Random Permutation on a GPU – Is Your Algorithm Unbiased for $n \neq 2^m$?

Michael Waechter, Kay Hamacher, Franziska Hoffgaard, Sven Widmer, Michael Goesele
TU Darmstadt
Darmstadt, Germany

1. Introduction

Parallel random permutation algorithms are an essential building block and can, e.g., be used in statistical science and modeling or in bioinformatical phylogenetic reconstruction. Many of these algorithms can be classified into one of five categories: Rand_Sort, Rand_Dart, Rand_Shuffle, Rand_Dist and permutation networks [Cong and Bader 2005]. Permutation networks meet all requirements for efficient implementation on a GPU (e.g., scalability with number of processors, working in-place, and small amount of synchronizations). Many of these algorithms operate on input sizes that are a power of two and it is often assumed that these algorithms can be easily generalized to arbitrary n. We show that this simplifying assumption is not necessarily correct since it may result in a biased algorithm (i.e., not all possible permutations are generated with equal likelihood). We also present an iterative approach to correct this bias, which can be applied to almost all permutation algorithms.

2. Bias

The bias of a permutation algorithm can be described using stochastic permutation matrices. For some permutation algorithms the resulting distribution may be non-uniform if applied to arbitrary n. We define a bias measure for a permutation matrix M_n:

$$B(M_n) = \frac{1}{n^2} \sum_{i,j} \frac{|M_n(i,j) - 1/n|}{1/n} = \frac{1}{n} \sum_{i,j} |M_n(i,j) - 1/n|$$

The bias $B(M_n)$ for a butterfly network using one iteration (k=1, green curve) is shown in Figure 1 for various input data sizes n. k gives the number of iterations of the permutation applied to the input array (see Section 2.1). We obtained a similar bias for the permutation algorithm described by Waksman [1968].

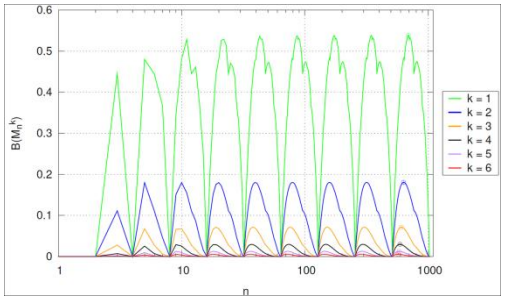

Figure 1. *Bias $B(M_n)$ for varying array size and iterations*

2.1 Bias Reduction

An iterative application with positive permutation matrices can reduce the bias for all permutation algorithms. In Waechter et. al [2011], we prove that the bias converges exponentially against 0 as illustrated in Figure 1 for k = 1 to 6 iterations. Furthermore the bias can be reduced for butterfly networks by cyclic shifting of the array content between the algorithm iterations. However, the shifting offsets need to be selected carefully, since otherwise shifting might increase the bias.

3. Results

We compare an implementation of the butterfly network on an NVIDIA GeForce GTX 480 with varying numbers of bias reduction iterations and different array sizes against a Rand_Sort implementation using Radix Sort from the CUDPP library.

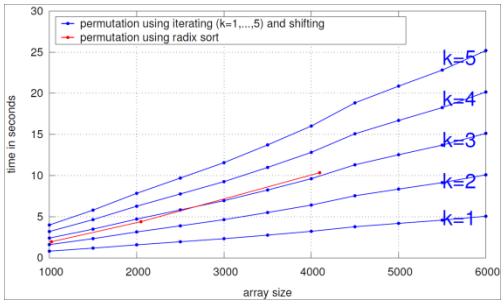

Figure 2. *Speed of randomly permuting arrays of various sizes.*

As shown in Figure 2, the runtime of the permutation network with up to three bias reduction iterations is competitive with the optimized CUDPP code. Rand_Sort stores, however, an additional key per element and is therefore not in-place.

4. Conclusions

Permutation networks are well suited for a GPU implementation. The bias introduced by shuffling a non power of two number of array elements is, however, an issue for many permutation networks. It can be reduced at an exponential rate by applying the permutation algorithm repeatedly. For butterfly networks, cyclic shifting with a correct offset yields a further improvement. In a practical scenario, the bias may be tolerable and can be traded off against computational speed by varying the number of iterations.

References

CONG, G., BADER, D.A. 2005. An empirical analysis of parallel random permutation algorithms on SMPs. In *Proc. 18th ISCA International Conference on Parallel and Distributed Computing Systems (PDCS 2005)*.

WAKSMAN, A. 1968. A permutation network. *Journal of the ACM 15(1)*.

WAECHTER, M., HAMACHER, K., HOFFGAARD, F., WIDMER, S., GOESELE, M. 2011. Is Your Permutation Algorithm Unbiased for $n \neq 2^m$?. In *Proc. 9th International Conference on Parallel Processing and Applied Mathematics (PPAM 2011)*, to appear. www.gris.tu-darmstadt.de/research/captreal/projects