一种面向高阶胖树源路由网络的组播实现方法

曹继军 王永庆 刘 路
(国防科技大学计算机学院 长沙 410073)

摘 要 组播是一种多机通信系统中可支持多种聚合通信服务的重要操作。与基于单播和路径的方法相比,基于树的组播通常具有较高的效率。针对高阶胖树源路由网络,提出了一种新型实现方法—DMFTAR。该方法将组播功能实现分为组播服务层、组播路由层和组播转分层等3个层次,其特点是基于分布式组播转发表和异步数据复制实现组播操作。理论分析表明,与传统的基于多头微片虫蚀异步数据复制实现方法相比,DMFTAR方法通信开销低且扩展性好。

关键词 源路由,胖树,组播算法,单播路由表
中图法分类号 TP393 **文献标识码** A

Implementation Method for High-radix Fat-tree Deterministic Source-routing Interconnection Network

CAO Ji-jun WANG Yong-qing LIU Lu
(School of Computer, National University of Defense Technology, Changsha 410073, China)

Abstract Multicast is an important operation in multicomputer communication systems and can be used to support several other collective communication operations. Comparing with the unicast-based multicast approach or path-based multicast approach, the tree-based multicast approach achieves more efficiency. This paper presented a Distributed Multicast-Forward-Table and Asynchronous Replication (DMFTAR) based method to implement the multicast operation. According to the DMFTAR method, the implementation is divided into Multicast Service Layer (MSL), Multicast Routing Layer (MRL) and Multicast Forwarding Layer (MFL). Theoretic analysis results show that the DMFTAR method achieves more scalability and less overhead than the traditional implementation method which is based on Multi-Head Worm-Hole Asynchronous Replication (MHWAR).

Keywords Source-routing, Fat tree, Multicast algorithm, Unicast routing table

1 引言

高性能计算系统由互连网络将大量处理节点连接而成。通过互连网络,分布在系统中不同节点上的多个进程协作完成计算任务。根据参与操作的处理进程数目的不同,通信模式可分类为单播通信和聚合通信两种。单播通信只涉及一个数据发送进程和一个数据接收进程,而聚合通信则涉及一组进程。统计显示,在许多大规模科学计算应用中,聚合通信时间占程序全部执行时间的60%以上[1],聚合通信开销占全部通信开销的80%以上[2]。因此,聚合通信的性能是影响互连网络性能的重要因素,一直是高性能计算领域的研究热点。

从应用软件角度,聚合通信服务的基本类型包括4种:多个点到点通信、一对多通信、多对一通信和多对多通信。除此之外,某些聚合通信是这些基本类型通信的组合,因此也称为合成聚合通信服务。从硬件系统角度,聚合通信的基本操作包括两种:一是组播(Multicast),即一个节点向一组节点发送消息;二是规约(Reduce),即一组节点向一个节点发送消息,并在该节点上完成求和、求最大值/最小值等操作。这两种基本操作可以用于实现其他聚合通信操作。可见,组播是一种重要的通信操作,优化组播通信的性能是改善系统聚合通信性能的关键,也是提高计算系统整体性能的重要措施。

根据现有研究,实现组播的方法大致分为3类:基于单播(Unicast-based)的方法、基于路径(Path-based)的方法和基于树(Tree-based)的方法。基于单播的方法中,源节点向多个目的节点单独发送单播消息[3]。该方法通常都是基于MPI(Message Passing Interface)的点到点消息传递接口实现,通过进程间的消息传递,最终完成集合通信的语义。该方法的优点是无需增加额外的硬件支持,实现较为简单;其缺点是无法有效地隐藏消息发送的启动延迟,组播延迟较大。基于路径的方法中,报文的头微片包含多个目的地址,路由器硬件通过内部通道吸收微片以传递给本地处理节点,同时复制微片到外部通道以路由到其余目的节点[4]。哈密顿路径算法[5]和BPCP-HL算法[6]等是这种方法的典型代表。该方法的优点是可以隐藏消息发送的启动延迟;其缺点是节点网络接口需要多次复制和转发消息,效率较低。例如,哈密顿算法通常会因为缺乏有效解决组播操作引起的死锁和报文复制的方法,而导致整个算法的性能受困于较长路径引起的高延迟或多个

