2008年 11月 第 11期
Vol. 29 No. 11 2008

非规则、核外并行计算研究综述

胡长军,张纪林,王 珏,李建江

(北京科技大学 信息工程学院,北京 100083)
E-mail: zhangjilin.bj@gmail.com

摘 要: 非规则、核外计算既是大规模并行应用普遍存在的问题,也是影响大规模并行应用效率的关键问题.本文从并行处理模型、运行支持库实现和并行优化三个方面对非规则、核外计算技术进行了全面综述,并对典型研究成果的特点和不足进行了分析.如何充分利用系统结构的特点和应用数据本身的特点,寻求非规则、核外计算处理的优化是现有技术发展的共同思想.在此基础上,指出了处理两类问题的技术相关性以及在SMP集群系统结构和网络存储环境下,解决非规则、核外计算的新思路:一是从问题描述、编译优化、运行支持等多层次协同研究充分利用系统结构特点的优化技术,二是从应用问题出发,在并行粒度确定、并行范例选择等方面统一非规则、核外计算的处理,三是研究新的支持非规则通信的优化技术和动态负载均衡方法.

关 键 词: 非规则计算;核外计算;并行模型;综述
中图分类号: TP31 文献标识码: A 文 章 编 号: 1000-1220(2008)11-1969-10

Overview of Technologies for Irregular and Out-of-Core Parallel Computing

HU Chang-jun, ZHANG Ji-lin, WANG Jue, LI Jian-jiang
(*Information Engineering School, University of Science and Technology Beijing, Beijing* 100083, *China*)

Abstract Irregular and out-of-core (ooc) computing significantly influence the performance of large scale parallel systems. From parallel model, implementation of runtime library and parallel optimization, this paper summarizes technologies for both problems, and gives characteristics and shortcomings of typical works. Much research has focused on how to fully exploit the potential of the system architecture and features of data structure for optimization of irregular and ooc parallel computing. Finally, based on above research, we give their relationship and trends under the SMP cluster and network storage. Firstly, in order to fully utilize the characteristics of the system architecture, we research the optimizing technologies by cooperating of representation of problems, compiler optimizations and run-time support, etc. Secondly, in application, combining the irregular with ooc computing by determining parallel paradigms and grains. Thirdly, exploiting new optimizing technology for irregular communication and new methods of dynamic load balance.

Key words irregular computing; out-of-core computing; parallel computing model; overview

1 引 言

当前,大规模并行应用在多数高性能计算机中的实际应用性能不到峰值性能的10%,其巨大差距已成为高性能计算领域特别关注的问题[1].造成这种差距的一个原因是:非规则、核外计算普遍共存于大规模并行应用中,并且是影响并行应用效率的关键问题[2].例如对于图1所示的程序片段,并行编译器的静态分析与优化方法都无法预知数据的存取模式和通信模式,因此静态分析方法无法针对这种非规则引用模式进行优化.不仅如此,大规模并行应用涉及的数据量是惊人的,内存容量往往不能满足涉及大数据量计算问题的存储要求.当程序运行时,某个时刻只能有部分数据被调入内存并参加计算,并且在适当的时候写回外存,通过内外存的数据交换来完成整个计算,这就是核外计算处理技术[3].显然,核外计算处理技术也是影响并行应用性能的关键.实际上,核外计算和非规则计算的解决在技术上是密切相关的.核外计算处理模型是非规则计算处理模型的基础,从本质上来说,非规则计

```
DO I= IBNONB,NBC
  DENBIO(I)= DNOILC(IKCB(I))* 2.309
END DO
```

图 1 某生产用油藏数值模拟程序中的非规则计算程序片段

Fig. 1 Code segments for irregular computing in petroleum reservoir simulation

算的处理依赖于负载的动态调度与均衡策略,依赖于对不同负载粒度引入的开销和负载迁移代价等的估算,二者都与核外计算的I/O模型密切相关.所以需要以提高大规模并行应用效率为目的,将两者统一起来研究.

收稿日期: 2007-07-03 基金项目: 国家"八六三"高技术研究发展计划基金项目(2006AA01Z105)资助;国家自然科学基金项目(60373008)资助;教育部科学技术研究重点项目(106019)资助. 作者简介: 胡长军,男,1963年生,博士,教授,博士生导师,主要研究方向为并行计算与并行编译技术,并行软件工程、网络存储体系结构、数据工程与软件工程;张纪林,男,1980年生,博士研究生,主要研究方向为并行计算与并行编译技术;王 珏,男,1981年生,博士研究生,主要研究方向为并行计算与并行编译技术;李建江,男,1971年生,博士,副教授,主要研究方向为并行计算、并行编译与多线程技术.

基于 SMP集群系统结构和网络存储环境来优化非规则、核外计算日益成为研究热点[4,5,8].与以往基于分布内存系统结构的非规则、核外计算处理技术不同,SMP集群同时具有节点内共享存储和节点间分布存储的特点,可扩展性好且结点内处理器间通信开销低.网络存储提供了高带宽、低延迟、大容量数据安全性存储和大流量数据传输的环境.如何利用 SMP集群系统结构和网络存储环境的这些新特点来优化非规则、核外计算的处理是提高并行应用系统性能的突破点.

接下来,本文对近年来非规则、核外计算并行优化方面的重要研究成果进行了综述,并分析各种技术的优缺点,对技术的进一步发展提出预测.本文其余部分的组织如下:第二、三部分分别综述非规则计算、核外计算及其并行优化模型;第四部分总结全文并提出对非规则、核外计算的进一步研究思路.

2 非规则计算及其并行模型

数组存取非规则问题是指:在循环中,数组存取的下标是循环索引变量的非封闭表达式,或者数组的下标表达式包含处理循环索引变量的函数或数组[6].有两种典型的表现形式:非规则单索引存取和简单间接数组存取,它们的定义分别如下:

定义 1.在给定循环中,如果数组下标始终由同一循环索引变量表示,并且存取是非规则的,则称这类数组的存取是非规则单索引存取.

定义 2.若数组 A的下标表达式包含其它数组 B,则称数组 A的引用是间接存取的.数组 A称为"主数组",数组 B称为"索引数组".若数组在 Do循环中,并且索引数组的下标是循环索引变量,则称为数组 A的存取是简单间接数组存取.

对非规则计算的研究主要集中在索引数组存取方面.为了对非规则计算进行合理且有效的并行化,需要解决如下关键技术:

(1) 在非规则计算过程中,动态地确定数据的存取模式;

(2) 建立全局地址空间和局部地址空间的对应关系以及查找不同的地址空间;

(3) 根据数据的存取模式动态地产生通信,并优化通信以进一步提高系统的性能;

(4) 在非规则计算过程中,动态地均衡负载;

(5) 自动生成非规则计算的 SPMD程序.

非规则计算并行模型包括 PW(Post/Wait)模型、SE(Speculative Executor)模型,IE(Inspector/Executor)模型和 EIE(Extend Inspector/Executor)模型.

