一种求解输运方程的并行调度算法

周涤宇[1)] 刘 杰[2)]

[1)](西安卫星测控中心 西安 710043)

[2)](国防科学技术大学计算机学院 长沙 410073)

摘 要 高效并行扫描问题是调度问题的子集,调度问题是 NP 完全问题.针对输运问题的特点,如何按特定的计算次序调度本地网格单元,以保证最佳的计算与通信性能是一个难度很大的问题.文中设计了一种基于局部深度优先的优先级(PDFDS)算法,该算法具有局部性、通信量小、优先级队列好等特点.将 PDFDS 算法应用到求解二维粒子输运方程的程序中,与现有的调度算法相比,新算法具有更好的并行计算效果,对于大规模计算问题,可以扩展到 1024 个处理器,相对于 64 个处理器的并行效率达到了 96%.

关键词 输运方程;并行调度;优先级算法;非结构网格;并行算法

中图法分类号 TP301 **DOI 号**: 10.3724/SP.J.1016.2010.00833

A Parallel Scheduling Algorithm for Solving Transport Equations

ZHOU Di-Yu[1)] LIU Jie[2)]

[1)](*Xian Satellite Control Center*, *Xi'an* 710043)

[2)](*School of Computer Science*, *National University of Defense Technology*, *Changsha* 410073)

Abstract Parallel sweep scheduling problem belong to the precedence constrained scheduling problem, and is NP-complete. How to schedule the local grids according to the characteristic of transport problem and keep the well performance of computation and communication is a hard work. This paper presents a priority scheduling algorithm based on DFDS algorithm. The new algorithm has well locality and less communication, and creates well priority queue. The new parallel priority scheduling algorithm is used to solve the two dimension particle transport equations. Performance results show that the new algorithm has better parallel performance than many scheduling algorithms widely used now. For the large transport problem, parallel efficiency is 96% on 1024 processors relative to 64.

Keywords transport equation; parallel scheduling; priority algorithm; unstructured grids; parallel algorithm

1 引 言

输运方程[1]是核科学与工程应用中的一类重要的偏微分方程,通过大量微观粒子的输运,确定粒子在几何空间、能量、速度相空间和时间上的分布.由于所模拟物理问题的复杂性,粒子输运方程是一个涉及大量物理量的复杂微分积分方程,只能采用数值模拟方法求解,其庞大的计算量是粒子输运计算不可回避的最大问题之一,多年来妨碍着非定常粒子输运大规模计算的实际应用水平的提高,迫切需要解决大规模可扩展并行计算问题.

收稿日期:2008-01-03;最终修改稿收到日期:2009-01-06.本课题得到国家自然科学基金(60673150,60970033)和国家"八六三"高技术研究发展计划项目基金(2008AA01Z137)资助.**周涤宇**,男,1978 年生,硕士,工程师,研究方向为大规模科学与工程计算.E-mail: zhoudiyu@163.com. **刘 杰**,男,1969 年生,博士,副研究员,从事大型科学与工程计算中并行算法、并行应用软件和性能评价等方面的研究.

目前典型的求解非结构网格上输运方程的并行算法有:Plimpton、Hendrickson、Burns和Mclendon[2]设计的三维 Cartesian 坐标系下的基于网格区域分解的并行流水线 Sn 扫描算法;莫则尧[3]通过分析二维柱几何 Lagrange 坐标系非结构网格上间断有限元 Sn 方法的内在并行度,提出的相应的并行流水线 Sn 扫描算法.这两种算法具有一定的相似性,进行结点调度时没有考虑网格点间复杂的数据依赖关系.然而本地网格单元的执行顺序会直接影响并行算法的性能.所以为了获得高的并行计算性能,必须有好的调度算法.

使用对每个结点赋优先级的方式来确定结点的调度顺序是现在比较通用的一种方法.在可调度结点集合当中,优先级大的结点先调度,优先级小的结点后调度,所以调度问题转化成了优先级算法的设计问题.当前,比较典型的优先级策略有最晚完成时间策略[4]、最晚开始时间策略[5]、b-level 策略[6]、BFDS 策略[6]、DFDS 策略[6]、DFHDS 策略[6]、Forward-Backward 策略[7]、最短内部边界路径策略[8]、多优先级策略[9-10]、顺逆交替迭代调度算法[11]等.但是,以上的策略和算法都需要使用到全局结点的信息,当问题规模很大、使用处理器数量很多时,这些策略和算法的成本开销将会急剧增加,严重影响并行算法的并行性能.本文在 DFDS 策略的启示下设计了一种基于局部深度优先的并行优先级(PDFDS)算法,与现有的优先级算法进行比较,并行计算性能有较大幅度提高.