到稿日期:2012-02-29 返修日期:2012-07-06 本文受国家863高技术研究发展计划基金(2012AA01A301)资助。
曹继军(1979—),男,博士,助理研究员,主要研究方向为互连网络、组播技术等,E-mail:caojijun@nudt.edu.cn;王永庆(1973—),男,博士,副研究员,主要研究方向为高性能互连网络、链路层协议等;刘 路(1971—),男,硕士,副研究员,主要研究方向为高速互连网络、用户级通信等。

报文启动引起的高开销。基于树的组播方法中，源节点到各个目的节点间的路径构成树结构，源节点和目的节点分别为树根和树叶。在上述3种方法中，基于单播的方法可认为是软件实现方法，而基于路径和组播树的方法可认为是硬件方法[7]。

与前两种方法相比，基于树的组播使用了最短路径路由，因此具有较高效率。然而，在采用虫蚀方式的基于树结构实现组播操作的互连网络中，组播树的各个分枝间由于存在阻塞依赖关系而很容易造成死锁[8]。所以，基于树的组播方法通常使用虚切通路由(Cut-Through Routing)[9]，即路由器的缓冲区可缓存多个完整的消息，从而降低树分枝间发生阻塞依赖的可能性。为了实现基于树的组播，可以为消息构造多个头微片(Head Flit)，而路由器根据头微片信息进行消息分割并向多个输出端口复制数据[10]。数据复制有两种方法：同步复制和异步复制[11]。由于同步复制机制硬件实现较为复杂，因此异步复制机制成为实现中经常采用的方法。为方便起见，本文称上述多头微片虫蚀异步数据复制(Multi-Head Worm-Hole Asynchronous Replication，MHWAR)的组播实现方法为MHWAR方法。

2 高阶胖树源路由网络

由于具有高等分带宽、低网络直径、良好的扩展性和丰富的路径等特点，胖树被广泛应用于超级计算机和数据中心等商用网络中。而以高阶路由器(通常指端口数≥16)芯片为基础的胖树网络可称为高阶胖树网络，并且称采用确定性源路由的高阶胖树网络为高阶胖树源路由网络。在2010年11月世界超级计算机TOP500排名中，位列第一的天河一号(TH-1A)[12]计算机系统使用的便是高阶胖树源路由网络。为方便起见，本文称该网络为HFDS-IN(High-Radix Fat-Tree Deterministic Source-Routing Interconnection Network)。

HFDS-IN采用层次式模块化结构构建网络系统。网络的互连、交换和路由功能都由高阶路由器(High Radix Network Router，HRNR)芯片实现。以HRNR为基本单元，来构成计算机柜内的接入层交换模块，从而实现同一计算机柜内节点间的互连通信；还可以构成通信机柜内的汇聚层交换模块，从而实现不同计算机柜节点间的互连通信。因此，整个互连网络系统呈现两级层次式结构，即底层接入层和上层汇聚层。在接入层，每32个计算结点使用由8个HRNR芯片组成的交换模块实现互连。在汇聚层，由两种交换模块构成交换机以连接底层的交换模块，这两种模块分别由6个和2个HRNR芯片构成，它们之间采用正交方式连接。基于上述互连结构，计算节点间构成一棵高度为5的胖树，其拓扑结构如图1所示。

