学　　院	
编　　号	

西安交通大学

硕士研究生学位论文选题报告

学　　号：	3118311269
研 究 生：	张钰鑫
导　　师：	张兴军
论文题目：	面向众核架构的 CFD 多线程动态映射机制研究与实现
学　　科：	软件工程
填写时间：	年　　月　　日

西安交通大学研究生院制

硕士研究生学位论文选题报告管理规定

硕士学位论文选题报告是做好学位论文的基础，为了完善硕士研究生过程质量监控体系，提高硕士研究生培养质量，要求在校硕士生必须在入学第三学期内，完成学位论文选题报告。

一、硕士生在查阅一定的国内外文献资料基础上，填写完成《硕士士研究生学位论文选题报告》。

二、《硕士研究生学位论文选题报告》完成以后，应组织公开的学位论文选题报告会。

三、选题报告会由系、所以上单位负责组织，至少由 3 名副高以上人员组成，其中 1 名担任组长。并在选题报告会后负责就选题的意义、文献综述、研究内容、可能遇到的问题、是否同意选题等写出结论性的审查意见，并将结果和相关材料留学院备案。

四、《硕士研究生学位论文选题报告》由各学院归档。

五、《硕士研究生学位论文选题报告》必须采用 A4 纸双面打印，左侧装订成册，各栏空格不够时，请自行加页。格式可在研究生院主页下载。

论文题目：面向众核架构的 CFD 多线程动态映射机制研究与实现

论文类型：（1）基础研究；（2）应用基础研究；（3）应用研究；（4）其它
课题来源：（1）纵向课题；（2）横向课题；（3）自选课题；（4）其它

一、选题的科学依据（1、选题背景；2、理论意义和应用价值；3、国内外研究现状及发展趋势。附主要参考文献及出处）

1、选题背景

近年来，随着高性能计算不断发展，为匹配日益增长的计算需求，设计者在最初多核处理器计算机的基础上，扩展计算核心的数量，推出了规模更大、计算能力更强的众核处理器计算机，其大规模并行计算能力甚至达到 E 级以上。如今，很多大型的科学计算问题都需要使用众核计算平台，结合高性能计算技术，以最优的性能解决复杂的计算问题。众核系统拥有众多的计算核心，通过将计算任务分配给不同的处理核心可以大幅提高计算性能，因此如何合理高效进行线程调度，确保全部处理核心处于有效工作状态是当今多核系统研究的一个重要方向。

计算流体力学（CFD，Computational Fluid Dynamics）是高性能计算领域的重要应用之一[1]，很多问题根据离散化的数值计算方法，利用计算机对流体的流动特性等参数进行数值模拟。在多数 CFD 数值模拟程序中，通过前处理过程建立的网格模型复杂且网格规模庞大，为求解计算过程引入了庞大的数据量和计算量。

目前，众核计算平台提供的硬件资源已经能够满足一定量的计算需求，然而从软件上看，算法开发者也需要对应用程序做出适当调整以匹配硬件资源。在众核系统中，由于缺乏较好的调度机制和优化策略，CFD 应用程序不能充分利用众核系统的计算资源，导致系统整体性能不佳。在多线程编程技术中，目前利用较为普遍的 OpenMP 并行编程模型，在较细粒度能一定程度优化程序的执行效率，但多数情况编程人员只能使用手动编码的方法实现优化，程序的运行性能无法在运行时持续最佳状态的弊端，并且由于程序中复杂的数据依赖关系，仅仅依靠操作系统调度实现的映射使得处理器的计算资源得不到充分利用，并可能产生负载不均衡等问题。所以在现有多线程并行应用程序的基础上，实现动态的映射优化非常有研究意义与商业价值。

2、理论意义和应用价值

在高性能计算技术领域，映射问题一直是被广泛关注的课题。如何充分利用众核系统的计算资源，如何合理将计算任务映射到计算资源上，实现提升计算性能、降低系统能耗、提高系统可靠性是高性能计算研究领域的核心内容之一。同时，基于众核处理器的系统架构，需要针对共享缓存的通信模式，对线程访问共享数据问题进行优化。结合这两点关键问题，研究线程到处理核心的关键性、创新性的映射优化具有重大的意义。此外，研究过程要适应具体的众核计算机体系结构，同时达到有效提升访存效率和程序整体执行效率的目标。

CFD 方法在实际工程应用中逐渐发挥着重要的作用，如果能针对面向众核计算平台的 CFD 应用程序，实现良好的映射优化，不仅能更快更好地执行 CFD 数值模拟，更有助于将映射方案扩展到更多类似的依赖众核计算平台解决的大型计算问题中。

