

混合模型下 LAMMPS 并行探究

豆育升,刘相金,白明泽,孙延维

(重庆邮电大学,计算机科学与技术学院,重庆 400065)

摘 要: 面对分子动力学模拟在串行计算上的速度瓶颈问题,设计一种并行计算模型来加快计算速度,在分子动力学模拟软件 LAMMPS 下引入 MPI+OpenMP 并行模型,这个模型可以在 LAMMPS 下提供两级并行处理,充分利用集群优势,提高模拟计算速度.实验结果表明,在十节点下,采用并行模型加速比至少提高 8 倍以上.

关键词: 分子动力学;集群;MPI+OpenMP;消息传递;LAMMPS

中图分类号: TP399

文献标识码: A

文章编号: 1000-7180(2014)10-0143-04

Study on Research for LAMMPS Based on Hybrid Parallel

DOU Yu-sheng, LIU Xiang-jin, BAI Ming-ze, SUN Yan-wei

(Computer Science and Technology College, Chongqing University of Posts and
Telecommunications, Chongqing 400065, China)

Abstract: Facing the speed bottleneck problem in serial calculations on the molecular dynamics simulation, we design a parallel computing model to accelerate the speed of calculation. the MPI+OpenMP parallel model is drowed into molecular dynamics simulation software LAMMPS. This model can provide the parallel processing in LAMMPS. It is make full use of cluster advantages, and improves the simulation speed. Experimental results show that by ten nodes, Using the parallel model, The speedup is improve at least 8 times.

Key words: molecular dynamics; cluster; MPI + OpenMP; message passing; LAMMPS

1 引言

分子动力学(Molecular Dynamics, MD)^[1]在高性能计算领域有重要的应用,它通过各个粒子的初始速度,求解各粒子的运动方程,从而得到不同时刻粒子的位置信息,然后通过统计物理方法计算出在实验室条件下较难得到的粒子之间的宏观属性,这些属性如物质的微观结果、热力学性质等.作为一种重要的科学研究方法,MD模拟已经在众多中领域得到广泛应用.MD模拟对计算资源尤其是计算速度有着巨大的需求,是高性能计算的典型应用之一.因此,对串行的MD程序进行并行化已经很有必要,它能够快速完成MD计算任务.最常用的MD软件有LAMMPS就是本文研究的一个应用,最新的LAMMPS本身自带OpenMP包,可以支持线程级并行.

随着高性能计算的快速发展,采用对称多处理

(Symmetric Multi-Processing, SMP)的集群系统逐渐成为研究高性能计算机的主流体系结构.主要通过MPI来增加计算核心数.然而,普通应用的基于消息传递的MPI并行编程由于信息传递的时间过长,从而带来性能损失.最近提出一种基于混行编程的模型,其节点间采用MPI消息传递机制进行通信,其节点内部采用基于共享存储模式的并行算法设计,该模式在任务切换时增大了任务粒度,从而有效降低在集群节点间的通信量.

MPI在并行MD领域有广泛的应用, LAMMPS大多采用MPI实现并行,可在MPICH2下等标准的MPI下实现并行,在共享存储里面主要由OpenMP^[2]、TBB^[3]、Pthreads^[4-11]等,其中, Pthreads需要专门相关人员掌握操作系统的底层知识, TBB程序不支持C,只支持C++. OpenMP相对简单并且LAMMPS本身包含OpenMP包,所

以本文提出了将 MPI+OpenMP 混合模型应用到 MD 模拟中,来提高 MD 模拟的并行性能,从而实现在 LAMMPS 下基于 MPI+OpenMP 混合编程,测试表明,MPI+OpenMP 可以明显提升 MD 性能.

2 分子动力学算法与程序

2.1 分子动力学模拟算法

MD 模拟首先对模拟体系进行初始化,由于许多粒子并不对粒子 i 的力有贡献.因此,在计算力的时候可以排除那些并不起作用的粒子,这样可以根据每个对粒子产生力的范围,确定粒子近邻表;之后,根据近邻表计算粒子之间的作用力,以及获得粒子的新坐标.在循环过程中,可能某些粒子离开近邻表,这样就要每隔一段时间需要更新近邻表,并继续循环里面的步骤,直到计算结束.

MD 模拟中创建近邻表和力的计算是最费时的部分.整个模拟过程中,约占全部时间的 90% 以上.在力的计算过程中,假设粒子 i 的近邻表包括 j ,就可以运用式(1)来计算在 x 方向上的力.

$$f_x(r) = \frac{\partial u(r)}{\partial x} = -\left(\frac{x}{r}\right)\left(\frac{\partial u(r)}{\partial r}\right) \quad (1)$$

式中, r 是粒子 i 跟 j 之间的距离.

