多序列星比对算法的改进及其在Spark中的并行化研究

董改芳 付学良 李宏慧
(内蒙古农业大学计算机与信息工程学院 呼和浩特 010018)

摘 要 多序列星比对算法在确定中心序列时需要计算任意两个输入序列的距离及分数,其较高的时间复杂度耗费了大量时间,因此提出了通过综合计算每个序列产生的k-mers及各个k-mer在各序列中出现的次数来确定k-mers的拼接选择,由k-mers进行拼接从而得到中心序列。进而,在双序列比对过程中采用搜索两个序列最大相似子串的思想,改进的星比对算法的精度在一定程度上得到了明显提升。接着,将改进的星比对算法在Spark中进行并行化设计与实现。采用Spark的Yarn-Client运行模式,对正常人线粒体的多组数据进行实验,分析了算法性能上的不足及改进方向。

关键词 多序列比对,星比对算法,K-mer,Spark,RDD
中图法分类号 TP391 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.10.010

Improvement of Multiple Sequence Center Star Method and Its Parallelization in Spark

DONG Gai-fang FU Xue-liang LI Hong-hui
(College of Computer and Information Engineering, Inner Mongolia Agricultural University, Hohhot 010018, China)

Abstract Because center star alignment algorithm needs to calculate the distance and scores of any two input sequences when determining the central sequence, it caused the high time complexity. A strategy for determining the assembling selection of k-mers was proposed by synthesizing computing the k-mers generated by each sequence and the number of occurrences of each k-mer in each sequence. Furthermore, in the process of pair wise sequence alignment, the idea of searching two sequences of the largest similar sub-sequences was used. The accuracy of the improved center star alignment algorithm is improved with a certain degree. The improved center star alignment algorithm was parallelized designed and implemented in Spark. Spark's Yarn-Client running mode was used to experiment the multi-group data of normal mitochondria. The performance of the algorithm was analyzed and the direction of improvement was analyzed.

Keywords Multiple sequence alignment, Center star method, K-mer, Spark, RDD

基因序列比对是生物信息学的重要研究议题之一。通过比对计算序列间的相似性,可以探索和发现新的基因结构和功能,同时也可以获得不同物种间的进化联系。进一步,基因编码和基因调控也是通过比对手段完成的;甚至代谢途径、核酸和蛋白质的结构和功能等理性知识都需要通过序列比对来实现[1]。因此生物信息领域离不开序列比对。

然而,生物学数据的激增速度远超过计算机硬件体系结构和计算性能的提升速度[2],传统的单机串行比对算法已经过时,只有合理的并行化算法设计及在高性能硬件集群上的计算才能满足部分的现实计算需求。

已经有许多学者对多序列比对的并行化算法进行研究,同时开发了相应的计算工具。Jacek Blazewicz[3]于2013年发表了基于GPU的快速精确的多序列比对算法工具,取得了很好的效果。朱香元[4]提出一种在众核计算机上实现的多序列并行处理策略。Adam Gudy[5]提出了一种用于图形加速器的更快的多序列比对方法QuickProbs。还有一些其他的研究者[6-11]及团队[12-22]也对该算法进行了研究。

纵观这些研究,专门针对大规模DNA/RNA序列的算法和软件却很少。邹权于2015年对这方面进行了研究[23]。Chen X[26]于2017年在异构GPU/CPU平台上采用改进的星比对方法,充分利用GPU和CPU硬件设备对星比对方法进行加速,使其计算性能可达11倍的加速。本文以大规模DNA/RNA数据集为研究对象,针对星比对算法中确定中心序列时随机选择输入n个序列中的一个[24],无目的性的缺

到稿日期:2017-07-05 返修日期:2017-08-15 本文受国家自然科学基金(61063004,61363006),内蒙古自然科学基金(2015MS0605,2015MS0626,2015MS0627),内蒙古教育厅高校研究项目(NJZC059),教育部留学人员基金([2014]1685),内蒙古自治区科技计划项目:穿透降水量GSM网络在线监测与数据传输系统的研制资助。

董改芳(1979-),女,博士生,副教授,主要研究方向为基因拼接与比对算法及其并行化、智能计算与数据挖掘、群智能算法,E-mail:donggf@imau.edu.cn;**付学良**(1966-),男,博士,教授,主要研究方向为智能计算与数据挖掘、农业信息化,E-mail:fuxl@imau.edu.cn(通信作者);**李宏慧**(1970-),女,博士,教授,主要研究方向为网络优化。

