[19] 中华人民共和国国家知识产权局

[12] 发明专利说明书

专利号 ZL 02827774.0

[51] Int. Cl.
H04L 12/54 (2006.01)
H04L 5/14 (2006.01)
H04L 5/16 (2006.01)
H04L 12/66 (2006.01)
H04L 12/28 (2006.01)
H04J 3/16 (2006.01)

[45] 授权公告日 2008 年 7 月 23 日

[11] 授权公告号 CN 100405778C

[51] Int. Cl. (续)
G06F 15/00 (2006.01)
G06F 15/173 (2006.01)
G01R 31/00 (2006.01)

[22] 申请日 2002.12.5 [21] 申请号 02827774.0

[30] 优先权

[32] 2001.12.5 [33] US [31] 10/007,409

[86] 国际申请 PCT/US2002/038751 2002.12.5

[87] 国际公布 WO2003/048903 英 2003.6.12

[85] 进入国家阶段日期 2004.8.2

[73] 专利权人 瑞通网络公司

地址 美国加利福尼亚州

[72] 发明人 桑迪普·洛达 迪派克·阿特雷施

[56] 参考文献

CN1171685A 1998.1.28

WO9830059A1 1998.7.9

US6167027A 2000.4.18

US6144639A 2000.11.7

US5790131A 1998.8.4

审查员 时 鹏

[74] 专利代理机构 北京三幸商标专利事务所

代理人 刘激扬

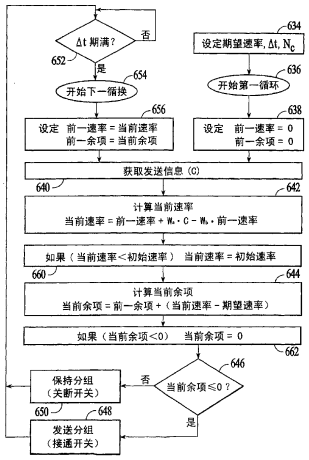
权利要求书 2 页 说明书 21 页 附图 12 页

[54] 发明名称

用于在基于分组的计算机网络中速率整形的方法和系统

[57] 摘要

本发明涉及一种用于在基于分组的计算机网络中速率整形的方法和系统。 在控制基于分组的流量的流动以满足期望的速率，包括计算在一个链路上基于分组流量的作为运动平均的当前速率，计算所计算的当前速率和期望速率之间的误差之和，并响应于所述计算的误差之和以确定分组是否能够流动。 当计算的当前速率和期望速率之间的误差之和表示当前速率小于期望速率时，允许分组流动，而当所述误差之和表示当前速率大于期望速率时，则不允许分组流动。 通过人工控制当前速率的最小值和误差之和还能够控制突发的数量。 流控制算法能够用于速率整形或速率限制。



1.一种用于控制基于分组的流量流动以满足期望速率的方法，该方法包括：

计算在一个链路上基于分组流量的作为运动平均的当前速率，所述当前速率如下计算：

当前速率 = 前一速率+($W_a \cdot C - W_b \cdot$ 前一速率)，

这里 C 表示比特数量，W_a 和 W_b 为加权，$W_a=1/(N_c \cdot \Delta t)$，和 $W_b=1/N_c$，Δt 为采样间隔和 N_c 为所述采样间隔的数量；

计算所述计算的当前速率和所述期望速率之间的误差之和，所述误差之和如下计算：

当前误差 = 前一误差+(当前速率-期望速率)；以及

响应于所述计算的误差之和以确定分组是否能够流动，如果所述误差之和小于误差门限则允许分组流动，以及如果所述误差之和大于所述误差门限，则不允许分组流动。

2.根据权利要求 1 所述的方法其中确定分组是否能够流动包括当所述误差之和表示获得的流量速率低于期望速率时允许分组流动，而当所述误差之和表示获得的流量速率高于期望速率时则不允许分组流动。

3.根据权利要求 1 所述的方法还包括，为所述当前速率设立一个最小值并且当所述当前速率低于所述最小值时调整所述当前速率的值。

4.根据权利要求 1 所述的方法包括为所述误差之和设立一个最小值并且当所述误差之和低于所述最小值时调整所述误差之和的值。

5.根据权利要求 1 所述的方法还包括：

为所述当前速率和所述误差之和设立一个最小值；

如果所述当前速率低于用于所述当前速率的所述最小值时调整所述当前速率的值；和

如果所述误差之和低于用于所述误差之和的所述最小值时调整所述误差之和的值。

6.根据权利要求 1 所述的方法，其中将所述当前速率计算为加权的运动平均。

7.一种用于控制输入链路和输出链路之间基于分组的流量流动以满足期望速率的系统，该系统包括：

连接于所述输入链路和所述输出链路之间的开关，当所述开关接通时允许流量从所述输入链路向所述输出链路流动，当所述开关关断时不允许分组流动；和

连接控制所述开关的速率整形逻辑设备，所述速率整形逻辑设备包括用于：

计算在所述输入和输出链路其中之一上基于分组流量的作为运动平均的当前速率的装置，所述当前速率如下计算：

当前速率 = 前一速率+($W_a \cdot C - W_b \cdot$ 前一速率)，

这里 C 表示比特数量，W_a 和 W_b 为加权，$W_a=1/(N_c \cdot \Delta t)$，和 $W_b=1/N_c$，Δt 为采样间隔和 N_c 为所述采样间隔的数量；

计算所述计算的当前速率和所述期望速率之间的误差之和的装置，所述误差之和如下计算：

当前误差 = 前一误差+(当前速率-期望速率)；以及

响应于所述计算的误差之和以确定分组是否能够通过所述输出链路流动的装置，如果所述误差之和小于误差门限则允许分组流动，以及如果所述误差之和大于所述误差门限，则不允许分组流动。

用于在基于分组的计算机网络中速率整形的方法和系统

技术领域

本发明涉及一种基于分组的计算机网络，并尤其涉及一种控制基于分组的网络内的流量速率。

背景技术

已知基于分组的网络用于以突发发送数据。突发流量能够导致网络拥塞并当输入缓冲器满时导致分组被丢弃。为了改善网络的性能并提供可靠的服务质量(QoS)，有必要能够控制网络中不同位置处的流量至特定的期望速率。控制流量的两种技术通常称为速率整形和速率限制。速率整形包括当分组到达一个位置时缓冲分组，并然后根据给定的算法控制分组流离开缓冲器以满足期望的速率。由于分组被缓冲，所以能够接收分组的突发，然后以可控的方式发送这些分组，从而不会丢弃进入的分组。速率限制包括测量到达一个位置的分组的速率，然后丢弃超出期望速率的分组。

一些用于实现速率整形或速率限制的公共算法包括熟知的漏桶和令牌桶算法。使用漏桶算法实现速率整形包括缓冲进入的分组然后以恒定的速率从缓冲器中发送这些分组，即漏速率(例如以字节/秒测量的)。例如，当接收大小为 C(即，几字节)的分组时，在其被从缓冲器发送之前，在缓冲器中将该分组保持 C 时间周期/漏速率。但是如果缓冲器为空的，那么如漏速率所指示的传输容

量变为未使用并且不用于累积分组的后续突发。所以，当分组的新突发在无流量的周期之后到达缓冲器时，新分组仍将被缓冲并以漏速率进行发送。这种技术不能为考虑到基于分组的网络流量的突发本质而提供灵活性。

使用令牌桶算法用于速率整形包括以恒定速率为桶分配比特值(即，1000 字节，50 单元等)并且只有必须数量的比特，字节，或单元在桶中累积时才允许从缓冲器中发送分组。在无流量的周期中，比特以恒定的速率继续在桶中累积并且流量的以下突发能够使用该累积的比特。尽管这种技术考虑到网络流量的突发本质而提供了灵活性，但是如果桶在无流量的周期中累积了太多的比特，那么将不可能整形后继的分组突发。可允许的突发量能够通过设定桶中所累积的大量比特的最大值来限制，但是桶的最大值必须至少和最大期望的分组一样大，以防止缓冲器中大的分组被截留(trap)。设定最大桶的大小为最大期望分组的大小其不利之处在于，这将允许相对小的分组的大量突发脉冲得不到整形。

能够用于速率整形或速率限制的另一个算法包括计算某一位置处的当前流量速率并比较当前速率和期望的速率以确定是否应该发送以下分组。根据该算法，如果当前速率小于期望速率，那么将允许分组流动，如果当前速率大于期望速率，那么将不允许分组流动。一种已知的算法诸如用于速率整形的算法，被称为指数加权运动平均(EWMA)算法。EWMA 算法的一个实例如下表示：

$$Current\_Rate_N = (1\text{-}Gain) \cdot Current\_Rate_{N-1} + Gain \cdot Bytes_N$$

其中，$t_{N+1} = t_N + \Delta t$

这里 $Current\_Rate_N$ 为第 N 个采样间隔的 EWMA，$Current\_Rate_{N-1}$ 为第 N-1 个采样间隔的 EWMA，以及 $Bytes_N$ 为在第 N 个采样间隔内发送的字节数量，每个采样间隔的周期为 Δt。

Gain 控制低通滤波器操作所必须的时间常量(即，频率响应)。当用于从缓冲器发送分组时，如果 $Current\_Rate_N$ 大于预定的期望速率(称之为 Desired_Rate)时，那么不发送任何分组。如果 $Current\_Rate_N$ 小于预定的期望速率时，那么发送这些分组。一种用于控制从缓冲器发送分组的算法在现有技术中如下表示：

如果($Current\_Rate_N$>Desired_Rate)

