一种适用于机群系统的用户层消息传递机制*

周桂林，戈 弋，李三立，黄震春，马群生

(清华大学 计算机科学与技术系，北京 100084)

E-mail: huang@tirc.cs.tsinghua.edu.cn

http://www.tsinghua.edu.cn

摘要：网络通信中的软件开销是目前影响机群系统性能的瓶颈，为了提高机群系统的通信性能，提出了一种用于机群系统的用户层快速消息传递机制ULFM(user-level fast message)，并在TH-GBNet上加以实现. ULFM以通信缓冲区的管理为核心设计了一种易于实现的用户层网络访问接口，通过综合采用用户层网络接口访问、零拷贝数据传递、精简通信协议等技术，可以有效地降低机群系统的通信开销，为应用程序提供实际的低延迟、高带宽的通信性能. 在TH-GBNet的实现中，ULFM节点间16字节数据包的单向延迟减小到仅为6μs，2K字节数据包的带宽达40MB/s.

关键词：机群系统；用户层；通信协议；消息传递；开销

中图法分类号：TP315 **文献标识码**：A

随着高速互联网络的出现和处理机性能的不断提高，利用网络将工作站/高档微机互连构筑成的机群系统正逐步成为大规模并行计算的主要平台. 在机群系统中，各节点通过相互之间的消息传递来实现数据的交换和同步，共同协作完成统一的任务，节点之间的通信效率对机群的整体性能有关键影响. 文献[1]指出在不提高节点间通信性能的情况下，单处理节点CPU的性能可提高32倍，而系统整体性能的提高不到10倍.

现代开关式网络，如ATM，Myrinet[2]等高速网络的出现已经可以为机群系统提供每秒几百兆到几G的通信带宽，但对于末端用户和应用程序来说，并没有从通信硬件性能的大幅提高中得到相应的收益. 其主要原因在于，传统机群系统中的通信都是基于标准的网络接口和通信协议，消息发送和接收端的协议处理时间、数据拷贝开销、缓冲区管理开销以及消息发送过程中操作系统在系统核心态和用户态之间的切换等时间开销，占整个消息传递延迟的绝大多数. 在硬件速度日益提高的情况下，与这部分开销相比，数据实际通过网络传输的时间甚至可以忽略不计. 例如，在Intel Paragon上，消息传递的50μs延迟中只有1μs是数据通过网络硬件传输的延迟，其他时间都花费在消息发送和接收端的软件开销上[3].

为降低通信的软件开销，为应用程序提供实际的低延迟、高带宽的通信性能，本文设计了一种用于机群系统的用户层快速消息传递机制ULFM(user-level fast message)，并在TH-GBNet[4]中加以实现. 本文第1节介绍ULFM减小通信中软件开销的主要思想. 第2节和第3节是ULFM的

* 收稿日期：1999-08-21；修改日期：2000-02-18

基金项目：国家自然科学基金资助项目(69973024)

作者简介：周桂林(1972--)，男，湖北孝感人，博士，工程师，主要研究领域为高性能计算环境，快速消息传递机制；戈弋(1973-)，男，浙江义乌人，博士，工程师，主要研究领域为并行处理，高性能计算，计算机体系结构；李三立(1935-)，男，上海人，博士，教授，博士生导师，中国工程院院士，主要研究领域为高性能计算，指令级并行，并行系统体系结构；黄震春(1975-)，男，辽宁辽阳人，博士生，主要研究领域为高性能微体系结构，网格计算；马群生(1942)，男，河北秦皇岛人，副教授，主要研究领域为计算机系统结构，并行分布处理结构.

设计及其在 TH-GBNet 中的实现.第 4 节给出性能评测.第 5 节是相关研究工作及总结.

1 ULFM 的主要思路

我们分析最普遍的机群系统应用程序进行发送和接收消息的语句:

Send(dstAddress, sendBuffer, msgLength);

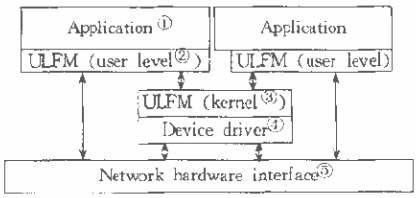
Receive(srcAddress, recvBuffer, msgLength).