本文第 2 节介绍输运调度模型;第 3 节分析几种基于深度优先的优先级算法;第 4 节给出一种局部深度优先的并行优先级(PDFDS)算法;第 5 节对算法进行性能分析和预测;第 6 节对算法进行测试对比;第 7 节进行总结.

2 输运调度模型

求解离散输运方程的标准迭代方法是源迭代法,交替求解局部的散射源和全部流驱动碰撞的变换算子.离散纵标(Sn)方程由输运方程导出,流驱动碰撞算子通常由"扫描"方法变换而来.在每次扫描中,对离散纵标集中的每个方向,流驱动碰撞算子通过网格上的每个空间局部单元以特定的顺序进行求解.而顺序依赖于网格点的相互作用,对于某个特定的角方向,下游的网格点需要等待上游网格点计算结果.整个的计算过程,就像一个平面波在向前传播,"下游"依赖"上游"的计算.分配到不同处理器的网格点,如果存在上下游的依赖关系,就需要通信,通信量大,同时下游处理机需要等待,这种特定的顺序使得数据具有强依赖性.如图 1 所示,网格 D 的求解依赖于它的上游网格 $D1$ 和 $D2$,保证在每个网格计算时,其进流边是计算区域的外边界的一部分,或者是已计算过的邻网格边界.

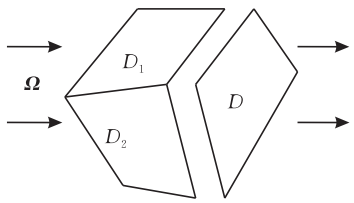


图 1 网格依赖关系图

输运问题中网格计算顺序的数据依赖关系可以抽象成一个带权有向图 $G=(V,E,\boldsymbol{p},W)$.这个有向图一般是无循环图,即 DAG 图.图中的结点集 $V=\{1,2,\cdots,n\}$代表所有网格单元的计算任务,图中的有向边集合 $E=\{(i_1,j_1),\cdots,(i_m,j_m)\}$代表网格间所有的数据依赖关系.非负向量 $\boldsymbol{p}$ 称为结点权重向量,元素 p_i代表结点 i 的时间开销;非负矩阵 W 称为有向边权重矩阵,元素 $w_{k,j}$表示有向边(k,j)的时间开销.如果有向边$(k,j)\in E$,则称该有向边与结点 k,j 相关联,结点 k 是该有向边的前端点,结点 j 是该有向边的后端点;并称结点 k 是结点 j 的直接前驱,结点 j 是结点 k 的直接后继.没有直接前驱的结点称为源点;没有直接后继的结点称为汇点.定义有向路经为一组结点序列$(u_1,u_2,\cdots,u_s)$,满足$(u_i,u_{i+1})\in E$,任意 $1\leq i<s$.如果图 G 中存在从结点 k 到结点 j 的有向路经,则称结点 k 是结点 j 的前驱,结点 j 是结点 k 的后继.路径长度指路径上包含的所有结点和有向边的权重总和.

在并行调度模型中,将有向图 G 划分成 P 个互不重叠的子图,分别分配给 P 个处理器.如果结点 i 属于处理器 p,则称处理器 p 是结点 i 的属主,结点 i 是处理器 p 的本地结点.如果结点 i 的前驱(后继)结点 j 和结点 i 同属一个处理器,则称结点 j 是结点 i 的本地前驱(后继)结点.如果有向边(k,j)两端结点 k 和 j 分属不同的处理器,则(k,j)称为截弧;如果有向边(k,j)两端结点 k 和 j 属于同一个处理器,则(k,j)称为内部弧.一般来说,内部弧的权重总是设为 0,而截弧的权重总是大于等于 0.

在使用并行流水线 Sn 扫描算法[3]求解输运方程的过程中，某一时刻任一个处理器可能没有结点可供调度(空闲)，也可能有一个或多个结点可供调度. 这些可供调度的结点是指那些其直接前驱结点已经全部计算完的结点，或者是进流边界上的结点. 调度算法的作用主要是在处理器有多个结点可供调度时选择“最”合适的结点进行调度，从而最大限度地减少所有处理器总的空闲时间，提高扫描算法的并行效率.

3　优先级调度算法分析

优先级调度算法是给每个结点按某种规则设置一个优先级，在结点调度时依照优先级的大小来确定调度的先后顺序. 现有的并行流水线 Sn 扫描算法[3]主要使用的是追尾法(补尾法)，即给每个结点设相同的优先级，这样结点将按照满足调度要求的先后顺序进行调度. 追尾法的算法成本开销很小，但是它没有考虑网格点间的数据依赖关系，所以很难得到较高的并行效率. 下面是几种基于深度优先的优先级算法.

b-level 优先级算法. b-level 优先级设为源点最大(MAX)，源点的直接后继次之，依次类推，将整个 DAG 图划分为 n 个等级. 图 2 为一个简单的 b-level 等级示意图，取 MAX 等于 10，结点旁边的数字代表该结点的 b-level 等级.