陷，以及通过 $n(n-1)/2$ 次两两比对计算序列间距离时花费时间较长的缺陷，提出通过由 n 个序列产生 k-mers，计算 k-mers出现的次数，由 k-mers 的次数确定可以进行拼接的 k-mers，从而拼接得到中心序列的改进策略。使用该策略的改进算法可在一定程度上提升比对精度。由于 Spark 具有基于内存计算的特性，将改进的星比对算法在 Spark 中进行了并行化设计与实现，分析了算法并行化性能的不足及改进方向。

1 星比对算法及其改进

1.1 基本星比对算法

基本星比对算法是专门针对多序列比对的一种启发式搜索方法[23]。其主要思想是根据 n 个输入序列的距离确定中心序列，并将中心序列和其他序列进行比对。比对过程中遵循“一旦是空格，永远是空格”的思想。当中心序列与其他序列两两比对结束后，由于在中心序列与每个序列的双序列比对结果中都加入了空格，因此把这些双序列比对后加入中心序列的空格汇总到同一个中心序列，同时把在中心序列相应位置加入的空格加到比对序列上。星比对算法如算法 1 所示。

算法 1 星比对算法

输入：k 条序列 $s_1, s_2, \cdots, s_k$

For i=1 to k do

  For j=1 to k do

    计算 s_i 与 s_j 的最佳比对得分 $V(s^i, s^j)$；

  End

End

确定中心序列 c，使得和项 $\sum_{i=1}^{k} V(c, s^i)$ 达到最小值；

Foreach 星形中心之外的序列 do

  按照“一旦有空位，永远是空位”的原则将新序列合并到多序列比对中；

End

1.2 星比对算法的改进

星比对算法的优点是避免了渐进式比对方法中所有参与的序列都需两两比对，两两比对的次数为 $n(n-1)/2$，而星比对方法中两两比对的次数为 $2*n$，效率明显提升。星比对算法的缺点是在计算中心序列时需要计算序列之间的比对得分或者距离，整个算法的大多数时间都花费在计算中心序列上。因此，本文提出基于综合计算 k-mers 的次数，根据 k-mers 的次数拼接 k-mers 而得到中心序列的改进方法，如算法 2 所示。

算法 2 改进的星比对算法

输入：k 条序列 $s_1, s_2, \cdots, s_k$

For i=1 to k do

  分解 s_i 为 k-mers

End

删除重复的 k-mers，为剩余 k-mers 编号，记录 k-mers 出现的次数及出现的序列

根据 k-mers 之间的重叠关系和 k-mers 出现的次数，拼接 k-mers 得到中心序列 c

Foreach 星形中心之外的序列 do

  按照“一旦有空位，永远是空位”的原则将新序列合并到多序列比对中；

End

2 改进的星比对算法的并行化

2.1 k-mer 的定义及中心序列 s_c^1 的计算

定义 1 已知字符串 s 和整数 $k(k<|s|)$，从 s 的第一个字符开始到第 $|s|-k+1$ 个字符，依次截取长度为 k 的子串，每个子串就是由 k 生成的一个 k-mer。

假设 s=agctgctaa，k=6，则 s 生成的 k-mers 为 agctgc，gctgct，ctgcta，tgctaa，gctaaa。

为了便于表示，把最初得到的中心序列称为 s_c^1。在改进的星比对算法中，首先对每个序列计算产生多个 k-mers，然后把 n 个序列产生的 k-mers 进行去重，并为剩余的 k-mers 编号，记录其出现的次数及来自序列的编号，根据 k-mers 重复的次数进行 k-mers 拼接，得到一条新的序列，这条拼接而成的新序列即 s_c^1。接下来描述这种改进的星比对算法的并行化设计过程。

2.2 算法的并行化

Spark 是由美国的 AMPLab 实验室于 2009 年开发出来的，由于它的计算是基于内存的，因此在同等情况下，Spark 与 MapReduce 相比体现出了强大的时间优势。