可以看出,消息传递主要涉及消息缓冲区、消息长度以及地址等信息.消息传递实际上就是完成发送缓冲区中消息长度的数据到目的地址接收缓冲区中的传输.在传统网络接口中,通信的应用程序具有各自的通信缓冲区,系统还维护一个统一的缓冲区,作为数据从应用缓冲区进出网络通道的中转站.其目的是实现统一的网络访问接口,提供应用间网络接口访问的保护机制,同时便于在目的节点完成统一的消息数据接收.但是,这种机制不可以避免地增加了相应的缓冲区管理、数据拷贝以及系统切换等开销.

从上面的分析还可以看出,经过缓冲区的数据传输和缓冲区的管理是整个消息传递过程的关键.ULFM 的主要思路就是把通信缓冲区作为整个消息传递过程的核心,设计相关的支持用户层访问的应用网络接口,同时综合零拷贝消息传递、精简通信协议等技术,以减小通信过程中的软件开销.

2 ULFM 消息传递机制

ULFM 是一层软件结构,位于网络接口硬件和应用程序之间,为应用程序提供网络访问接口的抽象以及该抽象基础上的消息传递.如图 1 所示,ULFM 由两部分组成,分别位于系统核心态和用户态.位于用户态的部分直接与应用程序链接,为应用程序提供通信原语;位于核心态的部分对网络访问的临界资源进行管理和控制,实现多应用程序网络接口访问时的保护机制.应用程序通过 ULFM 提供的通信接口进行消息的发送和接收.



①应用程序,②用户级,③核心,④设备驱动程序,⑤网络接口硬件.

Fig. 1 The position of ULFM in communication

图1 ULFM在通信结构中的位置

2.1 应用网络接口

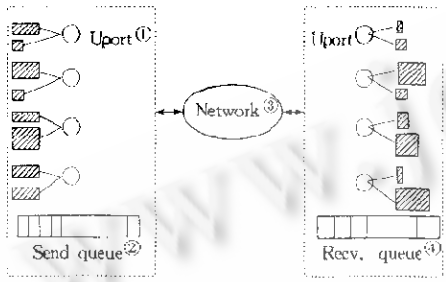
ULFM 把应用程序访问的网络接口抽象为通信端口(uport)和发送队列两个部分,如图 2 所示.通信端口 uport 的结构如图 3 所示,每个端口包含两个固定的位于物理地址空间的缓冲区,分别缓存进出该端口的消息数据.同时,端口中还包含与通过缓冲区进行消息传递的其他相关信息,包括缓冲区标记、进出缓冲区的数据长度以及发送缓冲区数据的目的地址信息等.端口从逻辑上可以分为相对独立的两部分,分别用于消息的发送和接收.各端口之间相互独立.系统通过一个端口表对所有的通信端口进行统一的管理.

2.2 消息的发送和接收

在 ULFM 网络接口中,应用程序通过 uport 通信端口进行消息的发送和接收.uport 由系统根据应用的申请进行建立,端口中缓冲区的大小由应用程序根据通信的需要来指定.端口建立以后存储映射到应用程序空间,由应用程序直接进行访问,包括直接在物理地址的端口缓冲区中组织发送

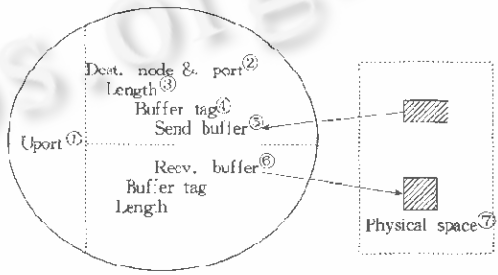
和接收的数据.通过通信端口可以进行多次数据的发送和接收.