不发送分组

否则

发送分组。

图 1 描述了在速率整形系统中应用上述 EWMA 算法产生的流量 104 与时间(以 Δt 增加)比值的当前速率的示意图，其中缓冲器被填满了分组。在开始时间，t_0，发送流量的突发并且按照 EWMA 计算的当前速率超出了期望的速率，其中期望速率通过水平虚线 106 表示。只要当前速率高于期望的速率，那么就不发送任何新流量并且当前速率稳定地下降。一旦当前速率下降到低于期望的速率(即，时间 t_2)，则发送流量的新突发并且当前速率跳回到期望速率之上。如图 1 所示，在满载的系统中，流量的当前速率几乎完全地高于期望速率，因为当前速率下降到低于期望速率时，将发送更多的分组。由于当前速率几乎完全地高于期望速率，因此在满载周期达到的传输速率将大于期望的速率，这里根据由时间周期分割的发送比特总数计算达到的速率，其中在该时间周期上发送这些比特。例如，在图 1 中，在满载的系统中在时间周期 t_0 到 t_9 上达到的速率 108 高于期望的速率。理想状态下，达到的速率应该等于期望的速率。

就需要能够控制特定网络流量至期望的速率，以及现有技术速率整形和速率限制算法的缺点而言，实际需要一种能够有效地

控制基于分组的流量速率的算法，即灵活地并容易地实现，特别是在硬件中实现。

发明内容

一种用于控制基于分组的流量流动以满足期望速率的方法和系统包括，计算在一个链路上基于分组流量作为运动平均的当前速率，计算计算的当前速率和期望速率之间的误差之和，并响应于计算的误差之和以确定分组是否能够流动。在一个实施例中，当当前速率和期望速率之间的误差之和小于或等于零时，从缓冲器发送分组到链路上，并且当误差之和大于零时，在缓冲器中保持该分组。由于以响应于当前速率和期望速率之间的误差之和来控制分组流动，所以即使在突发流量环境中，也能保持所获得的流量速率接近于期望的速率。

在本发明的一个实施例中，比较误差之和与门限值以确定是否允许有任何分组流动。如果误差之和低于该门限则允许分组流动，如果误差之和高于该门限，则不允许分组流动。在一个实施例中，门限值为零，如果误差之和小于或等于零则允许分组流动。在另一个实施例中，当误差之和表示达到的流量速率低于期望的速率时允许分组流动，而当误差之和表示达到的流量速率高于期望的速率时则不允许分组流动。

为了控制在无流量周期过后允许流动的突发的大小，可为当前速率设立一个最小值。如果当前速率低于该最小值，则将计算的当前速率调整为该最小值。同理，为误差之和建立一个最小值，如果误差之和低于最小值，则将计算的误差之和调整为该最小值。这些技术都控制从静默周期“所借”带宽的大小。特别是，后者的(第二)技术防止大量的负余项累积。

在一个实施例中，计算当前速率和误差之和以及确定是否在每个周期分组能够重复流动，其中每Δt发生一次循环。

在一个实施例中，如下计算当前速率：

当前速率 = 前一速率+($W_a \cdot C - W_b \cdot$ 前一速率)，

这里C表示在当前周期中发送的比特数量，W_a和W_b为加权。当使用上述提供的表达式时，能够将用于下一个计算周期的前一速率设定为在上一个周期所计算的当前速率。在一个实施例中，W_a为N_C和Δt的函数，这里Δt为采样间隔和N_C为采样间隔的常量数量。具体而言，W_a可以被定义为$W_a = 1/(N_C \cdot \Delta t)$。在一个实施例中，$W_b$为$N_C$的函数。更具体而言，$W_b$可以被定义为$W_b = 1/N_C$。

当使用上述提供的速率表达式时，通过设立一个初始速率并如果当前速率降低到低于初始速率时将当前速率调整为初始速率，从而能够控制流量突发的大小。

误差之和可以如下计算：

当前误差 = 前一误差+(当前速率-期望速率)，

这里根据上述提供的表达式来计算当前速率。通过为当前误差设立一个最小门限，并如果当前误差降低到低于该门限时，将当前误差调整为该门限，从而能够控制分组突发的大小。当使用上述提供的误差表达式时，将下一个计算周期的前一误差设定为在上一个计算周期中计算的当前误差。在一个实施例中，如果当前误差小于或等于零则允许分组流动，如果当前误差大于零则不允许分组流动。

本发明能够以控制基于分组的流量流动的方法和系统来实施。该方法和系统能够用于速率整形或速率限制并且，通过调整用户提供的初始速率，期望速率，N_C和Δt的输入参数，从而能够容易地调整算法的性能特性。

本发明还能够以特别用于速率整形的方法来实施，其中响应于计算的当前速率和期望速率之间的计算误差之和，从而从缓冲器中发送分组。

根据以下结合附图通过实例的方式示例本发明原理的详细描述，本发明的其他方面和好处将显示出来。

附图说明

图 1 描述了根据现有技术在速率整形系统中应用 EWMA 算法产生的流量速率与以 Δt 增加的时间之比值的示意图，其中缓冲器被填满了分组。

图 2 描述了根据本发明的实施例实现速率整形算法的速率整形系统。

图 3A 描述了根据本发明的实施例在实现速率整形算法中产生的流量轮廓图。

图 3B 描述了图 3A 中计算的当前速率和期望速率之间误差之和的图。

图 3C 描述了响应于图 3B 中当前速率和期望速率之间误差之和而产生的开关控制信号图。

图 4 描述了根据本发明的实施例用于控制基于分组的流量的流率以满足期望的速率。

图 5A 描述了根据本发明的另一个实施例使用图 4 所示的速率整形算法产生的未调整速率轮廓的图，其中包括一部分调整的速率轮廓。

图 5B 描述了根据本发明的另一个实施例在计算速率和根据图 5A 中速率轮廓产生的期望速率之间误差之未调整的和，其中包括一部分调整的误差之和。

图 6 描述了根据本发明的另一个实施例，用于控制基于分组的流量的流率以满足期望速率的方法流程图，其中期望速率已考虑无流量的周期而被修改。

图 7 描述了根据本发明的一个实施例，连接至相同输出链路并由缓冲器专用开关控制的两个缓冲器的示意性实施例，其中实现了图 4 或图 6 的速率整形算法的多个实例以允许未使用的带宽由其它缓冲器使用。

图 8 描述了根据本发明的一个实施例，通过图 7 的一个选择器实现的逻辑矩阵实例以使得能够共享可用的带宽。

图 9 描述了根据本发明的一个实施例，QoS 系统的一个实施例，其中使用图 4 或图 6 的速率整形算法从多个缓冲器发送分组。

具体实施方式

一种用于控制基于分组的流量流动以满足期望速率的方法和系统包括，计算在一个链路上基于分组流量的作为运动平均的当前速率，计算被计算的当前速率和期望速率之间的误差之和，并响应于计算的误差之和以确定分组是否能够流动。在一个实施例中，当当前速率和期望速率之间的误差之和小于或等于零时，从缓冲器发送分组到链路上，并且当误差之和大于零时，在缓冲器中保持该分组。由于响应于当前速率和期望速率之间的误差之和来控制分组流动，所以即使在突发流量环境中，也能保持获得的流量速率接近于期望的速率。

图 2 描述了为本发明关键的能够用于实现速率整形算法的速率整形系统的实例。速率整形系统包括一个缓冲器 112，一个开关 114，以及速率整形逻辑 116。缓冲器在输入链路 118 上接收发来的分组并缓冲这些分组直到其被发送到输出链路 120。该开关为缓

冲器和输出链路之间的网关。该开关受速率整形逻辑的控制并且速率整形逻辑实现如下所述的速率整形算法。速率整形算法的一个实施例需要提供四个域(初始速率，期望速率，Δt 和 N_C)中的初始化信息如用户定义的输入参数给速率整形逻辑。域初始速率，期望速率，Δt 和 N_C 如下定义。在工作中，速率整形逻辑接收实际的发送信息，与输出链路相关的 C，并输出一个开关控制信号(Switch_Control)给开关。速率整形逻辑产生的控制信号使开关接通(如果它不总是接通)，以便从缓冲器发送分组或关断(如果它不总是关断)，以便在缓冲器中保持该分组。尽管图 2 所述的开关作为从缓冲器发送分组的机制，但是其它的机制也能够用于从缓冲器发送分组。如这里所述，输入和输出链路可能包括一个位于集成电路的总线，位于电路板上的总线，介质诸如电线，光纤，连接两个网络设备的 RF 信号，或其它用于传送分组化信息的链路。输入和输出链路还可能包括上述链路的任意组合。

在执行本发明算法中的基本步骤之一包括计算基于分组的流量的运动平均，当前速率。以下描述了一种用于计算基于分组流量的当前速率和运动平均的技术，然后描述了一种用于计算计算的当前速率和期望速率之间误差之和的技术。参考图 3 到图 9 更详细地描述速率整形算法的一个实施例。

在给定时间周期 T 的基于分组的流量速率(即，比特/秒)能够如下表示：

$$R = \frac{\sum C}{T}$$

其中 ΣC 为时间 T 中发送的比特总数。在 N 个采样间隔的采样域中，速率能够如下表示：

$$Rate_N = \frac{\sum_{i=0}^{N} C_i}{N \bullet \Delta t}$$

这里Δt为采样间隔，N为在其上计算速率的采样间隔的数量。这意味着有关在第N个间隔之前发生的采样间隔的以下速率：

$$Rate_{N-1}=\frac{\sum_{i=0}^{N-1}C_i}{(N-1)\bullet\Delta t}$$

接下来在采样间隔0到N中发送的比特之和能够如下表示：

$$\sum_{i=0}^{N}C_i=\sum_{i=0}^{N-1}C_i+C_N$$

利用 $Rate_N$ 和 $Rate_{N-1}$ 代替上述式子：

$$Rate_N \cdot N \cdot \Delta t=Rate_{N-1} \cdot (N-1) \cdot \Delta t+C_N$$

因此能够递归表示 $Rate_N$ 为：

$$Rate_N=Rate_{N-1}+\frac{C_N}{N\bullet\Delta T}-\frac{Rate_{N-1}}{N}$$

或

$$Rate_N=Rate_{N-1}+W_a \cdot C_N- W_b \cdot Rate_{N-1}$$

其中如下定义加权 W_a 和 W_b：

$W_a=\frac{1}{N\bullet\Delta t}$，和

$W_b=\frac{1}{N}$

尽管在上述推导中N为变量，但是也能够将N设定为常量值以简化速率计算。在一个实施例中，N被设定为常量值，N_c，用于计算加权 W_a 和 W_b，因此：

$W_a=\frac{1}{N_c\bullet\Delta t}$，和

$W_b=\frac{1}{N_c}$

这里 N_c 表示采样间隔数量的常数。速率计算能够如下表述：

当前速率 = 前一速率+[$W_a \cdot C-W_b \cdot$ 前一速率]

