E.ENG.27

GRAPHICS PROCESSING UNITS: TO USE OR NOT TO USE?

D. R. V. L. B. Thambawita, N. C. Ellepola, R. G. Ragel, D. Elkaduwe

Department of Computer Engineering,
Faculty of Engineering, University of Peradeniya

String matching is a very important aspect in various databases and text processing applications. Bioinformatics, signature based anti-virus software and many other important applications highly depend on the efficiency of string matching tools. With the advent of parallel computing, traditional sequential string matching drawbacks were phased out improving the application's performance. Over the past few years, the use of Graphic Processing Units (GPUs) to achieve parallelism has shown promising results, in which the GPUs exhibit SPMD (Single Program Multiple Data) programming model. NVIDIA has introduced CUDA (Compute Unified Device Architecture) programming API enabling programmers to use threaded processors of a GPU to achieve higher data parallelism.

In our research, we consider a basic string matching algorithm as a benchmark for comparing CPU and low end GPU performance for single string matching. Here, we consider changing memory types (global memory, constant memory, shared memory), data file size and the number of threads in both CPU and GPU and analyse them in order to compare their performance trade-offs. We utilize the maximum work load on both GPU and CPU when comparing the string by repeating the same pattern in the data file.

In our approach, we observe the performance while altering the data file size, and experiments indicate that, when we only consider the kernel execution time, with the increment of the data file size, the rate of increment of the time taken to match the strings decreased in GPU in contrast to the CPU. However, in most GPU kernels data must be moved on to the device prior to being used by the kernel, which introduces an additional time for the computation. In our experiment, this causes performance deterioration with the increment of the data load to the device due to context initialization time.

In this context, when the GPU load is low, low end GPUs show ill performance in basic string matching operations compared to that of the CPU due to the process initialization of the GPU. The performance of the GPU is gradually increased with the input data file size. As the next phase of the project, we are planning to conduct the experiment using different data types and different GPUs with higher bandwidth capabilities to minimize the effect of data transfer overhead.