Spark 的核心组件是 RDD，也称为弹性分布式数据集。Spark 的并行化程序设计思想是：首先将输入数据转换成 RDD，然后对此 RDD 相继实施一系列算子进行计算，从而达到数据处理的目的。由于 RDD 分布在不同的计算结点上，因此有 RDD 操作的位置就存在并行。若要提高算法效率，必须尽可能地将 RDD 中的数据并行执行。

2.2.1 s_c^1 的设计

把 Spark 中的改进星比对算法称为 IM-CS-Spark，IM-CS-Spark 中求解 s_c^1 的设计思路如图 1 所示。

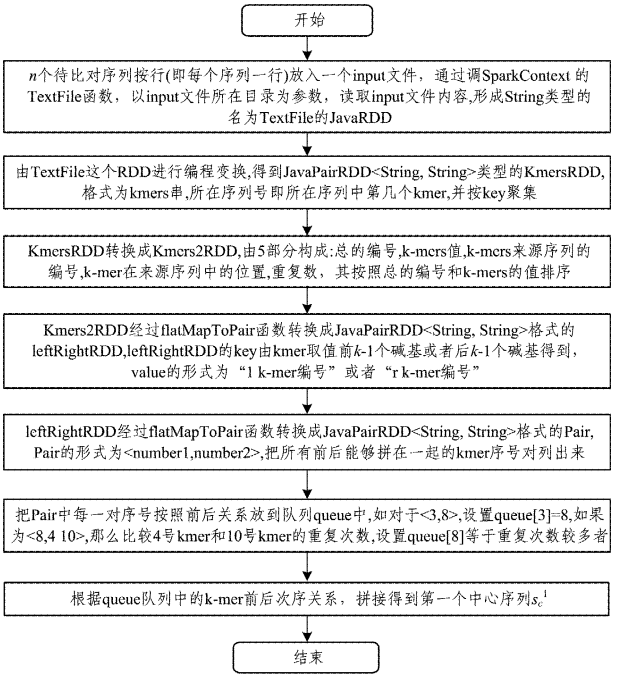

图 1 IM-CS-Spark 的流程图

2.2.2 双序列比对思想

求得 s_c^1 之后，将 s_c^1 与输入的 n 个序列进行两两比对，需

要调用双序列比对算法 n 次。在求得 s_c^2 之后，依然将 s_c^2 与输入的 n 个序列进行两两比对，再调用双序列比对算法 n 次，因此整个算法将调用双序列比对算法 $2*n$ 次，双序列比对算法的设计好坏将直接影响最终的比对精度和效率。本文的双序列比对采用尽可能从两个待比对的序列中搜索最大相似子串的思想。

假设输入序列中心序列为 s_c^1，s_c^1 的长度为 n，输入序列之一为 S_i，S_i 的长度为 m，双序列比对算法的具体步骤如下：

(1)从 s_c^1 中第 $i(1\leqslant i\leqslant n)$个位置起查找与 S_i 最长的不交叠共同子串 out_i，将 out_i 记录输出。

(2)对于每个 out_i，定位其在 s_c^1 中的位置 l_1 和 S_i 中的位置 l_2，计算 $div=l_1-l_2$。

(3)若 $div<0$，说明子串 out_i 在 s_c^1 中的位置比在 S_i 中的位置靠前，在 s_c^1 中插入 div 个空格。

(4)若 $div>0$，说明子串 out_i 在 S_i 中的位置比在 s_c^1 中的位置靠前，在 S_i 中插入 div 个空格。

(5)所有 out_i 处理结束后，如果两个插入空格的串长并不相同，则直接在较短串的后面插入相差的空格。

2.2.3　s_c^2 的计算

假设输入的 n 条序列如下($n=3$)：

1)MEKKTLGFLFLLLLVLAADV

2)MARSVPLVSTIFVFLLLLVA

3)MARSLPLVSTIFVFFLLLLV

经过 2.2.1 节的计算，s_c^1 如下：

MEKKTLGFLFLLLLVA

并且，s_c^1 与 3 个序列的双序列比对结果如下：

s_c^1　MEKKTLGFLFLLLLVA* * * *
1　MEKKTLGFLFLLLLVLAADV
s_c^1　MEKKTLGFL* * * *FLLLLVA
2　MARSVP LVSTI FVFLLLLVA
s_c^1　MEKKTLGF L* * * * *FLLL LVA
3　MARSLP LVSTI FVFFLLL LV *

