第 3 章　多物理模拟与千万亿次计算

Steven F. Ashby

Lawrence Livermore National Laboratory

John M. May

Lawrence Livermore National Laboratory

3.1　引言

近几年由于超级计算机的发展和程序可扩展性的提高，科学计算和计算科学领域又掀起了新一轮的研究高潮。超级计算机和可扩展程序代码使用三维的解决方法，利用更多精确的参数,使真实模拟物理现象成为可能。举个例子，而今的程序和计算机可以通过三维的解决方案，利用复杂的辐射传播模型来模拟超新星的各种物理过程与现象。这种模拟技术极大地促进了科学的发展。

科学家对包含多物理模型（multiple physical model）和高可扩展性的代码越来越感兴趣。在材料科学领域，计算科学家期望通过从微尺度和中型尺度等多个尺度来模拟材料的极限属性。

万亿次超级计算机在一定范围内已经被用来进行复杂的模拟，但是它的性能还不足以模拟精确的跨越多尺度的模型。这种计算需求要求千万亿次计算能力平台的出现。工业界声称千万亿次计算需要大规模的并行，首先需要增加计算结点的数目，现在则更加需要增加每块芯片的核心数目（详见章节3.2）。

这种增加的计算能力将会被应用在新的场景中，过去计算科学家从时间和空间的角度来处理问题，但是现在科学家希望整合时间和空间以及不同的物理规则。换句话说，在每一维将使用2n个处理器来模拟4种不同的尺度，取代使用8n个处理器来解决三维模拟的问题（以前在n台处理器上进行）。

多物理应用（multiphysics applications）程序和大规模千万亿次计算机两种趋势的综合需要一种新的方法来进行大规模的科学模拟。在过去，大多数模拟是数据并行，使用所有机器的处理器对分布式数据进行相似的操作。多尺度、多物理模拟存在内在的任务并行，并且需要一致的编程模型（详见章节3.3）。这些应用也要求可扩展的算法来适应问题的多尺度、多物理特性（详见章节3.4）。

3.2　下一代超级计算机

在20世纪90年代早期，超级计算机逐渐演变到如今非常熟悉的架构：通过高速网络来互联大量的通用处理器。这些处理器就是用于工作站和商业服务器中的处理器。由于使用通用处理器，省去了在处理器方面的开发费用，通过使用现有的组件，系统设计者可以对超级计算机进行快速的改进。一些计算机厂商，像Cray和NEC，继续在超级计算机上使用向量处理器，而不是通用的商业处理器，但是这些厂商目前仅仅占据很少的市场份额：截

至2006年11月，世界上最快的500台超级计算机中仅有7台是向量机[15]。

在过去的15年间，主要通过两个方面来提高峰值处理能力：加快单个处理器的时钟频率和增多处理器的数量。表3.1比较了两台分别是1993年和2006年全世界TOP500的机器。这个例子表明：增加处理器数目与增强时钟频率都是提高峰值的有效途径。这两种技术使得峰值性能增长了3110倍。

表 3.1　两台超级计算机的特点比较。Intel Delta 在 1996 年 6 月排第 8 位，Cray XT3 在 2006 年 11 月排第 2 位。Operon 系统中每个处理器有 2 个处理核心[15]

系统	处理器	CPUs	时钟频率	浮点指令峰值
Intel Delta	I860	512	40MHz	20.48×10^{9}
Cray XT3	Opteron	26 544	2.4GHz	127.4×10^{12}
增长率		51.8	60	6 220

虽然通过增强时钟频率就可以提高性能，但是现在，我们需要更多的处理器数目来发掘潜在的并行性。应用程序开发人员必须保证算法的规模和足够的并行度来使所有的处理器同时处于繁忙状态。

然而，单个处理器的性能很可能在未来的增长速度和过去二十年相比将逐渐放缓。增加性能最显而易见的方法就是提高CPU时钟频率，这种方法需要缩小芯片的体积。但是减少片上部件间的距离将会增加部件间的漏电，增加芯片的功耗，更坏的是这还会影响到芯片的散热。温度的增加将会降低CPU的可靠性并且会进一步导致漏电的增加。这些问题使得开发时钟频率高于4GHz的商用处理器变得很困难。其他的增加芯片计算能力的技术也不能使芯片性能得到很大的改进。

