

基因测序数据处理方法和基因测序数据处理装置

本申请要求于 2020 年 10 月 22 日提交中国专利局、申请号为 202011139823.4、发明名称为“基因测序数据处理方法和基因测序数据处理装置”的中国专利申请的优先权，其全部内容通过引用结合在本申请中。

5 技术领域

本发明涉及数据处理技术领域，具体涉及一种基因测序数据处理方法和基因测序数据处理装置。

背景技术

10 随着基因测序技术的不断发展，该方法被广泛地应用于新物种、病毒以及疾病的研发分析中；与此同时大量的基因测序数据大量涌出，如何高效地完成对这些数据进行分析处理就显得尤为重要。

目前的基因分析流程中，绝大部分步骤（例如基因比对过程）都只能运行的在 x86 构架上面，例如传统的比对算法 bwa 使用的是 bwt 算法、
15 非精确比对算法的 Smith-Waterman 算法也是基于 x86 架构的 SSE2 指令实现的。

虽然基于 x86 实现的 BWT 比对算法在 x86 构架的 CPU 上面运行速度比较快，但无法大批量同时计算，并且 BWT 算法因无法适应 GPU 的 SIMT 的运行模式，导致了 BWT 在 GPU 运行的效率大大降低，从而影响整个比
20 对过程的效率。同样的，现有的 Smith-Waterman 算法仅运行在 x86 架构上面，在 ARM 平台中缺少了 SSE2 加速的支持，运行速度较慢；而且该算法也同样不适合在 GPU 架构上进行运算。

发明内容

25 有鉴于此，本发明提供了一种基因测序数据处理装置和基因测序数据处理方法，以解决现有的基因测序数据分析处理流程步骤只能运行在 x86 构架且上且在 GPU 上运行速度慢，从而造成基因测序数据处理过程效率低的问题。

-2-

本发明实施例中提供了一种基因测序数据处理方法，所述方法应用于基因测序数据处理装置，其中所述基因测序数据处理装置为异构多核构架，包括：ARM 构架、GPU 构架以及 PCI 总线；所述 ARM 构架通过所述 PCI 总线连接所述 GPU 构架；所述 ARM 构架包括至少一个 CPU 模块；所述 GPU 构架包括至少一个 GPU 模块；所述方法包括以下步骤：

步骤 S1：空闲状态的所述 CPU 模块分批读取基因测序数据得到分批基因测序数据；

步骤 S2：空闲状态的所述 CPU 模块对基因分析方法进行分切得到第一算法和第二算法；

10 步骤 S3：空闲状态的所述 CPU 模块根据所述第一算法对所述分批基因测序数据进行切分得到各短序列，并把各所述短序列和所述第二算法发送至空闲状态的所述 GPU 模块；

步骤 S4：空闲状态的所述 GPU 模块根据所述第二算法对各所述短序列进行计算，并把计算结果发送至空闲状态的所述 CPU 模块；

15 步骤 S5：空闲状态的所述 CPU 模块根据所述计算结果和所述第一算法计算得到分批处理结果；

重复步骤 S1~S5，直至将所述基因测序数据处理完成，空闲状态的所述 CPU 模块将各所述分批处理结果进行整合运算，得到最终处理结果。

可选地，

20 空闲状态的所述 CPU 模块扫描各所述 GPU 模块，确定空闲状态的 GPU 模块数量以及各空闲状态的 GPU 模块的数据处理量，并根据所述空闲状态的 GPU 模块数量以及各所述数据处理量分批读取基因测序数据。

可选地，

所述基因分析算法包括基因比对算法、Dotplot 算法、blast 算法、PAM
25 算法、HMM 算法以及 AI 推断算法。

可选地，

所述基因比对算法包括 BWT 算法，所述第一算法包括锚点切割算法；

空闲状态的所述 CPU 模块将所述分批基因测序数据采用锚点切割算法进行锚点定点，并以所述锚点定点为中心分别向前后延伸 N 个 bp 长度，

-3-

并采用 NEON 指令对所述分批基因测序数据进行 $2N+1$ 个 bp 长度的切割, 得到各所述短序列, 其中 N 为任意正整数。

可选地,

在得到各所述短序列的步骤中, 包括: 采用以下公式计算得到各所述

5 短序列:

$$(2 * N + 1) * x < L$$

其中, x 表示锚点个数, N 表示延伸的 bp 数量, L 表示所述分批基因测序数据的长度。

可选地,

10 所述第二算法为 Hash 算法; 空闲状态的所述 GPU 模块根据所述 Hash 算法对各所述短序列进行 Hash 运算, 得到 Hash 计算结果, 并将所述 Hash 计算结果发送至空闲状态的所述 CPU 模块; 其中所述 Hash 计算结果为 BWT 算法矩阵的值, 用于 BWT 算法矩阵的计算。

可选地,

15 所述第一算法还包括 BWT 矩阵变换算法;

