doi:10.3969/j.issn.1672-5565.2014.03.05

新型图形硬件支持下的动态规划局部比对算法加速研究

张 林
(浙江中医药大学生命科学学院,浙江 杭州 310053)

摘 要:为探索准确、高效、低成本、通用性并存的生物序列局部比对方法。将点阵图算法、启发式算法等各种序列局部比对算法中准确性最高的动态规划局部比对算法在计算机中实现,并通过流式模型将其映射到图形硬件上以实现算法加速,再通过实例比对搜索数据库完成比对时间和每秒百万次格点更新(MCUPS)性能值评测。结果表明,该加速算法在保证比对准确性的同时,能显著提升比对速度。与目前最快的启发式算法相比,比对平均加速为14.5倍,最高加速可达22.9倍。

关键词:生物分子;序列局部比对;动态规划局部比对算法;图形硬件
中图分类号:Q-332,R318 **文献标志码**:A **文章编号**:1672-5565(2014)-03-179-06

Research of dynamic programming local alignment algorithm acceleration based on a new type of graphics hardware

ZHANG Lin
(*College of Life Science, Zhejiang Chinese Medical University, Hangzhou* 310053, *China*)

Abstract: This paper is aimed to explore biological sequence local alignment method with accuracy, efficiency, low-cost and universality. We presented dynamic programming local alignment algorithms with higher accuracy than the other local alignment algorithms, such as lattice diagram algorithm and heuristic algorithm, in computer and mapped it to the graphics hardware by stream model to speed up the algorithm. The alignment time and million cell updates per second (MCUPS) were used to evaluate the performance of the accelerated algorithm by an example of database alignment scanning. The result showed that the accelerated algorithm greatly improved the alignment speed and ensured the alignment accuracy at the same time. The alignment speed averagely was 14.5 times and maximally 22.9 times as fast as that of heuristic algorithm with highest speed at present.

Keywords: Biological macromolecule; Sequence local alignment; Dynamic programming local alignment algorithm; Graphics hardware

生物序列分析是生物信息学研究的一个重要手段,它是了解基因组结构和功能的基本途径[1]。而在生物序列分析中,生物序列比对是最常用和最经典的研究手段[2]。基于真核表达的剪接特点和蛋白质的模块性质,目前,序列局部比对已成为生物序列比对中的主要分析方式。其通过将查询序列与整个生物数据库的所有序列进行局部比对,来完成一系列的生物学分析,探索生命起源和存在的内在规律。

而在后基因组时代,生物数据库含有海量数据。因此,在保证比对准确的同时如何提高其效率,已成为序列局部比对的瓶颈。现在比较重要的是动态规划局部比对算法[3],准确性高是其最大优点,但存在着效率问题。为了增加动态规划局部比对算法的效率,最早提出的解决方案是加入启发式算法,例如现在常见的 FASTA 和 BLAST 算法[4],但其准确性受到影响。现在的思路是采用专门的生物信息处理机器和专有硬件提高动态规划局部比对算法的比对速度,但其缺点是投入太大,实现困难。

近年来,图形硬件(计算机显卡)发展迅速,主

收稿日期:2014-05-04;**修回日期**:2014-06-09.
基金项目:浙江省医药卫生科技计划项目(2013KYA137);浙江省自然科学基金项目(LY13H020007);浙江省中医药科学研究基金计划(2011ZB027)资助。
作者简介:张林,男,硕士,讲师,研究方向:生物信息学医药分析、生物信息数据挖掘与分析;E-mail:shibailezlzl@sina.com.

要具有以下优点。第一,浮点运算能力强,显存与位宽大,非常适合高密度的运算,也非常适合流式处理[5]。第二,并行度很高,具有统一渲染构架,非常适合并行运算。第三,可编程,具备了通用计算能力[6-7],在图像处理[8]、计算几何[9]、科学计算[10]、生物数据分析等领域已有多处应用[11-12]。第四,价格便宜,性价比非常高。

鉴于上述背景,本文拟以动态规划局部比对算法为研究对象,将其实现,然后利用图形硬件将其加速,并通过一系列实例进行加速性能评测,探索兼具准确性、高效性、通用性的生物序列局部比对方法。

1 实验材料与方法

1.1 实验材料

电脑配置如下:CPU:Intel 酷睿 i7 4960X(频率 4 GHz、15 MB 三级缓存);内存:DDR3(16 GB);硬盘:1 TB 。

3 款最新的计算机显卡(图形硬件),品牌为 NVIDIA,型号为 Geforce GTX 760、Geforce GTX 770、Geforce GTX 780Ti。安装在上述配置的电脑上分别运行。

