# CUDA 下受体评分网格生成并行算法*

李正夫[1a,2]，王希诚[1b]，郭 权[2]

(1. 大连理工大学 a. 计算机科学与技术学院；b. 工业装备结构分析国家重点实验室，辽宁 大连 116024；2. 大连东软信息学院 计算机科学与技术系，辽宁 大连 116023)

**摘 要**：针对分子对接中生成评分网格需要花费很多的计算时间这一问题，提出了一种基于统一计算设备架构(CUDA)的评分网格生成并行算法。该算法把传统计算方法中三维计算空间中的一维通过在图形处理单元(GPU)上进行并行处理，使得总生成时间得到了降低，提高了评分网格的生成效率。实验结果表明，借助于 GPU 的浮点计算能力，提出的并行算法对比传统的计算方法可以显著缩短评分网格的生成时间，为评分网格的生成提供一种新的方式。

**关键词**：统一计算设备架构；并行算法；评分网格；分子对接

**中图分类号**：TP311；TP301.6 **文献标志码**：A **文章编号**：1001-3695(2013)03-0814-03

doi:10.3969/j.issn.1001-3695.2013.03.044

# Parallel algorithm for score grid computing of receptor with CUDA

LI Zheng-fu[1a,2], WANG Xi-cheng[1b], GUO Quan[2]

(1. *a. School of Computer Science & Technology, b. State Key Laboratory of Structural Analyses for Industrial Equipment, Dalian University of Technology, Dalian Liaoning* 116024, *China*; 2. *Dept. of Computer Science & Technology, Dalian Neusoft Institute of Information, Dalian Liaoning* 116023, *China*)

**Abstract**: The time of generating the scoring grid in molecule docking is very huge. Aiming at this problem, this paper introduced a parallel algorithm of score grid based on the compute unified device architecture (CUDA). In traditional methods, it needed to calculate every node in three-dimensional space. This algorithm took the calculation of one dimension on graphic processing unit (GPU). It reduced the total computational time. The experimental results show that the parallel algorithm with the help of GPU, can significantly shorten generation time of the score grid compared with the traditional calculating method of DOCK. The algorithm offers a new way for generating the scoring grid.

**Key words**: computer unified device architecture(CUDA); parallel algorithm; score grid; molecule docking

近些年来，随着计算机辅助药物设计的兴起，新药品的研究和设计都进入了一个新的时代。药物分子设计已作为一种实用化的工具介入到了药物研究的各个环节，并已成为创新药物研究的核心技术之一。据统计，由于分子模拟和计算机辅助药物设计的介入，使得平均一种新药研发的周期缩短了 0.9 年，直接研发费用降低 1.3 亿美元[1]。

计算机虚拟药物筛选一般是针对选定的靶标，使用高性能并行计算机，从庞大的小分子库中使用计算模拟的方法筛选出具有药理活性的化合物。分子对接法是模拟分子结合情况的一种计算方法，它是虚拟药物筛选中十分重要的方法。我国目前医药工业生产的药品大多是仿制国外的品种，自主创新的新药仅占极少部分，造成这种情况的一个原因就是计算模拟平台和分子对接软件的缺失与落后。发展医药研究的新平台和新算法成为发展我国医药研究水平的一个重要方面。

## 1 药物分子对接和受体评分网格

### 1.1 药物分子对接

很多生物大分子(受体)重要的生理和药理功能是通过它与小分子(配体)的相互作用体现出来的。分子对接法是指将小分子配体放置于大分子受体的活性位点处，并且寻找其合理的取向和构像，使得配体和受体的形状和相互作用的匹配最佳[2]。但是，配体与受体的相互作用是一个十分复杂的过程。配体和受体在结合时，存在着静电相互作用、氢键相互作用、范德华相互作用和疏水相互作用，结合的部位一般是蛋白质的氨基酸残基。静电相互作用、氢键相互作用和范德华相互作用控制着配体与受体的结合。疏水作用是配体与受体结合的驱动力量，只有疏水作用强，配体和受体才能排除水分子而结合在一起。在配体与受体结合时，配体分子还要取得一定的构像，使其自身来适应受体结合部位的“空腔”形状和构像变化。同

**收稿日期**：2012-08-15；**修回日期**：2012-09-23 **基金项目**：国家自然科学基金资助项目(11072048,61170168)

**作者简介**：李正夫(1978-)，男，黑龙江安达人，博士研究生，主要研究方向为云计算、并行计算、计算机辅助药物设计(lizhengfu@sohu.com)；王希诚(1946-)，男，博导，教授，主要研究方向为生物力学、并行计算、优化算法；郭权(1972-)，男，教授，博士，主要研究方向为并行计算、网络安全.

