广西科学院学报
Journal of Guangxi Academy of Sciences
2017,33(1):7~11
Vol. 33, No. 1 February 2017

网络优先数字出版时间:2017-02-28 DOI:10.13657/j.cnki.gxkxyxb.20170228.001
网络优先数字出版地址:http://kns.cnki.net/kcms/detail/45.1075.N.20170228.1016.002.html

基于 K-spectrum 的下一代测序数据的纠错算法分析*
K-spectrum-based Analysis for Error Correction of Next Generation Sequencing

赖德焕[1],陈庆锋[1,2**],黄丽宇[3],梁家海[4]
LAI Dehuan[1],CHEN Qingfeng[1,2],HUANG Liyu[3],LIANG Jiahai[4]

(1.广西大学计算机与电子信息学院,广西南宁 530004;2.广西大学亚热带农业生物资源保护与利用国家重点实验室,广西南宁 530004;3.广西大学信息网络中心,广西南宁 530004;4.钦州学院电子与信息工程学院,广西钦州 535000)
(1. School of Computer, Electronic and Information in Guangxi University, Nanning, Guangxi, 530004, China; 2. State Key Laboratory for Conservation and Utilization of Subtropical Agro-bioresources, Guangxi University, Nanning, Guangxi, 530004, China; 3. Information Network Center, Guangxi University, Nanning, Guangxi, 530004, China; 4. School of Electronics and Information Engineering, Qinzhou University, Qinzhou, Guangxi, 535000, China)

摘要:【目的】对现有的下一代测序(Next Generation Sequencing,NGS)纠错算法和工具进行分析,提出基于Hadoop平台的纠错算法,以解决大数据处理中计算机内存不足和运行时间长的问题,提升纠错性能。**【方法】**使用特定的数据对现有的基于K-spectrum的纠错算法进行测试,对各纠错工具的运行时间、内存峰值和纠错结果进行比较来衡量纠错工具的性能。在此基础上提出Hadoop分布式并行纠错算法(Parallel algorithm),并与串行程序、Lighter和Racer进行比较,分析分布式并行实现的可行性。**【结果】**现有的基于K-spectrum的纠错工具普遍存在较大的内存消耗现象,其中Racer和Sga的纠错效果较好。而Hadoop分布式并行纠错算法对计算机单机内存的消耗较低,当数据量超过一定值时,并行分布式程序的运算时间比串行单机程序明显减少。**【结论】**本研究提出的Hadoop分布式并行纠错算法不仅降低了内存消耗,而且提高了运算性能,更有利于大规模基因数据的分析处理。

关键词:NGS 基因错误修正 Hadoop K-spectrum
中图分类号:TP301.6 **文献标识码:**A **文章编号:**1002-7378(2017)01-0007-05

Abstract:【Objective】The existing Next Generation Sequencing (NGS) error correction algorithms and tools are analyzed and summarized, and an error correction tool based on Hadoop platform is proposed to solve the problem of insufficient memory and long running time in large data processing. **【Methods】**The existing K-spectrum-based error correction algorithm is tested with the specific data, and the performance of the error correction software is measured by comparing the run time, peak memory and error correction result of each correction tool. A new error correction algorithm is designed by combining Hadoop parallel distributed program and the algorithm proposed in this paper. A comparison is made between the serial program, Lighter and Racer to analyze the feasibility of distributed parallel program. **【Results】**The existing error correction tools based on K-spectrum method generally have large

收稿日期:2016-12-20
作者简介:赖德焕(1993-),男,硕士研究生,主要从事生物信息学和数据挖掘研究。
*国家自然科学基金项目(61363025)和广西自然科学基金重点项目(2013GXNSFDA019029)资助。
**通信作者:陈庆锋(1972-),男,教授,主要从事数据挖掘和生物信息学研究,E-mail:qingfeng@gxu.edu.cn。

memory footprint, in which Racer and Sga have better error correction effect. And Hadoop distributed parallel error correction algorithm shows lower memory consumption on single computer. When the data size exceeds a certain value, comparing with the time of the serial single program, the parallel and distributed computing time significantly reduces. **【Conclusion】**The parallel error correction program combined with Hadoop improves the memory and operation performance of the NGS error correction program based on K-spectrum, which is good for the analysis and processing of large scale gene data.

