

Efficient data structures for bioinformatics information representation**DESCRIPTION****5 TECHNICAL FIELD**

[0001] This invention discloses a Genomic Information Storage Layer (Genomic File Format) which defines a genomic data structure that includes the collection of heterogeneous data associated to the information generated by devices and applications related to genome sequencing, processing and analysis during the different stages of genomic data processing (the so-called "genomic information life cycle").

10

BACKGROUND

[0002] Genomic or proteomic information generated by DNA, RNA, or protein sequencing machines is transformed, during the different stages of data processing, to produce heterogeneous data. In prior art solutions, these data are currently stored in computer files having different and unrelated structures. This information is therefore quite difficult to archive, transfer and elaborate.

[0003] The genomic or proteomic sequences referred to in this invention include, for example, and not as a limitation, nucleotide sequences, Deoxyribonucleic acid (DNA) sequences, Ribonucleic acid (RNA), and amino acid sequences. Although the description herein is in considerable detail with respect to genomic information in the form of a nucleotide sequence, it will be understood that the methods and systems for storage can be implemented for other genomic or proteomic sequences as well, albeit with a few variations, as will be understood by a person skilled in the art.

[0004] The genomic or proteomic information life cycle from data generation (sequencing) to analysis is depicted in figure 1, wherein the different phases of the genomic life cycle and the associated intermediate file formats are shown. As shown in figure 1, the typical steps of the genomic information life cycle comprise: Sequence Reads Extraction, Mapping and Alignment, Variant Detection, Variant Annotation and Functional and Structural Analysis.

[0005] Sequence Reads Extraction is the process – performed by either a human operator or a machine - of representation of fragments of genetic information in the form of sequences of symbols representing the molecules composing a biological sample. In the case of nucleic acids such molecules are called “nucleotides”. The sequences of symbols produced by the extraction are commonly referred to as “reads”. This information is usually encoded in prior art as “FASTA” files including a textual header and a sequence of symbols representing the sequenced molecules.

[0006] When the biological sample is sequenced to extract DNA of a living organism the alphabet is composed by the symbols (A,C,G,T,N).

[0007] When the biological sample is sequenced to extract RNA of a living organism the alphabet is composed by the symbols (A,C,G,U,N).

[0008] In case the IUPAC extended set of symbols, the so called “ambiguity codes” are also generated by the sequencing machine, the alphabet used for the symbols composing the reads is (A, C, G, T, U, W, S, M, K, R, Y, B, D, H, V, N or -).

[0009] When the IUPAC ambiguity codes are not used, a sequence of quality scores can be associated to each sequence read. In such case prior art solutions encode the resulting information as a “FASTQ” file.

[0010] Sequence alignment refers to the process of arranging sequence reads by finding regions of similarity that may be a consequence of functional, structural, or evolutionary relationships among the sequences. When the alignment is performed with reference to a pre-existing nucleotides sequence referred to as “reference sequence”, the process is called “mapping”. Sequence alignment can also be performed without a pre-existing sequence (i.e. reference genome) in such cases the process is known in prior art as “de novo” alignment. Prior art solutions store such information in “SAM”, “BAM” or “CRAM” files. The concept of aligning sequences to reconstruct a partial or complete genome is depicted in figure 2.

[0011] Variant Detection (a.k.a. variant calling) is the process of translating the aligned output of genome sequencing machines, to a summary of the unique characteristics of the organism being sequenced that cannot be found in other pre-existing sequences or can be found a few pre-existing sequences only. These characteristics are called “variants” because they are expressed as differences between the genome of the organism under study and a reference genome. Prior art solutions store this information in a specific file format called “VCF” file.

[0012] Variant Annotation is the process of assigning functional information to the genomic variants. This implies the classification of variants according to their relationship to coding sequences in the genome and according to their impact on the coding sequence and the gene product. This is in prior art usually stored in a “MAF” file.

[0013] Analysis of DNA strands (variant, CNV = copy number variation, methylation etc.) to define their relationship with genes (and proteins) functions and structure is called functional and structural analysis. Several different solutions exist in the prior art for the storage of this data.

[0014] A simplified vision of the relation among the file formats used in genome processing pipelines is depicted in figure 3. In this diagram file inclusion does not imply the existence of a nested file structure, but it only represents the type and amount of information that can be encoded for each format (i.e. SAM contains all information in FASTQ, but organized in a different file structure). CRAM contains the same genomic information as SAM/BAM, but it provides more flexibility in the type of compression that can be used, therefore it is represented as a superset of SAM/BAM.

