METHODS AND SYSTEMS FOR GENOTYPING GENETIC SAMPLES

RELATED APPLICATIONS

This application claims priority to U.S. Patent Application No. 61/892,662, filed October 18, 2013, which is incorporated herein by reference in its entirety.

FIELD OF THE INVENTION

The invention relates to methods and systems for genotyping genetic samples.

BACKGROUND

Advances in sequencing technology make it possible to sequence the genome of an individual in a week or less. Typically, a genetic sample is isolated, broken into pieces, amplified, and then sequenced on a high-throughput system such as Illumina™ sequencing (Illumina, Inc., San Diego, CA). This process produces a huge number of sequence reads that must subsequently be assembled to create the sequence. The sequence provides very little useful information by itself, however, because the diagnostic and predictive value of the information depends upon the relative position of the sequence in the genome. That is, it is only possible to determine the presence of, e.g., a marker for a disease, when the relative position of the sequence within the genome is known. Additionally, once the sequence at a particular position is known, higher-level information, such as phenotypes, allelic identity, genotypes, etc., can be determined.

Because of the size and variability within each genome, locating where each read belongs is a substantial hurdle to genotyping in next generation sequencing (N.G.S.) The problem of determining a genotype based on a set of reads is naturally framed in evidential terms. Each read provides evidence of some genotype or set of genotypes, and combining the evidence from all the reads lets us conclude something about the subject's genotype. Both parts of this process--interpreting what a single read evinces about the subject's genome and aggregating the evidence of many reads--present problems, however. Furthermore, when dealing with actual genetic data, the sheer number of reads, and the presence of larger structural variations within the genetic data create a challenge analogous to a jigsaw puzzle with millions of pieces and little variations between pieces.

State-of-the-art alignment methods use massive computing power to align overlapping reads to a reference to produce an assembled sequence that can be probed for important genetic or structural information (e.g., biomarkers for disease). Ultimately, the goal of sequence alignment is to combine the set of nucleic acid reads produced by the sequencer to achieve a longer read (i.e., a contig) or even the entire genome of the subject based upon a genetic sample from that subject. Because the sequence data from next generation sequencers often comprises millions of shorter sequences that together represent the totality of the target sequence, aligning the reads is complex and computationally expensive. Additionally, in order to minimize sequence distortions caused by random sequencing errors (i.e., incorrect sequencing machine outputs), each portion of the probed sequence is sequenced multiple times (e.g., 2 to 100 times, or more) to minimize the influence of any random sequencing errors on the final alignments and output sequences generated. Finally, once all of the data corresponding to all of the nucleic acid reads is collected, the reads are aligned against a single reference sequence, e.g., GRCh37, in order to determine all (or some part of) the subject's sequence. In many instances, the individual reads are not actually displayed, but rather an aligned sequence is assembled into a sampled sequence, and the sampled sequence is provided as a data file.

Typically a sequence alignment is constructed by aggregating pairwise alignments between two linear strings of sequence information. As an example of alignment, two strings, S1 (SEQ ID NO. 20: AGCTACGTACACTACC) and S2 (SEQ ID NO. 21: AGCTATCGTACTAGC) can be aligned against each other. S1 typically corresponds to a read and S2 correspond to a portion of the reference sequence. With respect to each other, S1 and S2 can include substitutions, deletions, and insertions. Typically, the terms are defined with regard to transforming string S1 into string S2: a substitution occurs when a letter or sequence in S2 is replaced by a different letter or sequence of the same length in S1, a deletion occurs when a letter or sequence in S2 is "skipped" in the corresponding section of S1, and an insertion occurs when a letter or sequence occurs in S1 between two positions that are adjacent in S2. For example, the two sequences S1 and S2 can be aligned as below. The alignment below represents thirteen matches, a deletion of length one, an insertion of length two, and one substitution:

(S1) AGCTA–CGTACACTACC (SEQ ID NO. 20)

(S2) AGCTATCGTAC– –TAGC (SEQ ID NO. 21)

One of skill in the art will appreciate that there are exact and approximate algorithms for

sequence alignment. Exact algorithms will find the highest scoring alignment, but can be computationally expensive. The two most well-known exact algorithms are Needleman-Wunsch (*J Mol Biol,* 48(3):443-453, 1970) and Smith-Waterman (*J Mol Biol,* 147(1):195-197, 1981; *Adv. in Math.* 20(3), 367-387, 1976). A further improvement to Smith-Waterman by Gotoh (*J Mol Biol,* 162(3), 705-708, 1982) reduces the calculation time from $O(m^2n)$ to O(mn) where m and n are the sequence sizes being compared and is more amendable to parallel processing. In the field of bioinformatics, it is Gotoh's modified algorithm that is often referred to as the Smith-Waterman algorithm. Smith-Waterman approaches are being used to align larger sequence sets against larger reference sequences as parallel computing resources become more widely and cheaply available. See, e.g., Amazon.com's cloud computing resources available at http://aws.amazon.com. All of the above journal articles are incorporated herein by reference in their entireties.

The Smith-Waterman (SW) algorithm aligns linear sequences by rewarding overlap between bases in the sequences, and penalizing gaps between the sequences. Smith-Waterman also differs from Needleman-Wunsch, in that SW does not require the shorter sequence to span the string of letters describing the longer sequence. That is, SW does not assume that one sequence is a read of the entirety of the other sequence. Furthermore, because SW is not obligated to find an alignment that stretches across the entire length of the strings, a local alignment can begin and end anywhere within the two sequences.

The SW algorithm is easily expressed for an *n* x *m* matrix *H*, representing the two strings of length *n* and *m*, in terms of equation (1) below:

$$H_{k0} = H_{0l} = 0 \quad (for\ 0 \le k \le n\ and\ 0 \le l \le m) \tag{1}$$

$$H_{ij} = max\{H_{i-1,j-1} + s(a_i, b_j), H_{i-1,j} - W_{in}, H_{i,j-1} - W_{del}, 0\}$$

$$(for\ 1 \le i \le n\ and\ 1 \le j \le m)$$

In the equations above, $s(a_i,b_j)$ represents either a match bonus (when $a_i = b_j$) or a mismatch penalty (when $a_i \neq b_j$), and insertions and deletions are given the penalties W_{in} and W_{del}, respectively. In most instances, the resulting matrix has many elements that are zero. This representation makes it easier to backtrace from high-to-low, right-to-left in the matrix, thus identifying the alignment.

Once the matrix has been fully populated with scores, the SW algorithm performs a

backtrack to determine the alignment. Starting with the maximum value in the matrix, the algorithm will backtrack based on which of the three values ($H_{i-1,j-1}$, $H_{i-1,j}$, or $H_{i,j-1}$) was used to compute the final maximum value for each cell. The backtracking stops when a zero is reached. See, e.g., FIG. 3(B), which does not represent the prior art, but illustrates the concept of a backtrack, and the corresponding local alignment when the backtrack is read. Accordingly, the "best alignment," as determined by the algorithm, may contain more than the minimum possible number of insertions and deletions, but will contain far less than the maximum possible number of substitutions.

When applied as SW or SW-Gotoh, the techniques use a dynamic programming algorithm to perform local sequence alignment of the two strings, S and A, of sizes m and n, respectively. This dynamic programming technique employs tables or matrices to preserve match scores and avoid recomputation for successive cells. Each element of the string can be indexed with respect to a letter of the sequence, that is, if S is the string ATCGAA, $S[1]$ = A, $S[4]$ = G, etc. Instead of representing the optimum alignment as $H_{i,j}$ (above), the optimum alignment can be represented as $B[j,k]$ in equation (2) below:

$$B[j, k] = \max(p[j, k], i[j, k], d[j, k], 0) \qquad (\text{for } 0 < j \leq m\ , 0 < k \leq n) \qquad (2)$$

The arguments of the maximum function, $B[j,k]$, are outlined in equations (3)-(5) below, wherein MISMATCH_PENALTY, MATCH_BONUS, INSERTION_PENALTY, DELETION_PENALTY, and OPENING_PENALTY are all constants, and all negative except for MATCH_BONUS. The match argument, $p[j,k]$, is given by equation (3), below:

$$\begin{aligned} p[j,k] &= \max(p[j\text{-}1,k\text{-}1], i[j\text{-}1,k\text{-}1], d[j\text{-}1,k\text{-}1]) + \text{MISMATCH_PENALTY, if } S[j] \neq A[k] \quad (3) \\ &= \max(p[j\text{-}1,k\text{-}1], i[j\text{-}1,k\text{-}1], d[j\text{-}1,k\text{-}1]) + \text{MATCH_BONUS, if } S[j] = A[k] \end{aligned}$$

the insertion argument $i[j,k]$, is given by equation (4), below:

$$\begin{aligned} i[j,k] = \max(&p[j\text{-}1,k] + \text{OPENING_PENALTY}, i[j\text{-}1,k], d[j\text{-}1,k] + \\ &\text{OPENING_PENALTY}) + \text{INSERTION_PENALTY} \end{aligned} \qquad (4)$$

and the deletion argument $d[j,k]$, is given by equation (5), below:

$$\begin{aligned} d[j,k] = \max(&p[j,k\text{-}1] + \text{OPENING_PENALTY}, i[j,k\text{-}1] + \\ &\text{OPENING_PENALTY}, d[j,k\text{-}1]) + \text{DELETION_PENALTY} \end{aligned} \qquad (5)$$

For all three arguments, the [0,0] element is set to zero to assure that the backtrack goes to completion, i.e., $p[0,0] = i[0,0] = d[0,0] = 0$.

The scoring parameters are somewhat arbitrary, and can be adjusted to achieve the behavior of the computations. One example of the scoring parameter settings (Huang, Chapter 3*: Bio-Sequence Comparison* and *Alignment*, ser. *Curr Top Comp Mol Biol.* Cambridge, Mass.: The MIT Press, 2002) for DNA would be:

MATCH_BONUS: 10

MISMATCH_PENALTY: −20

INSERTION_PENALTY: −40

OPENING_PENALTY: −10

DELETION_PENALTY: −5

The relationship between the gap penalties (INSERTION_PENALTY, OPENING_PENALTY) above help limit the number of gap openings, i.e., favor grouping gaps together, by setting the gap insertion penalty higher than the gap opening cost. Of course, alternative relationships between MISMATCH_PENALTY, MATCH_BONUS, INSERTION_PENALTY, OPENING_PENALTY and DELETION_PENALTY are possible.

Once the alignment is complete, the aligned sequences can be assembled to produce a sequence that can be compared to a reference (i.e., a genetic standard) to identify variants. Only after the assembled reads are compared to the reference, is it possible to make determinations of the genotype of the sample. After comparing the assembled sequence to the reference sequence, the differences are catalogued and then compared to reference mutation files such as variant call format (VCF) files or a single nucleotide polymorphism database (dbSNP). This standard method of genotyping is time consuming, however, and often requires massive duplication of read coverage to assure that sequencing/amplification errors are not mistaken for true mutations.

The entire genotyping process is further complicated by the presence of structural variations in the genetic sample, i.e., longer (250 bp or more, e.g, 1000 bp or more) sequences that are inserted into or deleted from the "regular" genome. There may also be duplications, inversions, or translocations. In many instances, a sample is "called" for one genotype rather because of the presence of such structural variations, mutations, inversions, translocations, etc. In other instances, mutations *within* the structural variation result in a sample being "called" for

yet a different genotype. In other instances, the sequences of interest have moved (relative to the "normal" position) because of their proximity to a structural variation.