Key words: NGS, gene error correction, Hadoop, K-spectrum

0 引言

【研究意义】下一代测序(Next Generation Sequencing, NGS)[1]是第二代 DNA 测序技术,其主要平台有 Illumina、Polonator、SOLiD 测序仪等。这些平台都使用循环芯片的测序方法,对充满 DNA 样本的芯片进行重复的 DNA 聚合酶反应和荧光序列读取反应。与传统的测序方法相比,NGS 具有高通量、成本低、效率高的优点,但得到的序列通常包含大量的错误,需要利用计算机对序列进行辅助修正,其修正过程是先检测出序列中错误的碱基,然后通过特定的算法对其进行修正。计算机纠错处理的核心是设计一个好的纠错算法,以提高序列修正的性能和质量。因此,本研究对现有的纠错工具进行比较,给相关研究提供参考依据;设计与实现并行分布式纠错算法,解决大规模基因数据分析处理中计算机内存消耗大和运算时间长的问题。**【前人研究进展】**目前的错误修正算法主要基于以下 3 种方法实现[2-3]:通过进行 K-mer[4]统计的基于 K-spectrum 方法、基于后缀树的方法和基于 MSA[5]的方法。其中,基于 K-spectrum 方法的工具主要有 Sga[6]、Racer[7]、Musket[8]和 Lighter[9]等,基于后缀树方法的工具主要有 SHREC[10]、HiTec[11]等,基于 MSA 方法的工具主要有 ECHO[12]、Coral[13]等。

基于 K-spectrum 的错误修正算法简单易懂,修正效果较好,且易于在各种平台上实现。其中心思想是首先把每个短序列划分为长度为 k 的 K-mers,然后将所有的 K-mers 进行排列并计数。当一个 K-mer 出现的次数达到一定的阈值 N 时,则认为该 K-mer 是 *solid* 的,否则认为该 K-mer 是非 *solid* 的,即该序列含有错误碱基,接着用 *solid* 序列去修正含有错误碱基的序列。基于 K-spectrum 方法的 4 种主要纠错工具各具特点:Sga 是 2012 年提出的一个基因组装工具,同时使用 FM-index 和 BWT 算法对序列进行压缩存储。其自带 Map 功能,淘汰了不能匹配的数据,所以修正性能较好。Racer 是 2013 年提出的一个纠错工具,因为 Racer 把序列存放到 hash 表中,所以程序对数据处理占用内存较小且运行速度较快。Musket 是一个专为 Illumina 测序产生的序列进行纠错的工具,主要使用双端纠错、单端推进和选举法进行纠错。Lighter 通过采样算法获得一系列尽可能与参考基因相似的 K-mers,然后从中找出属于 *solid* 的 K-mers 存放到 Bloom filter。Lighter 通过对 K-mers 的采样处理加快了程序对 K-mers 的统计操作,进而缩短程序的运行时间。**【本研究切入点】**由于目前的纠错软件主要是在单机上运行的,如要处理大规模的基因数据,则要求用高性能计算机来实现。Hadoop[14]的 HDFS 系统可以快速地访问存储在各计算节点的数据,非常适用于开发处理海量数据集的应用程序,且 HDFS 系统具有极高的容错性,用户可以将 Hadoop 部署在多台性能一般的计算机集群上组成分布式系统,所以在 Hadoop 上实现分布式并行纠错算法有利于解决数据的存储和运算的优化问题,且在生物计算领域,越来越多的问题已通过 Hadoop 平台得到解决,如基因表达双聚类和重叠社区算法等。**【拟解决的关键问题】**对现有基于 K-spectrum 的纠错工具进行比较,找出性能较好的纠错工具,然后在借鉴前人的纠错算法的基础上,在 Hadoop 平台上实现分布式并行纠错算法,降低 NGS 错误修正算法对计算机内存的消耗,提高其计算性能。

1 基于 K-spectrum 方法的常用纠错工具性能分析

1.1 数据来源

本研究采用的实验数据 Staphylococcus aureus (SA)、Rhodobacter sphaeroides (RS)、Human Chromosome 14(HC14)、Bombus impatiens(BI)来自 GAGE 网站(http://gage.cbcb.umd.edu/data/index.html),对应的参考基因组数据来自 NCBI,实验数据的大小依次为 242 MB、436 MB、9.6 GB、92 GB。

1.2 数据预处理

在进行实验前,首先需要删除含 A、T、C、G 以外的其它任何字母的基因序列。本实验使用的基因数据都是双链的,每一个基因数据都有两个 Fastq 文件,一个对应序列的左链,一个对应序列的右链。在处理一个文件的增删时,必须要同时对另一个文件进行对应的增删操作,以保证序列是 paired-end 的。接着便可使用纠错工具对序列进行纠错处理,并用 BWA 工具把序列映射到参考基因组上,得到每个序列在参考基因组上的起始位置。