[0015] The use of multiple file formats for the storage of genomic information is highly inefficient and costly. Having different file formats at different stages of the genomic information life cycle implies a linear growth of utilized storage space even if the incremental information is very small compared to the initial volume of sequencing data. This is becoming not sustainable in terms of both space and generated costs and it is hindering genomic applications from reaching a wider portion of population. Further disadvantages of known prior art solutions are listed below.

[0016] 1. Accessing, analysing or adding annotations (metadata) to raw data stored in compressed FASTQ files or any combination thereof requires decompression and recompression of the entire file with extensive usage of computational time and resources.

10 [0017] 2. Retrieving specific type of information such as read mapping position, read variant position and type, indels position and types or any other metadata and annotation contained in aligned data stored in BAM files requires the access to the whole data volume associated to each read. Selective access to a single class of metadata is not possible with prior art solutions.

[0018] 3. Prior art file formats require that the whole file is received at the end user before processing can start. For example the alignment of reads could start before the sequencing process has been completed relying on an appropriate data representation. Sequencing, alignment and analysis could proceed and run in parallel.

[0019] 4. Structuring and being able of distinguishing genomic data obtained by different sequencing processes according to their specific generation semantic (e.g. sequencing obtained at different time of the life of the same individual, is not possible to be supported relying on prior art solutions. The same occurs for sequencing obtained by different types of biological samples of the same individual.

20 [0020] 5. The encryption of entire or selected portions of the data is not supported by prior art solutions. For example the encryption of selected DNA regions; only those sequences containing variants; chimeric sequences only; unmapped sequences only; specific metadata (e.g. origin of the sequenced sample, identity of sequenced individual, type of sample) is not possible.

25 [0021] 6. The transcoding from sequencing data aligned to a given reference (i.e. a SAM/BAM file) to a new reference requires to process the entire data volume even if the new reference differs only by a single nucleotide position from the previous reference.

[0022] 7. Transfer of genomic data is slow and inefficient because the currently used data formats are organized into monolithic files of up to several hundred Gigabytes of size which need to be entirely transferred at the receiving end in order to be processed. This implies that the analysis of a small segment of the data requires the transfer of the entire file with significant costs in terms of consumed bandwidth and waiting time. Often online transfer is prohibitive for the large volumes of the data to be transferred, and the transport of the data is performed by physically moving storage media such as hard disk drives or storage servers from one location to another.

[0023] 8. Processing the data is slow and inefficient for to the fact that the information is not structured in such a way that the portions of the different classes of data and metadata required by commonly used analysis applications cannot be retrieved without the need of accessing the data in its totality. This fact implies that common analysis pipelines can require to run for days or weeks wasting precious and costly processing resources for the need at each stage of accessing, parsing and filtering large volumes of data even if the portions of data relevant for the specific analysis purpose is much smaller. These limitations are preventing health care professionals from timely obtaining genomic analysis reports and promptly reacting to diseases outbreaks.

[0024] It exist a clear need to provide an appropriate genomic sequencing data and metadata representation (Genomic File Format) by organizing and partitioning the data so that the compression of data and metadata is maximized and several functionality such as selective access and support for incremental updates and other data handling functionality useful at the different stages of the genome data life cycle are efficiently enabled.

The main aspects of the disclosed solution are:

1. The classification of the sequence reads in different classes according to the results of the alignment with respect to a reference sequence in order to enable selective access to encoded data according to criteria related to the alignment results. This implies a specification of a file format that "contains" structured data elements in compressed form. Such approach can be seen as the opposite of prior art approaches, SAM and BAM for instance, in which data are structured in non-compressed form and then the entire file is compressed. A first clear advantage of the approach is to be able to efficiently and naturally provide various forms of selective access to the data elements in the compressed domain which is impossible or extremely awkward in prior art approaches.

2. The decomposition of the classified reads into layers of homogeneous metadata in order to reduce the information entropy as much as possible. The decomposition of the genomic information into specific "layers" of homogeneous data and metadata presents the considerable advantage of enabling the definition of different models of the information sources characterized by low entropy. Such models not only can differ from layer to layer, but can also differ inside each layer. This structuring enables the use of the most appropriate specific compression for each class of data or metadata and portion of them with significant gains in coding efficiency versus prior art approaches.

3. The structuring the layers into Access Units, i.e. genomic information that can be decoded either independently by using only globally available parameters (e.g. decoder configuration) or by using information contained in other Access Units. When the compressed data within layers are partitioned into Data Blocks included into Access Units different models of the information sources characterized by low entropy can be defined.