根据 Lennard-Jones 势能公式^[12]:

$$u(r) = 4\left(\frac{1}{r^{12}} - \frac{1}{r^6}\right) \quad (2)$$

化简得到如下:

$$f_x(r) = \frac{48x}{r^2}\left(\frac{1}{r^{12}} - 0.5\frac{1}{r^6}\right) \quad (3)$$

计算完所有的作用力,根据积分运动方程获得下一时刻的位置.

2.2 LAMMPS 模拟软件

LAMMPS 是应用广泛的 MD 软件,它遵循开源协议并且预留有添加新功能模块的接口;同时它采用空间分解并行算法,并行性能较好;此外,它还支持多数常见的势能函数.

当 LAMMPS 启动的时候,首先要创建一个 LAMMPS 对象,随后该对象创建一系列类的实例,这些实例主要是负责 MD 计算的整个任务.

LAMMPS 在进行 MD 模拟的时候,主要是一系列类的实例进行运算,其中并行计算的流程图如图 1 所示,LAMMPS 本身提供的 read_data 命令,首先读入一个 data 文件进行初始化,然后可以按照给每个进程分配一个计算子集来实现并行,主进程主要来负责任务的分配、数据划分和收集各节点计算结果;从进程接收主进程的数据,计算相应的作用

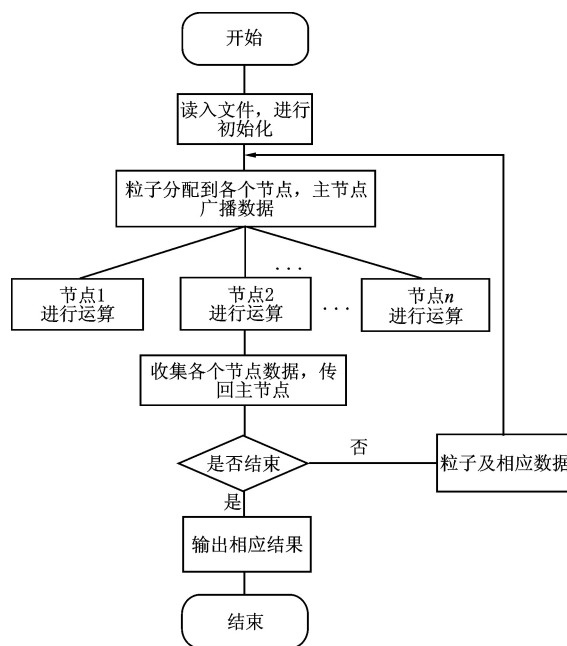


图 1 分子动力学并行计算流程图

力,最终把结果传输回主进程.一般 MD 程序计算花费的时间主要是各粒子力的计算和构建近邻表,由于在一次 MD 计算中,基本上只需要构建一次近邻表,所以粒子力的计算占据了大部分运行时间.所以这两部分都是 MD 并行化的主要目标.

3 MPI+OpenMP 混合编程模型

3.1 OpenMP 并行模拟

OpenMP^[6]主要用于共享内存并行系统的多线程程序设计的一套指导性的编译处理方案.OpenMP 支持的编程语言包括 C 语言、C++ 和 Fortran,相对于其他共享模型,OpenMP 的主要优点是:

(1)并行设计简单.使用 OpenMP 设计简单,只需一条编译制导语句就能实现多线程并行.

(2)扩展好.OpenMP 可以支持多种语言,不像 TBB 只支持 C++.

(3)移植性好.OpenMP 规范中规定的制导指令、运行库和环境变量,可以支持多种平台的运行.

OpenMP 的执行模式基本上是采用 Fork-Join 模式^[12].Fork-Join 模型在开始执行的时候,只有一个主线程存在.主线程在整个程序的运行过程中,当遇到编译制导语句的时候,就会派生出(或者是创建新线程或者唤醒已存在的子线程)子线程来执行并行任务.在并行任务计算时,主线程和子线程可以一起共同工作;在并行代码结束的时,子线程退出或者挂起,控制整个过程的流程就又回到主线程中,从而完成整个计算任务.

在 LAMMPS 软件中进行 OpenMP 并行,首先必须确定安装了 UESR-OMP 包.可以用如下命令来安装:

```
make yes-user-omp
```

make openmpi //可根据自己的需要选择一个执行

在安装过程中也许会出错,要在 src 目录下 make clean,确保之前的安装残余被清除,然后再进行上面的命令.