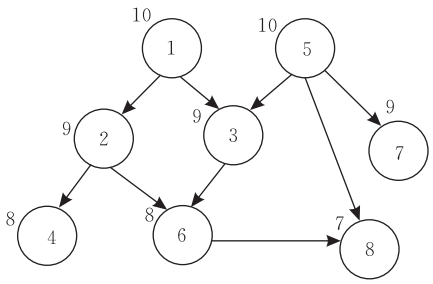


图 2　b-level 等级示意图

BFDS 优先级算法. 对任意一个结点，如果它有不属于自己同一个处理器的后继结点，设这些结点集为 V，则令该结点的优先级等于 V 中结点最大的 b-level 等级；如果 V 为空集则将该结点的优先级设为 0.

DFDS 优先级算法. 对任意一个结点，如果它有不属于自己同一个处理器的直接后继结点，设这些结点集为 U，则令该结点的优先级等于 U 中具有的最大 b-level 等级加上一个较大的常数，这个结点的前驱结点的优先级设为比它小一点的值；如果结点没有不属于自己同一个处理器的后继结点，则将该结点的优先级设为 0.

DFHDS 优先级算法. 对任意一个结点，如果它有不属于自己同一个处理器的直接后继结点，设这些结点集为 U，则令该结点的优先级等于 U 中具有的最大 b-level 等级乘以一个较大的常数，这个结点的前驱结点的优先级设为比它小一点的值；如果结点没有不属于自己同一个处理器的后继结点，则将该结点的优先级设为 0.

BFDS、DFDS、DFHDS 算法都是在 b-level 算法的基础上发展的，所以这 4 种算法都必须求得每个结点的 b-level 等级. 文献[3]中给出的追尾法、弧度法和坐标法并行性能相近，为叙述方便，我们仅对追尾法展开说明. 使用 b-level、BFDS、DFDS 和 DFHDS 4 种算法以及追尾法分别求解二维中子输运方程，使用 2～64 个处理器对相同规模的问题进行计算，图 3 给出的是分别使用 b-level、BFDS、DFDS、DFHDS 算法所用的并行计算时间除以使用追尾法所用的并行计算时间.

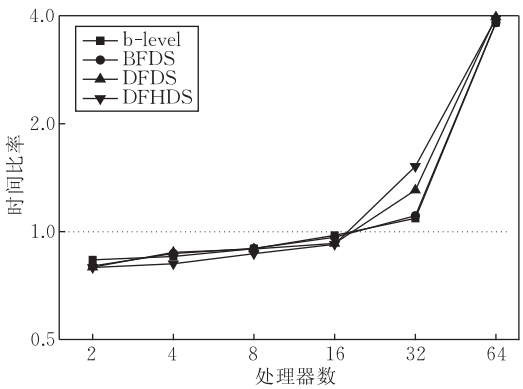


图 3　使用不同处理器数量，各算法与追尾法的并行性能比较

在图 3 中，实线在虚线上方，说明对应算法的并行性能比追尾法差；实线在虚线下方，说明对应算法的并行性能比追尾法好. 图 3 中 4 条实线相差不是很大，说明它们的并行性能差不多，而且每种算法与追尾法计算时间的比值随着处理器数量的增加呈增大趋势. 当处理器数量少于 16 时，b-level、BFDS、DFDS 和 DFHDS 算法的并行计算时间均要少于追尾法，但是当处理器数量大于 32 以后，4 种算法的并行计算时间都比追尾法的要大. 分析原因是，4 种算法优先级的计算都要求出每个结点的 b-level 等级，每个结点 b-level 等级的求解都要遍历每条源点到该结点的有向路径，而有向图的各个子图被分配

到不同的处理器,算法的通信开销将随着处理器数量的增加变得十分庞大.所以在处理器数量较多的情况下,使用 b-level、BFDS、DFDS 和 DFHDS 算法的并行性能要低于追尾法.

根据以上的测试分析,我们可以得出如下结论:(1)在较少处理器上运行时,b-level、BFDS、DFDS 和 DFHDS 算法的并行性能要优于追尾法;(2)在较多处理器上运行时,追尾法的并行性能要优于 b-level、BFDS、DFDS 和 DFHDS 算法,且随着处理器数量的增多,后 4 种算法的并行性能将越来越低.所以 b-level、BFDS、DFDS 和 DFHDS 算法不适宜用来进行可扩展并行计算.

