[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
H04L 12/56 (2006.01)

[12] 发明专利说明书

专利号 ZL 200480010998.8

[45] 授权公告日 2010 年 1 月 6 日

[11] 授权公告号 CN 100579060C

[22] 申请日 2004.3.17

[21] 申请号 200480010998.8

[30] 优先权

[32] 2003. 3.17 [33] US [31] 60/455,906

[32] 2003. 4.28 [33] US [31] 10/425,854

[86] 国际申请 PCT/US2004/008283 2004.3.17

[87] 国际公布 WO2004/084509 英 2004.9.30

[85] 进入国家阶段日期 2005.10.24

[73] 专利权人 高通股份有限公司

地址 美国加利福尼亚州

[72] 发明人 M·塔纳佳 R·潘卡

[56] 参考文献

CN1340279A 2002.3.13

WO02/07388A2 2002.1.24

WO01/63849A2 2001.8.30

WO02/056564A1 2002.7.18

US2002/0061007A1 2002.5.23

审查员 冯 楠

[74] 专利代理机构 上海专利商标事务所有限公司

代理人 钱慰民

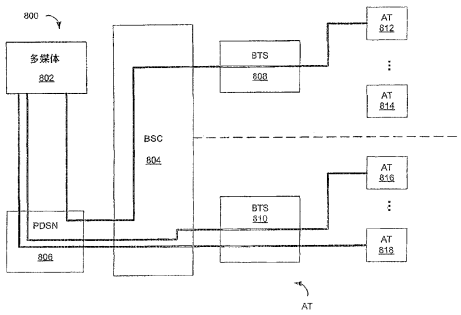
权利要求书 2 页 说明书 41 页 附图 27 页

[54] 发明名称

支持 IP 应用程序的通信系统中控制允许进入的方法和设备

[57] 摘要

一种在通信系统中进行允许进入控制的方法和设备。 接入网络(AN)元件确定可用的资源。 当可用资源足以支持一个请求应用程序流程的要求时，AN 允许该应用程序流程进入。 AN 周期性地，并在触发事件发生时更新可用资源的度量。 允许进入控制可与应用程序对应于每条流程类型的补偿因子和给定用户的总流程的补偿因子的调度器协同操作。



1．一种在支持因特网协议 IP 的应用程序的通信系统中进行允许进入控制的方法，所述通信系统包括一接入网络 AN 和多个接入终端 AT，所述 AT 中的每一个都向所述 AN 发送一请求数据率，所述通信系统支持对所述 AT 具有 QoS 要求的应用程序流程，所述方法包括：

确定所述通信系统中的可用资源；

接收对具有第一通信量概况和第一 QoS 概况的第一应用程序流程的请求；

确定所述可用资源是否支持对所述第一应用程序流程的请求；

如果所述第一应用程序流程有大于平均请求数据率的对应数据率，则拒绝所述第一应用程序流程；以及

如果所述对应数据率不大于所述平均请求数据率，且如果所述可用资源支持对所述第一应用程序流程的请求，则允许所述第一应用程序流程进入其中，确定所述通信系统中的可用资源包括：

确定所述通信系统中的保留资源；

将所述保留资源与所述通信系统的容量相比较；以及

将所述可用资源确定为所述容量和所述保留资源之差。

2．如权利要求 1 所述的方法，其特征在于，还包括：

确定所述第一应用程序流程的自适应预订因子。

3．如权利要求 2 所述的方法，其特征在于，还包括：

基于所述第一应用程序流程的服务质量 QoS 统计量对所述自适应预订因子进行更新。

4．如权利要求 3 所述的方法，其特征在于，还包括：

确定所述第一应用程序流程的 QoS 统计量；

将所述第一应用程序流程的 QoS 统计量与一扇区中的其它当前流程相比较；以及

响应于所述第一应用程序流程的 QoS 统计量与所述扇区中其它当前流程的比较，更新所述可用资源和保留资源。

5．如权利要求 4 所述的方法，其特征在于，还包括：

确定所述第一应用程序流程的用户是否出现在所述通信系统的一个扇区中；

以及

确定所述第一应用程序流程的采样持续时间，

其中，确定所述第一应用程序流程的 QoS 统计量包括：

确定在所述采样持续时间期间所述第一应用程序流程的 QoS 统计量。

6．如权利要求 5 所述的方法，其特征在于，第一采样持续时间与速率违反相关联，第二采样持续时间与延迟违反相关联。

7．如权利要求 5 所述的方法，其特征在于，将所述第一应用程序流程的 QoS 统计量与所述扇区中的其它当前流程相比较包括：

计算在一采样周期期间由具有 QoS 要求的流程所使用的第一时隙分数；

计算在所述采样周期期间由所述第一应用程序流程所使用的第二时隙分数；

计算对应于所述第一应用程序流程的 QoS 统计量的 QoS 流程的第三分数；

将所述第三分数与一对应阈值相比较；以及

响应于将所述第三分数与对应阈值相比较，对自适应预订进行估值。

8．一种用于在支持因特网协议 IP 应用程序的通信系统中进行允许进入控制的设备，所述通信系统包括一接入网络 AN 和多个接入终端 AT，所述 AT 中的每一个都向所述 AN 发送一请求数据率，所述通信系统支持对所述 AT 具有 QoS 要求的应用程序流程，所述设备包括：

用于确定所述通信系统中的可用资源的装置；

用于接收对具有第一通信量概况和第一 QoS 概况的第一应用程序流程的请求的装置；

用于确定所述可用资源是否支持对所述第一应用程序流程的请求的装置；

用于在所述第一应用程序流程含有比所述平均请求数据率大的对应数据率的情况下拒绝所述第一应用程序流程的装置；

用于在所述对应数据率不大于所述平均请求数据率、并且所述可用资源支持对所述第一应用程序流程的请求的情况下允许所述第一应用程序流程进入的装置，

其中用于确定所述通信系统中的可用资源的装置包括：

用于确定所述通信系统中的保留资源的装置；

用于将所述保留资源与所述通信系统的容量相比较的装置；以及

用于将所述可用资源确定为所述容量和所述保留资源之差的装置。

支持IP应用程序的通信系统中控制允许进入的方法和设备

根据35 U.S.C.§119的优先权要求

本专利申请要求转让给本发明受让人的于2003年3月17日提交的题为“System for Allocating Resources in a Communication System”的临时专利申请第60/455,906号的优先权，并因此这儿并入作参考。

背景

1．发明领域：

本发明涉及通信系统。特别地，这些实施例是针对在一个通信系统的多个用户之间分配通信资源。

2．有关技术：

已经给出了若干方案来解决在多个用户之间对通信系统中的单个节点所提供的有限通信资源进行分配的问题。此类系统的一个目标是在各个节点处提供足够的资源，以满足所有用户的需求并同时将成本最小化。由此，通常是以在各用户之间有效分配资源的目标来设计此类系统。

各种系统已经实现频分多址（FDMA）方案，该方案将资源并发地分配给每一个用户。此类系统中的通信节点通常在任何时间点向网络中的每个用户发送信息或从每个用户接收信息的带宽都是有限的。此方案通常涉及将总带宽的不同部分分配给单个用户。尽管对于用户要求与通信节点有不受干扰的通信的系统来说，这一方案可能是有效的，但是当此类持续的、不受干扰的通信并非必须时，对总带宽更好的利用是可以实现的。

其它在多个用户之间分配单个通信节点的通信资源的方案包括时分多址（TDMA）方案。在用户不要求与单个通信节点有持续的、不受干扰的通信的情况下，这些TDMA方案于在多个用户之间分配单个通信节点的有限带宽资源时特别有效。TDMA方案通常按指定的时间间隔，使单个通信节点的全部带宽专供每一个用户使用。在使用码分多址（CDMA）方案的无线通信系统中，这可通过在时分

复用基础上，按指定的时间间隔将所有代码信道分配给每一个用户来实现。通信节点实现与用户相关联的唯一的载波频率或通道代码以允许与该用户进行专有通信。TDMA 方案还可使用物理接触中继交换或分组交换在陆线系统中实现。

TDMA 系统通常以循环方式将相等的时间间隔分配给每个用户。这可能导致某些用户对某些时间间隔的不充分利用。类似地，其它用户可能有超过所分配时间间隔的对通信资源的需求，使得这些用户不能得到满足。系统操作员可选择负担增加节点带宽的成本以确保没有用户得不到满足，或者允许得不到满足的用户继续保持得不到满足的状态。

由此，需要提供一种根据一种在用户之间分配通信资源的网络策略，在通信网络的用户之间有效和公平地分配通信资源的系统和方法。与之一致的是，需要将系统所服务的用户数量最大化，包括，但不限于，提供各种响应于系统的具体要求、限制、和/或目标，在每条流程的基础上和/或在合计的基础上进行资源分配的机制。此外还需要将资源分配最优化的允许进入控制和先占方法。

附图简述

图 1 示出根据本发明的一个实施例的一种通信网络。

图 2A 示出根据本发明的一个实施例配置的基站控制器和基站设备的框图。

图 2B 示出根据本发明的一个实施例配置的远程站设备的框图。

图 3 示出在图 2A 所示的信道调度器的实施例中执行一种调度算法的流程图。

图 4 是一种支持多媒体应用程序的通信系统，其中每个应用程序通信由一个应用程序流程表示。

图 5 是一个应用程序流程队列。

图 6 是一张时序图，描述一个应用程序流程的一部分的信号定时。

图 7A 是一张时序图，描述对一个应用程序流程的抖动测量。

图 7B 是一张时序图，描述为处理一个应用程序流程，在若干时隙期间发送连续的 IP 分组。

图 8 是一张流程图，描述在一种通信系统中对若干应用程序流程进行调度。

图 9 是一张流程图，描述对若干具有不同服务质量（QoS）要求的应用程序流程进行调度。

图 10 是一张根据一个实施例的体系结构图，示出应用程序流程与一种调度算法一致的每个应用程序流程的定义。

图 11 是根据一个实施例识别分类的类型的表。

图 12A 描述根据一个实施例的一种调度算法的一部分，包括对应用程序流程的初始化。

图 12B 描述根据一个实施例的一种调度算法的一部分，包括根据类的类型对应用程序流程的处理。

图 12C 描述根据一个实施例的一种调度算法的一部分，包括对模式 I 的应用程序流程的处理、对模式 II 的应用程序流程的处理和对模式 III 的应用程序流程的处理。

图 12D 描述根据一个实施例的一种调度算法的一部分，包括对模式 I 的应用程序流程的处理。

图 12E 描述根据一个实施例的一种调度算法的一部分，包括自适应加权和以其为基础的调度。

图 13 示出一种基站收发器系统（BTS），用于在无线通信系统中实现一种用自适应加权算法来调度应用程序流程的算法。

图 14 是一张时序图，测绘诸如数据率（L_{MAX}）、保留资源（Res(t)）、和可用资源（Avail(t)）等最大资源作为时间的函数。

图 15 是一张时序图，测绘从高速率分组数据类型系统中的用户接收到的数据请求、和在时间 t 要保留的估算能力 L(t)作为时间的函数。

图 16 是一张信息流程图，描述用于支持多个具有服务质量（QoS）要求的应用程序流程的高速率分组数据类型系统的调度器，其中这些流程是应用每条流程的补偿来调度的。

图 17 是一张信息流程图，描述用于支持多个具有服务质量（QoS）要求的应用程序流程的高速率分组数据类型系统的调度器，其中这些流程是应用合计补偿来调度的。

图 18A 到 18E 示出一种在支持多个具有服务质量（QoS）要求的应用程序流程的高速率分组数据类型系统中进行允许进入控制的算法。

图 19 示出一种在支持多个具有服务质量（QoS）要求的应用程序流程的高速率分组数据类型系统中进行先占的算法。

图 20 是支持多个具有服务质量（QoS）要求的应用程序流程的高速率分组数据类型系统中的一种接入网络（AN）的框图。

具体描述

本发明的各个实施例针对于一种在通信网络中由单个通信节点服务的多个用户之间分配资源的系统和设备。在单独的离散的传输间隔，或称“服务间隔”，单个用户排斥所有其它用户地占用通信节点的有限资源。基于与单个用户相关联的权值或得分，单个用户被选择以占用该有限资源。较佳的是，对与单个用户相关联的权值的改变是基于该单个用户能够消耗该有限资源的瞬时速率。

参考附图，图 1 表示一种示例性的可变速率的通信系统。一个此类系统在转让给 Qualcomm 有限公司的于 1997 年 11 月 3 日提交的题为“Method and Apparatus for High Rate Packet Data Transmission”（高速率分组数据传输的方法和设备）的美国专利申请第 08/963,386 号（现为 2003 年 6 月 3 日公告的美国专利第 6574211 号）中描述，且这儿并入作为参考。可变速率通信系统包括多个单元 2A－2G。每个单元 2 由一个对应的基站 4 服务。各个远程站分布遍及该通信系统。在该示例性实施例中，每个远程站 6 都以任意的数据传输间隔在前向链路上与至多一个基站 4 通信。例如，在时隙 n 在前向链路上，基站 4A 排他地向远程站 6A 传输数据，基站 4B 排他地向远程站 6B 传输数据，而基站 4C 排他地向远程站 6C 发送数据。如图 1 所示，较佳的是每个基站 4 在任意给定时刻向一个远程站 6 传输数据。在其它实施例中，基站 4 可在特定数据传输间隔与一个以上远程站 6 通信，同时排斥与基站 4 相关联的所有其它远程站 6。此外，数据率是可变的，在一个实施例中，数据率是取决于接收远程站 6 所测得的载波干扰比（C/I），和要求的每比特能量噪声比（E_b/N_0）。为简化起见，图 1 中未示出从远程站 6 到基站 4 的后向链路。根据一个实施例，远程站 6 是具有由无线数据服务用户操作的无线收发器的移动单元。

示出一种示例性可变速率通信系统的基本子系统的框图在图 2A－2B 中示出。基站控制器 10 与分组网络接口 24、公共交换电话网络（PSTN）30、和通信系统中的所有基站 4（为简化起见，图 2 中仅示出一个基站 4）相接。基站控制器 10 协调通信网络中的远程站 6 与连接到分组网络接口 24 和 PSTN 30 的其它用户之间的通信。PSTN 30 通过标准电话网络（图 2 中未示出）与用户相接。

基站控制器 10 包含许多选择器元件 14，尽管为简化起见，图 2A 中仅示出一个。每个选择器元件 14 都被指派以控制一个或多个基站 4 和一个远程站 6 之间的通信。如果选择器元件 14 未被指派给远程站 6，则通知呼叫控制处理器 16 有寻呼远程站 6 的需要。呼叫控制处理器 16 随即指派基站 4 来寻呼远程站 6。

数据源 20 包含要传输到远程站 6 的大量数据。数据源 20 将数据提供给分组

网络接口 24。分组网络接口 24 接收数据，并将数据发送到选择器元件 14。选择器元件 14 将数据传输到每个与远程站 6 通信的基站 4。在该示例性实施例中，每个基站 4 都维护一个数据队列 40，它存储要传输到远程站 6 的数据。

