网络编码用于无线网络流媒体多播的优势分析

韩 莉[1,2] 钱焕延[1]

(南京理工大学计算机科学与技术学院 南京 210094)[1] (安徽大学计算机科学与技术学院 合肥 230039)[2]

摘 要 将网络编码与基于NACK的反馈机制相结合,提出了一种以端到端方式运作的轻量级可靠传输协议。不同于NORM协议基于时间片的反馈机制,本协议应用反馈轮(feedback round)机制实现NACK抑制和NACK积累,预防NACK风暴,防止抖动。实验表明,在最大发送速率为2Mbps、组大小为128的设定下,其有效吞吐量是同类NORM协议的1.5倍。在此基础上,协议以网络编码代替FEC作为差错恢复机制,网络编码具有更大的编码空间,在无线网络高丢失率的情况下,具有更好的补偿特性,更适应流媒体传输时延敏感的特点。

关键词 网络编码,无线网络,流媒体,反馈轮,多播

Benefits of Network Coding for Live Streaming Multicast over Wireless Channel

HAN Li[1,2] QIAN Huan-yan[1]

(School of Computer Science & Technology, Nanjing University of Science & Technology, Nanjing 210094, China)[1]

(School of Computer Science and Technology, Anhui University, Hefei 230039, China)[2]

Abstract For wireless channel is inherently lossy, it is challengeable to maintain the quality of service(QOS) for live streaming. We proposed a well-designed light-weighted negative-ACKnowledgment(NACK) oriented reliable multicast protocol, which combines the NC-based repair in its design. NORM repair process uses time slot as the base mechanism, which results in long repair cycle. In our protocol, feedback round algorithm is applied to optimize the repair processing. Benefits of network coding for live streaming multicast over wireless channel was also analyzed in this paper. Experiment results show that the payload throughput of our protocol is about 0.5 times greater than that of NORM.

Keywords Network coding, Wireless channel, Live streaming, Feedback round, Multicast

相对于有线网络,无线网络具有高误码率和高丢失率的特点,且无线网络中存在大量计算能力受限的弱终端,它们的纠错能力和缓存能力都较弱,所以对于包的缺失非常敏感。

流媒体传输具有时延敏感的特点,发送端到接收端超过几百毫秒的包基本就没有用处了,尤其在接收端数目较大的情况下,既要保证时延需求,又要完成较完整的接收,必须借助有效的差错控制机制来保证无线网络上的传输质量。

可靠多播协议机制的研究比过去对TCP传输机制的研究要困难得多。到目前为止,虽然已经提出了许多协议和模型,但仅有NORM[1]协议成为了因特网标准。NORM基于NACK与FEC来保证可靠传输,并针对流量控制、拥塞控制和端到端时延提出了解决方法。其既能够处理发送端与接收端之间对等的多播选路,也能够处理发送端与接收端之间的异构连接。但NORM协议基于时间片的反馈机制会引起较长的反馈周期,不适合无线网络上流媒体的多播传输。

本文以网络编码作为无线网络上可靠多播协议差错机制,将网络编码与基于NACK的反馈机制相结合,提出了一种以端到端方式运作的轻量级可靠传输协议。协议应用反馈轮(feedback round)机制,实现NACK抑制和NACK积累,预防NACK风暴,防止抖动。实验表明,在最大发送速率为2Mbps,组大小为128的设定下,吞吐量是同类NORM协议的1.5倍。在此基础上,本协议以网络编码取代FEC作为纠错机制,取得了更好的网络特性。

本文重点介绍了反馈轮机制,并在实验部分对反馈轮机制及网络编码用于无线网络流媒体多播传输的优势进行了验证及分析。

1 差错控制机制

1.1 基于否定确认的反馈机制

本文采用基于否定确认(NACK)的反馈机制[1],这是一种由接收者负责包丢失检测的模型,即只反馈未接收到的包信息,这样在丢失率不高的情况下就相对节约了网络带宽、减轻了发送端的负担以及减轻了"ACK风暴"的问题。然而当丢失率较高、多播组很大时,同样可能带来NACK风暴。我们使用了"反馈轮"机制来防止NACK风暴,取得了较好的效果。

