[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
H04L 29/06 (2006.01)

[12] 发明专利说明书

专利号 ZL 00817212.9

[45] 授权公告日 2006 年 8 月 16 日

[11] 授权公告号 CN 1270496C

[22] 申请日 2000.10.13 [21] 申请号 00817212.9

[30] 优先权

[32] 1999.10.14 [33] US [31] 60/159,360

[32] 2000. 3.28 [33] US [31] 09/536,639

[86] 国际申请 PCT/US2000/028326 2000.10.13

[87] 国际公布 WO2001/028180 英 2001.4.19

[85] 进入国家阶段日期 2002.6.14

[71] 专利权人 诺基亚有限公司

地址 芬兰埃斯波

共同专利权人 希姆·列

[72] 发明人 希姆·列 C·克兰顿 H·郑 Z·刘

审查员 程 东

[74] 专利代理机构 中国专利代理(香港)有限公司

代理人 栾本生 张志醒

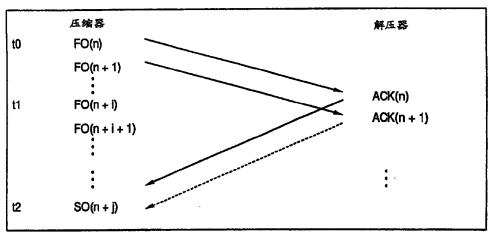
权利要求书 2 页 说明书 52 页 附图 20 页

[54] 发明名称

用于传送和接收包的方法和系统

[57] 摘要

本发明是一个用于同步发送器与接受器之间数据包中压缩报头的传送的方法和系统，具有最佳的无线应用，是对 RFC 2508 的一种改进。 按照本发明的在有向接收器传送每个含有报头的多个包的发送器的系统中同步发送器与接收器之间压缩报头的传送的方法包括从发送器向接收器传送一个当前包，当前包含有的信息是，发送器发送后继传送的包的准备就绪，后继传送的包中的报头是对照当前包中的报头而压缩的；从接收器向发送器传送一个表示接收器已经接收到当前包的确认包。



1. 在具有向接收器传送多个各含一个报头的包的发送器的系统中同步发送器与接受器之间压缩报头的传送的方法，包含:

从发送器向接收器传送一个当前包，当前包含有的信息是，发送器发送后继传送的包的准备就绪，后继传送的包中的报头是对照当前包中的报头而压缩的；和

从接收器向发送器传送一个表示接收器已经接收到当前包的确认包；其中

当前包的报头是完整报头或第一级压缩报头的其中之一；和

后继传送的包的压缩报头是第二级压缩报头。

2. 按照权利要求 1 的方法，其中:

发送器将已经被确认为被接收器接收到的当前包的报头存储起来，作为在后继传送的包的传送中使用的基准报头，并作为将被接收器用来解压后继传送的包的报头的基准报头;

接收器把被确认的当前包的报头存储起来，用来解压后继传送的包的压缩报头;

发送器用所存储的当前包的报头作为基准报头来传送后继的包;和

接收器用所存储的基准报头来解压后继接收的包的压缩报头，以产生不压缩的完整报头。

3. 按照权利要求 1 或 2 的方法，其中:

当前包的报头是一个完整报头；和

后继传送的包的压缩报头是第一级压缩报头。

4. 按照权利要求 1 或 2 的方法，其中:

当前包的报头是一个完整报头；和

后继传送的包的压缩报头是第一级压缩报头。

5. 一种系统，包含:

传送多个各含有一个报头的包的发送器;

接收所传送的多个包的接收器；其中

从发送器向接收器传送一个当前包，当前包含有的信息是，发送器发送后继传送的包的准备就绪，后继传送的包中的报头是对照当前包中的报头而压缩的；而接收器向发送器传送一个表示接收器已经接

收到当前包的确认包；其中

当前包的报头是一个完整报头；和

后继传送的包的压缩报头是第一级压缩报头。

6．按照权利要求 5 的系统，其中：

发送器将已经被确认为被接收器接收到的当前包的报头存储起来，作为在后继传送的包的传送中使用的基准报头，并作为将被接收器用来解压后继传送的包的报头的基准报头；

接收器把被确认的当前包的报头存储起来，用来解压后继传送的包的压缩报头；

发送器用所存储的当前包的报头作为基准报头来传送后继的包；和

接收器用所存储的基准报头来解压后继接收的包的压缩报头，以产生不压缩的完整报头。

7．按照权利要求 5 或 6 的系统，其中：

当前包的报头是一个第一级压缩报头；和

后继传送的包的压缩报头是第二级压缩报头

8．按照权利要求 5 或 6 的系统，其中：

当前包的报头是一个完整报头；和

后继传送的包的压缩报头是第二级压缩报头。

用于传送和接收包的方法和系统

技术领域

本发明涉及数据包传输中报头的压缩和解压。

背景技术

对于基于因特网协议（IP）的实时多媒体来说，除了用户数据报协议（UDP/IP）之外，实时传输协议（RTP）的使用占据主导的地位。RFC 1889 中详细描述了 RTP 概念，在此全文引用作为参考。组合的 IP/UDP/RTP 报头规模，对 IPv4 来说至少是 40 字节，对 Ipv6 来说至少是 60 字节。每个包共有 40－60 字节的开销，在频谱效率是基本问题的系统中（例如蜂窝网络中）可以说是较重的。因此，需要有合适的 IP/UDP/RTP 报头压缩机制。

RFC 2508 中描述了一种现行的报头压缩方案，作者为 S. CaSNER、V. Jacobson，名称为“Compressing IP/UDP/RTP Headers for Low Speed Serial Links”（由因特网工程特别工作组于 1999 年 2 月发表，这里全文引用），该方案能在点对点链路上把 40/60 字节的 IP/UDP/RTP 报头压缩到 2 或 4 字节。现有的报头压缩算法所根据的是，在一个完整的会话期间，包流中的 IP 包报头的多数字段保持不变。因此，就有可能通过在该解压器上建立一个压缩状态（完整报头信息）和通过只从压缩器向解压器传输最小量的报头信息而压缩报头信息。

例如在 RFC 2508 中所述的 IP/UDP/RTP 报头压缩方案，利用了下述的事实－报头中携带的某些信息字段或者 1)不改变（“类型 1”报头字段），或者 2)以一定程度上可预测的方式改变（“类型 2”报头字段）。其它字段叫作“类型 3”报头字段，其不同之处是它们必须以某种形式在每一个包中被传输（即，它们是不可压缩的）。

类型 1 报头字段的例子是 IP 地址、UDP 端口号、RTP SSRC（同步源）等等。在一个会话过程期间，只需一次性将这些字段传输到接收器/解压器（例如在会话建立时作为包的一部分）。类型 1 字段也称为“不变”字段。

类型 2 报头字段的例子是 RTP 时间戳、RTP 序列号和 IP ID 字段。所有都有一个按某个常量从包(n)到包(n+1)递增的倾向。因此，不必

在每一个报头内传输这些值。只需要让接收器/解压器知道与每个具有这个属性的字段相关联的常量递增值－以下称作第一级差（FOD－first order difference）。接收器/解压器在重构原始报头时用这些 FOD 来重新生成最新的类型 2 字段值。类型 2 字段是“变化”字段的一部分。

应当强调，有时类型 2 字段的不规则地变化。这类事件的频率取决于若干因素，包括正在被传输的媒体的类型（例如声音或视频）、实际的媒体源（例如对于声音来说，特性会因说话人而异）和同时共享相同 IP 地址的会话的数量。

类型 3 报头字段的例子是 RTP M-位（标记），其指示在媒体中出现某种边界（例如视频帧）。因为媒体一般以不可预测的方式变化，该信息不能被真实地预测。类型 3 字段是“变化”字段的一部分。

解压器保存含有所有与重建报头相关的信息的解压上下文信息。该信息主要是类型 1 字段、FOD 值、以及其它信息。当包被损失或损坏(corrupted)时，解压器可能丧失与压缩器的同步，以致不再能够重建包。同步的丧失可能发生在包在压缩器与解压器之间传输期间被丢失或损害的时候。

在上述条件下，压缩器在一个会话期间需要传输三种不同类型的报头:

完整报头（FH）：含有所有报头字段（类型 1、2 和 3）的全部集合.

该类型的字段对发送来说是最不优的，因为其规模大（例如对于 Ipv4 来说 40 字节）。一般最好是只在会话的开始发送一个 FH 包（以在接收器上建立类型 1 数据）。传输额外的 FH 包对压缩算法的效率有不利的影响。当压缩器传输 FH 包时，称作处于“FH 状态”。

第一级（FO）：含有最少的报头信息（例如类型 3 字段）、压缩器/解压器特定的控制字段(特定于在使用的压缩算法)和描述当前 FOD 字段中变化的信息。FO 包基本上是 SO 包（下文作说明），带有为在解压器的一个或多个类型 2 字段建立新 FOD 信息的信息。如果报头压缩在被应用到一个 VoIP（因特网协议上的语音）流，FO 包的传输可能被声音中的一段静音之后发生的突发谈话触发。这样的事件导致 RTP 时间戳值的某种预料不到的变化，需要在接收器由不是当前 FOD 的值更

新 RTP 时间戳。FO 包的大小取决于第一级差异改变的类型 2 字段的数量（以及每个变化的绝对值的量）。当压缩器传送 FO 包时，称其处于“FO 状态”。

第二级（SO）：SO包含有最小的报头信息（例如类型3字段）以及压缩器和解压器特定的控制字段。压缩器和解压器的最佳操作方式是传输和接收SO包，因为它们规模最小（数量级只有2字节或更少）。当压缩器传输SO包时，称其处于“SO状态”。SO包仅当当前报头符合一个FOD的模式时才被传输。

RFC 2508所根据的概念的是，多数时间，从一个包到下一个包改变的RTP字段，诸如RTP时间戳，能通过对所传送的SO包的线性插值而预测。实际上唯一必须要传送的是一个被用于错误和包损失检测的序列号（以及一个上下文信息ID）。当发送器确定线性插值不能针对紧邻的前导包而应用于当前包时，就传送一个FO包。为启动这个会话，传送一个FH包。此外，当接收器确定没有包损失时（包损失是通过一个序列号递增数大于1而检测到的），接收器将明确地请求发送器传送完整的报头，以便进行再同步。

然而，RFC 2508中定义的报头压缩并不很适合于某些带宽宝贵、经常出错的环境（诸如蜂窝或无线环境）。在RFC 2058压缩方案中，RTP时间戳被假设为多数时间有一个线性增加的模式。当报头符合该模式时，实际上只需要在压缩报头中发送一个短序列号作为SO。当报头不符合该模式时，则在FO压缩报头中发送当前报头与前一个报头的时间戳之间的差。

根据不同的运行环境，有各种各样的带宽要求。例如，在蜂窝系统中，一般非常需要尽可能地限制带宽的使用，因为这是一种稀缺资源。

RFC 2508缺少抗击报头错误或损失的强健性，因为只有当紧邻的前导报头也接收无误并被解压后才能解压当前的报头。RFC 2508中定义的压缩不能很好地适合于蜂窝环境，蜂窝环境中带宽宝贵，长错误突发(long error burst)并非鲜见。在RFC 2508中，检测到包损失时，后继的带压缩报头的包都被无效，并进一步要求需要发送更大的报头来从错误中恢复。大报头耗费带宽，引起难以控制的带宽需求高峰。

只用短序列号（位数有限的号）来检测包损失对于易出错误的网

络来说，例如对于任何时刻都可能发生长损失的无线网络来说，是不强健的。在这种情况中，长损失被定义为序列循环的损失或一连串包的损失。在长损失的情况下，在有由有限数目的位定义的包标识的序列循环(sequence cycle)的多个包内的一系列包会丢失，结果，被接收器的解压器（上传链路或下传链路中的接收装置）接收的包中的序列号绕回（重复）。例如，假设序列号由 k 位组成，则序列循环等于 2^k。

如图 1 中所示，压缩器（上传链路或下传链路中的传送装置）在时刻 t0 发送 seq=n 的包和跟随其后的 2^k 个包，在时刻 t1 从 seq=n+1 的包开始，在时刻 t2 以 seq=n 的包结束。在时刻 t3，压缩器再次发送一个序列号等于 n+1 的包。假设一个在时刻 t1 发送的序列号等于 n+1 的包直到序列号等于 n 的包在时刻 t2 被发送，它们都因为长损失而被丢失，则解压器仅仅收到在时刻 t0 发送的序列号等于 n 的包和在时刻 t3 发送的序列号等于 n+1 的包。根据在 RFC 2508 中定义的当前包损失检测方案，解压器得出没有包损失的结论并以错误的方式解压包。这不仅影响序列号等于 n+1 的包的解压的正确性，也影响后继的包。

发明概述

本发明是一种提供改进的在诸如无线通信等易于发生由衰减(fading)引起的包接收的定期中断的环境中传送和接收包的系统和方法。本发明与 RFH 2508 相比提供改进的包传送和接收性能，包括以上讨论的图 1 的现有技术的绕回问题。按照本发明的对当前包中的压缩报头的适当解码，不像 RFC 2508 那样取决于对紧邻的前导包的正确解压。

本发明提供一种在报头压缩级检测长损失的机制，以及在检测到长损失后的对应恢复方案。本发明总体适合于在出现长错误脉冲串时必须在传送器与接收器之间保持序列同步的通信协议。

自适应报头压缩的是用于强健报头压缩的总架构，能被参数化，以考虑逆信道的存在/不存在以及性能特征。该架构包括三个基本操作方式:

*双向确定性方式: 该方式被用于有一个“定义良好的”逆信道的情况中，可被用来从解压器向压缩器传输各类反馈信息。这种来自解压器的反馈的一个例子是确认，例如用于从较低的压缩状态前进到较高的压缩状态。

*双向机会性方式：该操作方式被用于的情形是，逆信道存在、但是“松散地”定义的，例如信道不是总是能用，或者可能较慢/不可靠。

*单向方式（悲观的或乐观的）：该操作方式被用于没有任何种类的逆信道的情形中。

按照本发明的在有向接收器传送多个各含一个报头的包的发送器的系统中同步发送器与接受器之间压缩报头的传送的方法，包括从发送器向接收器传送一个当前包，当前包含有的信息是，发送器发送后继传送的包的准备就绪，后继传送的包中的报头是对照当前包中的报头而压缩的；以及从接收器向发送器传送一个表示接收器已经接收到当前包的确认包。发送器可以将已经被确认为被接收器接收到的当前包的报头存储起来，作为在后继传送的包的传送中使用的基准报头，并作为将被接收器用来解压后继传送的包的报头的基准报头；接收器可以把被确认作为基准报头的当前包的报头存储起来，用来解压后继传送的包的压缩报头；发送器可以用所存储的当前包的报头作为基准报头来传送后继的包；且接收器可以用所存储的基准报头来解压后继接收的包的压缩报头，以产生不压缩的完整报头。当前包的报头可以是一个完整报头；后继传送的包的压缩报头可以是第一级压缩报头；当前包的报头可以是第一级压缩报头；后继传送的包的压缩报头可以是第二级压缩报头；当前包的报头可以是完整报头；后继传送的包的压缩报头可以是第二级压缩报头。

按照本发明的一个系统包括传送多个各含有一个报头的包的发送器和接收所传送的多个包的接收器；其中发送器向接收器传送一个当前包，当前包含有的信息是，发送器发送后继传送的包的准备就绪，后继传送的包中的报头是对照当前包中的报头而压缩的；接收器传送一个表示接收器已经接收到当前包的确认包。发送器可以将已经被确认为被接收器接收到的当前包的报头存储起来，作为在后继传送的包的传送中使用的基准报头，并作为将被接收器用来解压后继传送的包的报头的基准报头；接收器可以把被确认作为基准报头的当前包的报头存储起来，用来解压后继传送的包的压缩报头；发送器可以用所存储的当前包的报头作为基准报头来传送后继的包；且接收器可以用所存储的基准报头来解压所接收的后继传送的包的压缩报头，以产生不压缩的完整报头。当前包的报头可以是一个完整报头；后继传送的包

的压缩报头可以是第一级压缩报头；当前包的报头可以是第一级压缩报头； 后继传送的包的压缩报头可以是第二级压缩报头； 当前包的报头可以是完整报头； 后继传送的包的压缩报头可以是第二级压缩报头。

按照本发明的在有向接收器传送多个各含一个报头的包的发送器的系统中的一种解压在被接收器接收的当前包中含有的压缩报头的方法，包括在接收器用计数器确定在连续接收的包之间的消逝时间Δt；比较当前接收的包与紧邻的前一个接收的包之间的消逝时间Δt，以确定该消逝时间Δt 是否至少等于一个表明在当前接收的包与紧邻的前一个接收的包之间至少有一个包丢失的消逝时间；处理该表明至少有一个包丢失的消逝时间Δt，以确定在紧邻的前一个接收的包与当前接收的包之间的丢失的包的个数(a number of missing packets)；将该丢失的包的个数加到紧邻的前一个接收的包的包编号(packet number)上，以更新当前包在该多个包的传输的序列(a sequence of transmission)中的编号；用更新后的当前包的编号解压当前包的压缩报头。当前包的压缩报头可以是第二级压缩报头。在紧邻的前一个接收的包与当前接收的包之间的丢失的包的个数可以按下述计算：

$i*SEQ\_CYCLE+DIFFk(n_2,n_1)$，如果$(DIFF(n_2,n_1)+i*SEQ_CYCLE)*$(t 个时间单位)$\leqslant \Delta t<DIFF(n_2,n_1)+(i+1)*SEQ_CYCLE)*$(t 个时间单位)，其中 i 是等于或大于零的整数，n_2是在包括当前包的包的传输的序列中的一个序列号，n_1是在包括紧邻的前一个接收的包的包的传输的序列中的一个序列号，SEQ_CYCLE 等于2^k，其中 k 是序列号的位数，$DIFF(n_2,n_1)$是当前包的包号与紧邻的前一个接收包的包号之间的差，$n_2>n_1$时，$DIFF(n_2,n_1)=n_2-n_1$；$n_2\leqslant n_1$时，$DIFF(n_2,n_1)=n_2+2^k-n_1$。

按照本发明的系统包括传送多个各含有一个报头的包的发送器和接收所传送的多个包的接收器； 其中，接收器包含一个确定在连续接收的包之间的消逝时间Δt 的计数器；比较当前接收的包与紧邻的前一个接收的包之间的消逝时间Δt，以确定该消逝时间Δt 是否至少等于一个表明在当前接收的包与紧邻的前一个接收的包之间至少有一个包丢失的消逝时间；处理该表明至少有一个包丢失的消逝时间Δt，以确定在紧邻的前一个接收的包与当前接收的包之间的丢失的包的个数；

将该丢失的包的个数加到紧邻的前一个接收的包的包编号上，以更新当前包在该多个包的传输的序列中的编号；用更新后的当前包的编号解压当前包的压缩报头。当前包的压缩报头可以是第二级压缩报头。在紧邻的前一个接收的包与当前接收的包之间的丢失的包的个数可以按下述计算：

$i*SEQ\_CYCLE+DIFFk(n_2, n_1)$，if $(DIFF(n_2, n_1)+i*SEQ\_CYCLE)*(t$ 个时间单位$) \leqslant \Delta t < DIFF(n_2, n_1)+(i+1)*SEQ_CYCLE)*(t$ 个时间单位），其中 i 是等于或大于零的整数，n_2是在包括当前包的包的传输的序列中的一个序列号，n_1是在包括紧邻的前一个接收的包的包的传输的序列中的一个序列号，SEQ_CYCLE 等于 2^k，其中 k 是序列号的位数，$DIFF(n_2, n_1)$是当前包的包号与紧邻的前一个接收包的包号之间的差，$n_2>n_1$时，$DIFF(n_2, n_1)=n_2-n_1$；　$n_2 \leqslant n_1$时，$DIFF(n_2, n_1)=n_2+2^k-n_1$。

按照本发明的在有向接收器传送多个各含一个报头的包的发送器的系统中同步发送器与接受器之间第一级压缩报头的传送的方法，包括从发送器向接收器传送一个含有第一级压缩报头的当前包，其中该多个包中的当前包的编号(number)是通过一个有 1 位的扩展序列号(extended sequence number)编码的；响应对含有第一级压缩报头的当前包的接收，从接收器向发送器传送一个确认包，表示接收器已经接收到含有第一级压缩报头的当前包；响应对确认包的接收，发送器传送后继的各含有一个序列号的包，序列号有一个 k 位的非扩展序列号，l>k。发送器可以将已经被确认为被接收器接收到的当前包的报头存储起来，作为在后继传送的含有第一级压缩报头的包的传送中使用的基准报头，并作为将被接收器用来解压后继传送的包的报头的基准报头；接收器可以把被确认作为基准报头的当前包的报头存储起来，用来解压后继传送的含有第一级压缩报头的包的压缩报头；发送器可以用所存储的当前包的报头作为基准报头来传送后继的含有第一级压缩报头的包；接收器可以用所存储的基准报头来解压所接收的后继传送的含有第一级压缩报头的包的压缩报头，以产生不压缩的完整报头。接收器通过比较连续接收到的被传送报头的序列号可以检测到至少一个在后继传送的包中丢失的包。

按照本发明的在有向接收器传送多个各含一个报头的包的发送器的系统中同步发送器与接受器之间第一级压缩报头的传送的方法，包

括从发送器向接收器传送多个各含有第一级压缩报头的当前包，其中，多个包的每个包的编号，按照传送的顺序，由一个有 1 个扩展位的序列号定义；当差 DIFF 等于(CD_SN_CURR 和 CD_SN_LAST）之间的差时在当前包与上一个包之间检测到所传送的多个包中的至少一个丢失的包，其中 CD_SN_LAST 是当前包的绝对序列号，CD_SN_CURR 是上一个接收的包的绝对序列号，其中绝对序列号是由发送器和接收器保持的一个有 J 位的计数器，其中 J>=1 个扩展位。丢失的包数 N_{loss} 可以被计算为等于(CD_SN_CURR 和 CD_SN_LAST）之间的差。