数据是以数据分组的形式从数据队列 40 传输到信道元件 42。在该示例性实施例中，在前向链路上，“数据分组”指一定量的数据，它最多是 1024 位，并且要在一个“时隙”（诸如≈1.667 毫秒）之内传输到目标远程站 6。对于每个数据分组，信道元件 42 插入必需的控制字段。在该示例性实施例中，信道元件 42 将数据分组和控制字段进行 CRC 编码，并插入一组代码尾位。数据分组、控制字段、CRC 奇偶校验位、和代码尾位组成格式化的分组。在该示例性实施例中，信道元件 42 随即对已格式化的分组进行编码，并将已编码的分组内的符号交错（或重新排序）。在该示例性实施例中，用 Walsh 代码覆盖交错处理分组、并用短 PNI 和 PNQ 代码对经交错处理的分组进行扩展。已扩展的数据被提供给 RF 单元 44，它对信号进行正交调制、过滤和放大。前向链路信号是通过前向链路 50 上的天线 46 在空气中传输。

在远程站 6，前向链路信号由天线 60 接收并发送到前端 62 内部的接收机。接收机对信号进行过滤、放大、正交解调和量化。数字化的信号被提供给解调器（DEMOD）64，在该处用短 PNI 和 PNQ 代码去扩展，并用 Walsh 代码去覆盖。已解调的数据被提供给解码器 66，它执行在基站 4 处所作的信号处理功能的逆转功能，具体而言是去交错、解码、和 CRC 校验功能。已解码的数据被提供给数据接收机 68。

如以上所指出的硬件支持在前向链路上对数据、消息通信、语音、视频和其它通信进行可变速率的传输。从数据队列 40 传输的数据的速率变动以适应远程站 6 处的信号强度和噪音环境的变化。较佳的是，每个远程站 6 在每个时隙都向相关联的基站 4 传输一个数据率控制（DRC）信号。DRC 指一种控制机制，远程站通过这种机制为前向链路确定合乎需要的数据率，即，在远程站处接收数据的数据率。远程站经由 DRC 消息，向基站发送如同数据率请求或指令的期望数据率。DRC 信号向基站 4 提供信息，包括远程站 6 的身份，和该远程站要从其相关联的数据队列接收数据的速率。由此，远程站 6 处的电路测量信号强度并估计远程站 6 处的噪音环境，以确定要在 DRC 信号中传输的速率信息。

每个远程站 6 所传输的 DRC 信号沿着后向链路信道 52 行进，并在基站 4 处通过天线 46 和 RF 单元 44 被接收。在该示例性实施例中，DRC 信息在信道元件

42 中被解调，并被提供给位于基站控制器 10 中的信道调度器 12A，或提供给位于基站 4 中的信道调度器 12B。在第一示例性实施例中，信道调度器 12B 位于基站 4 中。在替换实施例中，信道调度器 12A 位于基站控制器 10 中，并连接到该基站控制器 10 内部的所有选择器元件 14。

在一个实施例中，信道调度器 12B 从数据队列 40 接收指示为每个远程站所排队的数据量（亦称队列大小）的信息。信道调度器 12B 随即基于 DRC 信息和基站 4 所服务的每个远程站的队列大小进行调度。如果在替换实施例中使用的调度算法请求队列大小，则信道调度器 12A 可从选择器元件 14 接收队列大小的信息。

本发明的各个实施例适用于可能支持可变速率传输的其它硬件体系结构。可以很容易地扩展本发明以覆盖后向链路上的可变速率传输。例如，基站 4 不是基于来自远程站 6 的 DRC 信号来确定基站 4 处接收数据的速率，而是测量从远程站 6 所接收的信号的强度并估计噪音环境以确定从远程站 6 接收数据的速率。基站 4 随即向每个相关联的远程站 6 传输要从远程站 6 出发在后向链路中传输的数据的速率。基站 4 随即可以类似本文中就前向链路所描述的方式，基于后向链路上不同的数据率来对后向链路上的传输进行调度。

同样，以上所讨论的实施例的基站 4 用 CDMA 方案，向所选择的一个或数个远程站进行传输，同时排斥与该基站 4 相关联的其余远程站。在任何特定时间，基站 4 通过使用一种分配给该接收基站 4 的代码，对所选择的一个或数个远程站 6 进行传输。但是，本发明还可适用于其它系统，这些系统使用不同 TDMA 方法提供数据来选择若干基站 4 并同时排斥其它基站 4，以对传输资源进行最优化的分配。

信道调度器 12 对前向链路上可变速率的传输进行调度。信道调度器 12 接收指示要向远程站 6 传输的数据量的队列大小，和来自远程站 6 的消息。较佳的是，信道调度器 12 对数据传输进行调度，以实现将数据吞吐量最大化并同时遵守公平性约束的系统目标。

如图 1 中所示，远程站 6 分布遍及通信系统，并可在前向链路上与零个或一个基站 4 通信。在该示例性实施例中，信道调度器 12 协调整个通信系统上的前向链路数据传输。一种用于高速数据传输的调度方法和设备在 2002 年 1 月 1 日授权的美国专利第 6,335,922 号中具体描述，该专利已转让给本发明的受让人，并这儿并入作为参考。

根据一个实施例，信道调度器 12 在计算机系统中实现，该系统包括处理器、随机存取存储器（RAM）、和用于存储要由处理器执行的指令的程序存储器（未

图示）。处理器、RAM 和程序存储器可由信道调度器 12 的各个功能占用。在其它实施例中，处理器、RAM 和程序存储器可为用于执行基站控制器 10 处的其它功能的共享计算资源的一部分。

图 3 示出一种调度算法的实施例，该算法控制信道调度器 12 对从基站 4 到远程站 6 的传输进行调度。如以上所讨论的，一个数据队列 40 与每个远程站 6 相关联。信道调度器 12 使每个数据队列 40 与在步骤 110 处估值的“权值”相关联，用于选择与基站 4 相关联的特定远程站 6，从在随后的服务间隔内接收数据。信道调度器 12 选择单个的远程站 6，以在离散的服务间隔内接收数据传输。在步骤 102，信道调度器为每个与基站 4 相关联的队列初始化权值。

信道调度器 12 按传输间隔或服务间隔从步骤 104 到 112 进行循环。在步骤 104，信道调度器 12 判定是否由于在前一个服务间隔中检测到的另一个远程站 6 与基站 4 之间的相关联，而有任何其它队列要添加。信道调度器 12 在步骤 104 还对与新队列相关联的权值进行初始化。如以上所讨论的，基站 4 按诸如时隙等规律的间隔，从与其相关联的每个远程站 6 接收 DRC 信号。

此 DRC 信号还提供信息供信道调度器在步骤 106 使用，以为与每个队列相关联的远程站确定消耗信息（或接收所传输数据）的瞬时速率。根据一个实施例，从任何远程站 6 传输的 DRC 信号指示该远程站 6 能够以多个有效数据率中的任何一个接收数据。

基于远程站 6 的相关联接收数据的瞬时速率（如在最近接收到的 DRC 信号中所指示的），信道调度器 12 在步骤 108 确定服务间隔的长度，在该服务间隔期间，数据被发送到任何特定的远程站 6。根据一个实施例，在步骤 106，接收数据的瞬时速率 R_i 决定与特定数据队列相关联的服务间隔长度 L_i。

信道调度器 12 在步骤 110 选择特定数据队列进行传输。所要发送的相关联数据量随即从数据队列 40 被检索，并被提供给信道元件 42，用于传输给与该数据队列 40 相关联的远程站 6。如以下所讨论的，信道调度器 12 在步骤 110 选择用于提供数据的队列，该队列在接下来的服务间隔中用包括与每个队列相关联的每个权值在内的信息来传输。与所传输的队列相关联的权值随即在步骤 112 得到更新。

本领域技术人员可以理解，信道调度器 12 可用各种方法来实现，而不会偏离本发明。例如，信道调度器 12 可用包括处理器、随机存取存储器（RAM）和用于存储要由处理器执行的指令的程序存储器（未图示）的计算机系统来实现。在其它实施例中，信道调度器 12 的各个功能可并入共享的计算资源，该共享的计算资源

还用于执行基站 4 或基站控制器 10 处的其它功能。此外，用于执行信道调度器功能的处理器可以是通用微处理器、数字信号处理器（DSP）、可编程逻辑设备、专用集成电路（ASIC）、或能够执行本文中所描述的各种算法的其它设备，而不会偏离本发明。

如图 1 的实施例中所示，远程站 6 是移动的，并且能够改变不同基站 4 之间的关联。例如，远程站 6F 起初从基站 4F 接收数据传输。远程站 6F 随后可能移出基站 4F 的单元并移入基站 4G 的单元。远程站 6F 随即可开始传输 DRC 信号来警告基站 4G，而不是基站 4F。通过不再从远程站 6F 收到 DRC 信号，基站 4F 处的逻辑推论远程站 6F 已经脱离且不准备再接收数据传输。与远程站 6F 相关联的数据队列随即可经由陆线或 RF 通信链路传输到基站 4G。

自适应加权调度算法

此外，当在无线通信系统中传输多媒体服务或其它具有各种传输要求的服务时存在一个问题，其中称为“流程”的多媒体服务传输引起突发性的通信量。突发性的通信量的特征由若干变量描述，包括突发性的量度、和平均数据率。此外，需要满足系统中各条流程中每一个的服务质量（QoS）要求。诸如比例公平（PF）算法等当前的调度方法一般基于按被请求的数据率（称为数据率控制数据请求或“DRC”）对吞吐量（记为“T”）的比率给出的度量，来选择要服务的流程。此类计算可能无法确保所有用户所需的 QoS。因此，纯粹的 PF 算法可能不能提供足够的复杂程度来满足访问多媒体或其它应用程序的用户的 QoS 要求。需要一种能够满足这些不同要求的调度器。

注意以下讨论考虑支持如 IS-856 中所描述的高速率分组数据（HRPD）服务的 cdma2000 系统。此系统是作为示例使用。本发明适用于能根据调度算法选择要服务的用户的其它系统。

在 HRPD 系统中，空中接口可支持多达 4 个并行的应用程序流。第一个流携带信令信息，其它三个流可用于携带具有不同服务质量（QoS）要求的应用程序或其它应用程序。

为清楚理解下文所呈现的一个实施例起见，提供以下词汇表。以下词汇表并不试图穷举。以下词汇表并不试图将本发明限制于此，而是为就一种支持自适应加权调度算法的通信系统的实施例的清楚和理解而提供的。

词汇表

接入网络（AN）－ 提供蜂窝网络和分组交换数据网络（通常指因特网）和 AT 两两之间的数据连通性的网络设备。HRPD 系统中的 AN 等效于蜂窝通信系统中的基站。

接入终端（AT）－ 提供到用户的数据连通性的设备。HRPD 系统中的 AT 对应于蜂窝通信系统中的移动站。AT 可连接到诸如膝上个人计算机等计算设备，或者它可以是诸如个人数字助理（PDA）等自含式数据设备。

应用程序流程 － 为给定应用程序流所指定的从源到 AT 的传输路径。每个应用程序流程由源、目的、通信量概况和服务质量概况来标识。

应用程序流 － 对应于一个应用程序的数据通信。大多数应用程序流都有指定的服务质量要求。

自动重复请求（ARQ）－ 发射机基于一事件的发生或者不发生而初始化数据重新传输的机制。

可用资源(t)：前向链路上在时间 t 无限制的带宽。

平均数据率（r）－ 给定应用程序流程一段时间的平均输入数据率。

平均延迟（AvgD）－ 从 AN 到 AT 的多个分组或比特上所承受的平均延迟。

突发性（σ）－ 对应用程序流程中的数据分组的突发性或密度与时间关系的度量。

数据率控制（DRC）－ AT 向 AN 传输所请求的数据率的机制。

亏数分组（defpkts）－ 在时隙 n 的开始处为流程 *k* 所定义。亏数分组是还未在流程中传输的分组，且 *defpkts* 具体定义为在 BTS 停留时间超过流程 *k* 的延迟阈值的相等大小的分组（例如，诸如媒体访问控制（MAC）分组等中间处理分组）的个数。

亏数位（defbits）－ 对应于亏数分组的位数。

延迟边界 － 从 AN 到 AT 传输一个数据分组所允许的指定时间。

延迟阈值 － 延迟边界或抖动边界的函数，用于计算 defpkts。

延迟补偿因子（Φ）－ 用于补偿延迟违反的补偿因子。

DRC 补偿因子（ß）－ 虑及与应用程序流程的用户相关联的数据请求要求的补偿因子。用于对应用程序进行恰当恢复。

增强抖动阈值（dv）－ 用于在检测到流程的两个相继 IP 分组之间的抖动违反时对增强抖动补偿函数进行计算。

流程权值（w）－ 适用于每个使用自适应加权调度算法的应用程序流程的初始权

值。自适应权值（aw）是权值的自适应值。

前向链路（FL）— 从 AN 到 AT 的传输空中链路。

排头（HOL）分组 — 队列中的第一个分组。

高速率分组数据（HRPD）— 以高数据率传输分组数据通信的数据服务。也称作高数据率（HDR），在题为“cdma2000 High Rate Packet Data Air Interface Specification”（cdma2000 高速率分组数据空中接口标准）的 IS-856 标准中详细说明。

抖动 — 所接收到的连续的分组两两之间的时间变化。

抖动边界（j）— 在给定应用程序流程的抖动上的边界。

增强抖动补偿因子（δ）— 为流程补偿抖动违反的补偿因子。

L_{max} — BTS 可在前向链路上传输数据的最大速率（例如，在 cdma2000 1xEV-DO 类型网络中为 2.4 兆比特/秒）。

L(t) — 基于先前的 QoS 违反的统计量和网络负荷有关的统计量对在时间 t 保留的前向链路容量的估计。

归一化亏数分组（ndefpkts）— 用亏数分组和该流程所请求的速率计算的归一化亏数分组。

归一化亏数位（ndefbits）— 对应于归一化亏数分组的归一化亏数位。

运动图像专家组（MPEG）— 传输多媒体材料的协议。

未决分组 - $pend_{k,j}[n]$ — 时隙 n 中 BTS 和 BSC 中流程 k 的 IP 分组 j 的未决字节数。

比例公平（PF）算法 — 根据按所请求的数据率对吞吐量的比率为每个 AT 计算的选择因子对数据通信进行调度的调度算法。

服务质量（QoS）— 涉及分组数据通信传输的要求，包括但不限于，延迟、请求速率、和抖动。

QoS 和网络补偿函数（Φ, γ, α, β, δ）— 如在自适应加权调度算法中使用的补偿函数。

服务质量组（QSG）— 具有相似 QoS 要求的应用程序类型组。

速率补偿因子（α）— 为补偿速率违反而计算的补偿因子。

服务速率（R）或所请求的速率（required_rate）— 流程所请求的速率。

Res(t)：前向链路上在时间 t 保留的带宽。

重新发送队列（Rx）— 存储为重新发送而调度的应用程序流程的重新发送队列。

后向链路（RL）— 从 AT 到 AN 的传输空中链路。

选择度量（Y）－为调度决定而比较应用程序流程所用的度量。

交通量概况（σ，r）－ 涉及突发性和数据率的度量。

传输队列（Tx）－ 为给定 BTS 存储应用程序流程的传输队列。

等待时间参数（γ）－ AN 内部排头 IP 分组的等待时间的度量。

将自适应权值应用于比例公平调度算法

为 cdma2000 1xEV-DO 网络的前向链路描述了比例公平（PF）调度算法，它基于度量 DRC/T 选择要服务的流程。PF 算法被设计成向每个用户提供大约相同个数的传输时隙。为加强这一调度算法，本文中所描述的是一种自适应加权 DRC/T 算法，它扩展并优化 DRC/T 算法以满足不同类型的应用程序的不同 QoS 要求。每个多媒体应用程序分别有各自具体的 QoS 要求。调度算法的目标包括满足各种 QoS 要求。本文中所给出的自适应算法（也称作自适应 w*DRC/T 算法）为其中应用程序流程包括多媒体应用程序服务的 cdma2000 1xEV-DO 网络的前向链路，提供各种优于 DRC/T 算法的性能优势。使用自适应算法，满足 cdma2000 1xEV-DO 网络前向链路上延迟和抖动敏感应用程序的延迟和抖动边界要求。此外，自适应调度算法确保满足速率要求和减少多媒体应用程序的平均延迟。尽管提供多媒体应用程序作为示例来说明自适应调度算法的实现，但是本文中所描述的方法和设备可适用于其它具有 QoS 要求或与其相关联的可量化要求的应用程序。