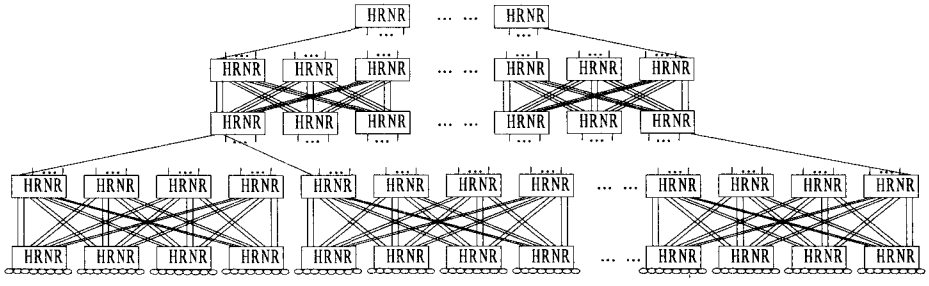

图1 HFDS-IN的拓扑结构

HFDS-IN采用最短路由，即先将数据报文向上转发到源和目的节点的最近公共祖先路由器，然后向下转发直到目的计算节点为止。通常，在上行阶段，接入层交换模块的任意一个上行端口都可被选择；而在下行阶段，汇聚层交换模块的部分端口可被选择。在设计路由过程中，主要需要考虑流量均衡问题，即如何避免产生热点路径，从而使得胖树中相同高度的路径上承担的流量大小尽量相等。

HFDS-IN的单播路由信息包含在报文的头微片的路由域中。路由域采用固定长度，其结构如图2所示。路由的最大跳数设计为9，所以路由域中的“有效Hop数”最大值为9，而每一跳分别由Hop1、Hop2、…、Hop9指定。单个Hop的长度为4位，因此可表示16个端口，其含义是将接收到的该微片及其属于同一报文的后续体微片转发到下一跳时的输出端口。微片长度设计为256位，所以除包含路由域等报文头信息(共48位，包含信用Credit、响应Ack和报文类型等)和链路层可靠性维护信息(共20位，包含CRC校验、微片类型和虚信道号等)外，头微片还包含188位的数据信息。

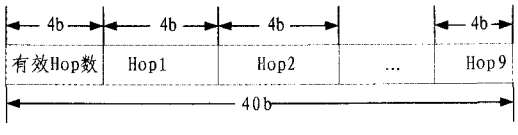

图2 路由域结构

由于HFDS-IN采用的是源路由，因此系统中每个节点都保存着一张单播路由表，该路由表的各表项指出了该节点到系统中任意节点的路由域。而单播路由表是路由设计人员在系统运行前根据拓扑结构和路由策略提前生成，并加载到各个计算节点上的。当通信时，节点通过查路由表获取到目的节点的路由域信息，并将该信息写入报文的头微片中。报文微片在路由的过程中，路由器将头微片中的“有效Hop数”减1，同时取出Hop1域中的信息作为该微片的输出端口，然后将Hop1、Hop2、…、Hop9中的信息依次左移4位。在正常情况下，当“有效Hop数”减少到0时，该报文将到达目的节点。

3 基于分布式组播转发表的组播

为了使HFDS-IN能够对组播提供高效的硬件支持，从而提高聚合通信的性能，本文提出了一种新型的组播实现技术，这种技术并不需要改变现有HFDS-IN的体系结构，只需要在HFDS-IN的实现上做一些扩展，即在对单播性能产生较小影响的前提下，在路由芯片中以尽量低的硬件代价实现组播操作。这种新方法的主要特点是采用基于分布式组播转发表和异步数据复制实现组播操作。为方便起见，本文称其为DMFTAR(Distributed Multicast-Forward-Table Based and Asynchronous Replication)方法。

