Parallel RNG Manager

Generated by Doxygen 1.8.6

Fri Feb 15 2019 07:20:55

Contents

1 Main Page

```
![Build Status](https://travis-ci.org/markjolah/ParallelRngManager.svg?branch=master)
```

Parallel RNG Manager

A ParallelRngManager simplifies the task of initializing and coordinating random number generation for multiple threads in OpenMP and other multi-threaded programming environments without the need for locks or the possibility of false sharing. A single integer value is used to seed a single random number generator that is partitioned into independent parallel random number generator streams.

Using a single random number generator seed makes deterministic testing and debugging of parallel stochastic algorithms practical. Additionally it is important to use a random number generator specifically designed for parallel use, as it is not in general safe to use independent random seeds for each processor if strong randomness properties and guaranteed a-correlation of the streams are arithmetically important considerations.

More generally, a *parallel random number generator* (PRNG) provides a set of N random number generator streams for multi-threaded applications, where each stream is produced from a single underlying random number generator with a single global seed. For certain classes of random number generators, a single stream can efficiently be partitioned into N threads without communication overhead. The `'parallel_rng::ParallelRngManager'` class functions as an OpenMP-aware manager for the PRNGs from the `Tina's Random Number Generator (TRNG) Library`.

Features

- *ParallelRngManager* is CMake based, and provides `ParallelRngManagerConfig.cmake` files allowing `find_pacakge(ParallelRngManager)` to find the package in either the build or install trees.
- *ParallelRngManager* can automatically configure and install TRNG and alongside itself if it does not exist on the system.
- *ParallelRngManager* is designed to work seamlessly with OpenMP. It automatically manages the number of RNG streams based on hardware concurrency and prevents false sharing.
- A *ParallelRngManager* object manages a single stream and uses OpenMP `get_num_threads()` to allocate the correct number of sub-streams, which are kept on separate cache lines using `'aligned_array::AArray<RngT>'`.

Installation

The easiest method is to use the default build script, which can be easily customized. The default build directory is `./_build` and the default install directory is `./_install`.

‘‘‘

git clone `https://github.com/markjolah/ParallelRngManager.git` cd ParallelRngManager ./build.sh

‘‘

If TRNG is not available on the system, it is important to have `CMAKE_INSTALL_PREFIX` set to a valid install directory, even if it is just a local directory, as the autotools build is designed to install into the `CMAKE_INSTALL_PREFIX` and ParallelRngManager is then expecting to find the TRNG library there.

CMake options

The following CMake options control the build.

- `BUILD_SHARED_LIBS` - Build shared libraries
- `BUILD_STATIC_LIBS` - Build static libraries
- `BUILD_TESTING` - Build testing framework
- `OPT_DOC` - Build documentation
- `OPT_INSTALL_TESTING` - Install testing executables in install-tree.
- `OPT_EXPORT_BUILD_TREE` - Configure the package so it is usable from the build tree. Useful for development.
- `OPT_ARMADILLO_INT64` - Use 64-bit integers for Armadillo, BLAS, and LAPACK.

Dependencies

ParallelRngManager is designed to be portable, but relies on several system development and numerical libraries. Currently Travis CI uses the *trusty* image to test ParallelRngManager Standard system dependencies

- ∗>=g++-4.9∗ - A `--std=c++11` compliant GCC compiler
- ∗>=CMake-3.9∗
- `*OpenMP*`
- `*Armadillo*` - A high-performance array library for C++.
- `*googletest*` - Required for testing (`BUILD_TESTING=On`)
- `*Doxygen*` - Required to generate documentation (`OPT_DOC=On`)
  - *graphviz* - Required to generate documentation (`make doc`)
  - *LAPACK* - Required for generate pdf documenation (`make pdf`)

Tina's Random Number Generator (TRNG)

The ParallelRngManager is a lightweight wrapper around the `Tina's Random Number Generator (TRNG)` library. This rather specialized numerical library is normally not available on most Linux distributions, so for convenience the ParallelRngManager CMake build system will automatically download, configure, build, and install TRNG (`libtrng4.so`) into the `CMAKE_INSTALL_PREFIX` path if it is not already present on the build system. This process uses the `AddExternalAutotoolsDependency.cmake` function from the `UncommonCMakeModules` dependency.