对于诸如网络浏览和游戏等具有速率和等待时间要求的应用程序，自适应调度算法提供速率保证并减少平均延迟。对于其它仅具有速率要求的应用程序，可使用自适应加权调度算法来满足速率保证。提供这些 QoS 保证的同时，自适应加权调度算法还作用于将总吞吐量维持在合理高的等级，并实现接近于当使用纯粹 PF 调度算法时所达到的总吞吐量。纯粹 PF 调度算法指使用 DRC/T 计算的算法。在给予有 QoS 违反的流程额外资源的同时，自适应加权调度算法以公平方式分配可用资源。本文中提供了各种与其一致的补偿机制。

图 4 示出支持多媒体应用程序的系统 800。再次注意，本发明适用于具有 QoS 要求的其它系统。系统 800 包括耦合到分组数据服务节点（PDSN）806 的多媒体源 802。PDSN 806 还耦合到基站控制器（BSC）804，基站控制器 804 可包括多个 BSC。BSC 804 经由基站收发器系统（BTS）808，810 与各个 AT 812、814、816、818 等通信。系统 800 可包括比图示更多的 BTS 和 AT。图示出三条流程：第一条流程从多媒体源 802 经由 PDSN 806、BSC 804、和 BTS 808、到 AT 812；第二条

流程从多媒体源 802 经由 PDSN 806、BSC 804、和 BTS 810、到 AT 816；第三条流程从多媒体源 802 经由 PDSN 806、BSC 804、和 BTS 810、到 AT 818。注意一个 AT 可以是多条流程的目的。在一个示例中，运动图像专家组（MPEG）类型应用程序的传输将音频和视频分成单独的流程。

系统 800 中要传输的每个应用程序流程具有：相关联的源地址；目的地址；和 QoS 要求。随即就从源到目的的传输对应用程序流程进行调度。应用程序流程穿过类似于图 4 中所示的路径。

每个 BTS 808、810 都适用于维护如图 5 中所示的流程队列。注意，每个 BTS 维护对应于其前向链路（FL）上的每个应用程序流程的一组队列。一个应用程序流程针对于一个 AT。但是，注意多条流程可针对于一个 AT。每条流程都具有一个与之相关联的服务质量组（QSG）类型。每个 QSG 由一组 QoS 参数定义。某一给定 QSG 的每条流程都有具体的值对应于组中每个参数。例如，一个 QSG 可由包括延迟和抖动的组来定义。具有这一 QSG 的哪些流程将指定对于延迟和抖动的要求。对于队列中的每条流程，BTS 维护包括以下三个单独队列的组：(1)原始传输队列（Tx）；(2)重新传输队列（Rx）；和(3)自动重复请求队列（ARQ）。在一个实施例中，ARQ 队列可对应于诸如先前决策 ARQ 等为在 BTS 和 AT 之间所执行的任何类型的重复机制存储流程的队列。多媒体应用程序可包括诸如视频会议等具有延迟边界要求的对延迟敏感的应用程序。延迟边界是从 AN 传输到 AT 接收所允许的指定时间。自适应加权算法致力于满足延迟边界要求和减少此类应用程序的 IP 分组所经受的平均延迟。对于同时具有速率和平均延迟要求的应用程序，自适应加权算法致力于满足速率要求和减少平均延迟。

对于诸如多媒体视频应用程序等某些类型的应用程序的另一个考虑事项是多媒体传输中连续的分组之间所经受的“抖动”。抖动指所接收的分组两两之间的时间变化。当连续波形稍早或稍迟到达接收机时，即发生抖动。在无线通信中，此类波形通常传递逻辑 1 或 0，它们随即在接收机处被解码。定义为抖动的时间变化使所接收到的传输的视觉效果产生畸变。自适应加权调度算法降低最坏情况的延迟变化，以及对延迟敏感的应用程序的相继分组之间的延迟变化。

满足各用户的 QoS 要求的同时，自适应算法还被设计成当若干应用程序流程“一致”时满足那些流程的速率要求。如果应用程序流程根据预先指定的通信量概况发送数据，则称其为“一致的”。如果具有速率要求的流程不一致，即，它们发送的数据多于其通信量概况中所预先指定的，则该算法将较高的优先权给予具有较

低数据率的流程。尽管本文中在 cdma2000 1xEV-DO 的上下文中描述自适应加权算法，但是这些概念和方法也可适用于其它类型的无线网络。

就多媒体应用程序流程而言，每条流程由以下各项定义：(1)通信量概况；(2)QoS 概况；(3)因特网协议（IP）源地址；和(4)IP 目的地址。流程还可包括：(5)L4 协议类型；(6)L4 端口号；和(7)L4 目的端口号，其中 L4 指协议栈中的传输控制协议（TCP）/用户数据报协议（UDP）层。例如，可将对应于一个 MPEG 应用程序的 MPEG 音频和 MPEG 视频视为两个单独的流程。

每条流程都由通信量概况指定，并受到监视或整形以确保其与该通信量概况一致。通信量概况由记为 σ 的表示突发性的度量的变量，和记为 r 的流程平均数据率定义。因此，每条流程由通信量概况（σ，r）描述。QoS 概况由以下各参数中的至少一个定义：(1)记为“D”的延迟边界，它定义 IP 分组从传输到接收所允许的时间。对于多媒体应用程序流程，系统可指定延迟边界。对于诸如网络浏览等一些其它应用程序流程，系统可指定平均延迟（AvgD）来取代或补充延迟边界；(2)记为“j”的抖动边界，它定义在 AT 处所接收的分组两两之间最大可允许的时间变化；和(3)记为“R”或“req_rate”的服务速率（请求速率）。

为定义延迟边界 D，参考图 6，它是包括各个 AN 元件和一个 AT 的时序图。多媒体流程从多媒体源（未图示）经由 PDSN、BSC、和 BTS 传输到 AT。一个 IP 分组在时间 t_0 从 PDSN 发送，并在时间 t_3 在 AT 处接收。参数 D 定义从时间 t_0 到时间 t_3 的最大可允许时间，即，D 指定对 t_3- t_0 的限制。

为定义抖动边界 j，参考图 7A，它是包括若干 AN 元件和一个 AT 的时序图。第一个分组在时间 t_1 从 PDSN 发出，并在时间 t_1’在 AT 处接收。第二个分组在时间 t_2 从 PDSN 发出，并在时间 t_2’在 AT 处接收。抖动边界 j 定义连续分组两两之间最大可允许的变化，其中变化给定为 (t_2’-t_1’)-(t_2- t_1)。图 7B 进一步详述在若干时隙上传输的若干连续 IP 分组。

在一个实施例中，QoS 概况被归类成组，称为 QoS 调度组（QSG）。表 1 列出了这些类别。

表 1

索引	延迟边界（D）	抖动边界（j）	服务速率（R）	平均延迟（AvgD）	应用程序示例
1	X	X	X	-	MPEG 会议，VoIP，视频流
2	-	-	X	X	网络浏览
3	-	-	X	-	FTP

4	-	-	-	-	尽力服务型

图 8 示出根据自适应加权调度算法对流程的处理。流程 900、902、904 和 906 由标记为“S”的调度单元 908 处理。调度单元 908 应用一种自适应加权调度算法，其中为每条流程使用 QSG 概况。QSG 概况识别用于如以下详述地计算自适应权值的变量。调度单元 908 随即将调度传输 910 输出到所选择的 AT。

描述称为 DRC/T 算法的 PF 调度算法，其中各分组被分类到 m 个队列，例如 Q1、Q2、……、Qm。令 DRC[k,n]为由移动站请求的 DRC，对应于时隙 n 的流程 k。调度器选择具有最高选择度量值的流程，其中：

$$Y[k,n+1]=DRC[k,n+1]/T_k[n+1],\forall k,\forall n \text{ 。} \tag{1}$$

Y[k,n+1]是时隙(n+1)中队列 Qk 的选择度量，并且

$$T_k[n+1]=(1-\frac{1}{t_c})T_k[n]+\frac{1}{t_c}R[k,n], \tag{2}$$

$$\frac{1}{t_c}\le 1,\text{并且}$$

$$0<1/t_c \tag{3}$$

如本文中所使用的，t_c是计算平均值所用的时间常数。

自适应 w*DRC/T 算法

在一个实施例中，称为“自适应 w*DRC/T”算法的自适应加权调度算法，向每条流程分配一个初始权值。假设分配给流程 k 的初始权值记为 w_k，由 AT 请求的对应于时隙 n 流程 k 的 DRC 是 DRC[k,n]。自适应 w*DRC/T 算法在每个时隙 n 为每条流程 k 计算以下度量：

$$Y_k[n]=aw_k[n]*DRC_k[n]/T_k[n] \tag{4}$$

此处，流程 k 和时隙 n 的吞吐量$T_k[n]$是如 PF 算法中为 DRC/T 所定义的。如在自适应加权调度算法中所使用的，$aw_k[n]$是流程 k 在时隙 n 的自适应权值。自适应 w*DRC/T 调度算法以数种模式工作，其中模式是由 QSG 定义的。流程 k 在时隙 n 的自适应权值$aw_k[n]$是基于如下所述的调度器模式和一组所选择的策略或机制所计算的。注意要为每条流程计算等式(4)，其中将根据专属于每条流程的公式来计算自适应权重。换言之，该调度算法考虑给定流程的 QoS 概况，并使用该 QoS 概况来构造对该流程的自适应权值的计算。以此方式，具有不同 QoS 要求的不同流程可具有不同地计算得到的自适应权值。该调度算法接下来选择具有最大$Y_k[n]$值的流程以在时隙 n 中服务。

自适应 w*DRC/T 调度器以以下模式工作：

模式 I [aw*DRC/T](r,d,j)：为延迟和抖动敏感应用程序而设计对延迟和抖动边界具有严格要求并要求某个最小速率。

模式 II [aw*DRC/T](r,d)：用于具有平均延迟和速率要求的应用程序。

模式 III [aw*DRC/T](r)：用于仅指定速率要求的应用程序。

模式 IV [DRC/T]：用于未指定任何 QoS 计划但由 DRC/T 算法服务的流程。基于 QoS 要求，可对给定流程使用特定模式的自适应 w*DRC/T 算法。还可对流程应用**模式 II** 以增加调度器给予该流程的吞吐量。例如，**模式 II** 可用于 FTP 应用程序，以潜在地增加对应应用程序流程的吞吐量。

以下给出分组应用程序（即，QSG）的一个示例：

组 I：对延迟边界和延迟变化具有严格要求类似于 IP 电话（VoIP）的应用程序。注意此类应用程序还常常具有速率要求。使用调度器**模式 I**。

组 II：对延迟边界和延迟变化具有严格要求的多媒体会议应用程序。即时这些应用程序中的一些是自适应的，为一致的高质量起见确保服务速率仍然是合乎需要的。使用调度器**模式 I**。

组 III：对于延迟边界、速率和延迟变化具有要求的视频流应用程序。使用调度器**模式 I**。

组 IV：具有速率和（平均）延迟要求的网络浏览应用程序——使用调度器**模式 II**。

组 V：具有速率要求的 FTP 应用程序——使用调度器**模式 III**。或者，使用具有不严格的延迟限制的调度器**模式 II**。

组 VI：尽力服务型应用程序——使用无自适应加权的 PF 算法，即，DRC/T 算法。

注意，数据库事务、游戏和其它应用程序也可分别根据 QoS 要求分类到合适的组中。

图 9 示出具有多个等级的自适应加权调度器，多个等级包括，但不限于，等级 I 和等级 II。等级 I 调度器具有多个调度器 S1、S2、S3、……、Sm，其中 m 指组的总数。图 9 中的每个等级 I 调度器运行自适应 w*DRC/T 调度算法的一个特定操作模式，并从该组中选择一条流程。首先，等级 I 调度器计算 Y 的部分，具体而言是吞吐量 T，以及速率补偿因子 α 。接下来，等级 II 调度器考虑各条流程，并向等级 I 调度器提供足够的输入，以由等级 I 调度器完成对选择度量 Y 的计算。

一旦为所有未决流程完成完成对 Y 的计算，等级 I 调度器对 Y 值进行估值，并选择具有最高 Y 值的流程。每个等级 I 调度器对一组具有相似 QoS 要求的流程进行估值。每个等级 I 调度器所选择的流程随即被提供给等级 II 调度器，以供与来自其它组的流程进行比较。等级 II 调度器考虑每组一个所选择的流程，并选择具有最高度量（aw*DRC/T）或 Y 值的流程。当调度器需要选择一条流程进行服务时，为每个时隙重复此过程。替换实施例可使用单等级的调度器，或可使用比图 9 中所示更多的等级。替换实施例可包括不同个数的等级 I 调度器，其中等级 I 调度器对应于流程组织。

一般而言，将自适应权值的计算是给定为若干参数的函数，并给定为：

$$a = f(\Phi, \gamma, \alpha, \beta, \delta)。 \tag{5}$$

延迟补偿函数记为 Φ。等待时间常数记为 γ。速率补偿函数记为 α。DRC 补偿函数记为 β。增强抖动补偿因子记为 δ。注意不是对于所有多媒体服务所有参数都具有实际的值。例如，当仅给定流程的 QoS 要求是指定的数据率时，那么变量 α 将被指定，（速率参数将有实际的值）而所有其它常数将被设为等于 1。在此情形中，自适应权值计算中将仅包括速率参数。在一个实施例中，自适应权重按下式计算：

$$a = \Phi * \gamma * \alpha * \beta * \delta \tag{6}$$

其中算子为乘号。以下讨论提供自适应权值计算中可包括的各种补偿项的细节。

对于模式 I 的应用程序，QoS 概况指定等式(6)中所指示的所有参数。自适应权值计算考虑由于延迟阈值违反而产生的延迟补偿，由于等待时间阈值违反而产生的延迟补偿，由于速率违反而产生的速率补偿，和由于加强抖动阈值违反而产生的加强抖动补偿。此概念提高违反指定 QoS 要求的流程的权值。一旦被违反 QoS 要求所触发，此类流程即被给予信任分。信任分是通过将流程的权值乘以延迟补偿函数的合适的值来实现的。这个结果还要乘以速率补偿和加强抖动补偿。

