高效的基于平面的层次化应用层组播树模型

许建真[1,2] 濮松兰[2] 张福炎[1]

(南京大学计算机科学与技术系 计算机软件新技术国家重点实验室 南京 210093)[1]

(南京邮电大学计算机学院 南京 210003)[2]

摘 要 应用层组播在Internet上有着广泛的应用。本文在K-叉树结构的基础上提出了一种K-叉平面结构,同时在平面的组织上结合了分层和分簇的思想。K-叉平面结构降低了子树之间的错误关联度;分层分簇降低了组播树的深度,减少了组播管理上的复杂度;在增加一定冗余的前提下,不同平面的簇与簇之间采用小概率通信,提高了数据恢复和数据传递的效率。整体而言,组播树的健壮性得到了很好的保证。仿真实验结果证明本模型能够很好地对端用户进行管理,提高了因某些结点意外失效造成的数据恢复和应用层组播的转发效率。

关键词 应用层组播,K-叉平面,分层分簇,组播管理

Efficient Plane-based Hierarchical Application Layer Multicast Model

XU Jian-zhen[1,2] PU Song-lan[2] ZHANG Fu-yan[1]

(State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China)[1]

(Campus Computer Network Center, Dept. of Computer, Nanjing University of Posts & Telecommunications, Nanjing 210003, China)[2]

Abstract Application Layer Multicast has been widely used recently. This paper proposes a K-plane construction which is based on K-tree, and uses the thought of hierarchies and clusters in planes organization. K-plane reduces the relation between sub-trees; hierarchies and clusters reduce the depth of the tree and the complexity of multicast in organization; giving the condition of adding some redundancy, different plane's clusters communicate with each other in probability of p, which improves data recovery and the efficiency of data delivery. By all accounts, the multicast tree robustness is well assured. Simulation experiment proves the model can organize the terminal users well, improve data recovery and increase the forwarding efficiency of ALM.

Keywords Application-layer multicast, K-plane, Hierarchies and clusters, Multicast organization

1 引言

随着Internet的发展,组播是互联网研究的重要课题。IP组播是一个非常有效的在路由器上实现"单播,尽力发送"的分发机制。缺点是:它要求所有介于组播源和接收者之间的路由器、集线器、交换机、TCP/IP栈、防火墙均需支持组播,可知实现IP组播对网络层设备有较大的依赖性,需要对组播路径上所有的网络层设备进行调整和升级,为大范围部署IP组播带来了阻碍,所以限制了IP组播的发展。

应用层组播不需要专门的路由器来支撑整个网络数据的分发,是通过端到端的机制实现组播转发功能,而不是基于核心路由器实现组播通信的所有功能。应用层组播最大的优点是:由于是依靠端系统,不需要调整现有的网络部署,具有灵活和易实施的特点,便于推广。所以应用层组播将在实际应用中取代IP组播,成为主流。

由于应用层是在底层IP网络的基础设施之上构建的虚拟、逻辑的网络,其路由和下层的IP网络的路由通常不一致,这就造成了应用层系统的时延增加和资源的浪费,同时由于某些结点突然失效后造成了数据传递的混乱,于是如何提高应用层组播的效率就成了迫在眉睫需要解决的问题。

为了改善各种问题,提出了很多优秀的ALM理论拓扑模型,这些模型可以归类为树优先模型(如Yoid[1]、HMTP[2])、网状优先模型(如Narada[3])、层次结构模型(如NICE[4])。本文提出的是一种基于平面的层次化结构模型。

2 基于平面的模型设计和分析

传统的处理丢失包的方法是通过寻找新的上游结点重新建立连接并传输数据,但是其新的父结点或者祖父结点可能本身就是一个失败的结点。单纯的子结点并不能够在短时间内知道到底是哪个上游结点出了问题,所以在重新传送数据之前子结点需要向上游结点发送多次发送请求,这将导致长时间的恢复延迟。