Incorporating structural variations into the state-of-the art genotyping methods is very difficult, however, because there are many known variants from which any particular read might originate. For each N structural variations, there are approximately 2^N different references that have to be compared to the assembled sequence in order to genotype the sequence. In other words, to accommodate one structural variant, the assembled sequence must be compared to at least 2 separate reference sequences to genotype the sample, but accommodating 20 possible structural variations requires comparing the sequence to roughly one million different references. Even using state-of—the-art methods with parallel computing, this is a very expensive proposition. Furthermore, this combinatorial explosion makes it impossible to feasibly genotype longer sequences that include hundreds of possible structural variations. Thus, approximations are made in the name of reducing computational time. In other words, current methods do not fairly represent the structural variations that are commonly found in many genomes.

SUMMARY

The disclosure provides methods and systems for efficiently genotyping sequence reads by aligning the reads directly to a reference sequence construct that simultaneously accounts for multiple alleles at multiple loci in the genome of the organism. Furthermore, the methods and systems of the disclosure make it possible to deal with structural variations in an efficient way, greatly reducing the computing power necessary to genotype genetic samples using next generation sequencing (N.G.S.). Additionally, because the reference sequence construct accounts for the various possible alleles within the construct, it is possible to directly genotype the sample by merely aligning the reads of the sample to the construct. Particular patterns of alignment are only possible for particular genotypes, thus it is not necessary to compare the assembled sequence to a reference sequence and then compare the variations to mutation files associated with that reference. Disclosed herein is a method of genotyping a genetic sample, comprising: obtaining a plurality of sequence reads corresponding to a genetic sample; aligning the plurality of sequence reads to a reference sequence construct that represents different genotypes as alternative sequences at a first position in the construct; and genotyping the sample based upon the alignment of the plurality of sequence reads against the reference sequence construct. Also disclosed herein is a system for aligning a plurality of reads comprising a processor and memory, wherein the memory comprises instructions that, when executed, cause the processor to obtain a plurality of sequence reads corresponding to a genetic sample; align the plurality of sequence reads to a reference sequence construct that represents different genotypes as alternative sequences at a first position in the construct; and genotype the sample based upon the alignment of the plurality of sequence reads against the reference

sequence construct. Disclosed herein is also a computer-implemented method of genotyping a genetic sample, comprising: providing a plurality of sequence reads corresponding to a genetic sample; aligning the plurality of sequence reads to a reference sequence construct that represents different genotypes as alternative sequences at a first position in the construct using a computer processor; and genotyping the sample based upon the alignment of the plurality of sequence reads against the reference sequence construct. Thus, in one aspect, the present invention refers to a method of genotyping a genetic sample, comprising obtaining a plurality of sequence reads corresponding to a genetic sample, wherein the plurality of sequence reads include portions of a structural variation; aligning the plurality of sequence reads to a reference directed acyclic graph (DAG) that accounts for variability in genetic sequences within a species, the reference DAG comprising a first path corresponding to a first allele and a second path corresponding to a second allele at a position in the reference DAG, wherein the first allele comprises a genetic structural variation, wherein the plurality of sequence reads that include portions of the genetic structural variation properly align to the reference DAG; and determining a genotype for the genetic sample based upon the number of sequence reads aligning to the first path and the second path, wherein the alignment of two or more related reads is used to determine the genotype, and the determined genotype comprises the structural variation. In another aspect, the present invention refers to a computer-implemented method of genotyping a genetic sample, comprising providing a plurality of sequence reads corresponding to a genetic sample, wherein the plurality of sequence reads include portions of a structural variation; aligning the plurality of sequence reads to a reference directed acyclic graph (DAG) that accounts for variability in genetic sequences within a species, the reference DAG comprising nodes, wherein the reference DAG comprises a first path corresponding to a first allele and a second path corresponding to a second allele at a position in the reference DAG using a computer processor, wherein the first allele comprises a genetic structural variation, wherein the plurality of sequence reads that include portions of the genetic structural variation properly align to the reference DAG; and determining a genotype for the genetic sample based upon the number of sequence reads aligning to the first path and the second path, wherein the alignment of two or more related reads is used to determine the genotype, and the determined genotype comprises the structural variation. In a further aspect, the present invention refers to a system for aligning a plurality of reads comprising a processor and memory, wherein the memory comprises instructions that, when executed, cause the processor to obtain a plurality of sequence reads corresponding to a genetic sample, wherein the plurality of sequence reads include portions of a structural variation; align the plurality of sequence reads to a reference directed acyclic graph (DAG) that accounts for variability in genetic sequences within a species, wherein the reference DAG comprises a first path corresponding to a first allele and a second path corresponding to a second allele at a position in the reference DAG, wherein the first allele comprises a genetic structural variation, wherein the plurality

of sequence reads that include portions of the genetic structural variation properly align to the reference DAG; and determine a genotype for the genetic sample based upon the number of sequence reads aligning to the first path and the second path, wherein the alignment of two or more related reads is used to determine the genotype, and the determined genotype comprises the structural variation.

The methods and systems of the invention transform linear, local sequence alignment processes such as, for example, Smith-Waterman-Gotoh, into multi-dimensional alignment algorithms that provide increased parallelization, increased speed, increased accuracy, and the ability to align reads through an entire genome. Algorithms of the invention provide for a "look-

back" type analysis of sequence information (as in Smith-Waterman), however, in contrast to known linear methods, the look back of the invention is conducted through a multi-dimensional space that includes multiple pathways and multiple nodes in order to provide more accurate alignment of complex and lengthy sequence reads, while achieving lower overall rates of mismatches, deletions, and insertions. In many instances several of the pathways represent a particular genotype for the organism. Thus, by aligning reads to a set of alleles representing a particular pathway, the genotype is immediately identified.

In practice, the invention is implemented by aligning sequence reads to a series of directed, acyclic sequences spanning branch points that account for all, or nearly-all, of the possible sequence variation in the alignment, including insertions, deletions, substitutions, and structural variations. Such constructs, often represented as directed acyclic graphs (DAGs), can be easily assembled from available sequence databases, including "accepted" reference sequences and variant call format (VCF) entries. When combined with DAGs, or other directional constructs, the disclosed algorithm thus provides a multi-dimensional approach to sequence alignment that greatly improves alignment accuracy and allows for sequence resolution not possible with conventional algorithms.

The invention additionally includes methods for constructing a directed acyclic graph data structure (DAG) that represents known variants at positions within the sequence of an organism. The DAG may include multiple sequences at thousands of positions, and may include multiple variants at each position, including deletions, insertions, translations, inversions, single-nucleotide polymorphisms (SNPs), and structural variations. It is also possible to tag each variant in the DAG with a genotype or other correlated diagnostic information, such as "breast cancer," thereby reducing the steps needed to acquire the valuable diagnostic information within the reads. In some embodiments, the variants are correlated to provide greater confidence in genotype calls. In some embodiments, the variants will be scored, weighted, or correlated with other variants to reflect the prevalence of that variant as a marker for disease.

The invention additionally includes systems for executing the methods of the invention. In one embodiment, a system comprises a distributed network of processors and storage capable of comparing a plurality of sequences (i.e., nucleic acid sequences, amino acid sequences) to a reference sequence construct (e.g., a DAG) representing observed variation in a genome or a region of a genome. The system is additionally capable of aligning the nucleic acid reads to

produce a continuous sequence using an efficient alignment algorithm. Because the reference sequence construct compresses a great deal of redundant information, and because the alignment algorithm is so efficient, the reads can be tagged and assembled on an entire genome using commercially-available resources. The system comprises a plurality of processors that simultaneously execute a plurality of comparisons between a plurality of reads and the reference sequence construct. The comparison data may be accumulated and provided to a health care provider. Because the comparisons are computationally tractable, analyzing sequence reads will no longer represent a bottleneck between NGS sequencing and a meaningful discussion of a patient's genetic risks.

BRIEF DESCRIPTION OF THE FIGURES

FIG. 1 depicts the construction of a directed acyclic graph (DAG) representing genetic variation in a reference sequence. FIG. 1(A) shows the starting reference sequence and the addition of a deletion. FIG. 1(B) shows the addition of an insertion and a SNP, thus arriving at the Final DAG used for alignment;

FIG. 2 depicts three variant call format (VCF) entries represented as directed acyclic graphs;

FIG. 3(A) shows a pictorial representation of aligning a nucleic acid sequence read against a construct that accounts for an insertion event as well as the reference sequence;

FIG. 3(B) shows the matrices and the backtrack used to identify the proper location of the nucleic acid sequence read "ATCGAA";

FIG. 4 depicts two alternative alleles at a position in a genome of an organism and a reference sequence construct incorporating both alleles. The second allele is different from the first in that it includes a long insertion, i.e., a structural variant. One pathway through the reference sequence construct represents the first allele and a second pathway through the construct represents the second allele;

FIG. 5 shows how four separate reads can be aligned to the reference sequence construct. Upon alignment, some reads can be identified as corresponding to only allele #1 or allele #2, thus removing the need to compare differences between the assembled reads and a reference sequence to mutation files to determine the presence of a particular allele;

FIG. 6 depicts three alternative alleles at a position in a genome of an organism and a reference sequence construct incorporating all three alleles. The second and third alleles are different from the first in that they include long insertions, i.e., structural variants. The second and third alleles differ in a single nucleotide polymorphism. One pathway through the reference sequence construct represents the first allele, a second pathway through the construct represents the second allele, and a third pathway through the construct represents the third allele;

FIG. 7 shows how three separate reads can be aligned to the reference sequence construct. Upon alignment, some reads can be identified as corresponding to only allele #1, allele #2, or allele #3, thus removing the need to compare differences between the assembled reads and a reference sequence to mutation files to determine the presence of a particular allele;

FIG. 8 depicts an associative computing model for parallel processing;

FIG. 9 depicts an architecture for parallel computation.

DETAILED DESCRIPTION

The invention includes methods for aligning nucleic acid sequences to a reference sequence construct, methods for building the reference sequence construct, and systems that use the alignment methods and constructs to produce alignments and assemblies. The reference sequence construct may be a directed acyclic graph (DAG), as described below, however the reference sequence can be any representation reflecting genetic variability in the sequences of different organisms within a species, provided the construct is formatted for alignment. In general, the reference sequence construct will comprise portions that are identical and portions that vary between different genotypes. Accordingly, the constructs account for varying alleles, i.e., correlating with different genotypes. The application additionally discloses methods for identifying a genotype or a risk of a disease based upon alignment of nucleic acid reads to various locations in the construct.

The invention additionally provides methods to make specific base calls at specific loci using a reference sequence construct, e.g., a DAG, that represents known variants at each locus of the genome. Because the sequence reads are aligned to the DAG during alignment, the subsequent step of comparing a mutation, vis-à-vis the reference genome, to a table of known mutations can be eliminated. Using the disclosed methods, it is merely a matter of identifying a nucleic acid read as being located at a known mutation represented on the DAG and calling that

mutation. Alternatively, when a mutation is not known (i.e., not represented in the reference sequence construct), an alignment may be found and the variant identified as a new mutation or genotype. The method also makes it possible to associate additional information, such as specific disease risk or disease progression, with known mutations that are incorporated into the reference sequence construct. Additionally, the reference sequence constructs make it possible to align sequence reads against potential structural variations without requiring massive computing resources.