1.2 实验方法

1.2.1 实现动态规划局部比对算法

实现动态规划局部比对算法,即寻找两条序列局部最优匹配的算法,又称 Smith-Waterman 算法。

1.2.2 利用图形硬件加速动态规划局部比对算法

将上述算法通过逻辑映射和物理映射移植到图形硬件上,并研究在图形硬件上研究对其加速。

1.2.3 加速后测评

查询对象和数据库选择:在 UniProtKB/Swiss-Prot 蛋白质数据库中选择 8 条不同长度的蛋白质查询序列,它们的 accession 编号分别为 Q64372(长度 73aa)、Q9CQW9 (长度 137aa)、Q60961 (长度 233aa)、Q8WU90 (长度 426aa)、Q8R2G6 (长度 949aa)、Q8CJ12 (长度 1 009aa)、Q01815 (长度 2 139aa)、Q0F9K2 (长度 4 113aa)。将上述 8 条蛋白质序列比对搜索 UniProtKB/Swiss-Prot 数据库(2014_02 版本,序列平均长度 375aa,共有1 283 282 个蛋白分子)。

实验组:比对执行动态规划局部比对算法(Smith-Waterman 算法),在上述三款图形硬件(见实验材料)上分别运行并加速。

对照组:作对照的是由 CPU 执行的 SSEARCH 程序。它采用的是 FASTA 启发式算法,在动态规划局部比对算法(Smith-Waterman 算法)基础上进行了启发式算法加速,比对速度快,但准确性受到影响。对照组其他比对条件、电脑配置均与实验组一致(见实验材料)。

评价参数:时间性能(执行时间)、*MCUPS*(Million Cell Updates Per Second)性能值。其中 *MCUPS* 为"每秒百万次格点更新",在生物序列比对中,其代表了每秒能计算完成的得分矩阵的格点的数量,其值越高,说明比对的速度越快、效率越高。设查询序列长度为 L,比对的数据库中有 n 个分子序列,其中第 i 个序列的长度为 L_i,查询序列比对搜索数据库所耗费的时间为 t。则

$$MCUPS = \frac{\sum_{i=1}^{n} L_i * L}{t * 10^6}。 \quad (1)$$

2 实验结果

2.1 实现动态规划局部比对算法

2.1.1 动态规划局部比对算法基本描述

设有两条生物分子序列,分别为 Y = $Y_{[1]}\cdots Y_{[p]}$(长度为 p)和序列 Z= $Z_{[1]}\cdots Z_{[q]}$(长度为 q)进行比对。采用简单打分:残基匹配得 2 分,不匹配与空位均扣 1 分,利用 c( )函数来表示序列残基间的得分。

最优比对得分矩阵(简称得分矩阵)的设置如下。设 E 为得分矩阵,E(i,j)表示序列 Y 的前缀 Y[1]…Y[i-1]Y[i]和序列 Z 的前缀 Z[1]…Z[j-1]Z[j]之间的最佳比对得分(1<=i<=n, 1<=j<=m)。E 具备初始条件:E(i,0)= E(0,j)= 0。

2.1.2 利用递归打分计算得分矩阵 E

递归打分函数如下:

E(i,j)= max{E(i-1,j-1) +c(Y[i],Z[j]); E(i-1,j)-1;E(i,j-1)-1;0}。

动态规划局部比对算法计算得分矩阵 E 具体实现如下:

```
i=1;
while(i<=p)
{
    j=1;
    while(j<=q)
    {if(E(i-1,j-1)+c(Y[i],Z[j])> E(i-1,j)-1)
        E(i,j)= E(i-1,j-1)+c(Y[i],Z[j]);
    else
            E(i,j)= E(i-1,j)-1;
        if(E(i,j)< E(i,j-1)-1)
            E(i,j)= E(i,j-1)-1;
        if(E(i,j)<0
```

```
            E(i,j)=0;
        j++;
    }
    i++;
}
```

通过上述运算,获得 E。

$$E=(e_{ij})_{pq}=\begin{bmatrix}0 & 0 & 0 & \cdots & 0\\ 0 & e_{11} & e_{12} & \cdots & e_{1q}\\ 0 & e_{21} & e_{22} & \cdots & e_{2q}\\ . & . & . & & .\\ . & . & . & & .\\ . & . & . & & .\\ 0 & e_{p1} & e_{p2} & \cdots & e_{pq}\end{bmatrix}$$