1.2 基于网络编码的差错恢复算法

网络编码过程一般基于随机线形编码[2],随机线性编码的运算一般定义在Galois限域GF(2^8)上[5]。设$x=x_0\cdots x_{k-1}$为原始数据,一个网络节点可以在GF(2^8)域中独立地选择一组

本文受安徽省自然科学基金项目(11040606Q07)资助。

韩 莉(1975—),女,博士生,讲师,主要研究方向为计算机网络技术及应用;钱焕延(1950—),男,教授,博士生导师,主要研究方向为计算机网络技术及应用。

随机编码系数 $c_1, c_2, \cdots, c_n$，生成一个 k 字节的编码块 χ：

$$\chi=\sum_{i=1}^{i=n} c_i b_i \tag{1}$$

由于每个编码块 χ 都是原始块的一个线性组合，因此编码块可以由线形组合中的参数集唯一确定。一个节点接收到 n 个线性无关的编码块 $\chi_1, \chi_2, \cdots, \chi_n$，就可以进行解码。

令 $\chi=[\chi_1, \chi_2, \cdots \chi_n]$，首先，根据每个块 b_i 的系数，构造一个 $n*n$ 阶矩阵 C，C 的每一行对应于一个编码块的系数。然后利用如下运算计算出原始数据 b，$b=[b_1, b_2, \cdots, b_n]$：

$$b=C^{-1}\chi^{\mathrm{T}} \tag{2}$$

我们使用渐进解码算法[4]，即不是等到 n 个网络编码包完全接收后才开始解码，而是每接收到一个网络编码包即进行解码，从而把解码时间分解到等待接收编码包的过程中去。

2 协议描述

本协议采用基于 NACK 的反馈机制，并采用网络编码作为差错恢复机制，实现了流媒体在无线网络上的可靠多播。

2.1 数据的表示方式

我们把流媒体数据分割成包，以包作为传输的基本单位。每个包的大小约为 1024 字节，在一个会话中，每个包有唯一的序列号。编码算法以组为单位，每个组含有 n 个包，n 越小，修复时延越短，更适合流媒体实时性的要求，我们设定 n 最大为 128。

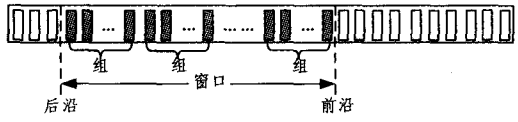

图 1 窗口的构成

发送端和接收端各维护一个滑动窗口，滑动窗口以组为单位。在一个会话中，每个组有唯一的组号。图 1 给出了接收缓存及发送缓存的构成，阴影部分为滑动窗口，其余部分为缓存尚未使用的部分；窗口中组的个数越多，表明延迟越大；为适用流媒体传输的要求，窗口总是向前滑动，发送端不处理对窗口之外的查询请求。

2.2 发送端描述

发送端的任务是从应用层接收数据，并按协议确定的发送速率 R 发送数据。

当发送完一组原始数据包时，发送端将根据当前的包丢失率 p，生成并发送 $n\times p$ 个网络编码包。同时，发送方还负责根据接收方的反馈发送补偿网络编码包，补偿包的个数为 $m\times(1+p)$，其中 m 为接收端请求的补偿包的个数。

2.3 接收方描述

接收端负责从网络接收数据，并将数据输出到应用层。同时，接收端还以 NACK 响应发送端的查询包，通告本节点的接收情况，辅助 GRTT 和包丢失率的统计。

3 反馈轮机制

不同于 NORM 协议，反馈轮算法中，NACK 请求不是由接收端自由发起的，而是在发送端的控制下完成的，轮构成了发送端与接收端、接收端与接收端之间的有效同步信号。

反馈轮(feedback round)算法的原理为：在发送端将时间分为连续的时间段，每段称为一个轮，用唯一整数(称为轮号)标识，相邻轮的轮号是连续的；两个轮次之间的时间与当前的平均往返时间 GRTT 相关，轮的长度的选择应该有助于发送端在一轮时间内接收到大部分接收端对本轮次的反馈，本文取两倍的 GRTT 时间。