3.1 基本思想

在 HFDS-IN 胖树中，每个 HRNR 芯片只是简单地根据报文中携带的输出端口信息进行报文转发，或将其转发给相邻的 HRNR 芯片或将其转发给相连的计算节点。这种简单的路由协议设计便于对关键部件和关键操作进行性能优化。虽然 MHWAR 方法可实现 HFDS-IN 的组播操作，但其主要缺点是扩展性差。因为在该方法下，每个组播报文的头微片数目等于组播组成员节点数目，而当前超级计算机规模不断增大，例如 TH-1A 系统包含 7168 个节点[13]，所以采用该方法实现全系统内节点间的组播将会使组播性能很差。为此，本文面向 HFDS-IN 提出的 MHWAR 方法尽量保持现有单播路由协议简单的特点，并解决 MHWAR 方法的扩展性问题。DMFTAR 方法的基本原理描述如下。

多个成员节点构成一个组播组，每个组播组由唯一的 GID(Group ID)标识。组播报文中通过携带 GID 指出要到达的组成员节点。每个 HRNR 芯片都维护一张组播转发表，其表项记录了 GID 与输出端口集合间的映射关系。对于某个标识为 GID 的组播组而言，其分布在多个 HRNR 芯片的组播转发表中索引为 GID 的表项便完全描述了该组播树的拓扑结构。组中任意计算节点都可以沿着该组播树的边向组内所有其他节点发送组播报文。组播转发表是以单播转发表为基础产生的。具体而言，首先，组播源节点通过查找单播路由表获取到组内其他各个节点的单播路径，这些单播路径集合就汇聚成一棵组播树。该过程无需进行复杂的计算过程，只需要多次查找单播路由表即可实现；然后，组播源节点将组播树的结构信息分发到该组播树所覆盖的 HRNR 芯片中。HRNR 负责组播转发，即首先根据组播报文中携带的 GID 查找组播转发表，以获得输出端口集合，然后向集合中的除报文接收端口外的各个端口发送组播报文。

就功能层次而言，DMFTAR 方法的具体实现可分为组播服务层(Multicast Service Layer，MSL)、组播路由层(Multicast Routing Layer，MRL)和组播转发层(Multicast Forwarding Layer，MFL)等 3 层。每一层要实现的功能以及 3 层间的关系如图 3 所示。具体实现时，组播服务层为软件实现，而组播路由层和组播转分层为硬件实现。

组播服务层(MSL)	1. 创建组播组 2. 通告组播组 3. 计算组播树 4. 分发组播树 5. 删除组播树 6. 组播数据
组播路由层(MRL)	1. 生成组播转发表表项 2. 删除组播转发表表项 3. 组播转发表老化处理
组播转发层(MFL)	异步复制组播数据

图 3 DMFTAR 方法的功能层次

3.2 组播服务层 MSL

目前，并行计算领域应用最广泛的编程模型是消息传递模型，而 MPI 已成为这种编程模型的代表和事实标准。MSL 主要向上层软件提供组播编程接口，基于 MSL 可以高效实现 MPI 的部分组通信功能。

MSL 实现的功能函数主要包括：(1) CreateGroup (Nodelist，GID)，即根据组成员节点列表创建组播组，并返回组标识 GID；(2) NotifyGroup(Nodelist，GID)，即向组成员节点通告创建的组播组标识；(3) CalculateMTree (GID，MTree)，即根据组内包含的成员节点和单播路由表计算组播树；(4) DispatchMTree(GID，MTree)，即将计算获得的组播树分发到相关 HRNR，并由 HRNR 将其转换为组播转发表表项；(5) DeleteMTree(GID)，即删除本地维护的组播组，并通告相关 HRNR 删除组播转发表的相关表项；(6) MulticastData(GID，DataBuff)，即向组内所有节点组播数据；组内任何成员节点在获得 GID 后，都可以通过该函数项组内的其它成员节点组播数据。在上述函数中，GID 可以由〈NodeID，Seq〉二元组信息构成，其中 NodeID 为创建组的节点的标识，Seq 为顺序分配的自然数序号。MTree 可通过多维链表实现。

算法 1 基于单播路由表的组播树生成算法

输入：组播组成员节点 $N=\{N_0,N_1,\cdots,N_{k-1}\}$；

单播路由表 URT；

输出：MTree.