- `TRNG Manual`
- `H. Bauke and S. Mertens. _Random Numbers for Large Scale Distributed Monte Carlo Simulations_.`

Other dependencies

ParallelRngManager uses these reusable header-only component libraries via `'git subrepo'`

- `AlignedArray` - Provides `aligned_array::AArray<T>` which is an STL conforming fixed-length array container which guarantees no two elements share a cache line, preventing false sharing in multi-threaded or OpenMP programs. ParallelRngManager stores RNG streams in an `AArray<RngT>` array to prevent false sharing.
- `AnyRng` - Provides `any_rng::AnyRng<result_type>` which is a type-erased STL random number generator type.
- `UncommonCMakeModules` - Provides `FindTRNG.cmake` `FindArmadillo.cmake` and other useful CMake functions like `ExportPackageWizzard.cmake`. ParallelRngManager only uses a small portion of these CMake modules but using a `git subrepo` pulls in the entire repository.

Documentation

The ParallelRngManager Doxygen documentation can be build with the `OPT_DOC` CMake option and is also available on online:

- `ParallelRngManager HTML Manual`
- `ParallelRngManager PDF Manual`

Testing

ParallelRngManager uses `googletest` for C++ unit testing and integrates with CTest. To build tests, enable the `BUILD_TESTING` CMake option and possibly also the `OPT_INSTALL_TESTING` option to install tests along with ParallelRngManager.

Tests can be run with: “‘

```
make test
```

“‘

2 Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

3 Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

std::exception

4 Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

5 File Index

5.1 File List

Here is a list of all files with brief descriptions:

6 Namespace Documentation

6.1 parallel_rng Namespace Reference

Classes

- class ParallelRngManagerError
- class ParallelRngManager

Typedefs

- using DefaultParallelRngT = trng::lcg64_shift

  Suggested default ParallelRNG type.
- using SeedT = uint64_t

  Use the true random interface to generate a truly random seed.
- using IdxT = arma::uword

Functions

- SeedT generate_seed ()
- IdxT openmp_estimate_max_threads ()

Use openmp to estimate the maximum number of threads that will be generated.

- template<class RngT = DefaultParallelRngT, class FloatT = double>
  ParallelRngManager< RngT, FloatT > make_parallel_rng_manager ()
- template<class RngT = DefaultParallelRngT, class FloatT = double>
  ParallelRngManager< RngT, FloatT > make_parallel_rng_manager (SeedT seed)

Variables

- rngs {num_threads,cache_alignment, RngT{seeder}}
- norm_dist {num_threads,cache_alignment, NormalDistT{}}

6.1.1 Typedef Documentation

6.1.1.1 using parallel_rng::DefaultParallelRngT = typedef trng::lcg64_shift

Suggested default ParallelRNG type.

lcg64_shift is one of the fastest ParallelRNG types with shifting to correct for poor lower order bit randomness in regular lcg64

Definition at line 58 of file ParallelRngManager.h.

6.1.1.2 using parallel_rng::IdxT = typedef arma::uword

Definition at line 72 of file ParallelRngManager.h.

6.1.1.3 using parallel_rng::SeedT = typedef uint64_t

Use the true random interface to generate a truly random seed.

Definition at line 71 of file ParallelRngManager.h.

6.1.2 Function Documentation

6.1.2.1 SeedT parallel_rng::generate_seed ( )

Definition at line 14 of file ParallelRngManager.cpp.

6.1.2.2 template<class RngT = DefaultParallelRngT, class FloatT = double> ParallelRngManager<RngT,FloatT> parallel_rng::make_parallel_rng_manager ( )

Definition at line 143 of file ParallelRngManager.h.

6.1.2.3 template<class RngT = DefaultParallelRngT, class FloatT = double> ParallelRngManager<RngT,FloatT> parallel_rng::make_parallel_rng_manager (SeedT *seed*)

Definition at line 149 of file ParallelRngManager.h.

6.1.2.4 IdxT parallel_rng::openmp_estimate_max_threads ( )

Use openmp to estimate the maximum number of threads that will be generated.

Definition at line 20 of file ParallelRngManager.cpp.

6.1.3 Variable Documentation

6.1.3.1 parallel_rng::norm_dist {num_threads,cache_alignment, NormalDistT{}}

Definition at line 173 of file ParallelRngManager.h.

6.1.3.2 parallel_rng::rngs {num_threads,cache_alignment, RngT{seeder}}

Definition at line 172 of file ParallelRngManager.h.