2.1.3　利用回溯获得最优比对序列

回溯算法实现如下:

```
define GY,GZ;     //存放最优比对序列
i=p,j=q;
while(i>0&&j>0&& E(i,j)! =0)
{
    if(E(i,j)= = E(i-1,j-1)+c(Y[i],Z[j]))
    {GY=i;GZ=j; i--;j--;}
    else if (E(i,j)= =E(i-1,j)-1)
    {GY =i;GZ ='-'; i--;}
    else
    {GY ='-';GZ =j;j--;}
}
rollback (GY); rollback (GZ);//     rollback ()
```

为序列反转函数

2.2　利用图形硬件加速动态规划局部比对算法研究

2.2.1　算法障碍与瓶颈分析

动态规划局部比对算法主要包含两个步骤,一是递归打分计算得分矩阵 E,二是回溯获得最优比对序列。其中,步骤一的计算复杂度为 O(p *q),步骤二的计算复杂度为 O[max(p,q)]。因此,步骤一是瓶颈,需利用图形硬件进行加速,而步骤二对计算时间的影响几乎为 0,可以在继续在 CPU 上执行。

2.2.2　递归打分计算得分矩阵的映射过程

经过分析发现,在递归打分计算得分矩阵 E 时,任意格点值的计算,只与其左边、上方、斜上方的三个格点相关(见图 1)。因此,可将 E 的计算作为流式处理,其中,打分过程映射为流处理核心(Kernel)、计算得到的 E 数据映射为流数据(Stream Data),循环进行(见图 2(a)),此为第一步——逻辑映射。此外,我们还要完成第二步,物理映射,将上述逻辑模型移植到图形硬件上完成。图形硬件具有良好的流处理能力,其片段着色器(Fragment Processors)可完成 Kernel 功能,纹理系统则可以纹理数据的形式保存 Stream Data,循环操作完成 E 的计算(见图 2(b))。

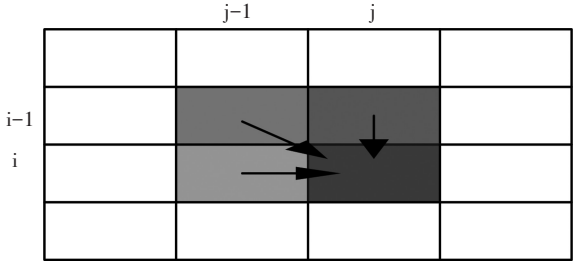

图 1　递归打分中的数据相关性

Fig.1　Data relativity of recursion marking

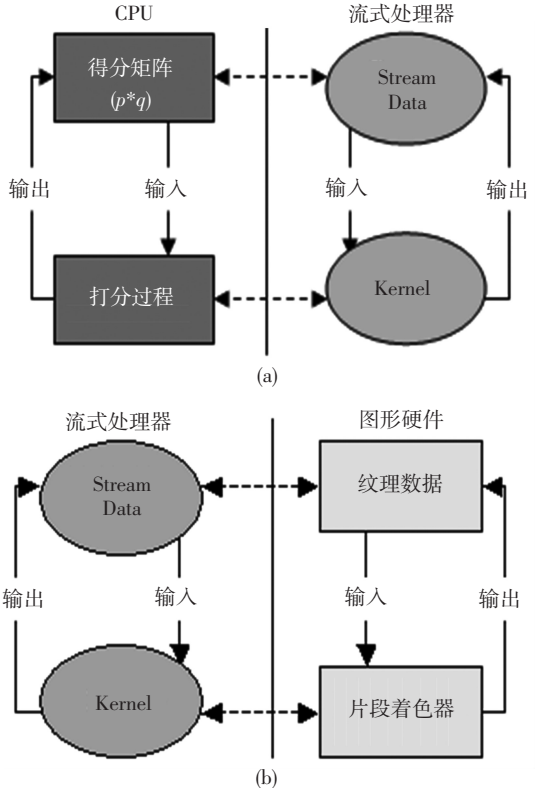

图 2　递归打分的映射过程

Fig.2　The mapping process of recursion marking

注:(a)递归打分的逻辑映射;(b)递归打分的物理映射。

Notes:(a) Logistic mapping of recursion marking;(b) Physical mapping of recursion marking.