这里当前速率与 $Rate_N$ 有关，前一速率与 $Rate_{N-1}$ 有关，这里使用 N_c 计算 W_a 和 W_b，并且C为与在最近采样间隔中发送的与比特数量相关的实际发送信息。

在图 2 的实施例中，C 表示在最后的采样间隔，发送给系统的输出链路 120 的实际比特数量。在图 2 中，逻辑线路 122 表示与提供给速率整形逻辑的输出链路相关的实际发送信息。尽管逻辑线路 122 与图 2 的输出链路相连接，但是实际的发送信息能够通过各种技术提供给速率整形逻辑，所使用的确切技术对于本发明而言不是关键。例如，实际发送信息可能被从缓冲器提供。

执行该算法的其它基本步骤包括计算计算的当前速率和期望速率之间的误差之和。在整个描述中，计算的当前速率和期望速率之间的误差被称为“余项”，尽管应该理解术语“误差”和“余项”可相互替换。在 N 个采样间隔的采样域中，第 N 个采样间隔的瞬时余项表示为 $\text{Residue}_N = \text{Rate}_N - \text{Desired\_Rate}$。第 N 个采样间隔中的余项之和在这里表示为：

$$\sum_{i=0}^{N} \text{Residue}_N = \sum_{i=0}^{N} (Rate_N - \text{Desired\_Rate})$$

这意味着在第 N 个采样间隔之前发生的有关采样间隔的以下式子为：

$$\sum_{i=0}^{N-1} \text{Residue}_{N-1} = \sum_{i=0}^{N-1} (Rate_{N-1} - \text{Desired\_Rate})$$

所以，第 N 个采样间隔的瞬时余项之和还能够以术语即第 N-1 个采样间隔余项之和加上第 N 个采样间隔的瞬时余项如下表示：

$$\sum_{i=0}^{N} \text{Residue}_N = \sum_{i=0}^{N-1} \text{Residue}_{N-1} + \text{Residue}_N$$

用 $\text{Residue}_N = \text{Rate}_N - \text{Desired\_Rate}$ 代入得到：

$$\sum_{i=0}^{N} \text{Residue}_N = \sum_{i=0}^{N-1} \text{Residue}_{N-1} + (\text{Rate}_N - \text{Desired\_Rate})$$

所以，第 N 个采样间隔的余项之和还能够以术语当前余项和前一个余项如下表示：

当前余项=前一余项+(当前速率 - 期望速率)

这里当前速率如上所述进行计算，前一余项与上一个采样间隔余项之和有关，并且期望速率被提供作为一个用户输入。

使用速率整形的算法包括在每个采样间隔都计算当前速率和当前余项，即每个Δt，并且然后基于当前余项的值产生一个开关控制信号。在一个实施例中，如果当前余项小于或等于零，则产生使开关发送分组的控制信号(即，“1”)，如果当前余项大于零，则产生使开关保持分组的控制信号(即，“0”)。当当前余项小于或等于零时发送分组，因为余项之和表示获得的速率当前等于或低于期望速率，反之当当前余项大于零时则保持这些分组，因为余项之和表示获得的速率当前低于期望速率。使用该算法，在时间的扩展周期中，分组流量获得的速率将接近期望的速率。尽管在本实施例中，余项值为零且为发送和保持分组的关键值，在其他实施例中该值可能被数学操作为除了零之外的特定值。

在一个实施例中，循环执行速率整形算法，这里每个循环间隔为与采样间隔相同的时间周期Δt。当开始新的循环时，要更新前一速率和前一余项的值以反映当前的值。具体而言，在前一个循环中计算的当前速率用于前一速率并且在前一个循环中计算的当前余项用于前一余项。整个速率整形算法能够以伪码来表示，例如表示为：

//速率整形算法：每个Δt执行。

//部分I：更新速率和余项：

当前速率 = 前一速率+$W_a \cdot C - W_b \cdot$ 前一速率；

当前余项 = 前一余项+(当前速率 - 期望速率)；

//部分II：更新切换控制：

如果(当前余项<=0)Switch_Control = 1；

否则 Switch_Control = 0；

前一速率 = 当前速率；

前一余项 = 当前余项；

图 3A，3B，和 3C 为一系列时间同步图，其表示实现上述速率整形算法的示例结果。图 3A 为通过实现上述算法产生的速率轮廓图。该图描述了分组流量的当前速率 304 的一个实例，该速率根据运动平均与分组流量的期望速率 306 相关的时间(以 Δt 增加)之比值而计算。图 3B 为根据图 3A 中的速率轮廓产生的余项 328 之和以及图 3C 为响应于余项之和根据速率整形算法产生的开关控制信号 330 的图。在本实例中，开关控制信号“1”使得分组被发送以及开关控制信号“0”使得分组被保持。

在描述图 3A-3C 的示例结果中，假设在时间 t_0 初始化该算法并且与此同时从缓冲器发送分组的突发。在时间 t_0 的被发送分组使得当前速率迅速上升，如图 3A 所示。在时间 t_1，当前速率的增加使得当前余项从零变为大于零的特定值，这反过来使得开关控制信号从“1”变为“0”，从而保持任何剩余的被缓冲的被保留。尽管切换不是发送新的分组，但是分组流量的当前速率在时间 t_0 和 t_8 之间如图 3A 所示以当前速率的向下斜率而降低。响应于当前速率，如图 3B 所示，只要当前速率高于期望速率则当前余项就增加，一旦当前速率低于期望速率就降低。当前余项的响应产生于如上所述的当前余项，当前速率，和期望速率之间的关系。在当前余项变得小于或等于零的该点上，开关控制信号从“0”改变为“1”，以允许从缓冲器发送分组的其他突发。例如，在时间 t_8，当前余项等于零，这使得开关控制信号从“0”变到“1”，如图 3C 所示。当开关控制信号变为“1”时，释放分组的其他突发并且当前速率迅速上升，如在时间 t_8 通过当前速率所示。参考图 3B 和 3C，每当当前余项变为零时开关控制信号从“0”切换到“1”，然后一旦当前余项增加到大于零则开关控制信号从“1”再切换回“0”。这种机制确保了在满载的系统中整个平均流量速率保持在

或接近期望速率。

图 4 描述了用于执行上述速率整形算法的方法流程图的实施例。在开始第一循环之前，在步骤 434 为速率整形算法设定期望速率值，Nc，和Δt。设定期望速率值，Nc，和Δt 可能包括手动用户输入或响应于较宽 QoS 系统产生的自动输入。在第一循环开始时，即步骤 436，在步骤 438 设定前一速率和前一余项为零。在步骤 440，获取发送信息，C。在步骤 442，如上所述使用所提供的表达式计算当前速率。用于计算当前速率的值包括前一速率，C，W_a，和 W_b。在步骤 444 计算当前速率之后，如上所述使用所提供的表达式计算当前余项。用于计算当前余项的值包括前一余项，当前速率，和期望速率。然后计算的当前余项用于在判断点 446 确定是否应该发送或保持这些分组。如果当前余项小于或等于零则，在步骤 448，发送分组；如果当前余项大于零，则在步骤 450，保持该分组。在一个实施例中，发送分组包括产生一个打开开关的控制信号，保持分组包括产生一个关断开关的控制信号。在判断点 452 作出发送或保持分组的判断之后，如果采样间隔Δt 已终止，在步骤 454 开始下一个循环。为了开始下一个循环，在步骤 456，在上一循环计算的当前速率值作为前一速率用于以下的速率计算，并在上一循环计算的当前余项的值作为前一余项用于以下的余项计算。在设定前一速率和前一余项的值之后，获得发送信息，C(步骤 440)的方法，计算当前速率(步骤 442)，计算当前余项(步骤 444)，以及如上所述重复地确定是否发送或保持分组(判断点 446)。每Δt 时间间隔连续地重复该过程。

尽管当速率整形系统的输入缓冲器满载时上述算法也工作正常，但是当突发通过无流量的周期时，速率整形算法可以允许发送发来的突发而不经过速率整形。当突发通过无流量的周期时，

可能发送这些分组而不经过速率整形，因为无流量的周期将允许积累相对大的负余项。根据上述算法，只要当前余项保持低于零，则相对大的负余项将允许发送分组的突发脉冲。为了在无流量周期过后防止流量的大量突发，在特定情况下，将当前速率和当前余项的值人工控制为预先设立的门限。具体而言，在一个实施例中，控制当前速率以便不允许它降低到低于预先设立的速率，这里称之为初始化速率，并控制当前余项以便它不会降低到低于零。保证当前速率和当前余项不会降低到低于特定的门限这样控制能够从无流量的前一个周期所“借”的带宽量。图 5A 和 5B 描述了无流量周期所引起的问题以及对当前速率和当前余项所作的调整以保证所有的分组被速率整形，即使在无流量周期过后。图 5A 描述了使用速率整形算法产生的速率轮廓图，以及图 5B 为由图 5A 的速率轮廓而产生的余项之和的时间同步图。