7 Class Documentation

7.1 parallel_rng::ParallelRngManager< RngT, FloatT > Class Template Reference

`#include </home/travis/build/markjolah/ParallelRngManager/include/ParallelRngManager/ParallelRngManager.h>`

Public Types

- using VecT = arma::Col< FloatT >
- using MatT = arma::Mat< FloatT >
- using NormalDistT = std::normal_distribution< FloatT >
- using UniformDistT = std::uniform_real_distribution< FloatT >
- using result_type = typename RngT::result_type

Public Member Functions

- ParallelRngManager ()
- ParallelRngManager (SeedT seed)
- ParallelRngManager (SeedT seed, IdxT max_threads)
- void seed (SeedT seed)
- void reset ()
- void reset (SeedT seed)
- void reset (SeedT seed, IdxT max_threads)
- SeedT get_init_seed () const
- SeedT get_num_threads () const
- RngT & generator ()
- any_rng::AnyRng< result_type > generic_generator ()
- result_type operator() ()
- FloatT randu ()
- FloatT randn ()
- VecT randu (IdxT N)
- VecT randn (IdxT N)
- MatT randu (IdxT rows, IdxT cols)
- MatT randn (IdxT rows, IdxT cols)
- template<class Weights = VecT, class IdxT = IdxT>
  IdxT resample_dist (const Weights &weights)
- template<class Weights = VecT, class IdxT = IdxT>
  arma::Col< IdxT > resample_dist (const Weights &weights, IdxT N)

7.1.1 Detailed Description

template<class RngT = DefaultParallelRngT, class FloatT = double>class parallel_rng::ParallelRngManager< RngT, FloatT >

Definition at line 80 of file ParallelRngManager.h.

7.1.2 Member Typedef Documentation

7.1.2.1 template<class RngT = DefaultParallelRngT, class FloatT = double> using parallel_rng::ParallelRngManager< RngT, FloatT >::MatT = arma::Mat<FloatT>

Definition at line 84 of file ParallelRngManager.h.

7.1.2.2 template<class RngT = DefaultParallelRngT, class FloatT = double> using parallel_rng::ParallelRngManager< RngT, FloatT >::NormalDistT = std::normal_distribution<FloatT>

Definition at line 85 of file ParallelRngManager.h.

7.1.2.3 template<class RngT = DefaultParallelRngT, class FloatT = double> using parallel_rng::ParallelRngManager< RngT, FloatT >::result_type = typename RngT::result_type

Definition at line 87 of file ParallelRngManager.h.

7.1.2.4 template<class RngT = DefaultParallelRngT, class FloatT = double> using parallel_rng::ParallelRngManager< RngT, FloatT >::UniformDistT = std::uniform_real_distribution<FloatT>

Definition at line 86 of file ParallelRngManager.h.

7.1.2.5 template<class RngT = DefaultParallelRngT, class FloatT = double> using parallel_rng::ParallelRngManager< RngT, FloatT >::VecT = arma::Col<FloatT>

Definition at line 83 of file ParallelRngManager.h.

7.1.3 Constructor & Destructor Documentation

7.1.3.1 template<class RngT , class FloatT > parallel_rng::ParallelRngManager< RngT, FloatT >::ParallelRngManager ( )

Definition at line 157 of file ParallelRngManager.h.

7.1.3.2 template<class RngT , class FloatT > parallel_rng::ParallelRngManager< RngT, FloatT >::ParallelRngManager (SeedT *seed*)

Definition at line 162 of file ParallelRngManager.h.

7.1.3.3 template<class RngT , class FloatT > parallel_rng::ParallelRngManager< RngT, FloatT >::ParallelRngManager (SeedT *seed,* IdxT *max_threads*)

Definition at line 167 of file ParallelRngManager.h.

7.1.4 Member Function Documentation

7.1.4.1 template<class RngT = DefaultParallelRngT, class FloatT = double> RngT& parallel_rng::ParallelRngManager< RngT, FloatT >::generator ( )

7.1.4.2 template<class RngT = DefaultParallelRngT, class FloatT = double> any_rng::AnyRng<result_type> parallel_rng::ParallelRngManager< RngT, FloatT >::generic_generator ( )

7.1.4.3 template<class RngT = DefaultParallelRngT, class FloatT = double> SeedT parallel_rng::ParallelRngManager< RngT, FloatT >::get_init_seed ( ) const

7.1.4.4 template<class RngT = DefaultParallelRngT, class FloatT = double> SeedT parallel_rng::ParallelRngManager< RngT, FloatT >::get_num_threads ( ) const

7.1.4.5 template<class RngT = DefaultParallelRngT, class FloatT = double> result_type parallel_rng::ParallelRngManager< RngT, FloatT >::operator() ( )