开始：初始化组播树 MTree 为空；

```
for (i=0;i≤k−1;i++)
  {
    查单播路由表 URT，获得 Ni 的路由域 URT^Ni =〈d,h=
      h0h1…hd−2hd−1〉;
    for (j=0;j≤d−1;j++)
    {
      If (h0h1…hj∈MTree and hj+1…hd−2hd−1∉ MTree)
      {
        将每跳入端口和路径 hj+1…hd−2hd−1纳入到 MTree;
        Break;
      }
    }
  }
```

结束

实现 MSL 的关键是如何根据单播路由表生成组播转发表，该问题涉及两个方面：一是如何根据组成员节点集合和单播路由表产生组播树；二是如何将组播树分发到相关的所有 HRNR 并将其转换为组播转发表表项。其中，第二方面的问题需要增加新的硬件资源加以解决，这将在 MRL 中详细讨论，而第一方面的问题可通过基于单播路由表的组播树生成算法来解决。为此，定义 1 将给出 HFDS-IN 单播路由表的定义，算法 1 将具体描述基于单播路由表的组播树生成算法。

定义 1 HFDS-IN 系统中每个节点都存储着一张单播路由表，假设系统中共有 N 个节点且节点编号为 0，1，2，…，$N-1$，假设编号为 m 的节点的单播路由表表示 URT_m，则 URT_m 的第 i 表项指明编号为 m 的节点到编号为 i 的节点的路由路径($m,i\in\{0,1,\cdots,N-1\}$)，该路径可表示为 $URT_m^i=\langle d,h=h_0h_1\cdots h_{d-2}h_{d-1}\rangle$，其中 d 为跳步数，即路径所经过的 HRNR 的数目；$h\in\{0,1,\cdots,f\}^d$ 为路由路径，h_k($k\in\{0,1,\cdots,d-1\}$)即为路径所经过的第 $k+1$ 跳交换机的输出端口。

例如，假设组播组成员节点构成的集合为 $N=\{17,22,26,99,107,211,280,284,288,292\}$，节点 211 负责创建组播组，节点 211 的单播路由表中包含的到组内其余节点的路由域分别为：$URT_{211}^{17}=\langle 3,311\rangle$、$URT_{211}^{22}=\langle 3,316\rangle$、$URT_{211}^{26}=$

$\langle 3,31a\rangle$、$URT_{211}^{99}=\langle 3,3f3\rangle$、$URT_{211}^{107}=\langle 3,3fb\rangle$、$URT_{211}^{280}=\langle 3,7e0\rangle$、$URT_{211}^{284}=\langle 3,7e4\rangle$、$URT_{211}^{288}=\langle 3,7e8\rangle$、$URT_{211}^{292}=\langle 3,7ec\rangle$。则节点 211 利用上述算法构造的组播树的结构如图 4 所示，其中，实黑色箭头两端的端口连接为固定的物理连接，例如光纤连接等。

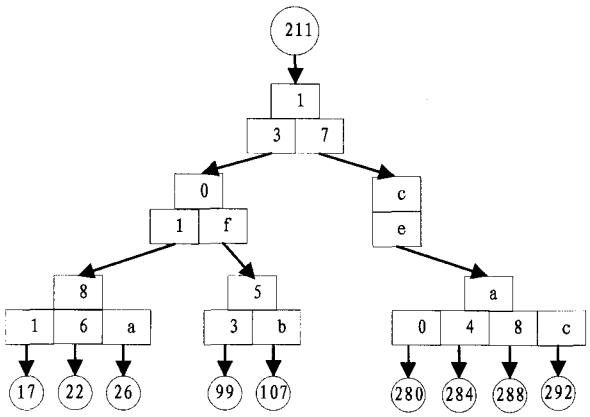

图 4　组播树示例

3.3　组播路由层 MRL

为了实现组播操作，需要在单播实现的基础上增加组播报文类型，为此可在报文头信息的报文类型域中对其进行区分。根据功能不同，组播报文分为组播控制报文和组播数据报文，而组播控制报文又可分为组生成报文和组删除报文等类型。除组播数据报文可包含多个微片外，其余报文均由一个微片构成，微片仍然固定为 256 位长度。MRL 主要功能是定义和处理组播控制报文。