2.1 构建组播树的基本思想

本算法将整个树分成了K个平面,平面和平面之间的关联度减少。一个平面内结点的错误只会影响到它本平面内的结点,而不会影响到别的平面内的结点,同时,数据的恢复不再单纯地依靠其上层结点,而是以一定的概率接收来自相邻平面的对等层代理传来的数据。在上层结点失效时,通过接收相邻平面对等层代理传来的数据,不仅可以实现快速的数据恢复,同时当本平面内上层代理的数据还没有到达下层代理的时候,可以提供快速的数据传递,更好地实现了数据传递的高效性。

许建真 博士,副教授,主要研究方向为网络与信息安全;**濮松兰** 硕士,主要研究方向为网络安全;**张福炎** 教授,博士生导师,主要研究方向为计算机应用技术。

如图1所示，组播数的构建采用 K-叉树的结构，即源点作为始结点，生成 K 棵子树。由 K 棵子树组成 K 个平面。平面内进行分层，在每一层引入分簇，为避免由于结点增加而导致层次结构过于庞大，不便于管理，本算法定义层内采用2-叉树结构，簇内结点的通信采用环结构。这样避免了由于结点数目过多而导致的树的高度太大，不便于管理的缺点。

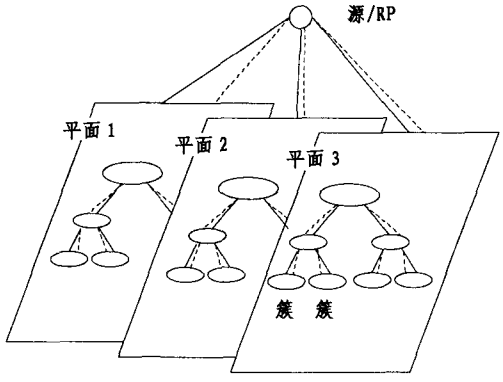

图1 组播树整体框架结构图

在应用层组播中，成员之间的路由路径可能潜在地增加了接收和重接收的延迟。以二叉树为例，当根结点 A 与下游结点 P 通信时，A 位于第0层，P 可能位于第六层或者更低层，花费的代价随树的深度的增加而增加。而如果采用 K-叉平面的话，相对于2-叉树中结点 P 的位置而言，树的深度可以得到明显的改善，同时使管理变得更易于实现。具体深度由簇内的结点数目而定。使用 K-叉平面的好处又表现在增加一定冗余的同时增强了在某些结点失效后保持了树的健壮性，减少了恢复的延迟。

2.2 工作原理

这一节详细地描述了本协议的基本构造和工作原理。

首先，定义这样一个被称为源的结点 RP，该结点不属于任何组播组，而是一台具备很强处理能力的主机。

其次建立一棵以源为根的 K-叉平面，为了减少由于簇内结点过多而导致本层的结点过于庞大，K 的初值的取值是关键。

接着，为了控制信息更快更好地在控制链路上汇聚而不产生高的控制开销，本算法采用了在每个平面内分层的结构，并且在每个层上以簇为单位构建2-叉树，而组播数据分发的传播路径整体选用树作为数据传输路径，本算法中数据传输路径是由控制路径隐性决定的。

每个簇内选取两个结点与上层簇进行数据通信。也就是通常所说的环首和副环首，这里分别称为代理结点和副代理结点。参见图2，第二层中簇 X 中的结点 $A2$ 和 $B2$ 不仅管理本簇内所有的结点，同时受上层簇内代理和副代理的管理和控制。即第一层中的代理 $A1$ 控制第二层中的簇 X、簇 X' 的代理 $A2$、$C2$，并通过代理进行数据传送。第一层的代理 $B1$ 控制第二层的副代理 $B2$、$D2$。簇内所有的结点通过代理实现与下层结点的通信。

代理和副代理的选取不是随机的，是根据该结点的承载能力、持有的数据资源量，在系统上存在的时间，以及到 RP 结点的距离的综合来选取的。该综合能力 Q 的计算见公式(1)，其中 S_i 表示结点 i 持有的资源量；N 表示该簇内所有的结点总数；$Time_i$ 表示结点 i 在系统上存在的时间；L_i 表示结点 i 到 PR 结点的距离；$f(x_i)$ 表示结点 i 承载能力；α、β、γ 为常数。

$$0<\alpha<1,0<\beta<1,0<\gamma<1$$

$$Q=\alpha f(x_i)+\beta\times\frac{1}{S_i}\times\sum_{j=1}^{n}S_j+\gamma\times\frac{1}{Time_i}\times\sum_{j=1}^{n}Time_j \quad (1)$$

$$f(x_i)=\alpha\times Time_i^2+\beta\frac{L_i(L_i-1)}{2\times\sum_{i=1}^{n}(2i-1)} \quad (1.1)$$