7.1.4.6 template<class RngT = DefaultParallelRngT, class FloatT = double> FloatT parallel_rng::ParallelRngManager< RngT, FloatT >::randn ( )

7.1.4.7 template<class RngT = DefaultParallelRngT, class FloatT = double> VecT parallel_rng::ParallelRngManager< RngT, FloatT >::randn (IdxT *N*)

7.1.4.8 template<class RngT = DefaultParallelRngT, class FloatT = double> MatT parallel_rng::ParallelRngManager< RngT, FloatT >::randn (IdxT *rows,* IdxT *cols*)

7.1.4.9 template<class RngT = DefaultParallelRngT, class FloatT = double> FloatT parallel_rng::ParallelRngManager< RngT, FloatT >::randu ( )

7.1.4.10 template<class RngT = DefaultParallelRngT, class FloatT = double> VecT parallel_rng::ParallelRngManager< RngT, FloatT >::randu (IdxT *N*)

7.1.4.11 template<class RngT = DefaultParallelRngT, class FloatT = double> MatT parallel_rng::ParallelRngManager< RngT, FloatT >::randu (IdxT *rows,* IdxT *cols*)

7.1.4.12 template<class RngT = DefaultParallelRngT, class FloatT = double> template<class Weights = VecT, class IdxT = IdxT> IdxT parallel_rng::ParallelRngManager< RngT, FloatT >::resample_dist (const Weights & *weights*)

7.1.4.13 template<class RngT = DefaultParallelRngT, class FloatT = double> template<class Weights = VecT, class IdxT = IdxT> arma::Col<IdxT> parallel_rng::ParallelRngManager< RngT, FloatT >::resample_dist (const Weights & *weights,* IdxT *N*)

7.1.4.14 template<class RngT = DefaultParallelRngT, class FloatT = double> void parallel_rng::ParallelRngManager< RngT, FloatT >::reset ( )

7.1.4.15 template<class RngT = DefaultParallelRngT, class FloatT = double> void parallel_rng::ParallelRngManager< RngT, FloatT >::reset (SeedT *seed*)

7.1.4.16 template<class RngT = DefaultParallelRngT, class FloatT = double> void parallel_rng::ParallelRngManager< RngT, FloatT >::reset (SeedT *seed,* IdxT *max_threads*)

7.1.4.17 template<class RngT = DefaultParallelRngT, class FloatT = double> void parallel_rng::ParallelRngManager< RngT, FloatT >::seed (SeedT *seed*)

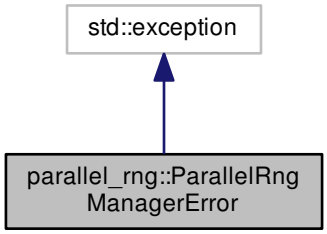
The documentation for this class was generated from the following file:

- ParallelRngManager.h

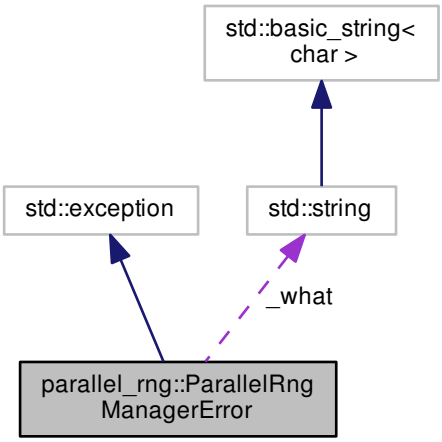
7.2 parallel_rng::ParallelRngManagerError Class Reference

`#include </home/travis/build/markjolah/ParallelRngManager/include/ParallelRng-Manager/ParallelRngManager.h>`

Inheritance diagram for parallel_rng::ParallelRngManagerError:



Collaboration diagram for parallel_rng::ParallelRngManagerError:



Public Member Functions

- ParallelRngManagerError (std::string what)
- const char ∗ what () const noexceptoverride

Protected Attributes

- std::string _what

7.2.1 Detailed Description

Definition at line 60 of file ParallelRngManager.h.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 parallel_rng::ParallelRngManagerError::ParallelRngManagerError (std::string *what*) [inline]

Definition at line 65 of file ParallelRngManager.h.

7.2.3 Member Function Documentation

7.2.3.1 const char* parallel_rng::ParallelRngManagerError::what ( ) const [inline], [override], [noexcept]

Definition at line 66 of file ParallelRngManager.h.

References _what.

