MGO: 一种针对异构集群的全局通讯优化算法

李春华, 周兴铭
(并行与分布处理国家重点实验室, 国防科学技术大学计算机学院, 湖南长沙 410073)

摘 要: 在异构集群环境中, 网络拓扑结构的不规则性, 以及计算机结点和网络性能的差异, 影响了全局通讯的性能. 针对这个问题, 本文提出一种全局通讯的多粒度优化算法, 该算法结合网络拓扑结构以及计算机结点和网络性能等参数来优化全局通讯路径. 模拟结果表明, 多粒度优化算法与相关优化算法相比, 能显著提高全局通讯性能, 并且性能提高百分比在一定范围内随着集群规模的扩大而增大.

关键词: 全局通讯; 集群; 拓扑结构; 消息传递
中图分类号: TP393 文献标识码: A 文章编号: 0372-2112 (2002) 11-1643-05

MGO: An Optimized Collective Communication Algorithm for Heterogeneous Cluster

LI Chun-hua, ZHOU Xing-ming
(*National Laboratory for Parallel and Distributed Processing, School of Computer, National University of Defense Technology, Changsha, Hunan 410073, China*)

Abstract: In heterogeneous cluster environment, the network topology is irregular, and performance gap exists among computer nodes and among interconnected networks. These factors affect the performance of collective communication. To solve this problem, we present MGO (Multi-Granularity Optimization) algorithm. In MGO algorithm, we optimize the path for collective communication by network topology and the performance of the computer nodes and the networks. The simulation results show that by comparing MGO algorithm with related optimized algorithm, the former can result in notable performance improvement. Moreover, the performance-enhancing ratio increases to some extent as the cluster scales up.

Key words: collective communication; cluster; topology; message passing

1 引言

随着高速网络技术和高性能微处理器技术的发展, 集群的规模日趋扩大: 从局域网扩展到园区网甚至广域网. 研究和应用趋势表明, 分布式异构集群正成为一种极具吸引力的并行计算平台.

集群环境下并行程序设计的主流编程模式是消息传递, 作为这种编程模式的支撑软件, 消息传递库主要为上层并行应用提供点对点通讯和全局通讯功能, 其中全局通讯又具体分为广播(broadcast)、分布(scatter)、聚合(gather)、全聚合(all-gather)、规约(reduce)、同步(barrier)等操作. 在点对点通讯性能给定的条件下, 全局通讯对上层并行应用的性能影响非常大. 在分布式异构集群环境中, 计算机结点的异构性、网络的异构性, 以及网络拓扑结构的不规则性, 严重影响着全局通讯的效率, 如果全局通讯路径对这些异构因素缺乏适应性, 就会导致全局通讯的性能下降. 为此, 本文主要研究利用集群中计算机结点性能、网络性能、网络拓扑结构等信息来优化全局通讯路径, 以改善全局通讯对分布式异构集群的适应性, 从而最终提高上层并行应用的性能, 如图 1 所示.

本文后继内容安排如下: 第 2 节介绍适用于集群环境的全局通讯优化方面的已有相关工作, 第 3 节描述本文提出的全局通讯的多粒度优化算法, 第 4 节给出其性能模拟结果, 最后第 5 节总结我们的工作.

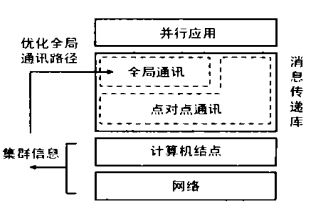

图 1 全局通讯优化

2 相关工作

我们把针对集群环境的全局通讯优化分为以下三类: 简单优化、细粒度优化、粗粒度+ 细粒度优化.

简单优化采用固定的全局通讯路径, 结点在路径中的位置由进程号来确定, 以 MPICH[1] 为代表. MPICH 是 Argonne 国家实验室和密西西比州立大学实现的 MPI 消息传递库, 也是目前最为流行的消息传递库. 它的某些全局通讯操作采取了简单的优化. 例如对于广播操作, MPICH 采用二项式树 BT(Binomial Tree) 来代替扁平树 FT(Flat Tree) 作为广播路径, 就是一

收稿日期: 2001-07-06; 修回日期: 2002-07-17

种简单优化. 简单优化适合于同构集群, 它的不足在于既没有考虑计算机结点和网络的性能, 也没有考虑网络拓扑结构, 因此对异构集群缺乏针对性.