相反，当流程表现为在接收超额的服务时，该流程可被处罚。可用各种方法中的任何一种来处罚流程。根据一种方法，可通过减少流程权值来直接处罚流程。根据另一种方法，可通过维持该流程权值并同时增加其它落后的（即，那些未达到所要求的 QoS 的流程）用户的权值来间接处罚流程。

有各种计算延迟补偿以考虑违反延迟阈值的机制。假设流程 k 的延迟阈值记为 dth_Φ_k，流程 k 在时隙 n 中由于延迟阈值违反而产生的延迟补偿记为 Φ_k [n]。为计算延迟补偿 Φ_k [n]，考虑每条流程所有三个队列（即，Tx、RTx 和 ARQ）中的分组。

对于每条流程，还对 Φ 指定最大和最小阈值，以确保一条流程不会消耗若干个相继的时隙而使其它流程匮乏时隙。这样设计还确保由于延迟阈值违反而产生的流程延迟补偿项至少和最小阈值一样好。令 $\Phi_{thres,min,k}$ 和 $\Phi_{thres,max,k}$ 是为每条流程 k 指定的最小和最大阈值。这（对于所有 k 和所有 n）导致：

$$\phi_{\text{thres,min,k}} \le \phi_k[n] \le \phi_{\text{thres,max,k}}, \forall k, \forall n。 \tag{7}$$

将使用以下定义来推导对延迟补偿的计算：

D[n]：定义在时隙 n 的开始处经受延迟阈值违反的一组流程（即，每个此类流程在时隙 n 的开始处都至少有一个超过该流程延迟阈值的分组）。

$defpkts_k$[n]：定义在时隙 n 的开始处流程 k 的“亏数”分组。亏数分组是还未在流程中传输的分组，*defpkts* 具体定义为在 BTS 中停留时间超过流程 k 的延迟阈值的同等大小的（MAC）分组的个数。

$required\_rate_k$：定义流程 k 的请求速率。

$ndefpkts_k$：定义流程 k 的归一化亏数分组个数，具体定义为：

$$ndefpkts_k = \frac{defpkts_k}{required\_rate_k} \tag{8}$$

注意 BTS、BSC 和 PDSN 中的分组可能是不相等的大小，因此，这里计算亏数位而不是分组的个数是有好处的。

如果流程的 HOL 分组在 BTS 队列中停留的时间段大于预先指定的阈值，则可用以下机制对该流程进行补偿。为此目的使用的等待时间阈值应当大于或等于为计算 Φ 所使用的阈值。对于流程 k，等待时间阈值记为 dth_γ_k，其中等待时间阈值受 $\text{dth\_}\gamma_k \ge \text{dth\_}\phi_k, \forall k$ 的约束。为了选择流程的 HOL 分组，首先考虑来自流程的 Tx、RTx 和 ARQ 队列的 HOL 分组，并基于 BTS 处的等待时间选择其中的一个，即，选择在 BTS 中等待了最长时间的那一个。令 $\gamma_k[n]$ 为在时隙 n 开始处流程 k 的等待时间补偿，$S_k[n]$ 为在时隙 n 处流程 k 的 HOL 分组在 BTS 队列中停留的时间。对于每条流程 k，还指定最小阈值 $S_{thres,min,k}$ 和最大阈值 $S_{thres,max,k}$，满足 $S_{thres,\min,k} \le S_k[n] \le S_{thresh,\max,k}, \forall k, \forall n$。

根据一个实施例，当流程经受延迟阈值违反或等待时间阈值违反时应用延迟补偿。该机制将 DRC 数据率请求应用到自适应权值。令 $\beta_k[n]$ 为时隙 n 中流程 k 的 DRC 调整函数。为每条流程 k 指定最小阈值 $\beta_{\min,thres,k}$ 和最大阈值 $\beta_{\max,thres,k}$，满足 $\beta_{\min,thres,k} \le \beta_k[n] \le \beta_{\max,thres,k}$。

尽管对于诸如视频/音频会议等一些应用程序来说，上述补偿机制帮助减少流

程中的延迟变化，包括更有效地延迟变化（抖动）控制并进一步减少延迟变化可能是合乎需要的。以下机制通过减少流程相继分组两两之间的延迟变化，来提供有效的延迟变化控制。具有较大 IP 分组大小的流程从此补偿机制中获益更多。

假设 at(k,j)为 BSC 入口处流程 k 的 IP 分组 j 的到达时间。令 dt(k,j)为此 IP 分组从 BTS 离开的时间，即，到此时间为止此 IP 分组的所有分段都已由 BTS 处的前向链路调度器所传输。令 $pend_{k,j}$[n]为 BTS 和 BSC 处流程 k 的 IP 分组 j 的字节总长度。假设 $dv_{k,target}$ 为流程 k 相继 IP 分组两两之间的目标延迟变化（抖动），$dv_{k,thres}$ 为此流程预先定义的增强抖动阈值，并使 $dv_{k,thres}<dv_{k,target}$。在一个实施例中，当相继 IP 分组两两之间的延迟变化超过 $dv_{k,thres}$ 时，该算法即为流程 k 触发增强延迟变化补偿机制。

图 10 提供对应于一个实施例的体系结构图。每个应用程序流程由通信量概况、QoS 概况和 DRC 请求（即，请求数据率）描述。每个通信量概况包括突发性的度量和平均数据率。每个 QoS 概况包括类类型和参数边界。类类型可以是模式 I、模式 II、模式 III 或模式 IV 中的一种。边界指定延迟、抖动、和所请求的数据率的边界。诸如网络浏览等一些应用程序可指定平均延迟而不是延迟边界。选择模式 I 的延迟阈值小于抖动边界；对于模式 II，选择延迟阈值小于平均延迟。选择增强抖动阈值小于抖动边界。替换实施例可将较多或较少信息应用于每个应用程序流程，其中 QoS 要求可专属于网络和配置。

图 11 是为每个类类型指定 QoS 要求和 QoS 参数的表。如图所示，模式 I 对应于最严格的要求，而模式 IV 对应于不指定任何 QoS 要求的尽力服务型。替换实施例可包括其它 QoS 要求、QoS 参数和/或模式。

图 12A 到 12E 作为有效应用程序流程的一部分，示出对应用程序流程的处理和对该应用程序流程的调度。图 12A 所示是对单个应用程序流程进行初始化和设置的流程图。该过程在步骤 1100 开始以选择用于每个补偿参数的机制。补偿参数包括，但不限于：延迟（Φ）；未决时间（γ）；DRC（β）、抖动（δ）、和速率（α）。在步骤 1102 为适用的补偿参数选择阈值。注意补偿参数可包括对于 AN 具有重要性的应用程序流程的任何参数。在步骤 1104 选择用于计算中间权值的算法，其中中间权值用于计算调度所用的自适应权值。在步骤 1106 设置比例参数（C）和优先权因子（Z），这两者都用于计算自适应权值。在步骤 1108 为此应用程序流程设置初始权值。在步骤 1110 对该应用程序流程的 QoS 要求进行估值。如果除了 DRC 要求所标识的速率以外没有指定的 QoS 要求，则适用默认条件。默

认条件为如上所述的“尽力服务型”。在此情形中，默认处理将用于此应用程序流程的所有补偿因子设为等于1。对于本实施例来说，在此情形中，对等式(6)的计算适用乘法算子，因此将各因子设为1有效地忽略那些因子，即，那些因子不影响加权。注意替换实施例可实现其它机制和函数，并因此适用其它机制来忽略具体或所有的补偿因子。

尽力服务型处理在步骤1112和1116继续。所得的调度因子计算是与比例公平计算一致的。如果应用程序流程具有QoS要求，则处理前进至步骤1114。步骤1114和1116指示处理在接下来的附图中继续。

图12B从步骤1114起继续图12A的处理。在步骤1120对当前时隙的处理开始。在步骤1122判定应用程序流程的类类型。在步骤1128处理模式I，在步骤1126处理模式II，在步骤1124处理模式III。在步骤1128监视模式I的QoS参数；在步骤1124监视模式III的QoS参数。随即在步骤1130、1140和1150处对QoS违反进行检查，这在图12C和12D中进一步详述。

对于模式I、II或III的应用程序，应用程序流程的处理在图12C的步骤1130继续。在步骤1132，该算法周期性地监视速率违反。注意周期性地进行速率补偿计算，并在其后的多个时隙中使用。如果在步骤1134检测到速率违反，则处理前进至步骤1138以计算速率补偿因子（α）。否则，在步骤1136将速率补偿因子（α）设为等于1。处理随即前进至步骤1160，这将在图12E中进一步详述。

对于模式I或II的应用程序，应用程序流程的处理在图12C的步骤1140继续。在步骤1142该方法在每个时隙对延迟和抖动违反进行监视。如果在步骤1144检测到延迟和/或抖动违反，则处理前进至1148以根据在初始化所选择机制计算延迟补偿因子（Φ）。对于具有请求的增强抖动补偿的模式I的流程，那么还计算增强抖动补偿因子（δ）。对于没有请求的增强抖动补偿的模式I流程和对于模式II的流程，δ设为1。否则，在步骤1146将延迟补偿因子（Φ）设为等于1，并将δ设为等于1。处理随即前进至步骤1160，这在图12E中进一步详述。注意对于模式I或模式II的应用程序流程，可一前一后地或并行地进行违反检查。换言之，可在时间上连续地或并行地进行速率违反和延迟/抖动违反检查。

对于模式I的应用程序，应用程序流程的处理在图12D的步骤1150继续。在步骤1152该方法对于等待时间违反进行监视。如果在步骤1154检测到等待时间违反，则处理前进至步骤1158以根据在初始化选择的机制来计算等待时间补偿因子（γ）。否则，在步骤1156将等待时间补偿因子（γ）设为等于1。处理随即前

进至步骤 1160，这在图 12E 中进一步详述。注意对于模式 I 的应用程序流程，可一前一后或并行地进行违反检查。换言之，可在时间上连续地或并行地进行速率违反、延迟/抖动违反和等待时间违反的检查，如步骤 1170 和 1172 所示。

图 12E 示出转自步骤 1160 和 1116 的处理。在步骤 1162 根据 QoS 参数和补偿因子，为应用程序流程计算自适应权值，给定为：

$$aw=f(\Phi,\gamma,\alpha,\beta,\delta) \tag{9}$$

在步骤 1164 按下式计算调度因子或调度度量：

$$调度因子=aw*(DRC)/T \tag{10}$$

在步骤 1166，该调度算法随即根据为每个有效的应用程序流程计算所得的调度因子，对应用程序流程进行调度。

图 13 根据一个实施例示出适合应用调度算法的 BTS 1200。BTS 1200 包括调度单元 1202、应用程序流程处理器单元 1206、QoS 参数估值 1204、自适应权值计算单元 1212 和 CPU 1208，以上每一个都耦合到通信总线 1210。调度单元 1202 通过为每个应用程序流程准备调度因子、随后根据这些调度因子在各个有效应用程序流程之间进行选择来进行调度。给定系统的策略和目标被结合到调度算法中。QoS 参数估值 1204 对 QoS 违反进行监视，并向调度单元 1202 和权值计算单元 1212 提供信息。应用程序流程处理执行处理，包括，但不限于，将分组引导到目标 AT，从目标 AT 接收用于调度的 QoS 信息、并将该信息提供给 QoS 参数估值 1204。BTS 1200 还包括存储器 1214，用于存储中间信息，和维护用于计算平均值、流程队列等的数据。违反检查是在 BTS 处进行的。一个实施例持续对为每条流程所发出的字节数进行计数，并将该信息用于速率违反的检查。当每个分组到达 BSC 处，它都被打上时间戳。只要分组仍停留在 AN、BSC 或 BTS，时间就持续渐增。BTS 将此时间用于检测阈值违反，并随即根据流程计算延迟、等待时间或增强抖动补偿函数。

允许进入控制

允许进入控制指允许请求数据服务的用户进入中的决策过程。当新用户请求诸如具有 QoS 要求的应用程序等数据服务时，AN 确定是否有可用资源来支持该使用。允许进入过程考虑所请求的应用程序、当前的使用、以及 QoS 和网络统计量。如果 AN 确定可支持该新用户，那么对应的应用程序流程即被允许。否则，如果当前没有可用资源，则应用程序流程被拒绝或放置在队列中以等待状态的改变。注意

新用户实际上可以是具有请求额外服务的当前有效的应用程序流程，即，额外应用程序流程。

除了允许进入控制以外，并作为其一部分，可实现一种用于终止有效应用程序流程的先占过程，其中对当前操作情况进行估值来作出先占的决策。在此情形中，就 QoS 违反以及数据率对每个当前流程进行估值。

本节给出一种自适应的每个扇区的允许进入控制算法。该允许进入控制算法决定在给定无线多媒体网络中是否允许（或先占）一条流程。因此可能能够确定给定网络中可允许的（每个类的）流程数量。本文中所给出的允许进入控制算法的若干实施例包括同时执行用户间和用户内 QoS 两者监视、并随即将此信息应用于允许进入和/或先占决策的机制。这些实施例被设计成确保对于所允许进入的流程和用户，每条流程和每个用户的 QoS 要求得到满足。这些机制有助于协调允许进入控制算法和分层结构调度算法。

调度和允许进入控制是无线网络中前向链路（FL）QoS 管理的一部分，其中此类管理是一个复杂的问题。QoS 管理是通信网络的设计和操作中一个重要的考虑因素。应用程序流程是根据如系统所定义的准则等来分类的。在一个实施例中，分类是根据 QoS 要求。首先，允许进入控制确定在当前操作情况下可允许进入的流程数量。该流程数量随即被分成每个类的流程数量。系统随即进行操作以满足每个所允许进入的流程的 QoS 要求。注意流程的数量可随时间并随应用程序的类型动态地改变。例如，在第一时间，接入网络（AN）可支持对每类应用程序准许具体数量流程的第一情形。在第二时间，AN 可支持对各类应用程序中的至少一类允许不同数量流程的第二情形。

调度器（即，调度算法）在所允许的流程中实现一种公平策略。调度器还试图试图对有 QoS 违反的流程进行恰当恢复。操作者的收入和利益是取决于所用调度算法的有效性。更有效和特征丰富的算法提供增加这些利益的机会。

就允许进入控制而言，一个实施例实现一种基于预订因素的方法。注意基于预订的方法常常用在有线网络的允许进入控制算法中。在无线网络中，每个用户的信道情况持续变化，因此如 BTS 调度器所见的前向链路容量也持续变化。基于有线预订因素的算法假设固定的链路容量，并因此不能直接适用于无线网络。