发送端在每个轮次广播修复查询信息，接收端仅在接收到查询包时才以 NACK 包响应；响应前，接收端需要等待一段回退时间，从而实现 NACK 抑制。

图 2 给出了反馈轮算法的图示。发送端在一轮的开始发送一个查询包 REQUEST，REQUEST 包通告了本次轮次的 roundID、本轮次的查询时间戳 sendTime，及当前发送窗口中 NC 组的最大组号。

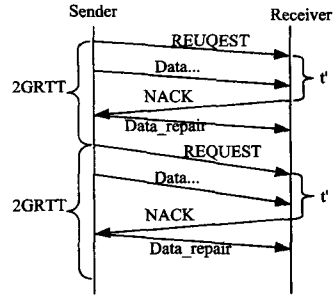

图 2 基于反馈轮的反馈算法

接收端接收到查询包后，等待一段随机回退时间 t'，t' 满足消减指数分布；在随机回退时间结束后，接受端统计当前各个 NC 组丢失的包数，将其写入 NACK 应答，同时还回填了发起此次反馈的反馈轮次 roundID、查询时间戳 sendTime+t' 及当前的丢失率 p。

发送端丢弃携带旧 roundID 的 NACK 请求，仅对当前轮次的 NACK 请求进行累积。

为减少延迟，发送端不会在一轮结束的时候才发送修复包，而是在每个发送周期的开始优先发送补偿包；其中发送周期由协议确定的发送速率 R 决定；这种策略不仅减少了延迟，并具有 NACK 抑制及消除抖动的作用。

3.1 随机回退时间的确定

给定最大随机回退时间 $T_{maxBackoff}$ 及组长度 $groupSize=n*packetSize$，随机回退时间 t' 的算法为：

令优化系数 $\lambda=\ln(groupSize)+1$；

在 $\lambda/(T_{maxBackoff}*(\exp(\lambda)-1))$ 与 $\lambda/(T_{maxBackoff}*(\exp(\lambda)-1))+\lambda/T_{maxBackoff}$ 之间以等概率取随机数 x；

取随机回退时间 $t'=T_{maxBackoff}/\lambda*\ln(x*(\exp(\lambda)-1)*(T_{maxBackoff}/\lambda))$；其中，$T_{maxBackoff}=GRTT$。

3.2 发送周期的确定

式(3)计算的是 TCP 发送速率的上限。如果发送端以低于或等于这个速率发送数据，则被认为具有 TCP 友好性[3]。

$$R'=S/(GRTT\times\sqrt{(2/3)\times p}+12\times\sqrt{(3/8)\times p}\times(1+32\times p^2)) \tag{3}$$

式中，R' 为计算出的发送速率上限；$GRTT$ 为数据包的往返时间；p 为接收端计算出的丢包事件率，发送端取所有接收端当前最大丢失率；S 是数据包大小 $packetSize$。

由于流媒体传输的实时性和传输质量的要求，需要确定一个最小的传输速率 R_{MIN}，R_{MIN} 的选择与应用中媒体的类型相关。发送端实际的发送速率 $R=\mathrm{MAX}(R_{MIN}, R')$。

则一个发送周期即为 $packetSize/R$。

3.3 包丢失率的计算

TFRC 协议对丢包的定义是：当连续收到 3 个大于应收

包序号的数据包时，则认为这个数据包已经丢失。但由于无线网络环境受干扰的可能性较大，会出现阵发性的包丢失及接收包间隔时间过长的现象，因此原有的丢包定义不再适用。

本协议使用一种简单的统计包丢失率的方法，即仅当数据包所在的序号从活动窗口移出时方确认为包丢失。算法描述如下：

第 i 组从窗口中被移出时，数据包的第一个序列号为 s_i，最后数据包的序列号为 e_i，该组为未被接收的数据包 l_i，则第 i 组的丢失率 p_i 为 $l_i/(e_i-s_i)$。应用加权平均值法求该接收端的丢失率：