将 3 组双序列比对结果在 s_c^1 中插入的所有空格都加到同一个 s_c^1 中，得到 s_c^2。因此 s_c^2 为：

MEKKTLGFL* * * * *FLLLLVA* * * *

将 s_c^2 再与 n 个输入序列进行两两双序列比对，得到 n 组双序列比对结果：

MEKKTLGFL* * * * *FLLLLVA* * * *
MEKKTLGFL* * * * *FLLLLVLAADV

MEKKTLGFL* * * * *FLLLLVA* * * *
MARSVPLVSTIFV*FLLLLVA* * * *

MEKKTLGFL* * * * *FLLLLVA* * * *
MARS LPLVSTIFVFFLLLLV* * * * *

将这 n 组双序列比对结果中加入了空格的 n 个原始序列放在一起即为最终的比对结果。最终结果为：

MEKKTLGFL* * * * *FLLLLVLAADV
MARSVPLVSTIFV*FLLLLVA* * * *
MARSLPLVSTIFVFFLLLLV* * * * *
&　　&&&&&&

其中，“&”表示对齐的列。20 列中有 7 列得到匹配。

3　实验设计与结果分析

3.1　数据与平台搭建

本文采用 5 组正常人的线粒体基因序列作为实验数据[23]，如表 1 所列。采用 3 台 ThinkServer 服务器搭建 1 个 Master，2 个 Slave 的集群。3 台 ThinkServer 服务器的硬件配置完全相同，有 2 个 4 核 CPU，每个 CPU 的主频为 2.4GHz，内存容量为 32G。

表 1　数据集

数据集	最大长度	最小长度	平均长度	序列数	文件大小
Mt Genome	16579	16556	16570	96	1M
1X				672	10M
2X				13440	213M
5X				33600	512M
10X				67200	1G

硬件平台上安装的软件有：Red Hat Enterprise Linux Server release 6.4，Centos release 6.5，Hadoop-2.2.0，JDK 1.7.0_25 和 Spark 1.5.2。

3.2　实验设计及结果

表 2 列出了改进星比对算法的精确度，通过综合计算 k-mers 的次数，根据 k-mers 在每个序列中出现的次数来确定中心序列，并且在双序列比对中，采用查找两个序列中最大相似子串的思想来进行双序列比对，可得到更加精确的比对精度。

表 2　比对精度比较

数据集	方法	精度
Mt Genome	IM-CS-Spark	165795
	文献[27]	181692
	ClustalW	182780

表 2 中的精度计算方法是对于比对结果中任意两对碱基，匹配得 0，不匹配得 1[27]。因此最终所得的分数越少，说明匹配的碱基对越多，精度越高。

IM-CS-Spark 在集群上采用 Yarn-Client 的运行模式，采用如下命令：

```
[imau@master ~]$ ./spark-submit
--master yarn-client
--executor-cores 1
--num-executors 2
--executor-memory 6g
--conf spark.default.parallelism=4
--class com.spark.main.Backup
/home/imau/Dong/2x.jar
```

在计算小于 13440 个序列时，参数 executor-memory 采用的是默认值，但是在计算数据 2x 时，出现了“lost Executor in slave1”错误，因此，将参数 executor-memory 的值设置为 6g，该错误即可消除，程序运行回归正常；如果 executor-memory的值设置得太大，如 8g，就会出现超出集群配置上限的错误。

表 3 列出了 IM-CS-Spark 与其他几种并行方法的时间比较。从表 3 可以看出，本文算法经并行化设计后，仅比

KAlign 的计算时间短，而远长于其他 3 种并行方法。这并不符合 Spark 由于基于内存计算，与 MapReduce 相比性能能够提升 100 倍的官网描述。

表 3 计算时间比较

	1x	2x	5x	10x
IM-CS-Spark	97m	36h	93h	180h
HAlign(Trie Tree)	3m16s	—	—	—
HAlign(Hadoop)	2m21s	10m53s	14m14s	28m28s
MAFFT	1m41s	175m	984m	—
KAlign	170m44s	—	—	—