4 基于局部深度优先的优先级(PDFDS)算法

我们将结点到汇点的有向路径中最长的一条的路径长度称为该结点的深度.b-level、BFDS、DFDS 和 DFDHS 都是优先调度深度大的结点,由上一节的分析我们知道,这些优先级算法在处理器数量很多的情况下并行性能较低.结合深度优先算法和追尾法的优点,综合考虑算法的成本开销和并行方式,我们设计了一种局部深度优先的并行优先级(PDFDS)算法.该算法的具体描述如图 4 所示.

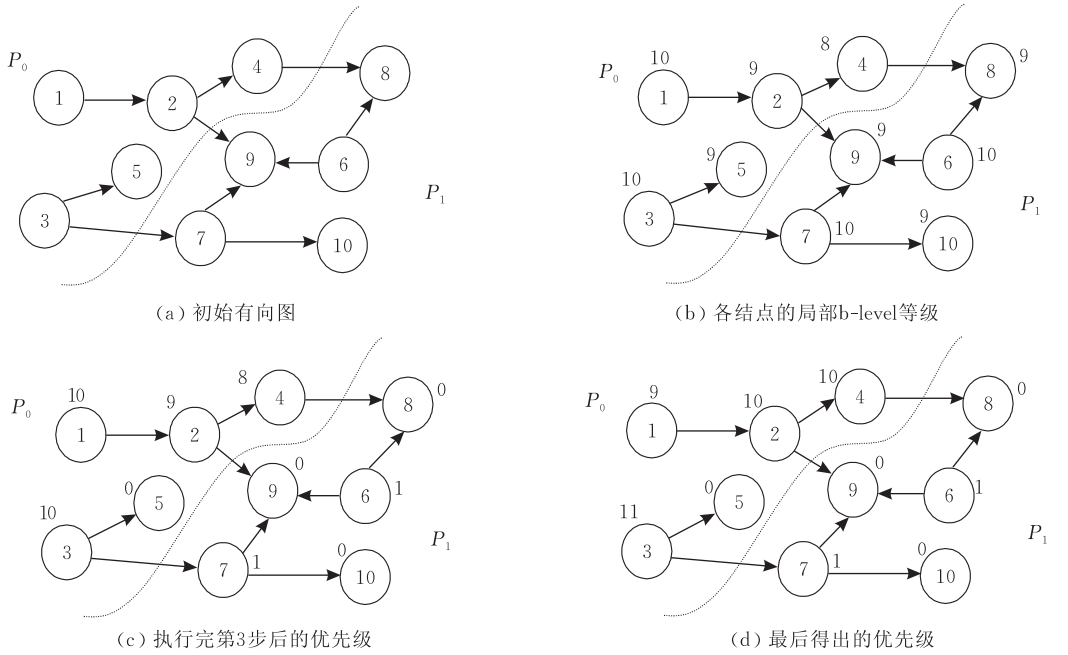
对每个角方向,每台处理机 P_i 执行:
1. 不考虑截弧,求出本地结点 x 的局部 b-level 等级:$blevel(x)$;
2. 局部区域各结点 x 的初始优先级 $priority(x)=blevel(x)$;
3. For(局部区域的所有结点 x){
  If(x 没有直接后继结点) then
    a)将 x 的优先级 $priority(x)$设为 0;
    b) For(x 的所有本地直接前驱结点 z){
      i) If(z 没有非本地后继结点,并且($priority(z)$大于 $priority(x)+1$)) then
          $priority(z)=priority(x)+1$;
        End If
      ii)对于 z 的本地直接前驱结点 z',令 $z=z'$,重复 i)直至 z 没有本地直接前驱结点;
    }
  End If
}
4. For(P_i的所有相邻处理机 P_j){
  将所有直接前驱结点的属主是 P_j 的本地结点 $x(k)$的结点信息打包,发送给处理机 P_j;
}
5. a)接收从相邻处理机发送来的消息,解包;
  For(消息中的每个结点 y){
    i) $x(j)$为 y 的本地直接前驱结点;$priority(y)$为 y 在其属主的优先级;
    ii)将 $x(j)$的优先级 $priority(x(j))$设为 $MAX+priority(y)$;如果 $x(j)$为多个接收结点的直接前驱结点,则取最大的一个 $MAX+priority(y)$;
    iii)对于 $x(j)$的本地直接前驱结点 $x(j')$,将其优先级 $priority(x(j'))$设为 $priority(x(j))-1$;如果 $x(j')$有多个本地直接后继结点,则取最大的一个 $priority(x(j))-1$;
    iv)对于 $x(j')$的直接前驱结点,令 $j=j'$,重复计算 iii)直至 $x(j)$没有本地直接前驱结点;
  }
  b)重复执行 a)直至接收完所有相邻处理机发送的消息;
6. 循环执行步 4 和步 5 $nstep$ 次