3、国内外研究现状及发展趋势

基于共享内存通信模式的线程映射问题是优化多线程程序并行效率的研究重点。大多数情况下，线程映射通过检测不同线程之间的通信情况来执行。将检测到的信息传递给执行映射的算法，映射算法综合考虑硬件拓扑信息和通信情况信息实时优化线程映射。

许多研究工作不断进行实验与探索，最终得出上述较为普遍且应用性强的映射方

法。最初研究人员通过选择在多种不同的映射方式中依据检测出的性能参数选择出最优的映射。Autopin[2]方法通过测量 IPC (Instructions per Cycle，每个时钟周期执行的指令数) 的值，选择具有最高 IPC 值的线程映射方案来执行应用程序。在研究[3]中提出了 BlackBox，是一种类似于 Autopin 的调度程序，同样在提供几种映射中通过测量每种映射获得的总体运行性能来选择最佳映射。根据组合的数学原理，当线程数不多时，可选择的线程映射方案本身数量较少，研究人员可以通过评估所有可能的映射方案；当线程数急剧增加，则无法评估所有的可能情况。BlackBox 选择在 1000 个随机的映射可能中挑选其中最佳的方案执行应用程序，然而该方法的稳定性较差。

研究发现，影响多线程执行效率最主要因素是访存性能，因此对于线程映射来说，最重要的信息是线程对共享数据的内存访问量，实际上与数据的内存地址并不相关。基于共享模式矩阵[4]的方法，研究者们得以分析线程映射前后内存的局部性。Cruz 团队[5]依据 NUMA 多核架构下的缓存一致性，提出了基于写请求的线程访问共享数据的统计方法，为执行映射算法提供了有力依据。此外，一些研究者采用硬件计数器间接统计通信情况，如 Azimi R 的工作[6]，它使用硬件计数器对每个拥有 L1 独立 cache 的核进行统计，记录远端核心对共享内存段产生的内存访问情况。然而这种方法检测的通信模式不完整，它没有考虑由本地 cache 解析产生的内存请求。此外，SPCD[7]和 kMAF[8]的映射方法利用并行程序产生的页错误检测通信情况，这种机制利用 pin 工具[9]跟踪产生页错误的线程以及错误发生的具体内存页，将具有大量共享页的线程放置在内存层次结构中彼此靠近的位置。

在对线程的分组和放置进行重新计算的映射算法中，目前研究主要采用基于图划分的算法思路。例如 Zoltan[10]和 Scotch[11]，他们使用通用的带权图表示通信关系，图的顶点表示不同线程，边表示线程间通信，权值表示通信量。然而基于图划分的映射问题属于 NP-hard 问题，只能以启发式方法进行解决。TreeMatch[12]算法使用了树形的表示方法，从映射结果看是性能更优化的算法。但是 TreeMatch 为内存层次结构的每一级都生成所有可能的组，因此该算法具有指数级的时间复杂度。同时 TreeMatch 不支持将多个线程映射到同一 PU (处理单元)上。另一种基于内存层次结构的通用性更强的映射算法 EagerMap[13]采用了迭代分组的策略，运用贪心算法的思想进行分组，获得了不错的性能。

多线程并行程序的映射工作中，动态的方法寥寥无几。近年来，随着需求的不断提高，在程序中以动态方式解决映射问题的研究也越来越多。上述的方法[5]和[8]都是代表性较强的动态映射方法。此外，还有研究者如 Matheus[14]将映射与机器学习方法相结合，提出了一种全新的解决方案等等。

在多核、众核处理为核心构建的计算平台下，多线程映射优化研究工作都各有特色，且体现出良好的优化效果。但对于在众核计算平台上运行的具体的大型 CFD 应用程序，鲜有能有效提高多线程访存效率和程序整体并行效率的出色、创新的动态映射优化机制。这正是我们需要经过研究解决的重点问题。

参考文献：

[1] Chung T J. Computational fluid dynamics[M]. Cambridge university press, 2010.

[2] Klug T , Ott M , Weidendorfer J , et al. autopin: automated optimization of thread-to-core pinning on multicore systems[C] // Transactions on High-performance Embedded Architectures & Compilers III. Springer-Verlag, 2011.

[3] Thread Assignment of Multithreaded Network Applications in Multicore/Multithreaded Processors[J]. IEEE Transactions on Parallel and Distributed Systems, 2013, 24(12):2513-2525.

