高维环网上的一种可扩展的全交换算法

刘 刚, 顾乃杰, 任开新, 熊 焰
(中国科学技术大学计算机科学技术系, 安徽合肥 230027)

摘 要: 全交换在并行计算领域中有着大量而且重要的应用, 例如 FFT 和矩阵运算等. 本文提出了一种适合环网结构的全交换算法. 算法中采用了新的网络划分技术及通信模式, 使高维环网上全交换算法的通信量的主项达到了理论下限, 这是已知的其他相关算法未能达到的, 且其启动次数与通信量均优于现有的其他同类算法. 本文所述的算法并不要求环网每一维上的处理器结点数目是 2 的方幂或某一个数的平方. 最后, 该算法简单规范, 易于硬件高效实现.

关键词: 全交换; 全对全私人化通信; 并行算法; 环网; 虫蚀路由; 集体通信
中图分类号: TP311 文献标识码: A 文章编号: 0372-2112 (2005) 09-1723-06

Efficient and Scalable Algorithms for All-to-All Personalized Communication on Multidimensional Tori

LIU Gang, GU Nai-jie, REN Kai-xin, XIONG Yan
(Dept. of Computer Science & Technology, University of Science and Technology of China, Hefei, Anhui 230027, China)

Abstract: All-to-all personalized communication, also known as complete exchange, is at the heart of numerous applications in parallel computing environment, such as FFT and matrix transpose. This paper presents new efficient algorithms for all-to-all personalized communication in ring and multidimensional torus. In this paper, we adopt new network-partitioning technique and communication pattern to improve the performance of all-to-all personalized communication for multidimensional torus-connected multiprocessors, and the number of nodes in each dimension needs not be a power of two and square of some number. The main item of the transmission time of the algorithms reaches the lower bound in theory, which can not be obtained in existing indirect algorithms proposed in the literature. Finally, the algorithms are conceptually simple and easily implemented in hardware.

Key words: complete exchange; all-to-all personalized communication; parallel algorithm; Torus(Tori); wormhole routing; collective communication

1 引言

分布存储的并行处理机主要应用于科学计算、工程模拟和信号处理等以计算为主的领域, 并且已经成功地用于商业和网络应用中. 在分布存储的多计算机系统中, 存储器分布于每个处理器结点中, 系统通过在高速互联网络上传递消息来实现结点之间的通信, 但是处理器结点之间的通信开销一直是制约并行机性能提高的瓶颈之一. 因此, 必须通过改进通信算法和创新网络拓扑结构来减少通信延迟, 提高并行机性能. 目前分布存储的并行机广泛采用网孔或环网等具有很好扩展性的拓扑结构来满足其高带宽和低时延的通信需求.

MPI(Message Passing Interface) 是目前应用最广的一种基于消息传递模型的并行编程接口, 它几乎被所有并行计算环境和流行的多进程操作系统所支持. 基于 MPI 开发的应用程序具有最佳的可移植性和较好的可扩展性, 因此它是目前超大规模并行计算(1000 个处理器) 最可信赖的平台. MPI 不仅支持点对点通信, 还支持集体通信. 最重要的集体通信操作之一是 MPI_Alltoall, 即全交换. 所谓全交换是指系统中的每个处理器结点给其他每个结点均发送一份内容不同但长度相同的消息. 若系统有 N 个处理器结点, 每个结点 $P_i(1 \leqslant i \leqslant N)$ 开始有 N 条长度均为 m 的不同消息 $B[i,1], B[i,2], \cdots, B[i,N]$ 准备要发送到其他所有结点, 通信结束后, P_i 中的消息应变为 $B[1,i], B[2,i], \cdots, B[N,i]$. 全交换在许多计算问题中有大量而重要的应用, 如 FFT[1], 图论算法[2], 并行排序[3] 和矩阵算法[4] 等, 并且也用于评测相互连接的网络的性能.

但是包括 MPI_Alltoall 在内的所有 MPI 集体通信函数的具体实现均是基于点对点通信. 这种方式没有充分利用互联网络所提供的低延迟和高带宽等先进的特性. 因此我们必须优化和改进 MPI_Alltoall 的具体实现. 本文主要研究在环网结构上的全交换问题.

收稿日期: 2004-10-10; 修回日期: 2005-01-25
基金项目: 安徽省自然科学基金(No. 03042210)

目前,有关全交换操作的研究成果已有很多. 这些成果大致可分成直接通信算法(如 D. Scott[5], Y. C. Tseng[6] 和 C. C. Lam[7]) 和间接通信算法(又称消息合并算法,如 N. S. Surrder[8], Y. C. Tseng[9~11] 和 Y. J. Suh[12,13]) 两大类. 直接通信算法是指需要传递的消息仅在源结点与目的结点之间直接传送. 虽然其通信量达到了理论下限,但是其启动次数与通信量等同[5],因而其分析性能和实际性能均不理想,不过它适合于硬件实现以及消息特别长的情况. 间接通信算法采用消息合并方式,大大减少了启动次数,适合于消息长度较短或启动时间较长的系统,但是其通信量却难以完全达到理论下限. 对网孔与环网结构而言,间接通信算法比直接通信算法更有效. Tseng[11]提出了沿对角线传送的策略,所有的结点沿对角线分组以彻底消除链路冲突. Suh[12] 提出了在二维和三维环网上具有最小启动开销的全交换算法. Suh[13] 提出了有效的高维全交换算法,网络的大小不要求是 2 的方幂或是某一个数的平方.