应用程序通过 uport 进行消息发送的过程如图 4 所示.发送消息时,应用程序首先在发送缓冲区中组织消息数据,并指定与此次消息发送相关的信息,包括发送数据的实际长度、目的节点与目的端口等;然后将该端口号推入发送队列,进行消息的发送.实际的消息传输在端口号进入发送队列后进行:网络接口根据端口号查询端口表,获取目的地址信息、发送缓冲区地址、数据长度等信息,然后直接将消息数据从物理地址消息缓冲区中发送到网络.在接收端,消息数据到达后由网络接口对数据包头信息进行分析,从中获取目的端口,然后根据目的端口号查询端口表,获取相应的接收缓冲区的地址,将消息数据直接接收到目的缓冲区中,并置相应的缓冲区标志位.应用程序通过查询端口接收缓冲区的标记来判断新消息数据的到达.



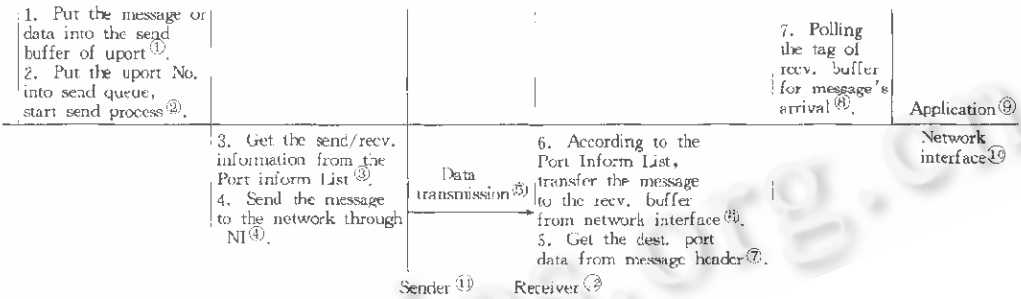
①端口,②发送队列,③网络,④接收队列.

Fig. 2 ULFM network interface
图2 ULFM网络接口示意图



①端口,②目的结点及端口,③长度,④缓冲区标记,⑤发送缓冲区,⑥接收缓冲区,⑦物理空间.

Fig. 3 The uport of ULFM
图3 ULFM通信端口



①在端口发送缓冲区中准备消息数据,②将端口号推入发送队列,进行发送,③根据端口号查询端口信息表,获取相关的发送信息,④将消息数据直接从缓冲区发送到网络,⑤数据传输,⑥分析数据包头,获取目的端口信息,⑦根据端口号查询端口信息表,然后直接将数据从网络传输到目的端口接收缓冲区中,⑧查询端口接收缓冲区标记判断是否到达,⑨应用,⑩网络接口,⑪发送方,⑫接收方.

Fig. 4 Communication procedure of ULFM
图4 ULFM消息传递过程示意图

在 ULFM 的消息传递中,通过存储映射机制,数据从应用缓冲区映射到的网络接口硬件可以访问的物理内存区域直接发送到网络.消息接收时,也可以直接将数据接收到目的缓冲区中,供应用程序直接访问.在整个消息传递过程中,可以没有任何冗余的数据拷贝,做到了真正零拷贝的数据传递.

2.3 缓冲区管理

在 ULFM 中,所有的通信缓冲区都作为端口的一部分在端口建立时进行分配,并且在端口的整个生命周期中都是固定的.端口建立后就可以在通信缓冲区中进行多次的消息发送和接收.缓冲区的管理开销主要在建立端口时,在以后的消息传递过程中不需要任何管理开销.为避免消息传递

过程中的数据拷贝,通信缓冲区都直接在网络硬件设备可以直接访问的物理内存空间中进行分配,然后通过映射,到达用户空间.随着内存价格的下降和系统配置的增加,物理内存已经不再是系统最紧缺的资源.因此,直接在物理内存空间分配缓冲区虽然会消耗一定的内存资源,但从提高性能的角度看仍然是值得的.

