Ruby trunk - Feature #11003

Fast modular exponentiation

03/25/2015 04:51 PM - venkatvb (venkatesh babu)

Status:	Closed	
Priority:	Normal	
Assignee:	matz (Yukihiro Matsumoto)	
Target version:		

Description

I would like to suggest, implementing "fast Modular Exponentiation " (http://en.wikipedia.org/wiki/Modular_exponentiation) for fixnum class.
Eg: A function like pow(a, n, MOD) can be computed more efficiently than (a**n) % MOD

Related issues:

Is duplicate of CommonRuby - Feature #12508: Integer#mod_pow	**Closed**

Associated revisions

Revision 9b09cc8a - 12/04/2017 02:35 AM - mrkn (Kenta Murata)

bignum.c, numeric.c: add Integer#pow(b, m)

This commit is based on the pull-request #1320 created by Makoto Kishimoto.
[Feature #12508] [Feature #11003] [close GH-1320]

- bignum.c (rb_int_powm): Added for Integer#pow(b, m).
- internal.h (rb_int_powm): Declared to refer in numeric.c.
- bignum.c (bary_powm_gmp): Added for Integer#pow(b, m) using GMP.
- bignum.c (int_pow_tmp1): Added for implementing Integer#pow(b, m).
- bignum.c (int_pow_tmp2, int_pow_tmp3): ditto.
- internal.h (rb_num_positive_int_p): Moved from numeric.c for sharing the definition with bignum.c.
- internal.h (rb_num_negative_int_p, rb_num_compare_with_zero): ditto.
- numeric.c(negative_int_p): Moved to internal.h for sharing the definition with bignum.c.
- numeric.c (positive_int_p, compare_with_zero): ditto.
- numeric.c (rb_int_odd_p): Exported (renamed from int_odd_p).
- internal.h (rb_int_odd_p): ditto.
- internal.h (HALF_LONG_MSB): Added.
- numeric.c (SQRT_LONG_MAX): Redefined by using HALF_LONG_MSB.
- test/ruby/test_numeric.rb (test_pow): Added for Integer#pow(b, m).

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@61003 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 61003 - 12/04/2017 02:35 AM - mrkn (Kenta Murata)

bignum.c, numeric.c: add Integer#pow(b, m)

This commit is based on the pull-request #1320 created by Makoto Kishimoto.
[Feature #12508] [Feature #11003] [close GH-1320]

- bignum.c (rb_int_powm): Added for Integer#pow(b, m).
- internal.h (rb_int_powm): Declared to refer in numeric.c.

- bignum.c (bary_powm_gmp): Added for Integer#pow(b, m) using GMP.
- bignum.c (int_pow_tmp1): Added for implementing Integer#pow(b, m).
- bignum.c (int_pow_tmp2, int_pow_tmp3): ditto.
- internal.h (rb_num_positive_int_p): Moved from numeric.c for sharing the definition with bignum.c.
- internal.h (rb_num_negative_int_p, rb_num_compare_with_zero): ditto.
- numeric.c(negative_int_p): Moved to internal.h for sharing the definition with bignum.c.
- numeric.c (positive_int_p, compare_with_zero): ditto.
- numeric.c (rb_int_odd_p): Exported (renamed from int_odd_p).
- internal.h (rb_int_odd_p): ditto.
- internal.h (HALF_LONG_MSB): Added.
- numeric.c (SQRT_LONG_MAX): Redefined by using HALF_LONG_MSB.
- test/ruby/test_numeric.rb (test_pow): Added for Integer#pow(b, m).

