Optimization and Implementation Performance of Sequence Alignment on the Intel Xeon Phi-based Heterogeneous System

Shaolong Chen*, Zhenzhen Luo, Wei Yang and Wenle Wang

School of software, Jiangxi Normal University, Nanchang JX, China

*Corresponding author email: schen@jxnu.edu.cn

Abstract. The heterogeneous system based on different architectures becomes a convenient solution in the high performance computing research when facing the expanding sequence data in bioinformatics analysis. Intel Xeon Phi-based cluster is one of the most utilized heterogeneous systems in recent years. Without accurate results from sequence alignment, the remaining two steps in the variant analysis, variant calling and variant annotation, cannot achieve the correct consequence. However, most sequence aligners are developed facing the multicore system and cannot take advantage of Intel Xeon Phi-based cluster. This paper explored the implementation modes on the Intel Xeon Phi-based heterogeneous system, including native, offload and symmetric modes. We indicate that native mode cannot take advantage of Intel Xeon Phi-based cluster through the evaluation of scalability of various modes under sequence alignment. Although offload mode owns a promising future, it is not easy to enhance performance without comprehensive coding ability. Finally, the symmetric mode could provide a low complexity solution that supports significant improvements in performance.

Keywords: Sequence alignment; Intel Xeon Phi; Heterogeneous system; Manycore; Multicore.

1. Introduction

Genome variant analysis in bioinformatics consists of three essential workflows, sequence alignment, variant identification and variant annotation[1]. Sequence alignment provides fundamental consequences for the other workflows and requests the most time-consuming and computing-cost of them[2]. The sequence data is growing up rather faster than we could handle with the current infrastructures in high-performance computing(HPC) with the next-generation sequence technique’s appearance. Variant analysis cannot come to a quick response without the efficient calculation in the sequence alignment. Thus, higher efficient utilization of modern hardware by sequence alignment application is indispensable.

The heterogeneous system consists of multicore and manycore architecture become the mainstream computing configuration in HPC since it is convenient to plug in more hosts to scale out. NUMA(Non-Uniform Memory Access) and UMA(Uniform Memory Access) are two of the most prevailing system architectures in multicore and manycore hosts. The algorithms of sequence alignment are mostly designed in the multicore, and usually implemented in the manycore. However, these two architectures are still different in some crucial computing circuit, which leads to an abnormal performance behavior when we implement multicore applications on manycore hosts.

The contributions in this paper we explore performance limitation on the Intel Xeon Phi-based heterogeneous cluster. We first explain the execution mode on this particular architecture, native mode, offload mode and symmetric mode. Furthermore, we exploit sequence aligners in terms of scalability of the threads among these three modes. Without a comprehensive coding ability, it is not convenient to efficiently utilize both systems under offload mode. The symmetric mode gives us a promising
implementation mode without complex coding, but it needs a complicated configuration beforehand based on the native mode execution. Thus, the symmetric mode could provide a low complexity solution that supports significant performance improvements based on Intel Xeon Phi’s heterogeneous cluster. The rest of the paper is organized as follows. Section 2 we evaluate a literature review on the related work about sequence alignment. Afterward, background and motivation are provided in section 3. Section 4 gives the experiment environment and section 5 evaluates the results of the experiments. Section 6 concludes our works and future research are discussed.

2. Literature Review on the Related Work
First, we pay attention to the related research on the performance improvement of sequence alignment in the heterogeneous system that concludes two architectures, multicore and manycore. Afterward, we give a conclusion according to the discussion we argue below.

More and more sequence alignment applications were developed by the researchers to resolve the booming growing data in biological sequence. The majority of sequence aligners belong to two kinds of index techniques, hash table and FM index. BWA[3], Bowtie[4], CUSHAW[5] and Blast[6] etc. These applications only concentrate on one particular architecture, either on multicore or on GPU. They are convenient for multicore architecture but complicated for transmitting to the manycore, particular on the Intel Xeon Phi manycore architecture. Although some related works[7-9] got a considerable improvement on heterogeneous system performance, they also display some disadvantages we cannot ignore today.