空闲状态的所述 CPU 模块采用所述 BWT 矩阵变换算法对所述 BWT 算法矩阵进行变换, 得到所述短序列的 BWT 变换结果。

可选地,

20 所述比对算法包括 Smith-Waterman 算法, 所述第二算法包括打分矩阵算法;

空闲状态的所述 GPU 模块根据所述打分矩阵算法、各所述短序列以及参考物种序列计算 Smith-Waterman 打分矩阵, 并将所述 Smith-Waterman 打分矩阵发送至空闲状态的所述 CPU 模块。

可选地,

25 在计算 Smith-Waterman 打分矩阵的步骤中, 包括:

采用以下公式计算 Smith-Waterman 打分矩阵:

$$M=R*C$$

$$R=a*L^2+b$$

其中, M 表示 Smith-Waterman 打分矩阵, R 为参考物种备选区间序列

的长度, C 表示对从空闲状态的所述 CPU 模块接收的到各短序列进行筛选拼接形成的短序列的长度, L 表示表示所述分批基因测序数据的长度, a 和 b 表示常数。

5 本发明实施例中提供了一种基因测序数据处理装置, 所述基因测序数据处理装置为异构多核构架, 所述基因测序数据处理装置执行所述的基因测序数据处理方法。

本发明实施例中的基因测序数据处理装置和基因测序数据处理方法, 方法应用于装置, 基因测序数据处理装置为异构多核构架, 包括 ARM 构架、GPU 构架和 PCI 总线, 其中 ARM 构架包括至少一个 CPU 模块, 而 GPU 构架包括至少一个 GPU 模块, CPU 模块通过 PCI 总线与 GPU 模块连接, 这两者之间可以进行信息的相互传输。其中方法包括空闲状态的 CPU 模块主要用于分批读取基因测序数据和对基因分析方法进行分切, 从而得到分批基因测序数据、第一算法 (该算法是 CPU 模块最适合运行的算法) 10 和第二算法 (该算法是 GPU 模块最适合运行的算法), 然后采用第一算法对分批基因测序数据进行切分, 得到一些列短序列, 并将这些短序列和第二算法通过 PCI 总线传输至处于空闲状态的 GPU 模块; GPU 模块就根据第二算法对这些短序列进行计算, 然后把计算结果返回至空闲状态的 CPU 模块; 空闲状态的 CPU 模块就根据计算结果和第一算法计算得到一个分配处理结果; 空闲状态的 CPU 模块和空闲状态的 GPU 模块重复执行上述步 15 骤, 直至将基因测序数据处理完成, 然后空闲状态的 CPU 模块将每个分批处理结果进行整合, 即可得到最终处理结果。该基因测序数据处理装置和基因测序数据处理方法将基因测序数据的分析方法 (即分析过程) 分割开来, 让其按照特性分别运行在 CPU 模块和 GPU 模块, 大大提高了基因测序数据分析的效率。另外, 该基因测序数据处理装置中可以设置多个 CPU 20 模块和 GPU 模块, 且多个 GPU 模块可以同时计算不同长度短序列, 可以解决 GPU 并行效率低的问题。

附图说明

为了更清楚地说明本发明实施例或现有技术中的技术方案, 下面将对

-5-

实施例或现有技术描述中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图仅仅是本发明的实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据提供的附图获得其他的附图。

图1为本发明实施例中的基因测序数据处理装置的结构示意图；

5 图2为本发明实施例中基因测序数据处理装置数据处理过程示意图；

图3为本发明实施例中CPU模块对分批基因测序数据进行锚点切割的示意图；

图4为本发明实施例中GPU模块采用Hash算法对短序列进行Hash运算的示意图；

10 图5为本发明实施例中基因测序数据处理方法流程图示意图。

具体实施方式

下面将对本发明实施例中的技术方案进行清楚、完整地描述，显然，所描述的实施例仅仅是本发明一部分实施例，而不是全部的实施例。基于
15 本发明中的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都属于本发明保护的范围。

名词解释：

基因（Gene，Mendelian factor）是指携带有遗传信息的DNA或RNA序列（即基因是具有遗传效应的DNA或RNA片段），也称为遗传因子，是控制
20 性状的基本遗传单位。基因通过指导蛋白质的合成来表达自己所携带的遗传信息，从而控制生物个体的性状表现。

基因测序是一种新型基因检测技术，从血液或唾液中分析测定基因全序列，从而预测罹患多种疾病的可能性、个体的行为特征及行为合理等。

短序列（read）：是一小段短的测序片段，是高通量测序仪产生的测序
25 数据，对整个基因组进行测序，就会产生成百上千万的read，然后将这些read拼接起来就能获得基因组的全序列。

比对分析：NGS测序下来的短序列（read）存储于FASTQ文件里面，虽然它们原本都来自于有序的基因组，但在经过DNA建库和测序之后，文件中不同read之间的前后顺序关系就已经全部丢失了。因此，FASTQ文件

-6-

中紧挨着的两条read之间没有任何位置关系，它们都是随机来自于原本基因组中某个位置的短序列而已。因此，我们需要先把这一大堆的短序列捋顺，一个个去跟该物种的参考基因组比较，找到每一条read在参考基因组上的位置，然后按顺序排列好，这个过程就称为测序数据的比对。