7.2.4 Member Data Documentation

7.2.4.1 std::string parallel_rng::ParallelRngManagerError::_what [protected]

Definition at line 63 of file ParallelRngManager.h.

Referenced by what().

The documentation for this class was generated from the following file:

- ParallelRngManager.h

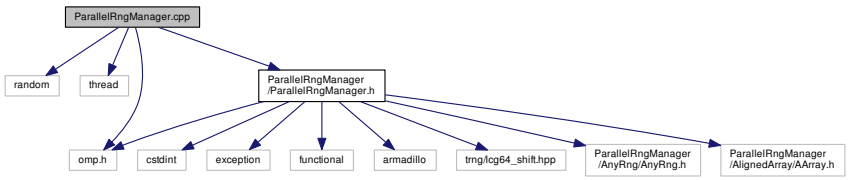
8 File Documentation

8.1 ParallelRngManager.cpp File Reference

Fast auto rng for parallel openmp code.

```
#include <random>
#include <thread>
#include "omp.h"
#include "ParallelRngManager/ParallelRngManager.h"
```

Include dependency graph for ParallelRngManager.cpp:



Namespaces

- parallel_rng

Functions

- SeedT parallel_rng::generate_seed ()
- IdxT parallel_rng::openmp_estimate_max_threads ()

  Use openmp to estimate the maximum number of threads that will be generated.

8.1.1 Detailed Description

Fast auto rng for parallel openmp code.

Author

Mark J. Olah (mjo@cs.unm DOT edu)

Date

2016-2017

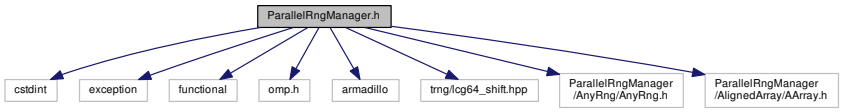
Definition in file ParallelRngManager.cpp.

8.2 ParallelRngManager.h File Reference

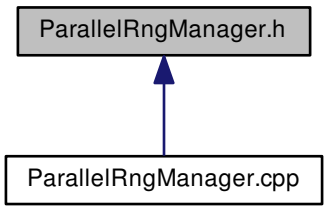
Adapts TRNG parallel RNG to armadillo, maintaining a per-thread RNG.

```
#include <cstdint>
#include <exception>
#include <functional>
#include <omp.h>
#include <armadillo>
#include <trng/lcg64_shift.hpp>
#include "ParallelRngManager/AnyRng/AnyRng.h"
#include "ParallelRngManager/AlignedArray/AArray.h"
```

Include dependency graph for ParallelRngManager.h:



This graph shows which files directly or indirectly include this file:



Classes

- class parallel_rng::ParallelRngManagerError
- class parallel_rng::ParallelRngManager< RngT, FloatT >

Namespaces

- parallel_rng

Macros

- #define DEBUG_ASSERT(...)
- #define ASSERT_SETUP(...)

Typedefs

- using parallel_rng::DefaultParallelRngT = trng::lcg64_shift

  Suggested default ParallelRNG type.
- using parallel_rng::SeedT = uint64_t

  Use the true random interface to generate a truly random seed.
- using parallel_rng::IdxT = arma::uword

Functions

- SeedT parallel_rng::generate_seed ()
- IdxT parallel_rng::openmp_estimate_max_threads ()

  Use openmp to estimate the maximum number of threads that will be generated.
- template<class RngT = DefaultParallelRngT, class FloatT = double>
  ParallelRngManager< RngT, FloatT > parallel_rng::make_parallel_rng_manager ()
- template<class RngT = DefaultParallelRngT, class FloatT = double>
  ParallelRngManager< RngT, FloatT > parallel_rng::make_parallel_rng_manager (SeedT seed)

Variables

- parallel_rng::rngs {num_threads,cache_alignment, RngT{seeder}}
- parallel_rng::norm_dist {num_threads,cache_alignment, NormalDistT{}}

8.2.1 Detailed Description

Adapts TRNG parallel RNG to armadillo, maintaining a per-thread RNG.

Author

Mark J. Olah (mjo@cs.unm DOT edu)

Date

2016-2017

Definition in file ParallelRngManager.h.

8.2.2 Macro Definition Documentation

8.2.2.1 #define ASSERT_SETUP(...)

Definition at line 45 of file ParallelRngManager.h.

8.2.2.2 #define DEBUG_ASSERT(...)

Definition at line 40 of file ParallelRngManager.h.

8.3 README.md File Reference

Index