图 5 所示为组播报文格式，其具体含义如下：(1)组生成报文的路由域(*RoutingField*)表示该报文经过的路径，而 *PortListx*($1\leqslant x\leqslant 9$)表示第 x 跳 HRNR 参与组播 GID 的端口列表；(2)组删除报文的路由域(*RoutingField*)表示该报文经过的路径，而 GID 表示要删除的组播转发表表项索引；(3)组播数据报文直接用 GID 指明报文对应的组播树。下面，定义 2 将给出 HFDS-IN 组播转发表的定义，算法 2 将具体的描述根据组播树对组播转发表进行配置的方法——组播树分发算法。

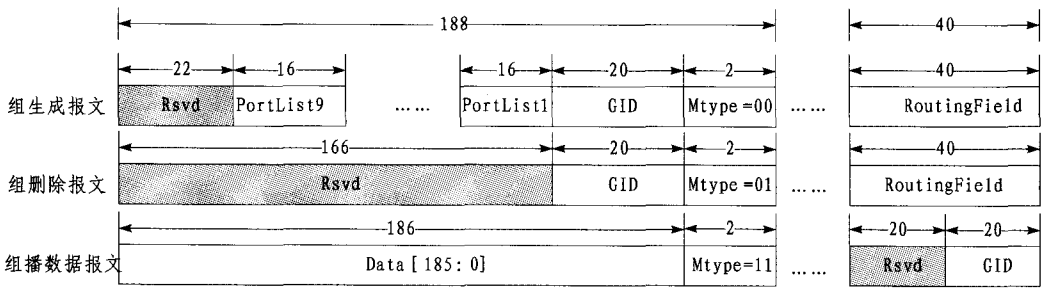

图 5　组播报文格式

算法 2　组播树分发算法

输入：组播树 MTree.

　　组播组标识 g

输出：组生成报文.

开始：对于 MTree 每个叶 HRNR 节点，执行

　　{

　　在 MTree 中获得到该 HRNR 的路由域$\langle d,h=h_0h_1\cdots h_{d-2}h_{d-1}\rangle$；

　　执行如下过程以构造组生成报文：

　　{

　　GID ← g;

　　RoutingField ← $\langle d,h=h_0h_1\cdots h_{d-2}h_{d-1}\rangle$;

　　for (x=1;x≤d;x++)

　　{

　　PortListx ← Mtree 中路径 $h_0h_1\cdots h_x$ 到达的 HRNR 的组播端口集合;

　　}

　　}

　　发送该组生成报文;

　　}

结束

定义 2　HFDS-IN 系统中每个 HRNR 都存储着一张组播转发表，假设组播转发表表示为 MFT，则 $MFT=\{\langle GID,PortList\rangle\}$，其中 GID 为组播组标识，$PortList=\{h\mid h\in\{0,1,\cdots,f\}\}$为参与该组播组的所有端口构成的集合。该表是以 GID 为关键字的索引表，所以组播组标识为 g 的本地参与端口集合可表示为 MFT^g。

假设组播组生成报文简单表示为$\{GID,RoutingFild,\{PortList1,PortList2,\cdots\}\}$，则对于图 4 所示的组播树，节点 211 只需要发送 3 个组生成报文便可以完成组播转发表的配置。这 3 个报文分别是$\{g,\langle 2,31\rangle,\{\{1,3,7\},\{0,1,f\},\{1,6,8,a\}\}\}$、$\{g,\langle 2,3f\rangle,\{\{1,3,7\},\{0,1,f\},\{3,5,b\}\}\}$和$\{g,\langle 2,7e\rangle,\{\{1,3,7\},\{c,e\},\{0,4,8,a,c\}\}\}$。当完成组播树分发后，组播组内其它节点在被告知创建的组播组的标识 g 后即可以直接向该组组播数据。