4. The information is structured so that any relevant subset of data used by genomic analysis applications is efficiently and selectively accessible by means of appropriate interfaces. These features enable faster access to

data and yield a more efficient processing. A Master Index Table and Local Index Tables enable selective access to the information carried by the layers of encoded (i.e. compressed) data without the need to decode the entire volume of compressed data. Furthermore, an association mechanism among the various data layers is specified to enable the selective access of any possible combination of subsets of semantically associated data and/or metadata layers without the need to decode all the layers.

5. The joint storage of the Master Index Table and the Access Units.

BRIEF DESCRIPTION OF DRAWINGS

Figure 1 is a block diagram of the typical genomic information life cycle.

10 Figure 2 is a diagram showing the concept of aligning sequences to reconstruct a partial or complete genome is depicted.

Figure 3 is a conceptual diagram illustrating a simplified vision of the relation among the file formats used in genome processing pipelines.

Figure 4 shows reads pairs mapped to a reference sequence.

15 Figure 5 shows an example of Access Units according to the principles of this disclosure.

Figure 6 shows an example of Access including a header and layers composed by data blocks.

Figure 7 shows the relation among genomic "Data Packets", "Blocks", Access Units, layers and Streams Reads Classes.

20 Figure 8 shows a master index table with the vectors of mapping loci of the first read contained by each Access Unit.

Figure 9 shows the generic structure of the Main Header and a partial representation of MIT showing the mapping positions of the first read in each pos AU of class P.

Figure 10 shows second type of data store in the MIT.

25 Figure 11 shows the Access Units containing reads of class P mapped on reference sequence no. 2 between position 150,000 and 250,000 are accessed using the values contained in the T1p vector.

Figure 12 Shows a modification in the reference sequence can transform M reads in P reads.

Figure 13 is a block diagram showing the genomic information life cycle according to the principles of this invention.

Figure 14 shows a sequence reads extractor according to the principles of this invention.

30 Figure 15 shows a genomic encoder 2010 according to the principles of this invention.

Figure 16 shows a genomic decoder 218 according to the principles of this invention.

SUMMARY

The features of claim 1 solve the problem of existing prior art solutions by providing

- 5 a method for the storage of a representation of genome sequence data in a genomic file format, said genome sequence data comprising reads of sequences of nucleotides, comprising the steps of: aligning said reads to one or more reference sequences thereby creating aligned reads, classifying said aligned reads according to different degrees of matching accuracy with said one or more reference sequences thereby creating classes of aligned reads; encoding said classified aligned reads as layers of syntax elements, structuring said layers of syntax
- 10 elements with header information thereby creating successive access units, creating a master index table, containing one section for each class of aligned reads, comprising the mapping positions on the reference sequence of the first read of each access units of each class of data; jointly storing said master index table and said access unit data.

- [0025] By jointly storing index tables and said representation of the genome sequence data, instead of
- 15 different separate files for each type of data of the genome sequence data representation as mentioned in the above life cycle description, many advantages are immediately apparent, namely:

- The results of any intermediate stage of genome sequence data processing can be incrementally added to the existing data without the need to transcode to a different file format. For example alignment information can be added to the raw data without the need to change the existing file format. Variants

20 calling results can be included in the existing aligned sequence data with an incremental update.

- The genomic sequence data can be retrieved according to their specific characteristics without the need to access the entire file or regions thereof that do not match the criteria of the query. For example queries can be executed to selectively access:
 - Sequence reads that perfectly match on one or more reference genomes

25 ○ Sequence reads that contain only mismatches where an “N” symbol is present instead of an actual nucleotide or amino acid symbol

 - Sequence reads that contain any type of mismatch in the form of substitution of symbols with respect to one or more genomes
 - Sequence reads that contain mismatches and insertion or deletions (indels)

30 ○ Sequence reads that contain mismatches, insertion or deletions (indels) and soft clipped symbols with respect to one or more reference genomes

 - Sequence reads that cannot be mapped with respect to the considered reference genome(s)
 - All the Single Nucleotide Polymorphisms (SNPs) which are present between specified thresholds of depth

35 ○ All chimeric sequence reads

- All sequence reads with quality scores above a specified threshold
- All metadata associated to a specified set of sequence reads

By classifying aligned reads according to a matching confidence with the reference sequence, selective access
 5 to encoded data according to criteria related to the alignment results can be achieved.