- 5 比对算法：序列比对的计算方法一般分为两类：全局性比对（global alignments）和局部比对（local alignments）。计算一个全局性的路线，是一个全局优化的形式，其强制按照整个长度的所有查询序列对齐。与此相反，局部比对只确定局部的相似而整个长序列却往往大相径庭。局部比对往往是可取的，但可能更难以计算，因为还有来自确定其他相似区域的挑战。
- 10 各种计算算法已应用于序列比对问题中，包括缓慢但正规的像动态规划的优化方法、高效率但不彻底的启发式算法，或大型数据库搜索设计的概率方法。

- ARM: ARM架构，高级精简指令集机器（Advanced RISC Machine，更早称作Acorn精简指令集机器，Acorn RISC Machine），是一个精简指令集（RISC）处理器架构家族，其广泛地使用在许多嵌入式系统设计中。由于节能的特点，在其他领域上也有很多作为。ARM处理器非常适用于移动通信领域，符合其主要设计目标为低成本、高性能、低功耗的特性。另一方面，超级计算机消耗大量电能，ARM同样被视作更高效地选择。安谋控股（ARM Holdings）开发此架构并授权其他公司使用，以供他们实现ARM的某一个架构，开发自主的系统单片机和系统模块（system-on-module，SoC）。
- 15 GPU: 图形处理器（Graphics Processing Unit，缩写：GPU；又称显示核心、视觉处理器、显示装置或绘图装置）是一种专门在个人电脑、工作站、游戏机和一些移动设备（如平板电脑、智能手机等）上运行绘图运算工作的微处理器。图形处理器使显卡减少对中央处理器（CPU）的依赖，并分担部分原本是由中央处理器所担当的工作，尤其是在进行三维绘图运算时，功效更加明显。
- 20 CUDA:（Compute Unified Device Architecture，统一计算架构）是由NVIDIA所推出的一种集成技术，是该公司对于GPGPU的正式名称。透过

这个技术，用户可利用NVIDIA的GeForce 8以后的GPU和较新的Quadro GPU进行计算。亦是首次可以利用GPU作为C-编译器的开发环境。NVIDIA营销的时候，往往将编译器与架构混合推广，造成混乱。实际上，CUDA可以兼容OpenCL或者自家的C-编译器。无论是CUDA C-语言或是OpenCL，
5 指令最终都会被驱动程序转换成PTX代码，交由显示核心计算。

BWT: (Burrows-Wheeler Transform, 简称BWT, 也称作块排序压缩), 是一个被应用在数据压缩技术(如bzip2)中的算法。该算法于1994年被Michael Burrows和David Wheeler在位于加利福尼亚州帕洛阿尔托的DEC系统研究中心发明。它的基础是之前Wheeler在1983年发明的一种没有公开的
10 转换方法。当一个字符串用该算法转换时, 算法只改变这个字符串中字符的顺序而并不改变其字符。如果原字符串有几个出现多次的子串, 那么转换过的字符串上就会有一些连续重复的字符, 这对压缩是很有用的。该方法能使得基于处理字符串中连续重复字符的技术(如MTF变换和游程编码)的编码更容易被压缩。

15 Smith-waterman: (Smith-Waterman algorithm) 是一种进行局部序列比对(相对于全局比对)的算法, 用于找出两个核苷酸序列或蛋白质序列之间的相似区域。该算法的目的不是进行全序列的比对, 而是找出两个序列中具有高相似度的片段。

HASH: 又称散列算法、哈希函数, 是一种从任何一种数据中创建小的数字“指纹”的方法。散列函数把消息或数据压缩成摘要, 使得数据量变小, 将数据的格式固定下来。该函数将数据打乱混合, 重新创建一个叫做散列值(hash values, hash codes, hash sums, 或hashes)的指纹。散列值通常
20 用一个短的随机字母和数字组成的字符串来代表。好的散列函数在输入域中很少出现散列冲突。在散列表和数据处理中, 不抑制冲突来区别数据, 25 会使得数据库记录更难找到。

SSE2: (Streaming SIMD Extensions 2), 是一种IA-32架构的SIMD(单一指令多重数据)指令集。SSE2是在2001年随着Intel发表第一代Pentium 4处理器也一并推出的指令集。它延伸较早的SSE指令集, 而且可以完全取代MMX指令集。

为了更详细说明本发明，下面结合附图对本发明提供的一种基因测序数据处理装置和基因测序数据处理方法，进行具体地描述。

图1为基因测序数据处理装置的结构示意图。如图1所示，一种基因测序数据处理装置，基因测序数据处理装置为异构多核构架，包括：ARM构架10、GPU构架20以及PCI总线30；ARM构架10通过PCI总线30连接GPU构架20；ARM构架10包括至少一个CPU模块；GPU构架30包括至少一个GPU模块；空闲状态的CPU模块用于分批读取基因测序数据得到分批基因测序数据，以及对基因分析方法进行分切得到第一算法和第二算法；根据第一算法对分批基因测序数据进行切分得到各短序列，并把各短序列和第二算法发送至空闲状态的GPU模块；空闲状态的GPU模块用于根据第二算法对各短序列进行计算，并把计算结果发送至空闲状态的CPU模块；空闲状态的CPU模块还用于根据计算结果和第一算法计算得到分批处理结果；空闲状态的CPU模块和空闲状态的GPU模块重复执行上述步骤，直至将基因测序数据处理完成，然后空闲状态的CPU模块并将各分批处理结果进行整合运算，得到最终处理结果。