综合能力 Q 最强的为该簇的代理，次之的为副代理。每个簇内的代理和副代理需要向根结点进行注册，根结点将这些信息根据区分平面的规则进行保存。

参见表1，源结点对各平面的控制信息采用三元组结构（平面 i，第 j 层，第 K 个簇）。源结点定期向平面 i 发送平面 $i-1$ 和 $i+1$ 的所有代理信息。平面 i 接收到根结点的控制信息后，第 j 层的代理选择平面 $i-1$ 和平面 $i+1$ 的第 j 层的代理信息进行保存，并将来自源结点的控制信息依次下传，直到最后。

表1 RP结点的控制信息

平面 1	平面 2	平面 i	……	平面 k
(1,1,1)	(2,1,1)	(i,1,1)	……	(k,1,1)
(1,2,1)	(2,2,1)	(i,2,1)	……	(k,2,1)
(1,2,2)	(2,2,2)	(i,2,2)	……	(k,2,2)
……	……	……	……	……
(1,j,k)	(2,j,k)	(i,j,k)	……	(k,j,k)

每一个簇内的代理和副代理定期的向 RP 发送刷新信息，若 RP 长期收不到簇的代理的刷新信息，则认为该簇的代理已经自动离开，则副代理自动取代代理的位置，并且该簇重新选择一个副代理。如果是长期收不到来自副代理的刷新信息，则重新选择一个综合能力次强的结点作为副代理。代理结点中不仅包含本簇内的所有结点的信息，同时保存通过 RP 结点定期刷新的相邻对等层的代理的信息。

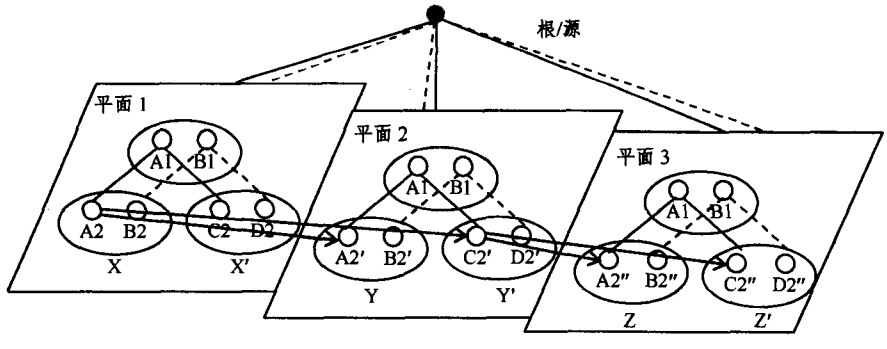

图2 簇的管理和通信

簇内采用 VRing[5]结构，现在的应用层组播技术主要是基于组播树，但是环结构相对组播树有它独到的优点：易于构造；具有内在的可靠性和容错性；所有结点都参与复制和转发数据；同时 ACK 需要量小；环自动提供冗余备份。那么它的缺点是可扩张性差、时延大、实时性差。但是本算法中引入了分层和分簇的思想，正好弥补了可扩张性差的缺点。由于平面之间是以小概率 q 发送数据，同时弥补了时延大和实时性差的缺点。

在图 3 中，建立了一组虚拟边，称为冗余边。结点按照 1-N 的顺序进行标记，组成员 i 按照 $(i+2)\%N$ 的规律建立冗余边。如图 3 中的(1,3)、(8,2)等。数据沿着实边和冗余边转发，沿实边接收到的数据沿实边和冗余边转发，沿冗余边收到的数据沿冗余边转发。