3.3 并行程序分析及改进方向

通过检验算法的并行化程序发现，在将中心序列 s_c^1 与 n 个输入序列进行双序列比对的过程中，将 n 个输入序列分别置于 n 个输入文件中，采用迭代的方法从序列 1 迭代到序列 n，序列 i 与中心序列 s_c^1 进行双序列比对时会读取第 i 个输入文件，并将此输入文件转换为 RDD。例如，序列 1 对应的文件为 input1. txt，该文件只有 1 行，由 input1. txt 生成一个 RDD，将此 RDD 与 s_c^1 进行比对，比对结束后缓存结果，然后进行下一次迭代，由序列 2 的原始数据生成一个 RDD，将此 RDD 与 s_c^1 进行比对，重复该过程直至处理完序列 n。实际上整个过程虽然用到了 RDD，但其是一个串行的过程，因为每个 RDD 中只有一个元素，只有 RDD 中有大量元素或者大量的行时，才能将这些元素分到不同结点上并行。因此该程序耗费时间较长，性能较差。

中心序列 s_c^2 与 n 个序列两两比对时，也是采取如上方法，也是一个串行的过程，势必耗费大量时间，这是程序性能较低的原因。因此，下一步考虑在双序列比对的过程中修正并行化程序的设计，将 n 个序列依然放在同一个输入文件中，当输入数据源转换为 RDD 后，每行是一个序列，同时每行也是 RDD 的一个元素。这样 RDD 的 n 个元素分布在不同的结点上，与 s_c^1 的双序列比对是一个并行的过程，必然会节省时间，也符合并行设计的思路。

结束语 本文在星比对算法的基础上，提出了一种通过生成各序列的 k-mers，计算各序列中 k-mer 出现的次数，以 k-mer 出现的次数为依据拼接而得到一条中心序列的改进策略，进而，在双序列比对算法中采用在两个序列中搜索最大相似子串的思想，使得比对算法的精确度具有一定程度的提升。之后，在 Spark 中对改进星比对算法进行并行化设计及编程，采用 Spark 的 Yarn-Client 模式进行运行，发现并行化算法的时间性能并未占优势。通过分析算法的设计流程，找到了应该改进的方向，即不应该采取迭代的方式进行两两比对，而应该将输入的 n 个序列置于同一个输入文件中，并将其转换为 RDD。之后取 RDD 中的元素与中心序列进行比对，从而使双序列比对过程为一个并行化过程。下一步工作将对比进行尝试。

另外，Spark 的集群性能与资源配置有极大的关系，比如在运行的过程中，设置最合适的参数，如 Executor-memory，spark. default. parallelism，num-executors，executor-cores 等，研究它们与输入数据的大小、类型有何种关系，这涉及到资源调优的研究。

最后，在 Spark 程序设计的过程中，为了能够使得性能最优，需要遵循一些原则，如避免创建重复的 RDD、尽可能复用同一个 RDD、对多次使用的 RDD 进行持久化等，这些将涉及到开发调优的研究。

今后将针对双序列比对并行化设计、资源调优、开发调优方面展开进一步的研究。

参 考 文 献

[1] Kevin. 生物序列比对研究的现实意义[EB/OL]. http://blog.sina.com.cn/s/blog_5f6687090 1013mvp.html. 2012.05.07.

[2] 生物大数据激增或揭示疾病如何发生[EB/OL]. http://www.cbdio.com/BigData /2016-08/17/content_5191896.htm.

[3] BLAZEWICZ J, FROHMBERG W, KIERZYNKA M, et al. G-MSA-A GPU-based, fast and accurate algorithm for multiple sequence alignment[J]. Journal of Parallel and Distributed Computing, 2013, 73(1): 32-41.

[4] ZHU X Y, LI K L, AHMAD S. A data parallel strategy for aligning multiple biological sequences on multi-core computers[J]. Computers in Biology and Medicine, 2013, 43(4): 350-361.

[5] GUDY A, DEOROWICZ S. QuickProbs-A Fast Multiple Sequence Alignment Algorithm Designed for Graphics Processors [J]. PLoS One, 2014, 9(2): e88901.

[6] SADASIVAM, SUDHA G, BAKTAVATCHALAM G. A novel approach to Multiple Sequence Alignment using hadoop data grids[J]. International Journal of Bioinformatics Research and Applications, 2010, 6(5): 472-483.

