Contribution ID: **98** Type: **Parallel talk**

# Efficient Pseudo-Random Number Generation for Monte-Carlo Simulations Using Graphic Processors

*Thursday, 8 September 2011 17:00 (25 minutes)*

The future of high power computing is evolving towards the efficient use of highly parallel computing environment. The class of devices that has been designed having parallelism features in mind is the Graphics Processing Units (GPU) which are highly parallel, multithreaded computing devices. One application where the use of massive parallelism comes instinctively is Monte-Carlo simulations where a large number of independent events have to be simulated. At the core of the Monte-Carlo simulation lies the random number generators. For GPU programming, the random number generator should have (a) good statistical properties (b) high computational speed (c) low memory use, and (d) a large period . The most commonly used Mersenne Twister generator has very good statistical properties with a long period of 2ˆ(19937)-1 , but not suitable for implementation in the GPU as it has a large state that must be updated serially. Each GPU thread must have an individual state in global RAM and requires multiple access per generator. The relatively large number of computation per generated number makes the generator too slow for GPU programming except in cases where the ultimate in quality is needed. In this paper, we have used a hybrid approach as used in NVIDIA CUDA library. The suggestion is to use a combination of three Tausworthe generator with different parameters along with a simple Linear Congruential Generator (LCG) where the mod operation is not performed explicitly. The period of these combinations is quite high (2ˆ121) and has good statistical properties as the defects of one generator gets compensated by other. This hybrid generator requires four random seeds which can be supplied using a CPU-side random number generator. We have carried out alias Monte-Carlo sampling using this hybrid generator where each GPU thread is used to generate random variable in parallel. This would mean each thread needs to be provided a random seed independently. In the present work, we have implemented alias sampling with NVIDIA GeForce GTX 480 GPU card using both CUDA and OpenCL kernels. It is noticed that the kernel execution in both cases is about 1000 times faster as compared to the CPU whereas the total code execution is only 10 times faster. This is due to the fact that memory copy from host to device or vice-versa is very slow. Therefore, we try to minimise memory access time and implement a simple scheme to generate random seed per thread on the fly from the formulae seed=1099087573*id where id is the thread index. This is known as quick and dirty LCG which has a period of 232 and mod operation is not explicitly needed due to overflow of unsigned integer. It is shown that this hybrid generator which takes seed on the fly is quite fast, reproduces the statistical properties reasonably well and can easily be implemented on each thread of GPU as well as CPU in an efficient way.

**Primary author:** Mr MOHANTY, Siddhant Ajit (CERN)

**Co-authors:** Dr MOHANTY, Ajit Kumar (CERN); Dr CARMINATI, Federico (CERN)

**Presenter:** Dr CARMINATI, Federico (CERN)

**Session Classification:** Thursday 08th - Computing Technology for Physics Research

**Track Classification:** Track 1: Computing Technology for Physics Research