应用程序和网络接口都可以直接对缓冲区进行访问和操作,二者之间通过缓冲区标记对缓冲区数据进行同步和维护.缓冲区标记分为1和0两种状态,1表示该缓冲区中存在有效数据,0表示缓冲区空闲,可以使用.缓冲区标记的状态变化如图5所示.在发送方,应用程序在将端口号推入发送队列启动消息发送前,对发送缓冲区的标志位进行置位,表示缓冲区中存在着有效的待发送数据;网络接口在完成发送缓冲区中消息数据的传输之后,对其标志位进行清除,以通知应用程序发送结束,可以在发送缓冲区中进行下一次消息传输.数据到达接收方后,网络接口对目的缓冲区的标志位进行检查.如果置位,则表明该缓冲区中还有有效的数据,为避免覆盖或者通信阻塞造成死锁,要产生中断,由系统进行处理,将到达的数据放入进程的消息缓存队列;如果没有,就直接将数据传输到缓冲区,然后置该缓冲区标志位,表明该端口接收缓冲区中有新的数据到达.应用程序通过查询端口接收缓冲区标志位进行数据的接收.缓冲区数据处理结束后,需要将接收标志位置零,以便网络接口进行该端口数据的接收.

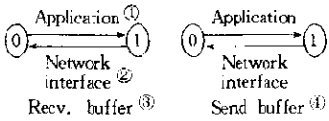


Fig. 5 State translation of tag
图5 缓冲区标记状态变化图

2.4 通信协议

在ULFM通信机制中,通信协议在通过uport端口缓冲区通信的基础上,完全在用户层实现.其目的是为机群节点之间提供可靠的、有序的数据传输.因此,不需要实现类似于TCP/IP的复杂通信协议.最基本的通信协议包括以下两个方面的内容:

(1) 数据包的校验、应答和重传机制,为应用程序提供无差错的数据传输.

(2) 进行必要的消息数据缓存管理,以确保消息传递的顺序.ULFM中消息的发送和接收以及协议的实现都在应用程序空间完成,因此会出现消息数据的到达顺序与接收处理函数不一致的情形.如图6所示,进程$P0$中有两个接收函数,第1个函数接收来自于进程$P1$的消息,第2个函数接收来自于进程$P2$的消息.在实际的程序运行环境中,进程$P2$的消息可能先于进程$P1$的消息到达$P0$,也就是说,在进程$P0$的第1个接收函数中可能首先收到进程$P2$发来的消息数据,这时就需要在第1个接收函数中将该消息进行暂时的缓存,以便接收进程$P0$发送的数据.$P0$中的第2个接收函数只需将缓存的消息数据取出即可.采用消息缓存可以保证没有一个消息能够长期独占进程的数据缓冲区,因此不会出现进程之间循环等待的死锁局面.

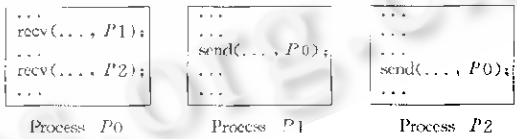


Fig. 6 The sequence of messages in process communication
图6 进程间消息传递顺序示意图

2.5 实现问题

ULFM通信机制作为一种独立于网络接口硬件的通信结构,可以在标准的网络接口硬件中通过软件进行实现,也可以利用定制的网络接口进行支持.标准的网络接口卡一般位于系统的I/O总线上,通过PIO和(或)DMA数据传输的形式与系统相连.在这种网络接口上实现ULFM,可以由ULFM核心部分在系统的物理内存空间维护一个端口表,为应用程序提供uport通信端口,发

送队列则可以通过系统调用来实现.在系统调用中,利用端口表中的信息访问网卡,将数据发送到网络.对消息的接收也可以在中断处理程序中完成.在这种形式的实现中由于需要系统调用和系统中断,会给通信性能带来一定的影响.对于如何定制网络接口硬件实现ULFM,我们将在第3节中加以讨论.