Because of the efficiency of the methods, it is feasible to quickly align a plurality of sequence reads against the same reference sequence construct. The reads are typically at least about 20 base pairs (bp) in length, e.g., at least about 50 bp in length, e.g., at least about 80 bp in length, e.g., at least about 100 bp in length, e.g., at least about 150 bp in length, e.g., at least about 200 bp in length. In some embodiments, the plurality will include greater than about 1000 sequence reads, e.g., greater than about 10,000 sequence reads, e.g., greater than about 100,000 sequence reads, e.g., greater than about 1,000,000 sequence reads. In some embodiments, as described below, the plurality of sequence reads are aligned to the reference sequence construct using parallel processing. In some embodiments, two or more sequence reads may be related in that they are known to originate from the same area of the original sample. In some embodiments, the sequence reads may be paired mates having an insert of variable length between the reads. Techniques for preparing paired mates are known for use with a variety of next generation sequencing techniques, such as Illumina™ sequencing.

Reference Sequence Constructs

Unlike prior art sequence alignment methods that use a single reference sequence to align and genotype nucleic acid reads, the invention uses a construct that can account for the variability in genetic sequences within a species, population, or even among different cells in a single organism. Representations of the genetic variation can be presented as directed acyclic graphs (DAGs) (discussed above) or row-column alignment matrices, and these constructs can be used with the alignment methods of the invention provided that the parameters of the alignment algorithms are set properly (discussed below).

In preferred embodiments of the invention, the construct is a directed acyclic graph (DAG), i.e., having a direction and having no cyclic paths. (That is, a sequence path cannot

travel through a position on the reference construct more than once.) In the DAG, genetic variation in a sequence is represented as alternate nodes. The nodes can be a section of conserved sequence, or a gene, or simply a nucleic acid. The different possible paths through the construct represent known genetic variation. A DAG may be constructed for an entire genome of an organism, or the DAG may be constructed only for a portion of the genome, e.g., a chromosome, or smaller segment of genetic information. In some embodiments, the DAG represents greater than 1000 nucleic acids, e.g., greater than 10,000 nucleic acids, e.g., greater than 100,000 nucleic acids, e.g., greater than 1,000,000 nucleic acids. A DAG may represent a species (e.g., *homo sapiens*) or a selected population (e.g., women having breast cancer), or even smaller subpopulations, such as genetic variation among different tumor cells in the same individual.

A simple example of DAG construction is shown in FIG. 1. As shown in FIG. 1(A), the DAG begins with a reference sequence, shown in FIG. 1(A) as SEQ ID NO. 1: CATAGTACCTAGGTCTTGGAGCTAGTC. In practice, the reference sequence is often much longer, and may be an entire genome. The sequence is typically stored as a FASTA or FASTQ file. (FASTQ has become the default format for sequence data produced from next generation sequencers). In some embodiments, the reference sequence may be a standard reference, such as GRCh37. As recognized by those of skill, each letter (or symbol) in the sequence actually corresponds to a nucleotide (e.g., a deoxyribonucleotide or a ribonucleotide) or an amino acid (e.g., histidine, leucine, lysine, etc.).

At the next step, a variant is added to the reference sequence, as shown in the bottom image of FIG. 1(A). As shown in FIG. 1(A) the variant is the deletion of the sequence "AG" from the reference between the lines in the figure, i.e., SEQ ID NO. 2. Graphically, this deletion is represented by breaking the reference sequence into nodes before and after the deletion, and connecting the nodes with an edge and also creating a path from one node to the "AG" and then to the other node. Thus, one path between the nodes represents the reference sequence, while the other path represents the deletion.

In practice, the variants are called to the DAG by applying the entries in a variant call format (VCF) file, such as can be found at the 1000 Genomes Project website. Because each VCF file is keyed to a specific reference genome at a specific location, it is not difficult to identify where the strings should be located. In fact, each entry in a VCF file can be thought of

as combining with the reference to create separate graph, as displayed in FIG. 2. Note the VCF entries in FIG. 2 do not correspond to the VCF entries of FIG. 1.

Moving to FIG. 1(B), a second VCF entry, corresponding to an insertion "GG" at a specific position is added to produce an expanded DAG, i.e., including SEQ ID NO. 3 and SEQ ID NO. 4. Next, a third VCF entry can be added to the expanded DAG to account for a SNP earlier in the reference sequence, i.e., including SEQ ID NOS. 5-8. Thus, in three steps, a DAG has been created against which nucleic acid reads can be aligned (as discussed below.)

In practice, the DAGs are represented in computer memory (hard disk, FLASH, cloud memory, etc.) as a set of nodes, S, wherein each node is defined by a string of characters, a set of parent nodes, and a position. The string is the node's "content," i.e., sequence; the parent nodes define the node's position with respect to the other nodes in the graph; and the position of the node is relative to some canonical ordering in the system, e.g., the reference genome. While it is not strictly necessary to define the graph with respect to a reference sequence, it does make manipulation of the output data simpler. Of course, a further constraint on S is that it cannot include loops.

In many embodiments, the nodes comprise a plurality of characters, as shown in FIGS. 1(A) and 1(B), however it is possible that a node may be a single character, e.g., representing a single base, as shown in FIG. 2. In instances where a node represents a string of characters, all of the characters in the node can be aligned with a single comparison step, rather than character-by-character calculations, as is done with conventional Smith-Waterman techniques. As a result, the computational burden is greatly reduced as compared to state-of-the-are methods. The reduced computational burden allows the alignment to be completed quicker, and with fewer resources. When used in next generation sequencing, where millions of small reads need to be aligned and assembled, this reduction in computational burden has tangible benefits in terms of reducing the cost of the alignment, while making meaningful information, i.e., genotype, available more quickly. In instances where a treatment will be tailored to a patient's genotype, the increased speed may allow a patient to begin treatment days earlier than using state-of-the-art methods.

Extrapolating this DAG method to larger structures, it is possible to construct DAGs that incorporate thousands of VCF entries representing the known variation in genetic sequences for a given region of a reference. Nonetheless, as a DAG becomes bulkier, the computations do take

longer, and for many applications a smaller DAG is used that may only represent a portion of the sequence, e.g., a chromosome. In other embodiments, a DAG may be made smaller by reducing the size of the population that is covered by the DAG, for instance going from a DAG representing variation in breast cancer to a DAG representing variation in triple negative breast cancer. Alternatively, longer DAGs can be used that are customized based upon easily identified genetic markers that will typically result in a large portion of the DAG being consistent between samples. For example, aligning a set of nucleic acid reads from an African-ancestry female will be quicker against a DAG created with VCF entries from women of African ancestry as compared to a DAG accounting for all variations known in humans over the same sequence. It is to be recognized that the DAGs of the invention are dynamic constructs in that they can be modified over time to incorporate newly identified mutations. Additionally, algorithms in which the alignment results are recursively added to the DAG are also possible.

In the instance of string-to-DAG alignment, the gap penalties can be adjusted to make gap insertions even more costly, thus favoring an alignment to a sequence rather than opening a new gap in the overall sequence. Of course, with improvements in the DAG (discussed above) the incidence of gaps should decrease even further because mutations are accounted for in the DAG.

Alignment Algorithm

In one embodiment, an algorithm is used to align sequence reads against a directed acyclic graph (DAG). In contrast to the algorithm expressed in the Background, the alignment algorithm identifies the maximum value for $C_{i,j}$ by identifying the maximum score with respect to each sequence contained at a position on the DAG (e.g., the reference sequence construct). In fact, by looking "backwards" at the preceding positions, it is possible to identify the optimum alignment across a plurality of possible paths.

The algorithm of the invention is carried out on a read (a.k.a. "string") and a directed acyclic graph (DAG), discussed above. For the purpose of defining the algorithm, let S be the string being aligned, and let D be the directed acyclic graph to which S is being aligned. The elements of the string, S, are bracketed with indices beginning at 1. Thus, if S is the string ATCGAA, $S[1]$ = A, $S[4]$ = G, etc.

For the DAG, each letter of the sequence of a node will be represented as a separate element, d. A *predecessor* of d is defined as:

(i) If *d* is not the first letter of the sequence of its node, the letter preceding *d* in its node is its (only) predecessor;

(ii) If *d* is the first letter of the sequence of its node, the last letter of the sequence of any node that is a parent of *d*'s node is a predecessor of *d*.

The set of all predecessors is, in turn, represented as *P[d]*.

In order to find the "best" alignment, the algorithm seeks the value of *M*[*j,d*], the score of the optimal alignment of the first *j* elements of *S* with the portion of the DAG preceding (and including) *d*. This step is similar to finding $H_{i,j}$ in equation 1 in the Background section. Specifically, determining *M*[*j,d*] involves finding the maximum of *a, i, e,* and 0, as defined below:

$$M[j, d] = \max\{a, i, e, 0\} \quad (6)$$

where

$e = \max\{M[j, p^*] + \text{DELETE_PENALTY}\}$ for p^* in $P[d]$

$i = M[j\text{-}1, d] + \text{INSERT_PENALTY}$

$a =$ $max\{M[\text{j-}1, p^*] + \text{MATCH_SCORE}\}$ for p^* in $P[d]$, if $S[j] = d$;

$max\{M[j\text{-}1, p^*] + \text{MISMATCH_PENALTY}\}$ for p^* in $P[d]$, if $S[j] \neq d$

As described above, *e* is the highest of the alignments of the first *j* characters of *S* with the portions of the DAG up to, but not including, *d,* plus an additional DELETE_PENALTY. Accordingly, if *d* is not the first letter of the sequence of the node, then there is only one predecessor, *p*, and the alignment score of the first *j* characters of *S* with the DAG (up-to-and-including *p)* is equivalent to M[*j*,p] + DELETE_PENALTY. In the instance where *d* is the first letter of the sequence of its node, there can be multiple possible predecessors, and because the DELETE_PENALTY is constant, maximizing [*M*[*j*, *p**] + DELETE_PENALTY] is the same as choosing the predecessor with the highest alignment score with the first *j* characters of *S*.

In equation (6), *i* is the alignment of the first *j*-1 characters of the string *S* with the DAG up-to-and-including *d*, plus an INSERT_PENALTY, which is similar to the definition of the insertion argument in SW (see equation 1).

Additionally, *a* is the highest of the alignments of the first *j* characters of *S* with the portions of the DAG up to, but not including *d,* plus either a MATCH_SCORE (if the *j*th character of *S* is the same as the character *d*) or a MISMATCH_PENALTY (if the *j*th character of *S* is not the same as the character *d*). As with *e*, this means that if *d* is not the first letter of the

sequence of its node, then there is only one predecessor, i.e., *p*. That means *a* is the alignment score of the first *j*-1 characters of *S* with the DAG (up-to-and-including *p),* i.e., M[*j*-1,*p*], with either a MISMATCH_PENALTY or MATCH_SCORE added, depending upon whether *d* and the *j*th character of *S* match. In the instance where *d* is the first letter of the sequence of its node, there can be multiple possible predecessors. In this case, maximizing {*M*[*j*, *p**] + MISMATCH_PENALTY or MATCH_SCORE} is the same as choosing the predecessor with the highest alignment score with the first *j*-1 characters of *S* (i.e., the highest of the candidate M[*j*-1,*p**] arguments) and adding either a MISMATCH_PENALTY or a MATCH_SCORE depending on whether *d* and the *j*th character of *S* match.

Again, as in the SW algorithm discussed in the Background, the penalties, e.g., DELETE_PENALTY, INSERT_PENALTY, MATCH_SCORE and MISMATCH_PENALTY, can be adjusted to encourage alignment with fewer gaps, etc.

