq_t *iseq, INSN *iobj)
case idAREF:   SP_INSN(aref);   return COMPILE_OK;
}
break;
+       case 2:
+     switch (ci->mid) {
+       case idASET:   SP_INSN(aset);   return COMPILE_OK;
+     }
+     break;
}
}
if (ci->flag & VM_CALL_ARGS_SKIP_SETUP) {
```

#4 - 12/11/2013 01:00 PM - ko1 (Koichi Sasada)

OK. Please enable it.

#5 - 12/11/2013 03:38 PM - tmm1 (Aman Gupta)

- Status changed from Open to Closed

- % Done changed from 0 to 100

This issue was solved with changeset r44136.
Aman, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

---

compile.c: add opt_aset instruction for faster Hash#[]= and Array#[]=

- compile.c (iseq_specialized_instruction): emit opt_aset instruction to optimize Hash#[]= and Array#[]= when called with Fixnum argument. [Bug #9227]