另外,在真实情况下,生物序列之间的打分规则会比较复杂,形成打分矩阵。而对于蛋白质分子的比对,由于蛋白质的进化性质,还需引入不同的取代矩阵进行打分。鉴于此,可将打分矩阵也保存在纹理中,供片段着色器读取。图 3 给出了递归打分计算得分矩阵算法的最终映射过程。

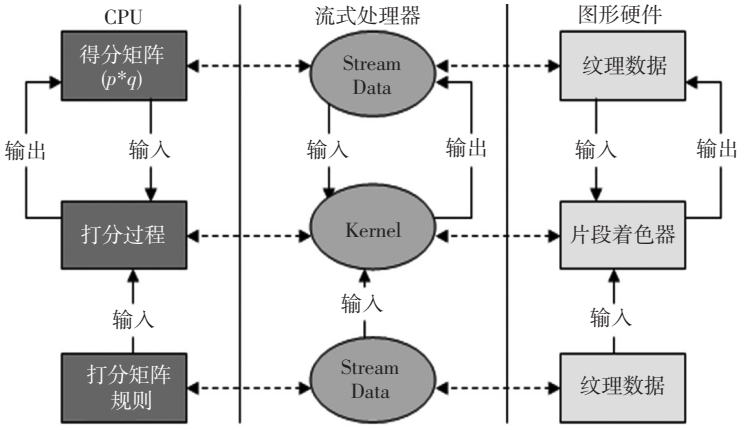

图 3 递归打分算法的最终映射过程

Fig.3 Final mapping process of recursion marking algorithm

2.2.3 利用逆对角线法对递归打分进行并行加速

通过上面分析发现,在递归打分计算得分矩阵 E 时,任意格点值的计算,只与其左边、上方、斜上方的三个格点相关,因此每一条逆对角线上格点的计算互不干扰(见图 4(a))。所以,我们可对同一条逆对角线上的所有格点并行计算,而图形硬件恰好具备良好的并行计算的能力,可完成此项操作(见图 4(b))。因为逆对角线的条数为 p+q−1,从而将计算复杂度从原来的 O(p * q)降低为 O(p+q−1)。

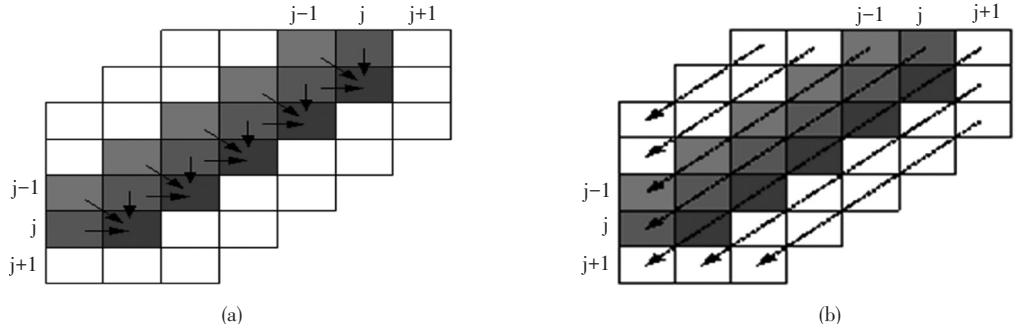

图 4 利用逆对角线在图形硬件上进行加速计算

Fig.4 Accelerated computing by the inverse diagonal rule on the graphics hardware

注:(a)逆对角线上的格点计算规律;(b)逆对角线上的并行计算。

Notes: (a) Computing rule of inverse diagonal; (b) Parallel computing of inverse diagonal.

综上所述,映射后利用图形硬件计算得分矩阵 并实现加速的具体算法如图 5:

```
Initialization buffers、textures、render targets    //初始化纹理、缓存与渲染对象
activate fragment program    //激活片段着色器程序
    for each inverse diagonal
    {
        bind buffers R_{x-1} and R_{x-2} as textures    /*选择3个独立的缓存R_{x-1}、R_{x-2}和R_x,分别代表逆对角线x-1、
                                                        x-2和x,接着将R_{x-1}、R_{x-2}绑定到纹理*/
        set rebder target ti byffer R_x    //将R_x设置成渲染对象

        set uniform variables    //设置必要的统一变量
        draw vertices and texture coordinates    /*片段着色器访问纹理获得数据,并发计算逆对角线x上格
                                                        点的得分权值,并完成渲染着色与纹理映射*/
        release texture buffers    //释放缓存与纹理的绑定
    }
deactivate fragment program    //释放片段着色器程序
read results from framebuffer    //从帧缓存中获得结果,得到得分矩阵E
```