按照本发明的系统包括传送多个各含有一个报头的包的发送器和接收所传送的多个包的接收器； 其中，由发送器向接收器传送一个含有第一级压缩报头的当前包，其中该多个包中的当前包的编号是通过一个有 1 位的扩展序列号编码的；响应对含有第一级压缩报头的当前包的接收，接收器向发送器传送一个确认包，表示接收器已经接收到含有第一级压缩报头的当前包；发送器响应对确认包的接收，传送后继的包，后继的包每个含有一个在这多个后继包中的 k 位的序列号，1>k。发送器可以将已经被确认为被接收器接收到的当前包的报头存储起来，作为在后继传送的含有第一级压缩报头的包的传送中使用的基准报头，并作为将被接收器用来解压后继传送的包的报头的基准报头；接收器可以把被确认作为基准报头的当前包的报头存储起来，用来解压后继传送的含有第一级压缩报头的包的压缩报头；发送器可以用所存储的当前包的报头作为基准报头来传送后继的含有第一级压缩报头的包；接收器可以用所存储的基准报头来解压所接收的后继传送的含有第一级压缩报头的包的压缩报头，以产生不压缩的完整报头。接收器通过比较连续接收到的被传送报头的序列号可以检测到至少一个在后继传送的包中丢失的包。

按照本发明的在有向接收器传送多个各含一个报头的包的发送器的系统中同步发送器与接受器之间报头的传送的方法，包括从接收器向发送器传送定期的确认，确认是以一个间隔时间分别向发送器传送的，使得发送器每隔 N 个包至少收到一个确认，其中 $N=2^k$，k 是用来对序列中的包编号的位的个数；如果发送器没有从接收器收到适当定时的确认时，接收器把定义序列号的位数增加到 1 个扩展位。接收器可以检测到丢失包的最大个数等于 2^1 个扩展位。发送器响应一个随后接

收到的确认，可以把含有 1 个扩展位的序列号中的位数减少到 k 个位。

按照本发明的在有向接收器传送多个各含一个报头的包的发送器的系统中在发送器与接收器之间传送有压缩报头的包期间保持序列同步的方法，包括通过从发送器向接收器传送一个有完整报头的包而启动有压缩报头的包的传送；在传送该有完整报头的包之后，从发送器向接受器传送有压缩报头的包，每个压缩报头含有与该有完整报头的包有关的信息；定期地从接收器向发送器传送一个确认包，表示有压缩报头的包已经被接收。发送器可以顺序地把一个对应有压缩报头的每个顺序包按 1 递增的序列号加到每个有压缩报头的包的压缩报头上，该序列号有一个预定的位数。当发送器没有接收到确认包时，可以将该序列号的位数增加到超过该预定的位数。

按照本发明的减少在一序列被传送数据包的报头中含有的位的个数方法，包括从发送器向接收器发送至少一个数据包序列，每个序列含有至少一个含有完整报头的包和随后的至少一个含有压缩报头的包，压缩报头的位数少于完整报头；响应由接收器接收的含有完整报头的数据包之一，从接收器向发送器传送一个表示接收器已经接收到这一个含有完整报头的数据包；响应该确认被发送器的接收，从发送器向接收器传送至少一个有被进一步压缩的报头的后继包，其压缩程度超过该至少一个序列中的至少一个报头的压缩程度。该至少一个序列的压缩报头可以是第一级压缩的报头；该至少一个后继包的压缩报头可以是第二级压缩的报头。可以传送多个序列的包。

接收器响应第一个接收到的含有完整报头的包而生成该确认。接收器可以响应对至少一个含有压缩报头的包的接收而向发送器传送至少一个额外的确认。该至少一个额外的确认可以响应该至少一个序列中的第一个包而生成。

按照本发明的减少在一序列被传送包的报头中含有的位的个数的方法，包括从发送器向接收器发送至少一个包序列，每个序列含有至少一个含有第一报头的包和随后的至少一个含有被压缩的位数少于第一报头的第二报头的包；响应由接收器接收的含有第一报头的包之一，从接收器向发送器传送一个表示接收器已经接收到这一个含有第一报头的包的确认；响应该确认被发送器的接收，从发送器向接收器传送至少一个有被进一步压缩得超过第二压缩报头的压缩程度的第三

报头的后继包。第二压缩报头可以是第一级压缩的报头；第三压缩报头可以是第二级压缩的报头。可以传送多个序列的包。接收器可以响应第一个接收到的含有第一报头的包而生成该确认。接收器可以响应对至少一个含有压缩报头的包的接收而向发送器传送至少一个额外的确认。该至少一个额外的确认可以响应该至少一个序列中的第一个包而生成。

按照本发明的减少在一序列被传送包的报头中含有的位的个数的方法，包括从发送器向接收器发送至少一个包序列，每个序列含有至少一个含有完整报头的包和随后的至少一个含有位数少于完整报头的压缩报头的包；响应由接收器接收的含有完整报头的包之一，从接收器向发送器传送一个表示接收器已经接收到这一个含有完整报头的包的确认。

按照本发明的减少在一序列被传送包的报头中含有的位的个数的方法，包括从发送器向接收器发送至少一个包序列，每个序列含有至少一个含有第一报头的包和随后的至少一个含有被压缩的位数少于第一报头的第二报头的包；响应由接收器接收的含有第一报头的包之一，从接收器向发送器传送一个表示接收器已经接收到这一个含有第一报头的包的确认。

按照本发明的从发送器向接收器传送一串有压缩报头的包内的包的方法，每个压缩报头含有一个序列号，标识每个包在该串中的位置，该方法在传送包之前处理该串，以检测该串何时含有至少一个丢失的或乱序的包；从发送器向接收器传送带压缩报头的串，其形式是一序列含有丢失的或乱序的包之前的包和丢失的或乱序的包之后的包；与至少一个丢失的或乱序的包之后的至少一个包一起传送包的数据串中任何丢失的或乱序的包的数量。接收器可以解压至少一个被接收的后继包，包括把任何丢失的或乱序的包的数量加到每个接收到的后继包的编号。接收器进行的解压可以使用在该至少一个序列之前作为该串包的一部分传送的一个被存储起来的基准包，并包括一个用来解压该至少一个后继包的序列号。该串可以含有至少一个丢失的包或至少一个乱序的包。

按照本发明的从发送器向接收器传送一串有压缩报头的包内的包的方法，每个压缩报头含有一个序列号，标识每个包在该串中的位置，

该方法在传送包之前处理该串，以检测该串何时含有至少一个丢失的包；从发送器向接收器传送带压缩报头的串，其形式是一序列含有丢失的包之前的包和丢失的包之后的包；与至少一个丢失的包之后的至少一个包一起传送任何丢失的包的数量。接收器可以解压至少一个被接收的后继包，包括把任何丢失的包的数量加到每个接收到的后继包的编号。接收器进行的解压可以使用在该至少一个序列之前作为该串包的一部分传送的一个被存储起来的基准包，并包括一个用来解压该至少一个后继包的序列号。

按照本发明的从发送器向接收器传送一串有压缩报头的包内的包的方法，每个压缩报头含有一个序列号，标识每个包在该串中的位置，该方法在传送包之前处理该串，以检测该串何时含有至少一个丢失的或乱序的包；从发送器向接收器传送带压缩报头的串，其形式是一序列含有丢失的或乱序的包之前的包和丢失的或乱序的包之后的包；与至少一个乱序的包之后的至少一个包一起传送任何乱序的包的数量。接收器可以解压至少一个被接收的后继包，包括把任何乱序的包的数量加到每个接收到的后继包的编号。接收器进行的解压可以使用在该至少一个序列之前作为该串包的一部分传送的一个被存储起来的基准包，并包括一个用来解压该至少一个后继包的序列号。

按照本发明的传送一串包的方法，包括用一个错误检测过程来处理该串包，以确定该串包中的任何含有错误的包；从发送器向接收器传送不带已经被除去的包的该串。错误检测过程可以用每个包内的一个校验和来确定该串包中的任何含有错误的包。错误检测过程可以处理每个包中的数据，以计算一个错误更正码并确定每个包中存储的错误更正码是否与所计算的错误更正码匹配，如果找不到匹配，就将该包从该串除去。至少有些包的报头在传送之前被压缩。至少有些包的报头的压缩在含有错误的包的去除之后发生。该串的传送可以在带宽有限的传送介质上，任何含有错误的包的去除，减少了传送期间对有限带宽的使用。

按照本发明的解压被传送的一串包的方法，各包含有一个标识传送之前每个被传送的包在该串包中的位置的序列号，该方法包括，从发送器向接收器传送该串包，所传送的包串中至少一个被接收的顺序包不在被接收的包串中；处理所接收的包的序列号，以确定所传送的

包串中的没有在被接收的包串中出现的顺序的包的个数(a number of sequential packets)；通过使用被确定没有接收到的顺序的包的个数，解压位于该至少一个没有出现在所接收包中的包之后的至少一个被接收的包的报头。对位于该至少一个没有出现在所接收包中的包之后的至少一个被接收的包的报头的解压，可以通过将该个数与一个插值函数的乘积加到位于该至少一个没有出现在所接收包中的包之前的被接收包的报头而进行。可能有多个被传送的包都不在接收的包中出现。对该至少一个所接收包中的位于该至少一个没有出现在所接收包中的包之后的至少一个压缩报头，可以通过把该个数与不同插值函数的多个乘积加到该至少一个所接收包中的位于该至少一个没有出现在所接收包中的包之后的至少一个压缩报头而进行解压。插值函数可以从位于该至少一个没有出现在所接收包中的包之后的多个包中的一个包到另一个包而线性地变化。插值函数可以在位于该至少一个没有出现在所接收包中的包之后的多个包中非线性地变化。插值函数可以是位于该至少一个没有出现在所接收包中的包之后的该至少一个接收的包的时间戳。插值函数可以代表位于该至少一个没有出现在所接收包中的包之后的该至少一个接收的包的序列号。插值函数可以是一个 IP ID。插值函数可以代表位于该至少一个没有出现在所接收包中的包之后的该至少一个接收的包的序列号。压缩报头可以是第二级压缩报头。

按照本发明的重新生成一串包内的压缩包的报头的方法，各包含有一个标识传送之前每个被传送的包在该串包中的位置的序列号，该方法包括，从发送器向接收器传送该串包，所传送的串内一个序列中至少一个被接收的包被接收时有错误的压缩报头；存储至少一个序列中的该至少一个被接收的有错误的包；当至少一个序列中的被存储的包的个数与一个通过处理该至少一个序列之前和之后的包的序列号而确定的数匹配时，通过把一个插值函数的函数加到至少一个序列的至少一个包的报头而重新生成至少一个被存储序列的压缩报头。该可以被加到该至少一个序列的至少一个包的报头的一个插值函数的函数，在该至少一个序列中顺序的包之间线性地增加。可以把多个不同插值函数的函数加到至少一个序列的至少一个包的报头。该可以被加到至少一个序列的至少一个包的报头的该多个插值函数的函数，可以在该

至少一个序列中顺序的包之间线性地增加。每个被存储的序列中的包的个数，可以根据紧邻在该至少一个序列之前的包的序列号与紧邻在该至少一个序列之后的包的序列号之间的差而确定。该插值函数可以是时间戳。该插值函数是序列号。该插值函数可以是IP ID。该多个插值函数可以是时间戳、序列号或IP ID。该至少一个包的每个压缩报头可以是第一级压缩报头。该至少一个包的每个压缩报头是第二级压缩报头。反馈可以是个确认包。

按照本发明的从压缩器向解压器传送报头的方法，包括从压缩器向解压器传送多个包；响应在解压器接收至少一个包，向压缩器传送至少一个反馈，表示解压器已经接收该多个包的至少一个；从压缩器向解压器传送至少一个额外的包，该至少一个额外的报头中的位数少于首先发生下述无论哪个情况时的至少一个包中的报头的位数：（1）该至少一个包的预定数量的包的传送；（2）接收至少一个反馈。该至少一个包的每个报头可以是完整报头；且该至少一个额外的包的每个报头可以是第一级报头。该至少一个包的每个报头可以是第一级报头。且该至少一个额外的包的每个报头可以是第二级报头。该反馈可以是确认包。该预定数量的包可以以某个选择标准为根据。

按照本发明的从压缩器向解压器传送报头的方法，包括从压缩器向解压器传送多个包；以及从压缩器向解压器传送至少一个额外的包，该至少一个额外的报头中的位数少于当该至少一个包的预定数量的包的传送发生时该至少一个包中的报头的位数。该预定数量的包可以以某个选择标准为根据。该选择标准可以以涉及从压缩器向解压器传送或从解压器向压缩器传送的通道状态。该选择标准可以以涉及从压缩器向解压器传送或从解压器向压缩器传送的通道状态。

按照本发明的在一个有向接收器传送每个包有一个报头的多个包的传送器的系统中在有压缩报头的包的传送期间在传送器与接收器之间保持序列同步的方法，包括通过从传送器向接收器传送一个有报头的包而启动有报头的包的传送；在该有一个报头的包的传送之后，从发送器向接收器传送有压缩报头的包，每个压缩报头都含有与该包的报头有关的信息；以及非周期性地从接收器向发送器传送表示有压缩报头的包已经被接收的确认包。

附图简述

图 1 表示 RFC 2508 的包丢失的缺陷。

图 2 表示一个可以在其中实施本发明的示例性系统。

图 3 概念性地表示压缩上下文信息。

图 4 概念性地表示解压上下文信息。

图 5A 和 5B 表示本发明的第一实施例。

图 6 表示按照本发明的压缩器利用确认从传送有较多位数的报头向传送有较少位数的报头的转换。

图 7 表示按照本发明的压缩器从传送有第一压缩级的报头向传送有第二压缩级的报头的转换。

图 8 表示当出现一个有包数大于 2^k 的回绕时包的检测和恢复，其中 k 个位被用来编码由 k 个位定义的序列号 n。

图 9 表示按照本发明的压缩器和解压器使用确认来控制在有 k 位和扩展的 1 位的序列号之间的转换以及转换回 k 位的操作。

图 10 表示按照本发明的通过在由解压器生成表示接收到 FH 包的确认之前传送一序列 FH 和 FO 包而取得的带宽缩减。

图 11 表示按照本发明的为恢复丢失的包的报头的由解压器接收的包的插值。

图 12 表示按照本发明的在包丢失时的序列补偿(compensation)。

图 13 表示按照本发明的为缩减带宽的解压器的错误检测上游(error detection upstream)。

图 14A－F 表示本发明所传送的包的格式。

图 15 表示按照本发明的仅当在从解压器接收到确认之后压缩器的压缩状态向更高状态的转换。

图 16 表示当收到确认时预定数量的包到达解压器之前压缩器的压缩状态向更高状态的转换。

图 17 表示当收到确认时预定数量的包到达解压器之后压缩器的压缩状态向更高状态的转换。

图 18 表示当仅当预定数量的包到达解压器之后压缩器的压缩状态向更高状态的转换。

图 19 表示本发明与 RFC 2508 之间的比较。

图 20 表示本发明的性能的图形分析。

最佳实施方式

图 2 表示一个可以在其中实施本发明的各种实施例的示例性系统。然而应当明白，本发明并不仅仅限于其，其它系统体系结构同样适用于实施本发明。终端 102 与 IP 网络 108 相连。终端 102 可以是、但不限于是运行 RTP/UDP/IP 的个人电脑等，提供在 RTP 包中包化的声音样本用于在 IP 网络上传输。终端 102 包括 RTP 端点 104，它将这个终端标识为 RTP 包的源或者目的地（例如包括 IP 地址、端口号等）。IP 网络在这里是包网络的一个例子，不过也可以用其它类型的包交换网络等等来代替。终端 102 也包括本地定时器 103，用于生成时间戳。

访问网络基础结构(ANI)110 和 120,可以驻留在基站子系统(BBS)中，它们连接到 IP 网络 108。这些 ANI 是网络实体和网络节点。许多是网络实体和网络节点并起移动压缩器和移动解压器作用的无线移动终端（图中显示了两个无线终端 130 和 150）通过无线电频率(RF)链路 140 连接到 ANI 110 和 120。当移动终端 130 和/或 150 之一移动时，有时由于移动范围超出与一个 ANI 的连接，有必要将终端续接到另一个 ANI。当报头压缩和解压在 ANI 中使用并位于 ANI 中时，这个过程也要求将压缩和解压上下文信息从一个 ANI（老的）传送到另一个 ANI（新的），以取得无缝的效果(seamlessness)，例如，如果移动终端 130 和/或 150 移动，并被从 ANI 110 续接到 ANI 120，如下所述，这种传送可能在不同时间发生，但是为了使中断最小化，应当在新 ANI 从老 ANI 接管报头压缩/解压功能时就完成。报头压缩/解压功能的重定位在新的网络实体接管的某个时刻发生。另一方面，上下文信息的传送可以在一段时间内在重定位之前进行。如图中所示，RF 链路 140 包括上传链路通信 142（从移动终端 130 和 150 向 ANI 110）和下传链路通信 144（从 ANI 110 向移动终端 130 和 150）。当一个或多个移动终端移动时，移动终端 130 和/或 150 从一个 ANI，例如 ANI 110，续接到另一个 ANI，例如 ANI 120。每个 ANI 与连接 IP 网络 108 的一个区域中的一个或多个无线（或无线电频率）终端（包括终端 130）接合，包括在（IP 网络提供的）有线信号与（从终端 130 和 150 或向终端 130 和 150 提供的）无线或 RF 信号之间的转换。这样，每个 ANI 允许将从 IP 网络 108 发送或接收的包，诸如但不限于 RTP 包，在 RF 链路 140 上发送到无线终端 130 和 150 的至少之一，并允许将要从终端 130 和

150传送的包，诸如RTP包但不限于RTP包，由IP网络108传送到另一个终端，如终端102。

每个ANI包括多个实体。为便于理解网络中所有ANI的体系结构和操作，对ANI 110给出更详细的描述和解释。所有ANI可以与ANI 110是相同的体系结构，但是所表示的详细程度不同。ANI 110包括一个或多个ANI适配器（ANI_AD），如ANI_AD 112（详细表示的）和ANI_AD 114，它们每个最好包括一个定时器113，用于提供时间戳。每个ANI_AD执行报头压缩（在下传链路通信之前）和解压（在上传链路通信之前）。从IP网络108接收的RTP包的报头（一个或多个报头字段，诸如时间戳和序列号）在通过下传链路通信142向终端130和150传送之前被ANI_AD 112压缩，从移动终端130和150接收的包报头在向IP网络传送之前被ANI_AD 112解压。ANI_AD 110起着传送器/接收器（收发器）的作用，特别是压缩器/解压器115的作用，压缩器在传送之前压缩数据包，解压器在接收之后解压数据包。ANI_AD 110与位于连接到IP网络110的区域(region)内的特定地区(area)或不同地区中的终端接合。ANI_AD 112包括一个用于执行基于定时器的解压技术的定时器113。ANI_AD 112也包括一个减振功能部件（JRF－jitter reduction function）116，其作用是测量在网络108上接收的包（或报头）上的振动并丢弃任何有过分振动的包/报头。

每个终端包括多个实体。对移动终端130给出更详细的解释，以便于理解网络中所有移动终端130和150的设计和操作，它们的设计和操作都是类似的。每个移动终端也可以起着压缩器/解压器的作用，与ANI 110和120以外通信，具体来说与其它网络通信。移动终端130包括一个RTP端点132，它是RTP包的源（传送器）和/或目的地（接收器）并且标识终端的IP地址、端口号等。移动终端130包括一个终端适配器（MS_AD）136，它执行报头压缩（要通过上传链路通信传送的包）和解压（通过下传链路通信接收的包）。因此，可以将终端适配器（MS_AD）136视为报头压缩器/解压器(收发器)137，类似于ANI_AD压缩器/解压器。术语MS_AD与AD有相同的意思。MS_AD 136也包括一个定时器134（接收器定时器），用于计算当前报头的RTP时间戳的近似值（或估算值）并测量在连续接收的包之间消逝的时间，以定位在向终端传送期间由诸如衰减(fading)的无线退化(wireless

degradation)引起的包损失。MS_AD 136可以用RTP报头中的额外信息来细算(refine)或更正时间戳近似值，如共同待审定专利申请号09/377,913所述的那样，该申请的申请日是1999年8月20日，被转让给同一个受让人，这里全文引用作为参考。可以根据RTP报头中提供的压缩时间戳更正或调整时间戳近似值。这样，就可以用一个本地时间和一个压缩时间戳来为每个RTP报头重新生成正确的时间戳。

RTP包，包括有压缩报头的包和有未压缩报头的包，是通过数据链路（诸如无线链路140）在网络中传送的，网络诸如是但不限于图2的示例性网络，数据链路的带宽非常宝贵，错误也并非鲜见。本发明并不仅限于无线链路，而是适用于范围很宽的各种链路（包括有线链路等等）。

图3概念性地表示压缩上下文信息和例子。压缩上下文信息是被压缩器为产生压缩报头而用作压缩算法的一个输入的报头中的任何类型的信息集合、信息、子集或信息子集的表示，可以从一个实体传送到另一个实体。另一个输入来自要被压缩的报头的报头源。

图4概念性地表示解压上下文信息和例子。解压上下文信息是被解压器为产生解压报头而用作解压算法的一个输入的报头中的任何类型的信息集合、信息、子集或信息子集的表示，可以从一个实体传送到另一个实体。另一个输入来自要被解压的报头的报头源。

压缩上下文信息和解压上下文信息二者都是动态的，就是说，它们可以被压缩器和解压器分别更新。更新的频率取决于报头压缩方案。可能导致在压缩器对压缩上下文信息更新的事件包括压缩一个输入报头、或接收来自解压器的反馈。可能导致在解压器对解压上下文信息更新的事件包括解压一个输入报头。

自适应报头压缩(ACE)是一种用于强健报头压缩的一般性架构，能够被参数化，以考虑存在/不存在和反馈通道的性能特征。该架构包括三种基本操作方式：

*双向确定性方式：该方式用于有一个可被用来从解压器向压缩器传送各种类型的反馈信息的“良好定义的”逆向通道的情况中。来自解压器的这种反馈的一个例子是ACK（确认），该确认例如用来从一个较低的压缩状态前进到一个较高的压缩状态。通过经良好定义的通道接收ACK，压缩器获知某特定报头已经被接收。压缩器利用该信息来更

可靠、更高效地进行压缩。

*双向时机性方式：该操作方式用于存在一个逆向通道、但逆向通道是“松散地”定义的的情况中。“松散地”定义的通道可能并不是总是可用的，或者可能是速度慢/不可靠的。有许多是两路双向的应用。一个主要的例子是谈话声音或图象。在这种情况中，内在地有一个能传送反馈的逆向通道。

*单向方式（悲观的或乐观的）：该操作方式在没有任何逆向通道时被使用。由于根本没有来自解压器的关于其当前状态的反馈，压缩器必须不时地向解压器发送某种刷新信息，刷新信息可在如果发生不正常情况时被用来重新建立同步。根据各种因素（例如通道状态），压缩器可能已知该方法在该方式中是悲观的（更频繁的刷新）或乐观的（不太频繁的刷新）。此外，有的事件能触发解压器发送 FH 信息，用来刷新解压器和减少不正确解压的机会。

