可靠组播通信中可扩展性问题的研究*)

An Approach of Extending Scalability of Reliable Multicast Protocols

付培荣 李 冀 魏明亮 乐正宜 陈贵海 谢 立

(软件新技术国家重点实验室 南京大学计算机系 南京210093)

Abstract The current prosperity of Internet multicast applications has been coming up under the push by the development of multicast technology and the pull by user requirements. The key issue on the scalability of reliable multicast protocols based on the retransmission error control mechanism is the reduction of control packets and retransmitted data packets. In this paper we discuss the key issues and introduce NAK-Election algorithm and NAK-Absorption Algorithm respectively as the solution to the problems. We thereby devise a reliable multicast protocol-RMTP. The RMTP, implemented under our NAK-Election and NAK-Absorption algorithms, exhibits a greatly enhanced scalable property as a result of the test shown in the last part of the paper.

Keywords Multicast, Scalability, Error control, RTT

1 引言

IP Multicast 技术，特别是 MBONE (Multicast Backbone)的发展，使得 Internet 上需要多目传输支持的应用，如视频会议，成为可能。可扩展性是组播协议研究的一个重要方面，它决定了一个协议在组规模基础上的可扩展能力与协议的适应性。

可扩展性包含的问题非常广泛，本文主要讨论基于重传的差错控制的可靠组播协议。这类协议的差错控制方法是影响可扩展性的主要因素。在差错控制方面，由于恢复时延、控制报文和重传数据数量随着组中成员增加而成倍上升，如何减少控制报文和重传数据成为保证可扩展性的关键问题。具体地说，由差错控制机制引起而影响可扩展性的问题可以归结为：(1)重传请求风暴。主机性能的不断提高，相比之下网络性能成为了性能的瓶颈。子网内的丢包很大程度上是网络拥塞所引起的，当子网内众多的节点同时发生这种情况时，就出现所谓的重传请求风暴，并且这个问题会随着组播的规模扩大迅速上升。这是传统可靠组播协议可扩展性差的重要原因之一。(2)重传报文数量。在传统的方案中，重传报文的数量在很大程度上取决于重传请求报文的数量，当组播规模较大时，重传请求风暴也会带来重传报文数量剧增，甚至导致网络拥塞。(3)恢复时延。当组播规模较大时，恢复延时会相对变长，严重影响协议的性能，甚至使协议不可用。在可靠组播协议中，提高协议可扩展性的处理方法往往会增加恢复时延(如随机滞后方案)，很难找到一个合适的方案，既能减少重传请求报文，又让恢复时延不受到影响。

针对可靠组播协议中影响可扩展性的三个问题，我们提出了一套可靠组播协议的差错控制方案，方案中的 NAK Election 算法和 NAK Absorption 算法分别解决重传请求风暴和重传报文数量，在本方案中恢复时延也得到了相应解决，从而有效地提高了可靠组播的可扩展性。

2 相关工作

在规模比较大的组播应用里，为了提高组播协议的可扩展性，差错控制通常主要采用两种方法：随机滞后方案(randomized backoff schemes)和分层方案(hierarchical schemes)。下面讨论这两类中可靠性的实现对可扩展性的影响。

2.1 随机滞后方案的可扩展性

使用这个方案的典型组播协议有 Scalable Reliable Multicast(SRM)。SRM 通过 NAK 抑制办法来减少 NAK 报文数量，它在设计抑制重传请求定时器时，定时器长度通常都是单播情况下接收方到发送方延时的数倍，因此，它的恢复时延比单播高很多。

一些模拟试验表明，在随机的网络拓扑结构上使用固定定时器长度，SRM 通常需要3倍于单播情况下 RTT(Round Trip Time)的时间来恢复报文，处理过程

*)本文研究得到国家863高科技项目"实用化的 JAVA 开发环境的开发"(863-306-ZT02-03-01)的资助。

中产生了5-10个重复报文。因此，如果采用动态自适应的办法调整定时器长度，将在定时器长度调整好后显示出更好的性能，有助于减少重复报文的数量。SRM没有讨论在收到重复多个NAK报文的情况下如何减少恢复数据报文的数量。

2.2 分层方案的可扩展性

在分层方案中，所有成员组织成一个树形层次结构，每一个成员都有一个父节点和若干子节点。所有成员的重传数据请求都发向其父节点，父节点负责重复报文的合并。分层方案根据树形层次结构，有动态和静态两种。

Rehable Multicast Transport Protocol(RMTP)采用静态的分层方案，它在接受者中间指定若干接收者负责发送ACK和数据恢复工作，以期减少发送方的处理负载。重传数据根据重传请求的数量采用组播或者单播实现，但这种方法只有在丢包很严重或者很少的情况下才体现出优越性，否则反而会带来很大的开销。