By encoding the classified aligned reads as layers of syntax elements, the encoding may be adapted according to the specific features of the data or metadata carried by the layer and its statistical properties.

By structuring the layers of syntax elements with header information in successive access units, the encoding, storage and transmission can be adapted according to the nature of the data. For example, the encoding can be
 10 adapted per access unit to use the most efficient source model for each data layer in terms of minimization of the entropy.

In accordance with one disclosed aspect, a method to extract reads of sequences of nucleotides stored in a genomic file, wherein said genomic file comprises a master index table and access units data stored according
 15 to the principles of this disclosure, said method comprising the steps of: receiving user input identifying the type of reads to be extracted, retrieving the master index table from said genomic file, retrieving the access units corresponding to said type of reads to be extracted, reconstructing said reads of sequences of nucleotides mapping said retrieved access units on one or more reference sequences.

[0026] The present invention further discloses a Genome Sequencing Machine comprising: A Genome
 20 Sequencing Machine comprising: a genome sequencing unit, configured to output reads of sequences of nucleotides from a biological sample, an alignment unit, configured to align said reads to one or more reference sequences thereby creating aligned reads, a classification unit, configured to classify said aligned reads according to matching accuracy degrees with said one or more reference sequences thereby creating classes of aligned reads; an encoding unit, configured to encode said classified aligned reads as layers of syntax elements, a
 25 subdividing unit, configured to structure said layers of syntax elements with header information thereby creating successive access units, an index table processing unit, configured to create a master index table, containing one section for each class of aligned reads, comprising the mapping positions on the one or more reference sequences of the first read of each access units of each class of data; a storage unit, configured to jointly storing said master index table and said access unit data.

30 In accordance with one disclosed aspect, an extractor to extract reads of sequences of nucleotides stored in a genomic file, wherein said genomic file comprises a master index table and access units data stored according to the principles of this disclosure, said extractor comprising: user input means configured to receive input identifying the type of reads to be extracted, retrieving means configured to retrieve said master index table from said genomic file, retrieving means configured to retrieve the access units corresponding to said type of
 35 reads to be extracted, reconstructing means configured to reconstruct said reads of sequences of nucleotides mapping said retrieved access units on one or more reference sequences.

[0027] In accordance with one disclosed aspect, a digital processing apparatus is programmed to perform a method as set forth in the immediately preceding paragraph. In accordance with another disclosed aspect, a non-transitory storage medium is accessed by a digital processing apparatus and stores instructions executable by the digital processing apparatus to perform a method as set forth in the preceding paragraph.

- 5 [0028] In accordance with another disclosed aspect, a non-transitory storage medium is readable by a digital processor and stores software for processing genomic or proteomic data represented as genomic or proteomic character strings comprising characters of a bioinformatics character set wherein each base or peptide of the genomic or proteomic data is represented in the format described in the preceding paragraphs. In some embodiments the software processes the genomic or proteomic data using digital signal processing
10 transformations.

DETAILED DESCRIPTION

Classification of the sequence reads

- [0029] The sequence reads generated by sequencing machines are classified by the disclosed invention into five
15 different “classes” according to the results of the alignment with respect to one or more reference sequences.

When aligning a DNA sequence of nucleotides with respect to a reference sequence five are the possible results:

1. A region in the reference sequence is found to match the sequence read without any error (perfect mapping). Such sequence of nucleotides will be referenced to as “perfectly matching read” or denoted
20 as “Class P”.
2. A region in the reference sequence is found to match the sequence read with a number of mismatches constituted by a number of positions in which the sequencing machine was not able to call any base (or nucleotide). Such mismatches are denoted by an “N”. Such sequences will be referenced to as “N mismatching reads” or “Class N”.
- 25 3. A region in the reference sequence is found to match the sequence read with a number of mismatches constituted by a number of positions in which the sequencing machine was not able to call any base (or nucleotide) OR a different base than the one reported in the reference genome has been called. Such type of mismatch is called Single Nucleotide Variation (SNV) or Single Nucleotide Polymorphism (SNP). The sequence will be referenced to as “M mismatching reads” or “Class M”.
- 30 4. A fourth class is constituted by sequencing reads presenting a mismatch type that includes the same mismatches of class M plus the presence of insertions or deletions (a.k.a. *indels*). Insertions are represented by a sequence of one or more nucleotides not present in the reference, but present in the read sequence. In literature when the inserted sequence is at the edges of the sequence it is referred to as “soft clipped” (i.e. the nucleotides are not matching the reference but are kept in the aligned
35 reads contrarily to “hard clipped” nucleotides which are discarded). Deletion are “holes” (missing

nucleotides) in the aligned read with respect to the reference. Such sequences will be referenced to as "I mismatching reads" or "Class I".