ACE 压缩器的特点可以总结为在一系列状态中前进。当压缩器有足够的信心相信解压器已经接到某信息时，压缩器离开一个较低压缩状态，并进入一个较高的压缩状态。

在 RTP 报头压缩的情况中，这些状态是完整报头、第一级和第二级状态。

*完整报头(FH)状态：压缩器在初始化时或者在发生某个例外事件时(CPU 崩溃或存储器丢失)进入这个状态。在这个状态中，压缩器基本上向解压器传送 FH 报头信息。FH 报头含有所有报头字段的全部集合，加上某种额外的信息。这个信息可包括压缩器/解压器特定的数据，诸如 CID(用来区别多个流的上下文标识符)。 压缩器处于该状态中，一直到其获得足够的信心相信解压器已经接到 FH 报头信息。FH 包对于传送来说是最不优化的，因为其规模很大（例如对 IPv4 来说至少 40 个字节，对 IPv6 来说至少 60 个字节）。 当压缩器有足够的信心相信解压器已经接到 FH 报头时，就离开这个状态，进入 FO 状态。这个信心可能来自例如对来自解压器的 ACK 的接收或发送预定数量的 FH。

*第一级(FO)状态：压缩器在其离开 FH 状态之后，当一个新串开始时，进入这个状态。在这个状态中，压缩器基本上传送 FO 报头信息。FO 报头含有压缩器/解压器特定的字段，以及描述在实质变化的(essential changing)字段中已经发生的不规则变化的信息。压缩器

处于该状态中，一直到其获得足够的信心相信 FH 报头信息已经被解压器接收。这个信心可能来自例如对来自解压器的 ACK 的接收或发送预定数量的 FO。

*第二级(SO)状态： 当要被压缩的报头符合一个串的模式，并且压缩器有足够信心相信解压器已经获得该串模式时，压缩器处于这个状态。在这个状态中，压缩器传送 SO 报头信息。 SO 报头基本上只含有一个序列号，压缩器/解压器的最佳操作方式是接收/传送 SO 包，因为它们的规模最小（数量级仅在若干位）。

总之，一个会话开始时，压缩器处于 FH 状态。在该阶段，压缩器向解压器发送一个完整报头，以在解压器中建立上下文。这启动一个串。压缩器然后进入 FO 或 SO 状态。在 FO 状态中，它向解压器发送必要的基本变化的字段更新信息。在 SO 状态中，它向解压器发送最少的信息。解压器根据在前面的 FH 和 FO 包中交换的信息进行简单的插值，直到该串结束。当另一个串开始时，压缩器再次进入 FO 状态，重复该过程。

双向传送方式利用确认来实现各种功能:

*通知压缩器 FH 信息已经被收到；在这种情况下，压缩器知道解压器已经获得解压 FO 报头所需的信息，因此压缩器就能可靠地转换到更高的压缩状态 FO； 这种 ACK 被称作 FH-ACK。

*通知压缩器 FO 信息已经被收到；在这种情况中，压缩器知道解压器已经获得解压 SO 报头所需的信息，因此压缩器就能可靠地转换到更高的压缩状态 SO； 这种 ACK 被称作 FO-ACK。

*通知压缩器一个有特定编号 n 的报头已经被收到；在这种情况中，压缩器知道解压器能确定序列号，不会有计数器绕回到报头编号 n+seq_cycle 而产生的歧义，其中 seq_cycyle 是计数器循环；压缩器能可靠地将 k 个位用于报头序列号，不用担心在解压器有歧义的或不正确的序列号解码； 这种 ACK 被称作 SO-ACK。

通过确认对从 FH 到 FO 到 SO 状态的转换的控制，保证没有错误传播。就是说，没有被错误地接收的压缩报头总是能被正确地解压，因为同步总是没有失去。

至于解压器何时发送确认以及发送的频率，有很多灵活性。ACE 对于丢失或延迟的 ACK 也极有弹性。压缩器不断地根据当前要被压缩

的信息以及所接收的 ACK 来改变其压缩策略。例如，FO-ACK 的丢失或延迟回导致压缩器在 FO 状态中保持更长时间。 SO-ACK 的丢失或延迟回导致压缩器发送更多的用于序列号的位，以防止由计数器绕回引起的在解压器上的任何不正确的解压。

ACK 可以周期性地或不定期地被传送。不定期的确认的频率可以从一个周期性速率降低或者提高。ACK 可以以较低的频率发送，因为 ACK 由于错误或网络堵塞而丢失，或者由于逆向通道断断续续地可用或某些解压器的状态而不能传送 ACK。 ACK 也可以以比传统的周期性 ACK 更紧密的间隔传送。例如，当逆向通道的负荷非常轻并且可用时，解压器可以更经常地传送 ACK，因此解压器能更有效和更可靠地操作。

因此，用于传送 ACK 的反馈通道可以有非常松散的要求。这是因为 ACK 只对压缩效率有影响，对正确性没有影响。 ACK 的延迟或丢失可能引起压缩报头的大小的增加，但是即使在这种情况中，该增加也是对数级的。

在双向确定性方式中，从 FH/FO 到 FO/SO 的转换是以确认为根据的。在单向方式中，从不发送 ACK，所以在向 FO/SO 转换之前发送的 FH/FO 包的数量取决于一个预定的或动态地/适应性地选择的值。在双向时机性方式中，ACK 可能迟到，所以从 FH/FO 到 FO/SO 的转换不是严格地以 ACK 为根据的，而是取决于下述哪个在先：1）传送一个预定的或动态地/适应性地选择的 FH/FO 数量；2）接收至少一个 ACK。

总之，要在转换到 FO/SO 状态之前发送的 FH/FO 包的数量取决于在转换之前需要一个 ACK 和/或可以预定或动态地/适应性地选择的可调参数 m。下面讨论四种情况。

图 15 表示压缩器的压缩状态只根据适当的 ACK 的到达而作的转换。一序列的至少一个特定状态的报头（图中所示的是 FH 或 FO 报头，但不限于此）被传送到解压器。解压器接收第一报头 FH(n) 或 FO(n)（可以不是第一报头的报头）后，回传一个 ACK(n)，该确认被接收后，使压缩器转换到一个更高的压缩状态，如图中所示的 FO(N+i+1) 或 SO(N+i+1)，它是紧接着 ACK(n) 的接收之后被传送的包。

图 16 和 17 表示压缩器的压缩状态根据参数 m 而作的转换，该参数是被传送的报头的数量或者 ACK（每当 m 个 FO/FH 报头被传送或者 ACK 被接收时，发生转换）。图 16 的实施例不仅仅限于由压缩器进

行的 FO/FO/SO 报头传送。在图 16 中，ACK(n)在所传送的 FO/FH 报头数达到预定的被传送 FO/FH 报头数 m 之前到达，这导致压缩器转换到一个更高的压缩状态，并传送包 FO(N+i+1)或 SO(N+i+1)。 图 17 的实施例不仅仅限于由压缩器进行的 FO/FO/SO 报头传送。在图 17 中，ACK 在 m 个报头被传送之后到达。

图 18 表示压缩器的压缩状态在发送设定数目的报头后没有任何确认就发生的转换。

在不同的方式下，压缩器和解压器的操作策略不同。

压缩器的操作方式

*严格地基于 ACK 的：在这个方式中，压缩器严格地依赖对 ACK 的接收。如果 ACK 由于丢失、通道可用性、解压器状态等原因而没有准时到达，压缩器就通过例如增加编码字段的长度而转换到一个压缩程度较低的方式，并且它在接收适当的 ACK 之后只能转换回到一个进一步压缩的方式。

*松散地基于 ACK 的：在这个方式中，ACK 被收到时，有助于改善效率和可靠性，但是压缩器并不严格地依赖对 ACK 的接收。如果压缩器接收到适当的 ACK，它就从一个压缩程度较低的状态转换到一个进一步压缩的状态，或者，如果它已经在处于该进一步压缩的状态，则保持在当前状态。如果它没有接收到 ACK，它就根据其它准则，从一个压缩程度较低的状态转换到一个进一步压缩的状态，该准则例如是发送一定数量的压缩程度较低的报头，而不是接收到 ACK。

*不以 ACK 为根据的：当没有逆向通道可用时，压缩器一般在这个方式中操作。在这种方式中，从一个压缩程度较低的状态向到一个更压缩的状态的转换不以 ACK 为根据。相反，压缩器可以在一定数量的压缩程度较低的报头被传送后从压缩程度较低的状态转换到更压缩的状态。这个数量可以是一个可调的参数。此外，有的事件能触发压缩器发送压缩程度较低的报头，以便刷新解压器和减少不正确解压的机会。

解压器的操作方式

*发送确定性 ACK：在这种方式中，解压器周期性地以及在接收到 FH/FO 包时发送 ACK。此外，解压器能在逆向通道负荷轻且可用时，比周期性地更频繁地向压缩器发送回 ACK。

*发送时机性 ACK：在这种方式中，解压器不是严格地按周期发送 ACK。它只在有机会发送 ACK 时才发送 ACK，例如在逆向通道可用于传输 ACK 时。

*无 ACK：在这种方式中，解压器根本不发送 ACK。

报头压缩和解压方案对其有用的一例应用是在蜂窝系统上传送 IP 语音(VoIP)(或 IP 电话)的情形。当 VoIP 被应用到蜂窝系统中时，重要的是使 IP/UDP/RTP 报头的开销最小化，因为无线或空中(RF)接口的带宽有限。在这样的系统中，例如 ANI_AD 把 IP 网络接合到一个运行 RTP/UDP/IP 的计算机终端(例如终端 130)，该终端有一个用于通过无线或 RF 链路接收 RTP 包的蜂窝或 RF 接口。这只是本发明的压缩/解压技术的一例应用。

定义

n：由压缩器分配的含在报头中的序列号。序列号 n 对每个新包按 1 递增，并且独立于 RTP 序列号。注意 n 可以或者以 k 位编码(=(CD_SN)k)，或者以扩展的 1 位编码(=(CD_SN)1_extended 位)。

CD_SN 139：对应于 n 的内部计数器。压缩器和解压器维持它们的计数器。内部计数器的大小被选择得足够大，以避免对会话持续时间来说有任何含糊，例如选择 32 位。

(CD_SN)k：CD_SN 的 k 个最低有效位。(CD_SN)k 通常在压缩报头中发送。

(CD_SN)1_extended：CD_SN 的 1_extended 个最低有效位。(CD_SN)1_extended 在扩展方式(extended mode)下在压缩报头中被发送。

S_RFH：已知其报头被解压器正确地重构的包的 CD_SN；S_RFH 被压缩器根据来自解压器的反馈连续地更新。S_RFH 是以 k 或 1_extended 个最低有效位的形式被发送的。

N_lapsed：由压缩器维持的一个计数器，用于跟踪自最后被确认的包以来已经被发送的包的个数。N_lapsed=当前 CD_SN-S_RFH，如果 S_RFH 总是被压缩器在收到一个 ACK 时设定得等于最后被确认的包。

R_RFH：在解压器的基准包的 CD_SN，它被设定为在 FO(n,S_RFH)包被收到时的 S_RFH。

SN(n)：由压缩器发送的第 n 个包的 RTP 序列号。如果压缩器不重

新排序，SN(n)不一定是单调递增的，其中 n 是由 k 或 1_extended 个位的位值定义的。

TS(n)：由压缩器发送的第 n 个包的 RTP 时间戳。

CFO(n)：在包 n 的当前第一级差。注意这是一个向量，其每个分量等于包 n 与包(n-1)中对应字段之间的差；例如，CFO(n)的时间戳分量的计算式是 TS(n)-TS(n-1)。

FO_DIFF(n_2, n_1)：包 n_2 相对于包 n_1 的当前第一级差，其每个单独的字段等于包 n_2 与包(n_1)中对应字段之间的差；例如，FO_DIFF(n_2, n_1)的时间戳的计算式是 TS(n_2)-TS(n_1)。

N_Last_Interr：对应于最近的中断的 CD_SN（即非线性变化）。每当 CFO(n)!=CFO(n-1)时，它就被（压缩器）更新为 n。

S_DFOD：在发送器的缺省第一级差。S_DFOD 是一个规定当前模式的向量。S_FDOD 被压缩器用来确定当前报头信息是否符合该模式。如果报头(n)=报头(n-1)+S_DFOD，则当前报头 n 符合该模式。当该模式不从一个串变到下一个时，S_DFOD 是静态的。否则，压缩器必须动态地确定 S_DFOD。

TS_DFOD 和 SN_DFOD：S_DFOD 中分别对应于 RTP 时间戳和序列号的分量。

R_DFOD：在发送器的缺省第二级差。R_DFOD 是一个规定当前模式的向量。R_FDOD 被解压器用来解压 SO 报头。当该模式不从一个串变到下一个时，R_DFOD 是静态的。否则，解压器必须动态地确定 R_DFOD。按照设计，在 SO 状态期间，S_DFOD 和 R_DFOD 总是相等。

Extended_flag：由压缩器维持的一个标记。如果它的值为真(TRUE)，则 1_extended 个位被用于 CD_SN，其它的序列参数将在报头中被发送。否则，(CD_SN)k 将被发送。注意该标记本身也在报头中传输，所以解压器知道所使用的是哪种 CD_SN 编码。

Header(n)：用来表示由压缩器发送的报头信息的统称。Header(n)可以以 FH(n)、FO(n, S_RFH)、SO(n)等各种形式被发送，这取决于压缩器 的状态。注意在报头中，n 实际上被编码为(CD_SN)k 或(CD_SN)1_extended，这取决于扩展标记(extended flag)。

Diff(n_2, n_1)：按(CD_SN)k 或(CD_SN)1_extended 编码的 n_2 与 n_1 之间的真实距离。Diff(n_2, n_1)=CD_SN(n_2)-CD_SN(n_1)，其中 CD_SN(n_2)

和 CD_SN(n_1)是分别对应于 n_2 和 n_1 的 CD_SN。例如，如果第一个包携带(CD_SN)k=4，第二个包携带(CD_SN)k=1，则真实距离是(16+1-14)而不是(1-14)=-13。

FH_Req：由解压器发送的要求压缩器在 FH 状态中操作的请求。例如在解压器刚刚从某次崩溃中恢复，不再有可靠的状态信息时，使用这个请求。

ACK(n)：是响应报头(n)而被发送的。ACK 的意思是包(n)被正确地接收。

seq_cycle: seq_cycle-1 是序列号所能达到的最大号，序列号达到最大号后就绕回到 0。 seq_cycle=2^k。

SEQ Ext_cycle: $2^{l\_extended}$。

HSW 117：压缩器保持一个滑动窗口(sliding window)，内含向解压器发送的报头。

(HSW)：header(n)、header(n+1)、header(n+2)等等。各报头可能已经被以 FH、FO 或 SO 的形式发送。当压缩器接收到一个 ACK(n)时，它将删除 $p<n_2-n_1$ 的 Header(p)并且把 Header(p=n)移到 Header(S_RFH)。静态字段不需要以多个项目的形式在 HSW 中存储。只需要静态字段的一个拷贝。注意在 HSW 中，每个报头都用一种颜色标记，或者是绿色或者是红色。

header(packet) 的颜色：如果 header(packet)中携带的是以 k 位编码的 CD_SN，其为绿色；否则是红色，例如所携带的是 l_extentded 位编码的 CD_SN 时。在实现中，可以用一个 1 位的标记来存储颜色。该颜色被用来辅助压缩器在接收 ACK 时唯一地标识报头。

OAW 135：解压器保持一个滑动窗口，内含其已经成功解压并确认过的报头： header(n_1)、header(n_2)、header(n_3)。注意不能确切地知道 ACK 被压缩器接收。该窗口将被称作未决 Ack 窗口(OAW)。 静态字段不需要以多个项目的形式在 HSW 中存储。只需要静态字段的一个拷贝。

R_Last_Decomp：被解压器最后重构的包的 CD_SN。解压器保持对应的完整报头 FH(LAst_Decomp)。

注意：

*除了下述变化之外，原始 IP、UDP 和 RTP 被传送。

*IP 报头的总长度字段(2 字节)被 extended_flag (1 位)、seq_num(4 或 8 位)和其它可选信息替换。如果 extended_flag 等于 1，seq_num 的长是 8 位，即该包被以扩展的方式发送。

*UDP 报头的长度字段(2 字节)也被用于报头压缩的上下文 ID 即 CID 替换。压缩器能同时彼此独立地压缩多个 RTP 流。每个流都被分配一个独有的 CID。在实现中，CID 可以是 1 字节或 2 字节长的，这取决于任何给定时刻 RTP 流的最大数。

本发明的可以在图 2 的系统中实施的第一实施例，增加了接收 FH 和 F0 类型的包的概率，以便能及时地进入最优的 S0 状态。接收的概率可以通过对 FH 和 F0 包应用下述形式的额外错误保护而得到增加:

*错误纠正码

*错误纠正码+交错(interleaving)

*FH 和 F0 报头数据的复制传送(duplicate transmission)

本发明的第一实施例针对在有损耗传送介质（例如无线）上的数据包的丢失提供额外的保护，而不向压缩器和解压器分配额外的通道资源。通过按如图 5A 和 5B 中所示的时间间隔 T 延迟数据流的传送，获得额外的带宽，使得有足够时间用于发送额外的数据包。任何时刻可用该延迟间隔来发送初始的 FH 包和 F0 包。

初始 FH 包的传送: 假设通信通道上的带宽已经按主要将传送 F0 和 S0 包的假设（极少的 FH 包被传送）而作了分配。在多数情况中这是一个合理的假设，因为压缩方案作为整体的有效性依赖于主要是 S0 状态中的操作。然而，在会话开始时，总是需要传送一定数量的 FH 包（理想情况下仅一个）。FH 包的大小显著地大于 S0 和 F0 包。这意味着要在为 S0 和 F0 包分配的相同时间内传送该包，就需要额外的带宽。FH 包的部分的传送在主要通道上发生，其余的某个二级通道上发生。

前面提到的如图 5A 和 5B 中所示的时间间隔 T，被用来传送在 FH 包内携带的额外的数据。这样做就消除了对二级通道的需要。节约是显著的，因为二级通道可能是（就协议来说）困难的、(如果需要是一个只是不时使用的专用通道的话）耗费大的、和（如果需要是一个共享的基于包的通道，则要遵守不希望有的竞争延迟）建立速度慢的。注意到根据 T 的值和传送块的大小，也有可能在“额外带宽”中包括错误纠正编码(ECC)。

F0 包的传送：该方案仅当传送 F0 包的需要与数据流传送中的一个停顿巧合时适用。因为按照定义，F0 包是在正常数据流中有中断的时候被生成的，这个条件几乎总是得到满足。例如，静音的间隔导致 RTP 时间戳中不规则的跳变(jump)，这会触发 F0 包的传送。

传送器/接收器（图 2 的压缩器/解压器）在采用额外的错误保护的情况下能在更理想的状态中操作。在现有技术中，在理想的 S0 状态中的操作不太频繁。

这种增加的在理想状态中的操作的代价是在数据流中总是出现的延迟。然而，这个延迟在特定于所传送的数据的限度内是容易能容忍的。

例如，假设语音数据+F0 类型的报头数据正好与大小 p 字节的蜂窝传送块相配，对应于等于 20mS 的时间间隔。假设以上述方式应用一个 'T'=20mS 的延迟，就有可能发送该 F0+语音数据+等量的数据以增加接收的概率。一些潜在的配置是：

*语音数据+F0 类型的报头+1/2 速率回旋码位(rate convolutional code bits)被发送，以增加数据接收的概率。这显著地增加了正确接收的概率。一个 1/2 速率代码完全利用该延迟所提供的额外带宽。

*可以发送语音数据的两个相同的拷贝+F0 类型的报头；这增加了数据接收的概率，尤其是传送介质遭受随机的包损失时。

本发明的可以在图 2 的系统中实施的第二实施例，根据的是多数时间或者恒定或者能被以线性方式插值的 IP/UDP/RTP 字段。在这些实例中，压缩报头只携带一个非扩展的有 k 个 2 位的序列号，它为线性插值提供了足够的信息。象 RFC 2508 一样，本发明在线性插值导致不正确的报头重构时，发送器传送 F0 差别信息(difference information)。与 RFC 2508 不同的是，该差别信息不是相对于当前包紧邻的前一个包，而是相对于一个已知被正确接收的基准包计算出来的。这个特点保证当前报头能被可靠地重构，即使一个或多个过去的包被丢失了。因为能以该方式重构报头，所以不需要发送完整报头。基准包是由接收器的压缩器按照从解压器接收的确认而确定和更新的。

压缩器用一个如图 6 中所示根据所接收的确认(ACK)知道被正确解

压的报头作为基准报头。压缩器发送一系列的完整包FH(n)...FH(n+i+1)...，它们覆盖的时间序列刚好到由接收完整包FH(n)的压缩器生成的确认ACK(n)被接收的时间t2之前。从FH到FO（所示的FO(n+1)）的转换是同步的 。至t2的时间序列依赖于往返无线电传送时间。

图7表示从传送第一级包FO(FO(n))到FO(n+i+1)的转换，其中至少一个确认ACK(n)以及可选地ACK(n+1)被解压器生成，生成之后，压缩器同步地转换到第二级包SO(SOn+j)，转换具有一个至时间t2的无线电延迟，这是进行所要求的。压缩器从FO状态转换到SO状态之前所需的ACK数取决于包与包之间的变化。例如，如果包与包之间的变化是线性的，具有常量的参数，则从FO状态转换到SO状态只需要一个ACK。为了能够解压当前的报头，解压器只需要知道基准报头，而不是象RFC 2508一样要知道紧邻的前一个报头。在这个方案中，压缩器在初始化时发送完整报头（FH类型的报头）。它在线性插值不适用时发送第一级差别信息（FO类型的报头）。然而，压缩器不传送SO差别报头，直到在先的FH或FO报头被确认并且线性插值适用。

在压缩器上，基准包被定义为是其ACK已经被接收的最后一个包。具体来说，当发送器接收到对包n的确认ACK(n)时，它将提取以前在存储器中存储的包n，将其作为基准包保存起来（包n被发送时被存储在存储器中）。之后，所有的报头压缩都参照该包来进行，一直到接收到一个新的ACK，此时，刚刚被确认的包变成新的基准包。存储着潜在的基准包的发送器中的存储器被称作被发送报头窗口（HSW），该窗口在图2中被表示为实体117。如果要在压缩报头中发送某个FO差别信息，压缩器将明确地加入基准包的序列号（基准序列号）。