由于在多个结点失效的情况下，还是容易造成簇内一定的丢包率，所以簇的大小也是一个应该考虑的因素。通常情况下，选择 6～10 个结点组成一个簇。

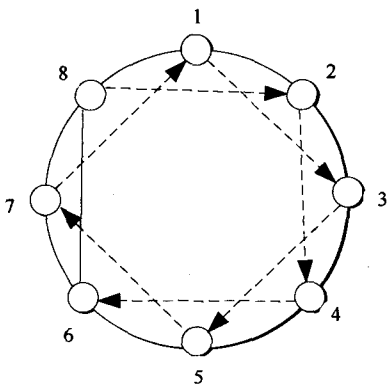

图 3　环结构

2.3　结点的加入

当结点 X 希望加入组播组时，结点 X 首先发 Request-join 信息给称为源的结点主机 RP。RP 根据一定的运算规则选择离结点 X 跳数最近的那个平面，并回传信息给结点 X。结点 X 通过与平面内簇中的代理结点匹配来选择合适的结点作为自己的代理结点，并加入到该簇中。

1. 若该簇已满，则继续向下匹配，直到找到合适的簇并加入之。

2. 若没有合适的簇，则在该平面内建立一个新的簇 α。

3. 若新簇 α 所在的层已经存在，则与上层代理建立连接。否则，执行 4。

4. 如果该簇 α 所在的新层的深度 H 满足 $H\leqslant 3K-1$，(K 是平面的个数)则与上层代理建立连接。否则，执行 5。

5. 将该平面内的深度为 H 的最后一个簇 β 与上层代理断开连接，连同该簇所组成的二叉树结构一起重新建立一个新的平面，簇 α 加入到新平面内。

平面分裂的优点在于保证树结构的深度不会随着结点的增加而无休止地增加下去，同时保证了树结构的健壮性。

2.4　结点的失效处理

为了保证因为某些结点的意外失效而造成子结点通信中断，同一平面的同一层以概率 p 在所有的代理间选择一个代理，向邻近平面的对等层的代理以小概率 q 发送数据。

如图 2 所示，平面 1 以 $A2$、$C2$ 作为第二层簇 X 和 X' 的代理，平面 2 以 $A2'$、$C2'$ 作为第二层簇 Y 和 Y' 的代理，平面 3 以 $A2''$、$C2''$ 作为第二层簇 Z 和 Z' 的代理。显然，簇 X、X'、Y、Y'、Z、Z' 位于对等层上。

1. 在同一层的三个平面内各以概率 P 选择一个代理用来与不同平面的对等层的代理进行通信。以第二层为例，假定代理 $A2$、$C2'$ 和 $A2''$ 分别是以概率 P 选出的代理。

2. 对等层的代理分别以小概率 q 进行数据传送。如图 2，$A2$ 分别以概率 q 向平面 2 的簇 Y 和 Y' 的代理 $A2'$ 和 $C2'$ 转发数据，平面 2 中的 $C2'$ 以概率 q 向平面 1 和平面 3 的代理 $A2$、$C2$ 和 $A2''$、$C2''$，转发数据，同理 $A2''$ 以概率 q 向平面 2 中的代理 $A2'$、$C2'$ 转发数据(图中只画出了部分冗余线)。

3. 当上层的代理或者副代理由于意外失效或者拥塞不能及时地向下层代理发送数据时，此时如果下层的代理能够收到来自不同平面的对等层的代理发来的数据，则可以继续向下层和本簇传递信息，极大地减少了由于等待恢复而产生的延迟。同时如果先于上层代理接收到数据，则将加快了数据的传输效率。如果发现数据已经接收到，则为冗余数据，自动丢弃。

3　仿真实验及性能评价

3.1　分析

为了客观地评价本算法相对于其余算法的优劣，引入了如下的参数来考察：

1. 深度

树的根或者源结点暂且算作树的第 0 层。随着 K 的取值不同，树的深度不同。树的深度的计算公式为：

$$(2^n-1)\times M\times K=N-1 \tag{2}$$

$$n=\log_2^{\frac{N-1}{K\times M}+1} \tag{3}$$

其中 n 表示深度；M 表示每一个簇中结点的数目；K 表示平面的个数；N 表示所有结点的数目。当 K 和 M 取值一定时，随着结点数目的增加，深度越深，这种增长的速度却越来越慢，趋于平稳。