本文先提出了一种高效的一维环上的间接全交换算法,其通信延迟不仅非常接近理论下限,而且其启动开销也很小. 然后我们运用网孔划分技术把一维环上的算法推广到二维环网上,并进一步提出了高效的四维环网上的全交换算法,其通信量非常接近理论下限 $N_1 \cdot \prod_{i=1}^{k} N_i \cdot m/8$. 我们还将算法拓展到三维环网、乃至 k 维环网上,使这些算法的通信延迟渐进最优,性能得到显著提高. 与现有的其他算法相比,本文中的算法有如下主要优点:

(1) 对于任何维数的环网结构,全交换算法的通信量的主项均达到了理论下限;

(2) 环网每一维上的处理器结点数目不要求是 2 的方幂或某一个数的平方;

(3) 该算法简单规范,易于实现.

2 通信模型与基本术语

本文所研究的可扩展的大型并行机中的计算结点是由具有双向通道的环网结构连接而成. 图 1 描绘了一个 4×4 二维环网及一个典型的计算结点的内部结构. 环网中的每一个结点由处理器,本地存储器,网络接口和路由器组成. 本文中我们假定每个处理器结点为单端口模式,即每个结点仅能在同一时间内发送和接受最多一个消息,且链路的带宽也仅为一个字节. 因为环网的拓扑结构很规则,所以我们提出的算法都采用了简单的维序选路.

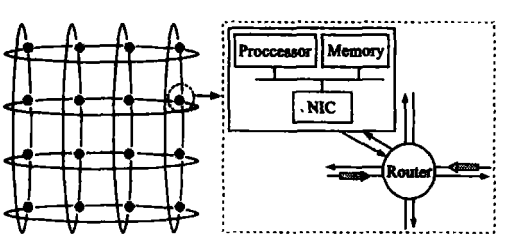

图 1 典型的 4×4 环网的系统结构

本文采用了著名的虫蚀路由技术. 若虫蚀路由的网络中没有链路冲突,则其中的通信延迟与消息传递的距离几乎无关. 我们设定发送一个长度为 m 的消息,所需的时间开销为 $T = t_s + m \cdot t_w$,其中 t_s 为发送消息所需的启动时间,它的系数我们称之为启动次数; t_w 为每字节的通信时间,它的系数称之为通信量; m 是消息长度(以字节表示). Tseng[11] 提出了全交换操作的通信量的理论下限.

引理 1: 对于一个 $N_1 \times N_2 \times \ldots \times N_k$ 的 k 维环网结构,其中 $N_1 \geqslant N_2 \geqslant \ldots \geqslant N_k$ 且 N_1 是偶数,全交换操作的通信量的理论下限为 $N_1 \cdot \prod_{i=1}^{k} N_i \cdot m/8$.

3 一维环上的全交换算法 AR1

对于由 p 个处理器结点构成的具有双向链路的一维环结构,若我们将一条双向链路视为两条单向通道,则一维环可以被视为两条单向的环形链路. 我们用 PE_i 来表示一维环结构上的处理器结点,其中 $0 \leqslant i \leqslant p-1$ 且 p 为偶数. 标号为奇数的结点(奇结点)与一条单向环形链路构成了一条单向的环结构,记作奇环;标号为偶数的结点(偶结点)与另一条单向环形链路也构成了一条单向的环结构,记作偶环. 为了最大程度的减少通信量,并且显著的降低启动次数,同时还要满足处理器结点的单端口限制,我们提出了一种新型的无阻塞全交换算法.

首先,奇结点和与之左相邻的偶结点互相传递数据. 奇结点把目的地址为其右半圈结点的消息传送给偶结点,同时偶结点也把目的地址为其左半圈结点的消息传送给奇结点. 然后,奇环上的结点向左循环传递消息,同时偶环上的结点向右循环传递消息. 这两个传递过程都仅需循环传递半圈. 最后一步是在奇结点和与之右相邻的偶结点之间进行,双方均把目的地址为对方的消息回传给对方. 图 2 是算法的正式描述: M_d^s 表示从源点 PE_s 发送到终点 PE_d 的消息, $M_{i,\ldots,j}^S$ 表示消息集合 $\{M_i^S, M_{i+1}^S, \ldots, M_j^S\}$, 而 $M_d^{i,\ldots,j}$ 表示消息集合 $\{M_d^i, M_d^{i+1}, \ldots, M_d^j\}$. 若 $i<0$ 或 $i \geqslant p$, 则用 $(i \pm p) \bmod p$ 来表示 i.

```
Algorithm AR1: /* 一维环上的全交换算法* /
BEGIN
For i= 0 To p/2- 1 Para_ Do
  PE_2i sends M^{2i}_{2i+ p/2, ..., 2i+ p- 1}  to  PE_{2i- 1};
  PE_2i- 1 sends M^{2i- 1}_{2i, ..., 2i+ p/2- 1}  to  PE_2i;
Endfor
For k= 0 To ⌈p/4⌉- 2 Do
For i= 0 To p/2- 1 Para_ Do
  PE_2i sends M^{2i- 2k, (2i- 1)- 2k}_{2i+ 2, ..., 2i+ (p/2- 1)- 2k}  to  PE_2i+ 2;
  PE_2i+ 1 sends M^{(2i+ 1)+ 2k, (2i+ 2)+ 2k}_{2i- (p/2- 2)+ 2k, ..., 2i- 1}  to  PE_2i- 1;
  Endfor
Endfor
For i= 0 To p/2- 1 Para_ Do
  PE_2i sends M^{2i- (2·⌊p/4⌋- 1), ..., 2i}_{2i+ 1}  To  PE_2i+ 1;
  PE_2i+ 1 sends M^{2i+ 1, ..., 2i+ 2·⌊p/4⌋}_{2i}  To  PE_2i;
Endfor
END
```

图 2 一维环上的全交换算法的正式描述

图 3 描绘出了算法 AR1 在有 12 个处理器结点的一维环上的运行情况. 算法共分 4 步, 其中奇环和偶环都分别沿相反的方向循环传送了两次. 每次循环传送消息数据时, 所传送的消息数目逐次递减.