细粒度优化以结点为单位, 依据计算机结点和网络性能来优化全局通讯路径, 例如 FNF 算法[2]、FEF 算法[3]、FCEF 算法[3]、LA 算法[3]等. 细粒度优化的基本思想是把结点分为发送消息结点和接收消息结点两类, 并依次从这两类结点中选择一个结点对作为发送结点和接收结点, 生成全局通讯路径的一条边, 迭代上述选择过程直到构成完整的通讯路径. 不同的细粒度优化算法所采用的结点选择策略有所不同, 如 FNF 算法优先选择性能高的结点, FEF 算法优先选择网络延时低的结点对, FCEF 算法优先选择先完成消息传递的结点对. 细粒度优化仅顾及了通讯路径局部的优化效果, 由于没有考虑网络拓扑结构, 就整个全局通讯路径而言, 还有很大的性能提高余地没有得到充分挖掘.

粗粒度+ 细粒度优化的基本思想是: 首先依据网络拓扑结构把整个集群分为不同的子集群(即粗粒度优化), 然后对不同的子集群分别进行细粒度优化. 例如 ECO[4]、MAGPIE[5,6]、MPICH-G[7,9]等系统对全局通讯就采取粗粒度+ 细粒度优化的方式. 粗粒度+ 细粒度优化的不足在于: 粗粒度优化与细粒度优化是分离的, 容易导致各子集群全局通讯操作完成时间差异大, 操作完成时间大的子集群成为制约全局通讯性能提高的瓶颈.

3 多粒度优化

针对上述相关工作的不足, 我们提出了全局通讯的多粒度优化(Multi-Granularity Optimisation). 多粒度优化的指导思想是把粗粒度优化和细粒度优化有机结合起来, 在优化过程中综合利用网络拓扑结构、网络性能、计算机结点性能等因素. 多粒度优化的基本框架包括多粒度优化算法及其所采用的通讯模型和拓扑结构模型, 下面逐一进行介绍.

3.1 通讯模型

通讯模型描述了结点间消息传递的基本过程, 多粒度优化框架中的通讯模型包含以下参数:

(1) L_{ij}(Latency): 网络延时, 短消息从源结点 i 传递到目的结点 j 时, 在网络中的传输时间;

(2) o_i(overhead): 短消息软件开销, 结点 i 发送或接收消息时的处理时间;

(3) b_{ij}(bandwidth): 应用层带宽, 结点 i 与结点 j 之间消息传递的应用层带宽;

两个结点 i、j, 结点 i 向 j 发送长度为 m 字节的消息, 其消息传递基本过程约定描述为: 首先源结点 i 对消息第一个字节进行发送处理, 花去处理时间 o_i, 然后消息进入网络传输, 经过网络延时 L_{ij}后到达目的结点 j, 结点 j 对接收到的消息进行接收处理, 花去处理时间 o_j后到达应用层, 再经过 $(m-1)/b_{ij}$时间, 后续的 $(m-1)$ 个字节也全部到达结点 j 的应用层, 整个过程如图 2 所示. 完成上述消息传递后, 结点 i、j 的就绪时间 RT_i、RT_j 分别为:

$$RT_i = o_i + \frac{m-1}{b_{ij}}$$

$$RT_j = L_{ij} + o_i + o_j + \frac{m-1}{b_{ij}}$$

此外, 我们还约定: 结点在就绪前不能再向其它结点发送消息或接收来自其它结点的消息.