2. 冗余

增加的冗余链路越多，需要维护的状态信息就越多，结点的处理负担也就越重，越不利于系统的可扩展性。采用如下公式：

$$\Delta=\frac{N_{rd}}{l} \tag{4}$$

Δ 表示增加的冗余链路的数量 N_{rd} 与总链路数量 l 的比值。这里环中的链路和冗余暂且不考虑，仅从整棵树的结构来考虑。

由图 2 可以知道，在每一层不同的簇间选择一个代理的概率是 p，n 表示树的深度。则整个平面间拓扑需要建立的冗余链路 t 为：

$$p\times 2^{K-2}(2^{n-1}-1)\leqslant t\leqslant p\times 2^{K-1}(2^{n-1}-1)$$
$$K=2,3,4\cdots \tag{5}$$

整个链路需要建立的链路数 l 为：

$$l=K\times 2\times(2^n-1) \tag{6}$$

$$\theta=\frac{N_{rd\_without(i)}}{N} \tag{7}$$

θ 表示在有冗余链路时，由于 i 个结点同时意外失效而造成的不能接收组播信息与总结点数的比值。当结点数 N 一定时，用户希望 $N_{rd\_without}(i)$ 越小越好。理想状态下，不能接收组播信息的结点数为 i。同时越小也就说明了本算法构建的组播树的健壮性高，不会因为某些结点的突然失效而导致混乱。

3. 延迟

根据构建组播树方法的不同，当一个结点意外失效时，可以有多种不同的方法恢复失效结点的子结点的正常通信。传统的方法是失效结点的子结点多次发送重传请求。传统结点恢复延迟计算为：假设结点 P 是结点 N 的父结点，当结点 P 意外失效时，结点 N 恢复正常数据传送时的延迟为：

$$\begin{cases} delay((N),par(p)) < delay(P,par(P)) \\ deg\,ree(N) > deg\,ree(p) \\ gain = \dfrac{delay(P,par(P)) + \sum_{x\in child(P)}(1+N(x)) * delay(x,P)}{delay(N,par(P)) + \sum_{x\in child(P)}(1+N(x)) * delay(x,N)} \end{cases} \tag{8}$$

$N(x)$代表结点 x 的子嗣结点数目，而 $par(p)$代表 p 结点的父结点。

本算法中，$par(p)$由对等层代理替换，即为 $proxy(p)$。

3.2 仿真实验及性能评价

本算法采用 NS-2 进行模拟，拓扑结构采用 Transit-Stub，在仿真实验中，采用统计的手段来统计上述各个参数值，进而比较各个算法策略的优劣。步骤如下：

在仿真实验中，逐步增加结点的规模。参与组播组实验的结点依次从 0 增加到 600 个，结点以每次增加 50 个的数目增长。假定 $p=3$，$q=0.5$，每个簇内的结点选择 8 个。根结点算第 0 层。

图 4 的仿真实验是用户结点到 RP 结点深度的累积分布。易知随着结点的增加，Narada 协议中 K-叉树的深度不仅比 K-叉平面的深度来得深，而且 K-叉树增长的幅度比 K-叉平面来得快。随着结点数目增加，极大地方便了组播树的管理，同时提高了数据传递的效率。