芯片设计者开始使用多核技术来提高性能[10]。目前许多处理器在一块芯片上整合两个计算核心。保持时钟频率不变，使用两个处理器核心，但是其芯片的能耗和使用一个单核两倍频率的芯片相比要低。当然为了使双核技术充分发挥就必须使两个核心并行工作成为可能。同时，主要的芯片供应商，像Intel、AMD和IBM已经开始销售或者宣布将会销售四核处理器。更高层次的芯片内部并行已经成为必然的发展趋势。

如果处理器的时钟频率仍然是确定的，基于商用处理的超级计算机只能通过增加并行度来提高性能。IBM的Blue Gene [9]是这方面的先行者。Blue Gene/L拥有65 536个处理器，每一个处理拥有两个IBM PowerPC共440个核心。每个处理器的频率仅仅是700MHz，这样的技术控制了元器件的开销和能耗。但是这样大规模的并行度对应用软件开发者提出了新的挑战，目前的软件并行度的数量级仅仅为10 000。章节3.5向用户展示了如何应对这样的挑战。随着核心数量级是100 000甚至是1 000 000，开发这样大规模的并行程序将成为主流。

3.3　适用于大规模并行机的编程模型

为目前的并行计算机所编写的代码使用的是数据并行模型，也就是单程序多数据（SPMD）。在这种模型中，所有的任务对数据的不同部分同时执行相同的算法。相对于数据并行，任务并行分配不同类型的任务给不同的进程或者线程。一种任务实现并行的方法是通过给一个进程中的多个线程分配独立的任务来处理同一个数据集。但是，科学计算通

常使用线程作为一种数据并行的扩展形式，比如说并发的运行独立的循环。

有效的数据并行程序开发的一个难点是使所有的处理核心同时保持繁忙。当一些任务需要等待其他任务完成，而等待时间又比较长的情况下，系统地整体性能会急剧的下降。随着并行度的增加，使一个程序保持负载均衡便愈来愈加困难：很难找到一种自然的方法把一个问题划分成100 000或者更多的可并发的子问题。即便是划分问题很简单，大量的任务间的通信代价将会消除掉并行所带来的性能优势。这一节的剩余部分介绍了几种有效使用大规模并行机的方法。

3.3.1 新型并行语言

一些研究小组正在为并行编程开发新的语言，以便可以加速应用软件的开发。他们的目的包括提高程序员的效率，并更加有效地使用并行资源和支持更高层次的抽象。在美国防御研究项目中开发出三种新的语言，它们是Chapel [5]、（Cray）、Fortress [1]（Sun）和X10 [6]（IBM）。尽管Chapel和X10中全局地址空间的分割对程序员是可见的，但是这三种语言在程序的地址空间方面仍然有独到的特点：这些语言同时包括了大量的表示循环并行和程序块并行的方法，在数据并行的基础上支持任务级并行，并且开发者设计了一个单独的程序来协调所有的操作。

尽管新的语言可以简化编写新的并行程序的工作，但是它们缺乏从现有的代码中抽取进一步并行性质的能力。在Lawrence Livermore美国国家实验室（LLNL），一些重要的模拟程序已经被连续开发了十年甚至更多年，每一个都有一百万行甚至更多的代码，用新的语言来重写这些程序，尽管可以提高效率，但是需要极高的代价，所以也就变得不可行了。

3.3.2 MPI-2

另外一种写一个任务的并行程序方法是利用MPI-2标准中的动态任务创建和单边通信的特点。MPI-2允许一段代码把自身划分成若干该部分，编译成不同的可执行文件。进程可以通过标准的MPI消息调用进行通信，也可以使用“单边”调用进行通信。所谓“单边”调用就是一个进程在另外一个进程中存储数据或者是通过远程获取数据。

尽管MPI-2具有基本的能力来实现任务并行程序，但是它是一个低等级的编程模型，并不像MPI标准那样被广泛的接受。MPI-2被广泛接受的一个障碍就是它出现的太晚了。尽管当1997年MPI-2商业化后很短的时间内已经有完整的实现出现，但是MPI-1实现仍然保持了广泛的应用。