5. A fifth class includes all reads that do not find any valid mapping on the reference genome according to the specified alignment constraints. Such sequences are said to be Unmapped and belonging to "Class U".

Unmapped reads can be assembled into a single sequence using de-novo assembly algorithms. Once the new sequence has been created unmapped reads can be further mapped with respect to it and be classified in one of the four classes P, N, M and I.

10 Decomposition of the genomic information into layers.

[0030] Once the classification of reads is completed with the definition of the classes, further processing consists in defining a set of distinct syntax elements which represent the remaining information enabling the reconstruction of the DNA read sequence when represented as being mapped on a given reference sequence.

A DNA segment referred to a given reference sequence can be fully expressed by:

- 15
 - The starting position on the reference genome (*pos*).
 - A flag signaling if the read has to be considered as a reverse complement versus the reference (*rcomp*).
 - A distance to the mate pair in case of paired reads (*pair*).
 - The value of the read length in case of the sequencing technology produces variable length reads. In case of constant reads length the read length associated to each reads can obviously be omitted and
- 20
 - can be stored in the main file header.
 - Additional flags describing specific characteristics of the read (duplicate read, first or second read in a pair etc...).
 - For each mismatch:
 - Mismatch position (*nmis* for class N, *snpp* for class M, and *indp* for class I)
 - Mismatch type (not present in class N, *snpt* in class M, *indt* in class I)
- 25
 - Optional soft clipped nucleotides string when present (*indc* in class I).

This classification creates groups of descriptors (syntax elements) that can be used to univocally represent genome sequence reads. The table below summarizes the syntax elements needed for each class of aligned reads.

	P	N	M	I
<i>pos</i>	X	X	X	X
<i>pair</i>	X	X	X	X
<i>rcomp</i>	X	X	X	X
<i>flags</i>	X	X	X	X
<i>rlen</i>	X	X	X	X
<i>nmis</i>		X		
<i>snpp</i>			X	

snpt			X	
indp				X
indt				X
indc				X

Table 1 – Defined layers per class of data.

Reads belonging to class P are characterized and can be perfectly reconstructed by only a position, a reverse complement information and a distance between mates in case they have been obtained by a sequencing technology yielding mated pairs, some flags and a read length.

- 5 Figure 4 illustrates how reads can be coupled in pairs (according to the most common sequencing technology from Illumina Inc.) and mapped onto a reference sequence. Reads pairs mapped on the reference sequence are encoded into a multiplicity of layers of homogeneous descriptors (i.e. positions, distances between reads in one pair, mismatches etc....).

A layer is defined as a vector of descriptors related to one of the multiplicity of elements needed to uniquely
10 identify the reads mapped on the reference sequence. The following are examples of layers carrying each of them a vector of descriptors:

- Reads position layer
- Reverse complement layer
- Pairing information layer
- 15 • Mismatch position layer
- Mismatch type layer
- Indels layer
- Clipped bases layer
- Reads lengths layer (present only in case of variable reads length)
- 20 • BAM flags layer

Data Blocks, Access units and Genomic Data Layer

[0031] The data structure further disclosed by this invention relies on the concepts of:

25 **A Data Block** is defined as a set of the descriptor vector elements, of the same type (e.g. positions, distances, reverse complement flags, position and type of mismatch) composing a layer. One layer is typically composed by a multiplicity of data blocks. A data block can be partitioned into Genomic Data Packets which consist in transmission units having a size typically specified according to the communication channel requirements. Such partitioning feature is desirable for achieving transport efficiency using typical network communication protocols.

30 **An access unit** is defined as a subset of genomic data that can be fully decoded either independently from other access units by using only globally available data (e.g. decoder configuration) or by using information contained in other access units. An access unit is composed by a header and by the result of multiplexing data blocks of

different layers. Several packets of the same type are encapsulated in a block and several blocks are multiplexed in one access unit. These concepts are depicted in Figure 5. Figure 6 shows an access unit consisting of a header and one or more layers of data blocks of the same nature. Figure 6 shows an example of a generic access unit structure depicted in Figure 5 in which

- 5
 - data blocks of layer 1 contain information related to the position of reads on a reference sequence;
 - data blocks of layer 2 contain information about reverse complementarity of reads;
 - data blocks of layer 3 contain information related to read pairing information;
 - data blocks of layer 4 contain information about the reads length.