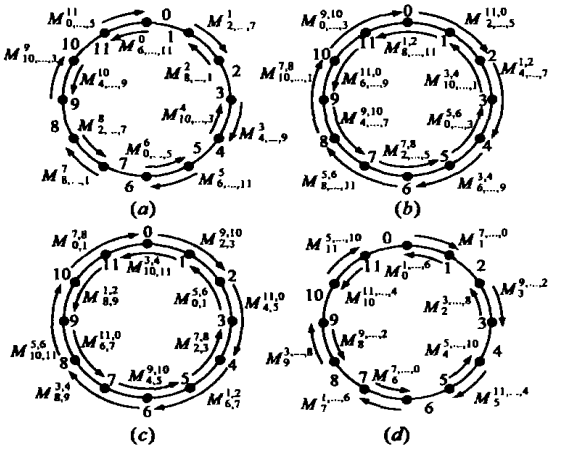

图 3　在具有 12 个处理器结点的一维环上全交换

下面对算法 AR1 的通信时间复杂度进行分析.

第 1 步互换消息的时间代价是 $t_s+ p/2\cdot mt_w$; 该算法的第 2 步是奇环和偶环的 $\lceil p/4\rceil-1$ 次循环传递消息. 当 $p/2$ 是偶数时, 第 2 步需要 $(p/4-1)\cdot t_s+ 2m\cdot\sum_{k=1}^{p/4-1}2k\cdot t_w$; 当 $p/2$ 是奇数时, 第 2 步需要 $(p-2)/4\cdot t_s+ 2m\cdot\sum_{k=1}^{(p-2)/4}(2k-1)\cdot t_w$; 第 3 步的时间代价是 $t_s+ 2\cdot\lfloor p/4\rfloor\cdot mt_w$. 因此, 整个算法的通信时间复杂度为:

$$T_{\text{ring}}=(\lceil p/4\rceil+1)\cdot t_s+(\lfloor p^2/8\rfloor+p/2)\cdot mt_w$$

其中通信量的主项为 $\lfloor p^2/8\rfloor\cdot m$, 由引理 1 可知, 已经达到了理论下限.

4　二维环网上的全交换算法 AT2

二维双向环网的每一行、每一列均可被视为一维环结构, 因此一维环结构上的全交换算法可以被应用于构造二维环网上的全交换算法. 我们对 $r\times s(r\leqslant s)$ 二维环网中的所有处理器结点进行编号, 第 i 行、第 j 列上的处理器结点被标记为 $p(x, y)$, 其中 $0\leqslant x\leqslant r-1$ 且 $0\leqslant y\leqslant s-1$, 它与 $p(x-1, y)$、$p(x+1, y)$、$p(x, y-1)$ 和 $p(x, y+1)$ 结点相连接.

我们运用 Gu[14] 提出的二维环网划分技术来实现二维环网上的全交换. 环网中满足 $(x+y)\bmod 2=0$ 的结点称为"偶结点", 满足 $(x+y)\bmod 2=1$ 的结点称为"奇结点". 首先, 整个二维环网被划分为 $r/2\times s/2$ 个 2×2 的子网孔, 每个子网孔均由两个偶结点与两个奇结点构成. 环网的每一行都间隔排列着 $s/2$ 个偶结点和 $s/2$ 个奇结点, 每一列都间隔排列着 $r/2$ 个奇结点和 $r/2$ 个偶结点. 同一行中的 $s/2$ 个偶结点可以构成一条逻辑环, 同一列中的 $r/2$ 个奇结点也可以构成一条逻辑环. 此全交换算法在通信过程中没有任何链路冲突.

我们现在分析二维环网上的全交换算法的通信时间复杂度:

二维环网上的全交换算法分为两个阶段: 第一阶段可分为两个步骤. 第一步是, 同一行中的 $s/2$ 个偶结点构成了一条逻辑环, 它们执行一维环上的全交换算法 AR1, 易知启动次数为 $\lceil s/8\rceil+1$, 通信量为 $(\lfloor s^2/32\rfloor+s/4)\cdot 2r\cdot m$; 与此同时, 同一列中 $r/2$ 的个奇结点也构成了一条逻辑环, 它们执行一维环上的全交换算法 AR1, 易知启动次数为 $\lceil r/8\rceil+1$, 通信量为 $(\lfloor r^2/32\rfloor+r/4)\cdot 2s\cdot m$, 但是由于 $r\leqslant s$, 所以第一步的通信开销为 $(\lceil s/8\rceil+1)\cdot t_s+(\lfloor s^2/32\rfloor+s/4)\cdot 2r\cdot mt_w$. 第二步是同一行中的 $s/2$ 个奇结点执行一维环上的全交换算法 AR1, 同一列中 $r/2$ 的个偶结点执行一维环上的全交换算法 AR1, 因此第二步的通信开销与第一步相等. 算法的第二阶段是在 2×2 子网孔中实现全交换. 第二阶段也分为两步, 每步均是相邻的结点之间互换一半的消息数据, 所以第二阶段中每一步的通信开销均为 $t_s+(r\cdot s/2)\cdot mt_w$. 故在 $r\times s$ 二维环网上的全交换算法的时间复杂度为:

$$T_{2D}=(2\cdot\lceil s/8\rceil+4)\cdot t_s+(4r\cdot\lfloor s^2/32\rfloor+2r\cdot s)\cdot mt_w.$$

由上可知, 算法的通信量的主项达到了理论下限.