安装完之后,就要修改里面的函数,修改 USER-OMP 包,使其能够并行计算.

(1)修改 neigh_half_bin_omp.cpp 里面的 void Neighbor::half_bin_newton_tri_omp(NeighborList * list) 函数,加入代码 #pragma omp parallel default(none) shared(list),使近邻表的循环构建做成并行处理.

(2)修改 pair_lj_cut_omp.cpp c 文件里面的 void PairLJCutOMP::compute(int eflag, int vflag) 函数,加入 #pragma omp parallel default(none) shared(eflag, vflag) 循环语句,使计算力的时候循环进行.

3.2 MPI+OpenMP 混合并行

图 2 是基于 MPI+OpenMP 混合模型的计算框架图.该图表示了 2 个节点之间计算的情况,图中每个矩形内表示一个节点,每个节点有一个进程,框内的流程就是进程的计算流程.进程进行 OpenMP 计算的步骤:首先初始化任务调度器,然后将到达的进程分配到各个线程进行计算,各个线程计算完毕之后,全部交给 MPI, MPI 也可以进行局部计算、通信、同步等操作. OpenMP 并行计算跟 MPI 局部计算可以同时进行,在整个过程结束的时候, MPI 进程也相应的结束.

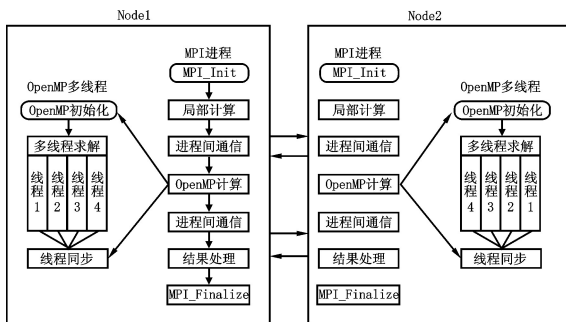


图 2 MPI+OpenMP 混合模型

从图 2 可以看出, LAMMPS 在节点并行是基于 MPI 通信的,因此,在节点间采用粗粒度消息传

递并行,在节点内采用细粒度共享内存.这样能更好地发挥整个体系结构的优点.

4 LAMMPS 嵌入 MPI+OpenMP 多层并行

接下来,把 LAMMPS 嵌入到混合多层并行的环境下,按照每步的要求配置 LAMMPS,将测试文件放在 LAMMPS 的 src 目录下.下面是整个配置过程:

(1)建立 ssh 信任连接,在用户目录下,配置 .ssh,使用 MD5 加密算法,生成密钥,然后在集群里面就可以共享密钥,主要是解决在节点间参与计算 MD 的时候,相互访问资源可以不需要输入密码.

(2)在 1 号节点下面安装 openmpi,主要是加入 MPI 库,使其能够在节点间互相访问资源,安装位置/usr/lib64/openmpi.

(3)测试 MPI,启动每个节点的 mpd,建立节点文件/hosts,主要包括节点名字跟 IP 地址,然后查看节点之间的通信环.

(4)安装 LAMMPS,安装 USER-OMP 包,修改 Makefile.g++ 文件,将 -fopenmp 加入 Makefile 文件中,编译 Makefile 文件,使模型适应有 OpenMP,将 OpenMP 带入整个模型.

最后,在测试命令如下: export OMP_NUM_THREADS = N (N 代表测试的线程数); lmp_openmpi -sf omp -in in.lj.

5 实验结果及分析

为了验证混合并行算法的正确性和时效性,本文在一个 SMP 集群架构上进行了测试.十台服务器作为计算节点,每个节点配备 Intel(R) Xeon(R) CPU E5-2620 0 @ 2.00 GHz(12 核)和 64 GB 内存.操作系统为 Red Hat Enterprise Linux Server release 6.3,内核版本 2.6.32, MPI 环境为 openmpi-1.2.6 版, LAMMPS 版本是 LAMMPS-22Mar13.