- 10 A **Genomic Data Layer** is defined as a set of genomic data blocks encoding data of the same type (e.g. position blocks of reads perfectly matching on a reference genome are encoded in the same layer).

A **Genomic Data Stream** is a packetized version of a Genomic Data Layer where the encoded genomic data is carried as payload of Genomic Data Packets including additional service data in a header. See Figure 7 for an example of packetization of 3 Genomic Data Layers into 3 Genomic Data Stream.

- 15 A **Genomic Data Multiplex** is defined as a sequence of Genomic Access Units used to convey genomic data related to one or more processes of genomic sequencing, analysis or processing. Figure 7 provides a schematic of the relation among a Genomic Multiplex carrying three Genomic Data Streams decomposed in Access Units. The Access Units encapsulate Data Blocks belonging to the three streams and partitioned into Genomic Packets to be sent on a transmission network.

- 20 **Source models, entropy coders and coding modes.**

[0032] For each layer of the genomic data structure disclosed in this invention different coding algorithms may be employed according to the specific features of the data or metadata carried by the layer and its statistical properties. The “coding algorithm” has to be intended as the association of a specific “source model” of the descriptor with a specific “entropy coder”. The specific “source model” can be specified and selected to obtain the most efficient coding of the data in terms of minimization of the source entropy. The selection of the entropy coder can be driven by coding efficiency considerations and/or probability distribution features and associated implementation issues. Each selection of a specific coding algorithm will be referred to as “coding mode” applied to an entire “layer” or to all “data blocks” contained into an access unit. Each “source model” associated to a coding mode is characterized by:

- 30
 - The definition of the syntax elements emitted by each source (e.g. reads position, reads pairing information, mismatches with respect to a reference sequence etc.)
 - The definition of the associated probability model.
 - The definition of the associated entropy coder.

For each data layer the source model adopted in one access unit is independent from the source model used by other access units for the same data layer. This enables each access unit to use the most efficient source model for each data layer in terms of minimization of the entropy.

5 TABLES

Master Index Table

[0033] In order to support selective access to specific regions of the aligned data, the data structure described in this document implements an indexing tool called Master Index Table (MIT). This is a multi-dimensional array containing two classes of data:

- 10 1. the loci at which specific reads map on the used reference sequences. These values contained in the MIT are the mapping positions of the first read in each pos access unit so that non-sequential access to each access unit is supported. These sections of the MIT contain one section per each class of data (P, N, M and I) and per each reference sequence.
2. pointers to the access units containing the data necessary to reconstruct the blocks of reads following those whose mapping positions are stored in the position vectors mentioned in point 1. Each vector of pointers is referred to as Local Index Table.

Access Units mapping positions

- [0034] Figure 8 shows a schematic of the MIT highlighting the four vectors containing the mapping positions on the reference sequence (possibly more than one) of each access units of each class of data.

The MIT is contained in the Main Header of the encoded data. Figure 9 shows the generic structure of the Main Header and an example of MIT vector for the class P of encoded reads.

The values contained in the MIT depicted in Figure 9 are used to directly access the region of interest (and the corresponding access unit) in the compressed domain.

- For example, with reference to Figure 9, if an analyst required to access perfectly matching reads mapped in the region comprised between position 150,000 and 250,000 on reference no. 2, a decoding application would skip to the class P position vector and the second reference in the MIT and would look for the two values k1 and k2 so that $k1 < 150,000$ and $k2 > 250,000$. In the example of Figure 9 this would result in positions 3 and 4 of the second block (second reference) of the MIT vector referring to mapping position of class P. These returned values will then be used by the decoding application to fetch the positions of the appropriate access units from the pos layer as described in the next section.

Access Units Pointers

[0035] The second type of data contained in the remaining vectors of the MIT (figure 8) consists in vectors of pointers to the physical position of each access unit in the encoded bitstream. Each vector is referred to as Local Index Table since its scope is limited to one homogeneous class of encoded information.

- 5 For each of the four classes of mapped reads (P, N, M, I) several types of access units are needed to reconstruct the encoded reads (pairs). The specific types of access units associated to each class of data depends on the result of the matching function applied on the reads in each classes with respect to one or more reference sequences as described above.