5　高维环网结构上的全交换算法

5.1　四维算法 PT4

如果二维环网可以看成是两个一维环的"迪卡尔乘积", 那么四维环网也能看成是两个二维环网的"迪卡尔乘积". 对于 $N_1\times N_2\times N_3\times N_4$ 的四维双向环网, 其中 $N_1\geqslant N_2\geqslant N_3\geqslant N_4$ 且 N_1, N_2, N_3 和 N_4 均是 8 的倍数, 每一个处理器结点被标记为 $p(x, y, s, t)$, $(0\leqslant x\leqslant N_1-1,\ 0\leqslant y\leqslant N_2-1,\ 0\leqslant s\leqslant N_3-1$ 和 $0\leqslant t\leqslant N_4-1)$. X 维与 Y 维构成了 X-Y 平面, S 维与 T 维构成了 S-T 平面. 满足 $(x+y+s+t)\bmod 2=0$ 的结点为偶结点, 满足 $(x+y+s+t)\bmod 2=1$ 的结点为奇结点. 算法第一阶段的通信模式为: 第一步, 所有的偶结点都执行 X-Y 平面上的二维全交换算法, 同时所有的奇结点都执行 S-T 平面上的二维全交换算法; 第二步, 所有的偶结点都执行 S-T 平面上的二维全交换算法, 同时所有的奇结点都执行 X-Y 平面上的二维全交换算法. 算法的第二阶段为, 在二元 4 立方体中实现全交换, 如图 4 所示, 每个结点将目的地址为对方的消息传送给对方结点.

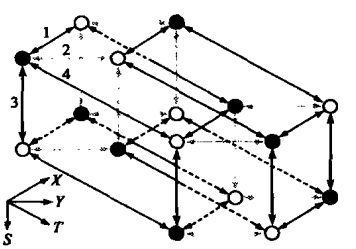

图 4　四维环网上全交换算法第二阶段通信模式

下面分析四维环网上的全交换算法的通信时间复杂度: 该算法分为两个阶段, 第一阶段又分为两个步骤. 第一步, X-Y 平面上的二维全交换算法的通信时间开销为 $(2\cdot\lceil N_1/16\rceil+4)\cdot t_s+(8\cdot\lfloor N_1^2/128\rfloor+2N_1)\cdot N_2N_3N_4\cdot mt_w$, S-T 平面上的二维全交换算法的通信时间开销为 $(2\cdot\lceil N_3/16\rceil+4)\cdot t_s+(8\cdot\lfloor N_3^2/128\rfloor+2N_3)\cdot N_1N_2N_4\cdot mt_w$, 但是由于 $N_1\geqslant N_3$, 前者大于后者, 所以通信时间开销以前者为主. 第一阶段第二步的通信时间开销与第一步相等. 第二阶段分为四步, 每步均是相邻的结点之间互换一半的消息数据, 所以每一步的时间开销为 t_s+

$(N_1N_2N_3N_4/2)\cdot mt_w$. 故四维环网上的全交换算法的时间复杂度为:

$$T_{4D}=(4\cdot\lceil N_1/16\rceil+12)\cdot t_s+(16\cdot\lfloor N_1^2/128\rfloor+6N_1)\cdot N_2N_3N_4\cdot mt_w.$$

由上式可知, 通信量的主项达到了理论下限.

二维环网上的全交换算法可推广到四维环网, 四维环网上的算法也可以推广到八维环网, 依此类推, 我们可以得到 k 维环网上的全交换算法, 其中 $k=2^d$.

对于 $N_1\times N_2\times\cdots\times N_k$ 的 k 维环网, 其中 $N_i(1\leqslant i\leqslant k)$ 必须是 $2k$ 的倍数且 $N_1\geqslant N_2\geqslant\cdots\geqslant N_k$, 它可以看作是两个 $k/2$ 维环网的"迪卡尔乘积". 该算法共分二个阶段, 第一阶段第一步是,"偶结点"在 $N_1,\cdots,N_{k/2}$ 维上执行 $k/2$ 维环网上的全交换算法, 同时"奇结点"在 $N_{k/2+1},\cdots,N_k$ 维上执行 $k/2$ 维环网上的全交换算法; 第二步是"偶结点"与"奇结点"交换执行第一步中的任务. 第二阶段是在二元 k 立方体中实现消息的完全传递. 我们利用二元 k 立方体中的全交换算法来实现 2^k 个结点的完全通信, 其时间复杂度为 $k\cdot t_s+k\cdot 2^{k-1}\cdot mt_w$. 通过归纳可知, 总的通信时间开销为:

$$T_{kD}=k\cdot(\lceil N_1/4k\rceil+\log_2 k+1)\cdot t_s+\{k^2\cdot\lfloor N_1^2/8k^2\rfloor+(\lfloor k^2/4\rfloor+k/2)\cdot N_1\}\cdot\prod_{i=2}^{k}N_i\cdot mt_w$$

当 $N_1\gg k$ 时, 高维环网上的全交换算法是非常有效的, 其通信量已经非常接近理论下限 $N_1\cdot\prod_{i=1}^{k}N_i\cdot m/8$.

5.2 修改后的算法 MT3

5.1 中的高维全交换算法仅能在维数为 2 的幂的环网上运行, 而对于其他维数的环网而言, 例如著名的 Cray T3D, 此算法则不能运行. 于是我们对上面的算法加以改进, 虽然改进后的算法的启动次数略有增加, 但是其通信延迟却保持不变.