参考图 5A，为了示例的目的假设，在时间 t_0 流量的初始化突发之后跟随一个无流量的周期。当时间经历时，计算的当前速率 504 向零降低除非手动进行调整。为了保持当前速率为最小值，调整当前速率为预定的初始速率 556，而不论何时计算的当前速率低于初始速率。图 5A 描述了如何调整当前速率为预定的初始速率，而不论何时计算的当前速率低于初始速率。

在一个实施例中，使用模拟来找到初始速率值。该模拟用于找到在满载有分组的系统中产生的当前速率的最小值。然后产生的最小值被设定作为初始速率。在一个实施例中，当开始模拟时，初始化当前速率为期望速率。

参考图 5B，不论何时来自图 5A 的当前速率 504 小于期望速率 506，都将减少当前余项 528。当当前速率小于期望速率时除非手动地调整它，否则当前余项将无期限地减少。为了将当前余项

保持在最小值，不论何时计算的当前余项小于零，都将当前余项调整为零。图 5B 描述了已被调整为零的当前余项的一部分，因为计算的当前余项小于零。尽管在本实施例中，当前余项被调整为当前余项的最小值零，但是当前余项能够被调整为某些其他的最小门限值。

用于更新的速率整形算法的伪码的一个实施例如下：

//经修改的速率整形算法：每个 Δt 执行。

//部分 I：更新速率和余项：

当前速率 = 前一速率+$W_a \cdot C - W_b \cdot$ 前一速率；

如果(当前速率<初始速率)当前速率 = 初始速率；

当前余项 = 前一余项+(当前速率 - 期望速率)；

如果(当前余项<0)当前余项 = 0；

//部分 II：更新切换控制：

如果(当前余项=0)Switch_Control = 1；

否则 Switch_Control = 0；

前一速率 = 当前速率；

前一余项 = 当前余项；

图 6 描述了用于执行经过上述修改的图 4 的速率整形算法的方法流程图。该方法流程图类似于图 4 的方法流程图，除了在步骤 634 设立除其他参数之外的初始速率，在其他步骤 660，如果当前速率小于初始速率则将当前速率设定为初始速率，在其他步骤 662，如果当前余项小于零则将当前余项设定为零，并且在判断点 646，如果当前余项等于零则发送分组，如果当前余项大于零则保持该分组。

在一个可替换的实施例中，能够在多缓冲器系统中实现上述速率整形算法以允许连接到相同链路的其他缓冲器使用可用的带

宽。具体而言，当第二缓冲器不能保持任何分组时(即，当第二缓冲器未充分使用它的可用带宽)，多缓冲器系统允许将分配给第二缓冲器的带宽由第一缓冲器借用。图 7 描述了连接到公共输出链路 764 并受缓冲器专用开关 714 所控制的两个缓冲器(缓冲器 1 和缓冲器 2)712 的示例性实施例。每个缓冲器专用开关通过缓冲器专用选择器 766 与两个缓冲器专用速率整形器 716 和 717 相连接。缓冲器专用速率整形器实现了两种版本的上述速率整形算法。如虚线 768 所示，每个选择器从其他缓冲器接收缓冲器状态信息并且如线 722 所示，每个速率整形器接收与其各自的缓冲器相关的实际发送信息。

在图 7 的实施例中，假设从缓冲器 1 发送的速率需要控制为期望速率并且发送的速率能够被控制为最大速率，如果在公共输出链路 764 上存在可用的未使用带宽。对于缓冲器 2 保持相同。为了在每个缓冲器都获得期望的速率整形，两个速率整形器与每个缓冲器相关联。参考缓冲器 1，一个速率整形器，RS1_Desired716，设定速率整形缓冲器 1 为期望速率(即，算法中的期望速率域被设定为期望速率)并且另一个速率整形器，RS1_Maximum717，设定速率整形缓冲器 1 为最大速率(即，算法中的期望速率域被设定为最大速率)。同样，参考缓冲器 2，RS2_Desired716，设定速率整形缓冲器 2 为期望速率(即，算法中的期望速率域被设定为期望速率)并且另一个速率整形器，RS2_Maximum717，设定速率整形缓冲器 2 为最大速率(即，算法中的期望速率域被设定为最大速率)。在图 7 的实施例中，当每个缓冲器中存在流量时，那么通过速率整形器 RS1_Desired 和 RS2_Desired 分别将每个缓冲器控制为它的期望速率。但是，如果其中的一个缓冲器没有已缓冲的流量，那么能够允许其他缓冲器

上升到其最大速率。

参考图 7 和图 8 为了示例的目的描述了利用双速率整形器对缓冲器 1 速率整形。图 8 描述了选择器 766 为缓冲器 1 实现的逻辑矩阵 770 的实例。如果从 RS1_Desired716 输出的控制信号为“1”，表示能够从缓冲器 1 发送分组，那么选择器输出一个控制信号“1”到开关并发送该分组。不管 RS1_Maximum717 的值如何以及是否在缓冲器 2 中存在分组，选择器将输出控制信号“1”。如果从 RS1_Desired 输出的控制信号为“0”，以及 RS1_Maximum 输出的控制信号为“0”，那么选择器输出一个控制信号“0”到开关而不管在缓冲器 2 中是否存在分组。如果从 RS1_Desired 输出的控制信号为“0”，以及 RS1_Maximum 输出的控制信号为“1”，那么选择器将查询缓冲器 2 的状态以确定是否能够发送分组。如果在缓冲器 2 中存在分组，选择器则输出一个控制信号“0”并从缓冲器 1 不发送任何分组。不从缓冲器 1 发送分组因为，保留用于缓冲器 2 中分组的带宽不能由其他缓冲器使用。但是，如果在缓冲器 2 中不存在分组并且 RS1_Maximum 为“1”，那么用于开关 1 的选择器将输出一个控制信号“1”到开关 1，并将从缓冲器 1 发送分组。从缓冲器 1 发送分组受到 RS1_Maximum 的控制，只要在缓冲器 2 中不存在任何分组以及 RS1_Desired 保持为“0”。

能够依比例决定参考图 7 所述的系统包括控制多于两个的缓冲器。在依比例决定的系统中，用于每个缓冲器的选择器监视所有其他缓冲器的缓冲器状态，或其他缓冲器的某些组合并使用缓冲器信息确定是否能够以指定的最大速率来发送分组。

上述速率整形算法还能够用于同时控制多个缓冲器。图 9 描述了其中通过相同速率整形逻辑控制多个缓冲器的 QoS 系统的一个实施例。QoS 系统包括一个分组分类器 972，缓冲器访问控制器

974，存储器 976，速率整形逻辑 916，以及调度器 978。存储器包括多个受缓冲器专用开关 914 控制的缓冲器(即，缓冲器 1 到 M)。缓冲器专用开关中的每一个通过如上所述的速率整形逻辑控制。速率整形逻辑接收与各自的输出链路 920 相关的实际发送信息，并使用实际发送信息执行速率整形算法和产生开关控制信号。

在工作中，分类器 972 确定每个发来的分组应该被缓冲到哪个缓冲器中。在分组分类之后，缓冲器访问控制器 974 确定是否能够将分组转发到它们的适当缓冲器中。在一个实施例中，缓冲器访问控制器使用加权随机提前丢弃(WRED)算法确定是否应该将分类的分组转发到各个缓冲器。速率整形逻辑 916 确定是否通过调度器 978 接收缓冲的分组。在一个实施例中，速率整形逻辑在每个循环中为每个缓冲器执行参考图 6 所描述的步骤并输出开关控制信号(Switch_1_Control 和 Switch_M_Control)给各个开关。根据开关的状态，调度器发送分组。例如，发送分组可能用于通过切换结构进行切换。

在一个实施例中，在硬件中实现参考图 2 到 9 描述的速率整形算法。如上所述，当前速率的计算需要两步乘法运算。在硬件实现中，乘法运算需要相对大量的资源并且期望限制硬件实现所需的资源。一种减少乘法运算所需资源的方法是将乘法运算减少为移位。以下将描述其中将两步乘法运算减少为移位的速率整形算法的硬件实施例。

加权常量 W_a 如上定义为 $1/(N_C \cdot \Delta t)$，N_C 为时间间隔的数量。N_C 能够被选择为 2 的幂并且在一个实施例中，N_C 为位于(包括)2^9 到 2^{13} 之间的 2 的幂。如上所述，每 Δt 时间间隔执行一次该算法。为了将乘法运算减少为移位，还选择时间间隔 Δt 以便它为 2 的幂，更具体而言，2 的负幂。在一个实施例中，时间间隔 Δt 为硬件时

钟时间(或周期)以及受速率整形算法控制的缓冲器数量的函数。在优选实施例中，Δt应该大于时钟时间乘以受速率整形算法控制的缓冲器数量(即，Δt>时钟时间·缓冲器数量)，从而能够为每个缓冲器在每个Δt时间中更新当前速率，当前余项，以及开关。注意到假设需要花费一个时钟周期来为单个的缓冲器更新当前速率，当前余项，以及开关。表1提供了Δt的示例值，该值能够用作受速率整形算法控制的缓冲器数量的函数以及时钟周期的函数。该表还包括了能够使用的最小Δt以及选择作为2的负幂的Δt。

表 1

缓冲器数量	时钟周期	最小Δt	所选择的Δt	2^(-x)幂的Δt
32	10ns	320ns	470ns	2^(-21)
256	10ns	2560ns	3810ns	2^(-18)
32	10ns	512ns	953ns	2^(-20)