As described in the equations above, the algorithm finds the maximum value for each read by calculating not only the insertion, deletion, and match scores for that element, but looking backward (against the direction of the DAG) to any prior nodes on the DAG to find a maximum score. Thus, the algorithm is able to traverse the different paths through the DAG, which contain the known mutations. Because the graphs are directed, the backtracks, which move against the direction of the graph, follow the preferred variant sequence toward the origin of the graph, and the maximum alignment score identifies the most likely alignment within a high degree of certainty. While the equations above are represented as "maximum" values, "maximum" is intended to cover any form of optimization, including, for example, switching the signs on all of the equations and solving for a minimum value.

Implementation of the disclosed algorithm is exemplified in FIG. 3, where a sequence "ATCGAA" is aligned against a DAG that represents a reference sequence SEQ ID NO. 10: TTGGATATGGG and a known insertion event SEQ ID NO. 11: TTGGATCGAATTATGGG, where the insertion is underlined. FIG. 3(A) shows a pictorial representation of the read being compared to the DAG while FIG. 3(B) shows the actual matrices that correspond to the comparison. Like the Smith-Waterman technique discussed in the Background, the algorithm of the invention identifies the highest score and performs a backtrack to identify the proper location of the read. FIG. 3(A) and (B) also highlights that the invention produces an actual match for the string against the construct, whereas the known methods (e.g., SW) would have been more likely

to align the string to the wrong part of the reference, or reject the string as not generating a sufficiently-high alignment score to be included in the alignment. In the instances where the sequence reads include variants that were not included in the DAG, the aligned sequence will be reported out with a gap, insertion, etc.

Applications of the Reference Sequence Construct

One benefit of the reference construct and alignment algorithm of the invention is its ability to align sequence reads to either a first sequence or a second sequence at a certain position of the reference sequence construct. That is, a reference sequence construct of the invention allows a sequence read to align against one of at least two different sequence paths at a certain position—e.g., a path that follows a sequence equivalent to a reference sequence and another path that follows a known sequence equivalent to the reference sequence including variants (e.g. mutations, polymorphisms, copy number variations, structural variations). Thus, known variations in sequences can be reliably accounted for and identified using techniques of the invention by aligning reads containing the known variation to a sequence path that includes that variation.

Two examples of using directed acyclic graphs (DAG) to genotype reads are shown in FIGS. 4-7. Figure 4 shows two potential alleles at a position in the genome of an organism:

SEQ ID. NO. 12: CCCAGAACGTTGCATCGTAGACGAGTTTCAGCATT

SEQ ID. NO. 13: CCCAGAACGTTGCTATGCAACAAGGGACATCGTA GACGAGTTTCAGCATT

In this example, the two alleles differ by a 15 base insertion, i.e., a structural variation. As shown in FIG. 4, the two alleles can be depicted in a single reference sequence construct where the two alleles correspond to different pathways through the construct.

As shown in FIG. 5, as reads are aligned to the reference sequence construct, the reads can be immediately correlated with one or both of the alleles. Using the disclosed alignment algorithms, a read either aligns with a path corresponding to allele #1, a path corresponding to allele #2, or a read can align to either because the paths are common in the region aligning with the read. Based upon the number of reads aligning to a particular path, it is possible to immediately call a set of reads as corresponding to allele #1 or allele #2 without the additional steps of assembling the aligned reads and comparing the assembled reads to a reference

sequence. Additionally, as shown in FIG. 5, the methods efficiently align reads that comprise mostly common sequences, i.e., as shown for Read #4. Using state-of-the-art methods, i.e., linear alignment, read #4 would likely be aligned to the first allele with a tail sequence that is either discounted or assumed to be a transposition of bases. However, using the disclosed methods, it is clear that read #4 actually correlates to allele #2.

The alignment of the various reads in FIG. 5 emphasizes how much cleaner and more accurate a DAG-based genotyping process is, when compared to a linear-sequence-based process, especially one in which only a single reference sequence is used and mismatches are handled separately. As soon as the sequences are aligned to the reference DAG, we have good information about where the read is in the genome. It is additionally possible to gain extra information from such a construct by weighting or correlating particular branched sequences, thus allowing instant recognition of the rarity of an allele or the potential consequence of carrying that allele.

The methods of the invention allow even more versatility when important genetic differences are present within structural variations, as shown in FIGS. 6 and 7. As illustrated in FIG.6, it is possible to incorporate the additional complexity of a third allele having a SNP within the structural variation. Thus, as shown in FIG. 7, when reads are aligned to reference sequence construct #2, the reads can align to a path corresponding to allele #1, a path corresponding to allele #2, a path corresponding to allele #3, or a portion of the path common to all three alleles. Based upon the number of reads aligning to a particular path, it is possible to immediately call a set of reads as corresponding to allele #1, allele #2, or allele #3, or some combination thereof, without assembling the aligned reads and comparing the assembled reads to a reference sequence.

The situations depicted in FIGS. 4-7 should not be seen as limiting, however, as a reference sequence construct can include a multitude of different pathways, and the construct may include a series of different alternative sequences corresponding to positions of genetic variability. Statistical analysis and cross-correlation can also be used to assess a confidence level in a genotype call when thousands (or millions) of reads are processed, as is the case with contemporary sequencing.

Importantly, the disclosed methods advantageously allow sequence reads that include portions of a structural variations to be aligned and consequently genotyped. In contrast, using

one-dimensional reference sequences alignment (state-of-the-art), these reads may be rejected due to a low alignment score, and any variation within the portion of the read corresponding to the structural variation ignored. In some instances, the structural variations are large, typically between 1 Kb to 3Mb in size. However, for purposes of this application, structural variants may include any variation within a sequence read that deviates from the reference for 3 or more consecutive base pairs. In certain embodiments, the sequence length of the structural variant is about 20 bp, 50 bp, 80 bp, 100 bp, 200, bp, 300 bp, 400 bp, 500 bp, 600 bp, 700 bp, 800, bp, 1 Kb, 1.1. Kb, 1.2 Kb, 1.3 Kb, 1.4 Kb, 1.5 Kb, 1.6 Kb, 1.7 Kb, 1.8 Kb, 1.9 Kb, 2.0 Kb…2.0 Mb, 2.1 Mb, 2.2 Mb, 2.3 Mb, 2.4 Mb, 2.5 Mb, 2.6 Mb, 2.7 Mb, 2.8 Mb, 2.9 Mb, 3.0 Mb, etc. Structural variations provide important insight into a subject as they contribute to genetic diversity and disease susceptibility.

Unlike the current invention, traditional alignment methods (e.g., linear reference sequences) are unlikely to identify structural variations, and even less likely to identify rare variants located near a structural variation. Rare variants include any mutations (such as indel or polymorphism) that are found with low probably in a given population. For example, a rare variant may have a minor allele frequency ranging from, for example, 25% or fewer; 20% or fewer; 15% or fewer; 10% or fewer; or 5% or fewer. (Minor allele frequency (MAF) refers to the frequency at which the least common allele occurs in a given population.) In some instances, rare variants include variants that have not yet been identified, i.e., the variants aren't represented in the reference to which the read is aligned. In some instances the rare variant has not been cataloged in a VCF file. From the perspective of the alignment mechanism, such variants are effectively never-before-seen regardless of their actual frequency in a population of samples. A rare variant located near a structural variant may be separated from the structural variant by about the length of the read, i.e., about 100 bp or fewer. The invention is not limited to this spacing however. In some instances, a rare variant located near a structural variant has a separation between the rare variant and the structural variant may range from about 1 bp to about 1 Mbp, e.g., about 10 bp to about 10,000 bp, e.g., about 100 bp to about 1000 bp. Accordingly, the invention additionally allows for genotyping samples based upon minor alleles nearby structural variations on a large scale, e.g., a chromosome or an entire genome.

Because some rare variants confer a substantial risk of disease, it is of critical importance to maximize one's ability to detect rare variants during sequence assembly, and subsequently

genotype such samples. The reference constructs of the invention minimize nonalignment of structural variants and rare variants during the alignment process because references constructs of the invention can account for many different known structural variants. By including at least two structural variants at a certain location in the reference construct, the invention allows for sequence reads that include a portion of at least one of the structural variants to align to the reference construct. That is, sequence reads that include a portion of a known structural variant are aligned and accounted for, whereas the same structural variant would fail to align in a linear reference structure. The result of the invention is that reads that include structural variants are able to properly align to the DAG with a high degree of reliability and accuracy because the reads are treated as alignable rather than un-alignable.

With the structural variant properly aligned, other sequence data that is part of the sequence read with the structural variant likewise aligns to the reference construct. For example, a rare variant that is close to a structural variant (such that a sequence read includes at least portions of the structural variant and the rare variant) will align to the reference construct along with the structural variant. Thus, a rare variant next to a structural variant will present in a large number of otherwise well-aligned and reliable reads because of the proper alignment of the structural variant in the sequence read to the DAG reference construct. The consistent presence of the rare variant causes it to be recognized as a legitimate genetic variant rather than sequencing error, even if the variant is not represented in the reference construct.

Opportunities for Parallelization

The sequential version of the Smith-Waterman-Gotoh algorithm has been adapted and significantly modified for massive parallelization. For example, an ASC model, called Smith-Waterman using *Associative Massive Parallelism* (SWAMP) is described in U.S. Patent Publication No. 2012/0239706, incorporated herein by reference in its entirety. Part of the parallelization for SWAMP (and other parallel processing systems) stems from the fact that the values along any anti-diagonal are independent of each other. Thus, all of the cells along a given anti-diagonal can be done in parallel to distribute the computational resources. The data dependencies shown in the above recursive equations limit the level of achievable parallelism but using a wavefront approach will still speed up this useful algorithm. A wavefront approach implemented by Wozniak (*Comput Appl in the Biosciences* (*CABIOS*), 13(2):145-150, 1997) on

the Sun Ultra SPARC uses specialized SIMD-like video instructions. Wozniak used the SIMD registers to store the values parallel to the minor diagonal, reporting a two-fold speedup over a traditional implementation on the same machine. Following Wozniak's example, a similar way to parallelize code is to use the Streaming SIMD Extension (SSE) set for the x86 architecture. Designed by Intel, the vector-like operations complete a single operation/instruction on a small number of values (usually four, eight or sixteen) at a time. Many AMD and Intel chips support the various versions of SSE, and Intel has continued developing this technology with the Advanced Vector Extensions (AVX) for their modern chipsets.

In other implementations, Rognes and Seeberg (*Bioinformatics* (*Oxford, England*), 16(8):699-706, 2000) use the Intel Pentium processor with SSE's predecessor, MMX SIMD instructions for their implementation. The approach that developed out of the work of Rognes and Seeberg (*Bioinformatics,* 16(8):699-706, 2000) for ParAlign does not use the wavefront approach (Rognes, *Nuc Acids Res,* 29(7):1647-52, 2001; Saebo et al., *Nuc Acids Res,* 33(suppl 2):W535-W539, 2005). Instead, they align the SIMD registers parallel to the query sequence, computing eight values at a time, using a pre-computed query-specific score matrix. Additional details of this method can be found in U.S. 7,917,302, incorporated by reference herein. The way Rognes and Seeberg layout the SIMD registers, the north neighbor dependency could remove up to one third of the potential speedup gained from the SSE parallel "vector" calculations. To overcome this, they incorporate SWAT-like optimizations. With large affine gap penalties, the northern neighbor will be zero most of the time. If this is true, the program can skip computing the value of the north neighbor, referred to as the "lazy F evaluation" by Farrar (*Bioinformatics,* 23(2):156-161, 2007). Rognes and Seeberg are able to reduce the number of calculations of Equation 1 to speed up their algorithm by skipping it when it is below a certain threshold. A six-fold speedup was reported in (Rognes and Seeberg, *Bioinformatics,* 16(8):699-706, 2000) using 8-way vectors via the MMX/SSE instructions and the SWAT-like extensions.

