Processing-in-Memory for Genomics

1. Context & Objectives

Genomics is poised to be the largest data producing scientific field expected to surpass astronomy in the next few years. For example, the Illumina NovaSeq genome sequencing platform is capable of sequencing approximately 8.000 genomes (800 terabytes) per year per instrument, and the throughput of other systems from Pacific Biosciences and Oxford Nanopore Technologies (ONT) is constantly being improved. Additionally small, handheld sequencing platforms such as ONT MinION and Flongle make it possible to sequence bacterial and viral genomes in the field, thus facilitating disease outbreak analyses such as COVID-19 [1], Ebola [2], and Zika [3]. Furthermore, the use of genome sequencing in the clinic to guide diagnosis and treatment is now on the rise. These technological improvements lead to the estimation that millions of genomes are expected to be sequenced each year. However, extremely fast analysis of such large amounts of data remains a daunting challenge.

Genomics data is currently processed using data centers and cloud platforms, which may necessitate fast network connectivity for data upload, analysis, and downloading the results. The lag between data generation and obtaining of the results due to both network and computing infrastructure may pose problems in rapid diagnosis, for example, in identifying infectious agents and choosing the correct antibiotics for sepsis and hospital infections, and tracking outbreaks that spread quickly in the population as evidenced by recent outbreaks. The problem is exacerbated in developing countries and during crises where access to data centers is even more limited [4].

Enabling on-site analysis of genomic data, i.e., where data is produced, requires energy efficient, and yet high performance edge computing. All current computing platforms are designed following the von Neumann architecture principles, originated in the 1940s, that separate computing units (CPU) from memory and storage. Processing-in-memory (PIM) is expected to fundamentally change the way we design computers devoted to processing huge amounts of data. These technologies consist of processing capability tightly coupled with memory and storage devices. As opposed to bringing all data into a centralized processor, which is far away from the data storage and is bottlenecked by the latency (time to access), the bandwidth (data transfer throughput) to access this storage, and energy required to both transfer and process the data, in-memory computing technologies enable processing of the data directly where it resides, without requiring movement of the data, thereby greatly improving the performance and energy efficiency of processing of massive amounts of data potentially by orders of magnitude.

Fortunately, recent memory and die-stacking technologies have the potential to alleviate the major bottlenecks caused by CPU-centric designs [5]. First, new non-volatile memory technologies (e.g., PCM, ReRAM, STT-MRAM) provide significant advantages over conventional DRAM in terms of memory capacity, energy efficiency (e.g., non-volatility means that internal refresh is not needed), and compute capability (i.e., computing by directly using memory cells). Second, die-stacking enables computation inside or near memory, by placing compute units close to memory layers. For example, the Hybrid Memory Cube Consortium (HMC) [6] and High Bandwidth Memory (HBM) [7] are industry's initiatives to build DRAM memories composed of multiple layers in a 3D design, providing substantially higher bandwidth and lower latency. HBM is already available in some recent GPU and FPGA systems. Third, some DRAM designs are being augmented with computation capability today [8].

Processing-in-Memory approaches can be broadly classified into two types [9]: 1) processing-near-memory, where processing elements are placed near memory arrays (e.g., inside the DRAM chip, or in logic layers of 3D-stacked memories); and 2) processing-using-memory, where basic computation is performed by exploiting the existing architecture of the memory arrays and the operational principles of the memory cells to enable bulk processing operations within the main memory with minimal changes. A combination of the two types is certainly possible and likely very useful.

The GenoPIM project targets the first approach. In the past, many research projects have investigated the possibilities to bring data closer to computation in that way, such as presented in [9]. The solution of the UPMEM company, partner of the GenoPIM project, goes one step further by the adoption of innovations such as DRAM Processing Unit (DPU), a processor integrated directly in the memory chip, on the DRAM die. An UPMEM PIM server counts no less than 2560 DPUs for 160 GB of PIM memory and 256 GB of legacy memory. UPMEM PIM unlocks unprecedented bandwidth and parallelism to architects and developers while halting off-chip data movement. With a patented design that preserves DRAM manufacturing processes, UPMEM PIM can be implemented on any type of platform.