2.1 PW模型与 SE模型

针对循环中存在提前退出语句(if… then goto/break语句)的非规则计算,文[6,7,34]论述了 PW模型和 SE模型.在 PW模型中,程序是 SPMD模式,并假设没有迭代相关.每个处理器通过插入 Post/Wait原语进行调度和同步.如图 2中阶段 1所示第 i次迭代必须接受到 $i-1$次迭代发送的 Post消息后才开始执行,如果第 $i-1$次迭代(在其它处理器上执行)没有提前退出,那么本次迭代将会继续执行,否则向第 $i+1$次迭代发送 Post消息,并且退出循环(见阶段 2).处理器检测提前退出条件(见阶段 3),若条件成立,则设置终止变量,向第 $i+1$次迭代发送 Post消息并终止循环(见阶段 4).若条件不成立,处理器则向第 $i+1$次迭代发送 Post消息并且继续执行当前迭代(见阶段 5).

```
Doall i= 1,n
Wait t(i)                                  — (阶段 1)
  If (terminate) then
  Post t(i+ 1)                             — (阶段 2)
Quit
End if
/* 计算第一部分* /
Compute_ part_ 1
/* 检测提前退出* /
If (x[pos[i]]) then                        — (阶段 3)
  Terminate= true
  Post t(i+ 1)                             — (阶段 4)
  Quit
Else
  Post t(i+ 1)                             — (阶段 5)
End If
/* 计算第一部分* /
Compute_ part_ 2
End do
```

图 2 PW模型实例

Fig. 2 An instance of PW model

而在 SE模型中,执行过程可分为 8步:

Step 1. 将迭代空间 n划分为 m个迭代阶段;

Step 2. 每个迭代阶段划分为若干迭代块,并将其依次均分到各个处理器中;

Step 3. 所有处理器并行地从第一迭代阶段执行迭代块,当第一阶段执行完毕后,继续执行第二迭代阶段;

Step 4. 全局变量 last_ iter、last_ stage和 last_ proc分别记录循环退出时的当前最小迭代、最小阶段和处理器 ID,初始值分别为 $n+1$、$m+1$和 0;

Step 5. 处理器在执行迭代之前首先比较当前迭代次数和 last_ iter变量.若前者大,则处理器退出循环并进入第 8步全局规约阶段;

Step 6. 在并行执行迭代过程中,若提前退出条件成立,则处理器比较当前迭代次数和变量 last_ iter的值.若前者小于后者,则修改变量 last_ iter、last_ stage和 last_ proc分别为当前迭代次数、当前迭代阶段数和当前处理器 ID.处理器退出循环并进入第 8步全局规约阶段;

Step 7. 在并行执行迭代过程中,处理器将局部规约结果 X写入本地拷贝 $X_{i,j}$(处理器 j中第 i迭代阶段规约值为 X)中.每个处理器的每个迭代阶段对应一个变量本地拷贝;

Step 8. 所有处理器均退出并行规约阶段后进入全局规约阶段,合并 $X_{i,j}$和 $X_{last\_stage,j}$.

在 SE模型中,所有处理器都并行执行,并将规约结果写入本地拷贝中,退出循环后把所有本地拷贝的内容提交给全

局变量.文[34]改进了SE模型,提出在并行执行循环的同时分析迭代之间的数据相关性,如果分析表明循环不能并行化,则整个计算将返回到起始点并串行执行,这种机制的实现起来比较困难.SE模型的主要缺点是当循环不能并行执行时,相关负载很大.

2.2 IE模型

非规则计算的核心模型是IE模型[9,10],其分为两个阶段,分别是Inspector阶段和Executor阶段,前者分析索引数组并生成通信调度,后者使用通信调度进行通信并完成计算.

Inspector阶段开销很大,但是通过对其优化可以提高IE模型的性能.模型性能的提高需要考虑五个方面:数据划分、计算划分、局部性、负载均衡和通信.

2.2.1 数据划分

数据划分方法除了规则的Block和Cyclic划分,近几年研究人员针对非规则计算的不确定性研究了多种启发式方法,根据不同的标准(空间位置、连通性和计算负载等)进行划分,实现了非规则的数据划分[9-11].此外,Gagan Agrawal第一次提出了自动数据划分方法,通过静态分析以确定数据划分,该方法通过启发式贪婪算法选择合适的数据划分[12].David E. Singh等人提出LCYT和LO-LCYT两种数据划分算法[13],前者利用无向加权图表示内存存取,并根据数据相关性对图进行划分,而后者则在前者的基础上优化内存存取,提高了空间和时间局部性.数据划分方法非常复杂,需要考虑向量权重、最终划分的数目和负载均衡标准等问题,其优劣直接影响模型的性能.

在数据划分中,对于采用非规则划分方式划分的数组,需要对每一个数组元素建立转换表以便于进行全局下标和局部下标之间的转换.使用转换表时存在两部分开销:建立转换表和查找转换表.当数据不在本机时需要进行通信,时间开销比较大,包括通信时间和非本机数据的查找时间.数据划分方式决定了数据的局部性和处理器之间的通信模式,会影响非本机数据引用的次数及效率,因此为了减少时间开销,一方面应根据本机所引用的数据进行数据划分,另一方面应恰当地使用软件缓存转换表、页式转换表和散列式转换表[9].

2.2.2 计算划分

根据内存结构的不同可将典型的计算划分算法分为两类:一类是在分布内存系统结构下,由CHAOS库实现的拥有者计算原则(Owner-Computes Rule)和大多数拥有者计算原则(Almost-Owner-Computes Rule)[10].但以上两种方法通信负载较大,因此Minyi Guo等人针对非线性数组下标的非规则循环提出了最小化通信原则[14].另一类是在共享内存系统结构下,由C-W. Tseng等人论述的Replicate Bufs方法和本地写方法[15],前者实现每个处理器计算部分迭代并将结果记入本地内存,然后将所有结果汇总到全局变量中.后者类似于分布内存系统结构中用到的拥有者计算原则,仅对本地数据赋值,不需要缓存和互斥同步[16].此外,还有E. L Zapata等人提出的DWA-LIP方法,在最大并行化的同时提高数据引用的局部性[17].