3.3.3 协作式并行

为了提供不同的并行方式，LLNL正在开发一个叫做协作式并行[14]的新并行编程模型，这种模型可以被已经存在的应用程序采用。

协作并行是一种基于任务的并行模式，或者说是一种多程序多数据编程模型，它可以实现目前的数据并行模型。基本的计算单元是一个并行组件，该组件包括一个或者多个执行在一个或者多个结点上的进程。一个并行组件可以是一个单进程的程序，或者是一个数据并行程序。一个模拟组件有一组面向对象的接口，该接口定义了一组方法来执行任务。当一个程序开始执行时，它分配一组确定的结点和处理器，运行时系统在这些结点的一个子集上运行模拟组件。这个初始的模拟组件内的任何线程或者进程可以在其他处理器上运

行新的模拟组件，并且这些新的模拟组件可以处理新的运行请求。每一个模拟组件可以执行一个不同的可执行程序，这些可执行程序可以单独开发。

当一个线程执行一个模拟组件，它获得一个标志句柄。这个句柄可以被传递到其他的线程或者进程，甚至是其他的模拟组件。拥有这个句柄的任何线程可以对该句柄标志的模拟组件进行远程方法调用。远程方法被调用的方式有三种：阻塞式，非阻塞式和单向（one-way）方式调用。

除了它们的远程方法调用接口，模拟组件之间是透明的。它们不会共享内存或者交换消息。对一个模拟组件调用一个远程方法不需要预先安装，而是获得目标模拟组件的句柄，当一个远程方法调用结束之后，单向远程方法调用的状态信息不会再被保存，这种设计避免了潜在的规模扩展问题。否则当大量的模拟组件不得不维护所有的它调用过的或者将要调用的模拟组件的信息时，该问题规模将非常巨大，以至于很难进一步处理。

当发现错误或者是其他的一些原因，模拟组件可以终止其他的模拟组件，当一个模拟组件被终止，它的父组件将会被通知。Co-op使用Babel中间件来实现，该中间件提供RMI方法，并且允许用其他语言来编写这些方法。这意味着用C/C++和Fortran编写的组件可以在不知道对方的编程语言的情况下互相调用。

3.3.4　协作式并行的应用实例

互相协作的并行使得应用程序可以同时发掘一些不同种类的并行。这种灵活性的使用包括分析导致负载不平衡的因素和加强联合计算。

分析负载不均衡因素。一些仿真过程由于某个子问题需要附加的计算，从而导致结点间负载不均衡。如果每个时间步都需要程序同步，那么当一些子问题需要额外的计算时，其他所有的任务都需要等待。为了减少等待时间，开发者可以分配一些计算量给服务器组件池，就像图3.1描述的那样。这里，为运行在许多核心上的大模拟组件执行普通的数据并行计算。当有任务确定它需要附加的计算量时，它对服务器代理发送一个请求，然后服务器代理把该请求转发给池中可执行该计算的一个可用的服务器。对于整个模拟程序，除非调用者可以在等待结果的同时继续其他工作，那么这个请求就像一个简单的函数调用，它甚至可以在第一个申请在执行的时候再次提交一个非阻塞申请。服务器可以运行一组并行的模拟组件，这就加入了另一个层次的并行。处理器可以申请一些处理器来执行附加的计

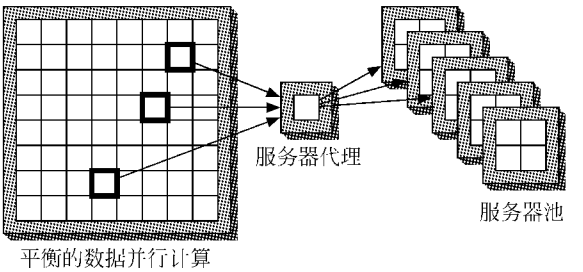

图3.1　互相协作的并行可以改善非平衡计算的性能。左边的方块描述了一组执行并行模拟数据的处理器。标记出来的方块是需要进一步处理的任务，他们把这些需要执行的处理转给右边的一个服务器池。每一个服务器自身是一个并行任务。一个服务器代理维护了服务器的空闲状态表并根据该表来进行任务分配。这种模式通过帮助复杂任务快速完成来该改进负载平衡