Since many bioinformatics algorithms focus on quickly processing massive amounts of data [10], these in-memory computing technologies can be greatly beneficial for improving performance and energy efficiency. The targeted breakthrough of GenoPIM is to invent and leverage in-memory computing architectures to fundamentally improve the performance and energy efficiency of various important genomic and bioinformatics algorithms such as sequence alignment, genome assembly, pangenome analysis, metagenome analysis, variation calling, etc. Our research will also need to include optimization and modification of several data structures for PIM architectures widely used in the bioinformatics field such as hash tables, de Bruijn and string graphs, and Bloom filters. These algorithms are bound by memory latency, bandwidth, and capacity, and performing them in memory can greatly accelerate them.

To summarize, the GenoPIM project aims to make **fast** (>20-50x faster than currently available)**, energy-efficient** (consuming >70% less energy than current platforms), and **cost-efficient** (>10-20x cheaper than standard servers) **genome analysis** a reality.

2. Methodology

The strategy proposed to achieve the goals of the GenoPIM project is based on the completion of 3 work packages (WPs), in 36 months. The WPs are the following:

- WP1: Genomic PIM data structures
- WP2: GenoPIM library
- WP3: PIM performance evaluation

WP1 aims to revisit the main genomic data structures for PIM architectures. WP2 will implement PIM data structures proposed by WP3, together with basic routines to access them, into a PIM genomic library. WP4 will evaluate PIM implementation on available UPMEM systems.

2.1. WP1: Genomic PIM data structures

Objectives

A key challenge to reach PIM promises is probably to have well defined data structures that match the architectural concept of processing-in-memory, together with optimized algorithms that are able to exploit these specific data structures. We will thus revisit the main data structures used in genomics applications and propose their PIM counterparts. The data structures that are currently targeted are de Bruijn (dBG) and string graphs, Bloom filters (BF), Counting Quotient Filters (CQF), minimal perfect hashing (MPHF), Burrows-Wheeler Transformation (BWT), and FM-index.

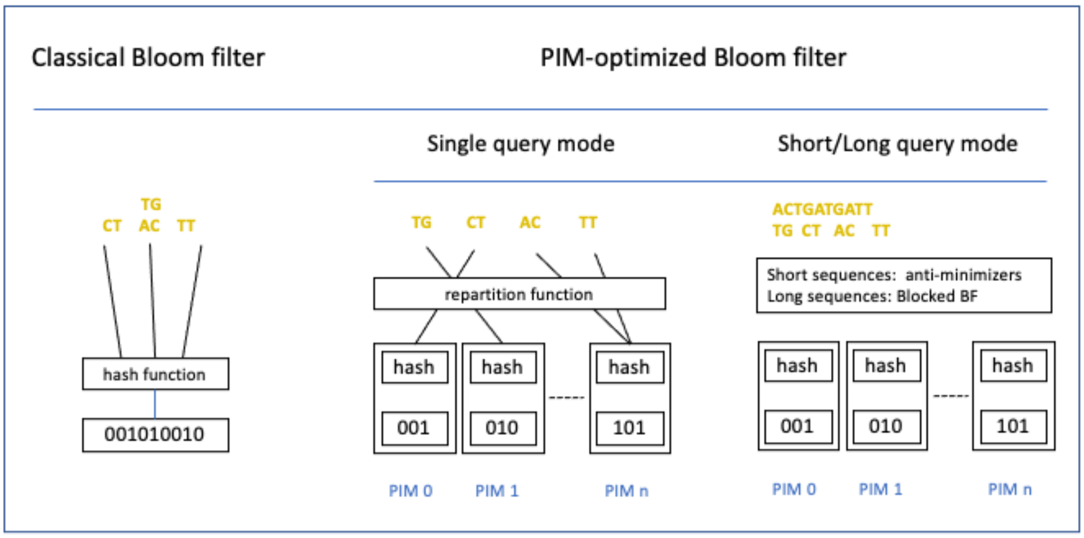
Working program

We will design the necessary architecture and pseudo-code to adapt the classical data structures to PIM. We start here by giving a concrete example on how this process might look like, using Bloom filters as a case study. A Bloom filter is a probabilistic data structure that consists of a bit vector and a hash function to store a set of elements, and only supports queries of the type “is this particular element in the set?”. A PIM Bloom filter will introduce a partitioning layer to divide the original Bloom filter in smaller chunks that can each reside next to their own PIM processor, and thus benefit from parallel queries. The key difficulty is in the design of an adequate partitioning strategy, given the particularities of genomics queries:

1. Single queries. In this regime, no speed-up is expected, as the query will be redirected to a single PIM processor. We will propose a naive partitioning of the Bloom filter so that each part resides next to a PIM processor.
2. Long queries. In a long sequence (e.g. longer than the number of PIM processors), queries to a Bloom filter will consist of consecutive fragments of the sequence being hashed and individually queried. By the pigeonhole principle, many (if not all) PIM processors will be responsible for handling several queries. Therefore we will seek to maximize query locality using the classical concepts of blocked Bloom filters [15] and minimizers [16]. A minimizer is the lexicographically smaller string of fixed length within a longer string. Our scheme will

ensure that nearby sequences will be queried not only within the same PIM processor but also often in the same memory page.

