Ruby master - Feature #10137

Introducing Incremental GC algorithm

08/15/2014 05:57 AM - ko1 (Koichi Sasada)

Status:	Closed
Priority:	Normal
Assignee:	ko1 (Koichi Sasada)
Target version:	2.2.0

Description

Abstract

Introduce *incremental GC* algorithm to reduce pause time of major/full GC. This ticket includes design and implementation note and a working patch.

Background and problem

Ruby 2.1 uses generational GC algorithm (named RGenGC) to improve GC throughput. Genrational GC algorithm separates existing objects into young generation objects and old generation objects. At the most of GC timing, GC only marks young generation objects (called minor GC). If there are no enough memory, then mark all of objects (called major GC or full GC). Minor GC is dramatically fast than major GC. So that total throughput of application improves (10% improvement in my RDoc benchmark, [ruby-list:49896] reported that GC intensive application is 6 times faster!).

However, major GC is needed periodically and it pauses same time as GC on Ruby 2.0 and before. This problem hits response time intensive application such as web application.

Proposal

Introduce *Incremental GC* algorithm for major GC.

Incremental GC algorithm is well-known GC algorithm to reduce GC pause time. Scatter GC (marking) process in several phases and run processes interleaving with Ruby's execution process. This is similar to current lazy sweep algorithm (in fact, lazy sweep is a half part of incremental GC algorithm).

Running ruby process with marking phase, it is possible to introduce critical bug because marked objects can points un-marked objects (on the incremental GC terminology, marked objects are called "Black" objects and un-marked objects are called "White" objects). Such un-marked objects can be left in un-marking and be swept.

```
# if `ary' is already marked, and obj is not marked
ary[0] = obj
obj = nil # no-body except `ary' refers `obj'
```

To prevent such destructive bug, we can use write barriers to detect such "marked objects" to "un-marked objects". We can care about such case.

Yes, MRI/CRuby has "WB-unprotected" objects such objects does not have write barriers because of compatibility or implementation issues. To care about such WB-unprotected objects, we need to traverse all of living WB-unprotected objects at a time in the last of incremental marking. This is new extending idea against traditional incremental GC algorithm (at least I surveyed).

Deisgn and implementation details are here:
http://www.atdot.net/fp_store/f.p61can/file.data-incremental-gc.pdf

Maybe a diagram at page 10 will help you to understand the flow of all GC process.

Code is here:
https://github.com/ko1/ruby/tree/rincgc

Compare with trunk:
https://github.com/ko1/ruby/compare/rincgc

Implementation note

WB-unprotected bitmap

As I said, we need to check all of living WB-unprotected objects at the last of incremental marking phase. To do it lightweight, introduce WB-unprotected bitmap intead of specific bit in RBasic::flags.

We can get all living WB-unprotected objects with the following pseudo code:

```
bits = mark_bits[i] & wb_unprotected_bits[i];
while (bits) {
  if (bits & 1) do something.
  bits >>= 1;
}
```

4 age promotion

Because we don't need to use WB-protected bit in RBasic::flags, we have another 1 bit in RBasic::flags. To utilize this bit, we add *age* of an object with exsiting promoted bit. Rename FL_WB_PROTECTED to FL_PROMOTED0 and FL_PROMOTED to FL_PROMOTED1.

These two bits represent object's age (0 to 3) and 3 means OLD objects.

Write barriers

We extend write barriers to detect such [marked objects -> un-marked objects] reference in incremental GC. It introduces some overhead.

Evaluation

Benchmark results on real Linux machine (*1)
http://www.atdot.net/sp/raw/yekban
*1: Intel(R) Xeon(R) CPU E5335 @ 2.00GHz, 4GB memory, 2.6.32-5-amd64 (Debian squeeze)

In most of case, there are only a few (~5%) performance down.
Incremental GC introduces some overhead. But I think they are acceptable speed-down.

Discouse benchmark (only on Virtualbox VM, so accuracy is not good)
http://www.atdot.net/sp/raw/g9uban

We can recognize reducing worst case seconds.

TODO

(1) Clean up codes

Now, we can not disable incremental GC codes and generational GC codes.

We need to add ability to enable/disable features with macros.

(2) Tuning parameters

Now the parameters are fixed in codes. mruby already have tuning parameters for incremental GC (matz said they are from Lua), "GC.interval_ratio" and "GC.step_ratio". We can import these functions (or making another interface to tell).

(3) Enter GC/ Exit GC internal events

This patch also includes function "gc_enter()" and "gc_exit()" which set and reset a "doing GC" flag.

If we introduce internal event to hook these functions, we can measure exact GC pause time (and mutators time).