Computing system architecture and sequence aligners are the two crucial elements we should compromise between them to achieve their best coporation in performance. Some existing works[10-12] reported that memory allocation and data partition significantly impact the ultima performance in such architectures. Macedo et al.[13] tried to exploit a parallel pipeline strategy by using multi-threading and multi-instance to promote sequence alignment procedure on the heterogeneous clusters. Jing Nagarajan et al.[14] introduced some skills on parallelization for the multicore system. However, these strategies lack a comprehensive concern that they are not compatible with the manycore architecture since they differ in many fields in the system.

In terms of performance improvement of sequence aligner, Olivier et al.[15] implemented OpenMP and Qthread skills on a multicore system. Paper[16] applied Intel Cilk and Pthread techniques on a manycore system. Chen et al.[2] concluded Pthread, OpenMP and Intel Cilk with FM algorithm in the heterogeneous structure. These propositions focus on thread parallelization of sequence alignment that cannot reach a significant amount of achievement in the final performance.

In term of performance improvement of computing system architecture, many studies[17-18] indicated that the memory-sensitive procedure exists in sequence alignment, and attempted some data structure optimizations to ameliorate system cache usage. These optimizations could nevertheless bring an improvement of performance in sequence aligners, they cannot be easily implemented in the manycore since most of them are specially planned for the multicore. Chen et al.[19] and mbwa[7] proposed an FPGA-base and Xeon Phi-based offload mode for the heterogeneous execution. Houtgast et al.[20-21] adopted GPU-based and FPGA-based native mode for the manycore implementation. Tens of studies [22-23] were proposed on FPGA and GPU manycore architectures while another popular manycore-Intel Xeon Phi, was neglected with few reviews about alignment improvement.

3. Background and Motivation

3.1. Sequence Alignment and Sequence Aligner
Sequence alignment involves mapping between the query and the reference to illustrate their difference and similarity based on three different results, match, mismatch and gap, as shown in figure 1. According to the number of sequences applying in the procedure, multiple sequence alignment(MSA) and pairwise sequence alignment(PSA) constitute two main groups in the sequence alignment. MSA focuses on mapping among more than two sequence data while PSA only concentrates on two, short read and genome reference. According to the mapping location of sequences, sequence alignment could be divided into two fundamental classes, global and local alignment(GA and LA). GA acquires for the most
optimal mapping according to the whole length between two sequences. LA obtains the most optimal mapping in the sub-sequences of two sequences. Needleman-Wunsch (NW) and Smith-Waterman (SW) are respectively the outstanding algorithms in global alignment and local alignment.

**Figure 1.** Three results in the sequence alignment.

Many sequence aligners depended on various algorithms that were proposed to deal with this situation in recent years. FM index, hash table and merge sorting constitute the three principal groups in the sequence alignment. The former two groups illustrate a smaller memory consumption than the algorithms in merge sorting. Thus, FM-based and hash table-based aligners, such as BWA, MAP, SOAP and Bowtie, are widely utilized in bioinformatics nowadays. We mainly work on BWA which uses FM index technique requiring a small amount of memory to align short read efficiently against a large genome reference. Backtrack, SW and MEM constitute the three kinds of algorithms in BWA. Algorithm Backtrack and MEM are the most usages in the bioinformatics HPC arena. Many applications are developed based on these two algorithms, mbwa[7] and pBWA[9] for instance.

### 3.2. Heterogeneous Architecture and Its Execution Modes

Mainstream computing infrastructure nowadays consists of multicore and manycore architectures. Multicore system represents the processor which could control whole things as a master node. The most widespread CPU-memory structure is NUMA architecture. Manycore system stands for the coprocessor which is quite convenient for scale-out with low cost, for example Intel Xeon Phi, GPU and FPGA. In this work, we keep an eye on the Intel Xeon and Intel Xeon Phi that constitute the heterogeneous cluster as figure 2 shows.