1.3 Gain 综合分析

通过对原序列、纠错工具生成的序列和参考基因组中对应的碱基进行比对,对修正后原序列中错误碱基的各种修改状态分别进行统计,其修改状态包括修改正确、修改错误、没有进行修改。并利用统计结果计算出衡量纠错工具性能的信息增益 Gain。Gain 数据的计算公式如下:

$$\text{Gain}=\frac{\text{TP}-\text{FP}}{\text{TP}+\text{FN}}, \tag{1}$$

其中:TP 表示纠错工具改对的碱基个数,FP 表示纠错工具改错的碱基个数,FN 表示纠错工具没有对错误碱基进行修改的碱基个数。

1.4 性能分析结果

Gain 综合分析结果如图 1 所示,Lighter 对小规模数据的错误修正效果较好,但在处理大规模数据时效果较差。从总体上看,无论是在处理小规模数据还是大规模数据上,Sga 和 Racer 都表现出了较好的修正效果。

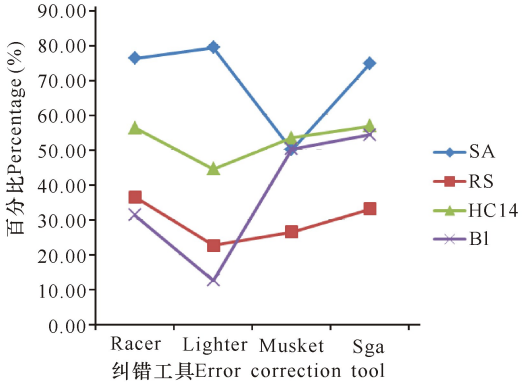

图 1 SA、RS、HC14、BI 的 Gain 信息

Fig. 1 Gain of SA, RS, HC14 and BI

图 2 和图 3 分别是 Lighter、Racer、Sga 和 Musket 处理 SA、RS、HC14、BI 数据时,计算机的内存峰值和运行时间比较。由图 2 和图 3 分析可知,对于相同的数据,Lighter 对内存和运行时间的消耗较少。说明 Lighter 采用的采样操作算法对提高程序的运行时间和内存的性能是有效的。Sga 在处理每一个数据时,消耗的时间总是远远大于其他纠错工具。但从实验结果上看,大多数的纠错工具都存在内存占用大的问题。为了解决该问题,可以利用分布式计算平台把普通的计算机有效地联合起来,对大规模数据进行分布式并行处理。

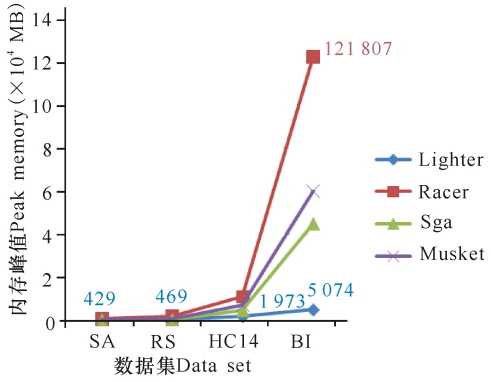

图 2 4 种算法的内存峰值比较

Fig. 2 Peak memory comparison of 4 algorithms

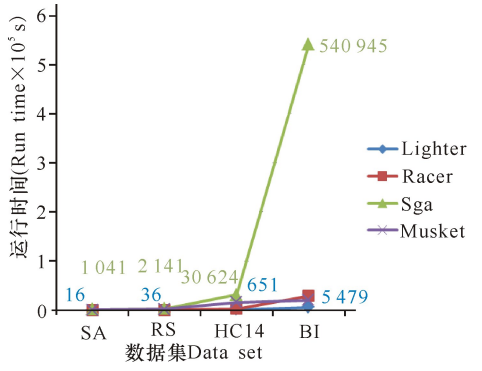

图 3 4 种算法的运行时间比较

Fig. 3 Running time comparison of 4 algorithms

2 Hadoop 分布式并行纠错算法

2.1 程序设计

本研究的 Hadoop 平台搭建在广西大学亚热带农业生物资源保护与利用国家重点实验室生物信息学团队的 Openstack[15] 平台上。其中,Openstack 平台搭建在 5 台曙光物理节点上,每个节点的主要配置为 6 核处理器,32 GB 内存和 300 GB 硬盘。