Summary

This feature proposal is to introduce incremental GC algorithm with working code.
Incremental GC algorithm reduce application's pause time of major GC.

Any feedback are welcome!

Thanks,
Koichi

Associated revisions

Revision 123eeb1c - 09/08/2014 04:11 AM - ko1 (Koichi Sasada)

- gc.c: add incremental GC algorithm. [Feature #10137] Please refer this ticket for details. This change also introduces the following changes.
  - Remove RGENGC_AGE2_PROMOTION and introduce object age (0 to 3). Age can be count with FL_PROMOTE0 and FL_PROMOTE1 flags in RBasic::flags (2 bit). Age == 3 objects become old objects.
  - WB_PROTECTED flag in RBasic to WB_UNPROTECTED bitmap.
  - LONG_LIVED bitmap to represent living objects while minor GCs It specifies (1) Old objects and (2) remembered shady objects.
  - Introduce rb_objspace_t::marked_objects which counts marked objects in current marking phase. marking count is needed to introduce incremental marking.
  - rename mark related function and sweep related function to gc_(marks|sweep)_(start|finish|step|rest|continue).
  - rename rgengc_report() to gc_report().
  - Add obj_info() function to get cstr of object details.
  - Add MEASURE_LINE() macro to measure execution time of specific line.
  - and many small fixes.
- include/ruby/ruby.h: add flag USE_RINCGC. Now USE_RINCGC can be set only with USE_RGENGC.
- include/ruby/ruby.h: introduce FL_PROMOTED0 and add FL_PROMOTED1 to count object age.
- include/ruby/ruby.h: rewrite write barriers for incremental marking.
- debug.c: catch up flag name changes.
- internal.h: add rb_gc_writebarrier_remember() instead of rb_gc_writebarrier_remember_promoted().
- array.c (ary_memcpy0): use rb_gc_writebarrier_remember().
- array.c (rb_ary_modify): ditto.
- hash.c (rb_hash_keys): ditto.
- hash.c (rb_hash_values): ditto.
- object.c (init_copy): use rb_copy_wb_protected_attribute() because FL_WB_PROTECTED is moved from RBasic::flags.
- test/objspace/test_objspace.rb: catch up ObjectSpace.dump() changes.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@47444 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 47444 - 09/08/2014 04:11 AM - ko1 (Koichi Sasada)

- gc.c: add incremental GC algorithm. [Feature #10137] Please refer this ticket for details. This change also introduces the following changes.
  - Remove RGENGC_AGE2_PROMOTION and introduce object age (0 to 3). Age can be count with FL_PROMOTE0 and FL_PROMOTE1 flags in RBasic::flags (2 bit). Age == 3 objects become old objects.
  - WB_PROTECTED flag in RBasic to WB_UNPROTECTED bitmap.
  - LONG_LIVED bitmap to represent living objects while minor GCs It specifies (1) Old objects and (2) remembered shady objects.
  - Introduce rb_objspace_t::marked_objects which counts marked objects in current marking phase. marking count is needed to introduce incremental marking.
  - rename mark related function and sweep related function to gc_(marks|sweep)_(start|finish|step|rest|continue).
  - rename rgengc_report() to gc_report().
  - Add obj_info() function to get cstr of object details.

    - Add MEASURE_LINE() macro to measure execution time of specific line.
    - and many small fixes.
- include/ruby/ruby.h: add flag USE_RINCGC. Now USE_RINCGC can be set only with USE_RGENGC.
- include/ruby/ruby.h: introduce FL_PROMOTED0 and add FL_PROMOTED1 to count object age.
- include/ruby/ruby.h: rewrite write barriers for incremental marking.
- debug.c: catch up flag name changes.
- internal.h: add rb_gc_writebarrier_remember() instead of rb_gc_writebarrier_remember_promoted().
- array.c (ary_memcpy0): use rb_gc_writebarrier_remember().
- array.c (rb_ary_modify): ditto.
- hash.c (rb_hash_keys): ditto.
- hash.c (rb_hash_values): ditto.
- object.c (init_copy): use rb_copy_wb_protected_attribute() because FL_WB_PROTECTED is moved from RBasic::flags.
- test/objspace/test_objspace.rb: catch up ObjectSpace.dump() changes.