2.2.3 局部性

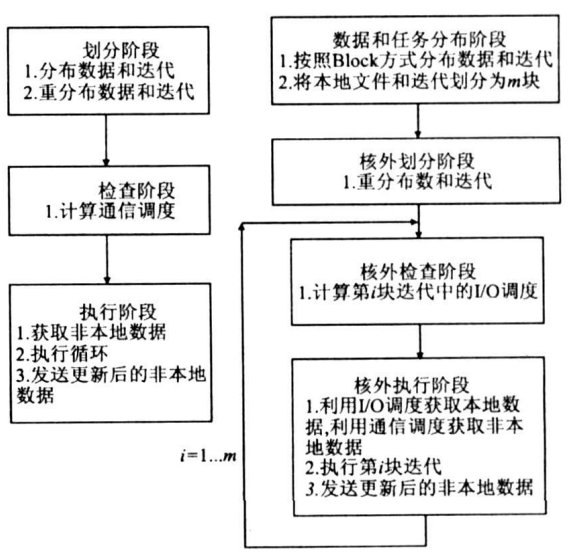
数据的非规则存取方式会导致数据的空间和时间局部性降低.因此,在提高时间局部性方面,可以使用数据复用技术[18,19].在提高空间局部性方面,一类是Alan Cox等人提出的使用空间曲线和行列排序这两种数据重排方法[20];另一类是通过使用最优化I/O算法选择硬盘文件的分布式方法[21].

2.2.4 负载均衡

非规则计算会导致负载失衡.因此,为了保持负载均衡,L. V. Kale等人在核外计算的基础上,针对非规则计算提出了使用对象合并策略代替任务合并策略的观点,分析自动选择负载均衡策略,实现精确的、细粒度的动态负载均衡[22,23].文[24-26]针对自适应非规则计算中的动态负载均衡问题,分析并提出了的改进算法.负载均衡需要任务的动态调度,文[27]利用动态任务图(DTG)算法实现了运行时存取任务.文[28]针对参数化的任务图提出了符号化线性聚类算法.这些都是对负载均衡进行优化的很好的策略,但均未考虑近年来国际流行的SMP集群系统结构和网络存储环境相结合的需要.任务池技术是在集群中实现非规则任务调度负载均衡的主要方法.文[5,29]对并行非规则算法任务池的几种实现做了评估并提出了任务池队列等改进算法.Banicescu Ioana等人针对集群计算实现了动态负载均衡库[30],利用循环调度技术提供的数据整合、资源管理和对象整合机制对非规则计算进行性能优化.针对异构集群中任务调度的特点,文[31,32]论述了启发式动态调度策略和负载均衡的深度优先算法.

2.2.5 通 信

数据访问模式的改变会引起通信调度的改变.为了避免在一次通信中非本地数据引用的重复,并保持一定的灵活性,需要有效的通信调度.文[11]给出了建立通信调度的过程,其优点是,在下标分析和转换阶段,散列表用于完成数组的全局下标和局部下标之间的转换,并删除重复的非本地数据引用.



(a) 核内方法 (b) 核外方法

图3 EIE模型的执行过程

Fig. 3 Execution process of EIE model

EIE模型将非规则计算和核外计算作为整体考虑,根据处理数据和内存的大小将模型分为核外和核内两种情况,如图 3所示.

核内情况下 EIE模型和 IE模型的执行过程一致.核外情况下 EIE模型分为四个阶段(如图 3(b)):

(1) 数据和任务分布阶段: 首先,迭代和数据以及间接数组在各处理器之间执行 Block 分布,数据存放在各处理器的本地文件中;然后,本地文件依照有效内存大小划分为 m块;最后,迭代部分根据文件的划分情况按照 Block方式划分为 m块迭代;

(2) 核外划分阶段: 为了提高性能,使用面向硬盘的划分方式对数据和迭代进行重分布;

(3) 核外检查阶段: 通过预处理决定第 i块迭代中的 I/O调度以及通信调度;

(4) 核外执行阶段: 首先,根据 I/O调度和通信调度分别获取本地和非本地数据;然后,执行迭代操作;最后,根据通信调度来发送 /接收计算后的非本地数据,并根据 I/O调度将非本地数据写回硬盘. (3)、(4)阶段根据迭代块数,循环执行 m次.

文 [2,33-35]论述了 EIE模型的具体实现,并将其应用在

表 1 非规则计算模型实现

Table 1 Implementations of irregular computing models

项目名称	支持模型	数据划分	计算划分	局部性	通信机制	其它特点
CHAOS/PARTI[10,36]	IE	根据空间位置、连通性和计算负载特性划分	1.大多数拥有者计算 2.拥有者计算	数据和迭代重分布	1. PVM和 MPI 2.软件缓存 3.通信整合	1.没有考虑携带相关 2.通过重复执行 Inspector阶段并结合计算划分方法使负载均衡
PILAR[37]	IE	规则分布	拥有者计算		1. MPI 2.通信模式的多种表示 3.非阻塞全局交换原语	1.利用面向对象技术 2.非规则存取中的规则特性
ICRL[9]	IE	规则分布	拥有者计算	数据重分布	1. PVM 2.复用通信调度	多种全局 /局部地址转换
Halos[38]	IE	注 1	拥有者计算	数据重分布	1. MPI 2.复用通信调度 3.通信整合 4.通信与计算重叠	支持 VFC编译器
EARTH-C[39]	IE	规则分布	Light Inspector划分		通信与计算重叠	1.通信独立于问题划分 2.支持自适应非规则计算
CoLuMBO[40]	IE	注 1	拥有者计算		1. MPI 2.通信与计算重叠	1.区间扫描 2.通信预接收
TriDenT[41]	IE	注 1	树结构划分		1. MPI 2.通信与计算重叠	1.非规则活跃(active)处理器组 2.携带相关处理 3.树结构优化非规则计算
UMA[17]	IE	规则分布	DWA-LIP 方法	1.数据和迭代重排 2.分类		利用 DWA-LIP方法发掘并行性和数据局部性
Titanium[44-46]	IE	规则分布	拥有者计算		1.通信与计算重叠 2.复用通信调度 3.可选通信模型	对核外计算进行优化
COSMIC[15]	IE	递归坐标对分(RCB)算法	本地写方法	1.计算复制 2. RCB算法	1.通信整合 2. flush-update协议	1.支持 SUIF/CVM系统 2. Overdrive协议减少页保护的负载 3.自动更新释放一致性模型(AURC)
TreadMarks[20]	类 IE	RCB算法		RCB算法	1.通信整合 2.流水算法 3. round-robbin算法	1.间接数组的规则片分析和动态检测 2.预取技术 3.无效一致性协议
CHAOS+[42]	EIE	同 CHAOS	同 CHAOS	同 CHAOS	同 CHAOS	1.局部冗余消除 2.隐藏 I/O开销 3.转换循环 4.重排计算
Lip[11,33]	EIE	规则分布		数据重分布	1. MPI 2.全交换通信	实现 Java接口
Polaris[6]	PW/SE/IE	1.遍历调度 2.预调度方法 3.规则分布	1.冲突域方法 2.复制方法 3.规约表方法			数据存取分析

(注 1:规则分布、Gen_Block分布、INDIRECT分布)

网格和集群中.相对于 IE模型,EIE模型的优点是将核外和非规则作为整体考虑.但是 EIE模型并没有考虑实际问题中 SMP集群系统结构和网络存储环境的应用.