- 10 In the previous example of figure 9, in order to access region 150,000 to 250,000 of reads aligned on the reference sequence no. 2, the decoding application retrieved positions 3 and 4 from the positions vector of class P in the MIT. These values shall be used by the decoding process to access the 3rd and 4th elements of the corresponding access units vector (in this case the second) of the MIT. In the example shown in Figure 11, the Total Access Units counters contained in the Main Header are used to skip the positions of access units related to reference 1 (4 in the example). The indexes containing the physical positions of the requested access units in
15 the encoded stream are therefore calculated as:

Position of requested AU = AUs of reference 1 to be skipped + position retrieved using the MIT

i.e.

First AU position: $4 + 3 = 7$

Last AU position: $4 + 4 = 8$

- 20 This means that the region of interest (class P reads mapped on reference sequence no. 2 between position 150,000 and 250,000 is contained in the access units pointed to by the pointers stored in the 7th and 8th column of the Master Index Table, row T1p (Type 1 Access Units of type p).

Figure 11 shows how elements of one vector of the MIT (e.g. Class P Pos) point to elements of one LIT (Type 1 pos vector in the example of figure 11).

25

Adapting the reference sequence

- [0036] The mismatches encoded for classes N, M and I can be used to create a “modified genome” to be used to re-encode reads in the N, M or I layer (with respect to the first reference genome, R_0) as p reads with respect to the “adapted” genome R_1 . For example if denoted with r_{in}^M the i^{th} read of class M containing mismatches with respect to the reference genome n , then after “adaptation” it could be obtained $r_{in}^M = r_{i(n+1)}^P$ with $A(\text{Ref}_n) = \text{Ref}_{n+1}$
30 where A is the transformation from reference sequence n to reference sequence $n + 1$.

Figure 12 shows how reads containing mismatches (M reads) with respect to reference sequence 1 (RS1) can be transformed into perfectly matching reads (P reads) with respect to reference sequence 2 (RS2) obtained from RS1 by modifying the mismatching positions. This transformation can be expressed as

$$RS2 = A(RS1)$$

- 5 If the expression of transformation A which goes from RS1 to RS2 requires less bits of the expression of the mismatches present in the M reads, this encoding method results in a smaller information entropy and therefore better compression.

In some circumstances one or more modifications in the reference genome can reduce the overall information entropy by transforming a set of N, M or I reads to P reads.

- 10 [0037] A system architecture according to the principles of this invention is now described according to Figure 13. At a source, one or more genome sequencing devices 130 and/or applications generate and represent genomic information 131 in a format which contains

- One or more sequences of symbols representing nucleic acids
- A unique identifier per each genomic sequence
- 15 • An optional quality value per each symbol
- Optional metadata
- One or more optional reference sequences to be used to further process the generated genomic sequences.

- 20 [0038] A reads alignment unit 132 receives the raw sequence data and either aligns them on one or more available reference sequences or assembles them in longer sequences by looking for overlapping prefixes and suffixes applying a method known as “de-novo” assembly.

[0039] A reads classification unit 134 receives the aligned genome sequence data 133 and applies a matching function to each sequence with respect to:

- one or more available reference sequences or
- 25 • to an internal reference built during the alignment process (in case of “de-novo” assembly) .

[0040] A layers encoding unit 136 receives the reads classes 135 produced by the classification unit 134 and produces layers of syntax elements 137.

[0041] A header and Access Units encoding unit 138 encapsulates the syntax elements layers 137 in Access Units and adds a header to each Access Unit.

- 30 [0042] A Master Index Table encoding unit 1310 creates an index of pointers to the received Access Units 139

[0043] A compression unit 1312 transforms the output of said representation in a more compact (compressed) format 1315 to reduce the utilized storage space;

[0044] A local or remote storage device 1316 stores the compressed information 1315.

[0045] A decompression unit 1313 decompresses compressed information 1315 to retrieve decompressed data 1317 equivalent to genomic information 131.

5 [0046] An analysis unit 1314 further processes said genomic information 1317 by incrementally updating the metadata contained therein.

[0047] One or more genome sequencing devices or applications 1318 might add additional information to the existing genomic data by adding the results of a further genomic sequencing process without the need to re-encode the existing genomic information; to produce updated data 1319. Alignment and compression shall be applied to the newly generated genomic data prior to merging them with the existing data.

10 [0048] One of the several advantages of the embodiment described above is that genome analysis devices and application which need to have access to the data will be able to query and retrieve the needed information by using one or more of the index tables.

[0049] A sequence reads extractor 140 according to the principles of this invention is disclosed in figure 14.