在解压器，当一个ACK(n)被发送时，header(n)被存储在未决确认窗口（OAW）中，该窗口在图2中被表示为实体135。之后，当收到一个FO差别信息时，解压器将根据基准序列号确定基准包，从OAW135提取对应的基准包，用其来重构完整报头。

本发明利用的在解压器生成的时间戳的线性，密切地符合一个是日常时钟的函数的线性模式。根据在接收器的解压器保持的定时器134并用由扩展的1位所定义的FO包的扩展序列号，所有在阀值内的长损失都能被检测到并被恢复。

对于所假设的语音来说，如果连续的语音样本和包之间的时间间隔是 t 毫秒(msec)，则（在时间 n*t 毫秒生成的）报头 n 的 RTP 时间戳，等于（在时间 0 生成的）报头 0 的 RTP 时间戳加 TS_stride*n，其中 TS_stride 是一个与发送器的声源的声音编解码器有关的常数。因此，由解压器接收的报头中的 RTP 时间戳遵循一个是时间函数的线性模式，但是紧密程度比压缩器低，这是由于压缩器与解压器之间的延迟振动。在正常操作中（没有崩溃或故障），延迟振动是有限的，以符合谈话实时通信的要求。

一个串以一序列的都符合某特定模式的报头的形式出现。具体来说，RTP 序列号(SN)从一个报头到下一个报头按 1 递增。RTP 时间戳(ST)不降低，并且遵循某个可预测的模式：如果报头 n_1 和 n_2 在相同的串中，则报头 n_2 的 TS 能从报头 n_1 的 TS 和模式函数推导出来。其它字段的值--可能除了 UDP 校验和与 IP Id 之外，在串中都不改变。因此，一旦某报头（例如 n_1）已经被正确地解压，同一个串中的随后的报头就能按照该模式通过插值而被解压。一旦压缩器确定某报头已经被成功地解压，并且该模式已经被解压器获得，压缩器只要传送一个 k 位的序列号-记为(CD_SN)k，当作同一个串中随后的包的压缩报头。(CD_SN)k 是一个较大(压缩器)解压器序列号(CD_SN)的 k 个最低有效位。

在该方案中，解压器用定时器 134 或另一个没有在图 2 中示出的定时器来获得在两个连续接收的包之间消逝的时间。根据这种计时信息和由压缩器所用的特定压缩规则以及序列号，解压器就能检测到长损失和/或从长损失恢复。

设

*HDR(i)是有短序列号 i 的报头

*HDR(n_1)是已经被成功地解压的最后一个报头

*HDR(n_2)是紧接着 HDR (n_1)后被接收的报头

*TS_stride 是每隔 t 毫秒的 RTP TS 增量

*SEQ_CYCLE 是 HDR 中所用的序列号的循环

*Diff(n_2, n_1)：是有序列号 n_2 的包与有序列号 n_1 的包之间的差。

作下述定义：

*当 $n_2 > n_1$ 时，Diff(n_2, n_1)=n_1

*当 $n_2 \leqslant n_1$ 时， $Diff(n_2, n_1)=n_2+2^k-n_1$

在紧接着接收 HDR (n_1) 之后接收到 HDR(n_2) 时，解压器确定在这两个包之间是否发生过长损失，即是否有 $Diff(n_2, n_1)$ 个包丢失或者 $SEQ\_CYCLE*p+Diff(n_2, n_1)$ $(p>1)$ 个包丢失。该检测方案也依赖 HDR(n_2) 的类型。根据在报头压缩方案中所定义的三种报头类型，列举以下的 2 种情形。

情形 1: HDR(n_1)，SO(n_2)

情形 2: HDR(n_1)，FO(n_2)

情形 1 的长损失检测和恢复

为检测情形 1 中是否有长损失，在压缩器中维持一个定时器(例如定时器 134)，每当一个包被接收时，定时器就被检查并更新。

设

*T 为 HDR(n_1) 被接收时的时间

*T+ΔT 为 HDR(n_2) 被接收时的时间、

仅当一个或多个 FH 或 FO 包已经被确认，并且适合从确认的报头线性插值时，压缩器才发送 SO 包。因此，如果 HDR(n_2) 是 SO，并且不管 HDR(n_1) 是什么类型，都适合从 HDR(n_1) 向 HDR(n_2) 线性插值。这基本上意味着从包 n_1 至包 n_2 的 RTP 时间戳和 RTP 序列号在压缩器上生成的时候，都密切地遵循一个作为日常时钟的函数的线性模式。因此，进入到解压器的报头也遵循一个是时间函数的线性模式，但是有因为发送器与接收器之间的振动引起的波动。

在振动上限总小于 ($SEQ\_CYCLE*t$) 毫秒的假设下，以下规则适用于检测长损失:

[规则 1]

· 如果 ($DIFF(n_2, n_1)$)*t 毫秒 $\leqslant \Delta t <$ ($DIFF(n_2, n_1)+SEQ\_CYCLE$)*t 毫秒)，则只有 $DIFF(n_2, n_1)$ 个包丢失;

· 如果 ($DIFF(n_2, n_1)$ +SEQ_CYCLE)*t 毫秒 $\leqslant \Delta t <$ ($DIFF(n_2, n_1)+2*SEQ\_CYCLE$) *t 毫秒)，则 ($SEQ\_CYCLE+DIFF(n_2, n_1)$) 个包丢失;

· 一般地，如果 ($DIFF(n_2, n_1)+i*SEQ\_CYCLE$)*t 毫秒 $\leqslant \Delta t <$ ($DIFF(n_2, n_1)+(i+1)*SEQ\_CYCLE$) *t 毫秒)，则 ($i*SEQ\_CYCLE+DIFF(n_2, n_1)$) 个包丢失;

为了从这种长损失恢复 RTP 时间戳和 RTP 序列号，只需要丢失包

的个数。

设

· N_{loss}是在包 n_1 与包 n_2 之间丢失的包数，可通根据规则 1 获得。

· TS(i)和 SEQ(i)是包 i 的 RTP 时间戳和 RTP 序列号。

该 RTP 时间戳和该 RTP 序列号可由包 n_1 的 RTP 时间戳和 RTP 序列号以及 N_{loss} 按下述公式获得:

$TS(n_2)=TS(n_1)+N_{loss}*TS\_STRIDE$

$SEQ(n_2)=SEQ(n_1)+N_{loss}$

图 8 表示本发明在情形 1 中检测长损失并从长损失恢复的实施例，该实施例检测在发生损失中的绕回时的长损失，其中损失的包数多于 2^k，k 包的序列号 SN(n)中的位数。假设解压器接收到在时间 t0 发送的 HDR(n)，而在时间 t1 至 t3 之间发送的所有包丢失，然后解压器接收到在时间 t3 发送的 SO(n+1)。由于在时间 t3 发送的包 n+1 是一个 SO 包，这表示从时间 t1 至 t3 发送的所有包都落入相同的串，即这些包的 RTP 序列号 Sn(n)遵循一个是日常时钟的函数的线性模式，所以在时间 t1 至 t3 之间丢失的包数能被解压器中维持的定时器 134 检测到。过程如下:

假设在连续语音样本和包之间的时间间隔是 t 微秒，解压器在接收到在 t0 时发送的 HDR(n)时启动本地定时器 134（具有值 t_s），然后定时器根据日常时钟递增。当在时间 t3 时发送的 SO(n+1)在接收器被接收时，定时器就达到 t_0，由于振动，t_0 大约等于 $t_s+(2^k+1)t$。由于 t_s 与 t_e（在 $(2^k+1)t$ 左右）之间的时间差大于 t 微秒，在 t0 时发送的包 n 和在 t3 时发送的包 n+1 将不是连续被发送的包，在这两个包之间发生了长损失。此外，由于 t_s 与 t_e 之间的时间差小于 $(2^{k+1}+1)t$，因此将没有一个序列循环的包损失。因此，解压器能得出有 2^k 个包丢失的结论。

情形 2 的长损失检测和恢复

长损失检测和恢复方案不能应用于在 SO、FO 或 FH 报头后收到 FO 报头的情形 2。在这个情况中，即使包 n_2 与 n_1 之间的时间差 Δt 大于 $(DIFF(n_2,n_1)+i*SEQ\_CYCLE)*t$ 毫秒，也不意味着发生过长损失，其原因可能是压缩器的静音期。在这个情况中，根据在 FO 报头中提供的信息 ，RTP TS 可能被成功地重新生成，但 RTP 序列号却没有，因为

仅根据定时并不能区分长损失或静音。为了解决这个问题，对所有包属于相同串的 FO 系列内的第一 FO 包使用一个有 l_extended 个位的（l_extended>k 非扩展位）的扩展序列号，直到有从解压器发回的对 FO 的确认。这个方案在长损失的上限小于 $2^{l\_extended}*t$ 微秒的假设下检测长损失并从其中恢复。

为了实现该方案，由压缩器维持一个对包的内部计数器 CD_SN 139。CD_SN 139 为从发送器的压缩器发出的每一个包计数。在修改的完整报头和压缩报头中的序列号只是 CD_SN 139 的非扩展的 k 或 l_extended 个最低有效位的一个快照(snap shot)。根据报头压缩/解压方案，解压器能重构所接收包的当前绝对数。

设

CD_SN_LAST 是所接收的最后一个包的绝对包号，即包 n_1;

CD_SN_CURR 是当前 FO 报头的绝对序列号，即包 n_2; 和

DIFF(CD_SN_CURR, CD_SN_LAST) 被定义为 (CD_SN_CURR-CD_SN_LAST)。

根据以下规则，能成功地检测到在上限 $2^{l\_extended}*t$ 微秒内的长损失。

[规则 2]

如果 DIFF(CD_SN_CURR, CD_SN_LAST)>2^k，则检测到一个长损失。

丢失的包数 N_{loss} 可用下式计算:

N_{loss} = DIFF(CD_SN_CURR, CD_SN_LAST) = (CD_SN_CURR-CD_SN_LAST)。

根据 N_{loss}，RTP 序列号就能被解压器按以下公式重新生成，而 RTP 时间戳就能被根据基准报头和对应的德尔塔(delta)而重新生成。

$SEQ(n_2)=SEQ(n_1)+N_{loss}$

由于扩展的序列号因为位数更多而比普通序列号使用更多的带宽，所以不建议频繁使用扩展的序列号，而是建议用于第一 FO 包，直到对该系列的 FO 的确认返回到压缩器。

应当注意到该方案不能检测到长于 $2^{l\_extended}*t$ 微秒的长损失并从其中恢复；因此应当仔细地选择 l_extended，以便能检测到不能被从协议栈的较低层指示的长损失。如果长损失长于 $2^{l\_extended}*t$ 微秒，则要求某较低层能提供这种极端长的损失的指示，以及应用在较低层的

断连接或其它长损失恢复方法，例如向压缩器发送请求。

当压缩器需要发送一系列 FO 类型的包，而所有包属于相同串时，压缩器使用带 l_extended 个位的序列号，一直到从解压器返回对应该系列包的至少一个确认。但是当需要发送 SO 包时，解压器只使用 k 位的序列号。

在压缩器上，定时器 134（或另一个未在图 2 中示出的定时器）每当有包到达时就被启动。定时器一直运行到下一个包到达。如果输入包是 FH 类型的，则不需要定时信息，定时器应当复位。

如果输入包是 FO 类型的，就用规则 2 来检查是否发生过长损失，如果发生过，则要用 FO 包传送的对应恢复方案来从长损失恢复。不需要定时信息，定时器应当复位。

如果输入包是 SO 类型的，就应根据定时器来计算当前包与以前接收的包之间的到达时间差，并用规则 1 来检查是否发生过长损失，如果发生过，则要用 SO 包传送的对应恢复方案来从长损失恢复。定时器 134（或另一个定时器）应当在检查后立即复位。

总之，对定时器 134（或另一个定时器）和第一 FO 包的扩展序列号的使用，能成功地检测到某阈值（$2^{l\_extended}*t$ 微秒）范围内的长损失并从其中恢复，改进强健性，而不增加太多开销和复杂性。

本发明的用图 2 的系统实践的另一个实施例，利用的事实是 IP/UDP/RTP 字段或者是恒定的或者能根据可预测的模式被插值。在这些实例中，压缩报头只携带有 k 个位的为插值提供足够信息的非扩展序列号。当插值导致一个或多个字段的不正确解压时，压缩器发送用于这些字段的额外信息（更新信息）。

为提供对错误和错误突发(error burst)的强健性，压缩器针对已知被正确解压的基准报头编码该更新信息。压缩器在接收到对应的确认时知道报头被正确地解压。确认机制保证即使有一个或多个报头丢失，当前报头也能被可靠地重构。

串被定义为一序列都符合某特定模式的报头。RTP 序列号(SN)从一个报头至下一个报头按 1 递增。RTP 时间戳(ST)是非递减的，并且遵循某种可预测的模式。如果报头 n_1 和 n_2 在同一个串中，则报头 n_2 的 TS 能从 n_2 与 n_1 之间的序列号偏差和 TS 的乘积以及模式函数导出。其它字段值，可能除了 UDP 校验和及 IP Id 外，在该串中都不变化。所

以，一旦某报头 n_1 被正确地解压，就可以按照该模式通过插值解压同一个串中后继的报头。一旦压缩器得到某报头已经被成功地解压、并且该模式已经被解压器获得的通知（如 ACK 所证明的那样），压缩器就发送一个序列号，作为同一个串中后继的包的报头。

就语音的情况而言，TS 有一个线性增加的模式。所以报头 (n_2) 的 TS 可以计算为：$TS(n_2)=TS(n_1)+TS\_stride*p$，其中 TS_stride 是两个连续报头之间的时间戳增量，p 是包 n_2 与 n_1 之间的序列号偏差。静音的间隔打破该线性关系，使一个串终止。新的串随新的短时谈话(talk spurt)而开始。

所以压缩器经历三个不同的阶段：初始化、更新和插值。会话以初始化阶段开始。在该阶段，压缩器向解压器发送完整报头，直到收到一个 ACK。完成初始化后，当某个串开始时，发送器的压缩器进入更新阶段，向解压器发送必需的更新信息。一旦压缩器接收到一个或多个 ACK，表明解压器已经获得进行插值所需的信息，压缩器就转换到插值阶段。在插值阶段，压缩器只发送序列号作为压缩报头，直到串结束。当另一个串开始时，传送器的压缩器进入另一个更新阶段，然后重复全部过程。在初始化、更新和插值阶段中发送的报头是 FH、FO 和 SO。FH、FO 和 SO 都携带一个序列号，压缩器发送的每个报头上的该序列号按 1 递增。SO 实际上只由该序列号组成。以下将响应 FH 和 FO 而发送的 ACK 分别称作 FH ACK 和 FO ACK。

长错误突发(error burst)和同步的丢失－周期性 ACK

如果用 k 位来编码上述序列号，序列号每隔 seq_cycle 个报头 $(seq\_cycle=2^k)$ 就绕回。因此，如果某错误突发持续的时间长于 seq_cycle 个报头的持续时间，解压器就不能无歧义地仅从序列号确定消逝的报头的数量，因此就不能进行适当的解压。为了解决这个回绕的长突发问题，要使用周期性的确认。图 9 表示本发明用 k 位序列号和 1 位扩展位号与由解压器每当检测到 2^k 个接收到的包时同步地生成的周期性确认(ACK)相结合的操作。假设压缩器在发送有 k 位序列号的 SO 包 n+i 之前接收到对包 n 的 ACK(n)，压缩器在发送包 $(n+i+2^k+N\_RT)$ 之前期待从解压器接收另一个对包 $(n+2^k+N\_RT)$ 的 ACK。假设 ACK $(n+2^k+N\_RT)$ 在传输中丢失，压缩器转换到使用 1 位的扩展状态并发送带有扩展的 1 位的序列号的包 $(n+i+2^k+N\_RT)$。当解压

器接收到带有扩展的 1 的序列号的包(n+2^k+i+N_RT)时，它向压缩器发送回 ACK(n+1+2^k+N_RT)。当压缩器接收到 ACK(n+1+2^k+N_RT)时，它转换到非扩展状态，并发送只带有 k 位的序列号的包 SO(N + j)。解压器被期待按一定的间隔发送 ACK，间隔的时间足够密，使得压缩器一般每隔 2^kseq_cycle 个报头接收一次 ACK。压缩器维持一个计数器 N_elapsed，用于跟踪自接收上一个 ACK 以来过去的包的数量。如果 N_elapsed>2^kseq_cycle 个报头，压缩器就在扩展方式中操作，在扩展操作中，有 l_extended 个位而不是更少的非扩展的 k 个位被用于序列号，其中 l_extended 个位的位数>非扩展的 k 个位的位数。l_extended 位的序列号和在非扩展的序列号中的缩减的 k 个位数可被看作为非扩展的序列号的最低有效位，l_extended 位是发送器的计数 CD_SN 139 的最低有效位。它们分别被记为 (CD_SN)k 和 (CD_SN)l_extended。N_elapsed 将对压缩器发送的每个包按 1 递增。序列号 SN 仅在压缩器从解压器接收到 ACK 时递增，并允许压缩器转换回正常方式，即用 CD_SN 的预定数量的最低有效位。如下所述，这些 ACK 可以是非周期性的，原因例如是发送的 ACK 的通道忙，要求以非周期性的方式延迟 ACK。

为计入往返延迟，接收器的解压器需要预计何时发送周期性的 ACK。解压器发送周期性 ACK 的时机，必须足够早，使压缩器至少每隔 seq_cycle 接收一次 ACK。考虑到往返时间，解压器必须至少每隔 (seq_cycle-N_RT)个报头发送一次 ACK。数 N_RT 是由解压器计算的对当前往返的估算值，可以根据最近的测量而动态地估算，也可以简单地设定为 RTT_n（下文作定义）。实践中，T_H 可以从发送器的编解码器特性导出，或者从观测到的实际间隔导出。

为了保证正确性并取得高性能，需要对压缩器与解压器之间通信通道(CD-CC)作出一些假设。通道可以是一个链路或多个链路的链接（网络或多个网络）。

经正向或逆向通道传送的包可能被丢失或破坏，但是它们的顺序被保持（即 FIFO 管）。将数量 MAX_EB 定义为在 CD-CC 上可能丢失的连续的包的最大数。在实践中，例如在蜂窝链路中，MAX_EB 被协议堆栈的较低层采用，协议堆栈的较低层在连续丢失的包达到阈值时决定断开连接。

通道可以在解压器上执行分段(fragmentation)或重装配,但是保留并提供正在传送的包的长度。注意此分段不同于 IP 分段。

该方案假设有一个检测压缩器与解压器之间的错误的机制。假设该通道提供该错误检测。如果不能从通道获得错误检测,可直接了当地扩展该方案,方法是在压缩器-解压器级上增加一个错误检测码。错误检测是有益的,不过是可选的。

压缩器与解压器之间的往返延迟被定义为发送和处理报头(n)至处理该报头和返回 ACK(n)的时间。为避免对被确认的原始消息的任何歧义,ACK 经历的往返延迟决不能长到正向中的(CD_SN)1_extended 已经绕回。合理的假设是,由于实时通信的要求,发送 header(n)和处理该报头的时间是有界的。返回 ACK(n)的时间取决于用于发送该确认的逆向通道。例如,如果该通道是基于竞争的,它可能经历排队延迟。假设较低层对 ACK 的传送采用一个延迟限度,方法是必要时丢弃那些在队列中停留的时间太长的 ACK。基于这些考虑,假设往返延迟有个上限: RTT_UB。此外,有一个额定往返延迟 RTT_n,它是正常操作期间最可能的往返延迟。显然,RTT_n< RTT_UB。对 RTT_n 的估计应当在实现之前作出,因为 RTT_n 被用来确定 k 的最佳值(见下文的解释)。注意在运行时,接收器需要估计实际的往返延迟。详细内容在下文讨论。

根据假设 1 和 4,为保障所提出方案的正确性,1_extended 的值应当满足以下条件:

1. $2^{1\_extended}$*T_H>=RTT_UB,其中 T_H 是两个连续报头之间的时间间隔;此要求旨在避免 ACK 的歧义

2. $2^{1\_extended}$ >MAX_EB;此要求旨在即使发生长突发时也维持序列同步

对 1 的值的选择,有某种灵活性。然而,为了取得最优性能,应根据通道错误突发(正向和逆向两个方向上)的分布和往返延迟而精确地选择。以下是一些要考虑的因素:

3. 2^k>RTT_n,这是为了减少压缩器因迟到的 ACK 而转换到扩展方式的机会。

4. 2^k>连续包丢失的最大可能数。这是为了减少压缩器因长错误突发而转换到扩展方式的机会。

5. K 不应太小。否则，从解压器发送到压缩器的周期性 ACK 就太多，导致逆向通道泛滥，降低压缩效率。另一方面，k 的值大时将导致每一个报头中携带的开销。

基本概念是，当通道条件恶化时，压缩器和解压器靠使用 (CD_SN)1_extended 来保障正确性。另一方面，在正常的通道条件下它将用较短的(CD_SN)k 位来获得最佳效率。下文讨论这两个方式之间的转换。

图 10 表示一个带宽缩减实施例，该实施例在由压缩器收到一个由解压器生成的、用于使压缩器把报头压缩转换到更高程度的报头压缩的确认之前从压缩器向解压器发送至少一序列的数据包，每个序列中的报头在不同的报头消费状态之间转换。因为压缩器在收到任何确认之前周期性地传送具有至少某种压缩的报头，在接收确认之前传送的位数较少，保存了带宽。在初始化阶段，压缩器可以但不限于在完整报头（FH）与第一级报头（FO）之间交替，即一组 FH，然后一组 FO，然后一组 FH，然后一组 FO，如此等等，直到接收到一个 ACK。各组都可以至少包括一个报头。解压器在正确接收到一个 FH 时发送一个 ACK。在图 10 中，交替的 FH 和 FO 报头是一个接一个被发送的，即 FH(0)，FO(1)，FH(2)，FO(3)，直到压缩器接收到包 0 的 ACK(0)，并从此用 FH(0)作为解压的基准报头。

多重插值

图 11 表示本发明的一个实施例，其中解压器进行多重插值。当一个或多个连续的属于同一个串的包被丢失或破坏时，解压器可以通过以必要的次数应用一个插值函数来解压后继的 SO 报头。如图 11 中所示，假设 SO(n+1)和 SO(n+2)都丢失。设 SO(i)中的对应变化值为 X(i)，则能重新生成 X(n+3)。如果插值函数是线性的，并且两个连续发送的包之间的偏差是 X_stride,则重新生成 X(n+3)的方法是在 SO(n)的基础上两次应用该插值函数，即 X(n+3)=X(n)+X_Stride*((n+3)-n)。