2.3 非规则计算模型的实现

对非规则计算进行优化的研究还有很多.在非规则任务方面,RAPID库[47]和 PRFX库[48]均利用 DAG图进行优化;不同的是,前者针对分布内存系统结构提出消息合并机制;后者针对 SMP集群提出基于任务范例和显式同步的编程模型,并结合 POSIX线程和单边通信库对非规则任务进行优化.文[49]通过全局控制状态的检测实现对典型非规则应用的并行化.在非规则数据结构方面,文[50]系统给出了非规则数据结构如图、树、集合等的表达和处理方法,文[51]使用称为非规则存取区域描述子(Irregular Access Region Descriptor)的数据结构和抽象数据类型结构来表示间接数组和动态共享模式.此外,文[52]在 CELL处理器上通过任务划分实现了列表序列这一典型非规则内存存取模式的并行化.文[53]给出了 OpenMP对典型应用程序中出现的非规则计算形式进行表达和处理的技术,结果表明利用 OpenMP规范和适当的运行支持系统可以获得与 HPF以及 MPI方式相当的效率,文[54,73]系统研究了 Fortran语言+ OpenMP对典型 benchmark程序中非规则计算形式的表达方法,并给出了采用源到源编译方式和多线程技术的实现及优化技术.文[55]对高速暂存存储器(Scratch-Pad)以及 Cache中出现的非规则数组存取进行了优化.文[56]针对 BIPS3D非规则典型应用,通过图划分所提供的信息对并行 I/O进行了优化.这些方法对单纯的非规则计算很有效,但均未考虑核外计算带来的复杂性.

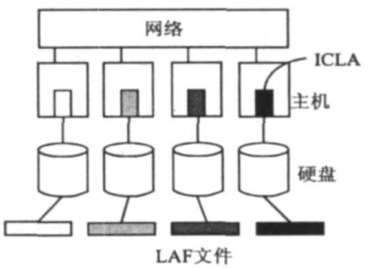
3 核外计算的并行模型

1996年 DUKE大学的 Zhongyong Li博士对核外计算进行了系统的分析[57],此后,针对核外计算进行优化的研究一直没有间断.基本上,这些研究可被归为虚拟内存优化和显式 I/O两类.显式 I/O按处理 I/O方式的不同,其模型可分为分布式 I/O模型和集中式 I/O模型.

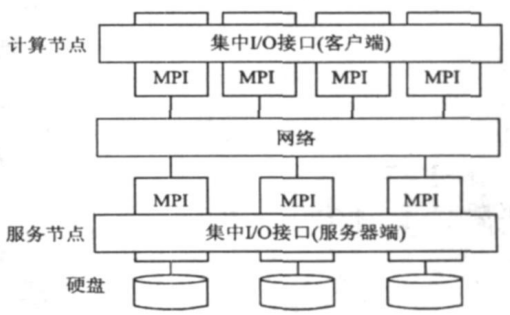
3.1 分布式 I/O模型

在分布式 I/O模型[3,59]中程序的主要执行模式是 SPMD模式.如图 4(a)所示,对全局核外数组进行划分,并保存在分布的 LAF(Local Array File)中.在各结点程序中开辟缓冲区,将 LAF中的数据分成若干数据块 ICLA(In-core Local Array)并依次读入缓冲区进行处理,然后在适当的时候将缓冲区数据写回 LAF.每个处理器都拥有单独的 LAF,可以直接对其进行访问.处理器获得的数据可分为两种:一种是本地核外数据,通过两阶段 I/O方法从核外本地数组中获得,另一种是非本地核外数据,需要通过消息传递从拥有者处理器获得.这种模型适用于每个处理器都有独立的 I/O系统的并行体系结构的情况,如多计算机系统和工作站集群系统.

在该模型中,优化的核心是根据内存空间大小来提高计算的局部性.首先,在数据块划分方面,文[61]给出了模型驱动的数据块自动划分方法.此外,在循环划分方面,Pawel等人提出了在分布式多层次存储系统结构中的划分方法[62].除了对分块和循环进行 I/O优化,还需要对通信进行优化.通信有两种优化方法:一种是核外通信方法,将通信和计算分为两个不同的执行阶段,要求编译器识别和优化聚合通信;另一种是核内通信方法,将在一个执行阶段中实现通信和计算.当计算需要数据块的非本地核外数据时则立刻发送或接收.第一种方法的缺点是核内数据块不能有依赖关系.第二种方法的缺点是优化非常困难,需要额外的编译时间.通信优化除了可以减少通信时间以外,也有利于减少全局的 I/O开销,因为编译器可以自由地重组数据块的访问次序,只有当数据块在内存中时才进行通信.最后,需要对底层 I/O进行优化,包含 I/O流水、I/O请求聚集、选择磁盘访问策略和 I/O模式等一系列问题.



(a) 分布式 I/O 模型



(b) 集中式 I/O 模型

图 4 显式 I/O模型

Fig. 4 Explicit I/O model

为了实现系统的高效 I/O,需要研究数据复用、数据筛选、集合 I/O以及数据预取等 I/O优化技术[3,59].

数据复用技术是指,已用数据在内存缓冲区中存有备份,避免了数据再次需要时硬盘的重复存取.这种方法的效率依赖于缓冲区分配的策略.在 Unix内核中有多种算法的实现.文[63]论述的复用方法使已装入内存的数据得到有效的复用,并利用计算和通信重叠提高核外计算性能.文[72]通过自动核外数据管理器允许常用数据驻留在内存中增加了数据复用,从而减少了磁盘 I/O次数.

数据筛选是指,将 I/O请求范围内的整块数据读入缓冲区中,再把需要的数据从缓冲区中萃取到 ICLA中.当读取无关数据带来的开销大于在非优化状态下读取的开销时,应该通过算法分析用户的请求并决定是否执行数据筛选.ROMIO[64]通过 MPI驱动的数据类型实现了数据筛选功能.

集合 I/O是通过合并不同进程的 I/O请求来提高 I/O性能.如果预知所有处理器的 I/O存取模式,就可以通过两阶段 I/O方法来实现数据的高效存取,这类优化策略的优点

是将所有文件存取整合为一个大而连续的文件存取,因此 I/O时间就会大大减少.虽然在重分布时会增加少量的通信开销,但却节省了大量的 I/O时间.

数据预取将 I/O时间和计算时间重叠以减少时间开销.在当前数据块被读取之后,立刻对下一数据块采用异步 I/O操作,实现时间重叠,因此在性能上会有明显的提高.

3.2 集中式 I/O模型

集中式 I/O模型[65]的基本思想是尽可能执行序列化的读写操作以提高 I/O性能.模型如图 4(b)所示.与分布式 I/O模型不同的是,在集中式 I/O模型中节点分为计算节点和服务节点两类.计算节点和服务节点之间通过网络进行连接.客户端的计算节点负责进行计算,服务器端的服务节点负责读写磁盘上的数据.集中式 I/O模型支持集合 I/O操作,在服务节点上直接调整 I/O请求.由于服务节点知道数组数据在磁盘和内存文件中的布局,因此服务节点通过序列化读写操作,提高了 I/O性能.