对于无线网络，一个实施例为 FL 管理提供一种自适应预订因素基（ASF）的允许进入控制算法，其中网络支持多个具有 QoS 要求的应用程序流程。无线网络中的 ASF 允许进入控制通过监视 QoS 统计量和网络统计量，来动态地更新预订因

素。可使用各种机制来执行更新功能。因此能够利用自适应预订因素来采取纠正性的行动。此外，ASF 还用于实现先占方法。

在每个时间 t 计算 ASF，AS(t)。该过程为 AS(t)确定最小阈值 $AS_{\min\_\ prespecified}$，和最大阈值 $AS_{\max\_\ prespecified}$，以使 $1 \le AS_{\min\_\ prespecified} \le AS(t) \le AS_{\max\_\ prespecified} < \infty, \forall t$。初始时，向 ASF 分配值 $S_{initial}$，以使 AS(0)= $S_{initial}$。

图 14 是一张时序图，测绘出作为时间函数的最大数据率、保留带宽、和可用带宽。BTS 在前向链路上可传输数据的最大速率（$L_{\max}$）提供资源分配的上界。对有效应用程序流程进行估值以确定保留带宽 Res(t)。使用 QoS 违反和网络负载有关的统计量，进行自适应预订因素的计算和对在时间 t 希望保留的前向链路容量 L(t)估计。注意 $L(t) \le Res(t)$ 是可能的。例如，假设所允许的流程在它们被允许进入的时候经历非常好的信道情况。现在，若干流程的信道情况变差并且一些流程没有达到相关联的 QoS 保证。在此情形中，系统可能希望在允许更多流程上变得更加保守，并设 Avail(t)=0。另一方面，如果 L(t)>Res(t)，则系统可设 Avail(t)=L(t)-Res(t)。值 L(t)即为基于先前的 QoS 违反和网络负荷有关的统计量，对在时间 t 希望保留的前向链路容量的估计，并根据 $L_{\max}$ 和 ASF 进行计算，如下：

$$L(t) = \frac{L_{\max}}{AS(t)}, \forall t, \tag{11}$$

其约束为：

$$L(t) \le \frac{L_{\max}}{AS_{\min\_\ prespecified}} < L_{\max} \text{ 并且} \tag{12}$$

$$L(t) \ge \frac{L_{\max}}{AS_{\max\_\ prespecified}} > 0 。 \tag{13}$$

可用带宽 Avail(t)按下式计算：

$$Avail(t)=\text{maximum}(L(t)\text{-Res}(t),0) \tag{14}$$

如图 14 中所示的各种资源的度量是由从用户接收的数据率控制（DRC）数据请求来确定的。每个用户在后向链路上发送 DRC 数据请求。在 cdma2000 1xEV-DO 或其它 HRPD 类型的系统中，用户在每个 RL 传输的时隙上发送 DRC 数据请求。如图 15 所示的，来自用户 1 的数据请求（DRC 1）和来自用户 2 的数据请求（DRC 2）随时间改变。这些数据请求和所要求的 QoS 确定保留带宽（Res）。注意提供该 DRC 值和 Res 之间的关系是作为示例。替换实施例和情形可能导致不同的关系。

每个应用程序流程都具有平均速率和突发性方面的指定通信量概况，其中流程 f_k 的通信量概况由（σ_k, r_k）给定。此处，r_k 是流程 f_k 的平均请求速率，σ_k 是突

发性的度量，其中流程f_k的所请求速率给定为 req_rate(f_k) = r_k。

根据一个实施例，允许进入控制就允许进入对流程 f_k 进行估值。允许进入控制首先为用户应用对应于流程f_k的观测到的 DRC，$u(f_k)$ 以满足所请求的速率，其中所观测到的 DRC 小于或等于该用户的平均 DRC 数据要求，如下所示：

$$req\_rate(f_k) \le Avg(DRC(u(f_k)))。 \tag{15}$$

随即进行 ASF 的计算 AS(t)，并将其用于按下式计算 Avail(t)：

$$Avail(t) = \max imum(\frac{L_{\max}}{AS(t)} - \mathrm{Re}s(t),0) \tag{16}$$

最后，对于流程f_k的允许进入决策在时间 t 考虑：

$$req\_rate(f_k) \le Avail(t)。 \tag{17}$$

如果流程f_k被允许进入，则如图 14 中所示的资源度量按下式更新：

$$\mathrm{Re}s(t) = \mathrm{Re}s(t) + req\_rate(f_k) \tag{18}$$

$$Avail(t) = Avail(t) - req\_rate(f_k) \tag{19}$$

AN 继续为所有所允许进入的流程监视 QoS 统计量，并监视网络有关的统计量。监视为适应预订因素提供反馈。

AS(t)的自适应方法：每个扇区的 QoS 和网络统计量

图 18A 到 18E 提供一种支持多个具有 QoS 要求的应用程序流程的系统的允许进入控制的方法 300 的流程图。在图 18A 中，当在判定菱形框 302 AN 接收到对新流程的请求时，在步骤 304 应用允许进入控制程序。否则，该过程等待新的流程请求。注意在此时间期间，AN 继续监视当前的操作情况，来进行当前有效流程的 QoS 统计量和流程的网络统计量。允许进入控制程序确定是否有资源可用以支持该新流程。诸如图 14 中所示的资源度量在步骤 305 得到更新。如果在判定菱形框 306，新的流程被允许进入，则处理前进至 307 以应用自适应调度过程。

步骤 304 的允许进入控制程序在图 18B 中进一步详述。在判定菱形框 308，如果流程f_k的请求速率大于流程f_k的平均 DRC 数据要求，则处理前进至步骤 312 以拒绝流程f_k进入。否则，处理返回判定菱形框 312 以确定流程f_k的请求速率是否大于在时间 t 的可用资源 *Avail*。如果所请求的速率小于 *Avail*，那么在步骤 314 该流程被允许进入，否则在步骤 312 拒绝该流程。

步骤 305 对资源度量的更新在图 18C 中进一步详述。在步骤 320，资源度量 *Avail* 和 Res 得到更新。在步骤 322QoS 统计量得到更新和监视。在步骤 324，ASF

基于步骤 320 和 322 的结果得到更新。在步骤 326，估计的资源等级 L 被重新计算。如果在步骤 328 有新的流程被请求，则处理返回以在图 18A 的步骤 304 进行处理。如果在步骤 328 没有新的流程被请求，那么在步骤 330 为每个用户确定用户在该扇区中的存在。在步骤 332 确定采样持续周期。

继续讨论图 18D，在步骤 340 为每条流程选择 QSG 参数。考虑两个并行的处理路径，其中第一路径在步骤 342 在速率采样间隔上处理速率违反。注意该速率采样间隔大于在步骤 332 计算所得的采样持续时间。第二路径将每个有效流程的处理具体化。对于给定的流程，该过程在步骤 344 确定在采样持续时间期间有延迟违反的 IP 分组的比率。在步骤 346，为所考虑的流程计算采样持续时间期间有抖动违反的 IP 分组的比率。随即在步骤 348 计算在采样持续时间期间流程所用时隙的分数。在步骤 350 该过程确定在采样持续时间期间给予具有 QoS 要求的流程的时隙的分数。在步骤 352，该过程就 QoS 违反进行检查，并在步骤 354 确定 QoS 分组 ID。

处理前进至图 18E，步骤 360 为每个 QoS 分组计算流程的数量。在步骤 362 该过程随即计算对应于每个 QoS 统计量的 QoS 流程的分数。随即在步骤 364 将步骤 360 和 362 的结果与预先确定的阈值相比较。注意阈值可在操作期间被动态更新。在步骤 366 据此调整 ASF。

图 18A 到 18E 提供允许进入控制方法的一个实施例。在下文讨论允许进入控制方法的其它细节。诸如 BTS 等 AN 元件为每条流程收集每个扇区的统计量，并将此信息用于每个扇区的允许进入控制和先占算法。仅在对应于流程的用户在该分区中的期间收集每个扇区的统计量。BTS 周期性地收集 QoS 和网络有关的统计量。令 T 为收集这些信息之前的那个时间周期，其中 Z 是采样索引，t=Z*T。

考虑扇区 s 中的流程 f_k，其中 $u(f_k)$是对应于流程 f_k 的用户。该用户在时间 t_{enter} 进入扇区 s。在扇区 s 内部，在持续时间$[t_{reserve}(f_k, s), t_{free}(f_k, s)]$内为此流程保留资源，其中资源在时间 $t_{reserve}$ 得到保留。在用户请求应用程序服务时，资源被保留。用户 $u(f_k)$ 在以上持续时间内，在 $t_{enter,j}(f_k, s)$第 j 次进入扇区 s，并在 $t_{leave,j}(f_k, s)$第 j 次离开。因此，$t_{reserve}(f_k,s) \le t_{enter,first}(f_k,s)$ 且 $t_{free}(f_k,s) \ge t_{leave,last}(f_k,s)$。注意，出于预计该用户可能在某个未来的时间点移到此扇区，经由 QoS 信令协议 AN 会被告知要为流程保留资源是可能的。此处，$t_{enter,first}(f_k,s)$ 是用户 $u(f_k)$第一次进入此扇区 s 的时间，而 $t_{leave,last}(f_k,s)$ 是在流程 f_k 的生命期期间此用户最后一次离开扇区 s 的时间。

仅当在时间 t 以下三个条件都得到满足时，该算法在时间 t 将 QoS 和网络有关

的性能统计量纳入考虑：

$$t_{reserve}(f_k,s) \le t \le t_{free}(f_k,s) \tag{20}$$

$$t_{enter\_latest}(f_k,s)+\delta(f_k) \le t \le t_{leave\_latest}(f_k,s)，\text{并且} \tag{21}$$

$$IN\_IP\_PKTS(f_k,t,s) > \theta(f_k) \tag{22}$$

其中，$\delta(f_k)$和$\theta(f_k)$是为流程f_k预先指定的，$IN_IP_PKTS(f_k,t,s)$是在扇区s中在时间段(max(t-T,$t_{enter\_latest}(f_k,s)$),t)期间由BTS前向链路调度器调度以供传输的流程f_k的输入IP分组的个数。如果IP分组的最后一位在扇区s中被传输，则该位被计入该扇区的IN_IP_PKTS。变量$\delta(f_k)$指在该扇区中的出现被视为重要之前的时间。换言之，一旦用户在扇区s中停留了δ秒，则该过程开始对资源进行估值。注意，一旦得到允许，用户无须要求重新允许即刻离开和重新进入该扇区。

Qos 和网络统计量随即被用于为每个请求应用程序服务的流程进行允许进入准则的估值。在时间段(max(t-T,$t_{enter\_latest}(f_k,s)$),t)期间扇区 s 中对应于流程 f_k 的延迟IP分组按下式计算：

$$Frac\_delayed\_IP\_Pkts(f_k,t,s) = \frac{DELAYED\_IP\_PKTS(f_k,t,s)}{IN\_IP\_PKTS(f_k,t,s)} \tag{23}$$

其中$DELAYED_IP_PKTS(f_k,t,s)$对应于到时间t为止在扇区s的BTS处延迟了超过流程f_k对应的延迟边界的IP分组的个数。一旦在扇区s中检测到IP分组的延迟违反，即渐增该扇区延迟违反的计数。到时间t为止流程f_k有抖动边界违反的IP分组对的分数按下式计算：

$$Frac\_jitter\_viol(f_k,t,s) = \frac{JTR\_VIOL\_PKT\_PAIRS(f_k,t,s)}{IN\_IP\_PKTS(f_k,t,s)-1}, \tag{24}$$

其中$JTR_VIOL_PKT_PAIRS(f_k,t,s)$对应于到时隙 t 为止流程 f_k 有抖动边界违反的（相继IP分组的）IP分组对的数量。当一条流程的两个相继IP分组在一扇区上传输时，为该扇区进行此计数。

在时间段($t_{enter\_latest}(f_k,s)$,t)期间流程f_k的速率违反，

$$Rate\_viol(f_k,t,s) = \frac{req\_rate(f_k)-served\_rate(f_k,t,s)}{req\_rate(f_k)} \tag{25}$$

其中$req_rate(f_k) > served_rate(f_k,t,s)$。否则当流程$f_k$在时间t没有速率违反时，计算在($t_{enter\_latest}(f_k,s)$,t)时间期间扇区$s$中流程$f_k$的$served_rate(f_k,t,s)$。

注意，对于速率违反，该过程将时间段($t_{enter\_latest}(f_k,s)$,t)作为采样持续时间应用，但是对于延迟和抖动违反，该过程将时间段(max(t-T,$t_{enter\_latest}(f_k,s)$),t)作为采样持续

时间应用。

在时间段(max(t-T,$t_{enter\_latest}(f_k,s)$),t)期间流程f_k所使用的时隙分数按下式计算：

$$Frac\_slots\_flow(f_k,t,s)=\frac{SERVED\_SLOTS(f_k,t,s)}{IN\_SECTOR(f_k,t,s)} \quad (26)$$

其中$SERVED_SLOTS(f_k,t,s)$是在由(max(t-T,$t_{enter\_latest}(f_k,s)$),t)定义的时间段期间流程f_k受到服务的时隙的数量，$IN_SECTOR(f_k,t,s)$是(max(t-T,$t_{enter\_latest}(f_k,s)$),t)期间时隙的总数。在时间段(t-T,t)期间给予扇区 s 中具有 QoS 要求的流程的时隙的分数由下式给出：

$$frac\_slots\_qos\_flow(t,s)=\frac{\text{分区}s\text{中在}(t-T,t]\text{期间给予QoS流路的时隙数}}{\text{T(时隙)}} \quad (27)$$

动态流程分类

对于每条流程f_k，以下四个阈值是预先指定的，并用于执行每条流程的 QoS 和信道情况有关的检查：

$Frac\_delayed\_IP\_pkts\_thres(f_k)$，（分数延迟 IP 分组阈值($f_k$)）

$Frac\_jitter\_viol\_IP\_pkts\_thres(f_k)$，（分数抖动违反 IP 分组阈值($f_k$)）

$rate\_viol\_thres(f_k)$，（速率违反阈值(f_k)）和

$Frac\_slots\_viol\_thes(f_k)$，（分数时隙违反阈值($f_k$)）。

系统在每个时间 T 之后周期性地适应 ASF，其中 T 是预先指定的值。在给定时间 t 当第 Z 次（即，t=Z*T）进行自适应检查时，该过程考虑扇区 s 中有一些资源得到保留的那些流程，即，满足$t_{reverse}(f_k,s)\le t\le t_{free}(f_k,s)$的流程。对于来自此组流程的每一条流程$f_k$，进行检查以对以下“*每条流程的阈值检查*”进行估值：

$$Frac\_delayed\_IP\_Pkts(f_k,t,s) > Frac\_delayed\_IP\_pkts\_thres(f_k) \quad (28)$$

（分数延迟 IP 分组(f_k,t,s) > 分数延迟 IP 分组阈值(f_k)）

$$Frac\_jitter\_viol\_IP\_pkts\ (f_k) > Frac\_jitter\_viol\_IP\_pkts\_thres(f_k,t,s) \quad (29)$$