In the SSE work done by Farrar (*Bioinformatics,* 23(2):156-161, 2007), a striped or strided pattern of access is used to line up the SIMD registers parallel to the query registers. Doing so avoids any overlapping dependencies. Again incorporating the SWAT-like optimizations (Farrar, *Bioinformatics* 23(2):156-161, 2007) achieves a 2-8 time speedup over Wozniak (*CABIOS* 13(2):145-150, 1997) and Rognes and Seeberg (*Bioinformatics* (*Oxford, England*), 16(8):699-706, 2000) SIMD implementations. The block substitution matrices and

efficient and clever inner loop with the northern (F) conditional moved outside of that inner loop are important optimizations. The strided memory pattern access of the sixteen, 8-bit elements for processing improves the memory access time as well, contributing to the overall speedup.

Farrar (*Sequence Analysis,* 2008) extended his work for a Cell Processor manufactured by Sony, Toshiba and IBM. This Cell Processor has one main core and eight minor cores. The Cell Broadband Engine was the development platform for several more Smith-Waterman implementations including SWPS3 by Szalkowski, et. al (*BMC Res Notes* 1(107), 2008) and CBESW by Wirawan, et. al (*BMC Bioinformatics* 9 (377) 2008) both using Farrar's striping approach. Rudnicki, et. al. (*Fund Inform.* 96, 181-194, 2009) used the PS3 to develop a method that used parallelization over multiple databases sequences.

Rognes (*BMC Bioinformatics* 12 (221), 2011) also developed a multi-threaded approach called SWIPE that processes multiple database sequences in parallel. The focus was to use a SIMD approach on "ordinary CPUs." This investigation using coarse-grained parallelism split the work using multiple database sequences in parallel is similar to the graphics processor units (GPU)-based tools described in the CUDASW by Liu, et al. (*BMC Res Notes* 2(73), 2009) and Ligowski and Rudnicki (*Eight Annual International Workshop on High Performance Computational Biology*, Rome, 2009). There have been other implementations of GPU work with CUDASW++2.0 by Liu, et. al. (*BMC Res Notes* 3(93), 2010) and Ligowski, et. al (*GPU Computing Gems, Emerald Edition*, Morgan Kaufmann, 155-157, 2011).

In other variations, small-scale vector parallelization (8, 16 or 32-way parallelism) can be used to make the calculations accessible via GPU implementations that align multiple sequences in parallel. The theoretical peak speedup for the calculations is a factor of m, which is optimal. A 96-fold speedup for the ClearSpeed implementation using 96 processing elements, confirming the theoretical speedup.

Parallel Computing Models

The main parallel model used to develop and extend Smith-Waterman sequence alignment is the ASsociative Computing (ASC) (Potter et al., *Computer,* 27(11):19-25, 1994). Efficient parallel versions of the Smith-Waterman algorithm are described herein. This model and one other model are described in detail in this section.

Some relevant vocabulary is defined here. Two terms of interest from Flynn's Taxonomy of computer architectures are MIMD and SIMD, two different models of parallel computing. A

cluster of computers, classified as a multiple-instruction, multiple-data (MIMD) model is used as a proof-of-concept to overcome memory limitations in extremely large-scale alignments. Section 8 describes usage of the MIMD model. An extended data-parallel, single-instruction multiple-data (SIMD) model known as ASC is also described.

Multiple Instruction, Multiple Data (MIMD)

The multiple-data, multiple-instruction model or MIMD model describes the majority of parallel systems currently available, and include the currently popular cluster of computers. The MIMD processors have a full-fledged central processing unit (CPU), each with its own local memory (Quinn, *Parallel Computing: Theory and Practice,* 2nd ed., New York: McGraw-Hill, 1994). In contrast to the SIMD model, each of the MIMD processors stores and executes its own program asynchronously. The MIMD processors are connected via a network that allows them to communicate but the network used can vary widely, ranging from an Ethernet, Myrinet, and InfiniBand connection between machines (cluster nodes). The communications tend to employ a much looser communications structure than SIMDs, going outside of a single unit. The data is moved along the network asynchronously by individual processors under the control of their individual program they are executing. Typically, communication is handled by one of several different parallel languages that support message-passing. A very common library for this is known as the Message Passing Interface (MPI). Communication in a "SIMD-like" fashion is possible, but the data movements will be asynchronous. Parallel computations by MIMDs usually require extensive communication and frequent synchronizations unless the various tasks being executed by the processors are highly independent (i.e. the so-called "embarrassingly parallel" or "pleasingly parallel" problems). The work presented in Section 8 uses an AMD Opteron cluster connected via InfiniBand.

Unlike SIMDs, the worst-case time required for the message-passing is difficult or impossible to predict. Typically, the message-passing execution time for MIMD software is determined using the average case estimates, which are often determined by trial, rather than by a worst case theoretical evaluation, which is typical for SIMDs. Since the worst case for MIMD software is often very bad and rarely occurs, average case estimates are much more useful. As a result, the communication time required for a MIMD on a particular problem can be and is usually significantly higher than for a SIMD. This leads to the important goal in MIMD programming (especially when message-passing is used) to minimize the number of inter-

processor communications required and to maximize the amount of time between processor communications. This is true even at a single card acceleration level, such as using graphics processors or GPUs.

Data-parallel programming is also an important technique for MIMD programming, but here all the tasks perform the same operation on different data and are only synchronized at various critical points. The majority of algorithms for MIMD systems are written in the Single-Program, Multiple-Data (SPMD) programming paradigm. Each processor has its own copy of the same program, executing the sections of the code specific to that processor or core on its local data. The popularity of the SPMD paradigm stems from the fact that it is quite difficult to write a large number of different programs that will be executed concurrently across different processors and still be able to cooperate on solving a single problem. Another approach used for memory-intensive but not compute-intensive problems is to create a virtual memory server, as is done with JumboMem, using the work presented in Section 8. This uses MPI in its underlying implementation.

Single Instruction, Multiple Data (SIMD)

The SIMD model consists of multiple, simple arithmetic processing elements called PEs. Each PE has its own local memory that it can fetch and store from, but it does not have the ability to compile or execute a program. As used herein, the term "parallel memory" refers to the local memories, collectively, in a computing system. For example, a parallel memory can be the collective of local memories in a SIMD computer system (e.g., the local memories of PEs), the collective of local memories of the processors in a MIMD computer system (e.g., the local memories of the central processing units) and the like. The compilation and execution of programs are handled by a processor called a control unit (or front end) (Quinn, *Parallel Computing: Theory and Practice,* 2nd ed., New York: McGraw-Hill, 1994). The control unit is connected to all PEs, usually by a bus.

All active PEs execute the program instructions received from the control unit synchronously in lockstep. "In any time unit, a single operation is in the same state of execution on multiple processing units, each manipulating different data" (Quinn, *Parallel Computing: Theory and Practice,* 2nd ed., New York: McGraw-Hill, 1994), at page 79. While the same instruction is executed at the same time in parallel by all active PEs, some PEs may be allowed to skip any particular instruction (Baker, SIMD and MASC: Course notes from CS 6/73301:

Parallel and Distributed Computing—power point slides, (2004)2004). This is usually accomplished using an "if-else" branch structure where some of the PEs execute the if instructions and the remaining PEs execute the else part. This model is ideal for problems that are "data-parallel" in nature that have at most a small number of if-else branching structures that can occur simultaneously, such as image processing and matrix operations.

Data can be broadcast to all active PEs by the control unit and the control unit can also obtain data values from a particular PE using the connection (usually a bus) between the control unit and the PEs. Additionally, the set of PE are connected by an interconnection network, such as a linear array, 2-D mesh, or hypercube that provides parallel data movement between the PEs. Data is moved through this network in synchronous parallel fashion by the PEs, which execute the instructions including data movement, in lockstep. It is the control unit that broadcasts the instructions to the PEs. In particular, the SIMD network does not use the message-passing paradigm used by most parallel computers today. An important advantage of this is that SIMD network communication is extremely efficient and the maximum time required for the communication can be determined by the worst-case time of the algorithm controlling that particular communication.

The remainder of this section is devoted to describing the extended SIMD ASC model. ASC is at the center of the algorithm design and development for this discussion.

Associative Computing Model

The ASsocative Computing (ASC) model is an extended SIMD based on the STARAN associative SIMD computer, designed by Dr. Kenneth Batcher at Goodyear Aerospace and its heavily Navy-utilized successor, the ASPRO.

Developed within the Department of Computer Science at Kent State University, ASC is an algorithmic model for associative computing (Potter et al., Computer, 27(11):19-25, 1994) (Potter, *Associative Computing: A Programming Paradigm for Massively Parallel Computers*, Plenum Publishing, 1992). The ASC model grew out of work on the STARAN and MPP, associative processors built by Goodyear Aerospace. Although it is not currently supported in hardware, current research efforts are being made to both efficiently simulate and design a computer for this model.

As an extended SIMD model, ASC uses synchronous data-parallel programming, avoiding both multi-tasking and asynchronous point-to-point communication routing. Multi-

tasking is unnecessary since only one task is executed at any time, with multiple instances of this task executed in lockstep on all active processing elements (PEs). ASC, like SIMD programmers, avoid problems involving load balancing, synchronization, and dynamic task scheduling, issues that must be explicitly handled in MPI and other MIMD cluster paradigms.

FIG. 8 shows a conceptual model of an ASC computer. There is a single control unit, also known as an instruction stream (IS), and multiple processing elements (PEs), each with its own local memory. The control unit and PE array are connected through a broadcast/reduction network and the PEs are connected together through a PE data interconnection network.

As seen in FIG. 8, a PE has access to data located in its own local memory. The data remains in place and responding (active) PEs process their local data in parallel. The reference to the word associative is related to the use of searching to locate data by content rather than memory addresses. The ASC model does not employ associative memory, instead it is an associative processor where the general cycle is to search-process-retrieve. An overview of the model is available in (Potter et al., *Computer,* 27(11):19-25, 1994).

The tabular nature of the algorithm lends itself to computation using ASC due to the natural tabular structure of ASC data structures. Highly efficient communication across the PE interconnection network for the lockstep shifting of data of the north and northwest neighbors, and the fast constant time associative functions for searching and for maximums across the parallel computations are well utilized by SWAMP

The associative operations are executed in constant time (Jin et al., 15*th International Parallel and Distributed Processing Symposium* (*IPDPS'*01) Workshops, San Francisco, p. 193, 2001), due to additional hardware required by the ASC model. These operations can be performed efficiently (but less rapidly) by any SIMD-like machine, and has been successfully adapted to run efficiently on several SIMD hardware platforms (Yuan et al., *Parallel and Distributed Computing Systems* (*PDCS*), Cambridge, M A, 2009; Trahan et al., *J. of Parallel and Distributed Computing* (*JPDC*), 2009). SWAMP and other ASC algorithms can therefore be efficiently implemented on other systems that are closely related to SIMDs including vector machines, which is why the model is used as a paradigm.

The control unit fetches and decodes program instructions and broadcasts control signals to the PEs. The PEs, under the direction of the control unit, execute these instructions using their own local data. All PEs execute instructions in a lockstep manner, with an implicit

synchronization between instructions. ASC has several relevant high-speed global operations: associative search, maximum/minimum search, and responder selection/detection. These are described in the following section.