Revision 61003 - 12/04/2017 02:35 AM - mrkn (Kenta Murata)

bignum.c, numeric.c: add Integer#pow(b, m)

This commit is based on the pull-request #1320 created by Makoto Kishimoto.
[Feature #12508] [Feature #11003] [close GH-1320]

- bignum.c (rb_int_powm): Added for Integer#pow(b, m).
- internal.h (rb_int_powm): Declared to refer in numeric.c.
- bignum.c (bary_powm_gmp): Added for Integer#pow(b, m) using GMP.
- bignum.c (int_pow_tmp1): Added for implementing Integer#pow(b, m).
- bignum.c (int_pow_tmp2, int_pow_tmp3): ditto.
- internal.h (rb_num_positive_int_p): Moved from numeric.c for sharing the definition with bignum.c.
- internal.h (rb_num_negative_int_p, rb_num_compare_with_zero): ditto.
- numeric.c(negative_int_p): Moved to internal.h for sharing the definition with bignum.c.
- numeric.c (positive_int_p, compare_with_zero): ditto.
- numeric.c (rb_int_odd_p): Exported (renamed from int_odd_p).
- internal.h (rb_int_odd_p): ditto.
- internal.h (HALF_LONG_MSB): Added.
- numeric.c (SQRT_LONG_MAX): Redefined by using HALF_LONG_MSB.
- test/ruby/test_numeric.rb (test_pow): Added for Integer#pow(b, m).

Revision 61003 - 12/04/2017 02:35 AM - mrkn (Kenta Murata)

bignum.c, numeric.c: add Integer#pow(b, m)

This commit is based on the pull-request #1320 created by Makoto Kishimoto.
[Feature #12508] [Feature #11003] [close GH-1320]

- bignum.c (rb_int_powm): Added for Integer#pow(b, m).

- internal.h (rb_int_powm): Declared to refer in numeric.c.
- bignum.c (bary_powm_gmp): Added for Integer#pow(b, m) using GMP.
- bignum.c (int_pow_tmp1): Added for implementing Integer#pow(b, m).
- bignum.c (int_pow_tmp2, int_pow_tmp3): ditto.
- internal.h (rb_num_positive_int_p): Moved from numeric.c for sharing the definition with bignum.c.
- internal.h (rb_num_negative_int_p, rb_num_compare_with_zero): ditto.
- numeric.c(negative_int_p): Moved to internal.h for sharing the definition with bignum.c.
- numeric.c (positive_int_p, compare_with_zero): ditto.
- numeric.c (rb_int_odd_p): Exported (renamed from int_odd_p).
- internal.h (rb_int_odd_p): ditto.
- internal.h (HALF_LONG_MSB): Added.
- numeric.c (SQRT_LONG_MAX): Redefined by using HALF_LONG_MSB.
- test/ruby/test_numeric.rb (test_pow): Added for Integer#pow(b, m).

Revision 956cfb97 - 12/06/2017 12:36 PM - nobu (Nobuyoshi Nakada)

numeric.c: rb_int_powm rdoc

- numeric.c (Init_Numeric): let rdoc know that rb_int_powm is defined in bignum.c. [Feature #12508] [Feature #11003]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@61057 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 61057 - 12/06/2017 12:36 PM - nobu (Nobuyoshi Nakada)

numeric.c: rb_int_powm rdoc

- numeric.c (Init_Numeric): let rdoc know that rb_int_powm is defined in bignum.c. [Feature #12508] [Feature #11003]

Revision 61057 - 12/06/2017 12:36 PM - nobu (Nobuyoshi Nakada)

numeric.c: rb_int_powm rdoc

- numeric.c (Init_Numeric): let rdoc know that rb_int_powm is defined in bignum.c. [Feature #12508] [Feature #11003]

Revision 61057 - 12/06/2017 12:36 PM - nobu (Nobuyoshi Nakada)

numeric.c: rb_int_powm rdoc

- numeric.c (Init_Numeric): let rdoc know that rb_int_powm is defined in bignum.c. [Feature #12508] [Feature #11003]

History

#1 - 03/30/2015 04:14 AM - mrkn (Kenta Murata)

- Description updated

#2 - 03/30/2015 04:18 AM - mrkn (Kenta Murata)

- Status changed from Open to Feedback

- Assignee set to matz (Yukihiro Matsumoto)

I have two questions:

- Do you have some concrete use cases in which this new feature is used?
- Why don't you make a gem library to provide this feature? or Are there gem libraries providing it?

#3 - 12/01/2017 08:33 AM - mrkn (Kenta Murata)

- Is duplicate of Feature #12508: Integer#mod_pow added

#4 - 12/01/2017 08:33 AM - mrkn (Kenta Murata)

- Status changed from Feedback to Closed

The same function was accepted in #12508