3.3 核外计算模型的实现

核外计算优化主要通过分布式 I/O模型、集中式 I/O模型和虚拟内存优化三种方法实现.虚拟内存优化是通过有效的管理页交换以提高核外计算的性能[43],而分布式 I/O模型与集中式 I/O模型是通过显式控制 I/O提高核外计算性能.两种 I/O模型的不同点是分布式 I/O模型的节点既执行计算又执行 I/O操作,而集中式 I/O模型的节点按照计算和 I/O操作功能的不同将其分开处理.下面对一些典型核外计算模型实现的特点进行介绍.

表 2 核外计算模型实现

Tabel 2 Implementations of out-of-core computing models

项目名称	应用平台	I/O优化策略	可移植性	典型应用
PASSION[3,59]	IBM SP2、Intel Delta 和 Intel Paragon	1.两阶段 I/O 2.数据筛选 3.数据预取 4.数据复用	设备无关	1.拉氏方程 2. LU分解
ChemIO[68]	分布/共享内存系统结构计算机、NOW	1.集合 I/O 2.独立 I/O 3.异步 I/O	文件系统无关	计算化学
TPIE[69]	Dell PowerEdge 2400、Sun Sparc20 和 DEC Alpha500	1.基于流机制 2.任意存取机制	操作系统无关	1. GIS 2.空间数据 3. VLSI设计
KelpIO[70]	分布内存系统结构计算机	1.合并 I/O操作 2.异步 I/O	底层 I/O无关	FFT
Jovian[66]	IBM SP1、Intel Pargon	1.集合 I/O 2.视图	文件系统无关	1. GIS 2.数据挖掘 3.非规则存取应用
ViPIOS[60,71]	集群结构和 MPP	1.数据预取 2.最佳数据存取策略 3.拓扑选择服务节点 4.视图 5.共享文件指针	标准 MPI接口、HPF编译系统无关	VFC编译系统
Panda[65]	Intel CFS、Intel Paragon、IBM SP2、Cray T3E、Sun工作站和 HP工作站	1.异步 I/O 2. Server-directed I/O 3.跳步存取	标准 MPI接口	CFD
Solar[67]	分布/共享内存系统结构计算机、NOW	1.异步 I/O 2.两阶段 I/O 3.跳步存取 4. Disk-directed I/O	设备无关	线性代数核外计算
MPI-IO[64]	分布内存系统结构计算机	1.显式偏移量 2.独立/共享文件指针 3.视图	语言无关	基于消息传递的应用
ViC* [58]	DEC2100	1.异步 I/O 2.数据预取		FFT
CHARM++[22,23]	分布/共享内存系统结构计算机、NOW	1.预测机制 2.预取机制	设备无关	动态非规则应用

3.3.1 分布式 I/O模型

PASSION:基于抽象磁盘系统的核外并行计算编译技术.针对分布式磁盘结构,提出了本地化优化方法,建立了一个和分布内存系统结构相对应的分布外存系统结构模型,从而为程序员在语言级编程方面提供了一致的数据分布模型,做到核内、核外的一致性处理,该工作对基于分布内存系统结构的数据并行语言是十分有效的[3,59].

ChemIO:利用单独存取文件、共享文件和磁盘内数组三类接口来优化化学计算中的核外情况[68].

TPIE:提出了 CRB-tree、BKD-tree等 I/O高效的动态数据结构,以及在处理大数据集问题中最小化 I/O通信的核外算法[69].

KelpIO:利用 Inspector/Executor通信范例,优化支持非规则 Block结构的科学计算.优化方法包括动态调整缓存大小、在程序中加入计算和 I/O功能以隐藏文件系统的延迟、合并 I/O操作和优化核外交换(swap)文件[70].

3.3.2 集中式 I/O模型

ViPIOS:基本思想是基于客户端/服务器端结构,将磁盘访问操作从应用程序中分离出来,由独立的 I/O子系统完成.系统的体系结构建立在服务进程上,由服务进程完成客户进程请求.服务进程的工作原理类似于并行数据库系统中的服务进程,并利用数据局部性提高数据访问性能.优化方面包括核外数据的划分及重分布、核外数据的通信、数据缓存和预

取等.此外,其它特点包括:支持集群系统、实两阶段数据管理、消息传递和共享文件指针[60,71].

Jovian:对集合 I/O进行优化,将多个 I/O请求封装为一个 I/O请求,减少了 I/O请求的数目.在多磁盘或磁盘阵列的多处理器结构中提高集中式 I/O性能.在存取数据方面,为了保证其灵活性,运行库有全局观点(global view)和分布式观点(distributed view)[66].前者将核内数据和核外数据统一存取,而后者则是在 I/O请求之前将本地核内数据索引转换为全局核外数据索引.

Panda:在分布内存系统结构和 SPMD模式以及网络环境下,基于客户端/服务器端结构提出了 Server-directed集中式 I/O模型.利用 I/O子系统的高层接口、子数组块存储来提高系统性能.在运行阶段实现计算节点和服务节点之间的通信,以及服务节点对多维数组的集中 I/O操作[65].

Solar:使用两阶段 I/O和预取方法实现核外密集矩阵的计算[67].

MPI-IO:将 MPI中消息传递的概念用于并行 I/O.利用显式偏移量、独立文件指针和共享文件指针,实现支持共享数据访问的功能.通过单个 I/O函数和派生数据类型访问非连续数据[64].类似实现还有 ROMIO和 MPI-2,与 MPI-IO的区别分别是 ROMIO是客户端/服务器端结构,MPI-2具有间隔存取功能.

3.3.3 虚拟内存优化方法

ViC*:将传统的页调用虚拟内存管理转换为由程序员定义的虚拟内存管理,有效的减少了核外数据的磁盘存取开销[58].定义了核外结构、编译器重构并行操作和利用 PDM模型管理 I/O数据传输的运行库.此外,ViC* 还支持数据并行.

CHARM++:提出了基于面向对象的优化模型以及提高虚拟内存中的页交换性能的两种优化技术.一种是预取技术,通过一个附加线程预取页,重叠缺页时间和应用的计算时间来有效的处理缺页问题.另一种是多线程对象内存管理技术,使用多辅助线程来管理对象内存,利用模型中的预测队列来预测内存存取模式,并将队列作为内存管理的工作区,以提高页置换算法效率[22].

针对上述模型实现,表 2对其应用平台、I/O优化策略、可移植性和典型应用进行了分类比较,以便进一步地研究.相对而言,显式异步 I/O性能高,但其复杂性也高.在实际应用中需要选择合适模型和存取策略,以适应不同的需求.总体来说,还没有完全支持 SMP集群系统结构和网络存储环境的模型实现.