Associative Functions

The functions relevant to the SWAMP algorithms are discussed below. Associative Search

The basic operation in an ASC algorithm is the associative search. An associative search simultaneously locates the PEs whose local data matches a given search key. Those PEs that have matching data are called responders and those with non-matching data are called non-responders. After performing a search, the algorithm can then restrict further processing to only affect the responders by disabling the non-responders (or vice versa). Performing additional searches may further refine the set of responders. Associative search is heavily utilized by SWAMP+ in selecting which PEs are active within a parallel act within a diagonal.

Maximum/Minimum Search

In addition to simple searches, where each PE compares its local data against a search key using a standard comparison operator (equal, less than, etc.), an associative computer can also perform global searches, where data from the entire PE array is combined together to determine the set of responders. The most common type of global search is the maximum/minimum search, where the responders are those PEs whose data is the maximum or minimum value across the entire PE array. The maximum value is used by SWAMP+ in every diagonal it processes to track the highest value calculated so far. Use of the maximum search occurs frequently, once in a logical parallel act, m+n times per alignment.

Responder Selection/Detection

An associative search can result in multiple responders and an associative algorithm can process those responders in one of three different modes: parallel, sequential, or single selection. Parallel responder processing performs the same set of operations on each responder simultaneously. Sequential responder processing selects each responder individually, allowing a different set of operations for each responder. Single responder selection (also known as pickOne) selects one, arbitrarily chosen, responder to undergo processing. In addition to multiple responders, it is also possible for an associative search to result in no responders. To handle this case, the ASC model can detect whether there were any responders to a search and perform a

separate set of actions in that case (known as anyResponders). In SWAMP, multiple responders that contain characters to be aligned are selected and processed in parallel, based on the associative searches mentioned above. Single responder selection occurs if and when there are multiple values that have the exact same maximum value when using the maximum/minimum search.

PE Interconnection Network

Most associative processors include some type of PE interconnection network to allow parallel data movement within the array. The ASC model itself does not specify any particular interconnection network and, in fact, many useful associative algorithms do not require one. Typically associative processors implement simple networks such as 1D linear arrays or 2D meshes. These networks are simple to implement and allow data to be transferred quickly in a synchronous manner. The 1D linear array is sufficient for the explicit communication between PEs in the SWAMP algorithms, for example.

Parallel Computing Systems

A generalized parallel processing architecture is shown in FIG. 9. While each component is shown as having a direct connection, it is to be understood that the various elements may be geographically separated but connected via a network, e.g., the internet. While hybrid configurations are possible, the main memory in a parallel computer is typically either shared between all processing elements in a single address space, or distributed, i.e., each processing element has its own local address space. (Distributed memory refers to the fact that the memory is logically distributed, but often implies that it is physically distributed as well.) Distributed shared memory and memory virtualization combine the two approaches, where the processing element has its own local memory and access to the memory on non-local processors. Accesses to local memory are typically faster than accesses to non-local memory.

Computer architectures in which each element of main memory can be accessed with equal latency and bandwidth are known as Uniform Memory Access (UMA) systems. Typically, that can be achieved only by a shared memory system, in which the memory is not physically distributed. A system that does not have this property is known as a Non-Uniform Memory Access (NUMA) architecture. Distributed memory systems have non-uniform memory access.

Processor–processor and processor–memory communication can be implemented in hardware in several ways, including via shared (either multiported or multiplexed) memory, a

crossbar switch, a shared bus or an interconnect network of a myriad of topologies including star, ring, tree, hypercube, fat hypercube (a hypercube with more than one processor at a node), or n-dimensional mesh.

Parallel computers based on interconnected networks must incorporate routing to enable the passing of messages between nodes that are not directly connected. The medium used for communication between the processors is likely to be hierarchical in large multiprocessor machines. Such resources are commercially available for purchase for dedicated use, or these resources can be accessed via "the cloud," e.g., Amazon Cloud Computing.

A computer generally includes a processor coupled to a memory via a bus. Memory can include RAM or ROM and preferably includes at least one tangible, non-transitory medium storing instructions executable to cause the system to perform functions described herein. As one skilled in the art would recognize as necessary or best-suited for performance of the methods of the invention, systems of the invention include one or more processors (e.g., a central processing unit (CPU), a graphics processing unit (GPU), etc.), computer-readable storage devices (e.g., main memory, static memory, etc.), or combinations thereof which communicate with each other via a bus.

A processor may be any suitable processor known in the art, such as the processor sold under the trademark XEON E7 by Intel (Santa Clara, CA) or the processor sold under the trademark OPTERON 6200 by AMD (Sunnyvale, CA).

Memory may refer to a computer-readable storage device and can include any machine-readable medium on which is stored one or more sets of instructions (e.g., software embodying any methodology or function found herein), data (e.g., embodying any tangible physical objects such as the genetic sequences found in a patient's chromosomes), or both. While the computer-readable storage device can in an exemplary embodiment be a single medium, the term "computer-readable storage device" should be taken to include a single medium or multiple media (e.g., a centralized or distributed database, and/or associated caches and servers) that store the one or more sets of instructions or data. The term "computer-readable storage device" shall accordingly be taken to include, without limit, solid-state memories (e.g., subscriber identity module (SIM) card, secure digital card (SD card), micro SD card, or solid-state drive (SSD)), optical and magnetic media, and any other tangible storage media. Preferably, a computer-readable storage device includes a tangible, non-transitory medium. Such non-transitory media

excludes, for example, transitory waves and signals. "Non-transitory memory" should be interpreted to exclude computer readable transmission media, such as signals, per se.

Input/output devices according to the invention may include a video display unit (e.g., a liquid crystal display (LCD) or a cathode ray tube (CRT) monitor), an alphanumeric input device (e.g., a keyboard), a cursor control device (e.g., a mouse or trackpad), a disk drive unit, a signal generation device (e.g., a speaker), a touchscreen, an accelerometer, a microphone, a cellular radio frequency antenna, and a network interface device, which can be, for example, a network interface card (NIC), Wi-Fi card, or cellular modem.

Sample Acquisition and Preparation

The invention includes methods for producing sequences (e.g., nucleic acid sequences, amino acid sequences) corresponding to nucleic acids recovered from biological samples. In some embodiments the resulting information can be used to identify mutations present in nucleic acid material obtained from a subject. In some embodiments, a sample, i.e., nucleic acids (e.g. DNA or RNA) are obtained from a subject, the nucleic acids are processed (lysed, amplified, and/or purified) and the nucleic acids are sequenced using a method described below. In many embodiments, the result of the sequencing is not a linear nucleic acid sequence, but a collection of thousands or millions of individual short nucleic acid reads that must be re-assembled into a sequence for the subject. Once the reads are aligned to produce a sequence, the aligned sequence can be compared to reference sequences to identify mutations that may be indicative of disease, for example. In other embodiments, the subject may be identified with particular mutations based upon the alignment of the reads against a reference sequence construct, i.e., a directed acyclic graph ("DAG") as described above.

For any of the above purposes, methods may be applied to biological samples. The biological samples may, for example, comprise samples of blood, whole blood, blood plasma, tears, nipple aspirate, serum, stool, urine, saliva, circulating cells, tissue, biopsy samples, hair follicle or other samples containing biological material of the patient. One issue in conducting tests based on such samples is that, in most cases only a tiny amount of DNA or RNA containing a mutation of interest may be present in a sample. This is especially true in non-invasive samples, such as a buccal swab or a blood sample, where the mutant nucleic acids are present in very small amounts. In some embodiments, the nucleic acid fragments may be naturally short,

that is, random shearing of relevant nucleic acids in the sample can generate short fragments. In other embodiments, the nucleic acids are purposely fragmented for ease of processing or because the sequencing techniques can only sequence reads of less than 1000 bases, e.g., less than 500 bases, e.g., less than 200 bases, e.g., less than 100 bases, e.g., less than 50 bases. While the methods described herein can be used to align sequences of varying length, in some embodiments, the majority of the plurality of nucleic acid reads will follow from the sequencing method and comprise less than 1000 bases, e.g., less than 500 bases, e.g., less than 200 bases, e.g., less than 100 bases, e.g., less than 50 bases.

Nucleic acids may be obtained by methods known in the art. Generally, nucleic acids can be extracted from a biological sample by a variety of techniques such as those described by Maniatis, et al., Molecular Cloning: A Laboratory Manual, Cold Spring Harbor, N.Y., pp. 280-281, (1982), the contents of which is incorporated by reference herein in its entirety.

It may be necessary to first prepare an extract of the sample and then perform further steps--i.e., differential precipitation, column chromatography, extraction with organic solvents and the like--in order to obtain a sufficiently pure preparation of nucleic acid. Extracts may be prepared using standard techniques in the art, for example, by chemical or mechanical lysis of the cell. Extracts then may be further treated, for example, by filtration and/or centrifugation and/or with chaotropic salts such as guanidinium isothiocyanate or urea or with organic solvents such as phenol and/or $HCCl_3$ to denature any contaminating and potentially interfering proteins. In some embodiments, the sample may comprise RNA, e.g., mRNA, collected from a subject sample, e.g., a blood sample. General methods for RNA extraction are well known in the art and are disclosed in standard textbooks of molecular biology, including Ausubel et al., Current Protocols of Molecular Biology, John Wiley and Sons (1997). Methods for RNA extraction from paraffin embedded tissues are disclosed, for example, in Rupp and Locker, Lab Invest. 56:A67 (1987), and De Andres et al., BioTechniques 18:42044 (1995). The contents of each of these references is incorporated by reference herein in their entirety. In particular, RNA isolation can be performed using a purification kit, buffer set and protease from commercial manufacturers, such as Qiagen, according to the manufacturer's instructions. For example, total RNA from cells in culture can be isolated using Qiagen RNeasy mini-columns. Other commercially available RNA isolation kits include MASTERPURE Complete DNA and RNA Purification Kit (EPICENTRE, Madison, Wis.), and Paraffin Block RNA Isolation Kit (Ambion, Inc.). Total

RNA from tissue samples can be isolated using RNA Stat-60 (Tel-Test). RNA prepared from tumor can be isolated, for example, by cesium chloride density gradient centrifugation.

Analytical Sequencing

Sequencing may be by any method known in the art. DNA sequencing techniques include classic dideoxy sequencing reactions (Sanger method) using labeled terminators or primers and gel separation in slab or capillary, sequencing by synthesis using reversibly terminated labeled nucleotides, pyrosequencing, 454 sequencing, allele specific hybridization to a library of labeled oligonucleotide probes, sequencing by synthesis using allele specific hybridization to a library of labeled clones that is followed by ligation, real time monitoring of the incorporation of labeled nucleotides during a polymerization step, polony sequencing, and SOLiD sequencing. Sequencing of separated molecules has more recently been demonstrated by sequential or single extension reactions using polymerases or ligases as well as by single or sequential differential hybridizations with libraries of probes. Prior to sequencing it may be additionally beneficial to amplify some or all of the nucleic acids in the sample. In some embodiments, the nucleic acids are amplified using polymerase chain reactions (PCR) techniques known in the art.