3. Short queries. In some applications e.g. read correction [17], short sequences are queried, whose lengths are shorter than the number of PIM processors. For those cases we will aim to distribute the query load as evenly as possible across PIM processors to maximize query speed, avoiding bottlenecks due to a single PIM handling several queries. We will introduce a novel algorithmic concept that we call “anti-minimizers”, ensuring that consecutive queries are not mapped to identical PIM processors. An anti-minimizer is a particular hash function that maps nearby fragments to as different locations as possible. For example, hashing the integer representation of a minimizer summed with its position on a sequence yields a candidate anti-minimizer.



We envision that the PIM Bloom filter will perform queries *t times faster than its classical counterpart, as long as more than t elements are queried in batch.*

The above description focused on Bloom filters to illustrate the planned work, yet in the WP we will carry an equivalent design process for several other data structures. Our priority order and motivation for this order is:

1. Bloom filter
2. Minimal perfect hashing (similarities of one specific work with Bloom filters [18])
3. Succinct bit arrays (similarities with Bloom filters)
4. Burrows-Wheeler transform (importance of the structure)
5. FM-index (similarities with Burrows-Wheeler transform)
6. de Bruijn graphs (similarities with FM-index)
7. Counting Quotient Filter (similarities with Bloom filters & perfect hashing)

We plan to use minimizers/anti-minimizers to partition BF & CQF, fixed-length prefixes to partition BWT, FM-index, and attempt both strategies for dBG.

2.2. WP2: PIM Genomic library

Objectives

Bioinformatics algorithms generally use specialized and high performance library routines that are combined together to reach a specific goal, or answer a precise biological question. Developing a programmer-friendly application programming interface (API) will allow other researchers to easily adopt PIM technologies. Furthermore, to make the adoption of our research easier, our programming PIM libraries will be similar to those that are developed for traditional computing platforms, such as SeqAn and GATB. Examples of basic bioinformatics routines are global and local alignment, text-pattern search, approximate pattern search, graph traversal, kmer analysis, etc.

Working program

The GenoPIM library will be developed progressively. It will include first basic routines that do not necessitate complex data structures and that can be deployed independently of WP 2. Then, depending on the progress of work package 2, more elaborate routines will be added. We propose to elaborate the GenoPIM library in two steps :

1. GenoPIM 0.5: basic routines based on partner preliminary works
2. GenoPIM 1.0: advanced routines based on WP 2

CNRS-IRISA and UPMEM partners have been working together for several years to evaluate the benefit of UPMEM PIM solution on several bioinformatics algorithms [11]. Until 2019, performance evaluations were carried out on a cycle-accurate simulator with very promising speed-up, especially on sequence comparison algorithms (protein alignment, DNA mapping) The executions on real hardware in 2020 have confirmed the good performances : speed-up of 20 to 30 were achieved on a 160 GB UPMEM systems. These experiments lead to develop several prototype software from which basic modules can be extracted and promoted as building blocks of the GenoPIM library.

We do not want the PIM library to be attached to a specific technology. Consequently, each module will have different implementations that will target different architectures. In the framework of this project, at least two implementations will be proposed : (1) Generic implementation; (2) DPU implementation

Generic implementation: it will run the routines on standard multi-processors. It will allow the designers to develop and experiment (in a functional way) applications for PIM architectures without real PIM hardware. Past experiments have also shown that the best performances are often reached by cleverly sharing the PIM and processor loads. In other words, pushing systematically all the computation into the processing elements of the memory may not be optimal. Thus, having routines that can be run indifferently on PIM processing units or on the processor cores may help to design the best implementation.

DPU implementation: it refers to the UPMEM PIM model. A DPU is an autonomous processor attached to a specific block of RAM. It can run its own code and communicate through specific channels to the host processor. There are no direct connections between DPUs.