[4] Diener M , Cruz E H M , Navaux P O A , et al. Communication-aware process and thread mapping using online communication detection[J]. Parallel Computing, 2015, 43:43-63.
[5] Cruz E H M, Diener M, Alves M A Z, et al. Dynamic thread mapping of shared memory applications by exploiting cache coherence protocols[J]. Journal of Parallel & Distributed Computing, 2014, 74(3):2215-2228.
[6] Azimi R , Tam D K , Soares L , et al. Enhancing operating system support for multicore processors by using hardware performance monitoring[J]. ACM SIGOPS Operating Systems Review, 2009, 43(2):56.
[7] Diener M , Cruz E H M , Navaux P O A . Communication-based mapping using shared pages[J]. IEEE 27TH INTERNATIONAL PARALLEL AND DISTRIBUTED PROCESSING SYMPOSIUM; (IPDPS 2013), 2013:700-711.
[8] Diener M , Cruz E , Alves M , et al. 。1[J]. IEEE Transactions on Parallel & Distributed Systems, 2016, 27(9):2653-2666.
[9] Luk C K , Cohn R S , Muth R , et al. 9 8 Pin: Building Customized Program Analysis Tools with Dynamic Instrumentation[C]// Proceedings of the ACM SIGPLAN 2005 Conference on Programming Language Design and Implementation, Chicago, IL, USA, June 12-15, 2005. ACM, 2005.
[10] K. D. Devine, E. G. Boman, R. T. Heaphy, R. H. Bisseling, and U. V. Catalyurek. Parallel hypergraph partitioning for scientific computing[C]// IEEE International Parallel & Distributed Processing Symposium (IPDPS), 2006.
[11] Fran ̧ois Pellegrini, Roman J . SCOTCH: A Software Package for Static Mapping by Dual Recursive Bipartitioning of Process and Architecture Graphs[C]// High-Performance Computing and Networking, International Conference and Exhibition, HPCN Europe 1996, Brussels, Belgium, April 15-19, 1996, Proceedings. Springer-Verlag, 1996.
[12] Jeannot E , Mercier G , Tessier F . Process Placement in Multicore Clusters:Algorithmic Issues and Practical Techniques[J]. IEEE Transactions on Parallel and Distributed Systems, 2014, 25(4):993-1002.
[13] E. H. M. Cruz, M. Diener, L. L. Pilla, and P. O. A. Navaux, An efficient algorithm for communication-based task mapping[J]. in Proc. Int. Conf. Parallel, Distrib. Netw.-Based Process., 2015, pp. 207–214.
[14] Serpa M S , Krause A M , Cruz E H M , et al. [IEEE 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP) - Cambridge, United Kingdom (2018.3.21-2018.3.23)] 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP) - Optimizing Machine Learning Algorithms on Multi-Core and Many-Core Architectures Using Thread and Data Mapping[J]. 2018:329-333.

二、主要研究内容和方案

以 OpenMP 编程模型为基础，将 CFD 数值模拟方法的串行应用程序并行化，就得到了经过细粒度并行的多线程并行程序，可以发挥众核 CPU 的性能。然而用高性能计算技术解决 CFD 问题时，将求解过程划分为前处理、求解器和后处理多个基本模块，其中前处理技术构建网格模型，使得求解时每个网格块上的边界数据需要与其相邻网格块中的数据进行交换，由此产生复杂的数据交换和数据共享。

在 OpenMP 编程模型中，数据交换和数据共享通过共享缓存的通信模式实现。在实际应用中，仅使用 OpenMP 对网格规模和计算量比较大的应用程序进行手动并行编程之时，往往无法考虑复杂的数据交换和数据共享对内存访问的影响。在这种情况下，执行任务的线程并未根据程序运行时数据实际共享的情况，在合适的 CPU 计算核心中执行。因此在实际并行程序运行过程中，常常出现较多的缓存失效，跨处理器甚至跨节点内存读写的问题。为解决这一影响程序整体执行性能和能耗的问题，需要在程序运行时根据实际的数据交换和数据共享情况，优化线程到处理器核心的映射，使线程间访存通信过程得到优化。

同时，通过实验发现 CFD 应用程序在执行过程中，数据通信模式并不固定，往往会因为代码中的计算需求发生明显变化。因此需要根据运行时不断变化的程序运行状态，动态地对映射作出调整，以适应程序的运行时特点。

此外，在实际应用场景中，CFD 应用程序往往具有较大的计算规模，为了以性能更好的方式完成数值计算，所需核心数往往到达成千上万的数量级。因此，对映射的优化研究是否能随核心数的增加而不断扩展，也是我们需要考虑的关键问题。