2.6 优缺点

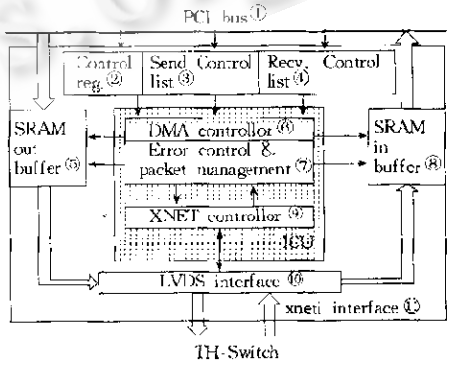
ULFM通信机制的关键在于对应用网络访问接口的抽象.由通信端口和发送队列组成的网络接口可以方便地支持用户层网络接口访问,实现零拷贝的消息传递以减小通信过程中的软件开销.实现用户层网络接口访问的难点在于多程序访问网络接口时的保护机制.在ULFM通信机制中,消息的接收由应用程序在各自申请的通信端口中进行,不会相互干涉.对于消息的发送,ULFM将其分为两个阶段进行.第1阶段,应用程序在各自的通信端口中准备消息数据及相关的信息,相互之间没有影响.第2阶段,应用程序将通信端口号推入发送队列,进行消息的发送.这种分阶段的结构将多程序访问网络接口时的保护问题集中在对发送队列的访问上,从而简化了保护机制的实现.对于发送队列的互斥访问可以采用软件锁的方法进行解决,也可以通过硬件实现发送队列的原子访问.后者将在第3节进一步加以讨论.

在ULFM通信端口中,消息缓冲区位于网络接口硬件可以直接访问的物理地址空间,应用程序通过存储映射后也可以对缓冲区进行访问,这样,消息数据就可以直接从缓冲区进出网络接口,实现零拷贝的数据传递.固定通信缓冲区的机制也可以避免传统通信结构中大量的缓冲区管理开销.

ULFM通信机制的不足在于,系统的物理内存资源毕竟是有限的,因此通信端口的数目和缓冲区的大小不能无限制地分配和使用.解决办法包括限制通信缓冲区的大小及通信端口的数目,必要时应用程序对通信端口进行复用.应用程序在通信结束后尽快对通信进行释放,以释放相关的缓冲区资源.

3 ULFM的实现

我们在TH-GBNet中通过定制的网络接口卡实现了ULFM用户层的快速消息传递机制.TH-GBNet[1]是以高档微机机群系统互连为目的而设计的互联网络.它能够为机群系统的高性能计算提供高带宽、低延迟的通信,同时具有良好的可扩展性.TH-GBNet主要由8端口动态交叉开关TH-Switch和连接节点与开关的网络接口卡xneti组成.



①PCI总线,②控制寄存器,③发送控制列表,④接收控制列表,⑤SRAM输出缓冲,⑥DMA控制器,⑦差错控制及分组管理,⑧SRAM输入缓冲,⑨XNET控制器,⑩LVDS接口,⑪xneti接口.

Fig.7 xneti interface
图7 xneti接口结构

定制的网络接口卡xneti对ULFM通信机制的实现提供了良好的支持.网卡的结构如图7所示.DMA引擎的嵌入,使xneti可以PCI bus master方式主动访问内存,完成消息数据从内存到网络接口之间的传输.对应于端口表,在xneti网卡的SRAM中维护了一个发送控制列表(SCT)和一个接收控制列表

(RCT),分别与 uport 通信端口中的发送部分和接收部分相对应.发送控制列表中的每一项包含一次消息发送的所有相关信息,包括消息缓冲区地址、长度、目的地址以及标志位等;接收控制列表中的每一项包含一次消息接收的相关信息,包括消息缓冲区地址、长度以及标志位等.发送队列也由硬件在控制寄存器中维护和实现,通过一个原子的内存操作进行访问.