具体地，基因测序数据处理装置为异构多核构架，即是ARM+GPU构架，其中ARM构架10包括CPU模块，GPU构架20包括GPU模块；CPU模块和GPU模块的数量不是固定的，可以根据实际的运算情况进行设置，例如根据基因测序数据的数量、CPU模块性能、GPU模块性能（例如GPU的显存、CUDA核心数、CUDA核心频率）、基因分析采用的算法复杂度等进行确定。

其中，每个CPU模块内部的核心（core）的处理或计算能力可以是相同的也可以不同的。同样每个GPU模块的处理或计算能力也可以是相同或不同的。可选地，GPU模块可以是GPU计算卡，其中GPU计算卡通常是采用SIMT架构。

在一种可选地实施方式中，CPU模块采用了NENO加速技术；采用该加速技术可以进一步提高CPU模块运行速度。

在一个可选地实施例方式中，基因测序数据装置可以使用NVIDIA发布的Jetson nano TX1，该设备使用的是Maxwell架构的GPU，有128个Cuda核

心、运算能力472G，同时Jetson-nano还有一个4核心A57处理器作为ARM CPU核心运算器。

基因分析方法是指对基因测序数据进行分析处理过程中使用的方法，其中包括序列对比、基因集富集分析（包括GO分析、KEGG分析）以及基因调控网络分析等。

在将基因分析方法进行切分得到的第一算法和第二算法时，主要是根据某个基因分析方法的特性进行分割，即就是把适合CPU模块处理的算法从该基因分析方法中分割出来形成第一算法；把适合GPU模块处理的算法也从该基因分析方法中分割出来形成第二算法；由此可见，第一算法和第二算法可以是该基因分析方法中一部分，可以由一个或多个小步骤组成，其中在分割过程中没有严格的算法规则，即只要符合分割原理即可。其中，分割原理主要包括：第一算法是通常需要大量逻辑判断、计算结果之间有依赖性，例如第二步计算依赖或以第一步计算结果作为基础、涉及到是或否的判断等；而第二算法通常是多个数据可以同时运行计算，且每个数据之间不涉及到逻辑判断或这个数据之间没有依赖性。

应当理解，本实施例中的“第一”、“第二”并不是对算法的限定，仅只是为了将这两者进行区分。

此外，由于ARM构架10中通常有多个CPU模块，每个CPU模块的运行或工作状态可能不相同，即有的CPU模块处于运行状态，而有的则处于空闲状态。同样，GPU构架20中的GPU模块也有类似的情况。因此，在本实施例中，采用空闲状态的CPU模块和GPU模块进行相应的操作，其中选择的CPU模块和GPU模块可以是所有空闲状态的模块，也可以是其中的一部分。

另外，所述的基因测序数据可以是任意物种的进行基因测序而得到的数据，其中包括DNA测序片段、RNA测序片段等。由于一次测序中会产生大量的数据，因此所述的基因测序数据的数据量会比较大，在对该数据进行分析处理时可以分批进行，从而避免数据传输的拥堵等。因此，在本实施例中，空闲状态的CPU模块分批读取基因测序数据，每次读取的基因测序数据的数量可以不相等，具体可以根据GPU模块数量以及每个GPU模块

数据处理能力以及CPU模块数据读取能力、PCI总线数据传输能力等综合考虑确定出最合适的基因测序数据的数量，从而能最大程度地保证数据处理效率最高。

在分批次读取基因测序数据后，通常还需要对分批基因测序数据进行切分，形成多个短序列。在本实施例中，采用第一算法对分批基因测序数据进行切分，其中切割成短序列长度可以是不相同的，且切割的短序列数量也并不是固定，可以根据分批基因测序数据的数量、空闲状态的GPU模块数量以及GPU处理能力综合考虑选择最合适的数值。

当空闲状态的CPU模块把各短序列以及第二算法传输至空闲状态的GPU后，空闲状态的GPU模块根据第二算法对各短序列进行计算，此时空闲状态的CPU模块可以进行下一次的分批基因测序数据读取、分割；而当空闲状态的GPU模块将短序列处理完成后，将计算结果又传输至空闲状态的CPU模块，CPU模块可以根据计算结果以及第一算法计算得到分批计算结果；由此不断重复，CPU模块与GPU模块之间形成流水线，直至将所有基因测序数据全部处理完成。