加权常量定义为 $1/(N_C \cdot \Delta t)$。W_{ra}常量定义为：$W_{ra} = \log_2(W_a)$并且如果 $1/(N_C \cdot \Delta t)$为2的幂并且大于1，那么 $W_a \cdot C$减少为以下式子：

$$W_a \cdot C \approx (C << W_{ra})$$

因此，乘法运算被减少为向左移位运算。

加权常量 W_b定义为 $1/N$。常量 W_{rb}能够定义为：$W_{rb} = \log_2(W_b)$并且如果 $1/N$大于2的幂并小于1，那么 $Previous_Rate \cdot W_b$减少为以下式子：

$$Previous\_Rate \cdot W_b \approx (Previous\_Rate >> W_b)$$

因此，乘法运算被减少为向右移位运算。

由于移位导致丢失比特，所以可能降低计算的精确性。为了避免精确性的降低，实施定点运算并且十进制之后的比特数量称

为精确度。硬件实现的速率整形算法能够以伪码来表示，为了示例的目的，如下表示：

//硬件实现的速率整形算法：每个Δt执行。

//部分I：更新速率和余项：

当前速率 = 前一速率+((C<<精确度)<<W_{ra}) - (前一速率>>W_{rb});

如果(当前速率<初始速率)当前速率 = 初始速率;

当前余项 = 前一余项+(当前速率 - 期望速率);

如果(当前余项<0)当前余项 = 0;

//部分II：更新切换控制：

如果(当前余项=0)开关控制 = 1;

否则开关控制 = 0;

前一速率 = 当前速率;

前一余项 = 当前余项;

在一个实施例中，在使用可变长度分组的环境中实现速率整形算法，例如，以太网环境，但是，速率整形算法还能够在使用固定长度单元(如分组)的环境中实现。例如，异步传输模式(ATM)环境。

尽管这里实现的算法用于完成速率整形，但是该算法还能够用于完成速率限制或任何其他速率控制技术。当该算法用于速率限制时，当当前余项低于门限(即零)时丢弃分组，并且当当前余项高于门限时发送该分组。

在上述算法的实施例中，以特定的方式使用提供的方程来计算当前速率和期望速率之间的误差之和。但是，应该理解，可以以不同方式来计算当前速率和期望速率之间的误差之和而不偏离这里所描述的并且以下请求保护的本发明的范围和实质。

不论以硬件或软件来实现速率整形算法，通过简单地调整任何用户输入域，如初始速率，期望速率，Δt 和 Nc 都能容易地调整算法性能。对于初始速率，较低的初始速率可有效地允许从无流量的前一周期“借”更多的带宽，允许在静默周期过后大量的分组突发流动。对于Δt，较大的Δt 趋于使得流量轮廓更具突发性，因为开关更新之间的时间周期较长。对于 Nc，Nc 的值越小，则计算的当前速率将越更接近地跟踪瞬时速率，产生了较快的响应时间。通理，Nc 的值越大，计算的当前速率将不能更接近地跟踪瞬时速率，产生了较慢的响应时间。

速率整形算法能够在一个宏观层次上实现，其中开关表示网络中的一个节点或在较低层次上实现，其中开关控制从特定缓冲器的发送功能。而且，开关能够是控制分组流入链路的任何机制。