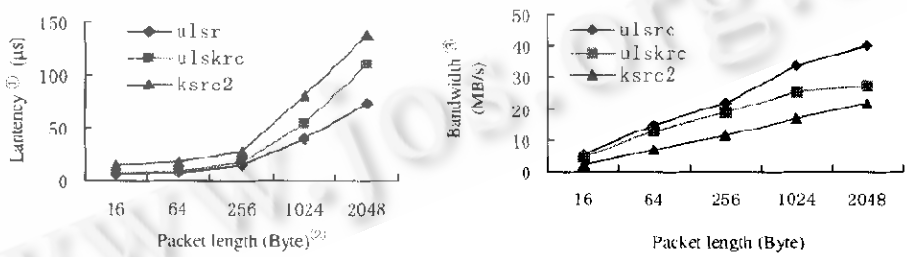
在 xneti 网络接口卡上,ULFM 通信机制的实现由于硬件的支持相对比较简单.主要包括利用发送控制列表和接收控制列表实现 uport 通信端口,提供用户层网络接口访问时的保护机制.由于对发送队列的访问是原子操作,保护机制的关键在于端口资源的建立、分配与管理.这项工作由设备驱动程序在操作系统核心统一进行.端口建立以后就归申请的应用程序所有,应用程序对该端口的操作不会对其他端口造成任何影响.端口使用结束后,由设备驱动程序进行相关资源的释放.网络接口硬件在进行数据进出网络接口的传输时也进行保护性检查,保证传输数据的长度不会超过缓冲区的边界.

4 性能评测

目前,我们已经完成了 TH-GBNet 中 Linux 操作系统环境下 ULFM 的开发.为了检测 ULFM 实际的通信性能,我们通过不同的应用程序进行了相关的测试.在试验环境中,Linux 的核心版本是 2.0.34,节点机的配置为 Pentium III 500MHz CPU,64M 内存,6.4GB IDE 接口的硬盘.整个机群系统由一个 TH-GBNet 交叉开关和 8 个节点机构成.

4.1 延迟与带宽

我们利用 Ping-Pong 基准测试程序,测量了 ULFM 通信机制下节点间应用进程之间的消息传递延迟和带宽.结果如图 8 和图 9 所示,其中 ulsr Ping-Pong 测试程序是完全在用户态进行的消息发送和接收.ulskrc 在目的节点进行的是核心态接收,有一次数据从系统核心缓冲区到应用进程空间的拷贝.ksrc2 在核心进行消息的发送和接收,在源和目的节点上都有一次核心缓冲区和应用空间之间的数据拷贝.



①延迟,②分组长度(字节),③带宽.

Fig. 8 Lantency and band curves for ULFM message passing

图 8 ULFM 中不同通信下的延迟与带宽

从实际的测量数据可以看出,ULFM 通信机制在用户层网络访问接口的方式中,16 字节消息数据的传递延迟仅为 6μs,对于 2K 的数据包,持续带宽可达40.26MB/s.从几种通信方式下的性能数据比较还可以看出,虽然各种方式完成的协议功能相同,但是,由于增加了用户态到核心态的切换和数据拷贝,ulskrc 方式下消息数据的传递延迟比 ulsr 有明显的增加.而完全在核心态进行发送和接收的 ksrc2 与完全在用户态进行发送和接收的 ulsr 相比,消息传递的延迟增加更大.以 16 字节的消息数据为例,前者是后者的两倍多.这也说明了采用用户层的网络访问接口,可以有效地降

低消息传递延迟,提高网络带宽的利用率.

4.2 并行应用程序

为了检测ULFM通信机制对实际机群并行计算程序的支持以及通行性能的改善对并行运行效率的影响,我们测量了并行求解π值(pi)、并行矩阵乘积(mm)、并行矩阵LU分解(lu)等并行计算程序利用ULFM通信机制在TH-GBNet上进行通信的运行时间.作为对比,我们测量了同样算法的并行应用程序在100M以太网上通过TCP/IP进行通信的运行时间.