Revision 47444 - 09/08/2014 04:11 AM - ko1 (Koichi Sasada)

- gc.c: add incremental GC algorithm. [Feature #10137] Please refer this ticket for details. This change also introduces the following changes.
    - Remove RGENGC_AGE2_PROMOTION and introduce object age (0 to 3). Age can be count with FL_PROMOTE0 and FL_PROMOTE1 flags in RBasic::flags (2 bit). Age == 3 objects become old objects.
    - WB_PROTECTED flag in RBasic to WB_UNPROTECTED bitmap.
    - LONG_LIVED bitmap to represent living objects while minor GCs It specifies (1) Old objects and (2) remembered shady objects.
    - Introduce rb_objspace_t::marked_objects which counts marked objects in current marking phase. marking count is needed to introduce incremental marking.
    - rename mark related function and sweep related function to gc_(marks|sweep)_(start|finish|step|rest|continue).
    - rename rgengc_report() to gc_report().
    - Add obj_info() function to get cstr of object details.
    - Add MEASURE_LINE() macro to measure execution time of specific line.
    - and many small fixes.
- include/ruby/ruby.h: add flag USE_RINCGC. Now USE_RINCGC can be set only with USE_RGENGC.
- include/ruby/ruby.h: introduce FL_PROMOTED0 and add FL_PROMOTED1 to count object age.
- include/ruby/ruby.h: rewrite write barriers for incremental marking.
- debug.c: catch up flag name changes.
- internal.h: add rb_gc_writebarrier_remember() instead of rb_gc_writebarrier_remember_promoted().
- array.c (ary_memcpy0): use rb_gc_writebarrier_remember().
- array.c (rb_ary_modify): ditto.
- hash.c (rb_hash_keys): ditto.
- hash.c (rb_hash_values): ditto.
- object.c (init_copy): use rb_copy_wb_protected_attribute() because FL_WB_PROTECTED is moved from RBasic::flags.
- test/objspace/test_objspace.rb: catch up ObjectSpace.dump() changes.

Revision 47444 - 09/08/2014 04:11 AM - ko1 (Koichi Sasada)

- gc.c: add incremental GC algorithm. [Feature #10137] Please refer this ticket for details. This change also introduces the following changes.
    - Remove RGENGC_AGE2_PROMOTION and introduce object age (0 to 3). Age can be count with FL_PROMOTE0 and FL_PROMOTE1 flags in RBasic::flags (2 bit). Age == 3 objects become old objects.
    - WB_PROTECTED flag in RBasic to WB_UNPROTECTED bitmap.
    - LONG_LIVED bitmap to represent living objects while minor GCs It specifies (1) Old objects and (2) remembered shady objects.
    - Introduce rb_objspace_t::marked_objects which counts marked objects in current marking phase. marking count is needed to introduce incremental marking.
    - rename mark related function and sweep related function to gc_(marks|sweep)_(start|finish|step|rest|continue).
    - rename rgengc_report() to gc_report().
    - Add obj_info() function to get cstr of object details.
    - Add MEASURE_LINE() macro to measure execution time of specific line.
    - and many small fixes.
- include/ruby/ruby.h: add flag USE_RINCGC. Now USE_RINCGC can be set only with USE_RGENGC.
- include/ruby/ruby.h: introduce FL_PROMOTED0 and add FL_PROMOTED1 to count object age.
- include/ruby/ruby.h: rewrite write barriers for incremental marking.
- debug.c: catch up flag name changes.
- internal.h: add rb_gc_writebarrier_remember() instead of rb_gc_writebarrier_remember_promoted().
- array.c (ary_memcpy0): use rb_gc_writebarrier_remember().
- array.c (rb_ary_modify): ditto.
- hash.c (rb_hash_keys): ditto.
- hash.c (rb_hash_values): ditto.
- object.c (init_copy): use rb_copy_wb_protected_attribute() because FL_WB_PROTECTED is moved from RBasic::flags.
- test/objspace/test_objspace.rb: catch up ObjectSpace.dump() changes.

Revision 47444 - 09/08/2014 04:11 AM - ko1 (Koichi Sasada)

- gc.c: add incremental GC algorithm. [Feature #10137] Please refer this ticket for details. This change also introduces the following changes.
    - Remove RGENGC_AGE2_PROMOTION and introduce object age (0 to 3). Age can be count with FL_PROMOTE0 and FL_PROMOTE1 flags in RBasic::flags (2 bit). Age == 3 objects become old objects.
    - WB_PROTECTED flag in RBasic to WB_UNPROTECTED bitmap.