（分数抖动违反 IP 分组(f_k) > 分数抖动违反 IP 分组阈值(f_k,t,s)）

$$Rate\_viol\ (f_k,t,s) > rate\_viol\_thres(f_k) \quad (30)$$

（速率违反(f_k,t,s) >速率违反阈值(f_k)）

$$Frac\_slots\_flow(f_k,t,s) > Frac\_slots\_viol\_thes(f_k) \quad (31)$$

（分数时隙流程(f_k,t,s) >分数时隙违反阈值(f_k)）

如果满足以下两个条件，则对流程f_k进行以上四个检查：

$$t_{enter\_latest}(f_k,s)+\delta(f_k)\le t\le t_{leave\_latest}(f_k,s)，\text{以及} \quad (32)$$

$$IN\_IP\_PKTS(f_k,t,s) > \theta((f_k)) \quad (33)$$

当对于一条流程，条件(32)、(33)中的至少一个没有满足，但$t_{reserve}(f_k,s) \le t \le t_{free}(f_k,s)$为真时，为该流程将每条流程的阈值检查的结果标记为 NA。

该过程计算在采样持续时间期间，为具有 QoS 要求的流程使用的时隙的分数，还计算此分数值的阈值，其中 *Frac_slot_thres_qos_flows(s)*是扇区 *s* 中在时间段 T 内分配给具有 QoS 要求的流程的时隙的分数的上阈值。*Frac_slot_thres_qos_flows(s)* 的值用来检查下式是否成立：

frac_slots_qos_flows(t,s) > Frac_slots_thres_qos_flows(s)。 (34)

该过程将当前的流程分类到以下 QoS 调度组（QSG）之一：

QSG I 或 Q_DJR：具有延迟、抖动和速率要求的流程，

QSG II 或 Q_RavgD：具有速率和平均延迟要求的流程，

QSG III 或 Q_R：具有速率要求的流程。

考虑一组属于 Q_DJR 类的流程。如果给定流程f_k对于 NA 类别是不合格的，并且或者满足：

$$Frac\_delayed\_IP\_Pkts(f_k,t,s) > Frac\_delayed\_IP\_pkts\_thres(f_k), \quad (35)$$

或者满足

$$Frac\_jitter\_viol\_IP\_pkts\ (f_k,t,s) > Frac\_jitter\_viol\_IP\_pkts\_thres(f_k), \quad (36)$$

此流程被指定为有 delay_or_jitter_viol(f_k,t,s)=Y（延迟或抖动违反(f_k,t,s)=是）。否则，该过程为此流程设 delay_or_jitter_viol(f_k,t,s)=N（延迟或抖动违反(f_k,t,s)=否）。另一方面，如果在 NA 类别中是合格的，那么 delay_or_jitter_viol(f_k,t,s)=NA。

表 2

QoS 状态组 ID（QS_GID）	QoS 类	delay_or_jitter_viol(f_k,t,s)	rate_viol(f_k,t,s)> rate_viol_thres(f_k)
1	用于 Q_DJR 流程	是	是
2	用于 Q_DJR 流程	是	否
3	用于 Q_DJR、Q_RavgD、Q_R 流程	否	是
4	用于 Q_DJR、Q_RavgD、Q_R 流程	否（或 NA）	否（或 NA）

在每个时间 t，当进行 AS(t)的自适应检查时，该过程如表 2 中所示对 QoS 流程（即，具有 QoS 要求的流程）进行分类。每条流程都被分配一个 QoS 状态组 ID（QS_GID）。

QS_GID = 1：有速率和延迟（或抖动）违反的 Q_DJR 类的流程。

QS_GID = 2：有延迟（或抖动）违反而没有速率违反的 Q_DJR 类的流程。

QS_GID = 3：没有延迟和抖动违反、但有速率违反的 Q_DJR 类的流程。对于自适应应用程序可能产生此情形。并且，有速率违反的对应于 Q_R 和 Q_RavgD 类的流程被分配到此组中。

QS_GID = 4：没有 QoS（速率、延迟和抖动）违反的流程。如上文所述的 NA 类别中的流程也被放到此组中。

预订因素的自适应

令 N_k(t,s)为在时间 t 对应于 QSG k 的流程个数，N(t,s)为在时间 t 在扇区 s 中有一些资源得到保留的流程总数。

$$N(t,s)=\sum_k N_k(t,s) \tag{37}$$

令 M 表示 QoS 状态组 ID（QS_GID），结果为：

$$N_{Q\_DJR}(t)=\sum_{M=1}^{4} N_{Q\_DJR}(t,M) \tag{38}$$

$$N_{Q\_RAvgD}(t)=\sum_{M=3}^{4} N_{Q\_RAvgD}(t,M) \tag{39}$$

$$N_{Q\_R}(t)=\sum_{M=3}^{4} N_{Q\_R}(t,M) \tag{40}$$

在时间 t（在扇区 s 中）在 QoS 状态组 M 中，对应于流程 k 的 QSG k 的流程的分数由下式给出：

$$F_k(t,M,s)=\frac{N_k(t,M,s)}{N_k(t,s)} \tag{41}$$

在时间 t 有延迟（或抖动）和速率违反的流程的分数由下式给出：

$$Frac\_flows\_DJR\_viol(t,s) = F_{Q\_DJR}(t,\ M{=}1,\ s) \tag{42}$$

在时间 t 有延迟（或抖动）违反，但没有速率违反的流程的分数由下式给出：

$$Frac\_flows\_DJ\_viol(t,s) = F_{Q\_DJR}(t,\ M{=}2,\ s) \tag{43}$$

在时间 t 仅有速率违反的流程的分数由下式给出：

$$Frac\_flows\_R\_only\_viol(t,s)=\frac{N_{Q\_DJR}(t,M=3,s)+N_{Q\_RAvgD}(t,M=3,s)+N_{Q\_R}(t,M=3,s)}{N_{Q\_DJR}(t,s)+N_{Q\_RAvgD}(t,s)+N_{Q\_R}(t,s)} \tag{44}$$

没有 QoS 违反（或在 NA 类别中的）流程分数由下式给出：

$$Frac\_flows\_no\_na\_viol(t,s)=$$

$$\frac{N_{Q\_DJR}(t,M=4,s)+N_{Q\_RAvgD}(t,M=4,s)+N_{Q\_R}(t,M=4,s)}{N_{Q\_DJR}(t,s)+N_{Q\_RAvgD}(t,s)+N_{Q\_R}(t,s)} \quad (45)$$

S(t)的自适应是在每个时间段 T 之后周期性地进行的，T 是预先指定的。出于自适应的目的，可如下考虑以上各个组。

表 3

QSG	QS_GID=1 的流程的分数	QID=2 的分数	QID=3 的分数	QID=4 的分数
QSG_DJR	Frac_flows_DJR_viol(t,s)	Frac_flows_DJ_viol(t,s)	与 Frac_flows_R_only_viol(t,s)结合	与 Frac_flows_no_na_viol(t,s)结合
QSG_RavgD	不适用	未使用	与 Frac_flows_R_only_viol(t,s)结合	与 Frac_flows_no_na_viol(t,s)结合
QSG_R	不适用	不适用	与 Frac_flows_R_only_viol(t,s)结合	与 Frac_flows_no_na_viol(t,s)结合

以下阈值是预先指定的，并在以下所示的自适应方法中使用。

Frac_flows_thres_DJR：有延迟（或抖动）和速率违反的流程的分数上的阈值

Frac_flows_thres_DJ：有延迟（或抖动）违反、且无速率违反的流程的分数上的阈值

Frac_flows_thres_R：有速率违反（且无延迟或抖动违反）的流程的分数上的阈值

Frac_flows_thres_ok_qos：无 QoS 违反的流程的分数上的阈值

每当为 AS(t)进行自适应检查之后，该过程立即按以下顺序继续执行：

步骤 1：如果 $Frac\_flows\_DJR\_viol(t,s) \geq Frac\_flows\_thres\_DJR$：

$AS(t^+) = f_{qos}*AS(t)+x_{qos}$，从而 $AS(t^+) \geq AS(t)$。此处，f_{qos} 和 x_{qos} 是预先指定的。否则，

步骤 2：如果 $Frac\_flows\_DJ\_viol(t,s) \geq Frac\_flows\_thres\_DJ$：

则 $AS(t^+) = f_{delay\_jitter}*AS(t)+x_{delay\_jitter}$，从而 $AS(t^+) \geq AS(t)$。此处，f_{delay_jitter} 和 x_{delay_jitter} 是预先指定的。否则，

步骤 3：如果 $Frac\_flows\_R\_only\_viol(t,s) \geq Frac\_flows\_thres\_R$：

则 $AS(t^+) = f_{rate}*AS(t)+x_{rate}$，从而 $AS(t^+) \geq AS(t)$。此处，f_{rate} 和 x_{rate} 是预先指定的。

否则，

步骤 4：如果 $Frac\_flows\_no\_na\_viol(t,s) \geq Frac\_flows\_thres\_ok\_qos$：

则 $AS(t^+) = f_{all\_qos\_flows} * AS(t) + x_{all\_qos\_flows}$，从而 $AS(t^+) \geq AS(t)$。此处，$f_{all\_qos\_flows}$ 和 $x_{all\_qos\_flows}$ 是预先指定的。否则，

步骤 5：如果 $Frac\_flows\_no\_na\_viol(t,s) \geq Frac\_flows\_thres\_ok\_qos$，并且如果 $Frac\_slots\_qos\_flows(t,s) < Frac\_thres\_slots\_qos\_flows$：

则 $AS(t^+) = f_{ok} * AS(t) + x_{ok}$，从而 $AS(t^+) \leq AS(t)$。此处，f_{ok} 和 x_{ok} 是预先指定的。否则，

步骤 6：如果 $Frac\_flows\_no\_na\_viol(t,s) \geq Frac\_flows\_thres\_ok\_qos$，并且如果 $Frac\_slots\_qos\_flows(t,s) \geq Frac\_thres\_slots\_qos\_flows$：

则 $AS(t^+) = AS(t)$。

先占方案

图 19 根据一个实施例示出一种先占方法 400。方法 400 由在判定菱形框 402 确定 ASF 是否得到增加而开始。当检测到 ASF 的增加时，处理前进至步骤 404 以确定有最高速率违反次数的流程。换言之，当 ASF 增加时，先占方法 400 开始标识要先占的那些流程。在本实施例中，具有速率违反的流程被表示为先占的最佳候选。替换实施例可给予其它流程优先权，并可以动态改变优先权方案。

如果在判定菱形框 406 达到先占的最大值 P_{MAX}（以下详述），则处理前进至步骤 408 以先占有最高延迟违反次数的流程。否则处理返回判定菱形框 402。当在步骤 408 先占一条流程之后，处理前进至判定菱形框 410 以确定是否有多个具有最高延迟违反次数的流程。对于多条流程，处理前进至步骤 412 以先占使用最多时隙的流程。通常，此流程将具有很低的数据率，并因此在给定时间段期间消耗最多的时隙。处理随即返回判定菱形框 402。

在一个实施例中，应用该先占方法 400，其中 P_max 是在任何给定时间点允许被先占的最大流程个数。考虑满足表 3 中所示的两个先占组的条件的流程子集。具体而言，先占组 1 包含属于 QSG_R 或 QSG_RavgD 的流程，并有

$$rate\_viol(f_k,t,s) > rate\_viol\_thres((f_k)，和 \quad (46)$$

$$Frac\_slot\_flow(f_k,t,s) > frac\_thres\_slots\_flow(f_k)。 \quad (47)$$

先占组 2 包含属于 Q_DJR QSG 的流程，并且所述流程含有

$$Frac\_delayed\_IP\_Pkts(f_k,t,s) > Frac\_delayed\_IP\_pkts\_thres(f_k)和 \quad (48)$$

$$\text{Frac\_slots\_flow}(f_k,t,s) > \text{frac\_thres\_slots\_flow}(f_k)。 \quad (49)$$

表 4

先占组 ID	QSG	QoS 违反	时隙阈值违反
1	Q_R，Q_RavgD	$rate\_viol(f_k,t,s) > rate\_viol\_thres((f_k)$	$\text{Frac\_slot\_flow}(f_k,t,s) > \text{frac\_thres\_slots\_flow}(f_k)$
2	Q_DJR	$\text{Frac\_delayed\_IP\_Pkts}(f_k,t,s) > \text{Frac\_delayed\_IP\_pkts\_thres}(f_k)$	$\text{Frac\_slots\_flow}(f_k,t,s) > \text{frac\_thres\_slots\_flow}(f_k)$

步骤 1：如果在某个时间点，AS(t)得到增加（如在 AS(t)的自适应方法中），则该过程查看是否有一个或数条流程是有资格先占的。注意，当 AS(t)未得到增加时，没有流程被先占。

步骤 2：考虑对应于先占组 1 的流程子集。从这些流程外，选择具有最大 *rate_viol* 值的 *P_max* 条流程。如果有平局，则先占具有较大 *Frac_slots_flow* 值的流程。如果 *P_max* 条流程被先占，那么不再为速率违反先占更多的流程。

步骤 3：考虑对应于先占组 2 的流程子集。这些流程具有延迟和抖动要求，并有 $\text{Frac_delayed_IP_Pkts}(f_k,t,s) > \text{Frac_delayed_IP_pkts_thres}(f_k)$和 $\text{Frac\_slots\_flow}(f_k,t,s) > \text{frac\_thres\_slots\_flow}(f_k)$。从这些流程外，选择 P_max 减去步骤 2 中先占的具有最大 *Frac_delayed_IP_Pkts* 的流程个数。如果有平局，则先占具有较大 *Frac_slots_flow* 值的流程。

用户内和用户间的 QoS

一个移动用户可能同时有多条流程，即，多个应用程序。如本文中所给出的，用户可指定以下各项：

每条流程的关于其是否对延迟和抖动敏感的指示。如果它对延迟和抖动敏感，则要指定延迟和抖动边界。

每个用户的总目标速率（ATR）。这是前向链路分层结构的调度器旨在给予此用户的目标速率。

允许进入控制

给定在时间 t 的 R(t)个用户，记为 U_1、U_2、……、$U_{R(t)}$，将 $num\_flows(U_j,t)$ 看作为在时间 *t* 用户 U_j 的流程个数。假设，在时间 t 用户 U_j 有(k-1) 条流程被允许