One example of a sequencing technology that can be used in the methods of the provided invention is Illumina sequencing (e.g., the MiSeq™ platform), which is a polymerase-based sequence-by-synthesis that may be utilized to amplify DNA or RNA. Illumina sequencing for DNA is based on the amplification of DNA on a solid surface using fold-back PCR and anchored primers. Genomic DNA is fragmented, and adapters are added to the 5' and 3' ends of the fragments. DNA fragments that are attached to the surface of flow cell channels are extended and bridge amplified. The fragments become double stranded, and the double stranded molecules are denatured. Multiple cycles of the solid-phase amplification followed by denaturation can create several million clusters of approximately 1,000 copies of single-stranded DNA molecules of the same template in each channel of the flow cell. Primers, DNA polymerase and four fluorophore-labeled, reversibly terminating nucleotides are used to perform sequential sequencing. After nucleotide incorporation, a laser is used to excite the fluorophores, and an image is captured and the identity of the first base is recorded. The 3' terminators and fluorophores from each incorporated base are removed and the incorporation, detection and identification steps are repeated. When using Illumina sequencing to detect RNA the same

method applies except RNA fragments are being isolated and amplified in order to determine the RNA expression of the sample. After the sequences are interrogated with the sequencer, they may be output in a data file, such as a FASTQ file, which is a text-based format for storing biological sequence and quality scores (see discussion above).

Another example of a DNA sequencing technique that may be used in the methods of the provided invention is Ion Torrent™ sequencing, offered by Life Technologies. See U.S. patent application numbers 2009/0026082, 2009/0127589, 2010/0035252, 2010/0137143, 2010/0188073, 2010/0197507, 2010/0282617, 2010/0300559, 2010/0300895, 2010/0301398, and 2010/0304982, the content of each of which is incorporated by reference herein in its entirety. In Ion Torrent™ sequencing, DNA is sheared into fragments of approximately 300-800 base pairs, and the fragments are blunt ended. Oligonucleotide adaptors are then ligated to the ends of the fragments. The adaptors serve as primers for amplification and sequencing of the fragments. The fragments can be attached to a surface and is attached at a resolution such that the fragments are individually resolvable. Addition of one or more nucleotides releases a proton (H^+), which signal detected and recorded in a sequencing instrument. The signal strength is proportional to the number of nucleotides incorporated. Ion Torrent data may also be output as a FASTQ file.

Another example of a DNA and RNA sequencing technique that can be used in the methods of the provided invention is 454™ sequencing (Roche) (Margulies, M et al. 2005, Nature, 437, 376-380). 454™ sequencing is a sequencing-by-synthesis technology that utilizes also utilizes pyrosequencing. 454™ sequencing of DNA involves two steps. In the first step, DNA is sheared into fragments of approximately 300-800 base pairs, and the fragments are blunt ended. Oligonucleotide adaptors are then ligated to the ends of the fragments. The adaptors serve as primers for amplification and sequencing of the fragments. The fragments can be attached to DNA capture beads, e.g., streptavidin-coated beads using, e.g., Adaptor B, which contains 5'-biotin tag. The fragments attached to the beads are PCR amplified within droplets of an oil-water emulsion. The result is multiple copies of clonally amplified DNA fragments on each bead. In the second step, the beads are captured in wells (pico-liter sized). Pyrosequencing is performed on each DNA fragment in parallel. Addition of one or more nucleotides generates a light signal that is recorded by a CCD camera in a sequencing instrument. The signal strength is proportional to the number of nucleotides incorporated. Pyrosequencing makes use of pyrophosphate (PPi)

which is released upon nucleotide addition. PPi is converted to ATP by ATP sulfurylase in the presence of adenosine 5' phosphosulfate. Luciferase uses ATP to convert luciferin to oxyluciferin, and this reaction generates light that is detected and analyzed. In another embodiment, pyrosequencing is used to measure gene expression. Pyrosequecing of RNA applies similar to pyrosequencing of DNA, and is accomplished by attaching applications of partial rRNA gene sequencings to microscopic beads and then placing the attachments into individual wells. The attached partial rRNA sequence are then amplified in order to determine the gene expression profile. Sharon Marsh, Pyrosequencing® Protocols in Methods in Molecular Biology, Vol. 373, 15-23 (2007).

Another example of a DNA and RNA detection techniques that may be used in the methods of the provided invention is SOLiD™ technology (Applied Biosystems). SOLiD™ technology systems is a ligation based sequencing technology that may utilized to run massively parallel next generation sequencing of both DNA and RNA. In DNA SOLiD™ sequencing, genomic DNA is sheared into fragments, and adaptors are attached to the 5' and 3' ends of the fragments to generate a fragment library. Alternatively, internal adaptors can be introduced by ligating adaptors to the 5' and 3' ends of the fragments, circularizing the fragments, digesting the circularized fragment to generate an internal adaptor, and attaching adaptors to the 5' and 3' ends of the resulting fragments to generate a mate-paired library. Next, clonal bead populations are prepared in microreactors containing beads, primers, template, and PCR components. Following PCR, the templates are denatured and beads are enriched to separate the beads with extended templates. Templates on the selected beads are subjected to a 3' modification that permits bonding to a glass slide. The sequence can be determined by sequential hybridization and ligation of partially random oligonucleotides with a central determined base (or pair of bases) that is identified by a specific fluorophore. After a color is recorded, the ligated oligonucleotide is cleaved and removed and the process is then repeated.

In other embodiments, SOLiD™ Serial Analysis of Gene Expression (SAGE) is used to measure gene expression. Serial analysis of gene expression (SAGE) is a method that allows the simultaneous and quantitative analysis of a large number of gene transcripts, without the need of providing an individual hybridization probe for each transcript. First, a short sequence tag (about 10-14 bp) is generated that contains sufficient information to uniquely identify a transcript, provided that the tag is obtained from a unique position within each transcript. Then, many

transcripts are linked together to form long serial molecules, that can be sequenced, revealing the identity of the multiple tags simultaneously. The expression pattern of any population of transcripts can be quantitatively evaluated by determining the abundance of individual tags, and identifying the gene corresponding to each tag. For more details see, e.g. Velculescu et al., Science 270:484 487 (1995); and Velculescu et al., Cell 88:243 51 (1997, the contents of each of which are incorporated by reference herein in their entirety).

Another sequencing technique that can be used in the methods of the provided invention includes, for example, Helicos True Single Molecule Sequencing (tSMS) (Harris T. D. et al. (2008) Science 320:106-109). In the tSMS technique, a DNA sample is cleaved into strands of approximately 100 to 200 nucleotides, and a polyA sequence is added to the 3' end of each DNA strand. Each strand is labeled by the addition of a fluorescently labeled adenosine nucleotide. The DNA strands are then hybridized to a flow cell, which contains millions of oligo-T capture sites that are immobilized to the flow cell surface. The templates can be at a density of about 100 million templates/cm^2. The flow cell is then loaded into an instrument, e.g., HeliScope.TM. sequencer, and a laser illuminates the surface of the flow cell, revealing the position of each template. A CCD camera can map the position of the templates on the flow cell surface. The template fluorescent label is then cleaved and washed away. The sequencing reaction begins by introducing a DNA polymerase and a fluorescently labeled nucleotide. The oligo-T nucleic acid serves as a primer. The polymerase incorporates the labeled nucleotides to the primer in a template directed manner. The polymerase and unincorporated nucleotides are removed. The templates that have directed incorporation of the fluorescently labeled nucleotide are detected by imaging the flow cell surface. After imaging, a cleavage step removes the fluorescent label, and the process is repeated with other fluorescently labeled nucleotides until the desired read length is achieved. Sequence information is collected with each nucleotide addition step. Further description of tSMS is shown for example in Lapidus et al. (U.S. patent number 7,169,560), Lapidus et al. (U.S. patent application number 2009/0191565), Quake et al. (U.S. patent number 6,818,395), Harris (U.S. patent number 7,282,337), Quake et al. (U.S. patent application number 2002/0164629), and Braslavsky, et al., PNAS (USA), 100: 3960-3964 (2003), the contents of each of these references is incorporated by reference herein in its entirety.

Another example of a sequencing technology that may be used in the methods of the provided invention includes the single molecule, real-time (SMRT) technology of Pacific

Biosciences to sequence both DNA and RNA. In SMRT, each of the four DNA bases is attached to one of four different fluorescent dyes. These dyes are phospholinked. A single DNA polymerase is immobilized with a single molecule of template single stranded DNA at the bottom of a zero-mode waveguide (ZMW). A ZMW is a confinement structure which enables observation of incorporation of a single nucleotide by DNA polymerase against the background of fluorescent nucleotides that rapidly diffuse in an out of the ZMW (in microseconds). It takes several milliseconds to incorporate a nucleotide into a growing strand. During this time, the fluorescent label is excited and produces a fluorescent signal, and the fluorescent tag is cleaved off. Detection of the corresponding fluorescence of the dye indicates which base was incorporated. The process is repeated. In order to sequence RNA, the DNA polymerase is replaced with a with a reverse transcriptase in the ZMW, and the process is followed accordingly.

Another example of a sequencing technique that can be used in the methods of the provided invention is nanopore sequencing (Soni G V and Meller, AClin Chem 53: 1996-2001) (2007). A nanopore is a small hole, of the order of 1 nanometer in diameter. Immersion of a nanopore in a conducting fluid and application of a potential across it results in a slight electrical current due to conduction of ions through the nanopore. The amount of current which flows is sensitive to the size of the nanopore. As a DNA molecule passes through a nanopore, each nucleotide on the DNA molecule obstructs the nanopore to a different degree. Thus, the change in the current passing through the nanopore as the DNA molecule passes through the nanopore represents a reading of the DNA sequence.

Another example of a sequencing technique that can be used in the methods of the provided invention involves using a chemical-sensitive field effect transistor (chemFET) array to sequence DNA (for example, as described in US Patent Application Publication No. 20090026082). In one example of the technique, DNA molecules can be placed into reaction chambers, and the template molecules can be hybridized to a sequencing primer bound to a polymerase. Incorporation of one or more triphosphates into a new nucleic acid strand at the 3' end of the sequencing primer can be detected by a change in current by a chemFET. An array can have multiple chemFET sensors. In another example, single nucleic acids can be attached to beads, and the nucleic acids can be amplified on the bead, and the individual beads can be transferred to individual reaction chambers on a chemFET array, with each chamber having a

chemFET sensor, and the nucleic acids can be sequenced.

Another example of a sequencing technique that can be used in the methods of the provided invention involves using an electron microscope (Moudrianakis E. N. and Beer M. Proc Natl Acad Sci USA. 1965 March; 53:564-71). In one example of the technique, individual DNA molecules are labeled using metallic labels that are distinguishable using an electron microscope. These molecules are then stretched on a flat surface and imaged using an electron microscope to measure sequences.

Additional detection methods can utilize binding to microarrays for subsequent fluorescent or non-fluorescent detection, barcode mass detection using a mass spectrometric methods, detection of emitted radiowaves, detection of scattered light from aligned barcodes, fluorescence detection using quantitative PCR or digital PCR methods. A comparative nucleic acid hybridization array is a technique for detecting copy number variations within the patient's sample DNA. The sample DNA and a reference DNA are differently labeled using distinct fluorophores, for example, and then hybridized to numerous probes. The fluorescent intensity of the sample and reference is then measured, and the fluorescent intensity ratio is then used to calculate copy number variations. Methods of comparative genomic hybridization array are discussed in more detail in Shinawi M, Cheung SW The array CGH and its clinical applications, Drug Discovery Today 13 (17–18): 760–70. Microarray detection may not produce a FASTQ file directly, however programs are available to convert the data produced by the microarray sequencers to a FASTQ, or similar, format.