时,为迎合配体的结合,受体的构像也会发生相应的变化。

20 世纪 80 年代,Kuntz 等人[3]发展了一种模拟小分子与生物大分子结合三维结构及其结合强度的计算方法——分子对接(docking)方法,并发展了第一个分子对接程序 DOCK。此后,在 DOCK 的基础发展了一系列分子对接方法,如 FlexX、AutoDock、GOLD 等,我国科研人员研制的 GAsDock[4] 和考虑蛋白质柔性的分子对接方法[5] 也得到了国内外很高的评价。到目前为止,DOCK 系列软件的最新版本为 DOCK 6.5。DOCK 系列软件凭借其使用相对简单、计算精度较高和公开源代码等特点,成为使用最为普遍的分子对接和虚拟筛选软件。

### 1.2　受体评分网格

由于分子对接的搜索空间巨大,如果要完全模拟配体与受体的结合,即允许配体与受体都可以根据匹配情况进行一定程度的构象变化,需要花费非常多的计算时间。通常的做法是,允许配体小分子进行一定程度的构象变化;而受体大分子则假定为刚性部件,不允许作任何构象变化。在这种前提下,为了进一步加快计算速度,普遍使用受体空间评分网格来替代真实的受体原子进行计算。

在进行分子对接计算时,使用评分网格上已经计算的节点进行差值得到对接点处受体的作用力,从而在高通量筛选时可以极大地缩短整体筛选时间。目前大多数的分子对接软件都使用评分网格方法进行对接计算,评分网格的生成精度直接关系到最后的分子对接结果。

生成受体空间网格一般先需要在受体活性位点外围创建一个三维空间上的虚拟盒子,需要指定盒子的中心坐标和盒子的三维长度,盒子需要把活性位点完全包围。DOCK 中使用交互式程序 Showbox 负责生成空间盒子,用户可以通过回答程序提出的一些问题,最后生成盒子;也可以使用程序完全自动生成盒子。Showbox 程序的输出是一个 PDB 格式的文件,可以使用查看 PDB 文件的软件观看包括盒子的受体空间三维结构。DOCK 中使用 Grid 程序在上述的盒子内计算生成对接所需要使用的空间评分网格。

一般使用 Grid 程序计算每一个网格节点上接触和能量两种打分,程序的输出文件分别是 *.cnt 和 *.nrg 两个文件。可以在 Grid 程序的输入文件中指定空间网格中相邻节点的距离、作用力的截断距离等参数信息,这些参数都将直接影响到 Grid 程序的执行时间和结果精度。

由于对三维网格中每一个节点都需要计算受体对其的作用力,而且受体中在截断距离内的每一个原子都需要计算,所以评分网格的生成需要非常长的计算时间。DOCK 中一次空间网格的生成计算方式是使用单个 CPU 进行串行计算。如果想要提高对接结果的精度,需要减小评分网格中相邻节点间距离,如果把评分网格中节点距离改为默认值的一半,所需要的计算时间大约是原来时间的 10 倍以上。

## 2　CUDA 和本文工作

### 2.1　CUDA 介绍

CUDA(compute unified device architecture,统一计算设备框架)是 NVIDIA 公司在 2006 年推出的针对其 GPU 设计提出的编程模型。CUDA 是一种通用并行计算架构,该架构能够使 GPU 解决复杂的计算问题。开发人员可以使用一种基于 C 语言的扩展语言来为 CUDA 编写程序,降低了入门和编程难度。可以在 CUDA 程序内部通过指令控制代码运行在 CPU 或是 GPU 上,即 host 端和 device 端,运行在 device 上的代码称为核函数。在 CUDA 下,可以应用到不同的数据集上的给定的一个指令序列被称为一个核心;GPU 是由运行在主机 PC 上的程序所控制的。主机使核心开始工作,并且在 CPU 工作内存和图形设备存储之间复制数据,各个核心是运行在 GPU 上的程序代码[6~8]。

GPU 具有非常强大的浮点计算能力,这是因为 GPU 是特别为计算密集、高并行度计算(如同图像渲染)设计的,因此将更多的晶体管用于数据处理而不是数据缓存和流控。GPU 非常适合处理那些能够表示为数据并行计算(同一程序在多个数据上并行执行)的问题。从设备的硬件架构来看,GPU 更适合进行细粒度的并行计算。

在 CUDA 计算模型下,GPU 执行的最小单元称为线程(thread),多个线程组成一个线程块(block)。GPU 的一次执行称为一个网格(grid),网格是由很多个线程块组成,它们之间的关系如图 1 所示。