如果插值函数不是线性的，并且以非线性的模式变化，则解压器能按照该线性模式重新生成报头。例如，如果连续的包之间的 X 的变化值属于模式 X_Stride1、X_Stride2、X_Stride1、X_Stride2 等等，则解压器重新生成 X(n+3)的方法是 X(n+3)=X(n)+(X_Stride1 +

X_Stride2)。

X 的例子可以是 RTP 报头中的时间戳和序列号，具有多重的使用不同插值函数的插值。插值函数的函数的一个例子是不同常数与一个恒定插值函数之间的乘积。

每个到达发送器的解压器的报头，可以被模型化为一个多维向量，向量的每个分量等于报头中的一个变化字段。例如，如果报头中只有 RTP 序列号 SN 和 RTP 时间戳是变化字段，则每个要压缩的报头对应于二维空间中的一个点。因此，随着时间的推移压缩器简单地观察到该多维空间中的一序列的点。

点的坐标可以通过向序列中紧邻的前导点应用一个多维插值函数而导出。插值函数可以因不同的包而不同（或在该空间中因点而异）。然而，如果插值函数对点的一个子序列是相同的，则该子序列变成一个串。注意插值函数可以有任何特性，虽然插值函数通常是线性的。

串的概念能由解压器进行多重插值而进一步得到优化。当属于一个串的一个或多个连续的 SO 报头在压缩器与解压器之间丢失或损坏时，解压器仍然能通过以必要的次数应用插值函数而解压后继的 SO 报头。

该次数是由 CD_SN 的跳变确定的。注意 CD_SN 中的同步在计数器有绕回时被保持。

如果在传输过程中有一个或多个 SO 报头损坏，解压器能根据以前和当前正确接收的报头和插值函数修复损坏的报头。当解压器接收到带损坏报头的包时，它把损坏的包缓存起来而不是丢弃损坏的包。在解压器接收到下一个没有损坏的包时，解压器将其中缓存的包数与当前正确接收的包与上一个正确接收的包之间的序列号的偏差作比较。如果序列号的偏差与其缓存的包数相符，并且所有这些包都属于同一个串，则解压器就能根据已知的插值函数恢复损坏的包。

如图 11 中所示，假设报头中的值 X 的插值函数是线性的，两个连续发送的包之间的偏差是 X_Stride，当解压器接收 SO(n+3)并且观察到当前接收的包假设 SO(n+3)与上次正确接收的包 SO(n)之间的序列号偏差是 2 而且在解压器中缓存的这两个包之间的包的个数也是 2，则解压器就能重新生成这两个损坏的包的值 X 为 X(n+1)=X(n)+X_Stride 和 X(n+2)=X(n)+2*X_Stride。值 X 的例子可以是 RTP 报头中的时间戳

和序列号。这个过程可应用于诸如流式(streaming)等容忍延迟的情形。

另一个增强是压缩器序列号补偿。在要压缩的报头的 RTP 序列号不按 1 递增（递增量大于 1 或递减），但压缩器确定该报头仍然属于与前一个串相同的串时，压缩器进行序列号补偿。这在串中有些报头在至压缩器的途中丢失或乱序时发生。在这种情况下，将报头按 FO 报头压缩，但是只把 RTP 序列号差 SND 作为更新信息发送。SND = (压缩报头的实际 RTP SN)-(通过从 CD_SN 直接插值而获得的报头的 RTP SN)。SND 允许解压器确定正确的 RTP 序列号。例如，考察都属于同一个串的具有 RTP 序列号 = 5、6、7、8 和 9 的报头的序列。在去往发送器的压缩器的途中，报头 7 和 8 丢失。结果，压缩器在报头 9 被接收时看到一个大于 1 的增量。然而，根据对未压缩报头的检查，发送器的解压器确定报头 9 与报头 6 属于同一个串。假设报头 6 是以 CD_SN=3 被压缩的。现在报头 9 被用 CD_SN=3 压缩，因为 CD_SN 总是按 1 递增的。发送器的压缩器也发送一个 SND = 9 - 7 = 2。接收器的解压器把 SND 加到 CD_SN，然后应用用于 SO 的标准解压算法。

在只有乱序的情况中（在压缩器之前没有包丢失），有一序列的带 SND 的 SO 报头，但 SND 终归将变成零。一旦 SND 是零，压缩报头中就不需要 SND，压缩报头只是一个正常的 SO。如果包在到达发送器的压缩器之前丢失，SND 将不会到达零。需要将 SND 在每个报头中携带，直到压缩器从解压器接收到一个确认。否则，如果含有 SND 的报头丢失，解压器就不能正确地进行解压。

图 12 表示的本发明的实施例中，当要被压缩的报头的 RTP 序列号因包丢失而不按 1 递增，并且压缩器确定该报头与前一个报头属于同一个串时，发生压缩器序列号补偿。假设在本例中，具有 RTP 序列号 N + 1、N + 2、N + 3 的包都丢失。压缩器只收到具有 RTP 序列号 N 和 N + 4 的包，而 N 和 N + 4 属于同一个串。当压缩器发送具有 RTP 序列号 N + 4 的包的压缩报头时，除了短序列号 n+2 外，压缩器还需要发送一个 RTP 序列号差 SND，本例中该 SND 等于(N+5)-(N-2)。当解压器接收到这种包时，它把该 SND 加到 CD_SN 上，以导出正确的序列号，然后应用标准的 SO 解压算法。

图 13 表示本发明的用于限制传输带宽的执行错误检测和错误检测

码的重新生成的实施例。在以下说明中假设错误检测机制，诸如 UDP 校验和(2字节)不在压缩器与解压器之间的通信通道上以(CD_SN)的形式传送。如果 UDP 校验和不被端点应用(end application)使用，这并非什么问题。如果端点应用使用 UDP 校验和，并且有必要端点至端点地发送 UDP 校验和，则可以以直接了当的方式通过加每个压缩报头中的未压缩 UDP 校验和来扩展本实施例。然而，即使 UDP 不是以 CD_SN 的形式发送，有些有关 UDP 校验和的信息也能被传送到压缩器。

一种选择是把端点至端点 UDP 校验和分解成两部分：将源与压缩器之间的片断称作上游片断，把压缩器与解压器之间的片断称作下游片断。使用校验和的错误检测过程可以只在上游片断中被传送。在发送一个 UDP 包之前，压缩器检查校验和是否与数据一致，然后压缩器压缩该包。如果不一致，该包将被丢弃，或者发送该包时，丢弃校验和，而增加一个错误标志，该错误标志通知解压器：所接收的包含有错误数据，以校验和为形式的错误检测位（或其它错误检测位）在传送之前被丢弃，因此节省传输带宽。解压器转而依赖 CD_CC 的错误检测功能。如果没有错误被报告给解压器，解压器在解压该包后重新计算校验和。

以上解决方案仅当在 CD_CC 中有错误检测机制并且与 UDP 校验和相比功能相同或更大时才有作用。

在压缩一个包之前，压缩器检查校验和是否与数据一致。如果是，如上所述，压缩器压缩该包时不把校验和加入压缩报头中，并把压缩报头传送到解压器。如果不一致，压缩器可以丢弃该包，或者传送带校验和的包，或者传送不带校验和但带有或不带有错误标记的包。

图 14A－F 表示可以用于本发明的实践的 SO、ACK、FO、FH、FO EXT 和 FH REQ 的格式的例子。采用以下缩写：PT 是包类型，C_RTP_SN 是压缩的 RTP 序列号，C_RTP_TS 是压缩的 RTP 时间戳，C_IP_ID 是压缩的 IP_ID。然而应当明白，本发明并不仅限于此。可以将 SO 包的 PT 字段编码为 0，将 ACK 包编码为 10，将 FO 包编码为 110，将 FH 包编码为 1100，将 FO_EXT 包编码为 11110，将 FO_REQ 包编码为 111110。在 FO 和 FO_EXT 包中，M 是在 RTP 报头中的一位标记。在 FO 包中，T 是一个一位标志，如果 C_RTP_TS 出现，其值为 1，否则为 0；I 是一个一位标志，如果 C_IP_ID 出现，该标志被设置为 1，否则为 0。在 FH

包中，如果包长度被解压器上的较低层提供，则 IP 和 UDP 报头能被压缩。FO_EXT 包仅当若干个非基本字段有变化时才被传送；用位屏蔽来指示哪些字段出现，C_RTP_TS 和 C_IP_ID 的总是出现使 T 和 I 位标志没有必要。最后，仅在例外情况下，例如系统崩溃时，才发送 FH_REQ 包。

如果多个 RTP 流被压缩，而较低层不提供流之间的差异，则可能需要将一个上下文标识符(CID)加到上述报头的每一个。CID 可能只对一个方向来说是需要的，诸如在蜂窝系统中当移动站（MS）在每个方向上只有一个流时；并且 CID 对下传链路通信（包括 ACK）来说是不需要的。对于上传链路通信（包括 ACK）来说必须包括数值 CID，因为在网络端的解压总是处理多个流。

以下是一例可以用来为压缩器写代码的伪码。

该例表示的是从更新阶段向插值阶段的转换需要两个 ACK 的情形。为简化起见，在伪码中没有表示如图 8 中所示的 FH 和 FO 包的交替以及序列号压缩。

本例中，假设 S_DFOD 和 R_DFOD 是非静态的。它们因此是由压缩器和解压器动态地按以下方式确定的:

·当压缩器接收过 ACK(n)和 ACK(n-p)且(n-p)>=N_Last_Interr 时，将数值 S_DFOD 计算为 CFO(m)。注意 p 未必等于 1。

·压缩器在接收到一个非 SO 包之后的第一个 SO 包时计算 R_DFOD。数值 R_DFOD 是用线性插值法根据在 OAW 135 中存储的最后两个确认报头计算的。

压缩器的行为可以被模型化为一个由下表规定的状态机。

为解决接收器回绕和长错误突发问题，压缩器期待每隔 seq_cycle 个报头至少接收一次 ACK，并维持一个扩展标志。如果该标志为真，压缩器将在扩展方式中操作，即发送(CD_SN)1_extended。否则，它就(CD_SN)k。注意 N_elapsed 一直递增，除非发送器接收到 ACK（详细内容参看伪码）。在扩展方式中，如果 ext_cycle 消逝而没有确认，发送器就转换到 FH 状态。

当至少两个有 CD_SN>=N_last_Interr 的包已经被确认时，压缩器进入 SO 状态。然后，压缩器将 S_DFOD 设置等于最近 CFO。

一开始，压缩器在 FH 状态开始会话。HSW 117 是空的。数值

N_elapsed 被设置为零。Extended_flag 被设置为假。

在续接的情况中需要执行额外的过程。为简化起见，这里不包括这些过程。

FH 状态

事件	行动
接收对 FH(n) 的确认 ACK(n)	· 见 Compressor processing ACK(n) 伪码 · 状态<-FO_STATE;

在 FH 状态中，发送 header(n) 的过程是

```
{
        calculate CFO(n) and update N_Last_Interr;
        send as FH(n);
        store header(n) in HSW, color B red;                    /* n in FH is coded in
k_extended bits */
}
```

FO 状态

事件	行动
接收对 FO(n,m) 的确认 ACK(n) 接收 FH_Req	· 见 Compressor processing ACK(n) 伪码 · 状态<-FH_STATE;

在 FO 状态中，发送 header(n) 的过程是

```
{
        calculate CFO(n) and update N_Last_Interr;

        if N_elapsed >= seq_cycle
                extended_flag B TRUE;
        else
                extended_flag B FALSE;
        if N_elapsed >= ext_cycle
        {
                send FH(n), store header(n) in SHW, color B red;
                state ß FH_STATE;
                N_elapsed B 0;
        }
        if received more than two ACKs, AND the most recent two CD_SNs ACKed >=
N_Last_Interr
        {
                S_DFOD B CFO(n);
                send SO(n), store header(n) in HSW, color B current_color();    /* see
function below */
                state B SO_STATE;
        }
        else
                send FO(n, S_RFH); store header(n) in HSW, color B current_color();

        N_elapsed B N_elapsed + 1;
}

current_color() {
        if extended_flag = TRUE
                return red;
        else
                return green;
}
```

SO 状态

事件	行动
接收 ACK(n)	· 见 Compressor processing ACK(n)伪码
接收 FH_Req	· 状态<-FH_STATE;

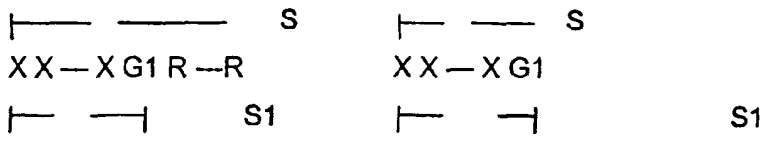
在 SO 状态中，发送 header(n)的过程是

```
        {
                calculate CFO(n) and update N_Last_Interr;

        if N_elapsed >= seq_cycle
                extended_flag  B TRUE;
        else
                extended_flag B FALSE;
        if N_elapsed >= ext_cycle
        {
                send FH(n), store header(n) in SHW, color = red;
                state B FH_STATE;
                N_elapsed B 0;
        }
        if CFO(n) = S_DFOD
                send SO(n); store header(n) in HSW, color B current_color();
        else
        {
                send FO(n, S_RFH); store header(n) in HSW, color B current_color();
                state B  FO_STATE;
        }
        N_elapsed BN_elapsed + 1;
}

Compressor processing ACK(n)
{
        if color of ACK(n) is green                          /* n is coded in k bits */
                h_ack ß a green header in HSW 117 whose (CD_SN)k = n;  /* see below for
detailsr */
        else                                                 /* n is coded in k_extended
bits */
                h_ack ß a red header in HSW 117 whose (CD_SN)k_extended = n;

        S_RFH ß CD_SN of h_ack;
        Delete all headers in HSW precding (older than) h_ack;
        Move h_ack to Header(S_RFH);
        N_elapsed β Diff(current|CD_SN, S_RFH):    |
}
```

可以证明，在以上过程中，HSW 177 中有一个且仅一个报头可被正确地标识为是被确认的报头，换言之，没有 ACK 的歧义。如果 ACK(n)是红的，n 被用 l_extended 位编码，只有一个红色的报头能与该 ACK 匹配，因为 HSW 117 中最多有 $2^{l\_extended}$ 个报头。否则，如果 ACK(n)是绿的，我们将显示它仍然能被唯一地映射到 HSW 117 中的一个绿色报头。

假设每次压缩器发送一个包后 HSW 117 就被拍一个快照，并用一串字母 R（代表红色）和 G（代表绿色）来代表它。设 S 为对应于某任意快照的串。注意 S 以由发送器发送的最老的包开始，以最年轻的包结束。此外，在压缩器发送两个连续的包之间，串 S 不改变，除非在此期间有一个 ACK 到达，这将有些来自 S 的开始的字母。

现在设 G1 代表 S 中最右边的（最年轻的）G，S1 是 S 的前缀，一直到（含）G1。那么只有两种如下所示的情形：



Case 1 Case 2

设 len(S1)表示 S1 的长度。在情形 1 中，由于在 G1 后有 R，len(S1)必定等于 seq_cycle(=2^k)。否则，压缩器就不会把 G1 后的包作为红色的发送。在情形 2 中，len(S1)≤seq_cycle 必定为真。否则，压缩器就不会把 G1 作为红色的发送。因此，在哪一种情形中，len(S1)都小于或等于 seq_cycle。

由于 G1 是 S 中最右边的绿色字母，这证明任何时候 HSW 177 中可能存在最多 2^k 个绿色报头。所以，当一个绿色 ACK 被压缩器接收时，该 ACK 中的 k 位的 CD_SN 可被用来唯一地标识 HSW 中的绿色报头。

注意到解压器必须与压缩器合作才能保证在这两种方式之间的转换期间保持 CD_SN 的同步。第一，如果解压器接收到红色包并且决定确认该包，它就必须发送一个携带(CD_SN)l_extended 的红色包。

第二，如果解压器接收到一个 FO 包 FO(n,m)，正确的基准包必定是 OAW 135 中的最年轻的（最近的）报头，其最低有效的 k（如果 m

是 k 位的）或 1_extended（如果 m 是 1_extended 位的）位与 m 匹配。注意这依赖于在每个方向上通道的行为像 FIFO 的假设。

图 19 表示第二个条件。在这个例子中，NT0 和 NT2 分别是时间 T0 和 T2 时 CD_SN 的值。假设在 T1，压缩器发送包 ACK(NT0)，其中 NT0 是以 1_extended 位编码的。在 T2，压缩器收到 ACK(NT0)。它然后计算 N_elapsed 等于(NT2-NT0)并发现 N_elapsed<seq_cycle。与此同时，RTP 包到达压缩器，压缩器决定用 header(NT0)作为基准，将该 RTP 包按 F0 包发送。N_elapsed<seq_cycle(=2^k)，所以该 F0 包中的 NT2 和 NT0 以 k 位编码。在 T3，F0 到达解压器。为了提取正确的基准报头，解压器只要从尾（最近的）到头（最老的）地搜索其 OAW，找到其 CD_SN 的最低有效 k 位与(NT0)k 匹配的第一个报头。

注意到在 T3，解压器的 OAW 135 可能含有数量多于 2^k 的报头。然而，上述操作总是给出正确的基准报头。因为正向通道的 FIFO 特性，在 T1 与 T3 之间被解压器收到（并因此确认）的无论什么，必定都是由压缩器在 T0 与 T2 之间发送的。换言之，如果 A 表示 OAW 135 中的在 header(NT0)之后被加入的报头的集合，B 表示 T2 时 HSW 117 中的报头的集合，则 A∈B 总是成立。由于|A|<2^k，于是有|B|<2^k。因此，集合 B 中没有两个报头使得它们的 CD_SN 的最低有效 k 位与包 F0(NT2,NT0)中的(NT2)k 匹配。

以下是用于解压器的伪码的例子：

解压器大多是由从压缩器所接收的报头（即 FH、F0 或 S0）驱动的。

下文中，“正确接收的”的意思是在接收的报头（无论 FH、F0 还是 S0）中没有检测到错误。除了上述的状态信息外，解压器也维持最后一个重构的报头即 header(R_Last_Decomp)的一个拷贝。当接收到 F0 包时，解压器将用以上关于压缩器的伪码所述的过程来提取正确的基准报头。

如果 FH(n)被正确地接收

```
{
    reconstruct header(n) from FH(n);
    send ACK(n);
    R_Last_Acked-n;
    store header(n) in the OAW 135 and also header(R_Last_Decomp);
}
```

如果 FO (n, m) 被正确地接收

```
{
    if header(m) cannot be found in the OAW 135    /* could only happen due to system failures
    */
            Send FH_Req;
    else
    {
        retrieve header (m) fro the OAW 135 or header (R_RFH);
        delete headers in OAW that are older than header (R_RFH);
        reconstruct header(n)-FO_DIFF(n, m) + header(m);
        if R_RFH!=m
            R_RFH-m and store header(m) as header(R_RFH);
        if FO(n, m) is one of the first two FO packets received or
            N_RT FO packets have been received since last ACKed FO packet
        {
```

```
                Send ACK(n);
                R_Last_Acked=n; store header (n) in the OAW:
            }
            store reconstructed header (n) in header(R_Last_Decomp);
}
```

如果 SO (n) 被正确地接收

```
{
    if it's the first SO packet after a non-SO packet
    {
        find the two most recently reconstructed headers in OAW 135;
        if not found                    /* could only happen due to system
failure */
            Send FH_Req;
        else                    /*let the two headers be header (p) and
 header (q), p<q*/
            R_DFOD+FO_DIFF(q,p)/Diff(q,p);
    }
    reconstruct header(n)=R_DFOD * Diff(n, R_Last_Decomp) +
header(R_Last_Decomp);
    store header(n) in header(R_Last_Decomp);

    if 1) (seq_cycle - N_RT) packets have elapsed since R_Last_ACKed, or,
```

/* time to send a periodic ack */

2) extended_flag in SO is ON and this is the first such packet, or,

/* the compressor switches to extended mode; send ack so the compressor returns to normal mode */

3) received more than N_RT packets with extended_flag ON since R_Last_ACK

/* the previous ack was apparently not received; send another ack */

{

Send ACK(n); n is coded in extended mode if conditions 2 or 3 are met

R_Last_Acked-n:

store Header (n) in the OAW;

}

}

store header(n) in the OAW 135 and also header(R_Last_Decomp);

}

如果 FO (n, m) 被正确地接收

{

if header(m) cannot be found in the OAW 135 /* could only happen due to system failures */

Send FH_Req;

else

{

5

```
                retrieve header (m) fro the OAW 135 or header (R_RFH);
                delete headers in OAW that are older than header (R_RFH);
                reconstruct header(n)-FO_DIFF(n, m) + header(m);
                if R_RFH!=m
                        R_RFH-m and store header(m) as header(R_RFH);
                if FO(n, m) is one of the first two FO packets received or
                        N_RT FO packets have been received since last ACKed FO packet
                {
                        Send ACK(n);
                        R_Last_Acked-n; store header (n) in the OAW:
                }
                store reconstructed header (n) in header(R_Last_Decomp);
}
```

如果 SO (n) 被正确地接收

```
{
        if it's the first SO packet after a non-SO packet
        {
                find the two most recently reconstructed headers in OAW 135;
                if not found                    /* could only happen due to system
failure */
                        Send FH_Req;
                else                    /*let the two headers be header (p) and
5
```

```
header (q), p<q*/
                    R_DFOD+FO_DIFF(q,p)/Diff(q,p);
        }
        reconstruct header(n)-R_DFOD * Diff(n, R_Last_Decomp) +
header(R_Last_Decomp);
        store header(n) in header(R_Last_Decomp);

        if 1) (seq_cycle - N_RT) packets have elapsed since R_Last_ACKed, or,
        /* time to send a periodic ack */
          2) extended_flag in SO is ON and this is the first such packet, or,
            /* the compressor switches to extended mode; send ack so the compressor returns
to normal mode */
          3) received more than N_RT packets with extended_flag ON since R_Last_ACK
            /* the previous ack was apparently not received; send another ack */
        {
           Send ACK(n); n is coded in extended mode if conditions 2 or 3 are met
           R_Last_Acked-n:
          store Header (n) in the OAW;
        }
}
```

HSW 117 and OAW 135

在最坏的情况中，往返延迟实际上等于 RTT_UB，OAW 135 或 HSW 117 可能需要保持 $2^{l\_extended}$ 个报头。然而，这是非常不可能发生的。在多数情况下，需要在 OAW 135 或 HSW 117 中保持的项数少于 2^k。在实践中这对 OAW 和 HSW 二者来说都意味着很小的项数。例如，假设每个包间隔 20 微秒，则 16(k=4)项将提供 320 微秒的往返延迟时间。