$$p=\partial * p+(1-\partial) * p_i$$

式中，∂ 为权值因子，取值为 0.1。

3.4 补偿包的发送

为减少延迟，发送端不会在一轮结束的时候才发送修复包，而是每经过一个发送周期就检查一次是否有补偿包需要发送，如果有，则发送补偿包。

设向量 X 存放了一轮中发送端对从接收端接收到的 NACK 的累积结果 $\{x_i, x_{i+1}, \cdots, x_{i+Gc-1}\}$；其中 x_{i+j} 表示所有接收端在组号为 $(i+j)$ 的 NC 组丢包的最大个数，Gc 是发送端发送窗口中 NC 组的个数。每一轮结束，X 清零。

Y 中存放的是在一个发送周期中累积的需要发送的补偿包的个数：$\{y_i, y_{i+1}, \cdots, y_{i+Gc-1}\}$；其中 y_{i+j} 表示本发送周期在组号为 $(i+j)$ 的 FEC 组需要发送的补偿包的个数。每完成一次发送，Y 清零。

假定一个 NACK 包中的请求修复列表为 Z：$\{z_i, z_{i+1}, \cdots, z_{i+Gc-1}\}$，其中 z_{i+j} 为第 $(i+j)$ NC 组需要发送的补偿包的个数。计算 X 和 Y 的算法描述为：

```
If(x_{i+j}=0) x_{i+j}=z_{i+j}
else
    if(z_{i+j}>x_{i+j})
        x_{i+j}=z_{i+j};
        new=z_{i+j}-x_{i+j};
        if(y_{i+j}=0) y_{i+j}=new;
        else y_{i+j}=y_{i+j}+new
```

4 实验结果

实验条件：Buffalo WHR-HPG54 无线 AP，及 41 台 DELL d630 笔记本，其中一台负责发送，40 台负责接收。终端操作系统为 windows XP，每组包的个数 n 最大为 128。

图 3 显示了随着接收端数目的增加，本协议与 NORM[1] 协议在有效载荷吞吐量上的对比结果。实验结果表明，本协议随着接收端数目的增加，接收速率较稳定。而 NORM 协议接收速率在 1Mbps 以下，并随着接收端数目的增加呈下降趋势。本协议较 NORM 协议，在有效载荷吞吐量上具有 1.5 倍的优势。

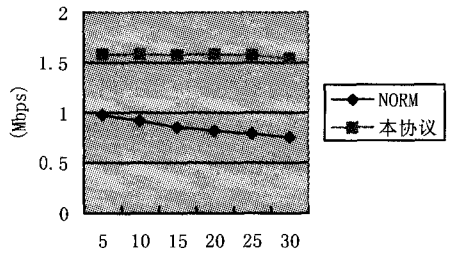

图 3 有效载荷吞吐量比较

图 4、图 5 分别显示了在本协议基础上，应用网络编码后多播协议性能的比较。

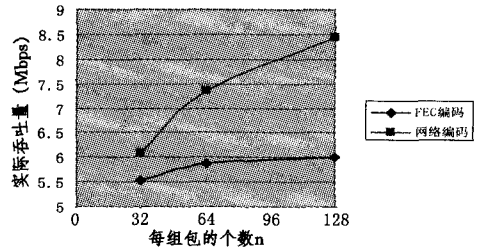

图 4 组播协议实际吞吐量

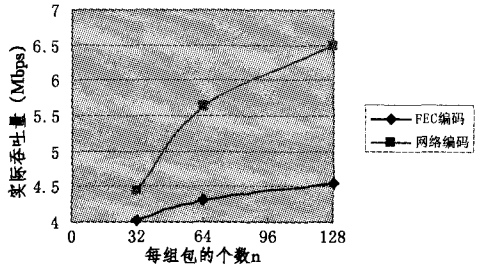

图 5 组播协议有效载荷吞吐量及实际发吞吐量的百分比

如图 4 和图 5 所示，无论是有效载荷还是实际吞吐量，网络编码都明显高于 FEC 编码，这是由于使用了渐进式的编码方法，将接收端的解码时间分解到等待接收包的时间中，并能及时发现无效编码包；FEC 编码总是在每个组接收到 n 个包后再进行解码，每个组都需要一个单独的解码时间，且只能在解码失败后才能发现无用编码包，浪费了接收等待时间。