本发明实施例中的基因测序数据处理装置，其中基因测序数据处理装置为异构多核构架，包括ARM构架10、GPU构架20和PCI总线30，其中ARM构架包括至少一个CPU模块，而GPU构架包括至少一个GPU模块，CPU模块通过PCI总线与GPU模块连接，这两者之间可以进行信息的相互传输。空闲状态的CPU模块主要用于分批读取基因测序数据和对基因分析方法进行分切，从而得到分批基因测序数据、第一算法（该算法是CPU模块最适合运行的算法）和第二算法（该算法是GPU模块最适合运行的算法），然后采用第一算法对分批基因测序数据进行切分，得到一些列短序列，并将这些短序列和第二算法通过PCI总线传输至处于空闲状态的GPU模块；GPU模块就根据第二算法对这些短序列进行计算，然后把计算结果返回至空闲状态的CPU模块；空闲状态的CPU模块就根据计算结果和第一算法计算得到一个分配处理结果，空闲状态的CPU模块和空闲状态的GPU模块重复执行上述步骤，直至将基因测序数据处理完成，然后空闲状态的CPU模块将每个分批处理结果进行整合，即可得到最终处理结果。该基因测序数据处理装

置和基因测序数据处理方法将基因测序数据的分析方法（即分析过程）分割开来，让其按照特性分别运行在CPU模块和GPU模块，大大提高了基因测序数据分析的效率。另外，该基因测序数据处理装置中可以设置多个CPU模块和GPU模块，且多个GPU模块可以同时计算不同长度短序列，可以解决GPU并行效率低的问题。

在一个实施例中，空闲状态的CPU模块还用于扫描各GPU模块，确定空闲状态的GPU模块数量以及各空闲状态的GPU模块的数据处理量，并根据空闲状态的GPU模块数量以及各数据处理量分批读取基因测序数据。

具体地，空闲状态的CPU模块在启动基因分析时，可扫描GPU模块，以确定当前可用的GPU的片数，以及可用的GPU模块的数据处理量，从而来确定本次分批读取基因测序数据的数量，然后根据这个数量读取基因测序数据。

为了便于理解本方案，结合附图1和附图2，给出一个基因测序数据处理装置工作流程的详细实施例，其中在本实施例中基因分析方法采用基因比对方法为例。

1.T1时刻：空闲状态的CPU模块接收基因测序数据D，启动比对任务程序，扫描当前可用的GPU模块的片数，记为G；根据数据D的测序长度记为L1；CPU模块对数据D进行分批读取，每次一批读取数据 D_i 数量记为K，K的值可根据GPU模块数量进行调整：计算公式： $K=A \times G$ ，其中A表示GPU模块一次能处理的数据量（在本实施例中选择每个GPU模块处理能力完全相同），对数据 D_i 按照第一算法进行切分，形成多个短序列。

2.T2时刻：将这些切分好的短序列通过PCI总线传输到其中一片空闲的GPU模块中，同时CPU模块可以接着对下一批数据 D_{i+1} 进行处理，形成了2级流水线。

3.T3时刻：当 D_i 数据传输到GPU中显存中，即可启动GPU第二算法，此时 D_{i+1} 进入了PCI传输阶段，CPU模块处理下一批数据 D_{i+2} ，形成了3级流水线。

4.T4时刻： D_i 数据计算完成，计算结果通过PCI回传给CPU模块，此时 D_{i+1} 进入了GPU模块计算阶段， D_{i+2} 进入PCI输入阶段， D_{i+3} 由CPU模

块处理数据，形成了4级流水线。

5. T5时刻：Di数据计算结果回传完毕后交给CPU模块采用第一算法继续完成比对算法后续阶段的操作，此时形成了5级流水线。

在一个实施例中，基因分析算法包括基因比对算法、Dotplot算法、blast
5 算法、PAM算法、HMM算法以及AI推断算法。

具体地，Dotplot算法、blast算法是一种序列比对算法。

PAM算法是一种数据挖掘的聚类算法，可以用在单细胞测序中来对细胞亚群等进行分析。

HMM算法，隐马尔可夫聚类算法，是统计模型，它用来描述一个含有
10 隐含未知参数的马尔可夫过程，可以用在靶基因的预测中。

AI推断算法（DeepVariant），深度的学习算法，可以用来识别基因突变等。可选地，AI推断算法可以是CNN（卷积神经网络）、RNN（循环神经网络）相关的推断算法。

可选地，当基因分析算法为Dotplot算法、blast算法、PAM算法时，通常
15 需要先对算法进行CUDA化。对算法进行CUDA化使得该方法更加合适运行在本发明实施例中的基因测序数据处理装置上。

在一个实施例中，基因比对算法包括BWT算法，第一算法包括锚点切割算法；空闲状态的CPU模块还用于将分批基因测序数据采用锚点切割算法进行锚点定点，并以锚点定点为中心分别向前后延伸N个bp长度，并采用
20 用NEON指令对分批基因测序数据进行2N+1个bp长度的切割，得到各短序列，其中N为任意正整数。