尽管已经描述并示例了本发明的具体实施例，但是本发明并不限制于这些描述和示例的具体形式或部分安排。本发明仅仅通过权利要求书来限定。

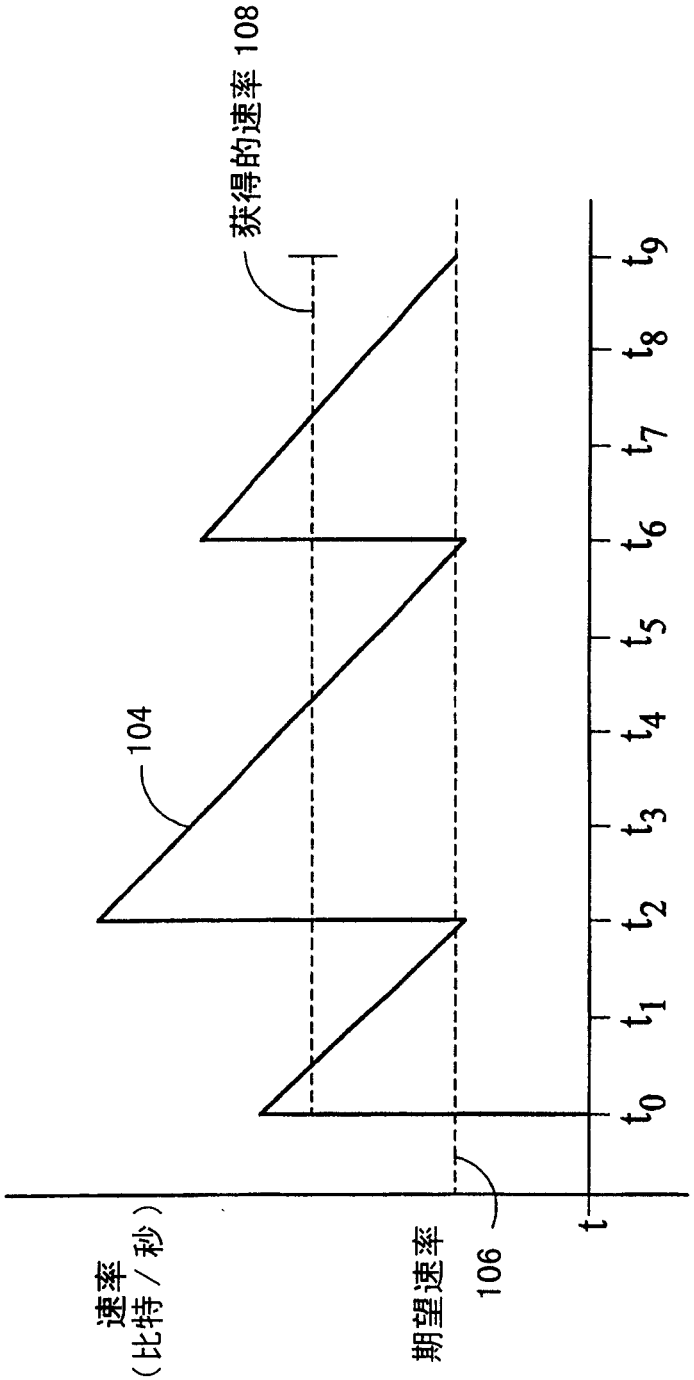


图 1

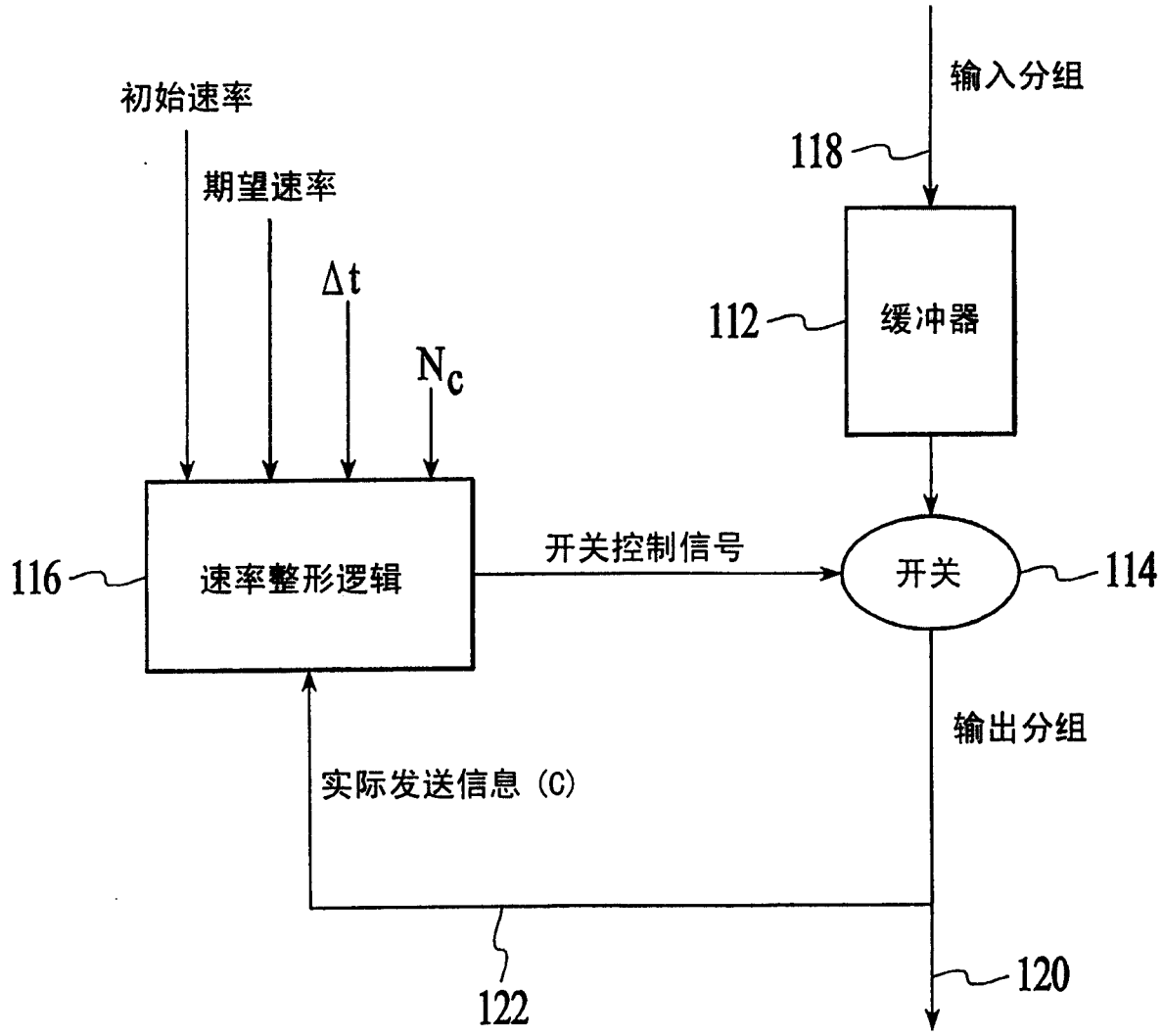

初始速率
期望速率
Δt
Nc
116
速率整形逻辑
开关控制信号
输入分组
118
112
缓冲器
开关
114
输出分组
实际发送信息（C）
122
120

图 2

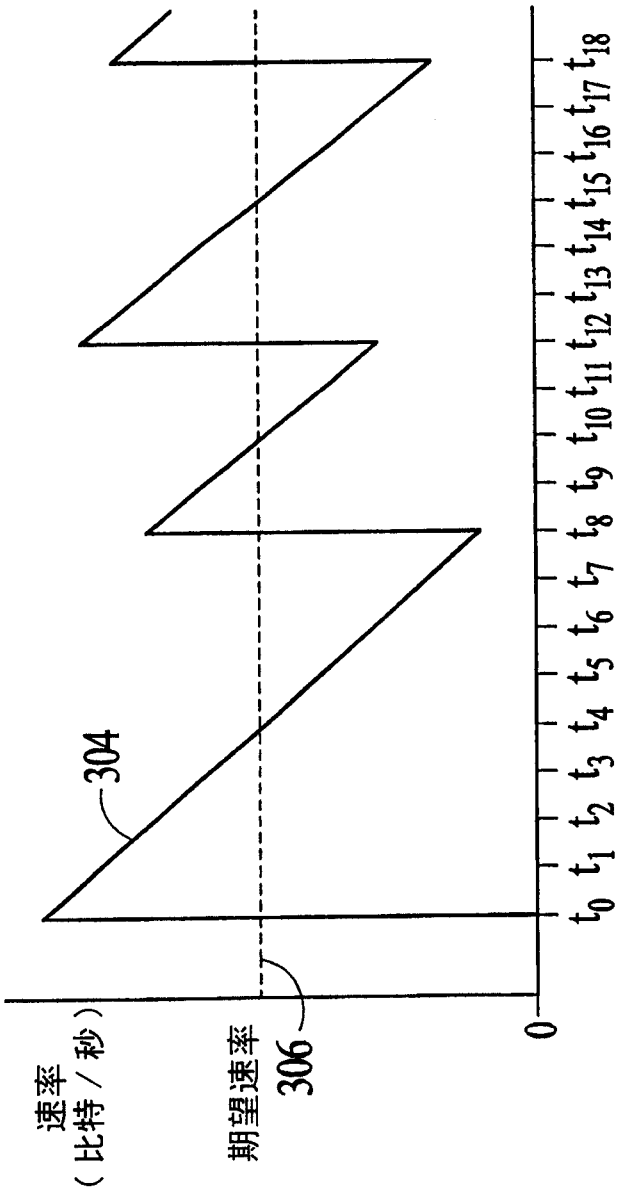

速率
（比特／秒）
期望速率
306
304
0
t_0 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10} t_{11} t_{12} t_{13} t_{14} t_{15} t_{16} t_{17} t_{18}

图 3A

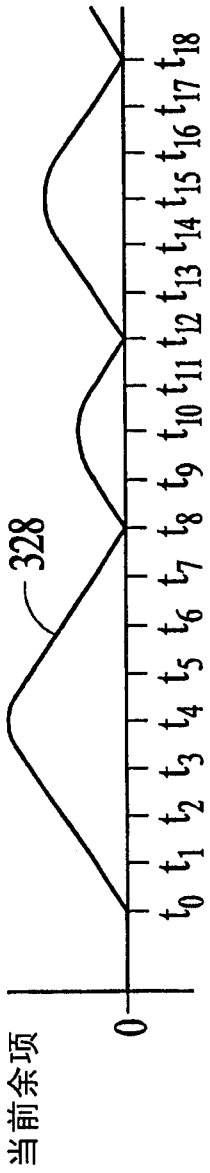

当前余项
0
328
t_0 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10} t_{11} t_{12} t_{13} t_{14} t_{15} t_{16} t_{17} t_{18}

图 3B

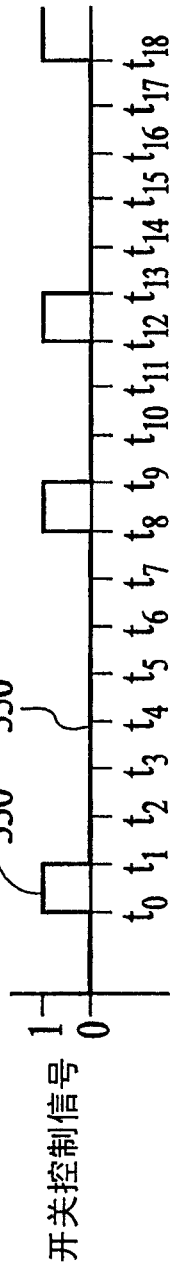

开关控制信号
1
0
330
330
t_0 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10} t_{11} t_{12} t_{13} t_{14} t_{15} t_{16} t_{17} t_{18}

图 3C

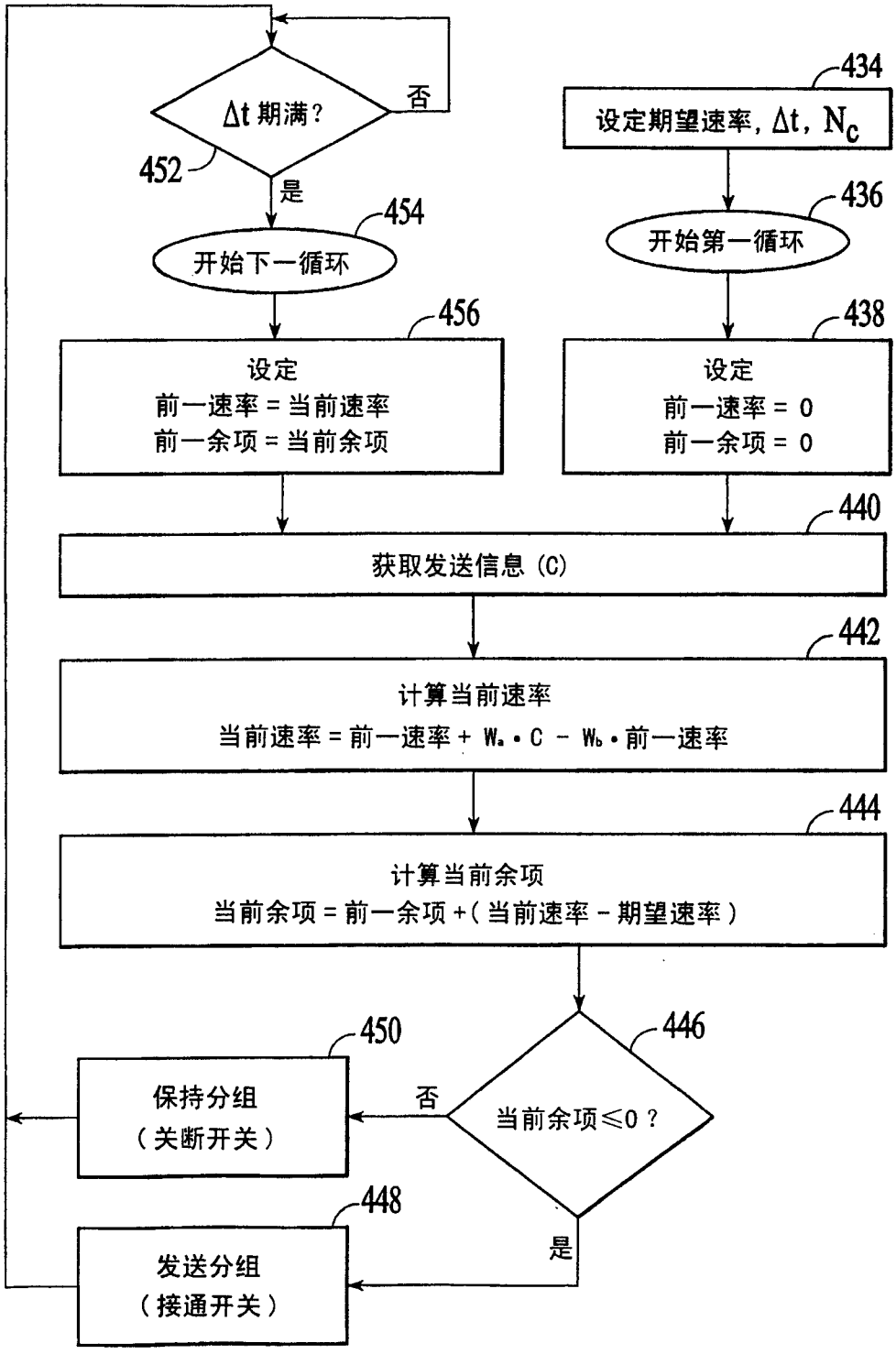

Δt 期满?
否
452
是
454
开始下一循环
456
设定
前一速率 = 当前速率
前一余项 = 当前余项
434
设定期望速率, Δt, N_c
436
开始第一循环
438
设定
前一速率 = 0
前一余项 = 0
440
获取发送信息 (C)
442
计算当前速率
当前速率 = 前一速率 + W_a • C - W_b • 前一速率
444
计算当前余项
当前余项 = 前一余项 +(当前速率 - 期望速率)
446
当前余项≤0 ?
否
450
保持分组
(关断开关)
是
448
发送分组
(接通开关)