进入，即 $num\_flows(U_j,t) = k-1$。为决定在时间 t 允许用户 U_j 的一个新流程 $f_{k,j}$ 进入，该过程使用用户 U_j 的所观测的 DRC 来检查以下：

$$\sum_{c=1}^{k} req\_rate(f_{k,j}) \le Avg(DRC(U_j))。 \tag{50}$$

当进行如前所述的 ASF 的自适应时，流程所取的时隙数和其对应的 DRC 也被纳入考虑。该过程按下式计算 *AS(t)*和 *Avail(t)*：

$$Avail(t) = \max imum(\frac{L_{\max}}{AS(t)} - \mathrm{Re}s(t),0) \tag{51}$$

如果下式成立，则该过程在时间 t 允许流程 $f_{k,j}$ 进入：

$$req\_rate(f_{k,j}) \le Avail(t) \tag{52}$$

如果此流程被允许进入，则作如下更新：

$$\mathrm{Res(t)} = \mathrm{Res(t)} + req\_rate(f_{k,j}), \tag{53}$$

$$Avail(\mathrm{t}) = Avail(\mathrm{t}) - req\_rate(f_{k,j}), \tag{54}$$

$$ATR(U_j,\mathrm{t}) = ATR(U_j,\mathrm{t}) + req\_rate(f_{k,j}), \tag{55}$$

该过程继续为所有被允许进入的流程和用户监视 QoS 统计量，并监视网络有关的统计量。使用这些信息来继续适应预订因素。随即计算并应用 ASF AS(t)。

分层结构的调度器

每条流程和每个用户的补偿：

每个延迟和抖动敏感的流程都被分配一个抖动阈值。对于用户 U_k 的每个对延迟和抖动敏感的流程 fx，该过程计算对应的延迟和抖动补偿Φ。如果流程在一个队列中没有任何超过延迟阈值的分组，那么

$$\phi(fx(U_k)) = 1。 \tag{56}$$

否则，计算

$$\phi(fx(U_k)) = C_{delay}(fx(U_k)) * \frac{ndefpkts(fx(U_k,n))}{ndefpkts_{\min}(n)} \tag{57}$$

此处，

$$ndefpkts(fx(U_k,n) = \frac{defpkts(fx(U_k,n)}{req\_rate(fx(U_k))} \tag{58}$$

对于用户 U_k 的每条流程 fx，

$$ndefpkts_{\min}(n) = ndefpkts的最小值 \tag{59}$$

考虑在时隙 n 的（所有用户的）所有流程，

$defpkts(fx(U_k,n) =$ 在时隙 n 用户 U_k 的流程 fx 违反其延迟阈值的未决 MAC

分组的个数。 (60)

对于在时隙 n 的用户 U_k 的速率补偿，定义，

$$\alpha(U_k,n)=\frac{ATR(U_k)}{ASR(U_k,n_{prev}(n))} \tag{61}$$

此处，

$ASR(U_k,n_{prev}(n))$ = 时隙 n_{prev} 中用户 U_k 的总服务速率，并且 (62)

$n_{prev}\le n$。 (63)

时隙号 n_{prev} 是当出于调度算法的目的监视速率时 n 上或之前的最后一个时隙。

该过程为用户 U_k 定义在任意时隙 n 的总延迟补偿。考虑此用户具有延迟要求的所有流程，并查看每个这些延迟敏感的流程的排头（HOL）MAC 分组。如果没有任何一个在系统中停留时间超过其延迟阈值，则：

$agg\_delay\_comp(U_k,\mathrm{n}) = 1$。 (64)

否则，为此用户考虑有 HOL 分组在系统中停留时间长于延迟阈值的流程子集。对于此用户，为有 HOL MAC 分组在系统中停留时期超过这些流程的延迟阈值的用户 U_k 的所有流程 fx，计算：

$$agg\_delay\_comp(U_k,\mathrm{n}) = \sum_x w(fx(U_k))*\phi(fx(U_k),n)\,, \tag{65}$$

此处，$w(fx(U_k))$ 是分配给用户 U_k 的流程 fx 的初始权值。

自适应权值计算

为了为此用户计算自适应权值，该过程在每个时隙为每个至少有一个延迟敏感流程的用户的每一条流程进行延迟阈值违反的检查。该过程为每个这样的用户 U_k 计算：

$agg\_delay\_comp(U_k,n)$ (66)

如果 $agg\_delay\_comp(U_k,n)>1$，则该过程如下为用户 U_k 计算自适应权值：

$aw^t\ (U_k,n) = agg\_delay\_comp(U_k,n) * aw^t\ (U_k,n\text{-}1)$ (67)

另一方面，如果 $agg\_delay\_comp(U_k,n)=1$，则该过程计算：

$aw^t\ (U_k,n) = \alpha(U_k,n_{prev}(n))*w(U_k)$ (68)

此处，$n_{prev,k}(n)$ 是当 $ASR(U_k,n_{prev}(n))$ 受到监视（并且因此 $\alpha(U_k,n_{prev}(n))$ 被计算）时在 n 之上或之前的最后一个时隙。该过程继续按下式为此用户计算最终的自适应权值：

$aw(U_k,n) = Z(U_k,n) * aw^t\ (U_k,n)$ (69)

此处，如果用户 U_k 任何延迟敏感流程的 RTx 或 DARQ 队列中没有任何分组，则 $Z(U_k,n)=1$。否则，$Z(U_k,n)=C(U_k)$。此处，$C(U_k)$是预先指定的常数。

用户和流程选择方法

该过程为每个在其队列中至少有一个分组的用户计算以下时隙 n 中的度量，

$$Y(U_k,n) = aw(U_k,n) * DRC(U_k,n) / T(U_k,n) \quad (70)$$

此处，$T(U_k,n) = \sum_{fz} T(fz(U_k,n)$ 是用户 U_k 的平均服务速率（即，包括所有对应的流程）。该过程选择具有最大 $Y(U_k,n)$值的用户。一旦用这一调度器选择了一个用户，该过程即为该用户选择一个流，根据每个如下方案进行服务。

考虑分到以下各组中的流程：

组 1：QSG_延迟_抖动。VoIP 流程。

组 2：QSG_延迟_抖动。视频会议流程。

组 3：QSG_延迟_抖动。视频流的流程。

组 4：QSG_速率_平均_延迟。具有速率和平均延迟要求的流程。

组 5：QSG_速率。仅具有速率要求的流程。

在执行本文中所描述的调度算法中，可遵循以下步骤。

步骤 1：考虑在该时隙中所选择用户的所有后备的流程。

步骤 2：考虑该用户对应于组 1 和 2 的流程。选择 HOL 分组已违反其延迟阈值并且最接近该流程延迟边界的一条流程。如果找到流程，即服务此流程。否则，前进至步骤 3。

步骤 3：考虑对应于组 3 的且其中 HOL 分组已超过延迟阈值的流程，并选择其中 HOL 分组最接近延迟边界的流程。如果找到流程，即服务此流程。否则，前进至下一个步骤。

步骤 4：考虑对应于组 4 的且其中 HOL 分组已超过延迟阈值的流程，并选择其中 HOL 分组最接近延迟边界的流程。如果找到流程，即服务此流程。否则，前进至下一个步骤。

步骤 5：从组 1 到 4 中选出一个后备流程来服务。给予具有最小组号的流程以优先权。如果已选择了流程，则服务它。否则，前进至下一个步骤。

步骤 6：考虑该用户对应于组 5 的后备流程。选择具有最大所请求速率/服务速率值的那一个。服务此流程。如果没有任何一个被选择，则前进至下一个步骤。

步骤 7：服务该用户的尽力服务型流程。如果有一个以上，选择具有最小服务速率值的那一个。

图 16 示出一种应用每条流程和每个用户的补偿的两级调度器。图 16 和图 17 中所示的调度器是用于如本文中所述的用户间和用户内 QoS 补偿的分层结构的调度器。如图 16 中所示，表示为等级 1 的第一级包括多个调度元件或节点 S1、S2、……、SM，其中每个节点处理一个不同的 QSG。在此例中，M 是要处理的 QSG 分组数。例如，调度节点 S1 处理 IP 电话（VoIP）类型的应用程序流程。尽管给出 IP 电话作为示例，但是任何分类成 QSG 1 的应用程序流程将在 S1 处得到处理。这些流程具有指定用于评估 QoS 要求的延迟和抖动边界。多个应用程序 IP 电话类型的流程在调度元件 S1 处得到处理。类似地，每个调度元件为一个指定的 QSG 处理流程。注意，替换实施例可向一个调度元件提供多个 QSG 的应用程序流程。注意，处理同一个 QSG 组可使用多个调度元件。

图 16 中所示的调度器的等级 I 计算每条流程的补偿的一部分。每个用户的多个应用程序流程在图 16 中示出。等级 II 调度元件完成每条流程的补偿的计算。

图 17 示出具有调度节点 S1、S2、……、Sz 的调度器。在此步骤，z 是用户的个数。注意，用户个数是动态的，因此当前调度节点的个数可动态改变。每个调度节点 S1、S2、……、Sz 都适用于从一个给定用户接收多条流程。调度节点 S1 为用户 1（U1）接收流程 F1 到 Fk。这里 k 是当前为用户 1 处理的应用程序流程的总数。使用由图 16 中的等级 I 和等级 II 调度器计算的每条流程的补偿，图 17 中的等级 II 调度器为每个用户计算总用户补偿。等级 II 调度器随即根据以上就用户选择方法所描述的自适应加权 DRC/T 算法来选择在时隙中要服务的用户。等级 II 调度器随后在从等级 I 调度器接收到的加权值之间进行选择。如所指示的，W(Uk)是分配给用户 Uk 的初始权值，ATR(Uk)是用户 Uk 的总目标速率。一旦选择了用户，对应于该用户的等级 I 调度器即根据上述的流程选择方法，为该用户选择该时隙中要服务的流程。

一种前向链路调度器可允许每个用户在一个时隙中在给定的DRC值处为要服务的流程指定承受的价格。一旦指定了价格，自适应帧结构调度器运行以满足不同类型的应用程序的 QoS 要求。对应的调度机制允许服务供应商在增加利益和满足应用程序的 QoS 要求目标之间寻找很好的平衡。该调度机制还提供终端用户的动态成本控制，并可用于具有速率和/或平均延迟要求的应用程序或用于流应用程序，等等。一个实施例提供一种定价选择，其中每条流程指定当其被服务时每个时隙的

价格。此价格依赖于在该时隙中用户为该流程所请求的 DRC 值。流程 *j*（即，访问流程 *j* 的用户）在时隙 m 愿意支付的价格记为 *c[j,m,DRC[j,m]]*。此处 *DRC[j,m]* 表示在时隙 m 中服务此用户的速率。用户可静态地指定价格，诸如为每个 DRC 值预先指定价格。或者，用户可动态地指定价格，诸如在应用程序的生命期期间改变价格。这允许用户对价格有某种程度的控制，以响应于变化的信道情况并达到期望的 QoS。操作者可使用这样的一个调度器连同为用户间和用户内 QoS 所给出的调度器。这允许操作者指定至少两种类型的定价方案。对于用户间和用户内 QoS 调度器，操作员可指定静态定价方案（基于静态服务等级协议），并且在同时允许基于自适应帧结构的调度器的动态定价方案。用户可选择对不同流程使用不同的方案。

一个实施例将时间分成若干个帧，并根据 DRC 值、QoS 要求、QoS 违反统计量和每个用户所指定的价格为每个时隙作调度决策。帧结构基本上给出一轮中应服务的用户队列的次序。网络在每轮调度中决定在给定时隙要为该轮服务哪条流程/用户以达到所期望的目标。帧结构，即在每一轮服务流程的次序，持续变化，并被称为基于 AFS 的算法。

以下定义对计算过程中使用的一些符号进行解释。给定 N 个队列（每条流程一个队列），假设如果以速率 *r[j]* 服务流程 *j*，则其 QoS 要求得到满足。还为每条流程 *j* 预指定初始权值 *w[j]* 和时间标度 *ts[j]*。该过程旨在为流程 *j* 提供速率保证，流程 *j* 在每个时间标度的整数倍的时隙（即，在每个 *m*ts[j]* 时隙，其中 *m* 是整数）受到监视。

令 *start[j]* 为当流程 *j* 最初开始受到考虑要在一轮中被服务的时隙。到时隙 z 结束时，系统希望服务 *S[j,z] = r[j]*(z-start[j])* 个比特，其中对于某个整数 m 有 *z=m*ts[j]*。使用一种调度机制，该系统能够平衡所希望分配给一个给定流程的时隙数和所希望为该流服务的比特数。

此外，其它用于 AFS 调度器的参数如下：

slots_alloc[j,n]：在第 *n* 轮分配给队列（流程）*j* 的时隙数。

slot_served[j,n]：当队列（流程）*j* 在第 *n* 轮得到服务时的时隙数。

S_r[j,n]：到第 *n* 轮结束时为止要为流程 *j* 服务的比特数。

round_len[n]：第 *n* 轮时隙数长度。

round_len_thres：一轮的长度以此阈值为上界。

B[n]：在时隙 *n* 的开始时后备队列的列表。

$R_{out\_round}[j,n]$：由调度器在第 n 轮为队列 j 服务的比特数。

$R_{out}[j,n,g]$：在时间间隔$[n,g]$期间为队列 j 服务的比特数，其中 $g \geq n$。

使用以上给出的描述，在第 n 轮开始队列 j 的亏数比特由下式给出：

$$def\_bits\_r[j,n] = \max(S\_r[j,n-1] - \sum_{k=1}^{n-1} R_{out\_round}[j,k], 0), \forall n, \forall j \qquad (71)$$

当亏数比特为正时，对应的流程在服务中落后，并要得到补偿。另一方面，流程所受到的额外服务不显示地受到处罚，但间接地受到处罚，因为此流程不会得到补偿，而在服务中落后的其它流程将得到补偿。

此外，在第 n 轮的开始流程 j 的归一化亏数比特由下式给出：

$$ndef\_bits\_r[j,n] = \frac{def\_bits\_r[j,n]}{S\_r[j,n-1]}, \forall j, \forall n \text{。} \qquad (72)$$

在第 n 轮的开始队列 j 的亏数时隙由下式给出：

$$def\_slots\_r[j,n] = \max(\sum_{k=1}^{n-1} slots\_alloc[j,k] - \sum_{k=1}^{n-1} slots\_served[j,k], 0), \forall n, \forall j \qquad (73)$$

该过程定义在第 n 轮的开始队列 j 的归一化亏数时隙如下：

$$ndef\_slots\_r[j,n] = \frac{def\_slots\_r[j,n]}{\sum_{k=0}^{n-1} slots\_alloc[j,k]}, \forall j, \forall n \text{。} \qquad (74)$$

令 $lslot[n]$ 为第 n 轮的最后一个时隙，$fslot[n]$ 为第 n 轮的第一个时隙。假设 $aw[j,n]$ 表示第 n 轮分配给流程 j 的（自适应）权值。此权值决定在第 n 轮分配给流程 j 的时隙数。

对用户所请求的 DRC 值进行排序。具体而言，如果 $DRC_1[B,S]$ 优于 $DRC_2[B,S]$，那么 $(B/S)_1 > (B/S)_2$。这里 B 是每个分组的比特数，S 是时隙数。

对于 AFS 调度器的每轮调度，该过程都为每一轮计算以上给出的状态变量，随后在每一轮的开始为每条流程计算权值，以在该轮向此流程分配某个数量的时隙。出于此目的，该过程使用一种自适应权值计算机制，并用一种每一轮的服务规律来为每一轮计算帧结构。