首先我们描述三维环网上的全交换算法, 然后再把此算法推广到 k 维环网结构上. 对于 $N_1\times N_2\times N_3$ 的三维环网, 其中 $N_1\geqslant N_2\geqslant N_3$ 且 N_1, N_2 和 N_3 均需是 6 的倍数, 每一个处理器结点被标记为 $p(x,y,z)(0\leqslant x\leqslant N_1-1, 0\leqslant y\leqslant N_2-1$ 和 $0\leqslant z\leqslant N_3-1)$. 为了最大限度地利用通信带宽以及完全避免链路冲突, 我们将三维环网上的结点分为 3 组, 即 $G_i=\{p(x,y,x)\mid(x+y+z)\bmod 3=i\}$, 其中 $0\leqslant i\leqslant 2$. 这样整个三维环网就被划分为 27 个 $N_1/3\times N_2/3\times N_3/3$ 的三维逻辑环网, 且沿着每条轴上的结点隶属于不同的三个三维逻辑环网. 因为处理器结点有单端口的限制, 所以同属于一组的处理器结点以某种顺序轮流沿着 x 轴、y 轴和 z 轴执行一维环上的全交换算法. 在这一阶段中, 同属于一个 $N_1/3\times N_2/3\times N_3/3$ 三维逻辑环网中的结点之间相互传送消息. 算法最后需要在 $3\times 3\times 3$ 的三维子网孔上实现消息的完全互换. 下面是通信模式的具体描述:

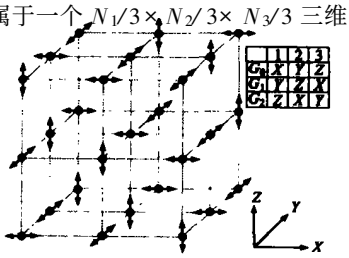

图 5 在三维环网上的全交换的第一阶段通信模式的描述

全交换算法的第一阶段分为三步, 如图 5 所示, 同隶属于一个组的处理器结点可以按照图 5 中所列的顺序轮流沿着 x 轴、y 轴和 z 轴执行一维环上的全交换算法, 即 G_0 中所有结点依次沿着 x 轴、y 轴和 z 轴执行一维环上的全交换操作, G_1 中所有结点依次沿着 y 轴、z 轴和 x 轴执行一维环上的全交换操作, G_2 中所有结点依次沿着 z 轴、x 轴和 y 轴执行一维环上的全交换操作.

{Step1 of Phase1}:

If $(x+y+z)\bmod 3=0$, then $p(x,y,z)$ 与 $p((x+3)\bmod N_1,y,z),\cdots,p((x+N_1-3)\bmod N_1,y,z)$ 形成了在 X 维上的一维环结构, 执行一维环上的全交换算法 AR1;

If $(x+y+z)\bmod 3=1$, then $p(x,y,z)$ 与 $p(x,(y+3)\bmod N_2,z),\cdots,p(x,(y+N_2-3)\bmod N_2,z)$ 形成了在 Y 维上的一维环结构, 执行一维环上的全交换算法 AR1;

If $(x+y+z)\bmod 3=2$, then $p(x,y,z)$ 与 $p(x,y,(z+3)\bmod N_3),\cdots,p(x,y,(z+N_3-3)\bmod N_3)$ 形成了在 Z 维上的一维环结构, 执行一维环上的全交换算法 AR1;

{Step2 of Phase1}:

If $(x+y+z)\bmod 3=0$, then $p(x,y,z)$ 与 $p(x,(y+3)\bmod N_2,z),\cdots,p(x,(y+N_2-3)\bmod N_2,z)$ 形成了在 Y 维上的一维环结构, 执行一维环上的全交换算法 AR1;

If $(x+y+z)\bmod 3=1$, then $p(x,y,z)$ 与 $p(x,y,(z+3)\bmod N_3),\cdots,p(x,y,(z+N_3-3)\bmod N_3)$ 形成了在 Z 维上的一维环结构, 执行一维环上的全交换算法 AR1;

If $(x+y+z)\bmod 3=2$, then $p(x,y,z)$ 与 $p((x+3)\bmod N_1,y,z),\cdots,p((x+N_1-3)\bmod N_1,y,z)$ 形成了在 X 维上的一维环结构, 执行一维环上的全交换算法 AR1;

{Step3 of Phase1}:

If $(x+y+z)\bmod 3=0$, then $p(x,y,z)$ 与 $p(z,y,(z+3)\bmod N_3),\cdots,p(x,y,(z+N_3-3)\bmod N_3)$ 形成了在 Z 维上的一维环结构, 执行一维环上的全交换算法 AR1;

If $(x+y+z)\bmod 3=1$, then $p(x,y,z)$ 与 $p((x+3)\bmod N_1,y,z),\cdots,p((x+N_1-3)\bmod N_1,y,z)$ 形成了在 X 维上的一维环结构, 执行一维环上的全交换算法 AR1;

If $(x+y+z)\bmod 3=2$, then $p(x,y,z)$ 与 $p(x,(y+3)\bmod N_2,z),\cdots,p(x,(y+N_2-3)\bmod N_2,z)$ 形成了在 Y 维上的一维环结构, 执行一维环上的全交换算法 AR1;

显而易见, 在算法的第一阶段中, 所有的链路均忙于传递消息, 而且负载均衡. 第一阶段之后, 根据消息的传递情况, 三维环网被划分为 $N_1/3\times N_2/3\times N_3/3$ 个 $3\times 3\times 3$ 的三维子网孔. 第二个阶段就是在 $3\times 3\times 3$ 的三维子网孔中实现消息的全交换, 如图 6 所示, 三维子网孔中的每个处理器结点可以依次沿着 x 轴、y 轴和 z 轴与其他两个相邻结点传递消息. 在具有 3 个结点的一维阵列上, 每个结点可以通过顺时针和逆时针两次循环来实现与左右相邻结点的全交换. 因此, 第二阶段的全交换操作就在以下的六个步骤中完成.