图 4 局部深度优先并行优先级(PDFDS)算法

算法第 6 步的 $nstep$ 是一个迭代控制因子.比如当 $nstep$ 取 0 时,图 4 中的第 4、5 步将不会执行,各处理机间没有通信;当 $nstep$ 取 1 时,截弧后端点的优先级可以累加到截弧前端点的优先级上;当 $nstep$ 取 2 时,在第一次迭代的基础上,将截弧后端点的优先级累加到截弧前端点的优先级上;当 $nstep$ 大于 2 时,以此类推.$nstep$ 取得越大,算法的成本开销(主要是通信开销)就越大,当 $nstep$ 取为 $P-1$ 时(P 为处理机台数),图 4 的第 4 步和第 5 步将迭代 $P-1$ 次,相当于进行了一次全局优先级的串行计算.所以 $P-1$ 为 $nstep$ 的上限,0 为 $nstep$ 的下限.为了减少算法的成本开销,一般依据并行机的性能选择 $nstep$ 值,并行机通信延迟相对越大,则 $nstep$ 取值越小,在通信延迟很大的机群可将 $nstep$ 取为 0.

图 5 为 PDFDS 算法求解过程的一个简单实例,其中 MAX 设为 10,$nstep$ 取 1.

对于图 5(a)的初始有向图,延虚线分成两个子图分属处理器 $P0$ 和 $P1$;图 5(b)为有向图各结点的局部 b-level 等级,需要注意的是,结点 7、8、9、10 的局部 b-level 等级分别为 10、9、9、9,而它们在全局有向图中的 b-level 等级分别为 9、7、8、8,其余结点的局部 b-level 等级和全局 b-level 等级相等;在图 5(c)中,汇点 5、8、9、10 的优先级被置为零,结点 8、9、10 的本地直接前驱结点 6 和 7 的优先级被置为 1,结点 5 的本地直接前驱结点 3 由于有非本地后继结点没有改变优先级;在图 5(d)中,结点 7、8、9 将各自



(a) 初始有向图　(b) 各结点的局部b-level等级　(c) 执行完第3步后的优先级　(d) 最后得出的优先级

图 5　PDFDS 算法求解过程示例

的优先级发送给它们的非本地直接前驱结点，使结点 3、4、2 的优先级分别等于 MAX 加上结点 7、8、9 的优先级，结点 1 为结点 2 的本地直接前驱结点，所以结点 1 的优先级等于结点 2 的优先级减 1.

5　算法分析

PDFDS 算法不仅考虑了结点的本地深度(图 4 的第 1～3 步)，而且考虑了结点与相邻处理机的局部深度(图 4 的第 4、5 步)，将截弧前端点的优先级按截弧后端点优先级的大小增加，可以使扫描计算时尽早计算需要通信且在相邻处理机其直接后继深度大的结点，将截弧前端点的计算结果尽早传递给截弧后端点，减少截弧后端点属主的等待时间，从而提高并行效率.

以图 5(a)为例与追尾法进行比较，PDFDS 算法和追尾法计算之后的优先级见图 6.

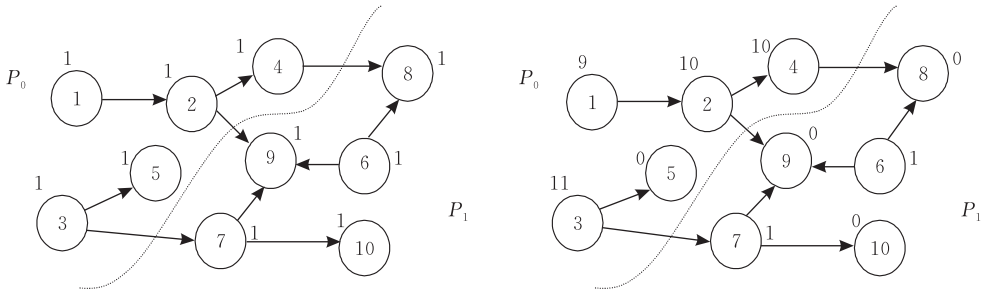
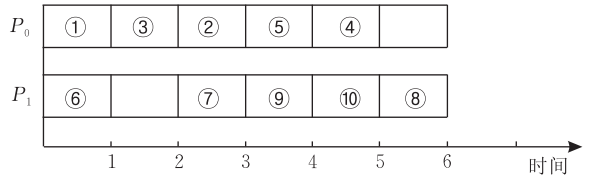


图 6　追尾法与 PDFDS 优先级比较

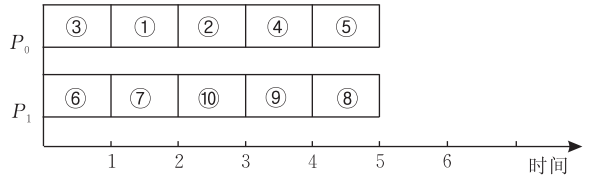
设所有结点的权重相等，所有弧的权重为 0. 图 7 为使用两种优先级进行结点调度的计算序列. 可以看到图 7 的(a)中两个处理器各有一个时间步没有计算任务，造成处理器空闲，而使用 PDFDS 算法可以避免这个时间步的空闲(见图 7 的(b)).