静态字段不需要在 OAW 135 或 HSW 117 中存储多项。只需要一份静态字段。

在 RFC 2508 中，每个压缩报头都携带序列号。在多数情况下，序列号足以用来通过线性插值重构完整报头。对于线性插值会导致报头重构不正确的那些包，压缩器发送相对紧邻的前导包的第一级差别信息。因此，一个包的损失将使后继的带压缩报头的包无效，因为丢失的包可能携带着 FO 差别信息。RFC 2508 只依赖 4 位的序列号来检测包损失。序列号每隔 16 个包就绕回。当发生一个比 16 个包更长的错误突发时，就有 16 分之一的概率检测不到错误，这个概率高得不能接受。此外，即使解压器能检测错误，要从错误恢复，解压器必须请求压缩器通过发送 CONTEXT_STATE（上下文状态）消息来发送一个大型报头。因此在所请求的报头到达接收器之前就有一个往返延迟。在实时谈话通信的情况中，这个延迟转变成谈话的中断。此外，发送大型报头就带宽而言是昂贵的。

本发明的一个实施例把 k 位的序列号(k 可被设置成 4)用于线性插值。同 RFC 2508 一样，当线性插值导致报头重构不正确时，压缩器发送 FO 差别信息。与 RFC 2508 不同的是，该差别是相对于一个已知被正确接受的基准包而计算的。该包不必是紧邻当前包的前导包。这个特点保证当前包能被可靠地重构，即使有一个或多个包丢失。由于能以那样的方式重构该包，就不需要发送完整报头。第一级差别信息多数时间能用比完整报头的绝对值更少的位来编码。FO 差别报头有额外字段携带基准号，即基准包的序列号。为了保障即使出现长错误突发也将检测出错误，解压器发送 ACK，其频繁程度足以使得压缩器每隔 seq_cycle 个包就至少接收一次确认。在没有这种 ACK 时，压缩器将假设可能有长错误突发。在多数情况下，压缩器只要转换到 l_extended 位的序列号就够了，其中 l_extended 大得足以避免歧义。在任何情况下，包的丢失将不使后继的带压缩报头的包无效。因此，当解压器检

测到包丢失时，它不必非要请求重新传送。

下面的图 20 表示现有技术的 RFC 2508 与本发明的比较结果。假设在这个测试中是 60 微秒的一路(固定的)延迟。RTP 包之间的相互间隔是 30 微秒。采用有不同平均包错误率的随机错误模型。将压缩率定义为压缩报头的平均大小与原始 IP/UDP/RTP 报头的大小之间的比率。注意在本发明中，将 ACK 包的大小包括在平均压缩报头大小的计算中。包错误率一高于 0.4%，本发明的性能就超过 RFC 1508。

本发明的强健性方案要求压缩器和解压器分别维持 HSW 117 和 OAW 135 队列。假设往返时间小于 320 微秒，队列的大小是 16 项 + 一份静态字段。

	1p4	1pv6
静态字段的大小（字节数）	18	40
一项的大小（字节数）	22	20
一个双向会话的 HSW 或 OAW 的大小（字节数）	（16*22 + 18）*2 = 740	（16*20 + 20）*2 = 720

约 1 兆字节的存储器将允许同时处理 1400 个会话。管理队列的处理负荷非常适度，因为涉及指针操作。

尽管就最佳实施例对本发明作了说明，应当明白在不偏离本发明的精神和范围的情况下可以对本发明作出许多改变。这种改变不超出后附的权利要求的范围。

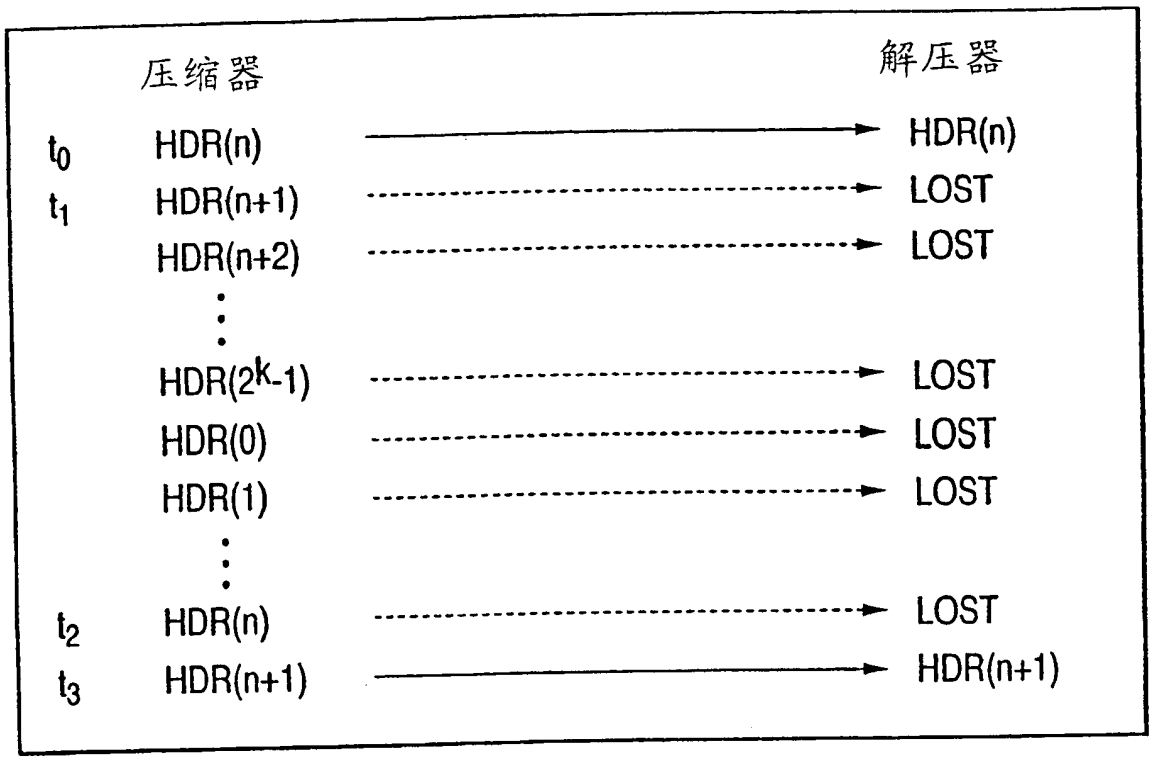

压缩器
解压器
t_0 HDR(n) HDR(n)
t_1 HDR(n+1) LOST
HDR(n+2) LOST
HDR(2^k-1) LOST
HDR(0) LOST
HDR(1) LOST
t_2 HDR(n) LOST
t_3 HDR(n+1) HDR(n+1)

图 1
(现有技术)

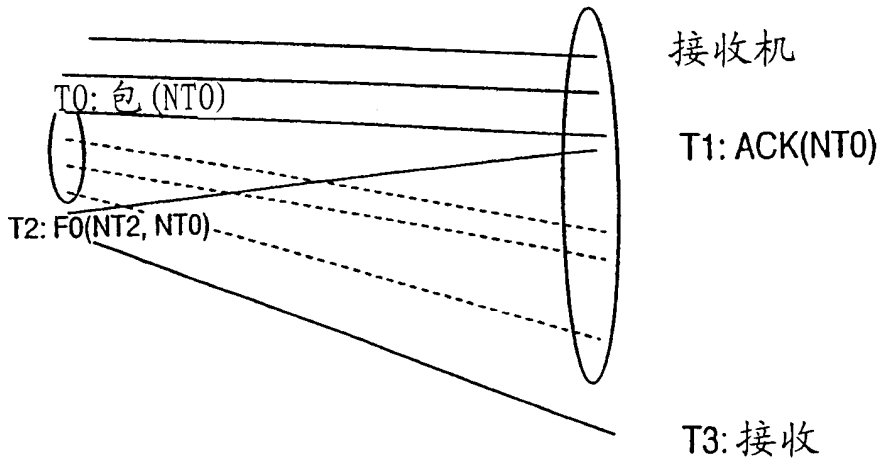

接收机
T0: 包 (NT0)
T1: ACK(NT0)
T2: F0(NT2, NT0)
T3: 接收

图 19

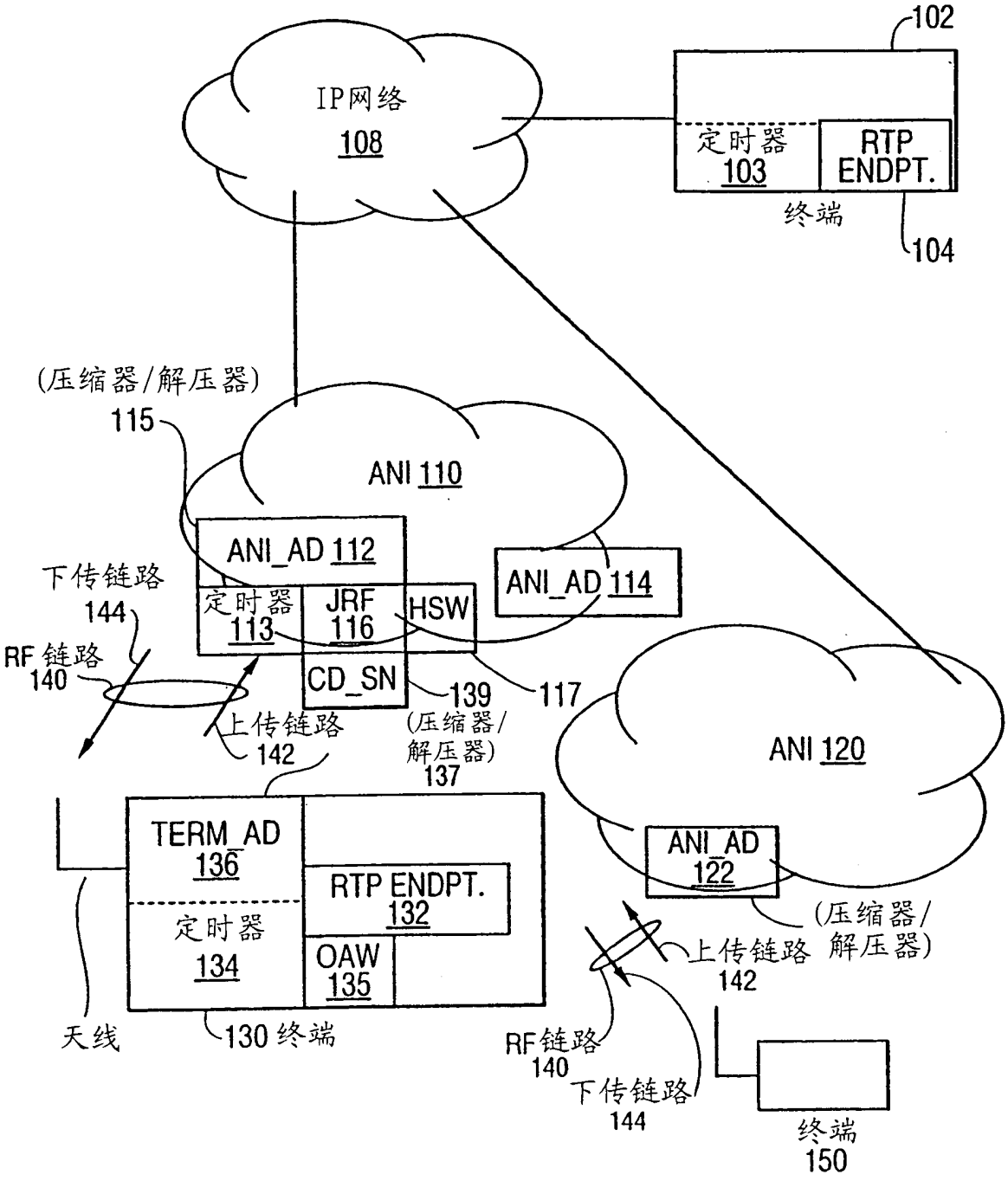

IP网络
108
102
定时器
103
RTP
ENDPT.
终端
104
(压缩器/解压器)
115
ANI 110
ANI_AD 112
ANI_AD 114
下传链路
144
定时器
113
JRF
116
HSW
RF链路
140
CD_SN
139
117
(压缩器/
解压器)
137
上传链路
142
ANI 120
TERM_AD
136
RTP ENDPT.
132
定时器
134
OAW
135
ANI_AD
122
(压缩器/
解压器)
上传链路
142
天线
130 终端
RF链路
140
下传链路
144
终端
150

