Random number generation with xoshiro

01/07/2022 06:34 AM - bbrklm (Benson Muite)

Status: Open
Priority: Normal
Assignee:
Target version:

Description
Xoshiro [https://prng.di.unimi.it/](https://prng.di.unimi.it/) random number generation is typically faster than Mersenne Twister currently used in Ruby in [https://github.com/ruby/ruby/blob/master/random.c](https://github.com/ruby/ruby/blob/master/random.c). It would be good to allow use of xoshiro either as an option or as default. Xoshiro is the default for Fortran [https://gcc.gnu.org/onlinedocs/gfortran/RANDOM_005fNUMBER.html](https://gcc.gnu.org/onlinedocs/gfortran/RANDOM_005fNUMBER.html) and Julia [https://github.com/JuliaLang/julia/tree/master/stdlib/Random/src](https://github.com/JuliaLang/julia/tree/master/stdlib/Random/src), and it's happy to implement this.

Related issues:
- Related to Ruby master - Feature #16827: C API for writing custom random numb... Closed

History

#1 - 01/07/2022 06:55 AM - shyouhei (Shyouhei Urabe)
I guess you can start implementing xoshiro as a separate gem now. I remember [@mrkn (Kenta Murata)](https://github.com/mrkn) once wanted to implement xorshift, and include/ruby/random.h was introduced (mainly) for that purpose. Not sure his current status though.

#2 - 01/07/2022 07:11 AM - bbrklm (Benson Muite)
Thanks for the feedback. Another good algorithm is pcg [https://rubygems.org/gems/pcg_random](https://rubygems.org/gems/pcg_random)

#3 - 01/07/2022 11:02 AM - mame (Yusuke Endoh)
- Related to Feature #16827: C API for writing custom random number generator that can be used as Random objects added

#4 - 01/08/2022 09:55 AM - nobu (Nobuyoshi Nakada)
That gem seems pretty old, before include/ruby/random.h was introduced. You can see a skeleton at ext/-test-/random/loop.c too.

#5 - 02/11/2022 03:26 PM - bbrklm (Benson Muite)
A work in progress Gem can be found at [https://gitlab.com/bkmgit/xoshiro256plusplus](https://gitlab.com/bkmgit/xoshiro256plusplus)
The default in [https://github.com/ruby/ruby/blob/master/random.c](https://github.com/ruby/ruby/blob/master/random.c) is to use uint32_t type in C. Will uint64_t also be supported when it is available?

It is also possible to just use xoshiro128++ which generates uint32_t type, which has portability and maintenance advantage, but also a speed disadvantage.

#6 - 02/12/2022 01:18 PM - bbrklm (Benson Muite)
Work in progress gem for xoshiro128++ [0]

Python continues to use Mersenne twister as default[1], but Numpy has been recently extended to include other options[2]. Some considerations for C++ are discussed in [3].

Randen [4] calls hardware implemented functions, but these may not be available on all architectures.

If there is a need for backwards compatibility, Mersenne twister can be kept as default, for example the Go community has chosen to keep an unusual random number generator as default[5][6][7][8]. My expectation is that xoshiro128++ is a reasonable general purpose replacement for Mersenne twister for core language. Java also chains together random sequences of bytes to generate random big integers[9], so this is probably ok for portability of the base language, though tests and other quality assurances would be nice as well. More specialized random number generators might be better in a separate gem.

One further issue might be using the random number generator with Ractors. Some of the random number generators provide a jump function to ensure that random sequences generated at in parallel do not overlap for a large fixed set of invocations. This is not in the interfaces available in include/ruby/random.h. GPU libraries also often make several independent streams of random numbers.

---

[0] [https://gitlab.com/bkmgit/xoshiro128plusplus](https://gitlab.com/bkmgit/xoshiro128plusplus)
[1] [https://docs.python.org/3/library/random.html](https://docs.python.org/3/library/random.html)
[2] [https://github.com/bashtage/randomgen/](https://github.com/bashtage/randomgen/)
@bbrklm (Benson Muite) I want the Xoshiro256 generator for Ruby, too. In addition to it, I want the dSFMT generator.

I think it is good to create a semi-standard gem library for providing such random number generators. The existence of the semi-standard library can prevent confusion by emerging many different libraries. I’m working on creating such a semi-standard gem library in a private repository under the ruby organization. Currently, I’m implementing SFMT and dSFMT in the repository. I want to include xoshiro-family and PCG generators in the repository before releasing it.

Would you please help to implement generators in the repository?


The Rust library has good documentation[6], as well as speed comparisons. Would be good to have something similar for Ruby.

Related issues on SIMD #16487 and #14328. SIMD.json[8] only has Ruby bindings, but Go and Rust ports because these languages have SIMD libraries. Maybe it would be good to have a SIMD gem for Ruby? Portability is also required, so while the approach in [8] gives some idea of performance improvements, using approaches similar to PeachPy [10] or Portable SIMD [11] might be better. SFMT implementation may give suggestions on how to make SIMD capabilities available for other native Gems.

[1] https://github.com/vigna/MRG32k3a
[8] https://simdjson.org/software/
[10] https://github.com/Maratyszcza/PeachPy