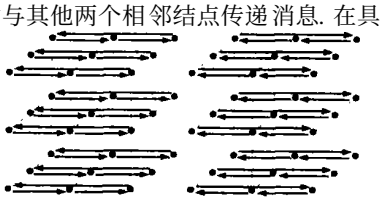

图 6 在 3×3×3 三维子网孔中沿 x 轴互换消息数据

{Step1 of Phase2}:
If $x \bmod 3 = 0$, then $p(x,y,z) \to p(x+1,y,z)$;
If $x \bmod 3 = 1$, then $p(x,y,z) \to p(x+1,y,z)$;
If $x \bmod 3 = 2$, then $p(x,y,z) \to p(x-2,y,z)$;
{Step2 of Phase2}:
If $x \bmod 3 = 0$, then $p(x,y,z) \to p(x+2,y,z)$;
If $x \bmod 3 = 1$, then $p(x,y,z) \to p(x-1,y,z)$;
If $x \bmod 3 = 2$, then $p(x,y,z) \to p(x-1,y,z)$;
{Step3 of Phase2}:
If $y \bmod 3 = 0$, then $p(x,y,z) \to p(x,y+1,z)$;
If $y \bmod 3 = 1$, then $p(x,y,z) \to p(x,y+1,z)$;
If $y \bmod 3 = 2$, then $p(x,y,z) \to p(x,y-2,z)$;
{Step4 of Phase2}:
If $y \bmod 3 = 0$, then $p(x,y,z) \to p(x,y+2,z)$;
If $y \bmod 3 = 1$, then $p(x,y,z) \to p(x,y-1,z)$;
If $y \bmod 3 = 2$, then $p(x,y,z) \to p(x,y-1,z)$;
{Step5 of Phase2}:
If $z \bmod 3 = 0$, then $p(x,y,z) \to p(x,y,z+1)$;
If $z \bmod 3 = 1$, then $p(x,y,z) \to p(x,y,z+1)$;
If $z \bmod 3 = 2$, then $p(x,y,z) \to p(x,y,z-2)$;
{Step6 of Phase2}:
If $z \bmod 3 = 0$, then $p(x,y,z) \to p(x,y,z+2)$;
If $z \bmod 3 = 1$, then $p(x,y,z) \to p(x,y,z-1)$;
If $z \bmod 3 = 2$, then $p(x,y,z) \to p(x,y,z-1)$;

整个通信过程中没有任何链路冲突.

三维环网上的全交换算法的时间复杂度分析如下: 算法总共有两个阶段, 第一阶段有三步, 每步的通信时间开销为 $\{\lceil N_1/12\rceil + 1\} \cdot t_s + (3 \cdot \lfloor N_1^2/72\rfloor \cdot N_2N_3 + N_1N_2N_3/2) \cdot mt_w$; 第二阶段有六步, 每步的通信时间开销为 $t_s + (N_1N_2N_3/3) \cdot mt_w$). 所以总的通信时间复杂度是:

$$MT_{3D} = (3 \cdot \lceil N_1/12\rceil + 9) \cdot t_s + (9 \cdot \lfloor N_1^2/72\rfloor \cdot N_2N_3 + 7N_1N_2N_3/2) \cdot mt_w$$

其中通信量的主项达到了理论下限.

5.3 k 维环网上的全交换算法 MTk

下面我们把三维环网上的全交换算法推广到高维环网. 对于 $N_1 \times N_2 \times \cdots \times N_k$ 的 k 维环网, 其中 $N_i (1 \leqslant i \leqslant k)$ 是 $2k$ 的倍数且 $N_1 \geqslant N_2 \geqslant \cdots \geqslant N_k$. 全交换算法共有两个阶段, 第一阶段由 k 步组成, 同隶属于一个组 G_i 的处理器结点按照某种顺序轮流沿着 k 条轴执行一维环上的全交换算法. 第二阶段分为 k 步, 在 $\underbrace{k \times \cdots \times k}_{k}$ 网孔中实现消息的完全传送, 每一步均是网孔中的结点沿着某一条轴线在一维阵列上实现消息的完全回送. 我们利用一维阵列上的全交换算法[14]来实现 k 个结点的完全通信, 该算法的时间复杂度为 $(p-1) \cdot t_s + \lfloor p^2/4\rfloor \cdot mt_w$, 其中 p 为一维阵列上的处理器结点个数.

该算法共分两个阶段. 第一阶段有 k 步, 每步均需时 $(\lceil N_1/4k\rceil + 1) \cdot t_s + (\lfloor N_1^2/8k^2\rfloor \cdot k \cdot \prod_{i=2}^{k} N_i + \prod_{i=1}^{k} N_i/2) \cdot mt_w$; 第二阶段也有 k 步, 每步均需要 $(k-1) \cdot t_s + (\lfloor k^2/4\rfloor / k) \cdot \prod_{i=1}^{k} N_i \cdot mt_w$; 所以总的通信时间复杂度是:

$$MT_{kD} = (k \cdot \lceil N_1/4k\rceil + k^2) \cdot t_s + \{k^2 \cdot \lfloor N_1^2/8k^2\rfloor + (\lfloor k^2/4\rfloor + k/2) \cdot N_1\} \cdot \prod_{i=2}^{k} N_i \cdot mt_w.$$

当 $N_1 \gg k$ 时, 高维环网上的全交换算法是非常有效的, 其通信量已经非常接近理论下限 $\frac{N_1}{8} \cdot \prod_{i=1}^{k} N_i \cdot m$.