算量，并且调用者可以同时调用这些计算服务器，这使得调用者可以快速完成它的附加任务。这种方式缩短了空闲任务的等待时间，因此整体的负载均衡得到改进。我们希望这个方法对材料科学应用程序和一些其他的领域带来帮助。

联合计算。协作式并行还被用来在现有的数据并行代码的基础上来开发程序。比如说，机翼周围空气流的多物理模型可以把流动的动态模型和结构化的机械模型综合起来；或者是气候模拟程序可以综合大气模型，海洋模型和海洋冰川模型。在任一个情况下，一个独立的模拟组件可以描绘系统中一个独立的参数。这些模拟组件可以定期的处理RMI调用来更新中央数据库或是使用与它们目前状态有关的数据来更新一组独立的模拟组件。一个全面的解决方法是使用一个RMI的收集表格，在这个表格中调用者可以对多个接受者发送请求；但是，协作式并行目前并不支持这种模型。尽管联合模拟可以使用目前的技术进行开发，但是它们需要处理负载均衡等复杂的问题。协作式并行应该提供一个简单的方法来建立多物理模拟，从而对目前的单参数代码进行整合。

3.4 多尺度算法

对多物理、多尺度现象的千万亿次模拟需要可扩展的算法和程序代码，这可以有效地增加计算机的性能。尤其是联合模拟的每个模拟组件在它的处理器空间中必须是可扩展的。模拟组件的可扩展性可以通过互相协助模拟的方法，使用MPMD模型来实现。这一节讨论了可扩展算法在单一物理现象研究中的重要性，并对一些可以扩展到多物理研究领域的新型的方法进行了前瞻性讨论。

在一些有趣的科学应用中，一个经典的应用是解决PDE方程。离散化策略的选择和大量PDE求解是紧密耦合的，并且共同决定模拟结果的精确性和可扩展性。程序代码（包括离散和求解）必须在底层算法和并行实现两个方面是可扩展的。这可以使一个应用程序的时间开销在机器规模和问题规模成正比的情况下是恒定的。实际上，可扩展性负担主要在于求解的过程，这是下一节的主题。

3.4.1 并行的多重网格方法

多重格网方法在上世纪七十年代首次被提出，对于各种PDE方程是一个可能的最优解。这里的最优指的是算法具有$O(n)$的复杂性，n是离散空间网络上格网结点的数量。相比之下，其他的解决方法的复杂度是$O(n^2)$或者更差。多重格网解法的最优性可以转换成数学上的可扩展性：一个解决方案需要的任务总量正比例于问题的规模。如果这种数值方法可以有效地进行并行实现（比如说，通过重叠通信和计算），那么整体算法（和那部分应用程序代码）是可扩展的。

对多重格网进行细致的探讨超出了本章的范围，但是我们先简单阐述一下其核心思想：多重格网方法利用层次化的原始细粒度格网，成功的加速了小规模问题的求解过程。尤其是，它必须是定义了粗粒度化和延伸性操作，这样就可以把中间格网从一个层次映射到另外一个层次（粗粒度化将问题限定在更粗的格网上进行求解，而延伸性通过内插在更细粒度格网中进行问题求解）。这些操作是和问题本身高度相关的，不同的问题有着不同的求解方式。目前大多数数学多重格网的研究集中在真实程序中定义的操作，使得算法保

持了最优性特征（和数学上的可扩展性）。冲着这个目标，在过去十年中，多重格网领域有重大进展。AMG方法假设底层网络不是结构化的，他们依赖的是底层系统离散方程的代数特征。这个信息被用来定义粗粒度化和延伸性操作。研究者已成功地把AMG方法应用到各种具有挑战性的偏微分方程组PDEs的求解中，这些方程组是从许多不同类别的应用产生的无结构格网离散化得到的，这些应用包括天体物理学，结构力学和流体力学。

在过去的20年里，人们在提高并行可扩展性方面做了大量的工作。在串行AMG方法中，顺序性是问题的关键所在，它致使在大规模并行机上不能得到很好的并行化。虽然计算的复杂性是最优化的，但是并行计算机的存储和通信开销随着并行机规模的扩大，增长得也非常明显。目前，这个问题还没有解决，但是目前粗粒度化策略上的进展改善了复杂性和初始化的开销，主要手段是使存储需求减少了一半，把执行时间降低了一个数量级[8]。