**Figure 2.** Heterogeneous system of processor and coprocessor.

**Figure 3.** Three basic execution modes.
Intel Xeon and Intel Xeon Phi have an identical core architecture based on x86 which enables an application developed in multicore that could be carried out in manycore with no difficulty. However, for the purpose of fitting the hardware, an application still should be selectively offload – choose highly parallel computing to the coprocessor – the offload mode is rather particular which enables corporations in this heterogeneous cluster. Thus, three execution modes in this heterogeneous cluster are based on different configurations, as figure 3 illustrates.

Native mode: the program is implemented on the processor or coprocessor independently. Symmetric mode: the program is meantime executed on processor and coprocessor. Offload mode: the program is implemented on the processor and selects highly parallel sections transform to the coprocessor in the meantime.

4. Hardware Environment and Source Data
All experiments were carried out at a minimal 6 execution of each case under identical configurations for acquiring mean values with high confidence spacing. To evaluate and compare their performance, we used two datasets, three hosts and three sequence aligners, as shown in tables 1, 2 and 3.

mBWA (offload mode) is initiated in Intel Xeon and automatically offload some calculation loops to Intel Xeon Phi. sBWA is initiated in Intel Xeon and Intel Xeon Phi in the meantime. The workload ratio between both architectures in sBWA is calculated based on the native mode case to approach a similar time-cost on both architectures. For instance, the workload ratio would be set up 4:6 in sBWA if the ratio of time-consuming for accomplishing the identical job on the native mode is 6:4.

5. Experiment Results and Evaluation
In this section we would evaluate the scalability of thread on our heterogeneous cluster. Depending on the Intel Xeon Phi-based system we implement, offload and symmetric mode are denoted as combined mode since both architectures are utilized in every case.

5.1. Scalability of Native Mode
xBWA and pBWA present a declining tendency in the efficiency of threads when an increasing number of threads are applied, as shown in figures 4 and 5. For example, 6, 12 threads in xBWA and 60, 120 threads in pBWA. These cases do not get a half time-cost when the thread amount doubles. Even though more threads are applied, less time-cost would be consumed, it reaches a relatively small reduction on the execution time in the cases of 18, 24 threads in xBWA and 180, 240 threads in pBWA. The
scalability behaviour of sequence aligner implies that it does not own advantage on such architecture’s native mode. Additionally, when comparing xBWA with pBWA at each case in figures 4 and 5, although pBWA utilizes 10X number of threads than xBWA, it still takes more time than xBWA in all cases. A large amount of thread does not become a benefit in performance in the experiments. Obviously, on one side, ring architecture with one small memory bank leads the Intel Xeon Phi is rather sensitive with data I/O, and memory congestion is increasing when a large thread is applied. On the other side, most sequence aligners are developed facing the multicore system instead of the manycore, leading to the original aligners cannot efficiently utilize the Intel Xeon Phi. Another interesting phenomenon occurs in the case of 12 threads in xBWA and 120 threads in pBWA. When comparing these cases above to 6 threads in xBWA and 60 threads in pBWA respectively, xBWA achieves a more similar time-cost than that of pBWA between these two cases. This behavior can be explained with locality and congestion in memory between two NUMA nodes in the Intel Xeon platform. The threads are attached to the processors under a numeric order. When we apply sequence aligner with 6 threads, these threads are allocated to core id 0, 1, 2, 3, 4, 5 in one NUMA node. However, when we execute sequence aligners with 12 threads, these 12 threads are distributed to the cores in two NUMA nodes, leading to large remote access from another NUMA node.