Tree-based Multicast Transport Protocol (TMTP)是一个动态分层方案，其中每一个区域都有一个域管理者。当新的域管理者要加入时，它通过特定算法(Expanding Ring Search)寻找一个存在的域管理者作为它的父节点，每一个接收者记录自己和域管理者的距离，域管理者记录和自己距离最长的接收者的距离，这些参数是域管理者用来设置报文的TTL值，限制报文能传播的范围，从而防止重传请求风暴，以增加它的可扩展性。虽然这种动态方案带来了一定的灵活性，但给控制带来很大的复杂性，此外，根据它寻找父节点的算法，找到的有可能不是最近的接点，这样就会增加它的报文恢复时延。

3 差错控制方案

我们设计并实现了一个组播协议的差错控制方案。在实现可靠性方面，我们采用了分层模型和随机滞后模型相结合的方法，由接收方负责丢包检测和重传数据请求抑制，中间节点负责状态维护和重传。为了增加协议的可扩展性，我们提出了接收方的NAK Election算法和发送方的NAK Absorption算法，NAK Election算法负责抑制接收方重复的NAK报文，重传数据请求；而发送方的NAK Absorption算法负责合并已经发送的重复的NAK报文，减少重传数据的重复传输。

3.1 重传请求风暴的抑制

重传请求风暴的防止是在接收方通过NAK Election算法完成的。NAK Election算法的目的是减少NAK报文数量，防止重传请求风暴。

NAK Election算法，是指在每个接收方检测到丢包时，并不马上发送NAK报文，而是延迟一个随机的时间：NAK ElectionTimer Interval。在NAK ElectionTimer期间，如果收到(其他接收接点发来的)NAK报文，且收到的NAK报文请求的数据报文就是自己所要请求的数据报文，就停止NAK ElectionTimer，等待发送方发送恢复报文；否则，如果NAK ElectionTimer超时，就自己发送NAK报文，进而抑制其它节点发送相同的NAK报文，起到了避免重传请求风暴的作用。

在此，我们必须考虑这种情况：两个机器同时发生了定时器超时，于是这两个机器分别抑制了对方发送NAK报文，这样就只有几乎同一个时刻的两个NAK报文，并导致后续的NAK得不到发送。为解决这个问题，我们采用下述策略：一旦一个机器发送了NAK，就让它继续发送下去，直到收到恢复报文。这样，在NAK Election阶段(从某个接收方发现丢包开始，到根据上述算法产生一个或多个NAK的发送者为止的时间内)获得胜利的节点都将按照报文恢复策略发送它们的NAK报文，而不能进一步减少这种报文。这就是说，我们的NAK Election算法只在Election阶段进行NAK报文的抑制。我们称之为NAK发送坚持。

本算法中，两个重要的参数对算法有重大的影响，一个是随机时延长度：NAK ElectionTimer Interval；另一个是两个间隔最短的随机时延之间相差的长度。这两个参数不是孤立的，而是密切相关的。随机时延长度影响恢复时延和重传请求报文的数量，随机时延太长，导致恢复时延的增加；太短，导致重传请求报文数量的上升。随机时延之间相差的最短长度影响重传请求报文的数量，太短(小于两台机器之间的网络时延)，会导致重传请求报文数量的上升；太长(超过两台机器之间的网络时延太多)，会导致恢复时延的增加或者重传请求报文数量的上升。所以，恰当地选择这两个参数是本算法中的关键问题。下面我们介绍二者的取值方法。

接收方随机数取值范围的确定：

1)随机数间隔。每一个随机数产生后，或者相等，或者它们之间相差RTT/2的整数倍。一台机器能够成功抑制另一台机器的NAK报文的条件是：两台机器M1，M2，分别有定时器T1，T2，定时器长度分别为I1，I2，假设机器M1定时器T1在t1时刻超时，此时NAK被M1发送，当M2收到这个NAK报文应该是t2时刻，$t2 = t1 + RTT/2$，只有M2的定时器长度$I2 > I1 + RTT/2$才行，否则，M2上的定时器T2也要超时，导致NAK碰撞。

2)取值区间。假设一次通讯中有n个接收者，接收者之间彼此的回程时延为RTT，随机数取值范围是

[0,L]，那么 L 的取值必须满足 L=RTT/2 * n，这个区间的宽度正好是机器的数目，随机数产生算法，随机数的分布情况也影响到了 L 的取值。我们的随机数产生采用著名的线性同余算法和48位整数算术运算来产生，它产生的是一个在一定区间均分布的整数。因此，每一台机器产生的在区间[0,L]的随机数的概率是均等的。我们有理由认为 n 台机器产生的随机数正好均匀分布在这个区间内，没有冲突，就没有报文重复。假如区间增大，就让恢复时延增加，但没有让 NAK 的数目更小，因为此时已经最小了；假如区间减小，就会让 NAK 数目增加，但同时又没有缩小时延，因为此时的时延已经最短了。