Another method of detecting DNA molecules, RNA molecules, and copy number is fluorescent in situ hybridization (FISH). In Situ Hybridization Protocols (Ian Darby ed., 2000). FISH is a molecular cytogenetic technique that detects specific chromosomal rearrangements such as mutations in a DNA sequence and copy number variances. A DNA molecule is chemically denatured and separated into two strands. A single stranded probe is then incubated with a denatured strand of the DNA. The signals stranded probe is selected depending target sequence portion and has a high affinity to the complementary sequence portion. Probes may include a repetitive sequence probe, a whole chromosome probe, and locus-specific probes. While incubating, the combined probe and DNA strand are hybridized. The results are then visualized and quantified under a microscope in order to assess any variations.

In another embodiment, a MassARRAY™-based gene expression profiling method is used to measure gene expression. In the MassARRAY™-based gene expression profiling method, developed by Sequenom, Inc. (San Diego, Calif.) following the isolation of RNA and reverse transcription, the obtained cDNA is spiked with a synthetic DNA molecule (competitor), which matches the targeted cDNA region in all positions, except a single base, and serves as an internal standard. The cDNA/competitor mixture is PCR amplified and is subjected to a post-PCR shrimp alkaline phosphatase (SAP) enzyme treatment, which results in the dephosphorylation of the remaining nucleotides. After inactivation of the alkaline phosphatase, the PCR products from the competitor and cDNA are subjected to primer extension, which generates distinct mass signals for the competitor- and cDNA-derives PCR products. After purification, these products are dispensed on a chip array, which is pre-loaded with components needed for analysis with matrix-assisted laser desorption ionization time-of-flight mass spectrometry (MALDI-TOF MS) analysis. The cDNA present in the reaction is then quantified by analyzing the ratios of the peak areas in the mass spectrum generated. For further details see, e.g. Ding and Cantor, Proc. Natl. Acad. Sci. USA 100:3059 3064 (2003).

Further PCR-based techniques include, for example, differential display (Liang and Pardee, Science 257:967 971 (1992)); amplified fragment length polymorphism (iAFLP) (Kawamoto et al., Genome Res. 12:1305 1312 (1999)); BeadArray™ technology (Illumina, San Diego, Calif.; Oliphant et al., Discovery of Markers for Disease (Supplement to Biotechniques), June 2002; Ferguson et al., Analytical Chemistry 72:5618 (2000)); Beads Array for Detection of Gene Expression (BADGE), using the commercially available Luminex100 LabMAP system and multiple color-coded microspheres (Luminex Corp., Austin, Tex.) in a rapid assay for gene expression (Yang et al., Genome Res. 11:1888 1898 (2001)); and high coverage expression profiling (HiCEP) analysis (Fukumura et al., Nucl. Acids. Res. 31(16) e94 (2003)). The contents of each of which are incorporated by reference herein in their entirety.

In certain embodiments, variances in gene expression can also be identified, or confirmed using a microarray techniques, including nylon membrane arrays, microchip arrays and glass slide arrays, e.g., such as available commercially from Affymetrix (Santa Clara, CA). Generally, RNA samples are isolated and converted into labeled cDNA via reverse transcription. The labeled cDNA is then hybridized onto either a nylon membrane, microchip, or a glass slide with specific DNA probes from cells or tissues of interest. The hybridized cDNA is then detected and

quantified, and the resulting gene expression data may be compared to controls for analysis. The methods of labeling, hybridization, and detection vary depending on whether the microarray support is a nylon membrane, microchip, or glass slide. Nylon membrane arrays are typically hybridized with P-dNTP labeled probes. Glass slide arrays typically involve labeling with two distinct fluorescently labeled nucleotides. Methods for making microarrays and determining gene product expression (e.g., RNA or protein) are shown in Yeatman et al. (U.S. patent application number 2006/0195269), the content of which is incorporated by reference herein in its entirety.

In some embodiments, mass spectrometry (MS) analysis can be used alone or in combination with other methods (e.g., immunoassays or RNA measuring assays) to determine the presence and/or quantity of the one or more biomarkers disclosed herein in a biological sample. In some embodiments, the MS analysis includes matrix-assisted laser desorption/ionization (MALDI) time-of-flight (TOF) MS analysis, such as for example direct-spot MALDI-TOF or liquid chromatography MALDI-TOF mass spectrometry analysis. In some embodiments, the MS analysis comprises electrospray ionization (ESI) MS, such as for example liquid chromatography (LC) ESI-MS. Mass analysis can be accomplished using commercially-available spectrometers. Methods for utilizing MS analysis, including MALDI-TOF MS and ESI-MS, to detect the presence and quantity of biomarker peptides in biological samples are known in the art. See for example U.S. Pat. Nos. 6,925,389; 6,989,100; and 6,890,763 for further guidance, each of which is incorporated by reference herein in their entirety.

Protein sequences for use with the methods, sequence constructs, and systems of the invention can be determined using a number of techniques known to those skilled in the relevant art. For example, amino acid sequences and amino acid sequence reads may be produced by analyzing a protein or a portion of a protein with mass spectrometry or using Edman degradation. Mass spectrometry may include, for example, matrix-assisted laser desorption/ionization (MALDI) time-of-flight (TOF) MS analysis, such as for example direct-spot MALDI-TOF or liquid chromatography MALDI-TOF mass spectrometry analysis, electrospray ionization (ESI) MS, such as for example liquid chromatography (LC) ESI-MS, or other techniques such as MS-MS. Edman degradation analysis may be performed using commercial instruments such as the Model 49X Procise protein/peptide sequencer (Applied Biosystems/Life Technologies). The

sequenced amino acid sequences, i.e., polypeptides, i.e., proteins, may be at least 10 amino acids in length, e.g., at least 20 amino acids in length, e.g., at least 50 amino acids in length.

Incorporation by Reference

References and citations to other documents, such as patents, patent applications, patent publications, journals, books, papers, web contents, have been made throughout this disclosure. All such documents are hereby incorporated herein by reference in their entirety for all purposes.

Equivalents

Various modifications of the invention and many further embodiments thereof, in addition to those shown and described herein, will become apparent to those skilled in the art from the full contents of this document, including references to the scientific and patent literature cited herein. The subject matter herein contains important information, exemplification and guidance that can be adapted to the practice of this invention in its various embodiments and equivalents thereof.

CLAIMS

1. A method of genotyping a genetic sample, comprising:

obtaining a plurality of sequence reads corresponding to a genetic sample, wherein the plurality of sequence reads include portions of a structural variation;

aligning the plurality of sequence reads to a reference directed acyclic graph (DAG) that accounts for variability in genetic sequences within a species, the reference DAG comprising a first path corresponding to a first allele and a second path corresponding to a second allele at a position in the reference DAG, wherein the first allele comprises a genetic structural variation, wherein the plurality of sequence reads that include portions of the genetic structural variation properly align to the reference DAG; and

determining a genotype for the genetic sample based upon the number of sequence reads aligning to the first path and the second path, wherein the alignment of two or more related reads is used to determine the genotype, and the determined genotype comprises the structural variation.

2. The method of claim 1, wherein the method does not include comparing the sequence reads to a variant call format (VCF) file or a single nucleotide polymorphism database (dbSNP).

3. The method of claim 1, wherein the reference DAG further comprises a third path corresponding to a third allele at the position in the reference DAG.

4. The method of claim 1, further comprising determining an allele frequency for the sample based upon the number of sequence reads that align to the first path and the second path .

5. The method of claim 1, further comprising determining a confidence value for the genotype based upon the number of overlapping base pairs between a sequence read and one of the first path and the second path, wherein more overlap correlates with greater confidence.

6. The method of claim 1, wherein the reference DAG represents a chromosome.

7. The method of claim 1, wherein the reference DAG represents a genome.

8. The method of claim 1, wherein the first path and the second path differ from each other by a base deletion, a base insertion, or a polymorphism.

9. The method of claim 1, wherein the sequence reads are obtained by sequencing a genetic sample of a subject with a method selected from Sanger sequencing, pyrosequencing, ion semiconductor sequencing, sequencing by synthesis, sequencing by ligation, and single-molecule real-time sequencing.

10. The method of claim 1, wherein the reference DAG construct comprises greater than about 1,000 base pairs.

11. The method of clam 10, wherein the reference DAG construct comprises greater than about 1,000,000 symbols.

12. The method of claim 1, wherein the plurality of sequence reads comprises greater than about 1000 sequence reads.

13. The method of claim 1, wherein the majority of the plurality of sequence reads are greater than about 50 base pairs in length.

14. The method of claim 13, wherein the majority of the plurality of sequence reads are greater than about 100 base pairs in length.

15. The method of claim 1, wherein the genetic sample originates in blood, urine, saliva, sputum, stool, nipple aspirate, sweat, hair follicle, buccal swab, or tissue.

16. The method of claim 15, further comprising isolating a plurality of nucleic acids from the sample.

17. The method of claim 1, further comprising diagnosing a disease based upon the genotype of the sample.

18. A computer-implemented method of genotyping a genetic sample, comprising:

providing a plurality of sequence reads corresponding to a genetic sample, wherein the plurality of sequence reads include portions of a structural variation;

aligning the plurality of sequence reads to a reference directed acyclic graph (DAG) that accounts for variability in genetic sequences within a species, the reference DAG comprising nodes, wherein the reference DAG comprises a first path corresponding to a first allele and a second path corresponding to a second allele at a position in the reference DAG using a computer processor, wherein the first allele comprises a genetic structural variation, wherein the plurality of sequence reads that include portions of the genetic structural variation properly align to the reference DAG; and

determining a genotype for the genetic sample based upon the number of sequence reads aligning to the first path and the second path, wherein the alignment of two or more related reads is used to determine the genotype, and the determined genotype comprises the structural variation.

19. The method of claim 18, wherein the plurality of sequence reads are provided in a non-transitory computer-readable medium.

20. The method of claim 18, wherein the reference DAG is provided in in a non-transitory computer-readable medium.

21. A system for aligning a plurality of reads comprising a processor and memory, wherein the memory comprises instructions that, when executed, cause the processor to:

obtain a plurality of sequence reads corresponding to a genetic sample, wherein the plurality of sequence reads include portions of a structural variation;

align the plurality of sequence reads to a reference directed acyclic graph (DAG) that accounts for variability in genetic sequences within a species, wherein the reference DAG comprises a first path corresponding to a first allele and a second path corresponding to a second allele at a position in the reference DAG, wherein the first allele comprises a genetic structural variation, wherein the plurality of sequence reads that include portions of the genetic structural variation properly align to the reference DAG; and

determine a genotype for the genetic sample based upon the number of sequence reads aligning to the first path and the second path, wherein the alignment of two or more related reads is used to determine the genotype, and the determined genotype comprises the structural variation.

22. The system of claim 21, wherein aligning comprises:

simultaneously comparing a string of symbols corresponding to a sequence read to a plurality of different genotypes at the position,

scoring overlaps between the string of symbols and each of the first path and the second path, wherein a higher score corresponds to a greater amount of overlap; and

identifying the overlaps corresponding to the highest score for the sequence read.

23. The system of claim 22, further comprising writing a file to memory corresponding to the identified genotype.

24. The system of claim 21, wherein the plurality of sequence reads are provided in a non-transitory computer-readable medium.

25. The system of claim 21, wherein the reference DAG is provided in in a non-transitory computer-readable medium.

26. The system of claim 21, wherein the system comprises a plurality of processors, and wherein each processor is configured to align a portion of the plurality of sequence reads to the reference DAG.

27. The system of claim 21, wherein the instructions additionally cause the processor to output the determined genotype for the sample.