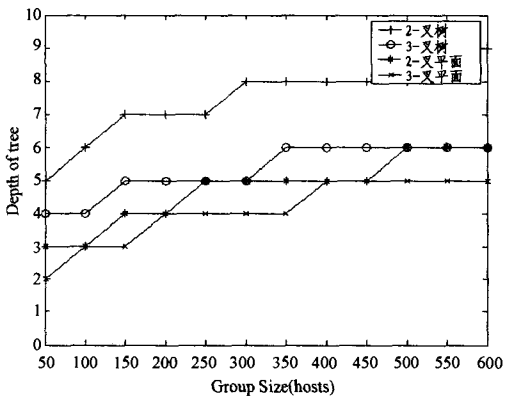

图 4 树的深度的变化曲线

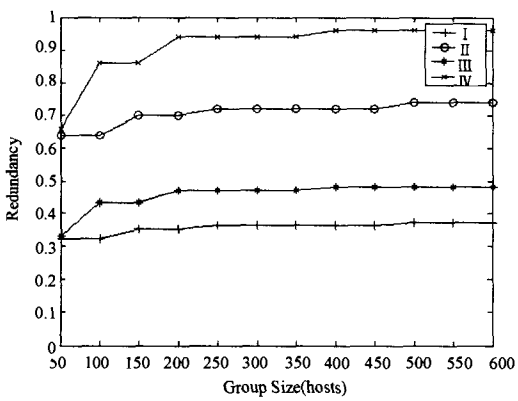

图 5 冗余度的变化曲线

图 5 的仿真实验统计地评价了随着结点的增长冗余链路与原始链路的比值。以 2-叉平面和 3-叉平面为例，2-叉平面的冗余度介于 I 和 II 之间，3-叉平面的冗余度介于 III 和 IV 之间。由图 5 的比较中可以直接的看出，随着 K 值的增加，冗余度增加。2-叉平面时介于 32%～75%左右，随着结点的增加还成小幅度的增长。3-叉平面时冗余度有所上升。

图 6 的仿真实验描述了在增加冗余的情况下，在不同的组成员规模时，以两个结点同时失效为例(假设根结点不产生失效)，不能够收到信息的结点数目与总结点数目的比值。在结点总数为 N 时，随机选取 $\lfloor\sqrt{N}\rfloor$ 次结点失效后，数据恢复延迟的平均值作为采样点($\lfloor\sqrt{N}\rfloor$表示根号取整)。易知，本算法在增加了一定的冗余链路后，数据恢复延迟明显地降低了，在两个结点同时失效时与理想水平也相当接近。

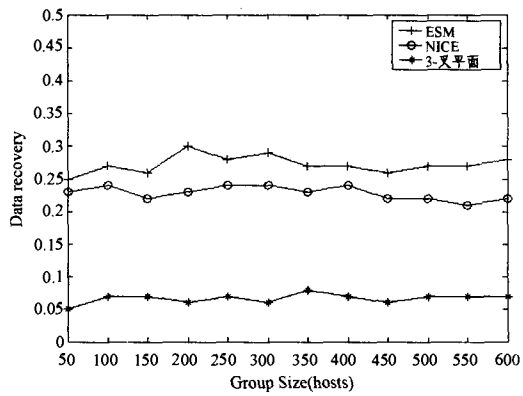

图 6 两个结点失效时平均数据恢复延迟变化曲线

结束语 本文所介绍的基于平面的层次化组播树算法能够很好地解决因组播网络频繁变化而导致的多点意外失效而造成的延迟。该算法将单一的拓扑树映射成一个基于平面的组播树，降低了单纯平面之间的关联度，同时平面之间通过代理建立冗余，并且以小概率传送数据，不仅降低了延迟，同时提高了数据传输的效率。在今后的工作中，会致力于降低算法的冗余度，使性能得到更好的提高。

参 考 文 献

[1] Yoid F P. Extending the Multicast Internet Architecture, 1999. White paperhttp://www.aciri.org/yoid/

[2] Zhang B, Jamin S, Zhang L. Host multicast: A Framework for delivering multicast to end users//Proc. of IEEE infocom. vol. 3, June 2002: 1366-1375

[3] Chu Y-H, Rao S G, Zhang H. A Case for End System Multicast //Proceedings of ACM SIGMETRICS. 2002, 20(8): 1456-1471

[4] Banerjee S, Bhattacharjee B, Kommareddy C. Scalable Application Layer Multicast // Proceedings of ACM Sigcomm. Aug. 2002: 205-217

[5] Sobeih A, Yurcik W, Hou J C. VRing: A Case for Building Application-Layer Multicast Rings [J/OL] (MASCOTS'04) 1526-7539/04 2004 IEEE

[6] Yoid F P. Extending the Multicast Internet Architecture, 1999. White paperhttp://www.aciri.org/yoid/

[7] Wong K-FS, Chan S-H G, Wong Wan-Ching, et al. Lateral error recovery for application-level multicast [J/OL]//Proceedings of INFOCOM. vol. 4, March 2004: 2708 - 2718

[8] Zhang B, Jamin S, Zhang L. Host multicast: A Framework for delivering multicast to end users//Proc. of IEEE infocom. vol. 3 June 2002: 1366-1375