程序的设计思想:把预处理后的文件只取基因序列交给 Hadoop 的任务机制,Hadoop 通过判断任务的数据量大小来决定该任务需要分配多少个节点进行处理。然后将每个节点的数据交给 Map 函数,Map 函数并行地处理该文件,并将序列数据以 $<key,value>$ 的形式输出,*value* 中存放序列数据,*key* 不赋值。文件读取完毕后,Map 函数会调用 Correct 函数进行纠错处理,其过程为程序按设定的

K-mer 长度，遍历所有的序列，分析每一步生成的 K-mer 是否在 Bloom Filter 中，如果存在则说明此 K-mer 是正确的，否则说明遍历到的碱基是错误的，把错误的碱基用 A、T、C、G 进行遍历替换。定义一个记录得分的变量 score，如果替换碱基后的 K-mer 在 Bloom Filter 中，则 score 加 1。找到一个得分值达到所检查 K-mers 数量的一半以上，且分值最大的碱基进行替换，否则不进行修改。把数据放到 Hadoop 平台上进行纠错的伪代码如下：

```
protected void map(LongWritable key,Text value,Context context) {
String line = value.toString(); //从 HDFS 上读取一行数据
String fixed = corrector.correct(line);//修正序列
text.set(fixed);
context.write(NullWritable.get(),text);//写入 HDFS
}
public String correct(String seq) {
……
for (int i = k - 1; i < sequence.length; i++) { //默认不修正首段
  if (! lastIsSolid(i)) { //判断序列是否在已构建的 Bloom Filter 中
      for (int i = 0; i < DNA.length; i++) { //DNA[]中存放 ATCG 四个元素
        ……
       for (j = curIndex - k + 1; j <= curIndex; j++) { //定义 K 的长度为 17
         for (p = 0; p < k; p++) { //截取 curIndex-k+1~curIndex 间的 kmer
              if (p + j >= sequence.length)
              break;
           kmer[p] = sequence[p + j];
         }
          if (p < k) break; //不能组成完整的 kmer
          …….
            kmer[pos--] = DNA[i]; //在 kmer 的 pos 位置替换 DNA[i]
           if (isExist(kmer)) score++; //如 kmer 在已构建的 Bloom Filter 中得分+1
       }
      }
      //如所检查的 kmers 数量一半以上是 soild 的则进行碱基替换
       sequence[i]= score >= checkCount / 2 ? DNA[rIndex] :sequence[curIndex];
  }
}
  return new String(this.sequence);//返回修改后的序列
}
```

2.2 算法验证

为了验证基于 Hadoop 平台的分布式并行纠错算法（Parallel algorithm）的可行性，本研究利用 SA、RS 和 HC14 数据对程序进行测试，并与程序的串行单机运行方式（Serial algorithm）、Lighter 和 Racer 做比较。其中本研究提出的算法的单机方式运行在广西大学计算机与电子信息学院的惠普服务器上，其配置为 32 核处理器，64 GB 内存和 1 TB 硬盘。实验得到的运行时间比较情况如图 4 所示，内存的比较情况如图 5 所示。

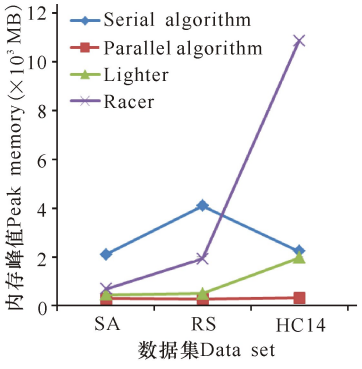

图 4 4 种算法的内存峰值比较

Fig. 4 Peak memory comparison of 4 algorithms

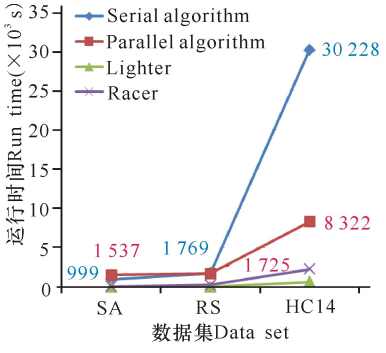

图 5 4 种算法的运行时间比较

Fig. 5 Running time comparison of 4 algorithms

通过对不同程序在处理相同数据时的运行时间和内存峰值进行比较，发现基于 Hadoop 平台的分布式并行纠错程序在内存的消耗方面相比其它程序少，且随实验数据规模的增大，其内存的变化非常平稳。由于程序设计得还不够完善，所以在运行时间上分布式算法的时间要比 Lighter 和 Racer 算法的运行时间长。与算法的单机运行方式比较时，发现当数据集小于 RS 时，单机运行方式较快，其主要原因是分布式并行算法在进行 MapReduce 运算时消耗的时间较多。但当数据集大于 RS 时，分布式并行算法运行的时间比单机串行算法的运行时间减少