6　性能分析和比较

现在对本文提出的算法的通信时间复杂度和已有的相关算法作对比, 并且对其性能进行详细的分析. 表 1 列出了目前已知的一些高性能的全交换算法的时间复杂度. Tseng[9~11]与 Suh[12]提出的二维环网上的全交换算法均要求环网每一维上的处理器结点数目是 2 的方幂或某一个数的平方, 而本文提出的所有算法与 Suh[13]的算法都没有此项限制. 若将本文提出的算法应用到 $2^d \times 2^d$ 的二维环网上, 则该算法的时间复杂度为 $T_{2D} = (2^{d-2} + 4) \cdot t_s + (2^{3d-3} + 2^{2d+1}) \cdot mt_w$. 在启动次数和通信量这两个重要指标上, 本文提出的二维算法 AT2 均大大优于 Tseng[9,11]的算法和 Suh[13]的算法. 虽然 Tseng[10]与 Suh[12]中的算法的启动次数均为 $O(d)$, 但是其通信量很大, 整体通信性能不如本文算法. 本文所述算法的通信量开销是所有已知单端口间接通信算法中最小的, 其主项均达到了通信量的理论下限.

表 1　环网上的全交换算法的性能对比

Topology	Algorithm	Startup Cost	Message Transmission Cost
$r \times s$ 2D Torus	Suh[13]	$(s/2+2) \cdot t_s$	$(r \cdot s^2/4 + r \cdot s) \cdot mt_w$
	AT2	$(2 \cdot \lceil s/8\rceil + 4) \cdot t_s$	$(4r \cdot \lfloor s^2/32\rfloor + 2r \cdot s) \cdot mt_w$
$2^d \times 2^d$ 2D Torus	Suh[12]	$(3d-3) \cdot t_s$	$\{9 \cdot 2^{3d-4} + (d^2 - 5d + 3) \cdot 2^{2d-1}\} \cdot mt_w$
$N_1 \times \cdots \times N_k$ kD Torus	Suh[13]	$k \cdot (N_1/4+1) \cdot t_s$	$(k \cdot N_1/8 + k/2) \cdot \prod_{i=1}^{k} N_i \cdot mt_w$
	PTk $k = 2^d$	$k \cdot (\lceil N_1/4k\rceil + \log_2 k + 1) \cdot t_s$	$\{k^2 \cdot \lfloor N_1^2/8k^2\rfloor + (\lfloor k^2/4\rfloor + k/2) \cdot N_1\} \cdot \prod_{i=2}^{k} N_i \cdot mt_w$
	MTk	$(k \cdot \lceil N_1/4k\rceil + k^2) \cdot t_s$	$\{k^2 \cdot \lfloor N_1^2/8k^2\rfloor + (\lfloor k^2/4\rfloor + k/2) \cdot N_1\} \cdot \prod_{i=1}^{k} N_i \cdot mt_w$

Suh[13]提出的高维环网上的全交换算法是目前我们仅知道的高维间接通信算法, 此算法不要求环网每一维上的处理器结点数目是 2 的方幂或是某一个数的平方. 本文提出的高维算法 MTk 不仅具备此特点, 而且在启动次数和通信量这两个重要指标上均大大优于 Suh[13]的算法, 当 $N_1 \gg k$ 时, 本文提出的高维全交换算法的通信量接近理论下限 $N_1 \cdot \prod_{i=1}^{k} N_i \cdot m/8$.

理想的情况是在超级并行机上运行该程序来进行性能评测, 然而现有的机器规模难以支持此类算法的可扩展性分析, 并且无法保证获取访问的计算结点由规则的网络拓扑结构连接. 因此, 大多数此类文章均是采用基于商用并行机的试验参数构建成的模型来分析. 本文采用基于 Intel Paragon 的参数构建的模型[15]来分析算法性能, 其中 $t_s = 75\mu s$, $t_w = 0.011\mu s$. 图

7显示了本文提出的二维全交换算法和已有的算法[12, 13]在16×16、32×32、64×64和128×128二维环网上的性能比较结果. 从图7中可以看到, 无论消息的长度是长还是短, 无论网络的规模是大还是小, 本文中算法的性能均优于已有的著名算法.