[7] SHU N J, ARNE E. KalignP: Improved multiple sequence alignments using position specific gap penalties in Kalign2[J]. Bioinformatics, 2011, 27(12): 1702-1703.

[8] FABIAN S, DAVID D, ANDREAS W, et al. Making automated multiple alignments of very large numbers of protein sequences [J]. Bioinformatics, 2013, 29(8): 989-995.

[9] KAZUTAKA K, TOH, HIROYUKI. Parallelization of the MAFFT multiple sequence alignment program[J]. Bioinformatics, 2010, 26(15): 1899-1900.

[10] LAN H D, CHAN Y D, XU K, et al. Parallel algorithms for large-scale biological sequence alignment on Xeon-Phi based clusters[J]. BMC Bioinformatics, 2016, 17(9): 267.

[11] 王洋子豪. 知乎[EB/OL]. https://www.zhihu.com/question/19903344 /answer/13779421.

[12] YANG C Y, ZHONG C. CPU and GPU co-ordinate parallel accelerate multiple biological sequence alignment[J]. Journal of Chinese Mini-Micro Computer Systems, 2016, 37(12): 2780-2784. (in Chinese)
杨春燕，钟诚. CPU 和 GPU 协同并行加速多生物序列比对[J]. 小型微型计算机系统，2016，37(12)：2780-2784.

[13] LIN J, TANG M, TONG R F. GPU accelerate biological sequence alignment[J]. Journal of Computer-Aided Design and Computer Graphics, 2010, 22(3): 420-427. (in Chinese)
林江，唐敏，童若锋. GPU 加速的生物序列比对[J]. 计算机辅助设计与图形学学报，2010，22(3)：420-427.

（下转第 84 页）

参 考 文 献

[1] KWON Y C,DYLAN N,JEFFREY G,et al. Scalable clustering algorithm for N-body simulations in a shared-nothing cluster [M]// Scientific and Statistical Database Management. Springer Berlin Heidelberg,2010:132-150.

[2] YANG X H,MO H J,VAN DEN B,et al. A halo-based galaxy group finder: calibration and application to the 2dFGRS[J]. Monthly Notices of the Royal Astronomical Society,2005,356(4):1293-1307.

[3] YANG X H,MO H J,VAN DEN B,et al. Galaxy groups in the SDSS DR4. I. The catalog and basic properties[J]. The Astrophysical Journal,2007,671(1):153.

[4] WILLIAM C,et al. UPC Language Specifications Version 1.3 [OL]. https://upc-lang.org/assets/Uploads/spec/upc-lang-spec-1.3.pdf.

[5] Intel® VTune® Amplifier XE[OL]. https://software.intel.com/en-us/intel-vtune-amplifier-xe.

[6] Technical Advances in the SGI® UV® Architecture[OL]. https://www.sgi.com/pdfs/4192.pdf.

[7] HAO H,SI Y M,WEI J W,et al. Optimizing Irregular Memory Access in Astrophysical Clustering Studies[J]. Journal of Frontiers of Computer Science and Technology,2017,11(1):80-90. (in Chinese)
郝赫,司雨蒙,韦建文,等. 天体物理成团研究中的非规则访存优化[J]. 计算机科学与探索,2017,11(1):80-90.

[8] MARK D,GEORGE E,CAROS F,et al. The evolution of large-scale structure in a universe dominated by cold dark matter[J]. The Astrophysical Journal,1985,292(2):371-394.

[9] NEAL K,LARS H,DAVID W. Galaxies and gas in a cold dark matter universe[J]. The Astrophysical Journal,1992,399(2):L109-L112.

[10] EISENSTEIN D J,HUT P. Hop:A new group-finding algorithm for n-body simulations[J]. The Astrophysical Journal,1998,498(1):137.

[11] LIU Y,LIAO W K,CHOUDHARY A. Design and evaluation of a parallel HOP clustering algorithm for cosmological simulation [C]// International Parallel and Distributed Processing Symposium,2003. IEEE,2003.

[12] FU B,REN K,LÓPEZ J,et al. DiscFinder:A data-intensive scalable cluster finder for astrophysics[C]// Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing. 2010:348-351.