了 72.5%。从整体上来看,Hadoop 分布式并行纠错算法能更有效地解决现有基于 K-spectrum 的 NGS 错误修正算法对大规模基因数据处理时,在内存和运行时间上的瓶颈问题。

3 结论

基于K-spectrum 的 NGS 错误修正算法直观易懂、错误修正效果较好,但在处理规模较大的数据时消耗大量内存。现有的基于 K-spectrum 的常用纠错工具中,Sga 和 Racer 的纠错效果较好,Lighter 的纠错效果较差,但 Lighter 的采样算法使数据存储空间大大减少,降低了对计算机内存的需求。本研究基于 Hadoop 的 HDFS 系统与 MapReduce 编程模式,提出了 Hadoop 分布式并行纠错算法,并与其串行程序、Racer 和 Lighter 进行比较,发现该算法降低了对计算机单机的内存消耗,并在一定程度上提高了纠错算法的运算性能,说明该算法是可行的。

参考文献:

[1] SHANKS M E, DOWNES S M, COPLEY R R, et al. Next-generation sequencing (NGS) as a diagnostic tool for retinal degeneration reveals a much higher detection rate in early-onset disease[J]. European Journal of Human Genetics, 2013, 21(3): 274-280.

[2] YANG X, CHOCKALINGAM S P, ALURU S. A survey of error-correction methods for next-generation sequencing[J]. Briefings in Bioinformatics, 2013, 14(1): 56-66.

[3] 江育娥,黄伟,林劼. 下一代测序纠错方法综述[J]. 北京工业大学学报, 2016, 42(3): 377-386.
JIANG Y E, HUANG W, LIN J. Error correction in preprocessing of next-generation sequencing[J]. Journal of Beijing University of Technology, 2016, 42(3): 377-386.

[4] CHOR B, HORN D, GOLDMAN N, et al. Genomic DNA k-mer spectra: Models and modalities[J]. Genome Biology, 2009, 10: R108.

[5] DIERINGER D, SCHLÖ TTERERER C. Microsatellite analyser (MSA): A platform independent analysis tool for large microsatellite data sets[J]. Molecular Ecology Notes, 2003, 3(1): 167-169.

[6] GONNELLA G, KURTZ S. Readjoiner: A fast and memory efficient string graph-based sequence assembler[J]. Bmc Bioinformatics, 2012, 13: 82.

[7] ILIE L, MOLNAR M. RACER: Rapid and accurate correction of errors in reads[J]. Bioinformatics, 2013, 29(19): 2490-2493.

[8] LIU Y C, SCHRÖDER J, SCHMIDT B. Musket: A multistage *k*-mer spectrum-based error corrector for illumina sequence data[J]. Bioinformatics, 2013, 29(3): 308-315.

[9] LI S, FLOREA L, LANGMEAD B. Lighter: Fast and memory-efficient sequencing error correction without counting[J]. Genome Biology, 2014, 15(1): R1.

[10] SCHRÖDER J, SCHRÖDER H, PUGLISI S J, et al. SHREC: A short-read error correction method[J]. Bioinformatics, 2009, 25(17): 2157-2163.

[11] ILIE L, FAZAYELI F, ILIE S. HiTEC: Accurate error correction in high-throughput sequencing data[J]. Bioinformatics, 2011, 27(3): 295-302.

[12] KAO W C, CHAN A H, SONG Y S. ECHO: A reference-free short-read error correction algorithm[J]. Genome Research, 2011, 21(7): 1181-1192.

[13] SALMELA L, SCHRÖDER J. Correcting errors in short reads by multiple alignments[J]. Bioinformatics, 2011, 27(11): 1455-1461.

[14] WHITE T, CUTTING D. Hadoop: The definitive guide[J]. O'reilly Media Inc Gravenstein Highway North, 2010, 215(11): 1-4.

[15] CORRADI A, FANELLI M, FOSCHINI L. VM consolidation: A real case based on open stack cloud[J]. Future Generation Computer Systems, 2014, 32: 118-127.

(责任编辑:陆 雁)