在一个实施例中，在得到各短序列的步骤中，包括：采用以下公式计算得到各短序列：

$$(2 * N + 1) * x < L$$

25 其中，x表示锚点个数，N表示延伸的bp数量，L表示分批基因测序数据的长度。

可选地，基因比对算法可以是BWT算法，第一算法可以是锚点切割算法和BWT矩阵变换算法；第二算法可以是Hash算法。具体过程为：如图3所示，空闲状态的CPU模块对数据Di采用第一算法（即锚点切割算法）进

行处理；首先对分批读取的长度为L的基因测序数据（即read）进行锚点定
点，并前后延伸N个bp长度，获得长度为2N+1的短read，然后使用NEON
指令对read进行2N+1长度的切割和搬运。锚点数量为x个的情况下，N的个
数有关系如下公式：

5
$$(2 * N + 1) * x < L$$

其中，x表示锚点个数，N表示延伸的bp数量，L表示分批基因测序数
据的长度。采用上述的方式可以得到多个短序列，这些短序列适合在GPU
模块上运行。

10 在一个实施例，第二算法为Hash算法；空闲状态的GPU模块还用于
根据Hash算法对各短序列进行Hash运算，得到Hash计算结果，并将Hash计
算结果发送至空闲状态的CPU模块；其中Hash计算结果为BWT算法矩阵的
值，用于BWT算法矩阵的计算。

具体地，基因比对算法可以是BWT算法，第一算法可以是锚点切割算
法和BWT矩阵变换算法；第二算法可以是Hash算法。如图4所示，将通过
15 第一算法计算出的短序列x*K 条短序列传入空闲状态的GPU模块中的显
存中，其中K表示Di的数量，短序列的数量与多个GPU模块的总显存呈正
相关。由于Hash算法利于GPU的SIMT架构的运行，使用GPU的核函数对多
个短序列进行hash计算，得到Hash计算结果，并将Hash计算结果发送至空
闲状态的CPU模块；其中Hash计算结果为BWT算法矩阵的值，用于BWT
20 算法矩阵的计算。采用Hash算法与传统其他计算（例如kmer计算位点算法）
相比，可以大大节约内存空间。

在一个实施例，第一算法还包括BWT矩阵变换算法；空闲状态的
CPU模块还用于采用BWT矩阵变换算法对BWT算法矩阵进行变换，得到短
序列的BWT变换结果。

25 具体地，基因比对算法可以是BWT算法，第一算法可以是锚点切割算
法和BWT矩阵变换算法。在GPU模块将Hash计算结果发送至空闲状态的
CPU模块后，CPU模块将根据Hash计算结果为BWT算法矩阵的值，用于
BWT算法矩阵的计算，然后采用采用BWT矩阵变换算法对BWT算法矩阵
进行变换，得到短序列的BWT变换结果。可选地，Hash计算结果为BWT

算法矩阵之间的关系可以表示为 $h=\text{Hash}(x,r)$, $Y=\text{BWT}(h,r)$ 。其中 h 表示Hash计算结果, Y 表示BWT算法矩阵, r 表示短序列。采用该方法可以快速准确地得到短序列的BWT变换结果, 从而快速完成对基因测序数据的压缩, 更加方便后续的处理。

- 5 在一个实施例中, 比对算法包括Smith-Waterman算法, 第二算法包括打分矩阵算法; 空闲状态的GPU模块还用于根据打分矩阵算法、各短序列以及参考物种序列计算Smith-Waterman打分矩阵, 并将Smith-Waterman打分矩阵发送至空闲状态的CPU模块。

在一个实施例中, 在计算Smith-Waterman打分矩阵的步骤中, 包括:

- 10 采用以下公式计算Smith-Waterman打分矩阵: $M=R*C$, $R=a*L2+b$;

其中, M 表示Smith-Waterman打分矩阵, R 为参考物种备选区间序列的长度, C 表示对从空闲状态的所述CPU模块接收的到各短序列进行筛选拼接形成的短序列的长度, L 表示表示分批基因测序数据的长度, a 和 b 表示常数。

- 15 具体而言, 传统的Smith-Waterman算法在GPU中运行效率比较低, 无法直接在本发明实施例中的基因测序数据处理装置, 因此对Smith-Waterman算法进行改进。具体的, Smith-Waterman算法中存在一个打分矩阵, 大小为 $R*C$; 将计算打分矩阵的步骤放在GPU模块中, 那么此时第二算法为打分矩阵算法。采用以下公式计算Smith-Waterman打分矩阵:

- 20 $M=R*C$, $R=a*L2+b$;

其中, M 表示Smith-Waterman打分矩阵, R 为参考物种备选区间序列的长度, C 表示对从空闲状态的CPU模块接收的到各短序列进行筛选拼接形成的短序列的长度, L 表示表示分批基因测序数据的长度, a 和 b 表示常数。另外, C 的长度与BWT算法中GPU模块计算出的Hash计算结果有关。采用
25 该方式可以对传统的Smith-Waterman算法进行改进, 使其适合于在GPU中运行, 且运行效率高。