图 5 在图形硬件中递归计算得分矩阵

Fig.5 Computing scoring matrix on the graphics hardware

2.3　动态规划局部比对算法加速后性能测评

2.3.1　执行时间

表 1 中显示的是:8 条长度不同的序列分别由 SSEARCH、Geforce GTX 760、Geforce GTX 770、Geforce GTX 780Ti 执行数据库比对搜索所耗费的时间。

表 1　时间性能评测结果

Table 1　Performance evaluation results in time

查询序列长度	SSEARCH(单位:秒)	Geforce GTX 760(单位:秒)	Geforce GTX 770(单位:秒)	Geforce GTX 780Ti(单位:秒)
73	52.9	15.1	13.0	10.8
137	101.3	17.1	14.7	12.2
233	170.5	20.1	17.3	14.4
426	328.9	31.8	27.3	22.8
949	686.1	63.0	54.2	45.2
1 009	789.5	57.7	49.7	41.4
2 139	1 715.9	113.0	97.2	81.0
4 113	3 419.9	208.2	179.2	149.3

2.3.2　MCUPS 性能值评测

8 条长度不同的序列分别由 SSEARCH、Geforce GTX 760、Geforce GTX 770、Geforce GTX 780Ti 执行数据库比对搜索的 MCUPS 性能值见图 6。

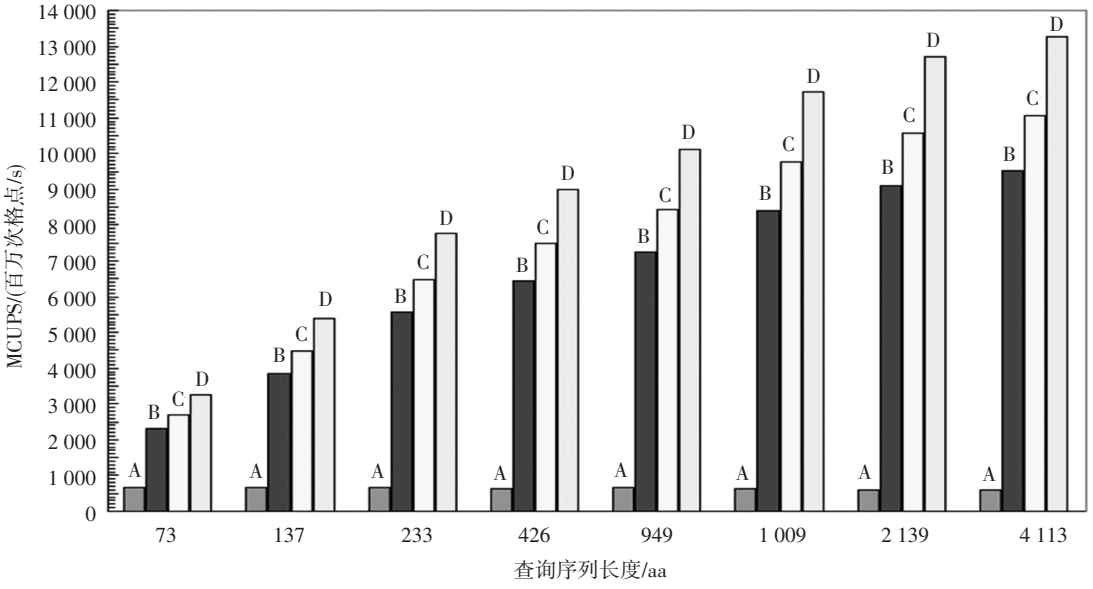

图 6　MCUPS 性能比较

Fig.6　Performance comparison in MCUPS

注:A:SSEARCH,B:Geforce GTX 760,C:Geforce GTX 770,D:Geforce GTX 780Ti。

Notes:A:SSEARCH,B:Geforce GTX 760,C:Geforce GTX 770,D:Geforce GTX 780Ti.