图 4

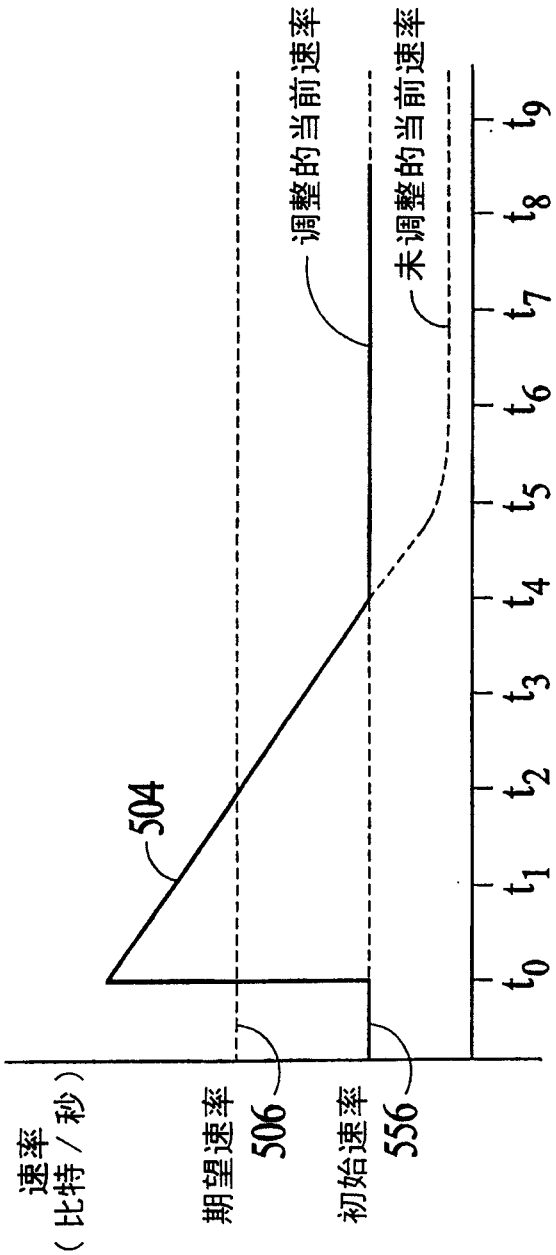

速率
（比特／秒）
期望速率
506
初始速率
556
504
调整的当前速率
未调整的当前速率
t_0 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9

图 5A

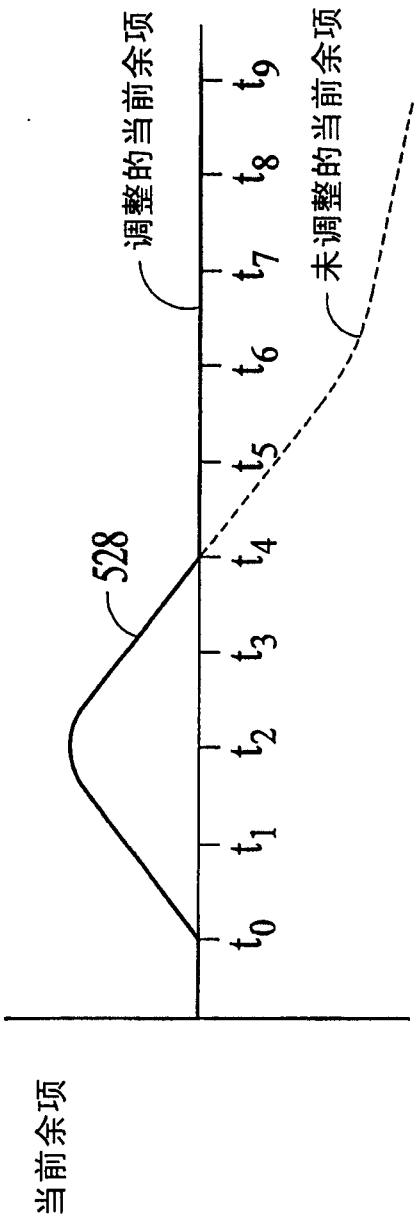


图 5B

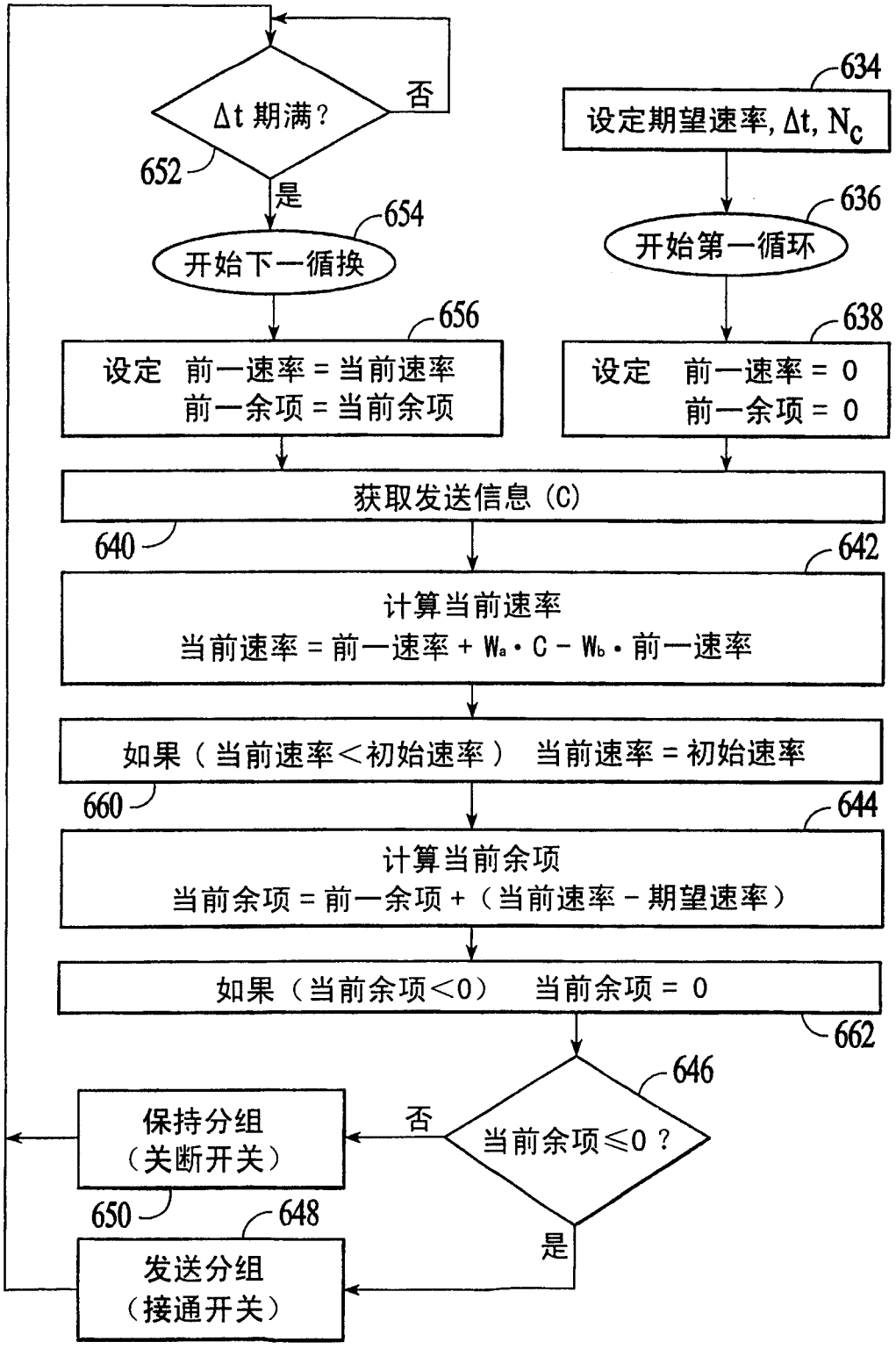

Δt 期满？
否
652
是
654
开始下一循换
656
设定 前一速率 = 当前速率
前一余项 = 当前余项
634
设定期望速率, Δt, N_c
636
开始第一循环
638
设定 前一速率 = 0
前一余项 = 0
获取发送信息 (C)
640
642
计算当前速率
当前速率 = 前一速率 + W_a • C − W_b • 前一速率
如果（当前速率＜初始速率） 当前速率 = 初始速率
660
644
计算当前余项
当前余项 = 前一余项 +（当前速率 − 期望速率）
如果（当前余项＜0） 当前余项 = 0
662
646
当前余项≤0？
否
是
保持分组
（关断开关）
650
648
发送分组
（接通开关）