Development of GenoPIM v1.0

The first version of the GenoPIM library will mainly include modules based on genomic sequence comparison. Some of them have more or less already been experimented in previous works on PIM. The others should be easily developed based on our expertise in this domain (i.e. they do not necessitate complex associated data structures) :

1. ungap protein alignment
2. ungap short DNA read alignment
3. short DNA read alignment with small gaps
4. protein alignment with small gaps

This family of algorithms, which are generally very time consuming, are commonly used as building blocks in many bioinformatics or genomics applications such as read mapping, variant calling, search of genomic banks, etc.

Development of GenoPIM v2.0

The following version of GenoPIM will add more complex modules related to graph structures used in genomics algorithms. These modules are not yet defined and will depend on WP2 progress.

For both the task the following actions will be performed :

1. Develop, test and debug the afford mentioned modules for both generic and UPMEM implementations
2. Write associated documentation
3. Make GenoPIM v1.0 and v2.0 available on github

2.3. WP2: PIM performance evaluation

Objectives

An important part of the project will be devoted to experimentally validating the concepts and tools developed through the WP 2 and 3. Bioinformatics use cases requiring the processing of large amounts of genomic data will be implemented, using the PIM genomic library. Tests will be mainly conducted on UPMEM servers in order to evaluate several metrics such as computation speed up or electric consumption.

Working program

Select and implement use-cases

This task aims to select a set of use-cases in the genomic field to demonstrate the benefit of PIM on this area according to different criteria : sequencing technologies, speed-up, energy consumption, total cost of ownership (TCO), etc. The number of genomic applications on which PIM can be evaluated is wide :

- Sequence alignment: DNA or protein
- Mapping
- Genotyping
- Variant calling on SNPs or on variant structures.
- Genome assembly: contigers, gap-filling, scaffolding
- Genome comparison
- Phylogenetics
- Metagenomic analysis
- …

The strategy will be to select use-cases of increasing complexity, from which building blocks from the GenoPIM library will be progressively available. We expect to test at least 5 different use-cases which will be chosen among the various possibilities to demonstrate the efficiency of the PIM architectures.

An important point is that, at this stage, together with the genomic applications, genomic data sets will also be fixed.

The application codes will be deposited on github.

Run and measure use-case performances

The goal of this task is to evaluate the PIM implementation of the different use-cases following two main criteria:

1. Computation efficiency
2. Energy consumption

UPMEM systems will be used as PIM architecture. Today a reference platform available at UPMEM consists in 2*Xeon Silver 4215 with 256GB RAM and 160 GB of PIM memory with 2560 DPUs clocked at 450MHz. Servers with higher PIM DIMM density such as and the Intel Icelake with 3584 DPUs is in the process of qualification while DPUs clocked at 600MHz with near 50% of improvement in power efficiency are under development. By the end of the GenoPIM project (2024), several major progress in the technology should allow experimentations with cutting-edge systems. The road map includes to qualify the densest platforms on the market such as the Cooper Lake 4* Xeon Gold 6328H with 5120 DPUs but also to support different protocols such as DDR5 and the CXL technology. A superscalar version of the UPMEM DPU is scheduled for as early as 2021 and would allow 2 instructions per cycle, essentially doubling its computing capabilities.

The first criteria can be evaluated in terms of speed-up compared to standard multi-core servers with or without UPMEM DIMM chips. A fair approach is thus to compare, for the same application and with the same data sets, the execution time of software optimized for multi-cores (t_{REF}) and software optimized for PIM (t_{PIM}). The speed-up (S) is given by:

$$S = t_{REF} / t_{PIM}$$

The second criteria, to be evaluated, needs to measure the power consumption of each system: p_{REF} for the system with UPMEM chip and p_{PIM} for the system with UPMEM chip. The energetic gain (E) can thus be calculated as:

$$E = (p_{REF} \times t_{REF}) / (p_{PIM} \times t_{PIM})$$

PIM efficiency can also be compared with other hardware accelerators such as FPGA or GPU boards which have recently been developed for genomic purposes. As an example, the Illumina DRAGEN using proprietary software on 8 FPGA Xilinx UltraScale Plus 16 nm FPGA [13], or Nvidia Parabricks using BWA-GATK4 on 8 NVIDIA®Tesla®V100 GPUs [14]. Both of them target variant calling applications.

