Package ‘rscimark’

March 17, 2016

Title  SciMark 2.0 Benchmark for Scientific and Numerical Computing

Description  The SciMark 2.0 benchmark was originally developed in Java as a benchmark for numerical and scientific computational performance. It measures the performance of several computational kernels which are frequently occurring in scientific applications. This package is a simple wrapper around the ANSI C implementation of the benchmark.

Maintainer  Jakob Bossek <j.bossek@gmail.com>

URL  https://github.com/berndbischl/rscimark

BugReports  https://github.com/berndbischl/rscimark/issues

License  BSD_2_clause + file LICENSE

Encoding  UTF-8

Imports  checkmate

Suggests  testthat

LazyData  yes

ByteCompile  yes

Version  1.0

RoxygenNote  5.0.1

NeedsCompilation  yes

Author  Bernd Bischl [aut],
        Jakob Bossek [aut, cre]

Repository  CRAN

Date/Publication  2016-03-17 13:18:56

R topics documented:

  rscimark ................................................................. 2

Index  3
**Description**

This function is a simple wrapper around the ANSI C version of the SciMark 2.0 benchmark which is a benchmark for numerical and scientific computing. Concisely performance measurements for the computational kernels *Fast Fourier Transformation* (FFT), *Gauss-Seidel relaxation*, *Sparse matrix-multiply*, *Monte Carlo integration* and *dense LU factorization* are computed.

In order to isolate effects of memory hierarchy the problem sizes, e.g., the size of the matrix for the dense LU matrix factorization, are pretty small. However, addressing the performance of the memory subsystem is possible by setting the `large` argument to TRUE.

**Usage**

```r
rscimark(large = FALSE, minimum.time = R)
```

**Arguments**

- `large` [logical(1)] Run large version of benchmark? Default is FALSE.
- `minimum.time` [numeric(1)] Minimum time to run each of the benchmarks, in seconds. Default is 2.

**Value**

numeric Named vector of time measurements with the following components:

- **Composite** Mean value of the remaining components.
- **FFT** Performance of the Fast Fourier Transformation (FFT).
- **SOR** Performance of the Jacobi Successive Over-relaxation (SOR).
- **MC** Performance of a Monte Carlo integration.
- **SMM** Performance of a sparse matrix multiplication.
- **LU** Performance of a dense LU matrix factorization.
Index

rscimark, 2