如图 4 所示，随着 n 的增加，FEC 编码与网络编码的吞吐量逐渐增加。这是由于 n 越大，每发送一组数据，用于补偿的时间就相对越短，因此带宽利用率越高。但在 $n=64$ 之后，FEC 编码的吞吐量增加速率减慢，且趋于平缓；而网络编码的吞吐量几乎呈线性速率增加。图 5 显示了有效载荷占实际吞吐量的百分比。随着 n 的增大，有效载荷所占的百分比有所增加，且网络编码有效载荷的比率稍高。这是由于 FEC 编码的编码空间狭小，随着 n 的增加，每组丢失包的数量增加，而每组能够获取的编码包的个数不变（仍是 128），因此接收端从发送端获取无用编码包的概率增加，用于补偿的时间增多，吞吐量增加缓慢以致不再增加；而网络编码的编码空间很大，以本协议为例，若使用 16bit 系数，每组可生成 $2^{16}-1$ 个编码包，接收端获取无用编码包的概率很小，对吞吐量不会产生影响，因此随着 n 的增加，吞吐量持续增加。

根据对实验过程的观察，两种算法的 CPU 占有率相同。由此可见，网络编码在不增加额外开销的基础上大大提高了可靠多播的吞吐量。

结束语 流媒体传输具有实时性的特点，为了达到最小的时延，必须做到：

1）尽量避免反馈和补偿过程。协议通过预补偿，即在发送新数据时发送一定量的冗余编码包，减少接受方包丢失。

2）尽量小的 NACK 反馈延迟，通过减小 NACK 发送间隔来实现。基于反馈轮的算法，两个 NACK 之间最大间隔仅为 2GRTT。

3）尽量小的补偿包发送延迟：发送方收到 NACK 后，立

（下转第 270 页）

背景下对本文动态任务分配机制的有效性进行实验分析。设MAS系统即时任务数为10，任务处理Agent初始数目为30。为模拟任务分配过程的动态性，定义一个整数变量$\varphi(-10\leqslant\varphi\leqslant10)$作为每次任务分配过程中加入和退出MAS任务分配的Agent数目，确定进行该动作的Agent由系统随机确定。参数设置：遗传算法编码位数与即时Agent数目一致，交叉概率P_c、变异概率P_m分别是均匀分布在区间(0.7,0.9)和(0.01,0.2)上的一个随机变量，模拟退火算子中初始温度T_0为100，温度更新参数μ为0.1，迭代结束温度ε为0.1。蚁群算法信息素因子α为2，启发因子β为3，信息素挥发系数ρ为0.1，q_0为0.9，ξ为0.1。因为系统消耗的时间代价是关键因素，所以将ω_1取为1，ω_2取为0。现将文献[1]中的HA[1]算法和文献[2]中HAGA[2]算法与本文的HAGA动态任务分配机制进行比较分析，表1是任务和Agent在不同规模时3种机制的优化比较，T为优化时间代价，G为迭代次数。图1为规模为10×30时的3种机制的进化曲线，采用样本无限逼近的方法得出，表示了优化时间和精度的关系。

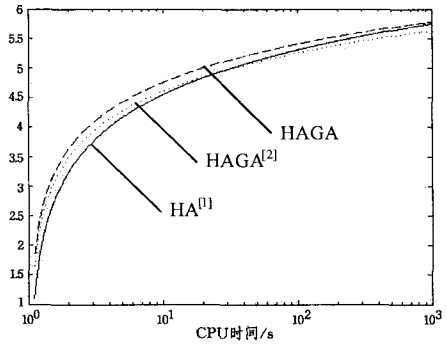

图1 优化进化曲线

表1 HA[1]、HAGA[2]和HAGA优化比较

	HA[1]		HAGA[2]		HAGA	
	T	G	T	G	T	G
10×20	38.19	451	32.59	367	30.13	321
10×24	40.13	479	32.93	385	30.87	335
10×27	43.93	529	36.89	476	31.19	369
10×31	47.85	539	37.61	519	32.49	487
10×38	55.03	545	45.32	527	33.57	495