LAMMPS 测试参数:晶格个数为 $30 \times 30 \times 30$, 原子个数 108 000, 势能选取最简单的 Lennard-Jones 势能,模拟步数 2 000.我们运用相对加速比 S 来测试整个模型的性能.

$$S = \frac{T_{\text{serial}}}{T_{\text{parallel}}} \quad (4)$$

实验结果看出,总共有 9 个线程,总的来看制约分子动力学模拟时间是进程之间的通信时间.在用 OpenMP 优化之后通信时间(3×3)所占比例为

$$12.9647/35.1455 = 36.89\%,$$

跟只有 MPI(9×1)没有用 OpenMP 优化通信时间

所占比例 $20.604/40.276=51.16\%$ 小很多. 通信时间的增长因为并发消息条数太多并且每条消息的数据量较大造成的, 它们对通信资源的剧烈竞争产生了网络拥塞. 混合模型虽然增加了单条消息的大小, 但是 OpenMP 优化之后, 由于成倍减少了并发消息条数, 在单节点下共享内存交换数据, 反而减少了节点之间的通信时间. 进而说明线程的通信时间比进程通信所花的时间短. 在引进多节点混合并行下, OpenMP 优化后能够减少进程之间的通信时间, 结果表明优化后的效果更能减少模拟时间.

由上面所述, MPI + OpenMP 混合模型嵌入 LAMMPS 中, MPI 可以增加计算分配线程的数量, 使 MD 模拟计算具有很好的扩展性; 在每个 MPI 进程下, 引入 OpenMP 优化之后, 使每个节点之间的通信时间大大降低, 缓解 MPI 节点之间的通信时间过长的不足, 从而表明 MPI + OpenMP 混合模型有利于 MD 模拟的计算.

6 结束语

本文针对 SMP 集群环境特点给出了 LAMMPS 的多重并行设计, 该方法实现了 LAMMPS 在 MPI + OpenMP 下的混合并行设计. 数据表明, 在 LAMMPS 的多重并行下其计算的加速比得到了提高, 分析表明, 多重混合模型可以有效的减少运算时间, OpenMP 可以达到非常好的共享内存, 减少 MD 模拟时间; MPI 在集群里面消息传递, 使 MD 模拟具有很好的扩展性. 在 SMP 集群上进行 MPI + OpenMP 混合并行应用, 对于提高 MD 并行计算性能研究具有较好的促进作用.

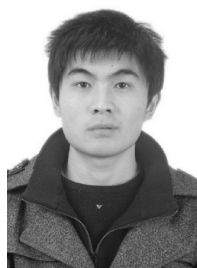
参考文献:

- [1] 潘卫, 陈燎原, 张锦华, 等. 基于 SMP 集群的 MPI + OpenMP 混合编程模型研究[J]. 计算机应用研究, 2009, 26(12): 1-2.
- [2] LAMMPS molecular dynamics simulator[EB/OL]. [2013-12-07]. <http://lammps.sandia.gov/>, September 13, 2012.
- [3] NAMD -scalable molecular dynamics[EB/OL]. [2013-12-07]. <http://www.ks.uiuc.edu/Research/namd/>, September 13, 2012.
- [4] Kale L, Skeel R, Bhandarkar M, et al. NAMD2: greater scalability for parallel molecular dynamics[J]. Journal of Computational Physics, 1999, 151(1): 283-312.
- [5] GROMACS[EB/OL]. [2013-12-07]. <http://www.gromacs.org/>, September 13, 2012.
- [6] OpenMP application program interface[EB/OL]. [2013-12-07]. <http://www.openmp.org/>, September 13, 2012.
- [7] 黄铠. 可扩展并行计算技术、结构与编程[M]. 北京: 机械工业出版社, 2000.
- [8] 赵永华, 迟学斌. 基于 SMP 集群的 MPI + OpenMP 混合编程模型及其有效实现[J]. 微电子学与计算机, 2005, 22(10): 7-11.
- [9] 白明泽, 程丽, 豆育升, 等. 基于 OpenMP 的分子动力学并行算法的性能分析与优化[J]. 计算机应用, 2012, 32(1): 163-166.
- [10] Frenkel Smit, 著, 分子模拟-从算法到应用[M]. 汪文川, 译. 北京: 化学工业出版社, 2002.
- [11] Nichols B, Buttler D, Farrell J P. Pthreads programming[M]. Sebastopol: O'Reilly & Associates, 1996.
- [12] Reinders J. Intel threading building blocks 编程指南[M]. 聂雪军, 译. 北京: 机械工业出版社, 2009.

作者简介:



豆育升 男, (1953-), 教授, 博士生导师. 研究方向为计算化学、高性能计算.



刘相金 男, (1987-), 研究生. 研究方向为并行计算.



白明泽 男, (1982-), 博士, 讲师. 研究方向为并行算法、设计、分子动力学模拟.



孙延维 男, (1979-), 研究生. 研究方向为高性能计算.