图 6

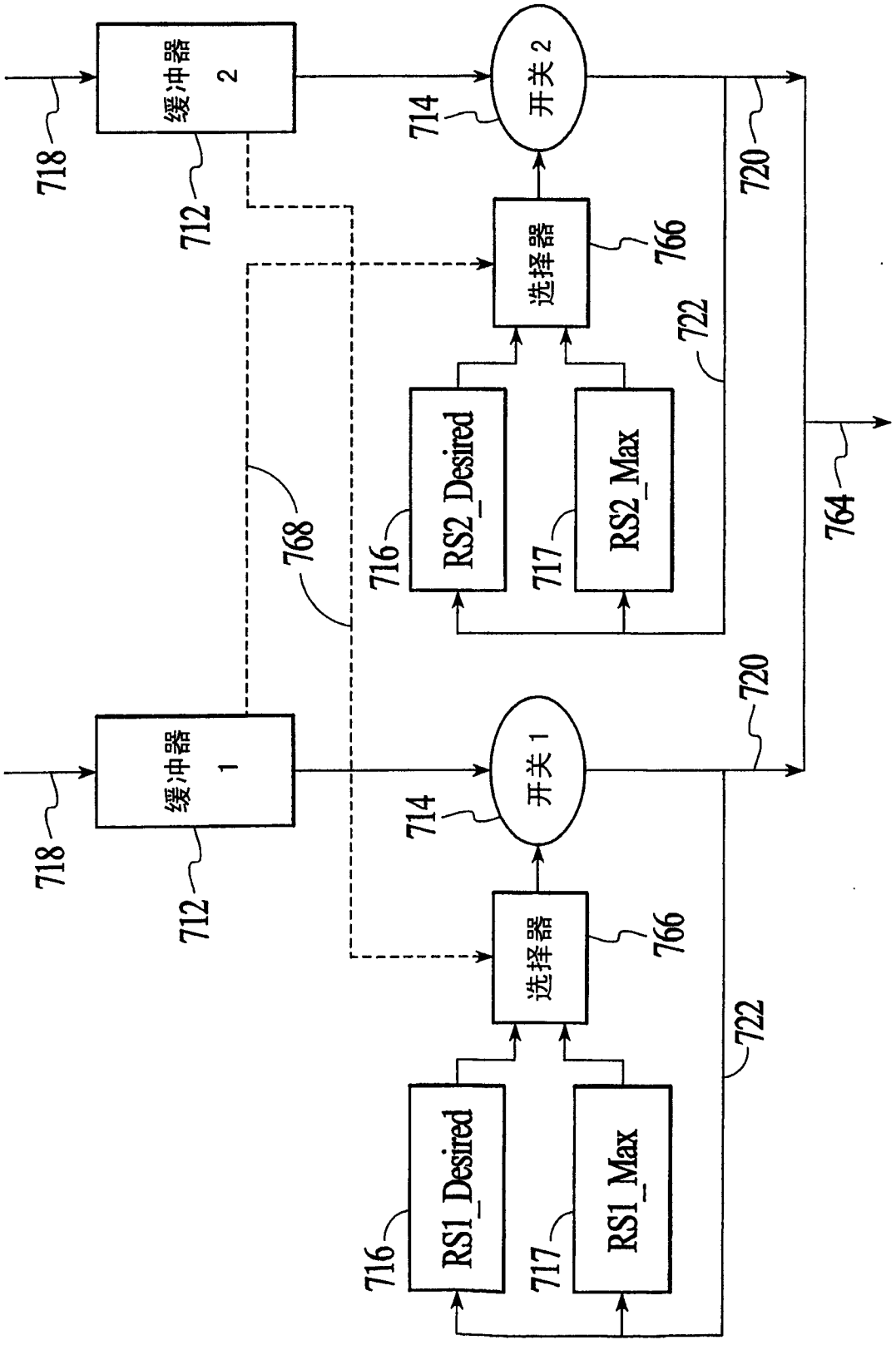

缓冲器 2
缓冲器 1
开关 2
开关 1
选择器
选择器
RS2_Desired
RS2_Max
RS1_Desired
RS1_Max
712
714
716
717
718
720
722
764
766
768

图 7

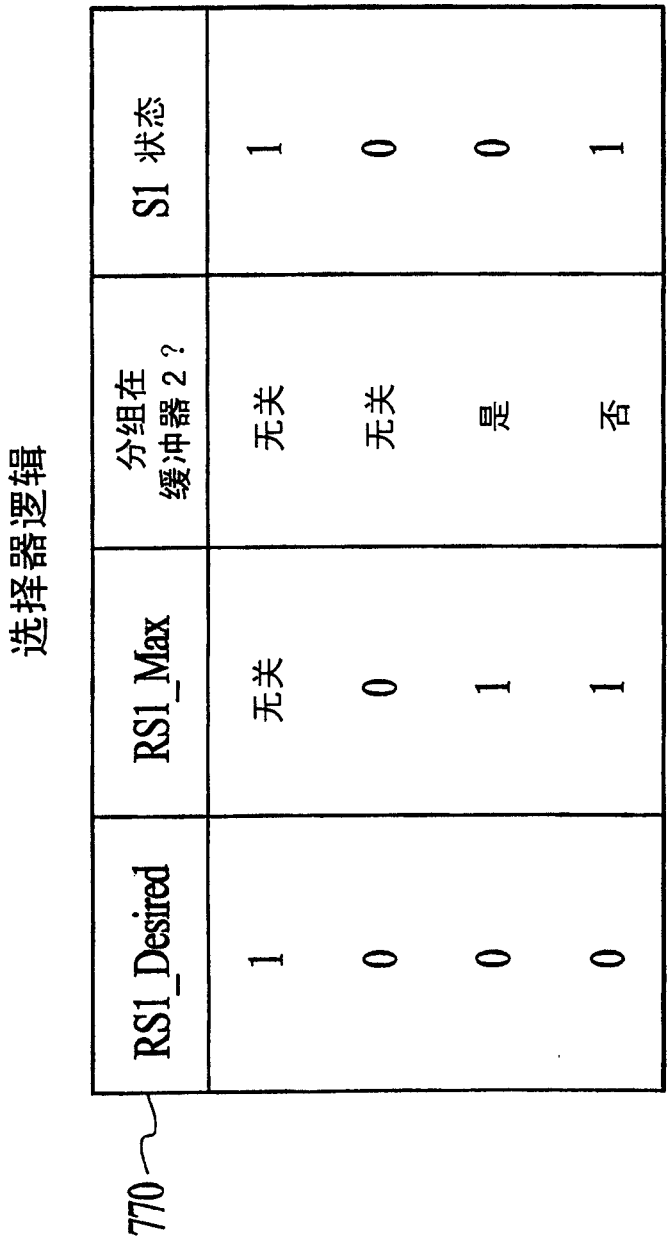

选择器逻辑
770
RS1_Desired
RS1_Max
分组在缓冲器2？
S1 状态
1
无关
无关
1
0
0
无关
0
0
1
是
0
0
1
否
1

图 8

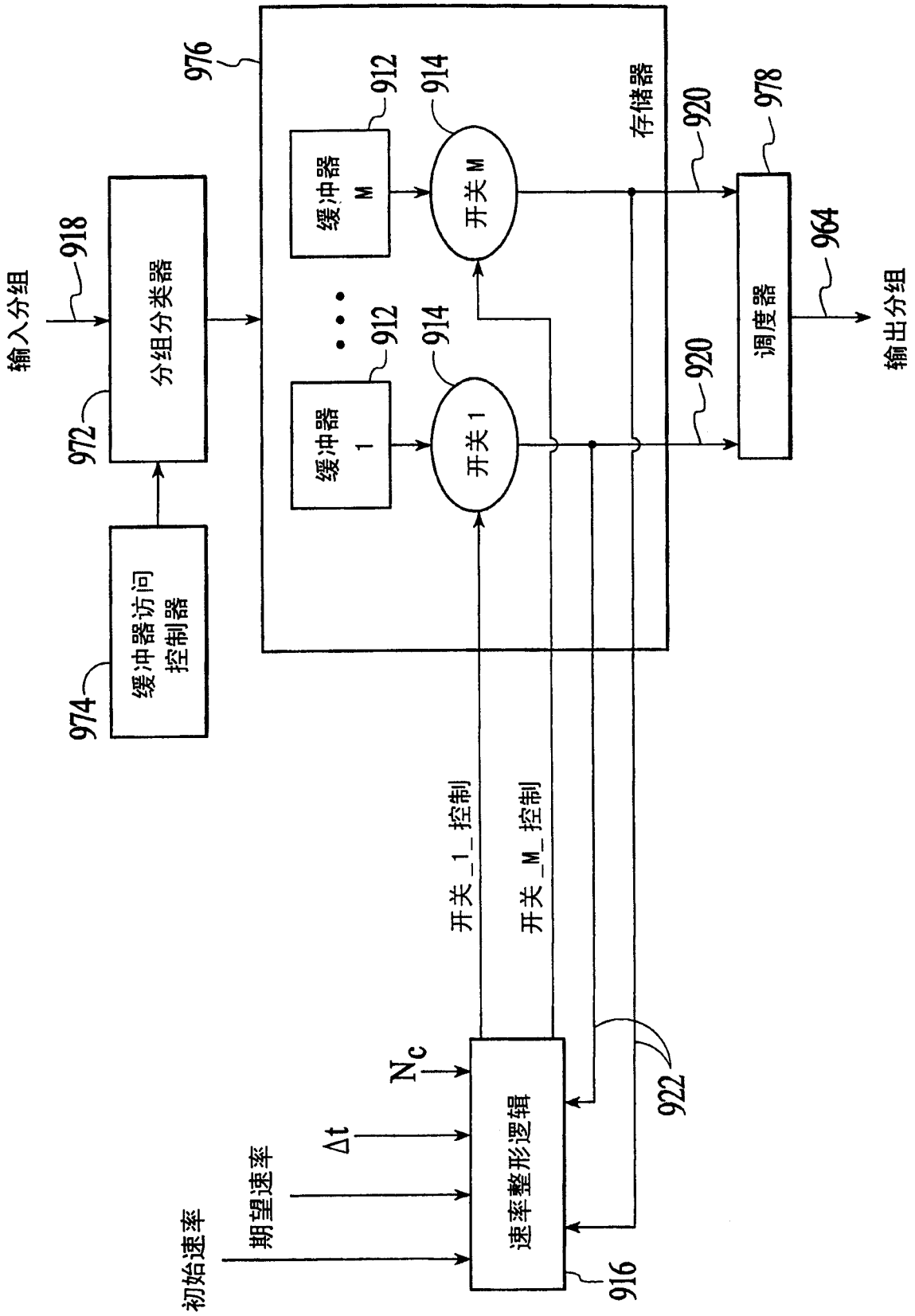

输入分组
918
分组分类器
976
缓冲器
M
912
开关 M
914
存储器
920
978
缓冲器
1
912
开关 1
914
920
调度器
964
输出分组
972
缓冲器访问
控制器
974
开关_1_控制
开关_M_控制
Nc
Δt
期望速率
初始速率
速率整形逻辑
922
916

图 9