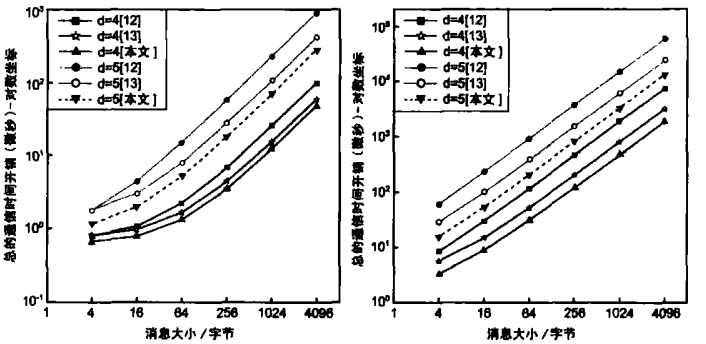

图7 在16×16、32×32、64×64和128×128二维环网上, 算法的性能评估

7 结论

本文先提出了一维环上的一种高效的全交换算法, 然后把该算法扩展到二维、三维和高维环网上. 本文采用新的网络划分技术来充分利用网络通信带宽, 消除链路冲突以及保证单端口模式, 并且采用消息合并方式来减少启动次数开销, 从而提高了算法性能. 本文所提出的通信模式使网络中所有结点的负载基本平衡, 从而能使全交换算法的通信量开销能非常接近理论下限, 算法的性能得到显著提高. 一维环上的全交换算法的时间复杂度为: $T_{ring}=(\lceil p/4\rceil+1)\cdot t_s+(\lfloor p^2/8\rfloor+p/2)\cdot mt_w$, 其通信量的主项达到了理论下限. $r\times s$ 二维环网上的全交换算法的时间复杂度为: $T_{2D}=(2\cdot\lceil s/8\rceil+4)\cdot t_s+(4r\cdot\lfloor s^2/32\rfloor+2r\cdot s)\cdot mt_w$, 其通信量的主项也达到了理论下限. 修改后的维环网上的全交换算法的通信开销为: $(k\cdot\lceil N_1/4k\rceil+k^2)\cdot t_s+(k^2\cdot\lfloor N_1^2/8k^2\rfloor\cdot\prod_{i=2}^{k}N_i+(\lfloor k^2/4\rfloor+k/2)\cdot\prod_{i=1}^{k}N_i)\cdot mt_w$. 当 $N_1\gg k$ 时, 算法的通信量非常接近理论下限 $N_1\cdot\prod_{i=1}^{k}N_i\cdot m/8$, 这在其他为数不多的高维算法中是从来没有过的.

虽然本文提出的算法要求每一维上的结点数目是某一个数的倍数, 但是通过添加虚拟结点的方式, 该算法也能适用于每一维上的结点数目为任意值的情况. 本文提出的算法简单规范, 易于硬件高效实现.

参考文献:

[1] 陈国良, 等. 并行FFT算法在3种并行计算模型上的设计和分析[J]. 软件学报, 1996, (7)(增刊): 57-63.

[2] 陈国良, 等. 并行图论算法研究进展[J]. 计算机研究与发展, 1995, 32(9): 1-16.

[3] 顾乃杰, 等. 并行双调排序算法的有效实现及性能分析[J]. 计算机研究与发展, 2002, 39(10): 1343-1348.

[4] 谢幸, 顾乃杰, 陈国良. 曙光1000上矩阵乘积算法的性能分析[J]. 计算机研究与发展, 1999, 36(7): 848-852.

[5] D S Scott. Efficient all-to-all communication patterns in hypercube and mesh topologies[A]. Proc Sixth Conf Distributed Memory Concurrent Computers[C]. Portland, 1991. 398-403.

[6] Y C Tseng, S K S Gupta. All-to-all personalized communication in a wormhole-routed torus[J]. IEEE Trans. Parallel and Distributed Systems, 1996, 7(5): 498-505.

[7] C C Lam, C H Huang, et al. Optimal algorithms for all-to-all personalized communication on rings and two dimensional tori[J]. Journal of Parallel and Distributed Computing, 1997, 43: 3-13.

[8] N S Sundar, D N Jayasimha, et al. Complete exchange in 2D meshes[A]. Proc. of 1994 Scalable High Performance Computing Conf[C]. Knoxville, 1994. 406-413.

[9] Y C Tseng, S Gupta, et al. An efficient scheme for complete exchange in 2D tori[A]. Proc. Int'l Parallel Processing Symp[C]. Santa Barbara, 1995. 532-536.

[10] Y C Tseng, S Y Ni, et al. Toward optimal complete exchange on wormhole-routed tori[J]. IEEE Trans. Computers, 1999, 48(10): 1065-1082.

[11] Y C Tseng, T H Lin, et al. Bandwidth-optimal complete exchange on wormhole-routed 2D/3D torus networks: A diagonal-propagation approach[J]. IEEE Trans. Parallel and Distributed Systems, 1997, 8(4): 380-396.

[12] Y J Suh, S Yalamanchili. All-to-all communication with minimum start-up costs in 2D/3D tori and meshes[J]. IEEE Trans. Parallel and Distributed Systems, 1998, 9(5): 442-458.

[13] Y J Suh, K G Shin. All-to-all personalized communication in multidimensional torus and mesh networks[J]. IEEE Trans. Parallel and Distributed Systems, 2001, 12(1): 38-59.

[14] Gu Naijie. Efficient indirect all-to-all personalized communication on rings and 2-D tori[J]. Journal of Computer Science and Technology, 2001, 16(5): 480-483.

[15] S H Bokhari. Multiphase complete exchange: a theoretical analysis[J]. IEEE Trans. on Computers, 1996, 45(2): 220-229.

作者简介:

刘 刚 男, 1978年3月出生于甘肃省兰州市, 2001年获中国科学技术大学工学学士学位, 现为中国科学技术大学计算机科学技术系博士研究生, 目前主要研究方向为并行分布式计算中的通信问题, 并行体系结构.
E-mail: liugang@mail.ustc.edu.cn.

顾乃杰 男, 1961年8月出生于江苏省南通市, 1983年于中国科学技术大学数学系计算数学专业本科毕业, 获理学学士学位; 1989年于中国科学技术大学计算机系计算机软件专业研究生毕业, 获工学硕士学位, 现为中国科学技术大学教授, 目前主要研究方向: 并行算法和并行处理, 并行体系结构, 并行分布式计算中的通信问题, IP层多播技术的研究. E-mail: gunj@ustc.edu.cn.