    - LONG_LIVED bitmap to represent living objects while minor GCs It specifies (1) Old objects and (2) remembered shady objects.
    - Introduce rb_objspace_t::marked_objects which counts marked objects in current marking phase. marking count is needed to introduce incremental marking.
    - rename mark related function and sweep related function to gc_(marks|sweep)_(start|finish|step|rest|continue).
    - rename rgengc_report() to gc_report().
    - Add obj_info() function to get cstr of object details.
    - Add MEASURE_LINE() macro to measure execution time of specific line.
    - and many small fixes.
- include/ruby/ruby.h: add flag USE_RINCGC. Now USE_RINCGC can be set only with USE_RGENGC.
- include/ruby/ruby.h: introduce FL_PROMOTED0 and add FL_PROMOTED1 to count object age.
- include/ruby/ruby.h: rewrite write barriers for incremental marking.
- debug.c: catch up flag name changes.
- internal.h: add rb_gc_writebarrier_remember() instead of rb_gc_writebarrier_remember_promoted().
- array.c (ary_memcpy0): use rb_gc_writebarrier_remember().
- array.c (rb_ary_modify): ditto.
- hash.c (rb_hash_keys): ditto.
- hash.c (rb_hash_values): ditto.
- object.c (init_copy): use rb_copy_wb_protected_attribute() because FL_WB_PROTECTED is moved from RBasic::flags.
- test/objspace/test_objspace.rb: catch up ObjectSpace.dump() changes.

Revision 47444 - 09/08/2014 04:11 AM - ko1 (Koichi Sasada)

- gc.c: add incremental GC algorithm. [Feature #10137] Please refer this ticket for details. This change also introduces the following changes.
    - Remove RGENGC_AGE2_PROMOTION and introduce object age (0 to 3). Age can be count with FL_PROMOTE0 and FL_PROMOTE1 flags in RBasic::flags (2 bit). Age == 3 objects become old objects.
    - WB_PROTECTED flag in RBasic to WB_UNPROTECTED bitmap.
    - LONG_LIVED bitmap to represent living objects while minor GCs It specifies (1) Old objects and (2) remembered shady objects.
    - Introduce rb_objspace_t::marked_objects which counts marked objects in current marking phase. marking count is needed to introduce incremental marking.
    - rename mark related function and sweep related function to gc_(marks|sweep)_(start|finish|step|rest|continue).
    - rename rgengc_report() to gc_report().
    - Add obj_info() function to get cstr of object details.
    - Add MEASURE_LINE() macro to measure execution time of specific line.
    - and many small fixes.
- include/ruby/ruby.h: add flag USE_RINCGC. Now USE_RINCGC can be set only with USE_RGENGC.
- include/ruby/ruby.h: introduce FL_PROMOTED0 and add FL_PROMOTED1 to count object age.
- include/ruby/ruby.h: rewrite write barriers for incremental marking.
- debug.c: catch up flag name changes.
- internal.h: add rb_gc_writebarrier_remember() instead of rb_gc_writebarrier_remember_promoted().
- array.c (ary_memcpy0): use rb_gc_writebarrier_remember().
- array.c (rb_ary_modify): ditto.
- hash.c (rb_hash_keys): ditto.
- hash.c (rb_hash_values): ditto.
- object.c (init_copy): use rb_copy_wb_protected_attribute() because FL_WB_PROTECTED is moved from RBasic::flags.
- test/objspace/test_objspace.rb: catch up ObjectSpace.dump() changes.

Revision 47444 - 09/08/2014 04:11 AM - ko1 (Koichi Sasada)

- gc.c: add incremental GC algorithm. [Feature #10137] Please refer this ticket for details. This change also introduces the following changes.
    - Remove RGENGC_AGE2_PROMOTION and introduce object age (0 to 3). Age can be count with FL_PROMOTE0 and FL_PROMOTE1 flags in RBasic::flags (2 bit). Age == 3 objects become old objects.
    - WB_PROTECTED flag in RBasic to WB_UNPROTECTED bitmap.
    - LONG_LIVED bitmap to represent living objects while minor GCs It specifies (1) Old objects and (2) remembered shady objects.
    - Introduce rb_objspace_t::marked_objects which counts marked objects in current marking phase. marking count is needed to introduce incremental marking.
    - rename mark related function and sweep related function to gc_(marks|sweep)_(start|finish|step|rest|continue).
    - rename rgengc_report() to gc_report().
    - Add obj_info() function to get cstr of object details.
    - Add MEASURE_LINE() macro to measure execution time of specific line.
    - and many small fixes.
- include/ruby/ruby.h: add flag USE_RINCGC. Now USE_RINCGC can be set only with USE_RGENGC.
- include/ruby/ruby.h: introduce FL_PROMOTED0 and add FL_PROMOTED1 to count object age.
- include/ruby/ruby.h: rewrite write barriers for incremental marking.
- debug.c: catch up flag name changes.
- internal.h: add rb_gc_writebarrier_remember() instead of rb_gc_writebarrier_remember_promoted().
- array.c (ary_memcpy0): use rb_gc_writebarrier_remember().
- array.c (rb_ary_modify): ditto.
- hash.c (rb_hash_keys): ditto.
- hash.c (rb_hash_values): ditto.
- object.c (init_copy): use rb_copy_wb_protected_attribute() because FL_WB_PROTECTED is moved from RBasic::flags.
- test/objspace/test_objspace.rb: catch up ObjectSpace.dump() changes.