![Figure 4. Scalability of xBWA](image1)

![Figure 5. Scalability of pBWA](image2)

5.2. Scalability of Offload and Symmetric Mode

An practical approach to utilize Intel Xeon Phi is offload mode and symmetric mode. As figure 6 shows, the case of 12+120 indicates there are 12 and 120 threads applying in the Intel Xeon and Intel Xeon Phi platform, respectively. mBWA, offload mode implementation, transmitting core computing to coprocessor Intel Xeon Phi, has considerably surpassed pBWA and achieve very comparable performance with xBWA.

![Figure 6. Scalability of two modes.](image3)

Additional, sBWA outperforms mBWA under the identical amount of the threads. sBWA illustrates a convenient method without a complicated coding ability to utilize a complex heterogeneous cluster.
However, the implementation of sBWA depends on the results from the native mode to establish the workload ratio between the two architectures. In order to implement sBWA, a comprehensive configuration beforehand is necessary for us. mBWA is automatically executed during the whole procedure when the program is initiated in the processor, but mBWA needs a comprehensive coding ability when you make a determination to select what loops pass to the coprocessors.

6. Conclusion and Future Work

We evaluate sequence alignment on one unique Intel Xeon Phi-based heterogeneous cluster that could support three kinds of implementation, native mode, offload mode and symmetric mode. Offload mode seems to be quite workable that could choose what calculation pass to the coprocessor. However, although 10 times of thread amount are applied in our experiments, mBWA achieves a rather similar speed-up with xBWA, as shown in table 4. Without a comprehensive coding ability, it is not convenient to efficiently utilize both systems under offload mode. sBWA gives us a promising implementation mode without complex coding, but it needs a complicated configuration beforehand based on the native mode execution. Thus, the symmetric mode could provide a low complexity solution that supports significant performance improvements based on Intel Xeon Phi’s heterogeneous cluster. The results of the experiments are highly convincing. The experiment was nevertheless carried out on human genome data, the methods could be utilized for all kinds. In order to achieve a precise consequence in the future, more sequence alignment algorithms are needed to explore.

| Name       | Time(s) | Speed(X) | Thread amount |
|------------|---------|----------|---------------|
| Native     | xBWA    | 2114     | 2.5X          | 24            |
| pBWA       | 3382    | 4X       | 240           |
| Offload    | mBWA    | 2427     | 3X            | 24+240        |
| Symmetric  | sBWA    | 1301     | 1.5X          | 24+240        |

Acknowledgments

This research is partially supported by the initial project (No.12020361) and youth talent project (No.12020778) of Jiangxi normal university, natural science foundation of Jiangxi province (No.20192BAB217007), and national nature science project (No.61967010).

References

[1] Lappalainen T, Scott AJ, Brandt M, Hall IM. Genomic analysis in the age of human genome sequencing. Cell 177.1 (2019): 70-84.
[2] Chen, Shaolong, and Miquel Angel Senar. Exploring efficient data parallelism for genome read mapping on multicore and manycore architectures. Parallel Computing, 87 (2019): 11-24.
[3] Li H and Durbin R. Fast and accurate long-read alignment with Burrows-Wheeler transform. Bioinformatics. 26(5):589-95, 2010.
[4] Langmead B. and Salzberg S. Fast gapped-read alignment with Bowtie 2. Nat Methods, 9:357-359, 2012.
[5] Yongchao Liu, Bertil Schmidt and Douglas L. Maskell. CUSHAW: a CUDA compatible short read aligner to large genomes based on the Burrows–Wheeler transform, Bioinformatics, 28(14):1830-1837, 2012.
[6] Stephen F Altschul, Thomas L Madden, Alejandro A Schäffer, Jinghui Zhang, Zheng Zhang, Webb Miller, and David J Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. Nucleic acids research, 25(17):3389–3402, 1997.
[7] Tsukasa Nakamura, Kazunori D Yamada, Kentaro Tomii, and Kazutaka Katoh. Parallelization of MAFFT for large-scale multiple sequence alignments. Bioinformatics, 34(14):2490–2492, 2018.
[8] René Rahn, Stefan Budach, Pascal Costanza, Marcel Ehrhardt, Jonny Hancox, and Knut Reinert. Generic accelerated sequence alignment in SeqAn using vectorization and multi-threading, Bioinformatics, 34(20):3437-3445, 2018.
[9] Jikai Zhang, Haidong Lan, Yuandong Chan, Yuan Shang, Bertil Schmidt, and Weiguo Liu, BGSA: a bit-parallel global sequence alignment toolkit for multi-core and many-core architectures, Bioinformatics, 35(13):2306-2308, 2019.