3　讨　论

3.1　时间性能评测分析

第一,无论是利用 SSEARCH 还是图形硬件进行序列比对,比对所需要时间都与序列长度成正比,序列长度越长,比对的时间也就越长,符合比对规律。

第二,NVIDIA Geforce GTX 760、NVIDIA Geforce GTX 770、NVIDIA Geforce GTX 780Ti 中任意一款图形硬件的比对时间,都要大大短于 SSEARCH。

第三,在 NVIDIA Geforce GTX 760、NVIDIA Geforce GTX 770、NVIDIA Geforce GTX 780Ti 三块图形硬件之间,比对所消耗的时间与硬件性能成反比,图形硬件性能越强,耗时越短。其中,GTX 780Ti 性能最强,耗时最短。

3.2　MCUPS 值评测分析

第一,NVIDIA Geforce GTX 760、NVIDIA Geforce GTX 770、NVIDIA Geforce GTX 780Ti 和 SSEARCH 相比,平均加速分别 10.4 倍、12 倍、14.5 倍,比对速度都有飞跃性地提高。

第二,随着查询序列长度的增加,比对加速越发

显著,查询序列长度越大、比对需要耗费的时间越多,图形硬件的加速效率则越高。

第三,从 760 到 780Ti,随着性能不断增强,比对速度提升显著。以此次测试中性能最高的 780Ti 为例,对于不同长度的查询序列,对比于 SSEARCH,其平均加速越为 14.5 倍;当查询序列长度增加到最高的 4113aa 时,比对加速达到了 22.9 倍。

3.3 综合分析

从上述时间性能和 MCUPS 性能值均可知,利用图形硬件进行加速的比对算法,其速度要远超利用启发式算法提速的比对算法,能极大地提升生物序列局部比对速度。此外,近年来图形硬件每月都有新款推出,硬件性能不断提升,通过图形硬件进行比对加速还有更为广阔的提升空间及更为光明的前景。第三,相比利用启发式算法进行加速,图形硬件加速算法严格执行动态规划局部比对算法,比对准确性不会有任何地牺牲。最后,图形硬件可在个人电脑上运行,价格非常便宜,性价比极高,相比于专门的生物信息处理机器和专有硬件来说,成本可以忽略不计,能大规模推广和应用,通用性极强。

4 结 论

综上所述,将准确率最高的动态规划局部比对算法予以实现,然后将其映射到图形硬件(计算机显卡)上实现算法加速,并通过实例评测其性能。结果表明该加速算法在保证比对准确性的同时,能显著提高比对速度。从而为实现准确性、高效性、通用性并存的生物序列局部比对方法做出新的探索,为生物学、医学、高性能计算的结合开拓新的方向。

参考文献(References)

[1] 陈铭,包家立.生物信息学[M].北京:科学出版社,2013.
CHEN Ming, BAO Jiali. Bioinformatics [M]. Beijing: Science Press,2013.

[2] WIKIPEDIA S. Bioinformatics: proteomics, hidden markov model, biostatistics, proteome, sequence alignment, full genome sequencing [M]. USA: University-Press, 2013.

[3] SMITH T, WATERMAN M. Identification of common molecular subsequences [J]. Journal of Molecular Biology, 1981,147 (1):195-197.

[4] David J.Multiple sequence alignment methods [M]. New York: Springer,2014.

[5] TANG M, TONG R F, NARAIN R, et al. GPU-based streaming algorithm for high-resolution cloth simulation [J]. Computer Graphics Forum,2013,32 (7):21-30.

[6] MANOCHA D. General-purpose computations using graphics processors[J].Computer,2005, 38(8):85-88.

[7] 张宝华,王海水,许禄.DNA 序列编码及相似度计算[J].高等学校化学学报,2006, 27(12):2277-2280.
ZHANG Baohua,WANG Haishui,XU lu. DNA sequence code and similarity computations[J]. Chemical Journal of Chinese Universities,2006, 27(12):2277-2280.

[8] WEISS, R, SHRAGGE J. Solving 3D anisotropic elastic wave equations on parallel GPU devices [J]. Geophysics, 2013,78(2): 7-15.

[9] BAI H T, LI Y G, CHEN L Y, et al. Parallel optimization of geometric correction algorithm based on CPU-GPU hybrid architecture [J]. Applied Mechanics and Materials, 2014,543(26):2804-2808.

[10] DANIE L, JESWIN G, JUSTIN H, et al. Stencil-Aware GPU optimization of iterative solvers[J]. SIAM Journal on Scientific Computing,2013,35(5):209-228..

[11] PAYNE B R, OWEN G S, WEBER I. Accelerating protein structure recovery using graphics processing units [A]. In Proceedings of International Conference on Computational Science (ICCS 05)[C]. Singapore: Springer-Verlag, 2005.

[12] MENG X D,JI Y Q. Modern computational techniques for the HMMER sequence analysis [J]. Bioinformatics, 2013,2013(03):13-17.