由图 7 看到，PDFDS 算法可以产生比追尾法更好的并行计算序列，但是 PDFDS 算法的成本开销(包括计算开销和通信开销)比追尾法要大. 怎样权衡算法成本和算法效果，下面对算法的成本开销进行分析.

假设每台处理机上平均分配了 M 个结点，将空间方向离散为 MS 个方向，对某一个方向 $i(i=1,2,\cdots,MS)$，需要与相邻处理机平均通信 η_i 次，平均通信长度为 τ. 设 PDFDS 算法第 1 步求解本地所有结点局部 b-level 值的开销为 γ_i 个计算时间步；PDFDS 算法第 2 步开销为 M 个计算时间步；设 PDFDS 算法第 3 步开销为 κ_i 个计算时间步；第 4 步至第 5 步的通信开销为 $(\alpha+\tau\times\beta)\times\eta_i$ 个时间步，其



(a) 使用追尾法优先级算法的结点调度方案,执行时间为6



(b) 使用PDFDS算法的结点调度方案,执行时间为5

图 7 使用两种优先级调度后的计算序列

中 α 为消息启动时间,β 为单位消息在网络上的传输时间,设第 5 步的计算开销为 λ_i 个时间步. 则PDFDS算法在每台处理机上的平均成本开销为

$$\sum_i \{\gamma_i + M + \kappa_i + [(\alpha + \tau \times \beta) \times \eta_i + \lambda_i] \times nstep\} \tag{1}$$

追尾法的成本开销为

$$M \times MS \tag{2}$$

其中 MS 为总方向数.

式(1)中 γ_i、κ_i、λ_i 的大小由有向图结构、区域划分以及负载平衡等因素决定,一般情况均远大于 M. 所以由式(1)与(2)对比可以看出,PDFDS 算法的成本开销要大于追尾法的成本开销,在多处理器上运行时,其通信开销占据了一定的比重. 在程序的实际运行过程中,影响并行计算性能的因素不光只有各处理机网格单元的计算顺序,如果优先级算法的成本开销过大同样会造成并行计算性能的下降. 所以,使用 PDFDS 算法必须在并行扫描计算时产生比它的成本开销更大的收益才有实际价值,也就是说使用 PDFDS 算法产生的计算序列要节约比优先级算法成本开销更多的扫描计算时间.

6 测试结果及分析

基于 MPI 设计实现了 PDFDS 算法,用于求解基于非结构网格的二维柱坐标非定常多群粒子(中子和光子)输运方程. 测试平台为一台 MPI 点点延迟时间为 5μs 的集群系统. 测试用例采用中子输运实际应用问题,S4 角方向,基于二维非结构网格离散,根据测试目的的不同,取不同的空间网格点和能群.

为了确定 PDFDS 算法中 *nstep* 在特定实验环境的较佳取值,我们在 3 个不同问题规模 $N1$、$N2$、$N3$ 上,分别取不同的 *nstep* 值进行了测试. 图 8 给出了归一化后的测试结果,即在同一问题规模上,对不同 *nstep* 取值的计算时间除以 *nstep* 取为 0 时的计算时间得到的值 *Ratio*. 其中 $N1$ 为 1024 个网格单元,$N2$ 为 4096 个网格单元,$N3$ 为 65536 个网格单元.

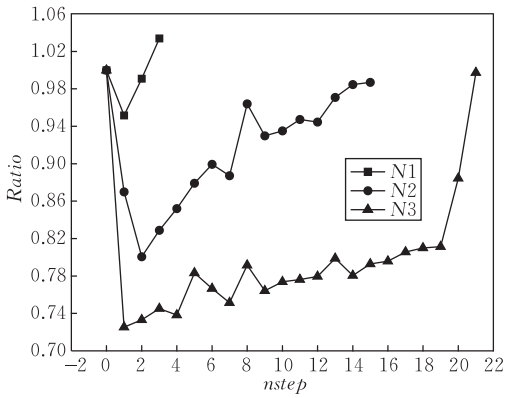


图 8 *nstep* 对并行性能的影响

从图 8 可以看出,在 3 条曲线上,*nstep* 取 1 或者 2 相比 *nstep* 取其它值能得到更好的并行性能. *nstep* 取得越大,PDFDS 算法中的通信开销越多,*nstep* 达到一定值以后,PDFDS 成本开销过大导致并行性能下降. 图 8 使用的是 64 个 CPU 进行的测试,若将 *nstep* 取最大值(63),PDFDS 算法就相当于 DFDS 算法,但很明显 DFDS 算法在这种情况成本开销太大,严重影响了并行算法的性能. 所以在本文实验条件下,将 PDFDS 算法中 *nstep* 取 1 或者 2.