根据上述的基因测序数据处理装置, 本发明实施例中还提供了一种基因测序数据处理方法。

如图5所示, 一种基因测序数据处理方法, 方法应用于基因测序数据处

理装置，包括以下步骤：

步骤S1：空闲状态的CPU模块分批读取基因测序数据得到分批基因测序数据；

5 步骤S2：空闲状态的CPU模块对基因分析方法进行分切得到第一算法和第二算法；

步骤S3：空闲状态的CPU模块根据第一算法对分批基因测序数据进行切分得到各短序列，并把各短序列和第二算法发送至空闲状态的GPU模块；

步骤S4：空闲状态的GPU模块根据第二算法对各短序列进行计算，并把计算结果发送至空闲状态的CPU模块；

10 步骤S5：空闲状态的CPU模块根据计算结果和第一算法计算得到分批处理结果；

重复步骤S1~S5，直至将基因测序数据处理完成，空闲状态的CPU模块将各分批处理结果进行整合运算，得到最终处理结果。

具体地，由于一次测序中会产生大量的数据，因此所述的基因测序数据的数据量会比较大，在对该数据进行分析处理时可以分批进行，从而避免数据传输的拥堵等。其中把空闲状态的CPU第i批读取的基因测序数据记为 D_i 。空闲状态的CPU模块读取基因测序数据 D_i ，以及对基因分析方法进行分切得到第一算法和第二算法；根据第一算法对基因测序数据 D_i 进行切分得到各短序列，并把各短序列和第二算法发送至空闲状态的GPU模块，
15 然后空闲状态的GPU模块根据第二算法对各短序列进行计算，并把计算结果发送至空闲状态的CPU模块；然后空闲状态的CPU模块根据计算结果和第一算法计算得到分批处理结果；此外，空闲状态的CPU模块读取基因测序数据 D_{i+1} ，对基因测序数据 D_{i+1} 进行分切，将分切后的基因测序数据 D_{i+1} 对应的短序列发送至空闲状态的GPU模块， D_{i+1} 表示第i+1批读取的基因
20 测序数据；空闲状态的GPU模块分切后的基因测序数据 D_{i+1} 对应的短序列进行处理，再把处理结果发送至空闲状态的CPU模块；空闲状态的CPU模块和处于空闲状态的GPU模块不断地进行基因测序数据读取、切分、传输、计算和回传（即不断重复步骤S1-S5），直到将所有的基因测序数据处理完毕，在该过程中空闲状态的CPU模块和空闲状态的GPU模块之间形成
25

流水线。

在一个实施例中，空闲状态的CPU模块扫描各GPU模块，确定空闲状态的GPU模块数量以及各空闲状态的GPU模块的数据处理量，并根据空闲状态的GPU模块数量以及各数据处理量分批读取基因测序数据。

5 在一个实施例中，基因分析算法包括基因比对算法、Dotplot算法、blast算法、PAM算法、HMM算法以及AI推断算法。

 在一个实施例中，基因比对算法包括BWT算法，第一算法包括锚点切割算法；空闲状态的CPU模块将分批基因测序数据采用锚点切割算法进行锚点定点，并以锚点定点为中心分别向前后延伸N个bp长度，并采用NEON
10 指令对分批基因测序数据进行2N+1个bp长度的切割，得到各短序列，其中N为任意正整数。

 在一个实施例中，在得到各短序列的步骤中，包括：采用以下公式计算得到各短序列：

$$(2 * N + 1) * x < L$$

15 其中，x表示锚点个数，N表示延伸的bp数量，L表示分批基因测序数据的长度。

 在一个实施例中，第二算法为Hash算法；空闲状态的GPU模块还用于根据Hash算法对各短序列进行Hash运算，得到Hash计算结果，并将Hash计算结果发送至空闲状态的CPU模块；其中Hash为BWT算法矩阵的值，用于
20 BWT算法矩阵的计算。

 在一个实施例中，第一算法还包括BWT矩阵变换算法；空闲状态的CPU模块采用BWT矩阵变换算法对BWT算法矩阵进行变换，得到短序列的BWT变换结果。

 在一个实施例中，比对算法包括Smith-Waterman算法，第二算法包括
25 打分矩阵算法；空闲状态的GPU模块还用于根据打分矩阵算法、各短序列以及参考物种序列计算Smith-Waterman打分矩阵，并将Smith-Waterman打分矩阵发送至空闲状态的CPU模块。

 在一个实施例中，在计算Smith-Waterman打分矩阵的步骤中，包括：采用以下公式计算Smith-Waterman打分矩阵： $M=R*C$ ， $R=a*L2+b$ ；其中，

- 17 -

M表示Smith-Waterman打分矩阵，R为参考物种备选区间序列的长度，C表示对从空闲状态的CPU模块接收的到各短序列进行筛选拼接形成的短序列的长度，L表示表示分批基因测序数据的长度，a和b表示常数。