并行的可扩展性问题在大规模并行机上变得更加重要，比如IBM的Blue Gene/L。考虑一个全局数据分布查询的操作：在一个传统的并行程序中（比如使用MPI的收集操作），这要求$O(p)$的存储和通信，其中p是处理器的数量。但是在一台拥有多于100 000个处理器（Blue Gene/L）（和未来的拥有1 000 000个处理器的千万亿次机器）的机器上，存储$O(p)$的数据是不切实际的。一个“想定划分”（assumed patition）的新颖算法[3]使用一个会合算法来响应拥有$O(1)$存储开销和$O(\log p)$计算开销的队列。

表3.2可以说明有效的并行多重格网方法的性能。它展示了Blue Gene/L上LLNL AMG solver的可扩展性：一个具有将近二十亿格网结点的问题在4秒钟内被解决。相对先前算法有16倍的提高是把改进的粗粒度化算法和前面提到的更快的通信例程综合起来的结果(注：底层算法仍然是数值性可扩展的）。

表 3.2　并行多重格网方法的执行时间（秒）。该表从两种数据查询技术比较了两种粗粒度化操作

		全局划分的数据查询		想定划分的数据查询	
处理器	未知数据	C-old	C-new	C-old	C-new
4 096	110.6M	12.42	3.06	12.32	2.86
64 000	1.73B	**67.19**	10.45	19.85	**4.23**

3.4.2　ALE-AMR 离散化

前面的篇幅主要是讲述对于偏微分方程求解器（PDE solver）数学和并行的性能来说，底层的空间格网是十分重要的。虽然我们很容易在结构化的Eulerian格网上定义和实现最佳的多重格网方法，但这些格网却不能很合理的代表比较重要的问题特征，如工程应用中复杂的移动部件。

为了消除这一缺陷，计算科学家一般采用下面两种方法中的一种来进行处理:自适应的格网细分（AMR）或是随机Lagrangian-Eulerian网格化方法（ALE）。在AMR中，根据对结果错误的不同反馈来细化格网。这使得通过较少的存储和计算开销就能获得更精化的格网精度。但是此时底层的格网依然保持本身的拓扑结构。在ALE方法中，格网会随问题的动态发展而进行移动，这使得对复杂的物理现象的追踪可以更为精确，但此时格网容易变得比较混乱，因此需要进行周期性的重新映射。如何构建一个强健的格网移动算法依然是ALE处理中最为关键的问题。

一个有趣的想法便是通过结合ALE和AMR以更好的适应多物理应用的千万亿次模拟。这一方法，被称为ALE-AMR方法，如图3.2所示。左图为标准的ALE，即初始的Eulerian格网，根据问题的动态变化而调整，最终格网必须被重新映射。中间临时格网可能出现扭曲元素，这会给数值计算带来困难。另一方面，它很合理的解决了一些复杂的问题，如震动前沿等等。右图是一个典型的AMR体系。格网被平均的剖分。在震动前沿处，剖分精度没有ALE高。中图为ALE-AMR方法，此处的关键在于划分格网时不采用形变的方法，而是采用动态增加或是删除点的方法，从而结合了两种方法的优点，同时也消除很多ALE中出现的格网扭曲问题。

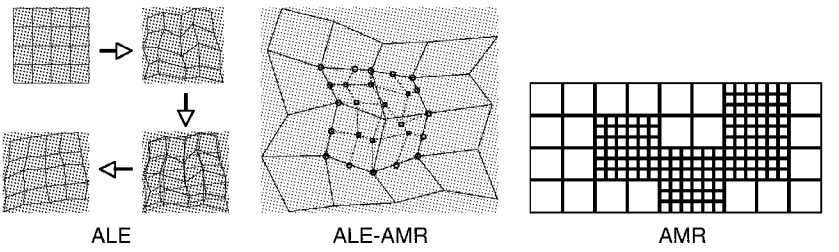

图 3.2 ALE-AMR（中间）把 ALE 格网的移动格网特点（左边）和 AMR 格网的自适应细分特点（右边）进行结合，得到了一种高效的格网离散技术，可以精确解决震动前沿一类的物理现象