与组播树分发算法相似，删除组播组时也只需要向组播树 MTree 叶 HRNR 发送组删除报文。所有接收到组生成或删除报文的 HRNR，将依据报文中的相关信息在组播转发表中增加表项或删除表项。为了应对 MSL 管理组播组过程中可能出现表项残留问题，MRL 需要对组播转发表中的表项进行超时老化处理。

3.4　组播转发层 MFL

MFL 的主要功能是定义和处理组播数据报文。图 5 已给出组播数据报文格式。HRNR 在接收到组播数据报文时，需要以报文中指定的 GID 为关键字，查组播转发表。假设报文进入该 HRNR 的端口号为 P_{in}，则该报文的输入端口集合为 $MFB^{GID}-\{P_{in}\}$。为了避免死锁，MFL 采用基于输入缓冲的异步复制方式[14]，即并不是组播数据报文申请仲裁的所有

端口都成功才流出；同时需要保证输入缓冲的容量至少能够容纳一个完整的组播数据报文。

4 性能分析

DMFTAR实现方法是在HFDS-IN单播实现方法的基础上扩展而成的，与单播实现方法相比，其特点体现在以下方面：(1)单播路由表分布式存储在各个结算节点上，而组播转发表分布式存储在各个HRNR芯片上。(2)单播报文中直接携带着每一跳HRNR芯片的输出端口信息，而组播报文间接携带着每一跳HRNR芯片的输出端口信息。(3)单播路由表由路由设计人员根据网络拓扑结构和路由策略生成，而组播转发表由算法根据单播路由表产生。

MHWAR方法每发送一个报文，需要首先在多个头微片中直接指明到目的节点的路径。DMFTAR方法先发送组播控制报文构造组播树，然后通过树结构组播报文。为了简单量化比较MHWAR与DMFTAR两种方法在HFDS-IN网络中的性能，不妨假设组成员节点数目为 N 的组播组通过HRNR互连网络构成规则的组播树(即满树)，且每个HRNR的出度为 m，再假设组中所有节点向其它任意节点组播 M 个报文(即All-to-All通信)，MHWAR方法在组播数据前不需要发送控制报文和组通告报文，但是每个节点发送一个数据报文需要发送的头微片数目为 $N-1$，所以整个All-to-All通信过程中，总共产生的头微片数目为 $M\cdot(N-1)^2$ 个；而在DMFTAR方法中，组播会话的启动节点在组播数据前首先需要向 $(N-1)/m$ 个HRNR发送控制报文以分发组播转发表，接着向其他 $N-1$ 个节点通告组播组信息，然后组内节点可通过一个头微片向组内其他所有节点组播数据，因此整个All-to-All通信过程中，总共产生的头微片数目为 $M\cdot N$。表1总结了这两种方法的性能比较结果。可见，DMFTAR比MHWAR方法通信开销低且扩展性好。

表1 DMFTAR与MHWAR两种方法的性能比较

比较内容	MHWAR	DMFTAR
组播数据时发送的头微片数	$M\cdot(N-1)^2$	$M\cdot N$
组播数据前发送的控制报文(微片)数	0	$(N-1)/m$
组播数据前发送的组通告报文(微片)数	0	$N-1$