(上接第 58 页)

[14] YE X C,LIN W,FAN D R,et al. Parallel optimization of protein sequence alignment algorithm in public nucleus structure[J]. Journal of Software,2010,21(12):3094-3105. (in Chinese)
叶笑春,林伟,范东睿,等. 蛋白质序列比对算法在众核结构上的并行优化[J]. 软件学报,2010,21(12):3094-3105.

[15] CAO Z Y,LANG X Y,LIU X,et al. Parallel optimization of super-large-scale sequence alignment calculation[J]. Journal of Computer Applications,2011,31(2):32-35. (in Chinese)
曹宗雁,郎显宇,刘昕,等. 超大规模序列比对计算的并行优化[J]. 计算机应用,2011,31(2):32-35.

[16] JIN X,LUO Z G,JIANG X Z,et al. Parallel Design and Implementation of Multi-Sequence Alignment Software T_Coffee[J]. Computer Applications and Software,2008,25(4):221-223. (in Chinese)
靳新,骆志刚,蒋晓舟,等. 多序列比对软件 T_Coffee 的并行化设计与实现[J]. 计算机应用与软件,2008,25(4):221-223.

[17] WEI S F,LIU Y,JIANG C Y. Parallel Research on Multiple Sequence Alignment of Genetic Annealing Based on GPU[J]. Computer Engineering and Design,2014,35(4):1247-1252. (in Chinese)
韦树烽,刘羽,蒋财运. 基于 GPU 的遗传退火多序列比对并行研究[J]. 计算机工程与设计,2014,35(4):1247-1252.

[18] WANG W D,TANG W,DUAN B,et al. Study on Parallel Acceleration Technique of High-throughput Gene Sequences based on HASH index[J]. Journal of Computer Research and Development,2013,50(11):2463-2471. (in Chinese)
王文迪,汤文,段勃,等. 基于 Hash 索引的高通量基因序列比对并行加速技术研究[J]. 计算机研究与发展,2013,50(11):2463-2471.

[19] ZHANG L,CHAI H,WO L K,et al. Research on Biological Sequence Alignment Algorithm and Graphics Hardware Acceleration[J]. Chinese Journal of Biomedical Engineering,2011,30(6):853-858. (in Chinese)
张林,柴惠,沃立科,等. 生物序列比对算法与图形硬件加速研究[J]. 中国生物医学工程学报,2011,30(6):853-858.

[20] ZHU X Y,LI R F,LI K L,et al. Research on Parallel Processing of Biological Sequence Alignment Based on Heterogeneous System[J]. Computer Science,2015,42(11):390-399. (in Chinese)
朱香元,李仁发,李肯立,等. 基于异构系统的生物序列比对并行处理研究进展[J]. 计算机科学,2015,42(11):390-399.

[21] ABUÍN J M,PENA T F,PICHEL J C. PASTASpark:multiple sequence alignment meets Big Data[J]. Bioinformatics,2017.

[22] ZAMBRANO-VEGA C,NEBRO A J,GARCÍA-NIETO J,et al. M2Align:parallel multiple sequence alignment with a multi-objective metaheuristic[J]. Bioinformatics,2017.

[23] ZOU Q,HU Q,GUO M,et al. HAlign:Fast multiple similar DNA/RNA sequence alignment based on the centre star strategy[J]. Bioinformatics,2015,31(15):2475-2481.

[24] LI D P,JU Y,ZOU Q. Application in tumor research of multiple sequence star alignment method based on MapReduce[J]. Cancer Progress,2016,14(6):510-513. (in Chinese)
李大鹏,鞠颖,邹权. 基于 MapReduce 的多序列星比对方法在肿瘤研究中的应用[J]. 癌症进展,2016,14(6):510-513.

[25] ZOU Q,GUO M,WANG X,et al. An Algorithm for DNA Multiple Sequence Alignment Based on Center Star Method and Keyword Tree[J]. Acta Electronica Sinica,2009(37):1746-1750.

[26] CHEN X,WANG C,TANG S,et al. CMSA:a heterogeneous CPU/GPU computing system for multiple similar RNA/DNA sequence alignment[J]. Bmc Bioinformatics,2017,18(1):315.

[27] ZOU Q,SHAN X,JIANG Y. A Novel Center Star Multiple Sequence Alignment Algorithm Based on Affine Gap Penalty and K-Band[J]. Physics Procedia,2012,33:322-327.