ALE和AMR的的结合引出了许多正在进行研究的数值计算和软件开发等方面的问题。但是ALE-AMR算法的一些基础已经被建立起来了，但是如何融入更多特殊的物理能力，比如滑动表面，全局耦合扩散物理和复合材料的处理等不断给这一领域带来了新的研究挑战。

3.4.3 离散-连续统混合算法

到现在为止，我们的讨论还停留在连续方法之上，即用近似空间格网方案解决一些数值计算方法。然而在多物理的应用中，我们需要一系列的模型来精确的模拟底层的物理现象。这些应用可能包括一些连续和离散的方法的混合体。例如，如图3.3中所示，是两种液体混合时所引发的震动。连续计算流体动力学方法（如Euler和Navier-Stokes）虽能很好地描述远离表面的流体运动，但它们却受到运算格网最小粒度的限制。相比之下，离散的方法（如直接模拟蒙特卡罗法）虽能较好地解决两液体的相交面处的震动模拟，但是开销却太大。

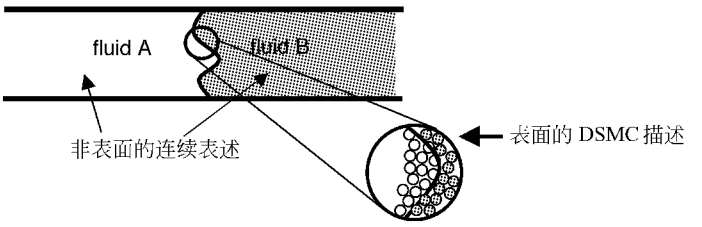

图 3.3 一种混合的多物理模拟演示。连续方法精确的描述液体在接触表面的运动，但是它需要一个离散的方法来模拟在接触表面的行为。因为在整个域中使用离散方法需要花费过高代价，所以混合算法的应用很具有优势

一些学者最近已经开始通过AMAR来研究混合的连续-离散方法。如以往章节所述，传统的AMR允许对一个围绕着动态运行和变化表明的连续运算进行细化，具体地说，就是AMR对感兴趣的特征附近的格网进行更细的剖分，如图3.3中的震动前沿，而且在细化后的格网内部采用同样的连续方法。而在混合算法中如AMAR中，处理粒度很细的格网尺度时，采用离散的原子方法。这使得人们可以在仅当必要的时候，才采用更合适（但是开销更大）地方法。如图3.4所示，它将离散方法嵌入在AMR格网层次体系中。

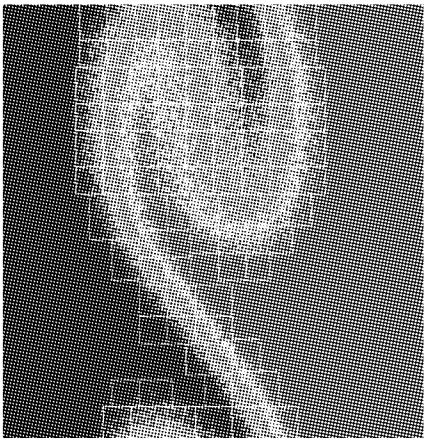

图 3.4　通过混合连续离散方法来模拟接触表面的运动。白色的格网表示在那里使用 Monte Carlo 方法来直接模拟，AMR 格网尺度来解决两种液体接触表面的物理现象。在其他部分使用连续尺度方法[13]

通过并行协作等一些MPMD编程模型，可以简化像AMAR这样的混合方法的实现。例如，你可以很方便地动态分配额外的处理器给更细粒度的格网，或是给直接蒙特卡罗模拟方法。虽然这些可以在数据并行的上下文中实现，但还是太过复杂，一个合理的MPMD编程模型应该使得这一切都更容易实现，但是这一切都还有待进一步的提高。

3.5　目前及将来的应用

Blue Gene/L这样的万亿次超级计算机的到来计算科学的一个新纪元，在这个新时代，科学模拟与仿真将真正成为与理论和实验并驾齐驱的方法，共同为科学发现作出贡献。在过去，仿真在很大程度上被看作是对理论的拓展，而且这些仿真通常缺乏基础设施，比如说，有限的计算资源限制了空间问题的解决。现今的超级计算机终于拥有了足够的计算能力（和存储），能够充分的仿真那些物理的现象——仿真那些经常能够提出新的理论或者指引未来的实验。这个部分将通过两个典型的应用例子来展示万亿次仿真技术的状态，它也通过对支持多物理仿真的协同并行方法应用的表述，对未来的千万亿次仿真提供了简单的概观。