4 结论和下一步的研究方向

非规则、核外计算的并行优化一直是并行计算与并行编译领域研究的挑战性课题,从上面的分析可以得出如下几个基本结论:

(1) 非规则、核外计算是并行优化研究的热点问题,是影响并行应用系统效率的关键问题;

(2) 目前,在并行模型、通信实现、优化等方面对这两类问题的研究是相当深入的,但是将两者结合起来研究还尚不多见,这不能不说是一种遗憾,因为它们不仅共存于大规模并行应用内,而且在解决的技术上也是密切相关的;

(3) 充分利用系统结构与数据本身的特点来寻求优化的途径是所有解决非规则、核外计算的共同思想;

(4) 非规则、核外计算的效率的提高还有很大的余地,需要寻找新的突破方向;

(5) 目前的处理技术多着眼于分布内存系统结构的集群系统,对 SMP集群系统结构和网络存储环境的关注较少.

事实上核外计算和外存系统结构密切相关,SMP集群系统结构丰富的主存层次也为非规则计算模型优化提供了新的空间.

在 SMP集群系统结构和网络存储环境逐渐流行的今天,为非规则、核外计算处理技术的突破提供了新的机会.如何充分利用 SMP集群计算系统结构和网络存储环境的新特点,以及在实际应用中如何把非规则、核外计算统一起来以充分利用应用数据的特点仍然是研究的重点.主要研究的方向将可能集中在如下几个方面.

4.1 多层次协同研究非规则、核外计算

所谓多层次协同研究非规则、核外计算,是指在问题的语言描述、编译处理、运行支持库等多层次上协同处理.在问题描述上,通过适当地设计并行指导语句,对数据的非规则特点和核外信息进行充分地描述,为编译优化提供足够的信息;而在编译处理和运行支持设计上,考虑如何利用描述信息和体系结构的特点进行足够的优化.例如对于 SMP集群系统结构和网络存储环境,可以在语言级考虑扩充典型的 OpenMP规范来描述非规则特点和核外信息.而在编译处理和运行支持设计上充分考虑:

(1) 针对 SMP集群系统结构和网络存储环境的结点内共享内存、结点间分布内存、结点间共享外存的特点,如何给出充分利用三层次存储访问的核内、核外一致性的非规则处理模型;

(2) 如何针对多层次的存储模式设计数据划分和计算划分模式,实现既支持核内又支持核外的运行支持系统;

(3) 如何利用网络存储提供的虚拟存储池功能最大限度地实现通信优化(尤其是并行 I/O优化)等.

4.2 综合研究非规则、核外计算

非规则、核外计算虽然是性质完全不同的两个问题,但它们不仅普遍共存于大规模并行应用中,而且在技术上也是相关的.所以在应用系统的框架中,将两个问题结合起来综合研究,是提高并行系统效率的又一途径.也就是从应用问题出发,在并行粒度确定、并行范例选择时就充分考虑非规则、核外计算的处理,而不是仅从理论上设计一些非规则、核外表达.实际上,在 SMP集群系统结构和网络存储环境下,既可以支持分布内存系统结构的数据并行范例(如 HPF),又可以支持共享内存系统结构的任务并行范例(如 OpenMP);既可以支持粗粒度并行(如过程间、构件间),又可以支持中粒度、细粒度的并行(如循环、过程内).通过选择适当的并行粒度和范

例来减少非规则、核外计算的处理代价,这是有前途的研究方向.

4.3 非规则通信模型和负载的动态均衡技术

非规则、核外计算必然导致非规则通信和负载的不均衡.群通信模式和拥有者计算原则显然不能适应这一需求,不仅如此,像 Remap、Scatter、Gather等方式以及 OpenMP编译系统的 Critical、Barrier等同步控制也很难适应非规则、核外处理的需求.

在非规则、核外并行的环境下,需要处理点点通信、单边通信、不同存储层次间的通信以及不同线程之间的通信等情况,需要在运行支持库的设计中,从并行粒度划分、存储策略确定等角度进行通信优化.动态、自适应的负载均衡技术也是影响非规则并行计算效率的关键,需要在通信优化的基础上,根据不同的非规则模式和运行信息研究优化的调度策略,包括负载粒度的细化、迁移、动态创建和合并机制,多层次存储体系下的数据存取代价估算,多范例、多粒度并行下的 I/O 代价估算模型,基于存储访问追踪技术的调度优化技术等.这些技术的突破是提高非规则、核外系统效率的关键,是下一步必须解决的问题.

References

[1] Daisuke Takahashi, Mitsuhisa Sato, Taisuke Boku. Performance evaluation of the hitachi SR8000 using OpenMP benchmarks[A]. In: Proc of the 4th International Symposium on High Performance. Comp[C]. Lecture Notes In Computer Science, 2002, 2327: 390-400.

[2] Brezany P, Choudhary A, Dang M. Language and compiler support for out-of-core irregular applications on distributed-memory multiprocessors[A]. In: Proc of the LCR98[C]. Berlin: Springer-Verlag Press, 1998, 71-78.

[3] Kandemir M, Choudhary A, Ramanujam J, et al. Compilation techniques for out-of-core parallel computations[J]. Journal of Parallel Computing, 1998, 24(3-4): 597-628.

[4] Chiu S, Liao W-k, Choudhary C. Processor-embedded distributed MEMS-based storage systems for high-performance I/O[A]. In: Proc of the 18th International Parallel and Distributed Processing Symposium[C]. IEEE Computer Society Press, 2004. 91-100.

[5] Hippold J, Runger G. Task pool teams for implementing irregular algorithms on clusters of SMPs[A]. In: Proc of the International Parallel and Distributed Processing Symposium[C]. France: IEEE Computer Society Press, 2003, 54.

[6] Yuan Lin. Compiler analysis of sparse and irregular computations[D]. Illinois: University of Illinois at Urbana-Champaign, May 2000.

[7] Rauchwerger L, Padua D. The LRPD test: speculative run-time parallelization of loops with privatization and reduction parallelization[J]. IEEE Trans. Parallel Distrib. Syst. 1999, 10(2): 160-180.

[8] Hongzhang Shan, Jaswinder Pal Singh, Leonid Oliker, et al. Message passing and shared address space parallelism on an SMP cluster[J]. Journal of Parallel Computing, February, 2003, 29(2): 167-186.

[9] Wang Li-hong, Fang Bin-xing, Hu Ming-zeng. Design and realization of runtime library support for irregular computation[J]. Journal of Harbin Institute of Technology (New Series), 2001, 8(2): 159-164.

[10] Saltz J, Ponnusamy R, Sharma S D, et al. A manual for the CHAOS runtime library[R]. Department of Computer Science, University of Maryland, Tech Rep: CS-TR3437, March 1995.

[11] Maciej Malawski, Katarzyna Zajac. Advanced library for parallelization of irregular and out-of-core problems[D]. Poland: University of Mining and Metallurgy Poland, 2001.