3. Consortium

The consortium is made of four partners :

1. CNRS-IRISA, GenScale team, Rennes, France
2. Institut Pasteur, Paris, France
3. UpMem Company, Grenoble, France
4. Bilkent university, Turkey

The partners have complementary expertise in bioinformatics, genomics, data structures, string processing algorithms, computer & memory architecture, and parallelism. Interaction between some partners already exists. GenScale and UPMEM pioneered research on evaluating genomic algorithms on PIM for several years [11,12]. GenScale and Pasteur have common work on genomic data structures. Bilkent University will be an external partner (not funded by the ANR). The bioinformatic group has a strong expertise in PIM and genomics with emerging collaborations with Pasteur and GenScale.

1 - CNRS-IRISA, GenScale

Description: The main goal of GenScale is to develop scalable methods, tools, and software for processing genomic data. The GenScale research is motivated by the fast development of next-generation sequencing (NGS) and third generation (TGS) technologies that provide very challenging problems both in terms of bioinformatics and computer sciences. GenScale research is organized along four main axes: data structures, algorithms, parallelism and applications.

2 - Institut Pasteur

Description: The 5-year group "Sequence Bioinformatics" at Institut Pasteur (started January 2019) performs computational research that is directly applicable to bioinformatics and biology. Concretely, the group develops and implements algorithms and data structures in the areas of genomics, metagenomics, pan-genomics, transcriptomics and proteomics. Ongoing projects include the analysis of human variation from whole-genome sequencing data (e.g. Alzheimer's disease), the development of algorithms on metagenomics microbial genome assembly, and a search engine for all previously sequenced human RNA-seq experiments.

3 - UPMEM

Description: Founded in 2015, UPMEM is a fabless semiconductor company making PIM a hardware reality for data centers. With a patented design that preserves DRAM manufacturing processes, UPMEM PIM can be implemented on any type of platform. The company's ambition is to bring acceleration to data intensive applications and also reduce hardware cost and energy consumption. A natural application of UPMEM' solution is in genomics, where the nature of the data implies calculation over large datasets. The company, with a growing team of 18 people, also counts numerous partnerships with prestigious labs and research centers across the globe.

4 - Bilkent University

Description: The Computer Engineering Department of Bilkent University is a research and innovation-oriented department with BS, MS, and PhD programs. The faculty is composed of leading researchers in many areas of computer engineering and science, including: algorithms, affective computing, artificial intelligence, big data, bioinformatics, computer architecture, computer graphics, computer networks, computer vision, data mining, data security, database systems, information retrieval, machine learning, parallel and distributed systems, performance evaluation, scientific computing, and software engineering.

4. Impacts and Benefits

Scientific and technological contributions to the foundation of a new future technology.

This project has a major scientific and technological contribution : setting the standards for PIM-enabled genomics research, which is currently lacking. With these contributions, we will aim to commoditize PIM technology by building foundations for fast and energy efficient computing platforms to perform complex computations, especially in the genomic field. These PIM computing platforms will consist of cheaper hardware, democratizing massive computing, for example, in small hospitals and clinics (e.g., for diagnosis and treatment guidance). Our algorithms and PIM designs will be independent of sequencing platform types, and we will address issues specific to each leading sequencing technology, making it possible to be used for different purposes in different experimental and computational settings. Thus, our research will make it feasible to analyze genomic data routinely without the need of building and maintaining large infrastructures.

Potential for future social or economic impact or market creation.

The memory market size was 8 billion € in 2019, with a growth rate over 20%, where ~25% is generated by genomics analyses[1]. If genomics continues to drive the memory market growth at 25%/year, it will alone reach a market size of 7.5 billion € in 2025. Successful completion of this project will introduce a "killer application" for processing-in-memory technologies, and will make them commonly used in the field of genomics. It will also enable other PIM-based projects to further co-develop hardware and software to address massive computation problems with utmost energy efficiency. Therefore, this project will help create new markets for PIM developers. It will also pave the way to more powerful wearable computers, and by popularizing and commoditizing PIM technologies for other computational problems. For example, PIM might enable full analysis of data generated using lab-on-a-chip platforms that can be used for non-genomics tests at point-of-care. Among the participants of this proposal, UPMEM as a European SME, will have opportunities to grow in the genomics market and will lead further developments in PIM research and fabrication, which will create new jobs.

Building leading research and innovation

PIM technologies are emerging and currently at their infancy, therefore the workforce in designing, implementing, and building PIM architectures is limited. In this project, we will hire new PhD students and postdoctoral fellows across various disciplines (academic participants), as well as engineers (industry participants). This project will also serve as a training platform for excellent young researchers to build their knowledge and experience in algorithms design, and their applications in genomics.