3.5.1　万亿次仿真的技术现状

在3.2节描述的LLNL Blue Gene/L超级计算机在科学计算和计算科学中是一个里程碑。在之前的一个时期，Blue Gene/L是第一台在LINPACK平台上超越100TELOPS的计算机

[15]。它达到了280.6TFLOPS，达到了机器理论峰值的76%。它也是第一台同时使用超过100000内核的计算机，因此重新定义了“并行计算”的含义。这是一项十分伟大的成就，因为它向科学计算领域提出了更大的挑战，使得他们去重新考虑并行性和可扩展性。很多观测者认为Blue Gene/L是在万亿次计算进展过程中的一个里程碑。

计算科学里还有很多里程碑式的时刻，在2005年的11月，Blue Gene/L进行了第一次科学计算仿真，运行速度在100TFLOPS以上。不到一年，两个附加的应用在Blue Gene上的运行速度超过了100TFLOPS，其中一次到了207TFLOPS。事实上，最近的两个Gordon Bell奖的获得者是LLNL的ddcMD和Qbox这两个应用代码。这两个程序都是在达到极端情况下对材料特性的分子动力学仿真。MD程序比其他种类的程序特别适合这种机器。

世界上第一个100 TFLOPS可持续计算（ddcMD）。2005年的Gordon Bell奖授予了计算物理学家Fred Streitz带领的完成世界上第一个100 TFLOPS的可持续计算的团队[17]。他们使用准势能（pseudopotentials），通过利用经典的分子动力学方法来模拟钽和铀金属的凝固。他们运行了许多的模拟，其中一些超过了5亿个原子。这些仿真开始跨越原子到介子的尺度。

ddcMD代码以102 TFLOPS的速度运行了7个小时，达到理论峰值的28%。这个程序的性能是由IBM的专家进行调优的，它也说明对于不同的仿真将程序扩展到131 072个核心上运行的可行性。

仿真的结果对于我们是十分重要的。dddMD是科学家第一次用充足的计算能力（在Blue Gene/L）和可扩展的应用程序代码（ddcMD）来彻底的解决物理学上的问题。第一次正确的模拟了1 600万个原子的晶体形成过程，而之前在小型机器上最多只能模拟64 000个原子。为了仿真感兴趣的，就强加了许多周期性的边界条件，这就导致了在仿真结果中出现了非物理现象本身的周期性。我们可以很容易在图3.5中发现这两个仿真的不同，图中显

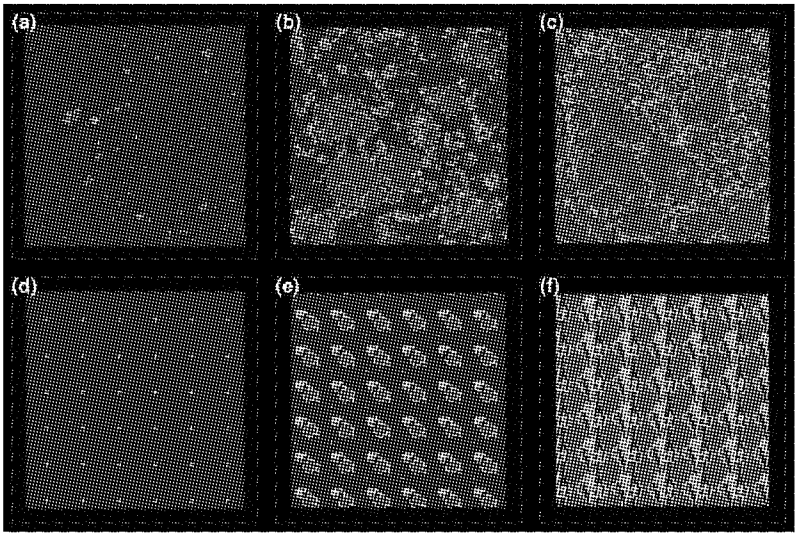

图 3.5 在熔化的金属元素钽中晶核形成及其成长过程的三维分子动力学模拟。在一个时间段里的三种照片显示了两种模拟。顶部对应着在 LLNL 的 Blue Gene/L 的超级计算机上模拟的 1 600 万原子。2005 年戈登贝尔奖获得者所进行的计算是第一个产生的物理上正确，与大小无关的结果。丰富的三维细节展现在平面切片上。而底部是在小型机上运行的 6 400 个原子模拟。周期性的边界条件用来生成完整的域，这导致了非物理现象的复制图案（图片来自 Streitz et al.[17]）