图 2

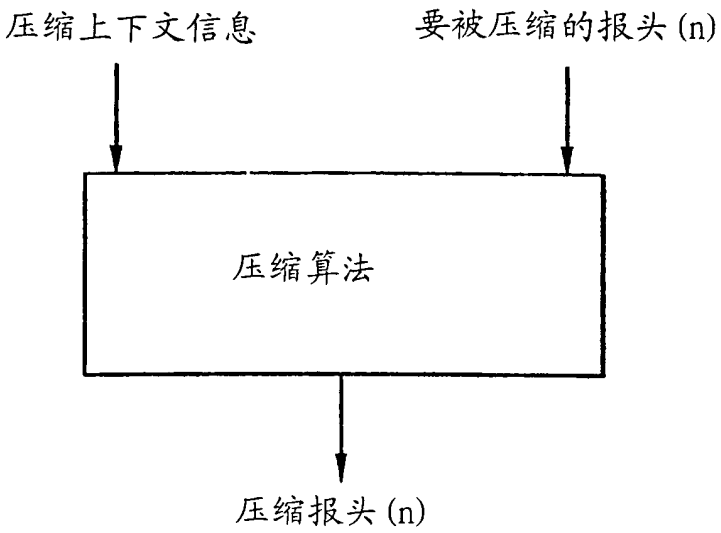


图 3

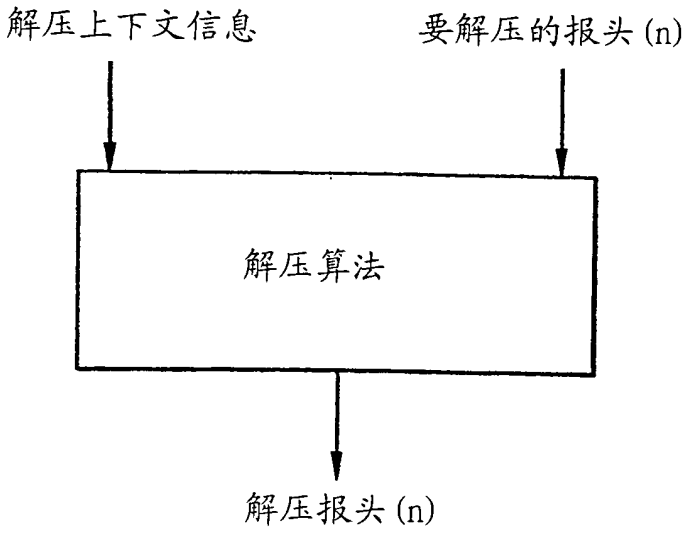


图 4

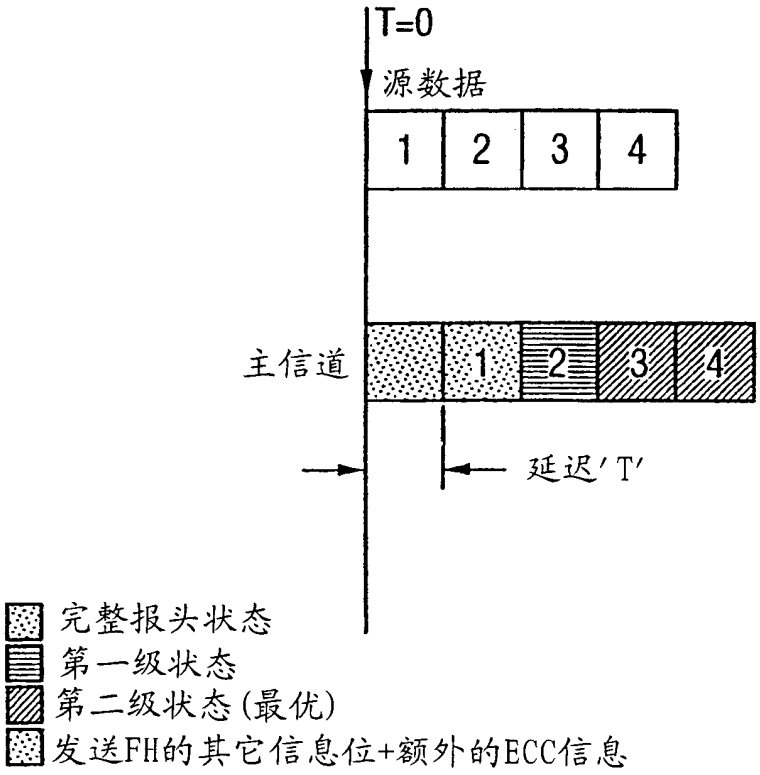

T=0
源数据
1
2
3
4
主信道
1
2
3
4
延迟'T'
完整报头状态
第一级状态
第二级状态(最优)
发送FH的其它信息位+额外的ECC信息

图 5A

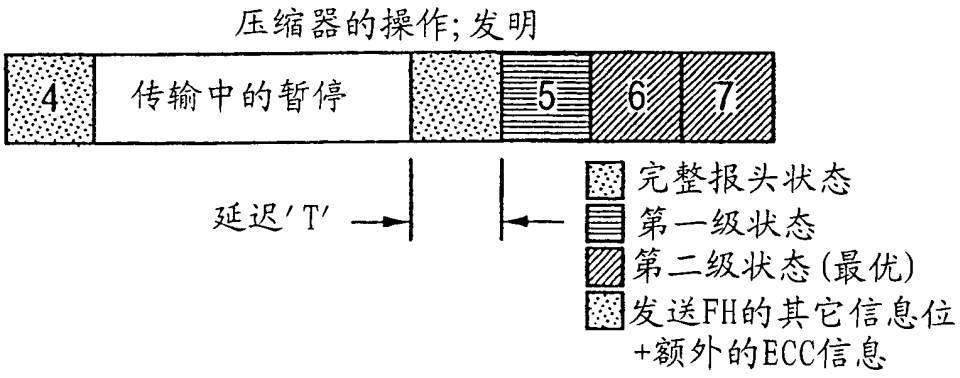

压缩器的操作;发明
4
传输中的暂停
5
6
7
延迟'T'
完整报头状态
第一级状态
第二级状态(最优)
发送FH的其它信息位
+额外的ECC信息

图 5B

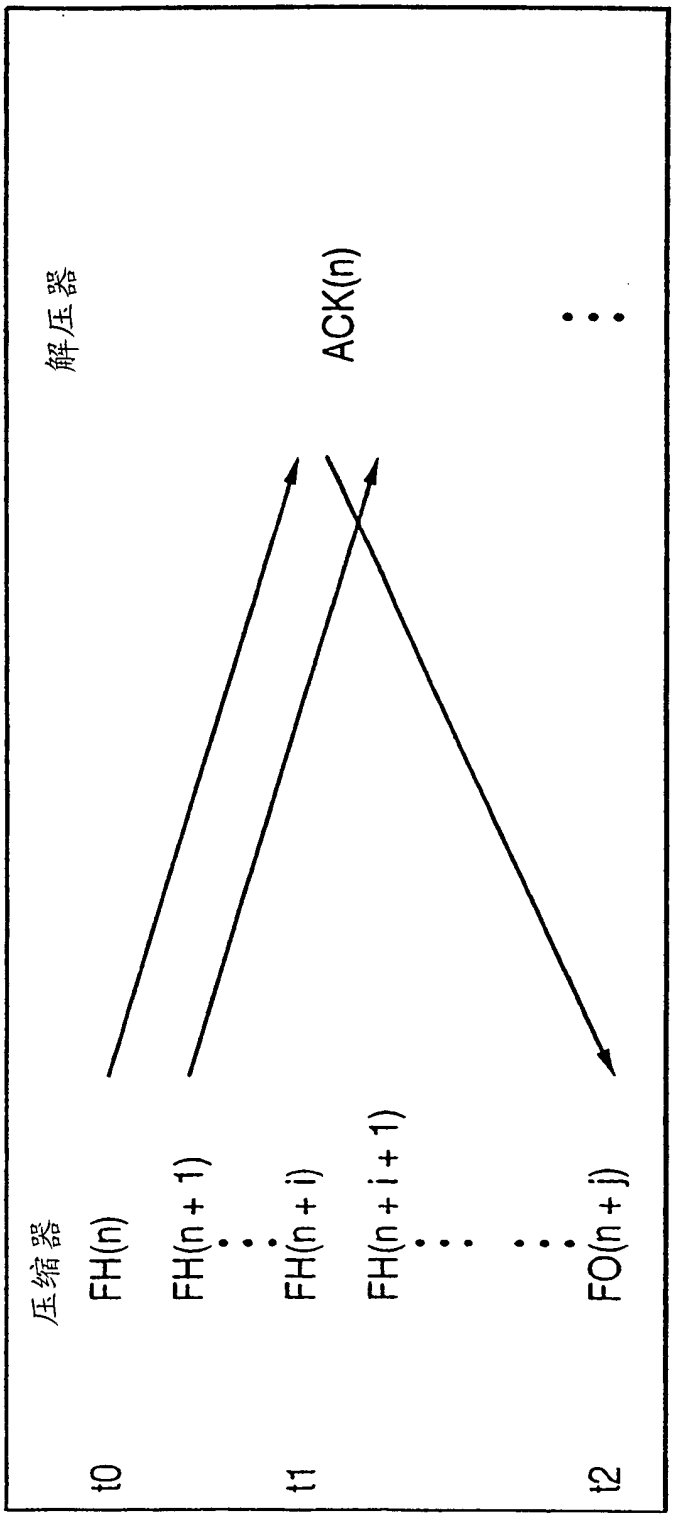


图 6

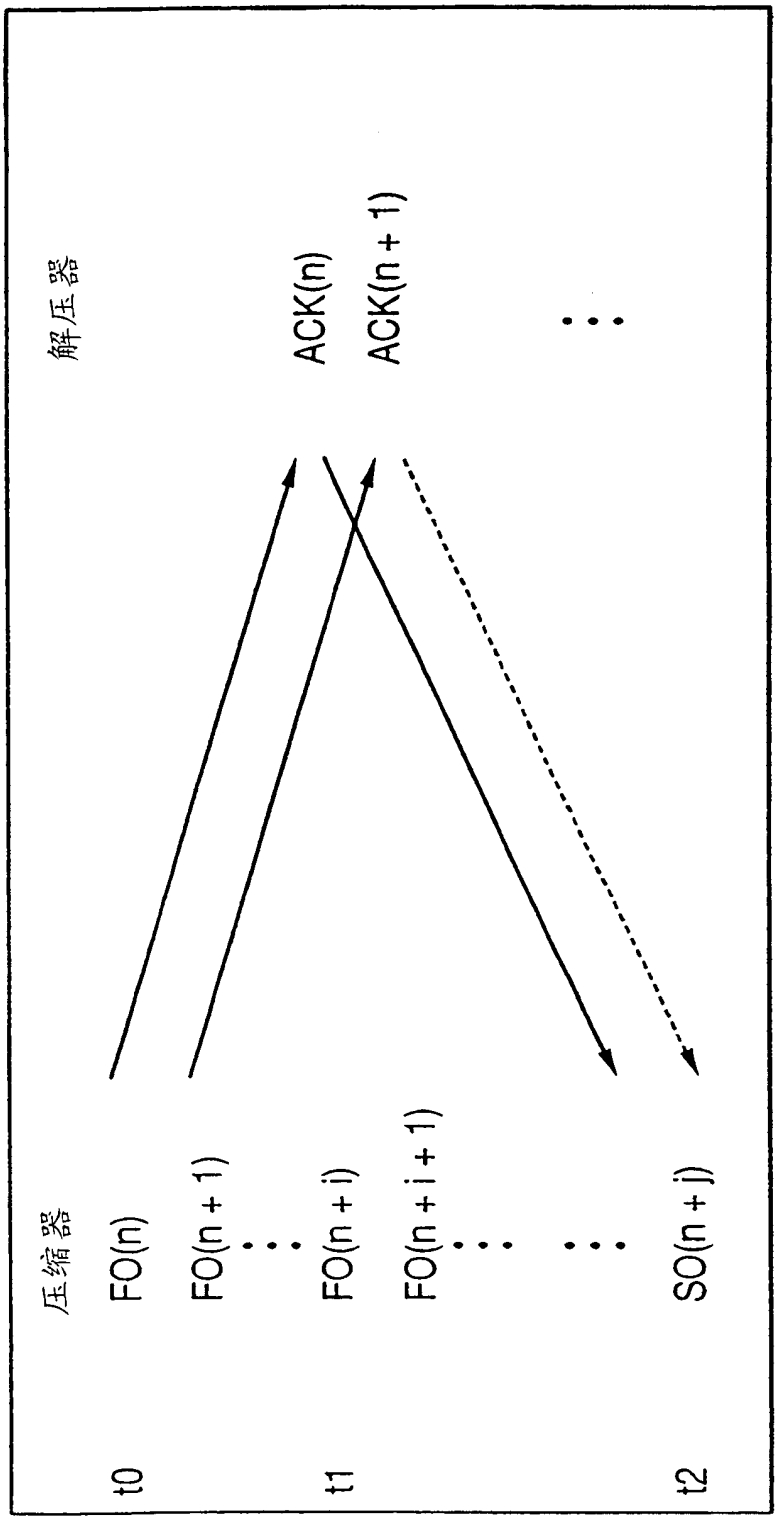


图 7

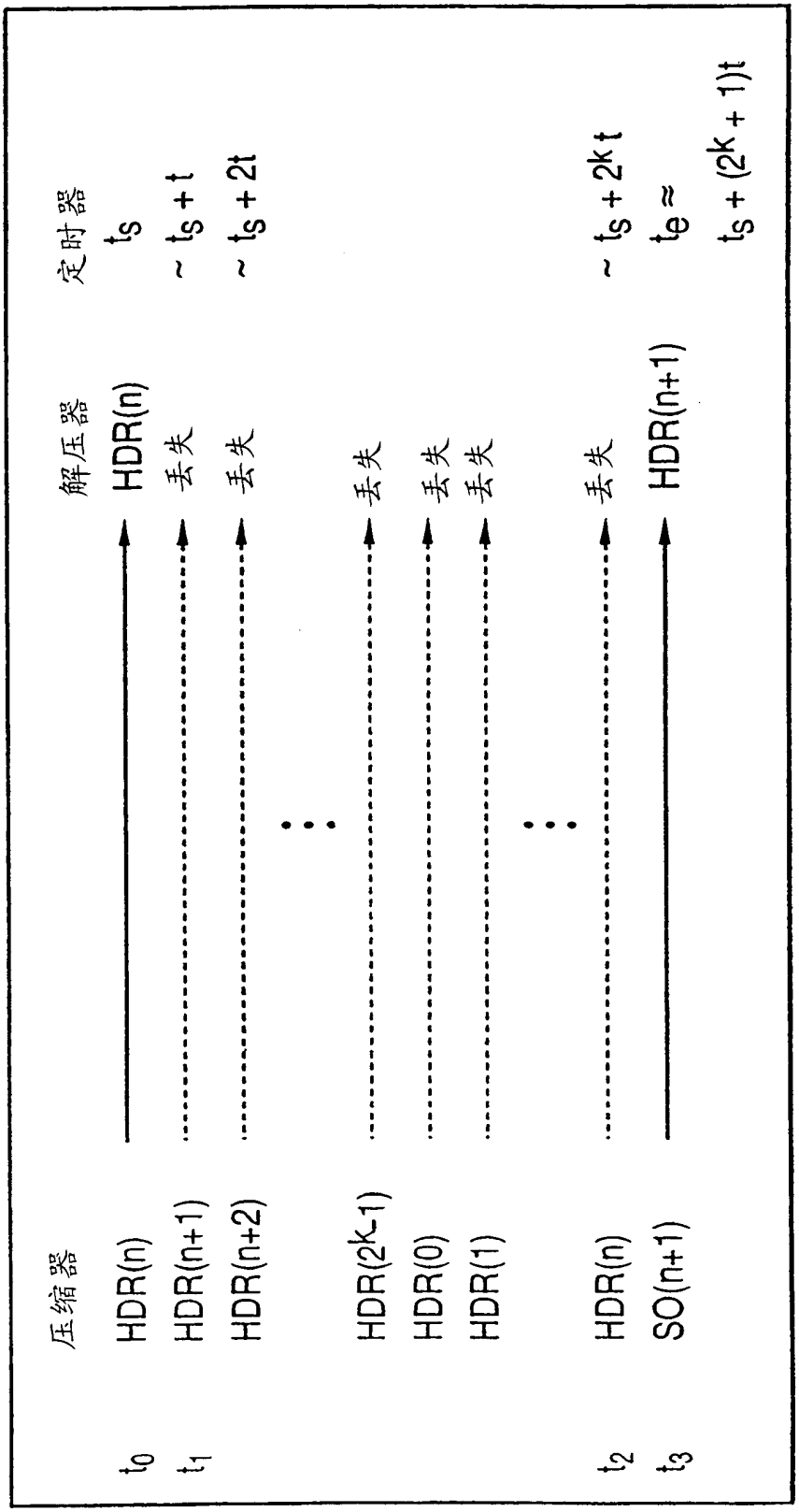

压缩器
解压器
定时器
t_0
HDR(n)
HDR(n)
t_s
t_1
HDR(n+1)
丢失
$\sim t_s + t$
HDR(n+2)
丢失
$\sim t_s + 2t$
HDR(2^k-1)
丢失
HDR(0)
丢失
HDR(1)
丢失
t_2
HDR(n)
丢失
$\sim t_s + 2^k t$
t_3
SO(n+1)
HDR(n+1)
$t_e \approx$
$t_s + (2^k + 1)t$

图 8

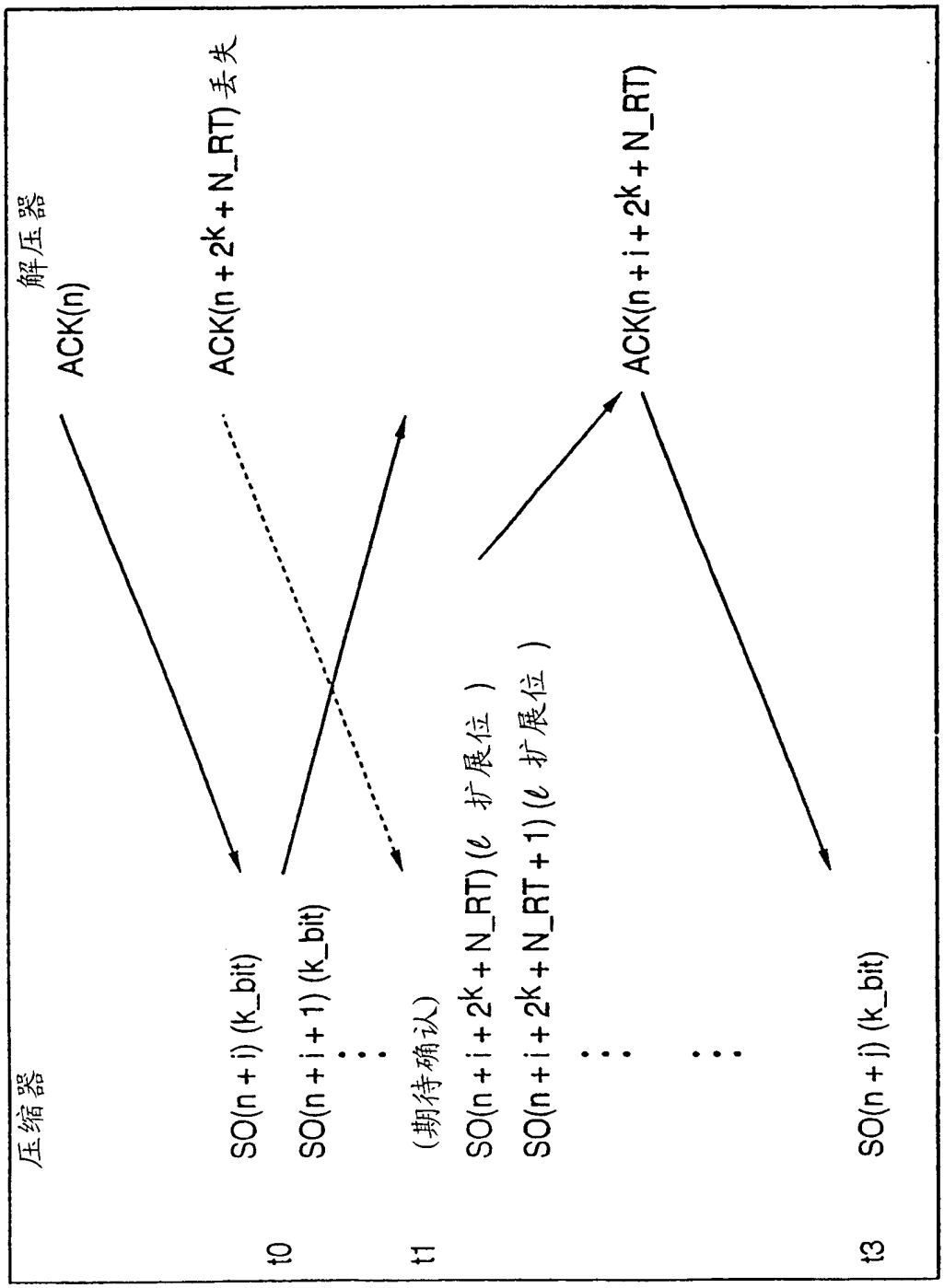

压缩器
解压器
ACK(n)
ACK(n + 2^k + N_RT) 丢失
ACK(n + i + 2^k + N_RT)
t0
SO(n + i) (k_bit)
SO(n + i + 1) (k_bit)
t1
(期待确认)
SO(n + i + 2^k + N_RT) (ℓ 扩展位)
SO(n + i + 2^k + N_RT + 1) (ℓ 扩展位)
t3
SO(n + j) (k_bit)

图 9

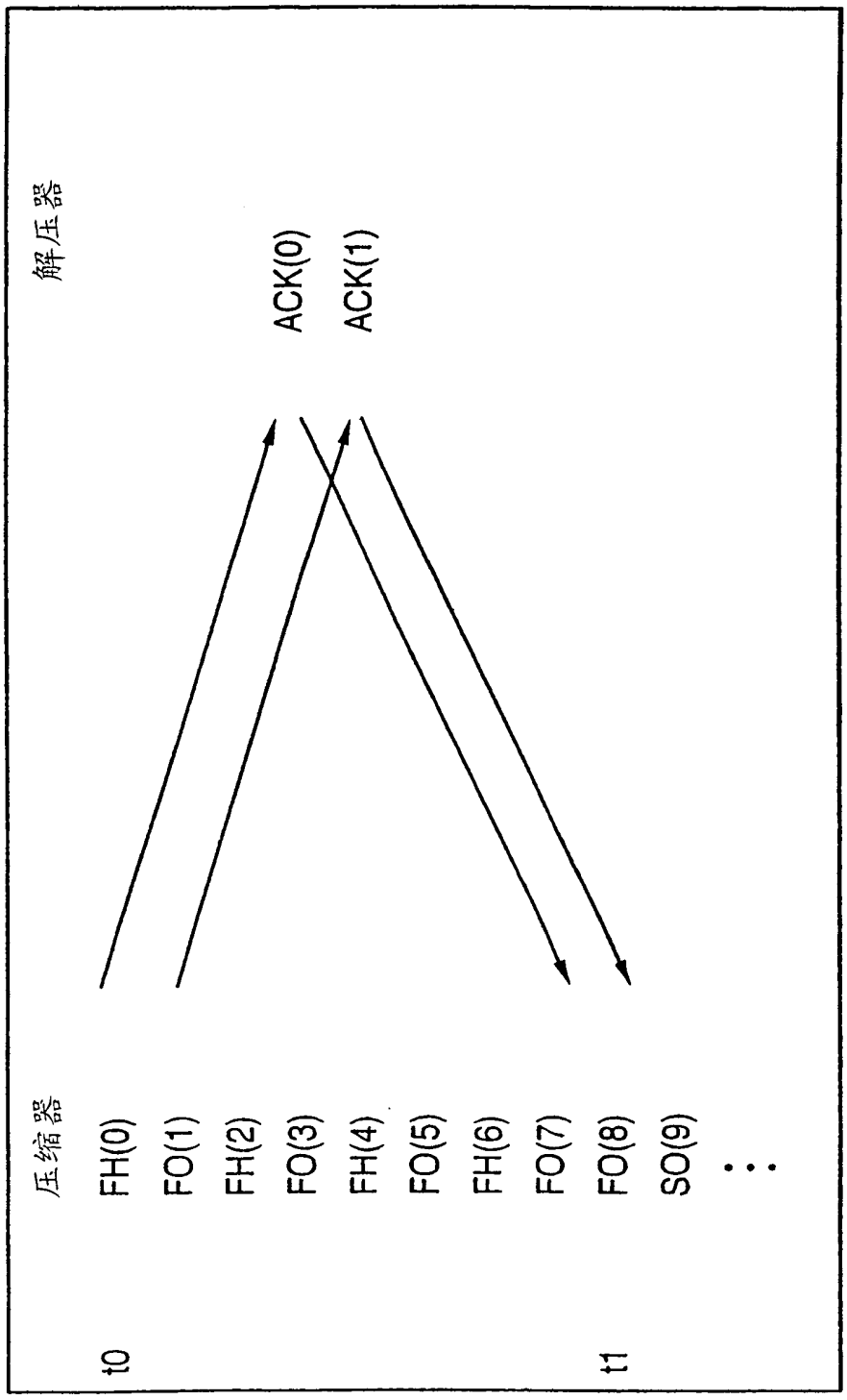


图 10

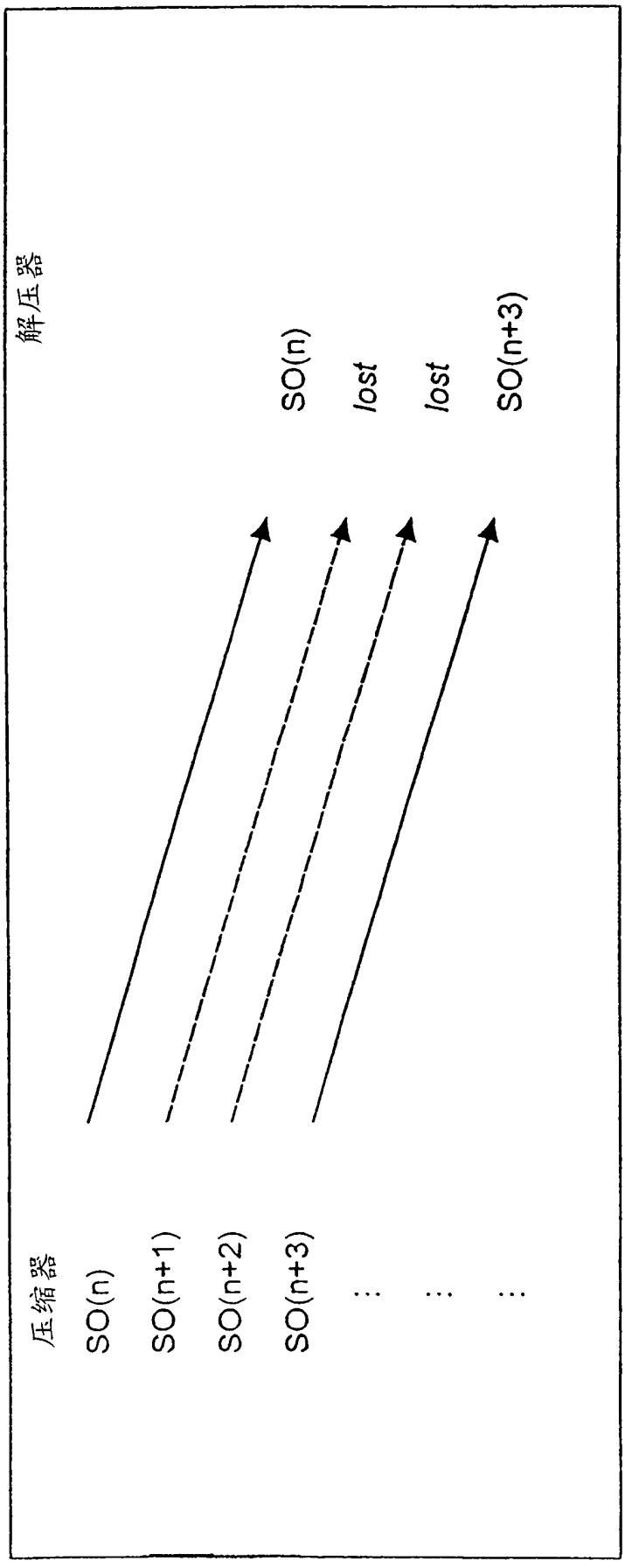


图 11

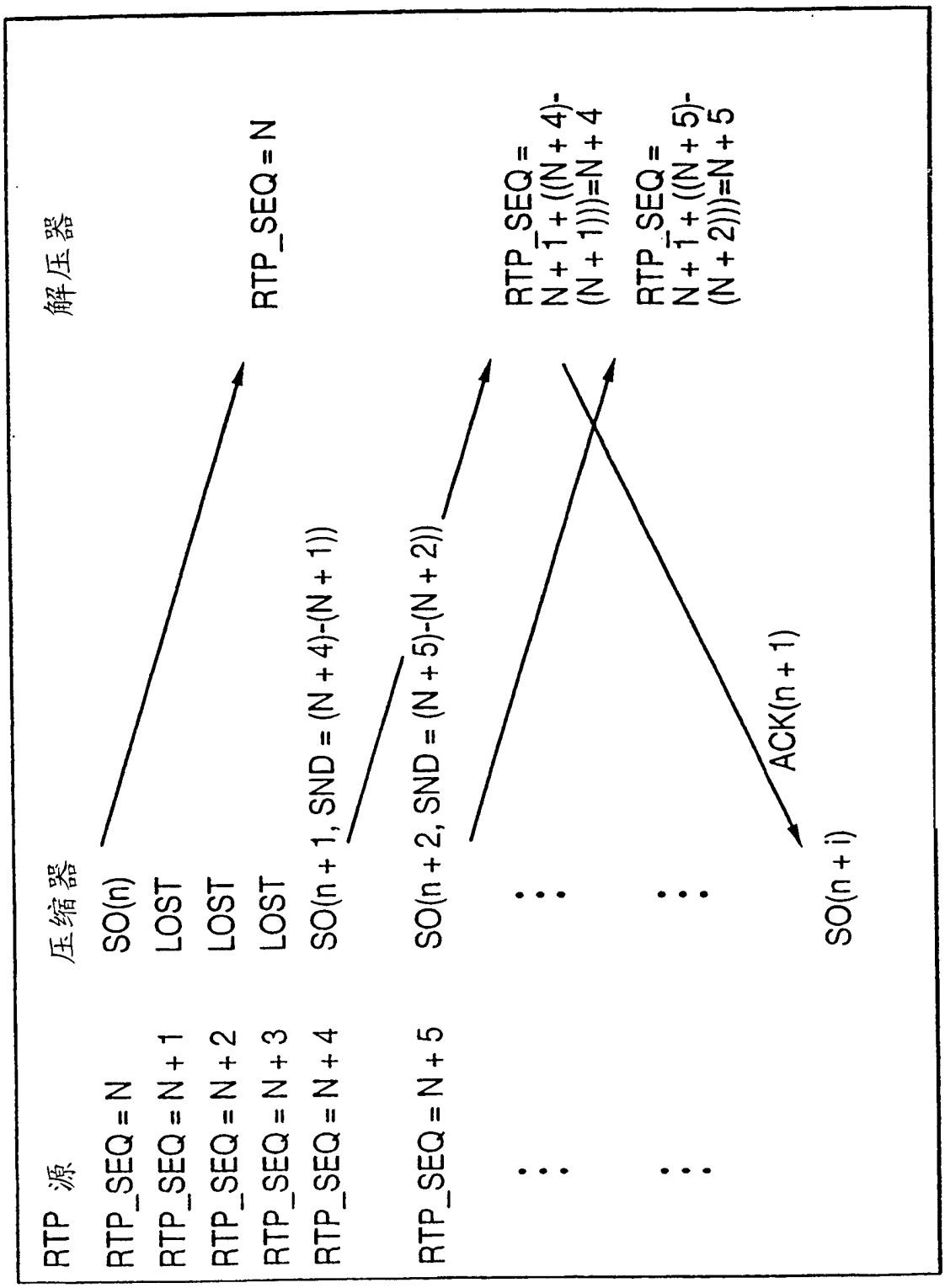

RTP 源
压缩器
解压器
RTP_SEQ = N
SO(n)
RTP_SEQ = N + 1
LOST
RTP_SEQ = N + 2
LOST
RTP_SEQ = N + 3
LOST
RTP_SEQ = N + 4
SO(n + 1, SND = (N + 4)-(N + 1))
RTP_SEQ = N + 5
SO(n + 2, SND = (N + 5)-(N + 2))
RTP_SEQ = N
RTP_SEQ = N + 1 + ((N + 4)-(N + 1)))=N + 4
RTP_SEQ = N + 1 + ((N + 5)-(N + 2)))=N + 5
ACK(n + 1)
SO(n + i)

图 12

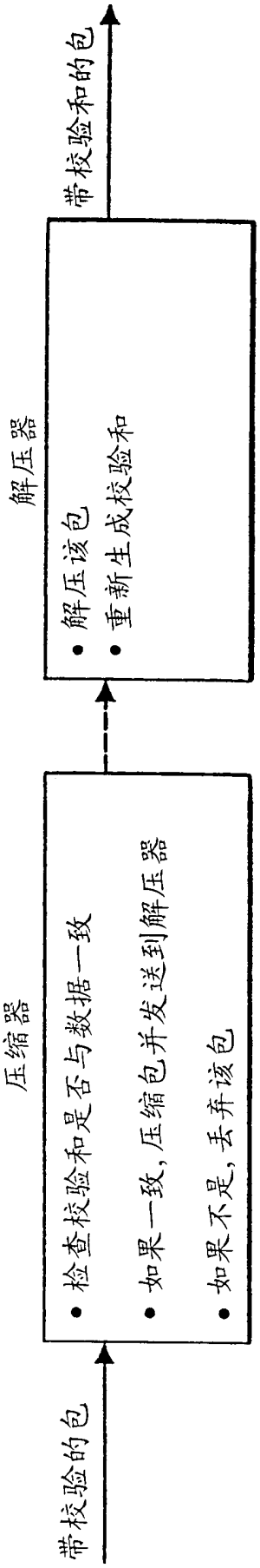

带校验的包
压缩器
• 检查校验和是否与数据一致
• 如果一致，压缩包并发送到解压器
• 如果不是，丢弃该包
解压器
• 解压该包
• 重新生成校验和
带校验和的包

图 13

1. SO包 (PT = 0)

图 14A

2. ACK包 (PT = 10)

图 14B

3. FO包 (PT = 110)

图 14C

M – RTP报头中的标记(1位)
T – 标志-如果C_RTP_TS出现,被设置为1,否则被设置为0(1位)
I – 标志-如果C_IP_ID出现,被设置为1,否则被设置为0(1位)

4. FH包 (PT = 1110)

图 14D

* 假设包长度是由位于解压器侧的较低层提供的,则可用报头压缩信息替换FH包中IP和UDP报头中的长度字段.