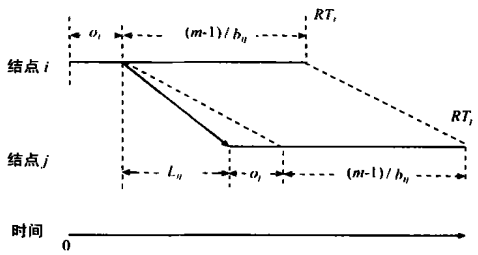

图 2 消息传递基本过程

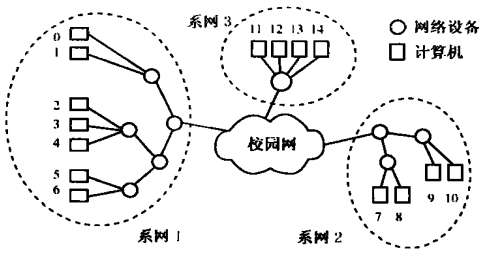

图 3 集群示例

3.2 拓扑结构模型

拓扑结构模型描述了集群的拓扑结构. 在多粒度优化基本框架的拓扑结构模型中, 重点突出集群拓扑结构的层次性, 因此采用树来描述. 例如图 3 所示的集群, 它的拓扑结构可以描述为如图 4 所示的拓扑结构树, 树中的叶结点为计算机结点, 非叶结点为连接结点, 对应着实际存在的一台或多台网络设备.

此外, 为描述方便起见, 我们约定以下一些概念.

子集群: 在拓扑结构树中, 对于其中任意一棵子树而言, 这棵子树上的所有计算机结点构成的集合称为一个子集群. 设子树的根结点为结点 r, 则称结点 r 为子集群的根结点, 并且记相应的子集群为 C_r. 例如图 4 中子集群 C_a= {结点 0, 结点 1, …, 结点 14}, 子集群 C_b= {结点 0, 结点 1, …, 结点 6}, 子集群 C_g= {结点 7, 结点 8}.

粒度: 子集群所包含的计算机结点的数目称为子集群的粒度. 例如图 4 中子集群 C_a 的粒度为 15, 子集群 C_b 的粒度为 7, 子集群 C_g 的粒度为 2.

层次: 子集群的根结点在拓扑结构树中的层次称为子集群的层次. 例如图 4 中子集群 C_a 的层次为 0, 子集群 C_b 的层次为 1, 子集群 C_g 的层次为 2.

标记: 在拓扑结构树中, 每个子集群的根结点都设置有一个属性位, 初始化为 0. 属性位为 0 的子集群称为未标记子集群; 属性位为 1 的子集群称为已标记子集群; 对子集群的根结点属性位置 1 称为对该子集群标记. 例如图 5 中子集群 C_a、C_b、C_f、C_i 是已标记子集群, 其余子集群 C_c、C_d、C_e、C_g、C_h、C_j

为未标记子集群.

当前标记层次: 未标记子集群的最小层次称为当前标记层次, 层次等于当前标记层次的子集群称为当前未标记子集群. 例如图 5 中当前标记层次为 1, 当前未标记子集群为 C_c、C_d.

最大未标记子集群: 在包含计算机结点 x 的所有未标记子集群中, 最大的子集群称为包含结点 x 的最大未标记子集群, 记为 MNC_x. 例如图 5 中结点 5 的最大未标记子集群为 C_j, 即 $MNC_5 = C_j$; 结点 7 的最大未标记子集群为 C_e, 即 $MNC_7 = C_e$.

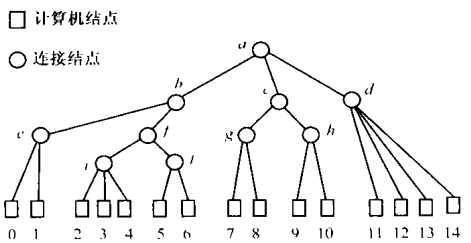

图 4　拓扑结构树

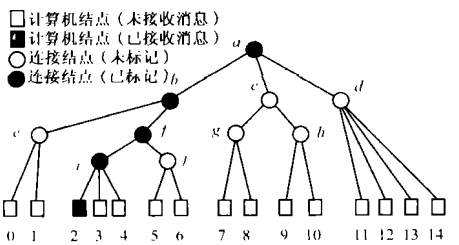

图 5　标记

3.3　算法

全局通讯路径优化过程是选择发送结点、接收结点并逐条生成通讯路径的边的过程, 在多粒度优化算法中, 结点的选择过程分两步: 候选结点选择以及具体结点选择, 这两步分别以上述拓扑结构模型和通讯模型为依据. 下面以一对多通讯的多粒度优化为例进行说明.