示了三个时间点的照片。在最上面一行可以看到平面切片上有丰富的三维细节信息，而在底部可以看到尚未仿真完成导致的复制模式。此外，他们还证明了对于该问题需要的原子个数不超过1 600万。

目前的世界纪录是207 TFLOPS（Qbox）的持续性能。2006年Gordon Bell奖授予了计算科学家Francois Gygi带领的完成207 TFLOPS鉬模拟的LLNL-UC Davis团队[12]。这个结果是目前在科学应用领域使用超级计算机达到性能的最高纪录。Qbox代码通过量子分子动力学来对物质的属性进行仿真。这是第一次基于密度函数理论（在平面波的基础上）和准势能（pseudopotentials）理论的Schrodinger 方程进行仿真计算。这部分具有多种功能的代码已经被应用在极端条件下（如高压和高温）的模拟凝聚态物质的各种应用。

Qbox代码是用C++和MPI编写的，在多物理条件下（平面波、电子态和k点）实现并行化。它使用了优化的ScaLAPACK和BLACS的线性代数方法，还有快速傅立叶变换，进行1 000个钼原子的测试。2005年11月，这个代码在Blue Gene/L机器上持续以64 TFLOPS速度运行。经过相当大的努力和调整，在不到一年的时间内代码已经达到以207 TFLOPS的速度运行。需要注意的是，这个代码的性能非常依赖于任务在处理器上的分布方式。最好的性能是在64×32×32处理器格网上，使用四分分布（quadpartite）方式取得了。

在以上描述的模拟成果后面很多重要的努力工作的作用是不能低估的：算法与计算机科学家很辛苦地在不断的调整这些代码以便代码能在全新的平台上性能最优。这些先驱者取得了荣誉，但是未来的成功衡量的标准将是进行类似的模拟可以通过怎样标准化而且很容易的方法来进行，下一节提到了实现这一目标的一种方法。

3.5.2　通过协作并行进行多物理模拟

前两个例子展现了在万亿次超级计算机上使用单物理（single-physics）码进行的大型仿真的技术。在许多我们感兴趣的科学应用中，我们需要整合多尺度与多物理范围而不是单一的领域。举例来说，计算科学家可能希望将现存的几份模拟代码整合变成一个单一的多物理仿真程序。正如前面所讨论，这些应用需要千万次计算的能力。问题是以怎样一种让应用开发者感到很友好的方式来充分利用这种计算能力。

其中一个方法，就是建立这种多物理应用代码的并行合作，比如在3.3节讨论的MPMD程序模型。虽然这个程序模型目前还处于早期发展阶段，但是已经做出的一些工作显示出它可以促进多物理应用的开发。值得一提的是，一个团队已经修改了其材料模型代码，使得粗粒度的材料计算可以使用细粒度的多晶体仿真计算所得到的参数。如果每次都要重新计算他们需要的细粒度参数，这些运算将使得仿真的执行时间非常长，这样就使得整个仿真不能在合理的时间内完成。但是，如果采用自适应性的采样（adaptive sampling）技术（来自最初的自适应制表技术[16]），仿真就能消除许多此类代价很高的计算。这是通过把前面细粒度计算的结果存在数据库里实现的，这样如果错误不是太大，后续的细粒度计算就可以在前面存储结果的基础上，通过内插或者外插的方法来得到。当必须要执行完整的细粒度计算的时候，仿真器就把任务分配给一个服务器来执行，如图3.1中所示。至今为止，运行这个应用的核心数目在超过4 000个，但是，减少细粒度计算（因为自适应采样）以及改进负载平衡（由于服务器池模型）的这两方面的共同努力已经表明其性能可以有1～2个数量级的提高，这种不同依赖于问题本身以及理想的精度要求。此外，该应用也展示了