Sustainable computing.

One key aspect of our research proposal is the ability of PIM to significantly reduce the power and energy consumption requirements of computing systems. In prior work, we have shown that the data movement between memory and processors consumes 67% of the total system energy in current mobile devices[2]. Such high energy consumption hampers the availability of mobile devices to perform demanding computations as genomic data processing requires. By minimizing the amount of data movement that is needed in computing systems, PIM has the potential to provide significant reduction in energy consumption. Although our primary aim is to significantly improve genome analysis, our proposed enhancements will also benefit computation at medium-scale (i.e., local clusters, private clouds), and large-scale (i.e., data centers, public clouds) computation. Therefore our gains in energy efficiency also affect computation at all levels for any type of data analytics. Energy consumption in data centers is expected to be near 8% of the world's total energy consumption in 2030. Even if a very modest reduction of 30% can be achieved by PIM across all applications, this would correspond to

[1] https://www.alliedmarketresearch.com/

[2] Boroumand et al., "Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks", ASPLOS, 2018.

more than 2% of the total energy consumption of the world. As a result, we expect that our research will help reduce the effects of energy consumption on the environment.

5. Bibliography

1. Wang et al., “Nanopore target sequencing for accurate and comprehensive detection of SARS-CoV-2 and other respiratory viruses”, medXiv, 2020.03.04.20029538, 2020
2. Arias et al., “Rapid outbreak sequencing of Ebola virus in Sierra Leone identifies transmission chains linked to sporadic cases”, Virus Evol. Jun 22;2(1):vew016, 2016
3. Shrivastava et al., “Whole genome sequencing, variant analysis, phylogenetics, and deep sequencing of Zika virus strains”, Sci Rep. Oct 26;8(1):15843, 2018
4. https://www.weforum.org/agenda/2020/04/coronavirus-covid-19-pandemic-digital-divide-internet-data-broadband-mobbile/
5. Mutlu et al., “Processing data where it makes sense: Enabling in-memory computation”, Microprocessors and Microsystems, 67:28-41, 2019.
6. Hybrid Memory Cube Consortium, HMC Specification 2.0, 2014
7. JEDEC, High Bandwidth Memory (HBM) DRAM, Standard No. JESD235, 2013
8. Ghose et al., “Processing-in-memory: A workload-driven perspective”, IBM J of Res. and Dev., 63(6):1-19, 2019
9. S. Ghose, A. Boroumand, J. S. Kim, J. Gómez-Luna and O. Mutlu, "Processing-in-memory: A workload-driven perspective," in *IBM Journal of Research and Development*, vol. 63, no. 6, pp. 3:1-3:19, 1 Nov.-Dec. 2019, doi: 10.1147/JRD.2019.2934048.
10. Stephens et al., “Big Data: Astronomical or Genomical?”, PLoS Biology, 13(7): e1002195, 2015
11. D. Lavenier, JF. Roy, D. Furodet, DNA Mapping using Processor-in-Memory Architecture, Workshop on Accelerator-Enabled Algorithms and Applications in Bioinformatics, Shenzhen, China, 2016
12. D. Lavenier, R. Jodin, R. Cimadomo, Variant Calling Parallelization on Processor-in-Memory Architecture International Conference on Bioinformatics and Biomedicine (BIBM 2020), Seoul, Korea, 2020
13. Illumina DRAGEN Bio-IT Platform v3.2.8. User Guide. 2019
14. https://www.nvidia.com/en-us/docs/parabricks/
15. F. Putze, Felix, P. Sanders, J. Singler, Cache-, hash-, and space-efficient Bloom filters. ACM Journal of Experimental Algorithmics, vol 14, 2009
16. M. Roberts, W. Hayes, B. R. Hunt, S. M. Mount, J. A. Yorke, Reducing storage requirements for biological sequence comparison, *Bioinformatics*, Volume 20, Issue 18, 2004
17. Y. Heo, X-L. Wu, D. Chen, J.Ma, W-M. Hwu, BLESS: Bloom filter-based error correction solution for high-throughput sequencing reads, *Bioinformatics*, Volume 30, Issue 10, 2014
18. A. Limasset, G. Rizk, R. Chikhi, P. Peterlongo, *Fast and Scalable Minimal Perfect Hashing for Massive Key Sets*, SEA 2017
19. R. Chikhi, J. Holub, P.Medvedev, Data structures to represent a set of k-long DNA sequences, arXiv:1903.12312v4, 2020