1、主要研究内容

(1) 动态多线程映射机制研究

动态多线程映射对应用程序中线程间通信以及访存性能进行优化，达到有效提升访存效率和程序整体执行效率的目标。在研究过程中，运行平台的体系结构、应用程序的访存特性、动态地调整线程到核的映射情况，是设计出有效动态映射的三个关键因素。其中程序的访存特性表示执行的线程对共享缓存的访问情况，以及多个线程访问共享数据产生的通信情况。访存特性会随着程序运行发生改变，相应的线程分配也要做出调整，这也就是设计动态映射的原因。

(2) CFD 应用程序多线程间通信量检测方法研究

在这一部分的研究过程中，需要结合具体的 CFD 应用程序，对其进行多线程映射优化。采用 OpenMP 模型并行化的 CFD 应用程序，数据交换和数据共享通过共享缓存的通信模式实现。共享内存的通信模式对性能产生的影响更多地体现在节点内部。在并行计算机体系结构下，一个节点内的内存结构呈现出层次化的特点，即存在多个核心共享一个某一级缓存的结构。所以，我们需要实现寻找额外开销更小，准确性更高的线程间的通信量检测方法。

(3) 动态映射机制在众核计算机上的可扩展性研究

由于大部分 CFD 应用程序需要更多的核心数以支持计算需求，应用程序在运行时通常是跨节点运行，甚至在超算平台上以跨机柜、跨分区的方式运行。因此在实现了节点内部的线程到计算核心的映射优化后，为使应用程序在整体运行时都能得到线程映射级别的优化，应当考虑该映射优化方法是否可以随核数和节点数的增多而同样适用。为此，我们需要研究动态映射机制的可扩展性。

2、研究方案

根据上述研究内容，研究方案分为以下几个部分：

(1) 确定动态多线程映射优化整体流程

为了实现动态的线程到核的映射优化，需要确定一个在程序运行时能动态执行映

射的框架。这个框架包含动态地检测应用程序的运行时线程间通信，设计算法完成对线程分组划分，动态执行优化后的映射。

(2) 检测记录节点内线程间通信量信息

为优化通信、提高访存效率，在优化线程到核映射过程的初始阶段，我们需要首先检测节点内线程之间依靠共享数据进行通信的通信量信息，以此信息来作为判断程序运行时状态的前提信息。这个过程不仅包括利用工具检测得到我们想要的通信量信息，还包括用一个数据结构（向量、矩阵等）统计出检测到的信息。这个统计结果将是判断程序运行时状态，执行线程映射优化的关键数据。

(3) 设计合适的线程分组划分方法

在程序执行状态不佳的情况下，需要设计合理的映射算法，将线程以成组映射的方式绑定到具体的计算核心上，让一组通信密集、处理同一块共享数据的线程映射到共享同一级缓存或同一个内存单元的核心上，这就是线程分组映射方法的研究。

(4) 研究如何动态执行映射

优化后的映射方式一定不同于操作系统本身的调度方式，因此在动态执行映射时，需要更改线程到核的绑定方式。在不影响程序本身运行结果的前提下，动态地迁移线程，使之匹配计算好的映射优化规则。一般情况下，需要改变多线程并行程序的运行时环境。

(5) 研究上述动态映射机制在众核平台的可扩展性

研究如何实现将整体的动态映射优化机制实现在核数更多、节点更多的众核计算平台上，从而实现映射优化的可扩展性。针对具体的 CFD 应用程序，可通过改变执行的线程数、修改并行程序代码等方法研究可扩展性的实现工作。

三、研究计划及预期进展（应该规划到完成毕业论文吧）

时间	研究内容	预期效果
2020.4~2020.6	动态线程到核映射优化机制的研究	分析可行性及整体开销。明确建立动态映射机制的框架，以及框架内各个模块的方法。
2020.7~2020.9	节点内线程到计算核心的映射优化研究	前期有效组织动态机制的各个模块；

		后期实现针对 CFD 应用程序的节点内动态映射优化，实验结果良好
2020.10~2020.12	动态映射机制在众核平台上的可扩展性研究	前期调研多线程的可扩展性。后期实现针对 CFD 应用程序动态映射方法的可扩展性，将方法实现在 E 级超算平台。

四、指导教师意见：

选题结合国家重点研发计划课题，属于学科前沿问题。选题的科学依据阐述清楚，主要研究内容明确，研究方案具体、合理。

同意开题。

签名： 

日期：2020.3.25

五、选题汇报记录（着重记录对选题报告提出意见、评价、问题及对问题的解答等，如要修改时，指出修改的部分）

六、论文选题汇报评语（由参加开题报告的专家、教授填写）

开题报告小组签名：组长_____

成员_____ _____ _____ _____

时间：　　年　　月　　日