Ruby master - Feature #14710

I'd like to know from C API that "It has only one reference to Ruby object" to determine whether it is a temporary object.

04/24/2018 11:28 PM - naitoh (Jun NAITOH)

Status:	Open
Priority:	Normal
Assignee:	
Target version:	

Description

I'd like to know from C API that "It has only one reference to Ruby object" to determine whether it is a temporary object.

Because broadcasting with python numpy is faster for

```
(1 line case)
y = x + 1 + 1
```

than

```
(2 line case)
y = x + 1
y + 1
```

.

In 1 line case, since the result of x + 1 is not stored in the variable, it is determined to be a temporary object (in case of "refcnt == 1") and the in-place operation is performed as it is.
(A new calculation result storage area is not used.)

Please see the comment of numpy.
https://github.com/numpy/numpy/blob/ac76793dafcd6e5f24ed052fc40413f29ebc5306/numpy/core/src/multiarray/temp_elide.c#L276

Ruby's Numo::NArray also wants to speed up the computation speed by performing similar processing, but since I think that Ruby Object does not have a reference counter, I think that it can not be realized with the current Ruby.

Is not there a good idea?

Benchmarked code

Python numpy Benchmarked code

```
$ cat inplace.py

from benchmarker import Benchmarker
import numpy as np

## specify number of loop
with Benchmarker(5000, width=40) as bench:

    @bench(None)                ## empty loop
    def _(bm):
        x = np.ones([1000,784], dtype=np.float32)
        for i in bm:
            pass

    @bench("y = x + 1.0; y + 1.0")
    def _(bm):
        x = np.ones([1000,784], dtype=np.float32)
        for i in bm:
            y = x + 1.0
            y + 1.0
```

```
    @bench("x + 1.0")
    def _(bm):
        x = np.ones([1000,784], dtype=np.float32)
        for i in bm:
            x + 1.0

    @bench("x + 1.0 + 1.0")
    def _(bm):
        x = np.ones([1000,784], dtype=np.float32)
        for i in bm:
            x + 1.0 + 1.0

    @bench("x + 1.0 + 1.0 + 1.0")
    def _(bm):
        x = np.ones([1000,784], dtype=np.float32)
        for i in bm:
            x + 1.0 + 1.0 + 1.0

    @bench("x += 1.0")
    def _(bm):
        x = np.ones([1000,784], dtype=np.float32)
        for i in bm:
            x += 1.0
```

Ruby Numo::NArray Benchmarked code

```
$ cat inplace.rb
require 'benchmark'
require 'numo/narray'

num_iteration = 5000

Benchmark.bm 40 do |r|
  x = Numo::SFloat.ones([1000,784])
  r.report "y = x + 1.0; y + 1.0" do
    num_iteration.times do
      y = x + 1.0
      y + 1.0
    end
  end

  x = Numo::SFloat.ones([1000,784])
  r.report "x + 1.0" do
    num_iteration.times do
      x + 1.0
    end
  end

  x = Numo::SFloat.ones([1000,784])
  r.report "x + 1.0 + 1.0" do
    num_iteration.times do
      x + 1.0 + 1.0
    end
  end

  x = Numo::SFloat.ones([1000,784])
  r.report "x + 1.0 + 1.0 + 1.0" do
    num_iteration.times do
      x + 1.0 + 1.0 + 1.0
    end
  end


  x = Numo::SFloat.ones([1000,784])
  r.report "x.inplace + 1.0" do
```

```
    num_iteration.times do
      x.inplace + 1.0
    end
  end

  x = Numo::SFloat.ones([1000,784])
  r.report "(x + 1.0).inplace + 1.0" do
    num_iteration.times do
      (x + 1.0).inplace + 1.0
    end
  end

  x = Numo::SFloat.ones([1000,784])
  r.report "(x + 1.0).inplace + 1.0 + 1.0" do
    num_iteration.times do
      (x + 1.0).inplace + 1.0 + 1.0
    end
  end
end
```

Result

Python numpy Result

```
$ python inplace.py
## benchmarker:         release 4.0.1 (for python)
## python version:      2.7.5
## python compiler:     GCC 4.8.5 20150623 (Red Hat 4.8.5-16)
## python platform:     Linux-3.10.0-514.6.1.el7.x86_64-x86_64-with-centos-7.3.1611-Core
## python executable:   /usr/bin/python
## cpu model:           Intel(R) Core(TM) i7-4650U CPU @ 1.70GHz  # 2299.822 MHz
## parameters:          loop=5000, cycle=1, extra=0

##                                          real    (total    = user    + sys)
(Empty)                                   0.0022    0.0000    0.0000    0.0000
y = x + 1.0; y + 1.0                      6.5245    6.0100    5.9900    0.0200
x + 1.0                                   3.2048    2.9600    2.9500    0.0100
x + 1.0 + 1.0                             4.4024    4.0500    4.0500    0.0000
x + 1.0 + 1.0 + 1.0                       5.7188    5.2700    5.2500    0.0200
x += 1.0                                  1.2219    1.1300    1.1300    0.0000
```

Please look at the column of total.
x + = 1.0 is the calculation result in pure in-place.

Ruby Numo::NArray Result (current master 7bba089, Ruby 2.5.1)

```
$ ruby inplace.rb
                                             user     system      total        real
y = x + 1.0; y + 1.0                     6.014225   1.091965   7.106190 (  7.709003)
x + 1.0                                  3.013632   0.697821   3.711453 (  4.023327)
x + 1.0 + 1.0                            6.175142   0.958922   7.134064 (  7.769464)
x + 1.0 + 1.0 + 1.0                      9.290738   1.716084  11.006822 ( 11.982892)
x.inplace + 1.0                          1.496372   0.011782   1.508154 (  1.636363)
(x + 1.0).inplace + 1.0                  5.151556   0.687806   5.839362 (  6.340946)
(x + 1.0).inplace + 1.0 + 1.0            6.714924   0.784504   7.499428 (  8.138534)
```

In Numo::NArray, "y = x + 1; y + 1" and "y = x + 1 + 1" have the same calculation speed.
Since Numo::NArray can not determine the temporary object, in order to do the same processing it is necessary to explicitly set inplace().