候选结点选择: 从未接收到消息的结点中选择满足一定条件的结点作为候选接收结点. 候选接收结点分为两类, 分别保存在集合 $B1$、$B2$ 中. 候选接收结点选择以拓扑结构树为依据进行: 当前未标记子集群的所有结点划入集合 $B1$; 集合 $B2$ 与发送结点相关, 对于某个发送结点, 包含该发送结点且层次为当前标记层次的子集群中未接收到消息的结点都划入集合 $B2$. 在这一步, 已接收到消息的结点不用选择, 都作为候选发送结点, 保存在集合 A 中. 只有具备候选资格的结点才能参与下一步具体结点选择.

具体结点选择: 从集合 A 中选择结点 s 作为发送结点; 从集合 $B1$、$B2$ 中分别选择 $r1$、$r2$ 作为接收结点. 依次生成全局通讯路径的两条边: 先从结点 s 到 $r2$, 然后从结点 s 到 $r1$. 具体结点选择以通讯模型为依据来计算, 原则是使得 s 依次发送消息给 $r2$、$r1$ 后结点 $r1$ 的就绪时间最小.

完成上述两步结点选择过程后, 把结点 $r1$、$r2$ 移入集合 A, 同时标记包含 $r1$ 或 $r2$ 的所有未标记子集群, 再循环重复结点选择过程, 直到所有的子集群都已标记.

在上述优化过程中, 注意以下几点: ①利用拓扑结构树进行候选结点选择体现了粗粒度优化, 利用通讯模型进行具体结点选择体现了细粒度优化; ②候选结点选择和具体结点选择相互影响: 候选结点选择的结果确定了具体结点选择的范围, 具体结点选择的结果确定了需要标记的未标记子集群, 从而确定了下一循环结点选择过程的候选结点. 因此, 粗粒度优化与细粒度优化是有机结合的; ③候选结点选择与当前标记层次、当前未标记子集群相关, 就整个优化过程而言, 随着优化的进行, 当前标记层次逐渐变大、当前未标记子集群粒度逐渐变小, 也就是说粗粒度优化的粒度在由大变小.

由于优化过程有机结合了粗粒度优化与细粒度优化, 并且粗粒度优化的粒度由大到小, 可以说整个过程包含了多种粒度的优化, 所以我们称之为多粒度优化.

完整的一对多通讯多粒度优化算法描述如下:

变量说明:

$Depth$: 当前标记层次;

A: 候选发送消息结点集合, 包含已经接收到消息的计算机结点;

$B1$: 候选接收消息结点集合 1, 包含当前未标记子集群中的所有计算机结点;

$B2$: 候选接收消息结点集合 2, 它与发送结点相关, 对于某个发送结点, 包含该发送结点且层次为 $Depth$ 的子集群中未接收到消息的计算机结点都属于该集合;

C: 集群中所有计算机结点的集合, $C = \{$结点 0, 结点 1, …, 结点 $P\}$

RT_i: 结点 i 的就绪时间, $0 \leqslant i \leqslant P$;

L_{ij}: 短消息从结点 i 传递到结点 j 时, 在网络中的传输延时, $0 \leqslant i \leqslant P$, $0 \leqslant j \leqslant P$;

o_i: 结点 i 发送或接收短消息时的处理时间, $0 \leqslant i \leqslant P$;

b_{ij}: 结点 i 与结点 j 之间消息传递的应用层带宽, $0 \leqslant i \leqslant P$, $0 \leqslant j \leqslant P$;

s: 指向所选择的具体发送结点, $0 \leqslant s \leqslant P$;

$r1$, $r2$: 指向所选择的具体接收结点, $0 \leqslant r1$, $r2 \leqslant P$;

i, $j1$, $j2$, RT, rt: 临时变量;