关于基因测序数据处理方法的具体限定可以参见上文中对于基因测序
5 数据处理装置的限定，在此不再赘述。

以上所述仅是本发明的优选实施方式，应当指出，对于本技术领域的普通技术人员来说，在不脱离本发明原理的前提下，还可以做出若干改进和润饰，这些改进和润饰也应视为本发明的保护范围。

权 利 要 求

1、一种基因测序数据处理方法，所述方法应用于基因测序数据处理装置，其特征在于，其中所述基因测序数据处理装置为异构多核构架，包括：ARM 构架、GPU 构架以及 PCI 总线；所述 ARM 构架通过所述 PCI 总线
5 连接所述 GPU 构架；所述 ARM 构架包括至少一个 CPU 模块；所述 GPU 构架包括至少一个 GPU 模块；所述方法包括以下步骤：

步骤 S1：空闲状态的所述 CPU 模块分批读取基因测序数据得到分批基因测序数据；

步骤 S2：空闲状态的所述 CPU 模块对基因分析方法进行分切得到第
10 一算法和第二算法；

步骤 S3：空闲状态的所述 CPU 模块根据所述第一算法对所述分批基因测序数据进行切分得到各短序列，并把各所述短序列和所述第二算法发送至空闲状态的所述 GPU 模块；

步骤 S4：空闲状态的所述 GPU 模块根据所述第二算法对各所述短序列进行计算，并把计算结果发送至空闲状态的所述 CPU 模块；
15

步骤 S5：空闲状态的所述 CPU 模块根据所述计算结果和所述第一算法计算得到分批处理结果；

重复步骤 S1~S5，直至将所述基因测序数据处理完成，空闲状态的所述 CPU 模块将各所述分批处理结果进行整合运算，得到最终处理结果。

20 2、根据权利要求 1 所述的基因测序数据处理方法，其特征在于，空闲状态的所述 CPU 模块扫描各所述 GPU 模块，确定空闲状态的 GPU 模块数量以及各空闲状态的 GPU 模块的数据处理量，并根据所述空闲状态的 GPU 模块数量以及各所述数据处理量分批读取基因测序数据。

3、根据权利要求 1 所述的基因测序数据处理方法，其特征在于，所述
25 基因分析算法包括基因比对算法、Dotplot 算法、blast 算法、PAM 算法、HMM 算法以及 AI 推断算法。

4、根据权利要求 3 所述的基因测序数据处理方法，其特征在于，所述基因比对算法包括 BWT 算法，所述第一算法包括锚点切割算法；

空闲状态的所述 CPU 模块将所述分批基因测序数据采用锚点切割算

法进行锚点定点，并以所述锚点定点为中心分别向前后延伸 N 个 bp 长度，并采用 NEON 指令对所述分批基因测序数据进行 2N+1 个 bp 长度的切割，得到各所述短序列，其中 N 为任意正整数。

5 5、根据权利要求 4 所述的基因测序数据处理方法，其特征在于，在得到各所述短序列的步骤中，包括：

采用以下公式计算得到各所述短序列：

$$(2 * N + 1) * x < L$$

其中，x 表示锚点个数，N 表示延伸的 bp 数量，L 表示所述分批基因测序数据的长度。

10 6、根据权利要求 3 或 4 所述的基因测序数据处理方法，其特征在于，所述第二算法为 Hash 算法；

空闲状态的所述 GPU 模块根据所述 Hash 算法对各所述短序列进行 Hash 运算，得到 Hash 计算结果，并将所述 Hash 计算结果发送至空闲状态的所述 CPU 模块；其中所述 Hash 计算结果为 BWT 算法矩阵的值，用于

15 BWT 算法矩阵的计算。

7、根据权利要求 6 所述的基因测序数据处理方法，其特征在于，所述第一算法还包括 BWT 矩阵变换算法；

空闲状态的所述 CPU 模块采用所述 BWT 矩阵变换算法对所述 BWT 算法矩阵进行变换，得到所述短序列的 BWT 变换结果。

20 8、根据权利要求 3 所述的基因测序数据处理方法，其特征在于，所述比对算法包括 Smith-Waterman 算法，所述第二算法包括打分矩阵算法；

空闲状态的所述 GPU 模块根据所述打分矩阵算法、各所述短序列以及参考物种序列计算 Smith-Waterman 打分矩阵，并将所述 Smith-Waterman 打分矩阵发送至空闲状态的所述 CPU 模块。

25 9、根据权利要求 8 所述的基因测序数据处理方法，其特征在于，在计算 Smith-Waterman 打分矩阵的步骤中，包括：

采用以下公式计算 Smith-Waterman 打分矩阵：

$$M=R*C$$

$$R=a*L^2+b$$

— 20 —

其中, M 表示 Smith-Waterman 打分矩阵, R 为参考物种备选区间序列的长度, C 表示对从空闲状态的所述 CPU 模块接收的到各短序列进行筛选拼接形成的

短序列的长度, L 表示表示所述分批基因测序数据的长度, a 和 b 表示
5 常数。

10、一种基因测序数据处理装置, 其特征在于, 所述基因测序数据处理装置执行权利要求 1-9 任一项所述的基因测序数据处理方法。