由表1可知，从运行时间上来看，HAGA在Agent规模动态变化时的优势更明显。文献[1]的HA[1]算法Agent规模较大时其运行时间和精度已经不再适用。文献[2]的HAGA算法存在一个遗传和蚁群动态衔接的问题，只存在实验经验而没有理论衔接点。

结合图1优化进化曲线综合分析，在Agent规模动态增加并且算法迭代接近最优值的情况下，HAGA[2]算法运行时间大量增加，说明该机制不能保证在求解时间上的可靠性。两种算法的动态衔接问题成为影响算法性能的关键因素。在Agent规模为10×30的情况下，文献[1]的HA[1]算法在搜索初期要花费大量的时间，而文献[2]中HAGA[2]算法在算法运行后期性能明显下降。由此，可以综合得出，不论Agent规模是否动态变化，本文的HAGA动态任务分配机制适应性良好，保持了良好的时间效率和求解精度。

结束语 本文提出了MAS中动态任务分配问题的数学描述，建立了动态任务分配数学模型和相应的目标函数。应用了混合遗传蚁群算法对Agent动态任务分配进行优化组合求解。该算法的遗传算法部分加入了模拟退火算子，蚁群算法部分把全局信息素更新和局部信息素更新结合起来。仿真实验表明，该机制克服了传统遗传算法、蚁群算法易陷入局部最优和求解效低的问题，实现了两种算法的动态融合，提高了搜索速度和全局优化能力，对动态任务分配问题有效，并且在求解时效和精度上有良好的表现。但是，本文的仿真实验环境并非严格的分布式计算环境，任务分配的动态因素设置有限。在今后的研究工作中，设想将其他资源代价因素考虑在内。

参考文献

[1] 严建峰，李伟华，刘明. 基于混合蚁群算法的MAS任务分配[J]. 计算机应用研究，2009，26(1)：68-70

[2] 梁军，程显毅. 基于混合蚁群遗传算法的Agent联盟求解[J]. 计算机科学，2009，36(4)：227-231

[3] 赵玉兰. 基于混合蚁群遗传算法求解Agent联盟[J]. 信息与电脑，2010，6：111-113

[4] 徐金荣，李允，刘海涛，等. 一种求解TSP的混合遗传蚁群算法[J]. 计算机应用，2008，28(8)：2084-2088

[5] 杜宗宗，刘国栋. 基于混合遗传模拟退火算法求解TSP问题[J]. 计算机工程与应用，2010，46(29)：40-46

[6] 玄光男，程润伟. 遗传算法与工程优化[M]. 于歆杰，周根贵，译. 北京：清华大学出版社，2004

[7] 刑文训，谢金星. 现代优化计算方法[M]. 北京：清华大学出版社，2005

(上接第259页)

即发送补偿，而NORM协议则在一个时间片后集中发送补偿包，不利于数据的及时恢复。当补偿包数目较大时，会造成一定的抖动。

4)本文算法采用滑动窗口机制，根据媒体数据的Deadline决定是否对丢失补偿，而不是保证绝对可靠。可以避免无限制的回滚造成的延迟。

实验表明，结合网络编码的反馈轮机制更能适应无线网络环境高误码率和丢失率的特点，更适用于流媒体多播的传输。

参考文献

[1] Adamson B, Bormann C, Handley M, et al. NACK-Oriented Reliable Multicast Transport Protocol[Z]. rfc 5740, 2009

[2] Rizzo L. Effective erasure codes for reliable computer communication protocols[J]. ACM Sigcomm Computer Communication Review, 1997, 27(2): 24-36

[3] Sisalem D, Wolisz A. MLDA: a TCP_friendly congestion control framework for heterogeneous multicast environments[C]// Proc. of the 8th International Workshop on Quality of Service. Pittsburgh: [s. n.], 2000: 65-74

[4] Li S Y R, Yeung R W, Cai N. Linear Network Coding[J]. IEEE Transactions on Information Theory, 2003, 49(371)