一对多通讯 MGO 算法:

输入: 一对多通讯的顶点 $s0$, 消息长度 m(字节);

输出: 一对多通讯路径;

[1] 令候选发送结点集合 $A = \{s0\}$, 标记包含结点 $s0$ 的所有子集群;

[2] 集合 $B1 =$ 当前未标记子集群中的所有计算机结点, $RT_{s0} = 0$, $Depth = 1$;

[3] while($B1 \neq \Phi$) do

{

[4] $RT = +\infty$;

[5] 对于集合 A 中的结点 i、集合 $B1$ 中的结点 $j1$ 以及集合 $B2$ 中的结点 $j2$($B2$ 为包含结点 j 且层次为 $Depth$ 的

子集群中未接收消息结点的集合), do:

{

[6] 如果 $j2 = = \phi$,

{

[7] $rt = RT_i + L_{ij1} + o_i + o_{j1} + \frac{m-1}{b_{ij1}}$;

[8] 如果 $rt < RT$, $s = i$, $r1 = j1$, $r2 = \phi$, $RT = rt$;

}

else

{

[9] $rt = RT_i + 2o_i + \frac{m-1}{b_{ij2}} + L_{ij1} + o_{j1} + \frac{m-1}{b_{ij1}}$;

[10] 如果 $rt < RT$, $s = i$, $r1 = j1$, $r2 = j2$, $RT = rt$;

}

}

[11] $A = A + \{r1\}$, $B1 = B1 - MNC_{r1}$, 标记包含结点 $r1$ 的所有未标记子集群;

[12] 如果 $r2 \neq \phi$

{

[13] $A = A + \{r2\}$, 标记包含 $r2$ 的所有未标记子集群;

[14] 输出边 $\vec{s\ r2}$;

[15] 修改结点 s、$r2$ 的就绪时间:

$$RT_s = RT_s + o_s + \frac{m-1}{b_{sr2}};$$

$$RT_{r2} = RT_s + L_{s2} + Or_2;$$

}

[16] 输出边 $\vec{s\ r1}$;

[17] 修改结点 s、$r1$ 的就绪时间:

$$RT_s = RT_s + o_s + \frac{m-1}{b_{sr1}};$$

$$RT_{r1} = RT_s + L_{s1} + Or_1;$$

[18] 如果 $B1 = = \phi$,

{

[19] 令 $B1 =$ 当前未标记子集群中的所有计算机结点;

[20] $Depth =$ 当前标记层次;

}

}

[21] $B1 = C - A$;

[22] while($B1 \neq \phi$) do

{

[23] $RT = +\infty$;

[24] 对于集合 A、$B1$ 中的结点 i、j 1, do

{

[25] $rt = RT_i + L_{ij1} + o_i + o_{j1} + \frac{m+1}{b_{ij1}}$;

[26] 如果 $rt < RT$, $s = i$, $r1 = j1$, $RT = rt$;

}

[27] $A = A + \{r1\}$, $B1 = B1 - \{r1\}$;

[28] 输出边 $\vec{s\ r1}$;

[29] 修改 s、$r1$ 的就绪时间:

$$RT_s = RT_s + o_s + \frac{m-1}{b_{sr1}};$$

$$RT_{r1} = RT_s + L_{s1} + o_{r1}$$

}

[30] 结束.

把上述一对多通讯 MGO 算法稍加改动: A 为候选接收结点集合, $B1$、$B2$ 为候选发送结点集合(就绪时间的计算、通讯路径边的方向也做相应修改), 即为多对一通讯优化的 MGO 算法, 由于篇幅所限这里不再复述.

对于短消息和长消息广播, 我们采用一对多通讯 MGO 算法生成的路径(优化时 m 不同); 对于短消息分布, 采用一对多通讯 MGO 算法生成的路径, 而长消息分布不采用 MGO 优化; 对于短消息聚合, 采用多对一通讯 MGO 算法生成的路径, 而长消息聚合不采用 MGO 优化; 对于短消息全聚合, 先采用多对一通讯 MGO 算法生成的路径, 再采用一对多通讯 MGO 算法生成的路径, 而长消息全聚合不采用 MGO 优化. 之所以对长消息的分布、聚合、全聚合不采用 MGO 优化, 是考虑到对于这些操作, 长消息合并会导致合并后的消息长度急剧增大.