而且，由于报文包含的微片数目是受限的，因此当组播规模较大时，MHWAR方法还需要进一步分解组播组，而DMFTAR不存在这样的问题。

结束语 在多机通信系统中，组播是一种可支持多种聚合通信服务的重要操作，因此一直是研究的热点。面向高阶胖树源路由互连网络，提出了一种组播实现方法——DMFTAR。该方法有效利用了该类互连网络的树结构和源路由两大特征，基于单播路由表直接生成组播树，并将组播树分发到网络的相关路由器中，从而构成分布式的组播转发表。同时，该方法在底层采用异步复制组播数据的实现方法，能有效避免死锁问题并降低实现难度。初步的理论分析结果表明，与采用多头微片虫蚀和异步数据复制的MHWAR方法相比，DMFTAR通信开销低且扩展性好。在本文工作的基础上，后续将进一步研究DMFTAR方法的实现问题，例如评估其硬件资源消耗情况，评测其对单播操作性能的影响等。

参 考 文 献

[1] Petrini F, Kerbyson D J, Pakin S. The Case of the Missing Supercomputer Performance[A]// Achieving optimal performance on the 8192 processors of ASCI Q. Proceedings of SC2003, 2003 [C]. New York, USA: ACM, 2003: 1-17

[2] Rabenseifner R. Automatic MPI Counter Profiling of all Users: First Result on a CRAY T3E 900-512[A]// Proceedings of the Message Passing Interface Developer's and User's Conference (MPIDC), 1999[C]. Atlanta, USA: HLRS, 1999: 77-85

[3] McKinley P K, Xu H, Esfahanianm A H, et al. Unicast-Based Multicast Communication in Wormhole-Routed Networks[J]. IEEE Transaction on Parellel and Distributed System, 1994, 5 (12): 1252-1265

[4] Fan K P, King C T. Turn Grouping for Multicast in Wormhole-Routed Mesh Networks Supporting the Turn Model[J]. The Journal of Supercomputing, 2000, 16(3): 237-260

[5] Lin X, Mckinley P K, Ni L M. Performance Evaluation of Multicast Wormhole Routing in 2D-mesh Multicomputers[A]// Proceedings of the Internal Conference on Parallel Processing, 1992 [C]. Piscataway, NJ: IEEE, 1992: 173-178

[6] Panda D K, Singal S, Kesavan R. Multidestination Message Passing in Wormhole k-ary n-cube Networks with base Routing Conformed Paths[J]. IEEE Transaction on Parallel and Distributed Systems, 1990, 10(1): 76-96

[7] 肖灿文，张民选，过锋. 二维环网中基于自适应维度气泡路由的组播算法[J]. 计算机研究与发展，2010，47(2)：353-360

[8] Duato J, Yalamanchili S, Ni L M. Interconnection Networks: An Engineering Approach[S]. Computer Society Press, 1997

[9] Kermani P, Kleinrock L. Virtual Cut-Through: A New Computer Communication Switching Technique[J]. Computer Networks, 1979, 3(4): 267-286

[10] Libeskind-Hadas R, Mazzoni D, Rajagopalan R. Tree-Based Multicasting in Wormhole-Routed Irregular Topologies[A]// Proceedings of the Merged Twelfth International Parallel Processing Symposium and the 19th Symposium on Parallel and Distributed Processing, 1998[C]. Washington, DC, USA: IEEE Computer Society, 1998: 224-229

[11] Ni L. Should Scable Parallel Computers Support Efficient Hardware Multicast? [A]// Proceedings of the 1995 ICPP Workshop on Challenges for Parallel Processing, 1995[C]. Washington, DC, USA: IEEE Computer Society, 1995: 2-7

[12] Top500 supercomputer sites[OL]. http://www.top500.org

[13] Xie M, Lu Y T, Wang K F, et al. Tianhe-1A Interconnect and Message-Passing Services[J]. IEEE MICRO, 2012, 32(1): 8-20

[14] Rajeev S, Craig B S, Dhabaleswar K P. Implementing Multidestination Worm in Switch-based Parallel System: Architectural Alternatives and their Impact[A]// Proceedings of the 24th International symposium on Computer Architecture, 1997[C]. New York: ACM, 1997: 50-61