Accelerating Bit-Parallel Approximate Matching On GPU Platforms For Small Patterns

Keh Kok Yong
Accelerative Technology Lab
MIMOS Berhad
Kuala Lumpur, Malaysia
kk.yong@mimos.my

Hong Hoe Ong
Accelerative Technology Lab
MIMOS Berhad
Kuala Lumpur, Malaysia
hh.ong@mimos.my

***Abstract* — Approximate pattern matching (APM) is commonly used in bioinformatics, network security and information retrieval to find a pattern from a large database. Unlike exact pattern matching, APM allows limited number of errors in search process. Bit-parallel APM represents the internal states as bit value, then utilizes the bit-level parallelism to speed up the pattern matching process. One of the notable examples is Wu-Manber algorithm that enjoy great speed performance in parallel hardware accelerator like GPU and Xeon Phi. In this paper, we extend the Wu-Manber implementation in GPU to achieve faster pattern matching speed for small patterns on various state of the art GPU platforms. The experimental results show that our proposed technique can improve the pattern matching speed by 3.32% - 17.15% for short patterns on K40c, GTX980 and GTX1080 respectively.**

Keywords— Approximate pattern matching, Wu-Manber algorithm, GPU, bit-parallel.

I. INTRODUCTION

Pattern matching identifies the location in a large string database at which one or more substring patterns appear. It is very useful for applications that need to monitor large volume of real-time data to identify malicious attack (network intrusion detection systems) [1,2]. Besides that, pattern matching is also used to recognize the human handwritten signature [3]. Bioinformatics is another important field that rely heavily on pattern matching locate specific amino acid sequences in biological databases [4.5], which can be very time consuming. Given performance requirements of such applications, implementation pattern matching algorithm needs to be parallel and efficient. Hence, hardware accelerator is becoming a popular choice for accelerating pattern matching algorithms.

Scaling up the throughput of pattern matching by utilizing multiple processing elements (e.g. multi-core CPU, many-core GPU and FPGA) is non-trivial, as the pattern matching algorithms can be regarded as an irregular problem that suffers from issues like irregular control flow and inconsistent data access. To overcome this, bit-parallel algorithms are developed [6] to enhance the parallelism of pattern matching algorithms, which is important for good performance in single instruction, multiple data (SIMD) parallel machine (e.g. GPU, AVX instructions and Xeon Phi). Although there are a number of successful implementation works [7,8,9] that utilizes hardware accelerators to improve the pattern matchinig speed, they are often targeting medium to long search patterns. To the best of our knowledge, not many existing works aim at improving the pattern matchinig speed for short patterns [10], especially for GPU platforms. This motivates us to investigate efficient implementation of pattern matching algorithm in latest GPU platforms for short pattern matching.

Aho-Corasick [11] and Wu-Manbar [12] are two popular algorithms for pattern matching. Wu-Manber algorithm allows approximate pattern matching, wherein the search results need not to be exactly same as the search pattern; some amount of errors are allowed. In this paper, we present an efficient implementation of Wu-Manber algorithm in various state-of-the-art GPU platforms that is suitable for short search pattern ($\leqslant 32$ characters), which improves on previous implementation in GPU platform.

II. BACKGROUND

This section presents the prior works related to pattern matching implementation in GPU, the Wu-Manber algorithm and some important information on the GPU architecture.

A. Related Work

Zha et al. [13] presented one of the pioneering work in efficient implementation of pattern matching algorithms in GPU platform. They optimized the implementation of Aho-Corasick and Boyer-Moore algorithms for GPU-GPU (input and output are in GPU memory) and CPU-CPU (input and output are in CPU memory) cases. They concluded that GPU implementation can be faster than single thread CPU implementation, but slower to multi-thread CPU version. Tumeo et al. [14] presented a thorough evaluation of software based Aho-Corasick implemnetaiton in various high-performance systems (Niagara 2, x86 multiprocessors, Cray XMT and InfiniBand cluster of x86 multicores with NVIDIA Tesla C1060 GPUs). Later on, Hsieh et al. [15] proposed a Deep Packet Inspection (DPI) engine accelerated by K20 GPU platform using variant of Aho-Corasick algorithm. The techniques proposed in this work can reduce the total required memory, remove thread divergence and optimize the memory access patterns in GPU.

Tran et al. [16] presented some techniques for implementing Wu-Manber algorithm in Kepler GPU and Xeon Phi. The key innovation in this paper is the successful mapping of the internal states of Wu-Manber algorithm to the machine word size (512-bit for Xeon Phi and 1024-bit for GPU). However, this work does not fully optimize the performance of GPU implementation, especially for small search patterns ($p \leq 32$). In particular, the GPU kernel execution needs to wait for all pattern matching process to complete before copying the result back to CPU. For small patterns, it is easy to find many matches, part of these results can be copied back to CPU before other matches are found. This allows the CPU to start consuming the partial results instead of waiting for all search process in GPU to complete, thus effectively overlapping the CPU and GPU operations to achieve higher performance. In this paper, we aim to improve this prior work by proposing the technique to overlap the CPU and GPU operations, which can be improve the pattern matching performance for small patterns.

978-1-5386-7167-2/18/$31.00 ©2018 IEEE