为了比较 PDFDS 算法与深度优先算法以及追尾法的并行性能,我们选取了 b-level、BFDS、DFDS 和 DFHDS 4 种优先级算法与 PDFDS 算法和追尾法进行了对比测试,使用 4～128 个处理器在 4096 个非结构网格单元上测试了 150 个时间步长,测试结果见图 9.

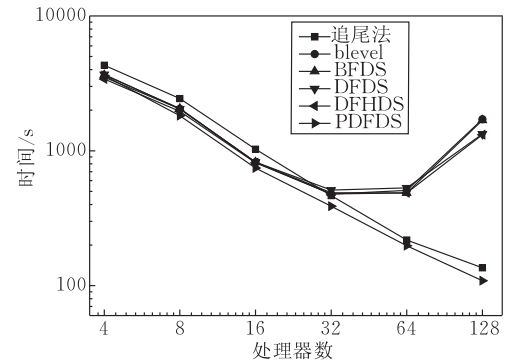


图 9 b-level、BFDS、DFDS、DFHDS、追尾法和 PDFDS 算法的并行性能比较

从图 9 可以看出，b-level、BFDS、DFDS、DFHDS 和 PDFDS 算法在处理器数量少于 16 时并行性能与 PDFDS 算法差不多，但当处理器数量大于 32 以后，4 种算法的并行性能明显下降，比 PDFDS 算法和追尾法都要差. 这是由于 b-level、BFDS、DFDS 和 DFHDS 算法在较多处理器上运行时，算法的成本开销太大导致了并行性能的下降，所以 b-level、BFDS、DFDS 和 DFHDS 算法不适合用来进行可扩展并行计算.

从图 9 中还可以看出，虽然使用 PDFDS 算法在 4～128 个处理器上的程序运行时间均要少于追尾法，但随着处理器台数的增多，性能提高的比率逐步减少，两条曲线有相交的趋势. 这是因为随着处理器台数的增加，分配到每台处理器上的网格点逐步减少，上下游关系较简单，调度算法的队列组合的种类减少，性能趋于一致.

为了更准确说明 PDFDS 算法在大规模调度问题上的优势，突出扫描调度算法的计算效果，我们将群数降为 1，处理器台数取为 256，对追尾法、弧度法、坐标法和 PDFDS 算法进行了对比测试，图 10 给出了测试结果. 从图 10 可以看出，当单个处理机上的网格点数较少时，PDFDS 算法的并行计算时间略长于追尾法、弧度法和坐标法，随着网格点数的增加，PDFDS 算法的并行计算时间比追尾法、弧度法和坐标法显著减少，网格点越多，性能提高越明显，说明 PDFDS 算法更适合求解大规模计算问题.

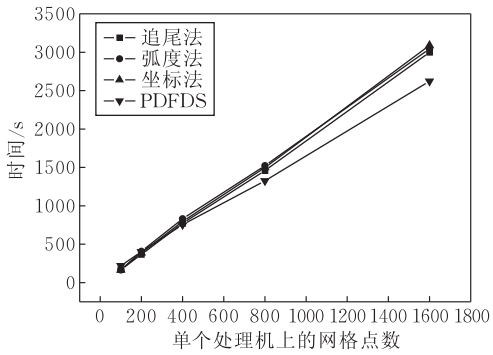


图 10　追尾法、弧度法、坐标法和 PDFDS 算法的并行性能比较

为进一步测试 PDFDS 算法对求解实际应用问题的可扩展性，我们把 PDFDS 算法移植到我们提出的一种改进的求解输运方程的并行 Sn 扫描算法[12]程序框架中，将计算问题的群数取为 24 群，网格单元取为 65536 个，S4 角方向，表 1 给出测试结果. 测试结果显示，并行算法具有良好的并行加速效果，扩展到 1024 个处理器时，相对 64 个处理器的并行效率达到 96%，说明 PDFDS 算法具有很好的适应性.

表 1　在 65536 个网格单元上测试

CPU 台数	运行时间/s	相对加速比	相对并行效率
64	1512	1.00	1.00
128	753.1	2.01	1.01
256	342.8	4.41	1.10
512	172.5	8.76	1.10
1024	98.47	15.35	0.96

7　结　论

通过前面的分析和测试我们知道，包括 b-level、BFDS、DFDS 和 DFHDS 算法在内的深度优先策略不适合进行可扩展并行计算. 而我们设计的局部深度优先的优先级(PDFDS)算法具有局部性、通信量小、优先级队列好等特点. 通过性能分析与测试，可以看出基于非结构网格区域分解，PDFDS 算法使用结点的局部深度作为结点调度的优先级，与现有的优先级算法比较，可以获得更加合理的并行计算序列，具有更好的并行计算效果，对于大规模计算问题，可以扩展到 1024 个处理器，相对于 64 个处理器的并行效率达到了 96%.