基于多粒度优化算法, 我们对 MPICH 1.2.0 的全局通讯层进行了补充、修改, 实现了部分全局通讯操作的多粒度优化, 我们称为 MPICH-MGO, 关于 MPICH-MGO 的具体情况将另文发表.

4 模拟

利用模拟器 CCSim 来研究异构集群环境中不同的全局通讯优化算法的优化效果. 模拟过程分三步: 首先随机生成不同配置的异构集群, 其中计算机结点数为 P(P = 4, 8, 16, 32, …, 512); 然后任意选择其中一个结点作为顶点, 分别执行 FT、BT、FEF、FCEF、LA、MGO 算法生成广播、分布、聚合、全聚合等全局通讯操作的路径, 其中 m = 1kB 时为短消息路径, m = 100kB 时为长消息路径; 最后利用路径和集群的配置信息分别模拟广播、分布、聚合、全聚合的通讯过程, 获得它们的完成时间. 重复上述模拟过程多次, 取平均值作为最终结果.

表 1 短消息广播完成时间(ms, Size = 1kB)

	FT	BT	FEF	FCEF	LA	MGO
4	952.3	1101.4	1176.89	1089.23	1080.26	868.15
8	1875.86	1808.43	2215.94	1897	1813.38	1359.98
16	3691.08	2551.2	3977.41	2747.5	2836.58	1838.55
32	7035.98	3307.44	6206.96	3396.18	3540.57	1964.7
64	13647.26	3900.3	10209.58	4185.06	4360.98	2067.95
128	26685.98	4825.95	15130.92	4567.74	5071.14	2139.8
256	53447.26	5962.87	21916.77	5116.04	5561.21	2242.4
512	105479.94	7150.52	32858.43	5626.32	5789.99	2265.62

最后得到的广播(短消息)完成时间如表 1 所示, 表中第一列表示结点数, 第一行表示不同优化算法的广播操作(短消息). 为了比较 MGO 算法的优化效果, 利用完成时间计算 MGO 优化广播相对于其它优化广播的性能提高百分比, 结果见图 6: 在结点数 P 为 4~512 时, MGO 优化广播(短消息)比 FT 广播(短消息)性能提高 9%~98%, 比 BT 优化广播(短消息)性

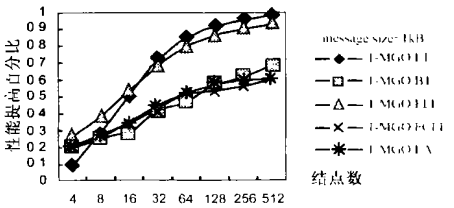

图 6　短消息广播性能对比

能提高 21% ~ 68%, 比 FEF 优化广播(短消息) 性能提高 26% ~ 93%, 比 FCEF 优化广播(短消息) 性能提高 2% ~ 60%, 比 LA 优化广播(短消息) 性能提高 2% ~ 61%. 并且随着集群规模的扩大, MGO 优化广播(短消息) 比其它优化算法的广播(短消息) 性能提高百分比也在一定范围内逐渐增大.

对于广播(长消息)、分布(短消息)、聚合(短消息)、全聚合(短消息), 相应的性能提高百分比曲线与广播(短消息)的类似, 如图 7 所示.