[12] Agrawal G. Automatic data partitioning for irregular and adaptive applications[A]. In: Proc of International Conference on Parallel Processing[C]. IEEE Computer Society Press, August 1998, 587-594.

[13] Maria J Martin, David E Singh, Tourino J, et al. Exploiting locality in the run-time parallelization of irregular loops[A]. In: Proc of the ICPP'02[C]. Canada: IEEE Computer Society Press, 2002, 27-34.

[14] Guo Min-yi. Automatic parallelization and optimization for irregular scientific applications[C]. In: Proc of 18th International Parallel and Distributed Processing Symposium (IPDPS'04), 2004, 228a.

[15] Han H, Tseng C-W. Efficient compiler and run-time support for parallel irregular reductions[J]. Journal of Parallel computing, 2000, 26(13-14): 1861-1887.

[16] Asenjo R, Corbera F, Gutiérrez E, et al. Optimization techniques for irregular and pointer-based programs[C]. In: Proc of the 12th Euromicro Conference on Parallel, 2004, 2-13.

[17] Eladio Gutiérrez, Oscar G Plata, Emilio L Zapata. Improving parallel irregular reductions using partial array expansion[A]. In: Proc of the 2001 ACM/IEEE conference on Supercomputing[C]. Colorado: ACM Press, 2001, 56.

[18] Juan C Pichel, Dora B Heras, Jose C Cabaleiro, et al. A new technique to reduce false sharing in parallel irregular codes based on distance functions[C]. In: Proc. 8th International Symposium on Parallel Architectures, Algorithms and Networks, 2005, 306-311.

[19] Van Achteren T, Lauwereins R, Catthoor F. Systematic data reuse exploration methodology for irregular access patterns[C]. In: IEEE/ACM 13th International Symposium on System Synthesis (ISSS'00), Madrid, Spain, 2000, 115-121.

[20] Hu Y, Cox A, Zwaenepoel W. Improving fine-grained irregular shared-memory benchmarks by data reordering[C]. In: Proc. SC'00, Dallas, TX: ACM Press, November 2000, 33.

[21] Krishnan S, Krishnamoorthy S, Baumgartner G, et al. Data locality optimization for synthesis of efficient[A]. Out-of-Core Algorithms, in 10th Annual International Conference on High-Performance Computing (HiPC-2003)[C]. Springer-Verlag Press, 2003, 406-417.

[22] Mani S Postnuru. Automatic out-of-core execution support for charm++[D]. Illinois: University of Illinois at Urbana-Champaign, 2003.

[23] Krishnan S, Kale L V. Automating runtime optimizations for load balancing in irregular problems[C]. In: Proc of the Conference on Parallel and Distributed Processing Technology and Applications, San Jose, August 1996, 1465-1476.

[24] Barker K J, Chrisochoides N P. An evaluation of a framework for the dynamic load balancing of highly adaptive and irregular parallel applications[A]. In: Proc. the ACM/IEEE Conference [C]. IEEE Computer Society Press, 2003, 45.

[25] Fedorov A, Chrisochoides N. Communication support for dynamic load balancing of irregular adaptive applications[C]. In: Proc. the 2004 International Conference on Parallel Processing Workshops, 2004, 555-562.

[26] Banicescu I, Velusamy V. Load balancing highly irregular computations with the adaptive factoring[C]. In: Proc. the International Parallel and Distributed Processing Symposium, IEEE Computer Society, 2002, 195.

[27] Jnhnson T, Davis T A, Hadfield S M. A concurrent dynamic task graph[J]. Journal of Parallel Computing, 1996, 22(2): 327-333.

[28] Cosnard M, Jeannot E, Yang T. SLC: symbolic scheduling for executing parameterized task graphs on multimachines[C]. In: Proc. the 28th International Conference on Parallel Processing, Japan, 1999, 413-421.

[29] Brezany P, Bubak M, Malawski M, et al. Irregular and out-of-core parallel computing on clusters[C]. In: Proc. the Conference PPAM 2001. Naleczow, Poland: Springer-Verlag Press, September 9-12, 2001, 299-306.

[30] Ioana B, Carino R L, Jakerick P P, et al. Design and implementation of a novel dynamic load balancing library for cluster computing[J]. Journal of Parallel Computing, 2005, 31: 736-756.

[31] Qin X, Jiang H. A dynamic and reliability-driven scheduling algorithm for parallel real-time jobs executing on heterogeneous clusters[J]. Journal of Parallel and Distributed Computing, 2005, 65(8): 885-900.

[32] Mohammed A M Ibrahim, Xinda Lu. Parallel execution of an irregular algorithm depth first search on heterogeneous clusters of workstations[A], 2001 Int'l Conf. on Info-tech and Info-net Proceedings[C]. IEEE Computer Society Press, 2001, 10: 328-551.

[33] Brezany P, Bubak M, Malewski M, et al. Large-scale scientific irregular computing on clusters and grids[A]. In: Proc. the 2nd International Conference on Computational Science[C]. Amsterdam: Springer-Verlag Press, 2002, 484-493.

[34] Gupta M, Nim R. Techniques for speculative run-time parallelization of loops[A]. In: Proc of the 1998 ACM/IEEE conference on Supercomputing[C]. IEEE Computer Society Press, November 1998, 1-12.

[35] Brezany P, Bubak M, Malawski M, et al. Irregular and out-of-core parallel computing on clusters[A]. In: Proc. the Conference PPAM 2001[C]. Naleczow, Poland: Springer-Verlag Press, 2001, 299-306.

[36] Ponnusamy R, Saltz J, Choudhary A, et al. Runtime support and compilation methods for user-specified data distributions [J]. IEEE Trans. on Parallel and Distributed Systems, 1995, 6(8): 815-831.

[37] Lain A, Banerjee P. Compiler support for hybrid irregular accesses on multicomputers[A]. In: Proc. International Conference on Supercomputing 1996[C]. Philadelphia, PA, USA: ACM Press, 1996, 1-9.

[38] Benkner S. Optimizing irregular HPF applications using halos [A]. In: Workshop Proceedings of the International Symposium on Parallel Processing [C]. Puerto Rico: Springer-Verlag Press, April 1999, 1015-1024.

[39] Gary Zoppetti, Gagan Agrawal, Rishi Kumar. Compiler and runtime support for parallelizing irregular reductions on a multithreaded architecture[A]. In: Proc of the IPDPS'02[C]. IEEE Computer Society Press, 2002, 18-27.

[40] Bregier F, Counilh M-C, Roman J. Scheduling loops with partial loop-carried dependcies[J]. Journal of Parallel Computing, 2000, 26: 1789-1806.

[41] Frédéric Bégier, Marie Christine Counilh, Jean Roman. Propositions for handling irregular problems with HPF2[Z]. HPF Users Group Meeting, Porto, Portugal, 1998.