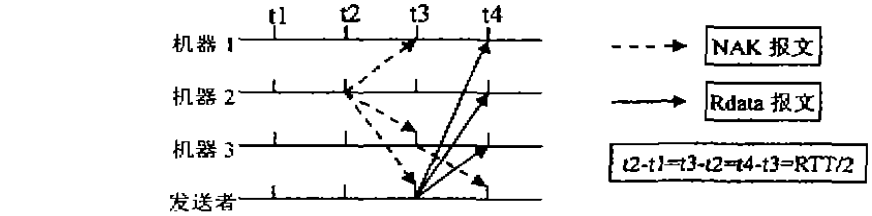
3.2 数据重传的合并

数据重传的合并是在接收方完成的。发送方的 NAK Absorption 算法的思想是：在发送方收到 NAK 报文时，设置发送方状态，启动 NAK 合并定时器，同时发送恢复报文；在 NAK 合并定时器期间，发送方将忽略对该恢复报文的数据重传请求，从而减少重传数据的不必要重传。在一个 RTT 时间内重复的 NAK 报文意味着在一个 RTT 时间内重复的重传数据报文，而这是不必要的。我们的 NAK 合并算法中的合并定时器长度的确定方法保证了在一个 RTT 时间内只有一个重传数据报文。

发送方 NAK 合并算法的合并定时器长度是本算法中最重要的参数，其确定方法如下：假如发送方接收到多个请求同一个数据报文的 NAK 报文，一定是有多个接收者的 NAK ElectionTimer 同时超时，所谓机器1和机器2同时超时，指的是机器2的 NAK ElectionTimer 在机器1的 NAK 报文在到达机器2之前超时，所以发送方的 NAK 合并算法的合并定时器长度最长只要2 * RTT/2就可以了。这样，发送方只要保持状态2 * RTT/2的时间就可以，采用这种较短的定时，不仅有利于减少中间节点的状态维护负载，而且可以降低内存的占用率。

3.3 算法分析

NAK Election 算法，极大地减少了多个接收方同时发生丢包情况下子网内的 NAK 报文数量；NAK 合并算法，进一步合并了多个接收方同时超时而发送的 NAK 报文，减少了子网内的恢复报文；二者结合使用，大大提高了可靠组播协议的可扩展性。我们采用的随机数产生器产生的随机数是在[0,MAX]均匀分布的，所以在多个接收者中，NAKElectionTimer 的长度为0的概率将随着接收者的数量上升而上升。这样，因为 NAK ElectionTimer 带来的时延就会变短，而在发送方，NAK 合并算法并没有给报文恢复带来额外的时延，所以，总的来说，协议的报文恢复时延是很短的。算法的图示见图1。



t1 时刻机器 1、机器 2、机器 3 都检测到了丢包，于是分别启动了各自的 NAK ElectionTimer，机器 2 的 NAK ElectionTimer 在 t2 时刻首先超时，并发送了 NAK，机器 1 的 NAK ElectionTimer 至此被终止，而机器 3 在收到机器 2 的 NAK 前也超时了，并发送了 NAK，发送者在 t3 时刻收到了机器 2 的 NAK，发送了恢复报文，在 t4 时刻机器 1、2、3 都收到了恢复报问，发送者在 t4 时刻收到了机器 3 的 NAK，由于发送方的 NAK 合并算法，发送方将不发送恢复报文。

图1 NAK 抑制与合并算法示意

4 性能评测

我们用 Java 实现了一个可靠的组播协议 RMTP，并针对在实现可靠性问题对可扩展性所可能产生的影响做了相关测试。考虑到 Java 本身性能对协议性能可能带来的影响，我们在实现与网络有关的部分和定时器有关的部分时使用了本地方法。测试结果表明我们的工作有助于提高协议的可扩展性。

测试环境：10M 和 100M 的局域网络，RS6000/AIX，SUN Solaris，Linux On Sparc，Linux Red Hat On PC。

随机时延太短，将增加 NAK 报文的数量，加重发送方的负载，随机时延太长，又会增加数据的恢复时延，降低服务质量。所以，在实际的应用中，使用固定区间 NAK Election 的随机时延区间不是一个好的办法，根据我们对随机时延区间的确定和实际的经验，我们

设计了一个根据组成员数目实时产生NAK Election随机时延区间的算法，我们称之为区间自适应，从而能够兼顾两者，使协议更具有适应性，特别是组成员变化较多的应用。