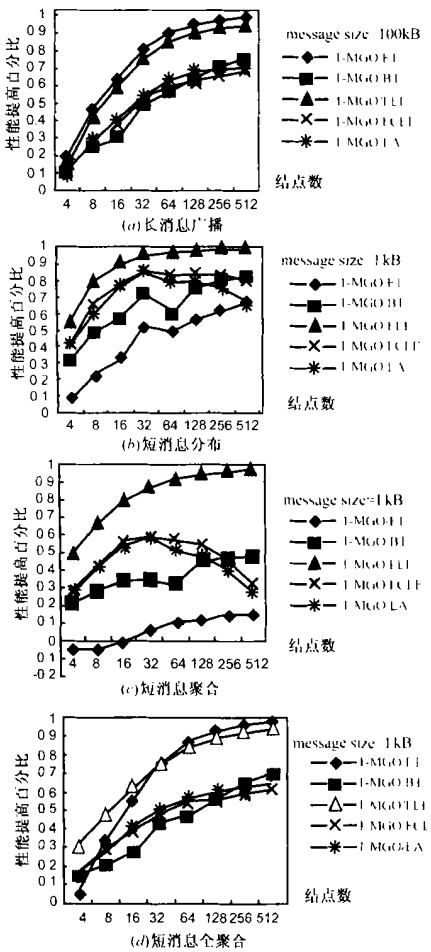

图 7　其它全局通讯操作性能对比

5　结束语

消息传递库的全局通讯对基于异构集群环境的并行应用性能影响很大. 针对此问题, 本文提出了一种全局通讯的多粒度优化算法, 该算法在生成通讯路径时综合利用网络拓扑结构、网络性能、计算机结点性能等集群信息, 把宏观优化和微观优化有机结合起来, 增强了全局通讯对异构集群的适应性. 模拟结果表明, MGO 算法优化效果明显, 与 FT、BT、FEF、FCEF、LA 优化算法相比, 它的广播、分布、聚合、全聚合的性能均有不同程度的提高, 并且性能提高百分比在一定范围内随着集群规模的扩大而增大.

参考文献:

[1] William Gropp, Ewing Lusk, et al. A high performance, portable implementation of the MPI message passing interface standard[J]. Parallel Computing, 1996, 22(6): 789-828.

[2] Mohammad Banikazemi, Vijia Moorthy. Efficient collective communication on heterogeneous networks of workstations[A]. International Conference on Parallel Processing[C]. Los Alamitos: IEEE Press, 1998. 460-467.

[3] Prashanth B Bhat, C S Raghavendra, et al. Efficient collective communication in distributed heterogeneous system[A]. Proceedings of the 19th IEEE International Conference on Distributed Computing Systems[C]. Austin, Texas: 1999. 15-24.

[4] Bruce B Lowekamp, Adam Beguelin. ECO: Efficient collective operations for communication on heterogeneous networks[A]. International Parallel Processing Symposium[C]. Honolulu, HI: 1996. 399-405.

[5] Thilo Kielmann, Rutger F H Hofman. MAGPIE: MPI's collective communication operations for clustered wide area systems[A]. Proceedings of the Seventh ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming[C]. Atlanta, GA: May 1999. 131-140.

[6] T Kielmann, H E Bal, et al. Bandwidthr efficient collective communication for clustered wide area systems[A]. Proceedings of the 14th International Parallel and Distributed Processing Symposium[C]. 2000. 492-499.

[7] Nicholas T Karonis, Bronis R De Supinski. Exploiting hierarchy in parallel computer networks to optimize collective performance[A]. Fourteenth International Parallel and Distributed Processing Symposium (IPDPS'00)[C]. Cancun, Mexico: May, 2000. 377-384.

[8] Ian Foster, Jonathan Geisler, et al. Wide-area implementation of the message passing interface[J]. Parallel Computing, 1998, 24(12-13): 1735-1749.

[9] Ian Foster and Nicholas T. Karonis. A grid-enabled MPI: Message passing in heterogeneous distributed computing systems[A]. Proceedings of Supercomputing'98[C]. Orlando, FL: Nov, 1998.

作者简介:

李春华　男, 1972 年 3 月生于广西柳州市, 博士, 国防科技大学计算机学院助理研究员, 感兴趣的研究方向为高性能计算、互联网技术与应用、移动数据通信、嵌入式系统.

周兴铭　男, 1938 年 12 月生于上海市, 中国科学院院士, 国防科技大学计算机学院教授、博士生导师, 主要研究领域为高性能计算机体系结构、并行与分布式数据库、先进计算机网络技术.