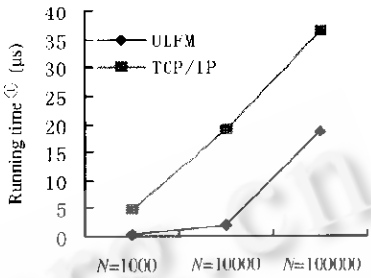


Fig. 9 Running time curve of pi

图9 pi运行时间图

pi:这是一个利用积分的方法求π的近似值的并行程序.当N取很大值时有

$$\pi=\int_0^1 f(x)\mathrm{d}x\approx\sum_{n=0}^{N-1} f\left(\frac{n+0.5}{N}\right)\times\frac{1}{N}=\sum_{i=0}^{M-1}\sum_{j=0}^{N/M-1} f\left(\frac{j\times M+i+0.5}{N}\right)\times\frac{1}{N}.$$

其中$f(x)=\frac{4}{1+x^2}$.为简单起见,我们假设M可以整除N.为了计算π的近似值,可以利用上式将整个计算量分给M个任务,同时在M台机器上进行运算.其中M台机器中的第i台机器运算$\sum_{j=0}^{N/M} f\left(\frac{j\times M+i+0.5}{N}\right)\times\frac{1}{N}$部分,所有$M$台主机上的子任务运行完自己的计算部分后将结果送到其中的一个任务,由该任务将M个结果求和,此即π的近似值.

mm:并行矩阵求积.假设要计算$N*N$规模的两个矩阵A与B之积C,并且利用网络上M台主机进行.首先由主任务在M台主机上派生出M个子任务.主任务在派生出M个子任务后将矩阵B广播到各子任务,然后依次将矩阵A的行向量每M个一组,向M个子任务各发送一个行向量,然后从子任务回收M个计算后所得结果的向量,直到矩阵A的行向量都发送完为止.各子任务在收到矩阵B后进入循环:从主任务接收一个行向量,与矩阵B相乘后得到一个行向量,然后将计算得到的行向量发送回主任务.主任务在收到所有计算结果后向各子任务发出终止信号,输出结果后退出.

lu:LU分解是求解线性方程组常用的算法.对于$Ax=b$的求解任务可以分为两个方面.一方面是并行计算矩阵A的LU分解,其中L,U分别是下三角阵和上三角阵,也即存在一个排列矩阵Q,使$AQ=LU$.另一方面是并行求解三角形方程组,即求解方程组$Ly=b$和$Ux=Y$.这里,我们只考虑将A进行LU分解,具体算法参见文献[5].

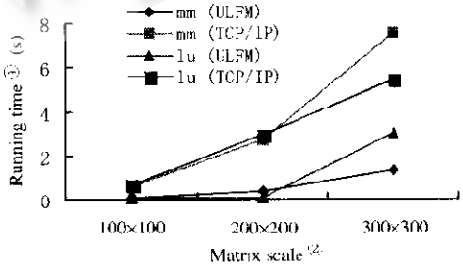


Fig. 10 Running time curve of mm and lu

图10 mm和lu运行时间图

3个并行应用程序的运行时间结果如图9和图10所示.可以看出,ULFM消息传递机制可以

为并行应用程序提供良好的通信支持.由于减小了通信过程中的软件开销,使应用程序的并行效率得到了很好的改善.与以太网的比较说明,要想提高机群系统的并行效率,通信性能至关重要.

5 相关工作及总结

对于通信中的软件开销问题,有许多项目进行了研究.U-Net[6]是在现有网络接口硬件的基础上,通过软件的设计提供用户层的网络访问接口来改进网络接口的通信性能.在快速以太网上,U-Net的消息传递延迟在30μs左右.ParaStation[7]在定制网络接口卡的基础上通过提供用户层网络访问接口,4字节的消息传递延迟在2.5μs左右,但是没有提供多程序访问的保护机制.FMP[8]在Myrinet的基础上通过采用内存映射技术、Pineline技术、CREDIT流控机制、网卡多线程技术来减小通信中的软件开销,单字节的消息传递延迟为11.2μs.在Shrimp[3,9,10]的VMMC网络接口上有一个网络接口页表(network interface page table),通过这个页表实现了节点间虚拟地址空间的内存页面映射,进程之间通过建立映射关系的页面进行通信.VMMC在Myrinet上的实现中,消息传递的延迟为9.8μs[10].