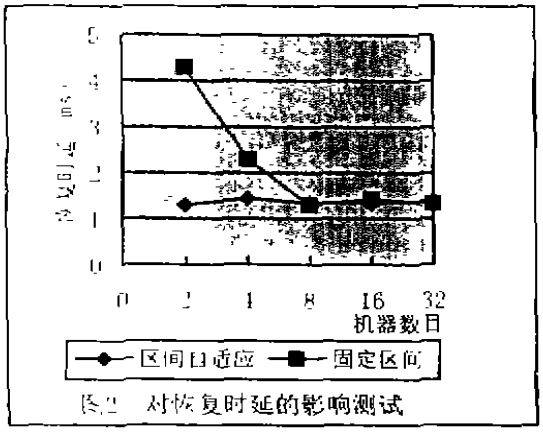


图2 对恢复时延的影响测试

在图2中，固定NAK Election的随机时延区间为[0，MAX]，可以看出随着接收者数目的增加，NAK Election算法对NAK报文抑制的效果以及对恢复时延的影响。

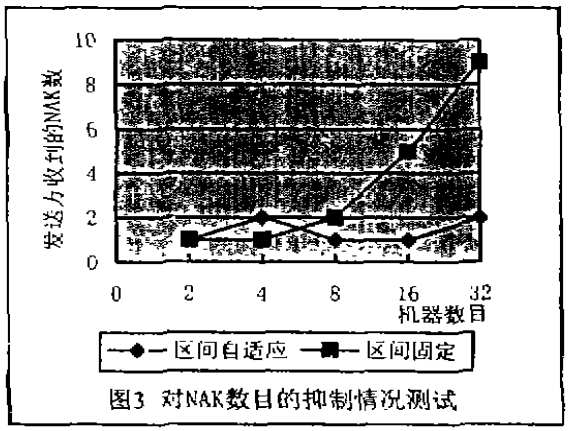


图3 对NAK数目的抑制情况测试

RTT取0.5ms，固定区间情况下取区间最大值4ms，正好是8台机器的自适应区间长度。从图上可看出，机器数目较多时，可减少恢复时延，但重复的NAK报文数量也增多。若机器数目比较少，重复的NAK报文比较少，但恢复时延会较大。

在图3中，根据接收者数目的多寡动态调整随机时延区间，可以看出随着接收者数目的增加，NAK Election算法对NAK报文抑制的效果以及对恢复时延的影响。

试验结果显示，通过这两个算法的结合，有效地减少了NAK报文的数量和重传数据报文的重发次数，提高了可扩展性，又不失低恢复时延这个优点。事实上，在算法足够好的情况下，我们实现了零恢复时延（零额外时延）。

我们的可靠组播协议用Java实现，但是由于Java线程无法做到真正的平台无关性，在Java线程的调度问题上存在很大的问题。另外，使用Java很难做一个很精确的定时器，所以我们使用了Java的本地方法技术实现的有关部分。定时器比较精确，性能也得到了很大的提高。

结论与今后工作 本文深入讨论了差错控制对组播协议的可扩展性问题的影响，并给出了具体解决方案。在性能测试中也体现了较好的效果，有效抑制了NAK报文，减少了重传数据报文，同时又兼顾低恢复时延。

本文主要讨论了在一个子网内的算法，而对于多个子网，需要对中间节点做一些改进，如本地恢复，重传状态维护，避免不必要的信息扩散到不相关的子网，从而进一步提高可扩展性。

参考文献

1 Floyd S,et al. A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing. IEEE/ACM Transactions on Networking,1997,5(6)

2 Deering S. Host Extensions for IP Multicasting. RFC 1112,January 1989

3 Lin J,Paul S. RMTP: A Reliable Multicast Transport Protocol. In:Proc. of IEEE Infocom,1996

4 Holbrook H,Singhal S,Cheriton D. Log-Based Receiver-Reliable Multicast for Distributed Interactive Simulation. Proceedings of ACM Sigcomm' 95,1995,25(4):328～341

5 Smith J M,et al. Activating networks:a progress report. Computer,1999,32(4):32～41

6 Nonnenmacher J,Biersack E,Towsley D. Parity-based loss recovery for reliable multicast transmission. In:Proc. ACM SIGCOMM'97,Cannes,France,Sept 1997

7 Holtmann M. A generic concept for large-scale multicast. In B Plattner,ed. Proc. Int Zuerich Seminar,volume 1044 of LNCS,Springer Verlag,1996 95～106

8 Towsley D,Kurose J,Pingali S. A comparison of sender-initiated and receiver-initiated reliable multicast protocol. IEEE Journal on Selected Areas in Communications,1997