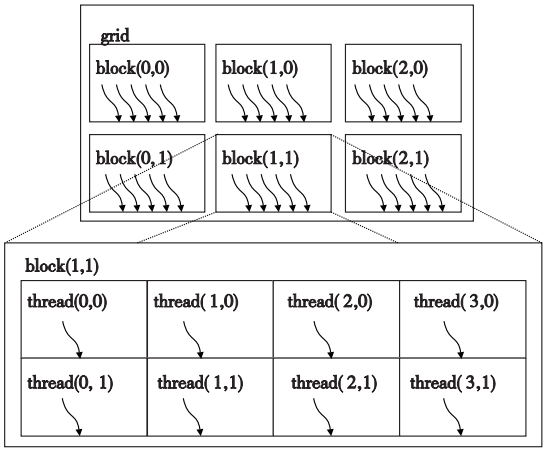


**图1　网格、线程块和线程关系示意图**

在执行期间,CUDA 线程可能访问来自多个存储器空间的数据。每个线程有私有的本地存储器;每个块有对块内所有线程可见的共享存储器,共享存储器的生命期和块相同,所有的线程可访问同一全局存储器[9]。

### 2.2　基于 CUDA 的评分网格生成算法

在分子对接设计中一般假定受体大分子为刚性,认为配体小分子为柔性,那么分子对接的过程可描述为求解如下优化问题的过程:

$$\min E = f(T_x, T_y, T_z, R_x, R_y, R_z, T_{b1}, \cdots, T_{bn}) \tag{1}$$

$$\text{s.t. } \underline{X} \leqslant T_x \leqslant \overline{X}, \underline{Y} \leqslant T_y \leqslant \overline{Y}$$

$$\underline{Z} \leqslant T_z \leqslant \overline{Z}, -\pi \leqslant \text{angle} \leqslant \pi$$

$$\text{angle} = R_x, R_y, R_z, T_{b1}, \cdots, T_{bn}$$

其中:$T_{b1}, \cdots, T_{bn}$是柔性配体中可旋转键的旋转角;$n$ 为可旋转键数目;$T_x$、$T_y$、$T_z$、$R_x$、$R_y$、$R_z$ 是配体中某一刚性子结构的中心坐标及旋转角,用于表示配体分子的取向信息。目标函数 $E$ 为分子间相互作用能:

$$E=\sum_{i=1}^{lig}\sum_{j=1}^{rec}\left(\frac{A_{ij}}{r_{ij}^{a}}-\frac{B_{ij}}{r_{ij}^{b}}+332.0\frac{q_iq_j}{Dr_{ij}^{a}}\right) \tag{2}$$

其中:$r_{ij}$为配体分子中原子 $i$ 与受体分子中原子 $j$ 之间的距离;$A_{ij}$、$B_{ij}$为范德华参数;$a$、$b$ 为范德华力指数;$q_i$、$q_j$ 为原子 $i$ 与原子 $j$ 的点电荷;$D$ 为介电常数;332 为将静电势能转换成标准单位(kcal/mol)的因子[10]。

在生成受体评分网格时,需要使用式(2)计算出受体大分子对于评分网格中每一个节点的相互作用能。在进行具体计算时,对于空间中的任意一个网格节点,受体中在以该节点为中心的三维截断距离之内的所有原子都需要进行考虑,意味着计算需要进行四次嵌套循环操作(受体原子和空间三维),这样就使得计算量急剧增大。

通过分析可以得知,传统评分网格计算所需要的计算量巨大,但是这些大量计算的过程基本相同,只是每次计算需要的数据集不一样,这就使得这些计算非常适合在有着众多计算核心的 GPU 上并行进行。一般来说,一次并行操作可以减少一次循环操作,所以基于 CUDA 的并行算法只需要三次嵌套循环操作,可以节约很多的计算时间。本文的并行算法去掉的是三维空间中的一维。

本文使用 CUDA,采用 CPU + GPU 混合编程模式,充分挖掘空间网格程序中的并行因素,使之尽可能地在 GPU 上来执行。但是 GPU 中的计算核心在进行计算时,所需要的计算数据必须放在设备显示内存中才可以进行。从程序设计方面来说,host 和 device 端之间需要进行大量的数据交换;从硬件结构的方面来说,在进行通用计算时,主机内存和显示内存之间需要使用 PCI-E 总线进行数据通信。在这样一种硬件结构的限制下,主机内存和显示内存间的数据传送速率就成为 GPU 并行计算的常见瓶颈问题之一。在使用 CUDA 进行计算时,如何有效地降低与显示内存之间的数据交换是一个非常值得考虑的问题。

对于近几年的显卡而言,高端显卡的显存容量已经达到了 2 GB 或 4 GB,足以满足比较大规模的数据存储。本文在进入并行计算之前,将计算所需要的数据尽可能多地一次性传入到显存中;在全部计算完成后,再将计算结果反馈回主机内存。这样就有效地减少了在计算过程中的显存与主存的数据交换数量,提高了程序整体的性能。