[10] Josefina Lenis and Miquel Angel Senar. A performance comparison of data and memory allocation strategies for sequence aligners on numa architectures. Cluster Computing, 20(3):1909-1924, 2017.

[11] Kazutaka Katoh and Hiroyuki Toh. Recent developments in the MAFFT multiple sequence alignment program, Briefings in Bioinformatics, 9(4):286-298, 2008.

[12] Gogol-Döring A and Chen W. An overview of the analysis of next generation sequencing data. Methods Mol Biol. 802:249-257, 2012.

[13] Emerson de Araujo Macedo and Azzedine Boukerche. Hybrid mpi/openmp strategy for biological multiple sequence alignment with dialign-tx in heterogeneous multicore clusters. In Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on, pages 418-425. IEEE, 2011.

[14] Nagarajan Kathiresan, Mohamed Ramzi Temanni and Rashid Al-Ali. Performance improvement of bwa mem algorithm using data-parallel with concurrent parallelization. In Parallel, Distributed and Grid Computing (PDGC), 2014 International Conference on, pages 406-411. IEEE, 2014.

[15] Stephen L Olivier, Allan K Porterfield, Kyle B Wheeler and Jan F Prins. Scheduling task parallelism on multi-socket multicore systems. In Proceedings of the 1st International Workshop on Runtime and Operating Systems for Supercomputers, pages 49-56. ACM, 2011.

[16] Charlotte Herzeel, Thomas J Ashby, Pascal Costanza and Wolfgang De Meuter. Resolving load balancing issues in bwa on NUMA multicore architectures. In International Conference on Parallel Processing and Applied Mathematics, pages 227-236. Springer, 2013.

[17] Beniamine David, Diener Matthias, Huard Guillaume and Navaux Philippe. TABARNAC: visualizing and resolving memory access issues on NUMA architectures. Visual Performance Analysis, 2015 Second Workshop on Austin, Texas, United States, 2015.

[18] Ben Langmead, Christopher Wilks, Valentin Antonescu and Rone Charles. Scaling read aligners to hundreds of threads on general-purpose processors, Bioinformatics, 35(3):421-432, 2019.

[19] Yu-Ting Chen, Jason Cong, Jie Lei, and Peng Wei. A novel high-throughput accelerator for read alignment. In Field-Programmable Custom Computing Machines (FCCM), 2015 IEEE 23rd Annual International Symposium on, pages 199-202. IEEE, 2015.

[20] Ernst Joachim Houtgast, Vlad-Mihai Sima, Koen Bertels and Zaid Al-Ars. An fpga-based systolic array to accelerate the bwa-mem genomic mapping algorithm. In Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS), 2015 International Conference on, pages 221-227. IEEE, 2015.

[21] Ernst Joachim Houtgast, Vlad-Mihai Sima, Koen Bertels and Zaid Al-Ars. Gpu-accelerated bwa-mem genomic mapping algorithm using adaptive load balancing. In International Conference on Architecture of Computing Systems, pages 130-142. Springer, 2016.

[22] Schatz M.C., Trapnell C. and Delcher A.L. High-throughput sequence alignment using Graphics Processing Units. BMC Bioinformatics, 8-474, 2007.

[23] Che-Lun Hung and Guan-Jie Hua. Local alignment tool based on Hadoop framework and GPU architecture", BioMed Research International, vol. 2014, 7 pages, 2014.