此外，本文给出的 PDFDS 算法可以用于非结构网格高维输运问题的并行计算，并且需要进一步验证对不同非结构网格划分问题的有效性.

致　谢　本文工作得到北京应用物理与计算数学研究所莫则尧研究员的大力支持和帮助，在此表示感谢；感谢阳述林研究员、魏军霞等同志与作者的有益讨论和宝贵意见！

参　考　文　献

[1] Du Shuhua et al. Computer Imitation of Transport Problems. Changsha: Hunan Science & Technical Press, 1989(in Chinese)
(杜书华等. 输运问题的计算机模拟. 长沙：湖南科学技术出版社，1989)

[2] Plimpton S, Hendrickson B, Burns S, McLendon W. Parallel algorithms for radiation transprt on unstructured grids// Proceedings of the SuperComputing'2000. Dallas, Texas, 2000

[3] Mo Ze-Yao, Fu Lian-Xiang, Yang Shu-Lin. Parallel pipelined S_n sweeping algorithm for neutron transport on unstructured grid. Chinese Journal of Computers, 2004, 27(5): 587-595(in Chinese)

(莫则尧，傅连祥，阳述林. 非结构网格上求解中子输运方程的并行流水线 S_n 扫描算法. 计算机学报，2004，27(5)：587-595)

[4] Davis E W，Patterson J H. A comparison of heuristic and optimum solutions in resource-constrained project scheduling. Management Science，1975，21(8)：944-955

[5] Kolisch R. Project scheduling under resource constraints-efficient heuristics for several problem classes. Heidelberg，Germany：Physica，1995

[6] Pautz Shawn D. An algorithm for parallel Sn sweeps on unstructured meshes. Nuclear Science and Engineering，2002，140(2)：111-136

[7] Li K Y，Willis R J. An iterative scheduling technique for resource-constrained project scheduling. European Journal CLF Operational Research，1992，56(3)：370-379

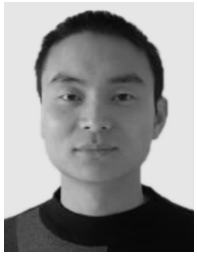
[8] Mo Ze-Yao，Zhang Ai-Qing，Cao Xiao-Lin. Towards a parallel framework of grid-based numerical algorithms on DAGs//Proceedings of the 20th IEEE International Parallel and Distributed Processing Symposium. Rhodes Island，Greece，2006

[9] Boctor Fayez F. Some efficient multi-heuristic procedures for resource-constrained project scheduling. European Journal of Operational Research，1990，49(1)：3-13

[10] Thomas P，Salhi S. An investigation into the relationship of heuristic performance with network-resource characteristics. Journal of the Operational Research Society，1997，48(1)：34-43

[11] Zhang Ai-Qing，Mo Ze-Yao. A new scheduling algorithm for digraph-based parallel computing//Proceedings of the 2006 National Annual Conference on High Performance Computing. Beijing，2006(in Chinese)
(张爱清，莫则尧. 有向图并行计算中一种新的结点调度算法//2006 全国高性能计算学术年会. 北京，2006)

[12] Zhou Di-Yu，Liu Jie. An improved parallel S_n sweeping algorithm for solving transport equations. Computer Engineering and Science，2008，30(4)：62-65(in Chinese)
(周涤宇，刘杰. 一种改进的求解输运方程的并行 S_n 扫描算法. 计算机工程与科学，2008，30(4)：62-65)



ZHOU Di-Yu，born in 1978，M. S.，engineer. His research interests include parallel algorithms and high performance computing.

LIU Jie，born in 1969，Ph. D.，associate professor. His research interests include parallel algorithms，parallel benchmarks and high performance computing.

Background

This research work is supported by the National Natural Science Foundation of China under grant Nos. 60673150，60970033 and the National High Technology Research and Development Program (863 Program) of China under grant No. 2008AA01Z137.

Particle transports are important in modeling many physical phenomena. Discrete ordinates (S_n) equations derived from the first-order form of the particle transport equations are usually solved by sweeping the particle flux across the computational grid. Tasks (cell-angle pairs) assigned to a processor cannot begin until cells upstream from them have been solved，and the processor may be idle for some time if the upstream tasks are on the other processors. For unstructured grids，there are several interesting challenges for the assignment of tasks to processors and the scheduling algorithm for the computation of grids.

The research work in this paper focuses on the scheduling algorithm which implement on distributed-memory parallel machines where the parallel computation is based on the geometry domain decomposition on the unstructured grids.