## 3 实验结果及分析

实验平台的操作系统为 Linux Red Hat Enterprise 5.5,64 bit;CPU 为 Intel Xeon E5620,主频 2.40 GHz;内存频率为 DDR3-1333 MHz,容量为 4 GB。GPU 为 NVIDIA 的 GeForce GT 440,内建了 96 个核心,显存大小为 512 MB,总线带宽 128 bit,GPU 频率为 810 MHz。

实验计算所使用大分子复合物的 PDB 编号为 1ABE。实验中选取评分网格节点间步长分别为 0.5、0.3、0.2 和 0.1,一般情况下步长为 0.3(或更小)的评分网格才可以用于实际的分子对接中。步长越小,所生成的评分网格中节点越多,进行分子对接时差值才更为准确。为了便于对比,本文评分网格程序与 DOCK 中评分网格程序都运行在同一个实验平台上,实验结果均为三次测试的平均值,计算时间的对比结果(单位:s)如表 1 所示。

表 1 本文程序与 DOCK 评分网格时间对比

| 序号 | 节点步长 | 节点总个数 | DOCK 结果 | 本文结果 |
|---|---|---|---|---|
| 1 | 0.5 | 214 720 | 46 | 15 |
| 2 | 0.3 | 954 450 | 224 | 36 |
| 3 | 0.2 | 3 176 738 | 805 | 116 |
| 4 | 0.1 | 25 154 871 | 13 982 | 1 989 |

从表 1 可以看出,对于评分网格步长(单位:埃)较大时,本文的并行算法所带来的性能提升相对较小,这主要是因为在计算强度不大时,在主机内存和显示内存的带宽限制下,GPU 浮点计算能力并没有完全发挥作用,较多的时间用于了数据的传输,而不是数据的计算上。而当评分网格步长较小时,这时所需要的计算量急剧增加,而需要传输的数据量并没有提高很多,这时 GPU 的潜力充分地发挥出来,从而使得 CUDA 下 GPU 并行算法可以获得非常高的效率。相比于 DOCK 的传统计算方法,在节点密度较大时,本文的并行算法速度最高可以提高到 7.03 倍左右,极大地缩短了计算时间。

## 4 结束语

本文提出了一种使用 CUDA 加速分子对接中受体评分网格计算的并行算法,在 NVIDIA 的 GT440 图形处理器上进行了规模不同的计算实验,并与 DOCK 中评分网格生成程序进行了时间上的对比。对比结果表明,在节点间步长较大时,本文算法对比于传统 DOCK 中计算方法时间提高幅度并不是很大;但是在步长较小需要大量计算时,本文算法可以显著缩短评分网格的生成时间。使用本文的并行算法可以在短时间内就完成评分网格的生成计算,也可以使用其生成更高密度的评分网格,使后续的分子对接计算出更为精确的对接结果。

**参考文献:**


[1] 李洪林,沈建华,罗小民,等. 虚拟筛选与新药发现[J]. 生命科学,2005,17(2):125-131.

[2] 陈凯先, 蒋华良, 嵇汝运. 计算机辅助药物设计——原理、方法及应用[M]. 上海:上海科学技术出版社,2000:25-33.

[3] KUNTZ I D, BLANEY J M, OATLEY S J, *et al*. A geometric approach to macromolecule-ligand interactions[J]. Journal of Molecular Biology,1982,161(2):269-288.

[4] LI Hong-lin, LI Chun-lian, GUI Chun-shan, *et al*. GAsDock: a new approach for rapid flexible docking based on an improved multi-population genetic algorithm[J]. Bioorganic & Medicinal Chemistry Letters,2004,14(18):4671-4676.

[5] 康玲,李洪林,王希诚. 一种考虑蛋白质柔性的分子对接方法[J]. 大连理工大学学报, 2008, 48(2):282-286.

[6] NICKOLLS J, ALLY W J. The GPU computing era[J]. IEEE Micro,2010,30(2):56-69.

[7] GEMBRIS D, NEEB M, GIPP M, *et al*. Correlation analysis on GPU systems using NVIDIA's CUDA[J]. Journal of Real-time Image Processing,2011,6(4):275-280.

[8] 杨梅,李志民,曹大勇. CUDA 架构下大规模稠密线性方程组的并行求解[J]. 计算机工程与应用,2011,47(32):27-30.

[9] NVIDIA Corporation. NVIDIA CUDA C programming guide (version 4.2)[R]. [S.l]:NVIDIA Corporation,2012:1-13.

[10] 郭权, 王希诚,李纯莲. 药物分子对接中应用网格的研究与进展[J]. 计算机研究与发展,2004,41(12):2054-2059.