History

#1 - 08/15/2014 06:01 AM - ko1 (Koichi Sasada)

- Description updated

Add an abstract section.

#2 - 08/15/2014 06:15 AM - mrkn (Kenta Murata)

- Description updated

#3 - 08/17/2014 02:55 AM - hsbt (Hiroshi SHIBATA)

I evaluated running time for test-all.

ref. https://twitter.com/k_tsj/status/500523453028388864

rincgc branch seems 2 times slower than trunk when sequential running.
I tested 3 times. it's same results.

trunk

ruby -v: ruby 2.2.0dev (2014-08-17 trunk 47206) [x86_64-darwin14]
make test-all TESTS='-j4' 291.72s user 65.89s system 241% cpu 2:27.94 total

ruby -v: ruby 2.2.0dev (2014-08-17 trunk 47206) [x86_64-darwin14]
make test-all 336.31s user 62.33s system 83% cpu 7:55.63 total

rincgc branch

ruby -v: ruby 2.2.0dev (2014-08-15 trunk 47192) [x86_64-darwin14]
make test-all TESTS='-j4' 359.57s user 74.27s system 213% cpu 3:23.33 total

ruby -v: ruby 2.2.0dev (2014-08-15 trunk 47192) [x86_64-darwin14]
make test-all 878.35s user 65.17s system 90% cpu 17:22.43 total

#4 - 08/17/2014 05:28 AM - ko1 (Koichi Sasada)

I had accidentally added GC.verify_internal_consistency method for each
test case (in after_tear_down.

After remove it,

trunk   : 798.756056s
before removal: 1981.145346s
after removal : 845.399831s

Not so bad.

#5 - 08/17/2014 05:55 AM - ko1 (Koichi Sasada)

FYI: here is a top slow tests with data.
http://www.atdot.net/fp_store/f.zdsfan/file.copipa-temp-image.png

Typically, GC.start slows down.

#6 - 08/27/2014 02:47 PM - matz (Yukihiro Matsumoto)

Go ahead. We need to experiment in real use cases.

Matz.

#7 - 09/08/2014 04:11 AM - ko1 (Koichi Sasada)

- Status changed from Open to Closed

- % Done changed from 0 to 100

Applied in changeset r47444.

- gc.c: add incremental GC algorithm. [Feature #10137] Please refer this ticket for details. This change also introduces the following changes.
  - Remove RGENGC_AGE2_PROMOTION and introduce object age (0 to 3). Age can be count with FL_PROMOTE0 and FL_PROMOTE1 flags in RBasic::flags (2 bit). Age == 3 objects become old objects.
  - WB_PROTECTED flag in RBasic to WB_UNPROTECTED bitmap.

    - LONG_LIVED bitmap to represent living objects while minor GCs It specifies (1) Old objects and (2) remembered shady objects.
    - Introduce rb_objspace_t::marked_objects which counts marked objects in current marking phase. marking count is needed to introduce incremental marking.
    - rename mark related function and sweep related function to gc_(marks|sweep)_(start|finish|step|rest|continue).
    - rename rgengc_report() to gc_report().
    - Add obj_info() function to get cstr of object details.
    - Add MEASURE_LINE() macro to measure execution time of specific line.
    - and many small fixes.
- include/ruby/ruby.h: add flag USE_RINCGC. Now USE_RINCGC can be set only with USE_RGENGC.
- include/ruby/ruby.h: introduce FL_PROMOTED0 and add FL_PROMOTED1 to count object age.
- include/ruby/ruby.h: rewrite write barriers for incremental marking.
- debug.c: catch up flag name changes.
- internal.h: add rb_gc_writebarrier_remember() instead of rb_gc_writebarrier_remember_promoted().
- array.c (ary_memcpy0): use rb_gc_writebarrier_remember().
- array.c (rb_ary_modify): ditto.
- hash.c (rb_hash_keys): ditto.
- hash.c (rb_hash_values): ditto.
- object.c (init_copy): use rb_copy_wb_protected_attribute() because FL_WB_PROTECTED is moved from RBasic::flags.
- test/objspace/test_objspace.rb: catch up ObjectSpace.dump() changes.