为了减小消息传递过程中的软件开销,ULFM通信机制通过设计以缓冲区为核心的应用程序网络访问接口,减小了缓冲区的管理开销,同时可以有效地实现用户层的网络接口访问和零拷贝的消息传递.精简的通信协议可以为应用程序提供可靠的、有序的节点间数据传输.该通信机制作为一个独立的通信结构,既可以在通用的网络接口卡上实现,也可以利用定制的网络接口加以支持.我们在TH-GBNet中的实现表明,ULFM通信机制可以有效地降低消息传递过程中的软件开销,提供低延迟、高带宽的通信性能.我们进一步的研究工作是在通用的网络和网络接口上实现ULFM通信机制,并进行相关性能的比较和评测.

References:

[1] Horie T. Improving AP1000 parallel computer performance with message communication. In: Proceedings of the 20th International Symposium of Computer Architecture. New York: ACM Press, 1993. 314~325.

[2] Boden, N. J. Myrinet: a gigabit-per-second local area network. IEEE Micro, 1995,15(1):21~28.

[3] Blumrich, M. A. Network interface for protected, user-level communication [Ph. D. Thesis]. Princeton University, 1996.

[4] Du, Yi, Li, San-li. Design and routing algorithm of the high-speed TH-switch in k-array n-cube interconnection network. Chinese Journal of Computers, 1998,21(10):873~880 (in Chinese).

[5] Hwang, Kai. Advanced Computer Architecture: Parallelism Scalability Programmability. McGraw-Hill, Inc., 1993.

[6] Von Eicken, T., Basu, A., Buch, V., *et al*. Vogels: U-Net: a user-level network interface for parallel and distributed computing. In: Proceedings of the 15th Symposium of Operating System Principles. 1995. 40~53.

[7] Warschko, T. M. The ParaStation project: using workstations as building blocks for parallel computing. In: Arabnia, H., ed. Proceedings of the Parallel & Distributed Processing Technique and Applications 1996, Vol I. 1996. 375~386.

[8] Shen, Jun. FMP: a fast messages passing for workstation clusters. Chinese Journal of Computers, 1998,21(7):595~602 (in Chinese).

[9] Dubnicki, C., Bilas, A., Li, Kai, *et al*. Design and implementation of virtual memory-mapped communication on myrinet. In: Proceedings of the 1997 Interational Parallel Processing Symposium. 1997. 388~396.

[10] Welsh, M., Basu, A., von Eicken, T. Low-Latency communication over fast ethernet. In: Proceedings of the Euro-Par'96. Springer Verlag, Lyon, France, 1996.

附中文参考文献:
[4] 杜毅,李三立.可扩展高速互连网络 TH-GBNet 的设计与实现.计算机学报,1998,21(10):873~880.
[8] 申俊.FMP:一种适用于机群系统的快速消息传递机制.计算机学报,1998,21(7):595~602.

A Fast Message Passing Mechanism at User-Level for Cluster Computing*

ZHOU Gui-lin, GE Yi, LI San-li, HUANG Zhen-chun, MA Qun-sheng

(*Department of Computer Science and Technology, Tsinghua University, Beijing* 100084, *China*)
E mail: huang@tirc.cs.tsinghua.edu.cn
Http://www.tsinghua.edu.cn

Abstract: The software overhead in IN (interconnection network) communication has become currently the bottleneck of cluster system performance. In this paper, a method is proposed to reduce the communication software overhead by providing a fast message passing mechanism at user level——ULFM (user-level fast message), that has been implemented in the IN TH-GBNet of cluster system. In ULFM, a new type of network access interface is provided to form a communication buffer centric application access to the cluster system IN. Several techniques, such as user level network access, zero copy message passing and simplified communication protocol, are integrated together for ULFM, that can effectively reduce the communication software overhead and achieve high communication performance with low latency and high bandwidth for application programs. Experimental results show that the latency at application level between different nodes for 16-byte message can be reduced to only 6μs, and the bandwidth can reach to 40MB/s for 2K byte length message.

Key words: workstation cluster; user level; communication protocol; message passing; overhead

* Received June 21, 1999; accepted February 18, 2000
Supported by the National Natural Science Foundation of China under Grant No. 69973024