5. FO_EXT 包　　(PT = 11110)

PT	C_RTP_SN	M	C_RTP_TS	C_IP_ID	位屏蔽	字段值	有效负载

- 仅当一个或几个非根本字段已经改变时才传送FO_EXT
  位屏蔽被用来指示该包中出现哪些字段.
- C_RTP_TS和C_IP_ID将总出现在FO_EXT包中. 因此T和I位标志不必要.

图　14E

6. FH_REQ 包　　(PT = 111110)

PT

完整报头请求包仅当在例外情形下, 例如系统崩溃时, 才被发送.

图　14F

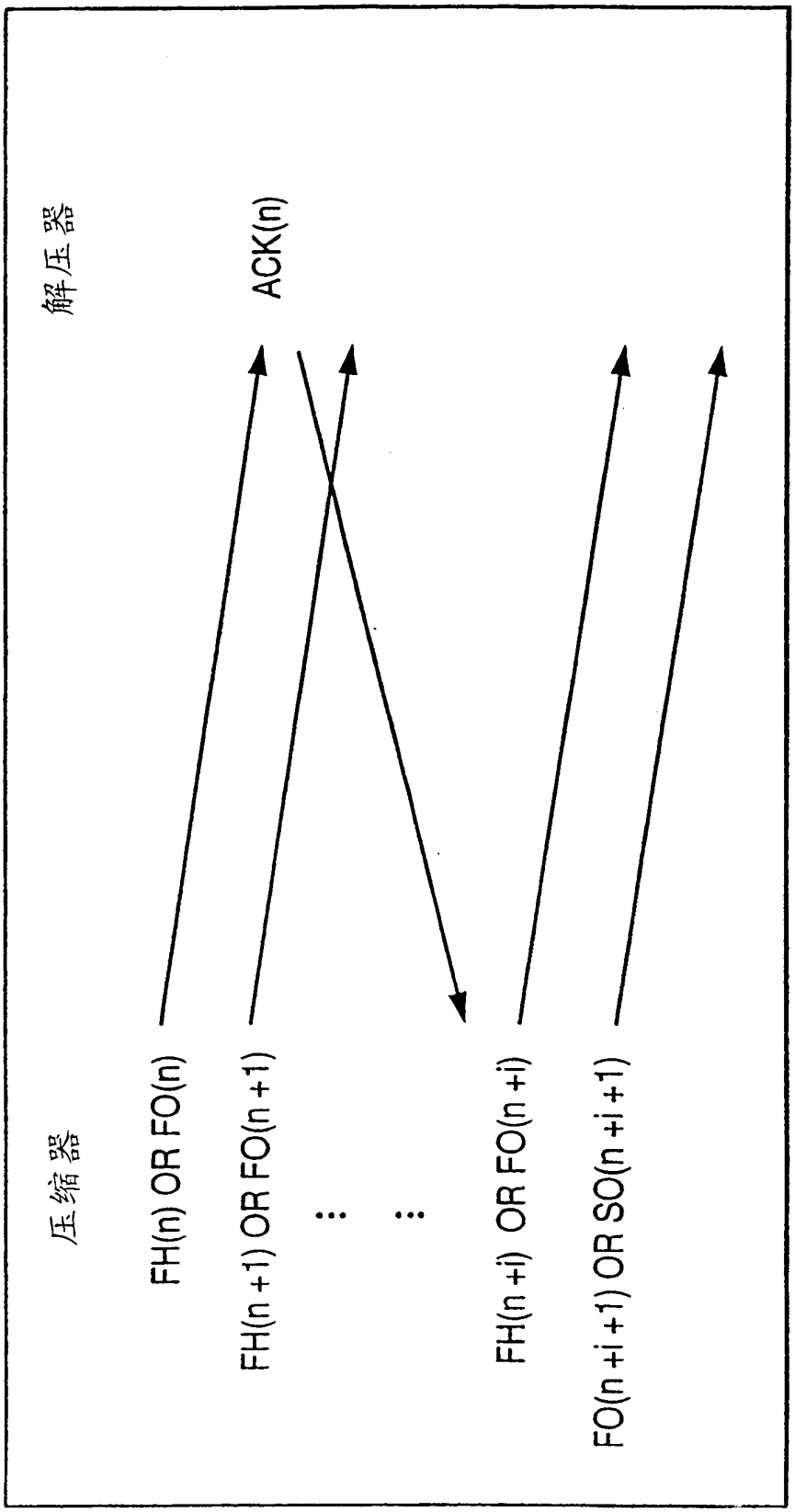

压缩器
解压器
FH(n) OR FO(n)
ACK(n)
FH(n +1) OR FO(n +1)
FH(n +i) OR FO(n +i)
FO(n +i +1) OR SO(n +i +1)

图 15

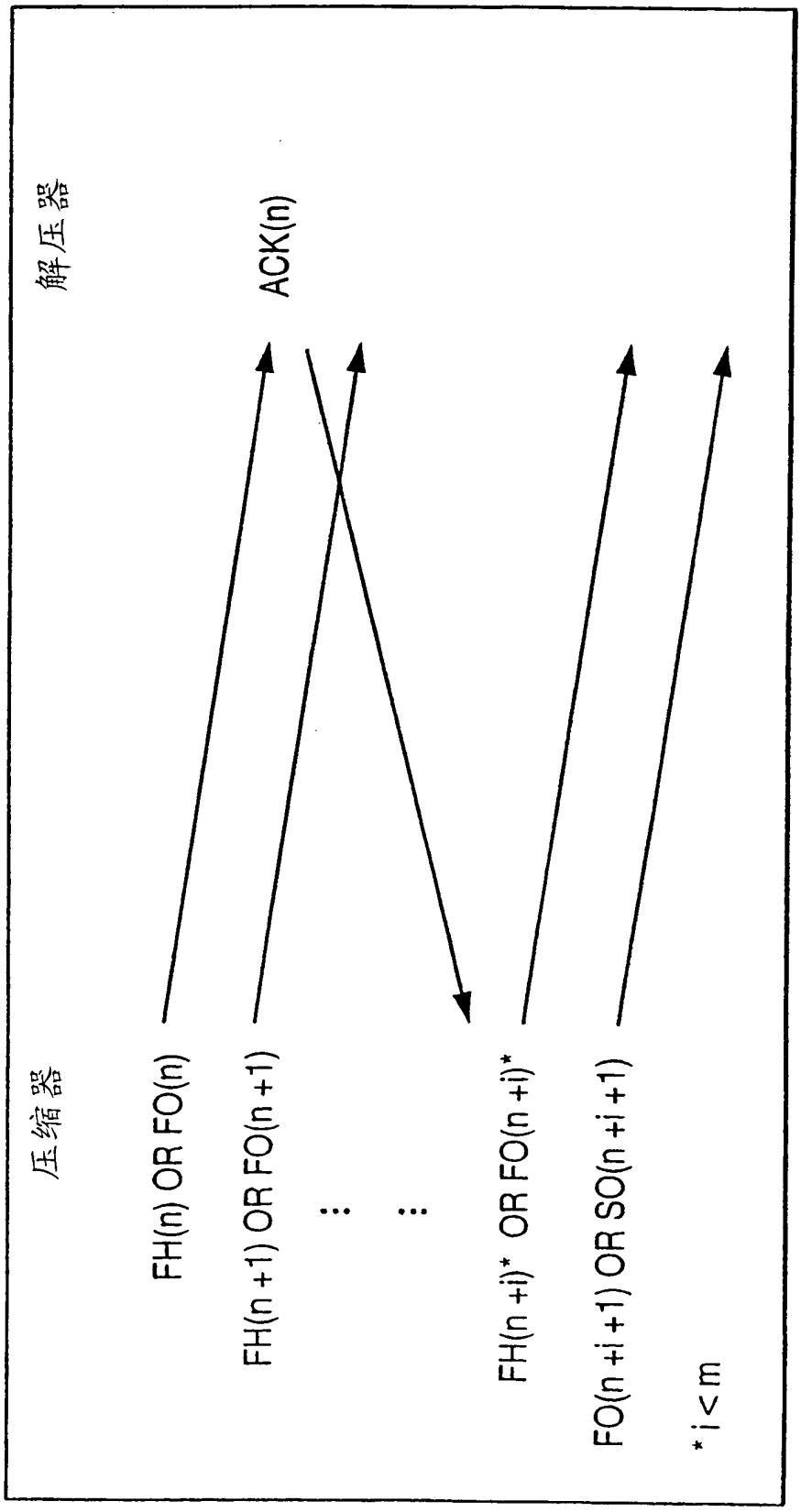

压缩器
解压器
FH(n) OR FO(n)
ACK(n)
FH(n +1) OR FO(n +1)
FH(n +i)* OR FO(n +i)*
FO(n +i +1) OR SO(n +i +1)
* i < m

图 16

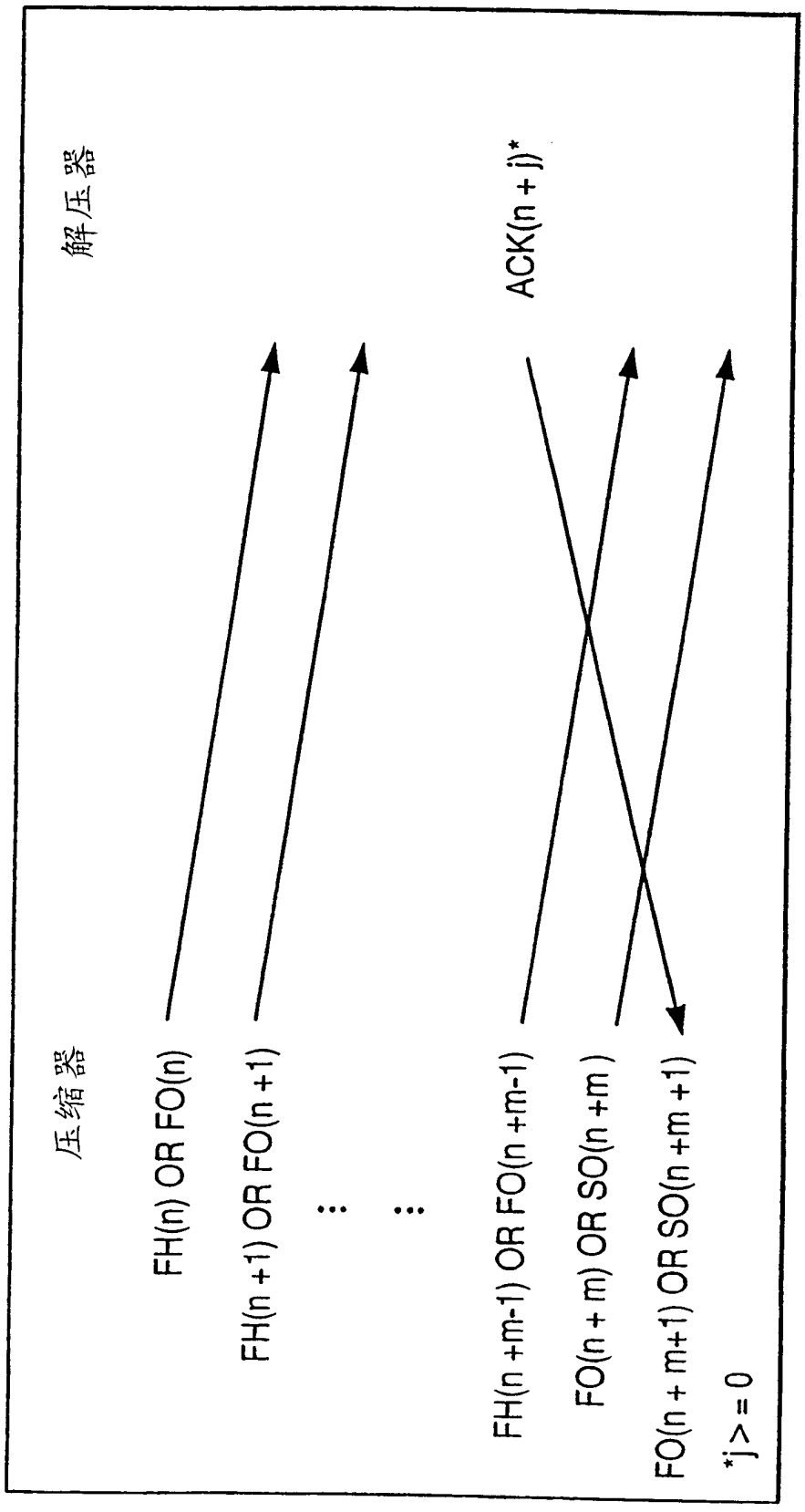

压缩器
解压器
FH(n) OR FO(n)
FH(n +1) OR FO(n +1)
FH(n +m-1) OR FO(n +m-1)
FO(n + m) OR SO(n +m)
FO(n + m+1) OR SO(n +m +1)
ACK(n + j)*
*j > = 0

图 17

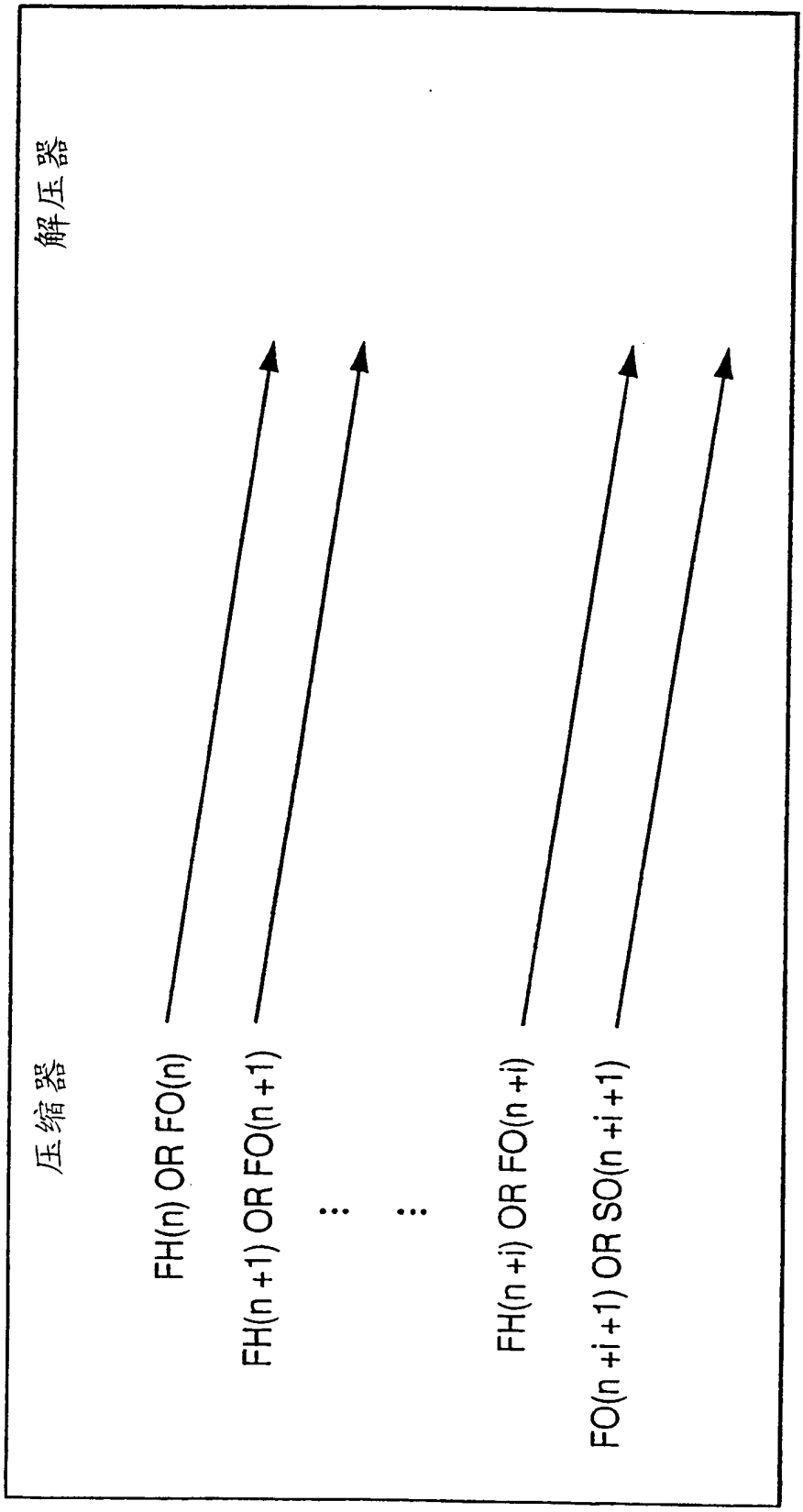


图 18

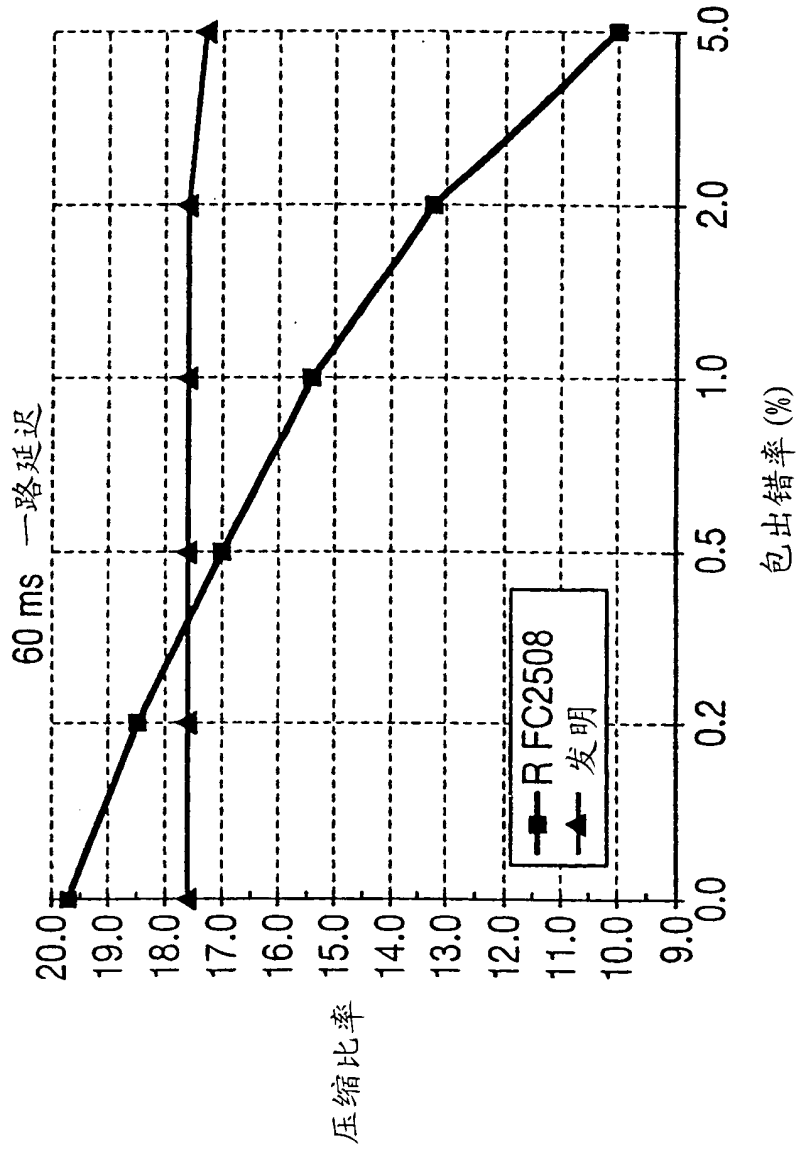

60 ms 一路延迟
压缩比率
20.0
19.0
18.0
17.0
16.0
15.0
14.0
13.0
12.0
11.0
10.0
9.0
0.0
0.2
0.5
1.0
2.0
5.0
包出错率(%)
RFC2508
发明

图 20