15 [0050] The extractor device 140 utilises the Master Index Table described in this invention to have random access to any sequence reads stored in a Genomic File Format according to this disclosure. The extractor device 140 comprises user input means 141 to receive from the user input information 142 on the specific data to be retrieved. For example the user can specify:

- a. A genomic region in terms of:
 - 20 i. Start and end absolute position on a reference genome
 - ii. One entire reference sequence (e.g. a chromosome)
- b. Only one specific type of encoded sequence reads such as:
 - i. Sequence reads that perfectly match on one or more reference sequences
 - 25 ii. Sequence reads which present exactly N mismatches with respect to one or more reference sequences
 - iii. Sequence reads which present a number of mismatches with respect to one or more reference sequences below or above a specified threshold
 - iv. Sequence reads which present inserts and deletions with respect to a reference sequence.

30 The MIT extractor 143 of figure 14 parses the main header of the Genomic File to access the contained information as depicted in Figure 9:

- c. A unique identifier
- d. The version of the used syntax
- e. The size in byte of the main header
- 35 f. The number of reference sequences used to encode the sequence reads

- g. The number of data blocks contained by the stream
- h. The references identifiers
- i. The Master Index Table.

The MIT parser and AU extractor 145 retrieves the requested access units by exploiting the following information
 5 of the Master Index Table:

- j. vectors of the positions on the reference genome of the first read in each access unit. Figure 9 shows how the decoding device can read such position and find which Access Unit contain the encoded reads mapped within the requested region.
- 10 k. The Local Index Table of each encoded layer. These vectors are used to retrieve the physical position of those access units identified in steps a which contain the sequence reads mapped on the genomic region requested by a user
- l. The Local Index Tables are defined per each class of data, therefore the extractor device will extract only those classes referring to the sequence reads requested by the user. For example in case of a request for perfectly matching reads only, the extracted device will access only the
 15 LIT related to class P as depicted in Figure 8.

Using the information found in the retrieved access units and the one or more reference sequences either encoded in the genomic bitstream or available at the extracting device, the reads reconstructor 147 is able to reconstruct the original sequence reads.

20 Figure 15 shows an encoding apparatus 207 according to the principles of this invention. Encoding apparatus further clarifies the compression aspects of the system architecture of figure 13, however the Master Index Table and access units creation are omitted in the encoder of figure 15, which produces a compressed stream without those metadata and structuring information. The encoding apparatus 207 receives as input a raw sequence data 209, for example produced by a genome sequencing apparatus 200. Genome sequencing
 25 apparatus 200 are known in the art, like the Illumina HiSeq 2500 or the Thermo-Fisher Ion Torrent devices. The raw sequence data 209 is fed to an aligner unit 201, which prepares the sequences for encoding by aligning the reads to a reference sequence. Alternatively, a de-novo assembler 202 can be used to create a reference sequence from the available reads by looking for overlapping prefixes or suffixes so that longer segments (called "contigs") can be assembled from the reads. After having been processed by a de-novo assembler 202, reads
 30 can be mapped on the obtained longer sequence. The aligned sequences are then classified by data classification module 204. The data classes 208 are then fed to layers encoders 205-207. The genomic layers 2011 are then fed to arithmetic encoders 2012-2014 which encode the layers according to the statistical properties of the data or metadata carried by the layer. The result is a genomic stream 2015.

35 Figure 16 shows a corresponding decoding apparatus 218. A decoding apparatus 218 receives a multiplexed genomic bitstream 2110 from a network or a storage element. The multiplexed genomic bitstream 2110 is fed to a demultiplexer 210, to produce separate streams 211 which are then fed to entropy decoders 212-214, to produce genomic layers 215. The extracted genomic layers are fed to layer decoders 216-217 to further decode

the layers into classes of data. Class decoders 219 further process the genomic descriptors and merge the results to produce uncompressed reads of sequences, which can then be further stored in the formats known in the art, for instance a text file or zip compressed file, or FASTQ or SAM/BAM files. Class decoders 219 are able to reconstruct the original genomic sequences by leveraging the information on the original reference sequences carried by one or more genomic streams. In case the reference sequences are not transported by the genomic streams they must be available at the decoding side and accessible by the class decoders.

[0051] In one or more examples, the inventive techniques herewith disclosed may be implemented in hardware, software, firmware or any combination. When implemented in software, these may be stored on a computer medium and executed by a hardware processing unit. The hardware processing unit may comprise one or more processors, digital signal processors, general purpose microprocessors, application specific integrated circuits or other discrete logic circuitry.

The techniques of this disclosure may be implemented in a variety of devices or apparatuses, including mobile phones, desktop computers, servers, tablets and the like.

[0052] Many others advantages are described in the following claims.

15