[42] Brezany P, Choudhary A, Dang M. Parallelization of irregular codes including out-of-core data and index arrays[A]. In: Proc of the conference Parallel Computing 1997[C]. North Holland: Elsevier Press, 1998, 132-140.

[43] Angela Demke Brown. Explicit compiler-based memory management for out-of-core applications[D]. School of Computer Science Carnegie Mellon University, 2005.

[44] Su J, Yelick K. Array prefetching for irregular array accesses in titanium[A]. In: Proc of the IPDPS'04[C]. IEEE Computer Society Press, 2004, 158.

[45] Su J, Yelick K. Automatic support for irregular computations in a high-level language[A]. In: Proc of the IPDPS'05[C]. IEEE Computer Society Press, 2005.

[46] Ferreira R, Agrawal G, Saltz J H. Data parallel language and compiler support for data intensive applications[J]. Journal of Parallel Computing, 2002, 28(5): 725-748.

[47] Fu Cong, Tao Yang. Run-time techniques for exploiting irregular task parallelism on distributed memory architectures[J]. Journal of Parallel and Distributed Computing, 1997, 42(2): 143-156.

[48] Cirou B, Marie Christine Counilh, Jean Roman. Programming irregular scientific algorithms with static properties on clusters of SMP nodes[A]. ICPP Workshops 2005[C]. IEEE Computer Society Press, 2005, 145-152.

[49] Borkowski J, et al. Global predicate monitoring applied for control of parallel irregular computations[C]. Proceedings of the International Symposium on Parallel Computing in Electrical Engineering, 2006, 233-238.

[50] Prechelt L, Stefan U H. Efficient parallel execution of irregular recursive programs[J]. IEEE Trans on Parallel and Distributed Systems, 2002, 13(2): 167-178.

[51] Singh D E, Rivera F F, Martn M J. Run-time characterization of irregular accesses applied to parallelization of irregular reduc-

tions[A]. In 30 Int'l Conference on Parallel Processing Workshops, Valencia[C]. Spain: IEEE Computer Society Press, 2001, 17.

[52] David A Bader, et al. On the design and analysis of irregular algorithms on the cell processor: a case study of list ranking[C]. 21th IEEE International Parallel and Distributed Processing Symposium (IPDPS), Long Beach, CA, March 26-30, 2007.

[53] Dimitrios S N, Constantine D P, Eduard A. Scaling irregular parallel codes with minimal programming effort[A]. In: Proc of the SC'01[C]. ACM Press, 2001, 16.

[54] Prins J S, Chatterjee M Simons. Irregular computations in fortran expression and implementation strategies[J]. Scientific Programming, 1999, 7(3-4): 313-326.

[55] Javed Absar M, Francky Catthoor. Compiler-based approach for exploiting scratch-pad in presence of irregular array access[A]. DATE 2005[C]. IEEE Computer Society Press, 2005, 1162-1167.

[56] Rosa Filgueira, et al. Optimization and evaluation of parallel I/O in BIPS3D parallel irregular application[A]. Parallel and Distributed Processing Symposium, IPDPS 2007[C]. IEEE International, 2007: 1-8.

[57] Gupta S, Li Z, Reif J H. Synthesizing efficient out-of-core programs for block recursive algorithms using block-cyclic data distributions[J]. IEEE Transactions on Parallel and Distributed Systems, 1999, 10(3): 297-315.

[58] Baptist L M, Thomas H Cormen. Multidimensional, multiprocessor, out-of-core FFTs with distributed memory and parallel disks. ACM Symposium on Parallel Algorithms and Architectures[A]. In: Proc. the 11th annual ACM symposium on Parallel algorithms and architectures[C]. Saint Malo, France: ACM Press, 1999, 242-250.

[59] Kandemir M, Choudhary A, Ramanujam J, et al. A unified framework for optimizing locality, parallelism, and communication in out-of-core computations[J]. IEEE Trans on Parallel and Distributed Systems, 2000, 11(9): 648-662.

[60] Brezany P, Mueck T A, Schikuta E. A software architecture for massively parallel input-output[A]. In: Proc. PARA-96[C]. Springer-Verlag Press, 1996, 85-96.

[61] Krishnan S, Krishnamoorthy S, Baumgartner G, et al. Efficient synthesis of out-of-core algorithms using a nonlinear optimization solver[A]. In: Proc of the IPDPS'04[C]. IEEE Computer Society Press, 2004.

[62] Drozdowski M, Wolniewicz P. Out-of-core divisible load processing[J]. IEEE Trans. Parallel Distrib. Syst. 2003, 14(10): 1048-1056.

[63] Caron E, Desprez F, Suter D. Out-of-core and pipeline techniques for wavefront algorithms[A]. In: Proc of the IPDPS'05[C]. IEEE Computer Society Press, 2005.

[64] Thakur R, Gropp W, Lusk E. A case for using MPI's derived data types to improve I/O performance[A]. In: Proc. SC'98. San Jose, CA[C]. IEEE Computer Society Press, 1998, 1-10.

[65] Yong E Cho, Marianne Winslett, Szu-Wen Kuo, et al. Parallel I/O for scientific applications on heterogeneous clusters: a resource-utilization approach[A]. International Conference on Supercomputing[C]. Greece: ACM Press, 1999, 253-259.

[66] Bennett R, Bryant K, Sussman A, et al. Jovian: a framework for optimizing parallel I/O[C]. In: Proc of the 1994 Scalable Parallel Libraries Conference, Oct 1994.

[67] Toledo S, Fred G Gustavson. The design and implementation of SOLAR, a portable library for scalable out-of-core linear algebra computations[C]. In: Proc of the 4th Annual Workshop on I/O in Parallel and Distributed Systems, Philadelphia, May 1996, 28-40.

[68] Nieplocha J, Foster I, Kendall R A. ChemIO: high performance parallel I/O for computational chemistry applications[J]. The International Journal of High Perfomance Computing Applications, 1998, 12(3): 345-363.

[69] Lars Arge, Octavian Procopiuc, Jeffrey Scott Vitter. Implementing I/O-efficient data structures using TPIE[A]. In: Proc of the 10th Annual European Symposium on Algorithms[C]. Springer-Verlag Press, 2002, 88-100.

[70] Broom B, Rob Fowler, Ken Kennedy. KelpIO: a telescope-ready domain-specific I/O library for irregular block-structured applications[A]. In: Proc of the 2001 IEEE International Symposium on Cluster Computing and the Grid[C]. IEEE Computer Society Press, 2001, 148-155.

[71] Greenough C, Fowler R F, Allan R J. Parallel IO for high performance computing[EB/OL]. http://www.cse.clrc.ac.uk/Activity/HPCI, 2003-12-03.

[72] Sriram Krishnamoorthy, et al. An extensible global address space framework with decoupled task and data abstractions[C]. IPDPS 2006.

[73] Ayon Basumallik, et al. Programming distributed memory systems using OpenMP[C]. Parallel and Distributed Processing Symposium, 2007.