自适应权值计算机制

令 $ndef\_bits\_r_{thres,min}$ 为预先指定的 $ndef\_bits\_r$ 的阈值。该过程在第 n 轮的开始定义一个数组 $ndefbits\_set[n]$ 如下：

$$ndefbits\_set[n] = \{k: ndef\_bits\_r_k \geq ndef\_bits\_r_{thres,min}\} \text{。} \qquad (75)$$

令 $S\_I[n]$ 为在第 n 轮的开始该数组中的流程个数。类似地，$ndef\_slots\_r_{thres,min}$ 是预

先定义的 $ndef\_slots\_r$ 的阈值。该过程定义在第 n 轮的开始的 $ndefslots_set[n]$如下：

$$ndefslot\_set[n] = \{k: ndef\_slots\_r_k \geq ndef\_slots\_r_{thres,min}\}。 \quad (76)$$

令 $S_II[n]$为在第 n 轮的开始此数组中的流程个数。

对于任意的流程 j，在第 n 轮的开始，定义对应的时隙补偿函数如下：

如果 $ndef\_bits\_r_j \geq ndef\_bits\_r_{thres,min}$

则 $slot\_comp[j,n] = slot\_comp\_I[j,n] * slot\_comp\_II[j,n]$， (77)

如果 $ndef\_bits\_r_j < ndef\_bits\_r_{thres,min}$，

则，$slot\_comp[j,n] = 1$。 (78)

此处，

如果 $ndef\_bits\_r_j \geq ndef\_bits\_r_{thres,min}$，

则 $$slot\_comp\_I[j,n] = \frac{ndef\_bits\_r_j}{ndef\_bits\_r_{avg}[n]}, \quad (79)$$

对于所有 k：$k \in ndefbits\_set[n]$， $$ndef\_bits\_r_{avg}[n] = \frac{\sum_k ndef\_bits\_r_j}{S\_I[n]} \quad (80)$$

如果 $ndef\_bits\_r_j < ndef\_bits\_r_{thres,min}$，

则 $slot\_comp\_I[j,n] = 1$。 (81)

对于每条流程 j，定义两个阈值，$slot_comp_I_{thres,min}[j]$和 $slot\_comp\_I_{thres,max}[j]$，以使：

$$slot\_comp\_I_{thres,min}[j] \leq slot\_comp\_I[j,n] \leq slot\_comp\_I_{thres,max}[j], \quad \forall j,\forall n \quad (82)$$

使用这些阈值连通本文所描述的自适应权值计算机制对，可预防任何流程在一轮中不公平地消耗大量时隙，并且同时不对超过给定限制的该流程进行处罚。

如果 $ndef\_slots\_r_j \geq ndef\_slots\_r_{thres,min}$，

则 $$slot\_comp\_II[j,n] = \frac{ndef\_slots\_r_j}{ndef\_slots\_r_{avg}[n]} \quad (83)$$

如果 $ndef\_slots\_r_j < ndef\_slots\_r_{thres,min}$，

则 $slot\_comp\_II[j,n] = 1$。 (84)

对于每条流程 j，定义两个阈值，$slot_comp_II_{thres,min}[j]$和 $slot\_comp\_II_{thres,max}[j]$，以使：

$$slot\_comp\_II_{thres,min}[j] \leq slot\_comp\_II[j,n] \leq slot\_comp\_II_{thres,max}[j], \quad \forall j,\forall n \quad (85)$$

在每一轮的开始，流程被分成如表 5 中所给出的四组。

表 5

组	$ndef\_bits\_r_{flow} \geq$	$ndef\_slots\_r_{flow} \geq$

	$ndef\_bits\_r_{thres,min}$	$ndef\_slots\_r_{thres,min}$
组 I	是	否
组 II	否	是
组 III	是	是
组 IV	否	否

在任意一轮 n 的开始，对于每个属于组 II 或 IV 的流程 j，使用 $slot\_comp[j,n]=1$。在第 n 轮的开始，对于每个属于组 I 的流程 j，应用下式：

$$slot\_comp[j,n] = slot\_comp\_I[j,n]。 \quad (86)$$

如果在第 n 轮的开始流程 j 属于组 III，则应用下式：

$$slot\_comp[j,n] = slot\_comp\_I[j,n] * slot\_comp\_II[j,n] \quad (87)$$

该过程随即计算流程 j 的自适应权值，其中在第 n 轮非空队列的自适应权值如下：

$$aw[j,n] = slot\_comp[j,n] * w[j] * \frac{r[j]}{r_{\min,n}}, \forall j \in B[n] \quad (88)$$

此处，$r_{\min,n} = \min\{r[k]; k : k \in B[n]\}$。对于每条流程 j，定义阈值 $aw_{thres,max}[j]$ 并用这个阈值确保：

$$aw[j,n] \le aw_{thres,max}[j], \ \forall j, \forall n。 \quad (89)$$

接下来，这些自适应权值被应用于计算将分配给每条流程的时隙数和每轮的长度，以使：

$$sloc\_alloc[j,n] \propto aw[j,n], \ \forall j, j : j \in B[n] \quad (90)$$

$$\sum_{j:j\in B[n]} slot\_alloc[j,n] \le round\_len[n] \le round\_len\_thres, \forall n。 \quad (91)$$

每轮的调度规律

一旦已分配好每条流程，则已计算了一轮中的若干时隙和该轮的长度。下一步是要选择在任何给定时隙中要服务的流程。在第 n 轮的任意给定时隙 m 中，如果在先前的时隙之一所选择的分组仍在受到服务，则无需选择要服务的新流程。另一方面，如果在第 n 轮的此时隙 m 中，没有任何分组正在受到服务，则以如下方式选择要服务的流程。对于调度器需要选择要服务的新流程的每个时隙 m，及对于每个流程 j，计算以下流程 j 的第 n 轮的时隙 m 的选择度量，

$$YY[j,n,m] = c[j,m,DRC[j,m]] * wait\_comp[j,n,m] * DRC[j,m], \ \forall n, \quad (92)$$

以使 $j \in B[n]$ 和 $slots\_served[j,n] < \theta(j) * slots\_alloc[j,n]$，此处，$\theta(j)$ 是为每条流程 j 预先指定的，$wait\_comp$ 是给予流程以改进其延迟边界的等待补偿。令 $wait[j,n,m]$

为在第 n 轮的时隙 m 的开始流程 j 的排头分组的等待时间。

$$wait\_comp[j,n,m] = \frac{wait[j,n,m]}{wait\_avg[n,m]}, \forall j, j: j \in B_j \quad (93)$$

且在第 n 轮的时隙 m 的开始，流程 j 至少有一个分组未决。接下来，为流程 k 计算：

$$wait\_avg[n,m] = \frac{\sum_k wait[k,n,m]}{wait\_num[n,m]}, \quad (94)$$

以使 $k \in B[n]$，且这些流程的个数为 $wait\_num[n,m]$。对于每条流程 j，对 $wait_comp_{thres,min}[j]$和 $wait_comp_{thres,max}[j]$这两个阈值进行赋值，并用于确保

$$wait\_comp_{thres,min}[j] \le wait\_comp[j,n,m] \le wait\_comp_{thres,max}[j], \quad \forall m, \forall n, \forall j \text{。} \quad (95)$$

具有选择度量 YY 的最大值的流程被选择由此 AFS 调度器在任何给定时隙中进行服务。

根据一个实施例的 AN 元件在图 20 中示出。AN 元件 500 接收应用程序流程数据并处理该数据以供向用户传输。AN 元件 500 对多个应用程序流程进行调度，其中每条流程都具有 QoS 要求。注意，如上文所述，应用程序流程可包括尽力服务型流程。AN 元件 500 包括流程分类单元 502，适用于识别与流程相关联通信量概况和 QoS 概况、并将流程映射到类（或 QSG）流程。流程分类器单元 502 耦合到调度器 504、允许进入控制单元 510 和 QoS 监视器 506。调度器 504 可实现各种调度算法中的任何一种，包括，但不限于，比例公平（PF）算法和自适应加权 PF 算法。允许进入控制单元 510 将一种允许进入控制方案应用到 AN 500 所接收到的应用程序流程。允许进入控制单元 510 基于 QoS 和网络统计量，对所请求的每个新流程进行估值，并确定是否有足够资源可用以支持该新流程。自适应单元耦合到允许进入控制单元，在那里 ASF 得到更新。自适应单元 512 适用于对当前有效的应用程序流程进行先占决策。先占考虑到某一给定流程有关数据率、所用时隙、和其它 QoS 和网络统计量流程的性能。QoS 监视器适用于监视所接收的应用程序流程的 QoS 要求。注意，AN 元件 500 通常接收多条流程，并在它们之间选择以供向用户传输。调度器 504 从允许进入控制单元 510 接收关于新流程是否被允许进入的信息。调度器 504 从 QoS 监视器 506 接收 QoS 统计量和其它信息，其中调度器 504 应用该 QoS 信息来选择传输的流程。

本文中所给出的是在无线通信系统中用于应用程序流程的允许进入控制、先占和调度的方法和设备。允许进入控制考虑新流程所请求的数据率，并将此与可用资源相比较。一旦被允许进入，流程即被提供给调度器，调度器适用于执行每条流

程和每个用户的分析，来选择在每个时隙或指定的时间段中进行传输的用户。

本领域技术人员将能理解，信息和信号可用各种不同的方法和技术来表示。例如，贯穿以上描述可引用的数据、指令、命令、信息、信号、比特、符号、和芯片可由电压、电流、电磁波、磁场或磁粒子、光场或光粒子、或其任意组合来表示。

此外，本领域技术人员还将理解，结合本文中所解释的各个实施例所描述的各种示例性逻辑框、模块、电路、和算法步骤可被实现为电子硬件、计算机软件、或两者的组合。为了清楚地说明硬件和软件的这一可交换性，以上就其功能一般描述了各种示例性组件、框、模块、电路、和步骤。此类功能实现为硬件还是软件是取决于特定的应用程序和制约整个系统的设计限制。本领域技术人员可为每个特定的应用程序以各种方式实现所描述的功能，但此类实现决定不应被解释成致使偏离本发明的范围。

结合本文中所揭示的各个实施例所描述的各种示例性逻辑框、模块、和电路可用通用处理器、数字信号处理器（DSP）、专用集成电路（ASIC）、场可编程门阵列（FPGA）或其它可编程逻辑设备、分立门或晶体管逻辑、分立硬件组件、或其设计成执行本文中所描述的各种功能的任何组合来实现或执行。通用处理器可以是微处理器，但是，替换地，处理器可以是任何常规处理器、控制器、微控制器、或状态机。处理器可实现为若干计算设备的组合，例如 DSP 和微处理器、多个微处理器、结合 DSP 核心的一个或多个微处理器的组合，或任何其它此类配置。

结合本文中所揭示的各个实施例所描述的方法或算法的步骤可直接在硬件中、在处理器执行的软件模块中、或这两者的组合中具体化。软件模块可驻留在 RAM 存储器、闪存、ROM 存储器、EPROM 存储器、EEPROM 存储器、寄存器、硬盘、可移动磁盘、CD-ROM、或现有技术中已知的任何其它形式的存储介质中。示例性存储介质被耦合到处理器，以使处理器可从该存储介质读取信息，并将信息写到该存储介质中。或者，存储介质可集成到处理器中。处理器和存储介质可驻留在 ASIC 中。ASIC 可驻留在用户终端中。或者，处理器和存储介质可作为分立组件驻留在用户终端中。

提供所揭示的实施例的以上描述，以使本领域中任何技术人员能够制造或使用本发明。对于本领域技术人员来说，对这些实施例的各种修改将是显而易见的，并且本文中所定义的普遍原理可适用于其它实施例，而不会偏离本发明的精神和范围。因此，并不试图将本发明限于本文中所示的实施例，而是要使其符合与本文中所揭示的原理和新颖特性相一致的最宽泛的范围。

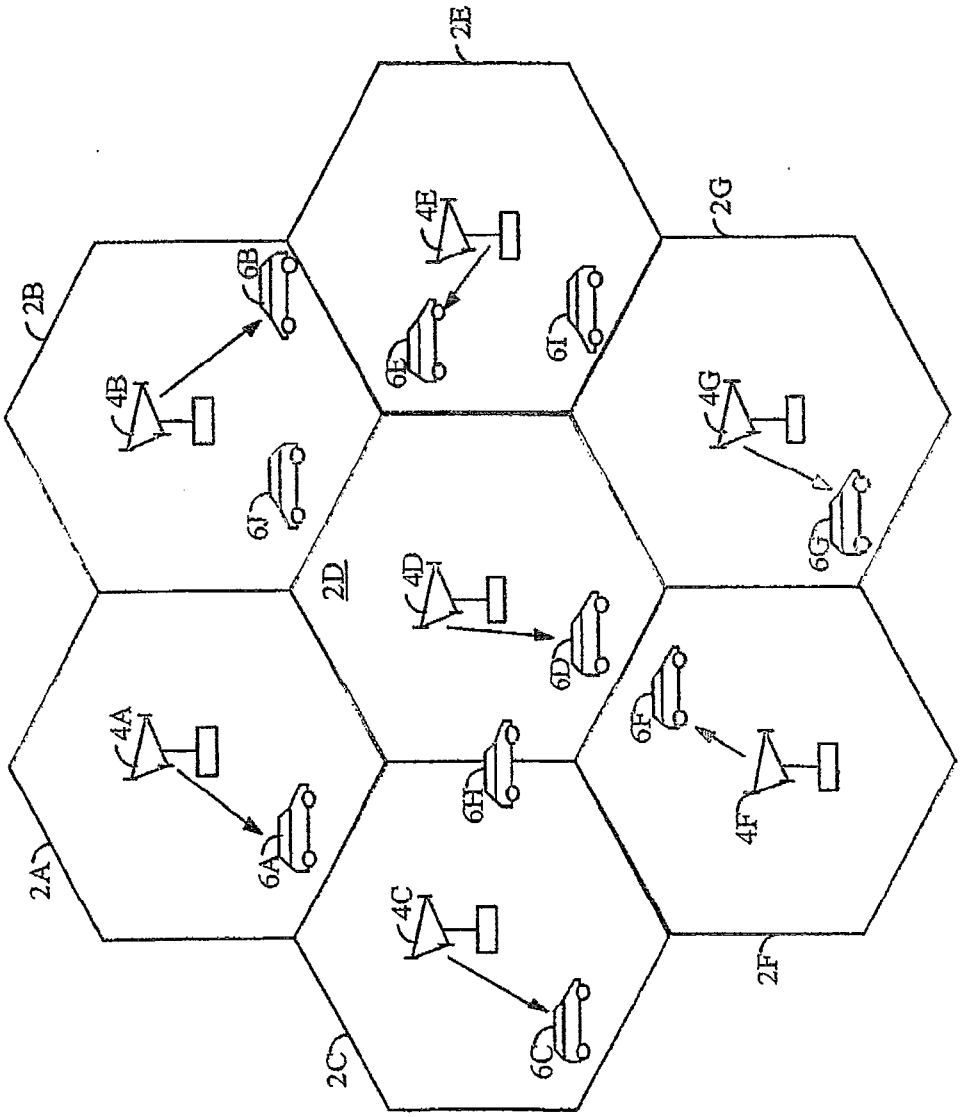


图 1

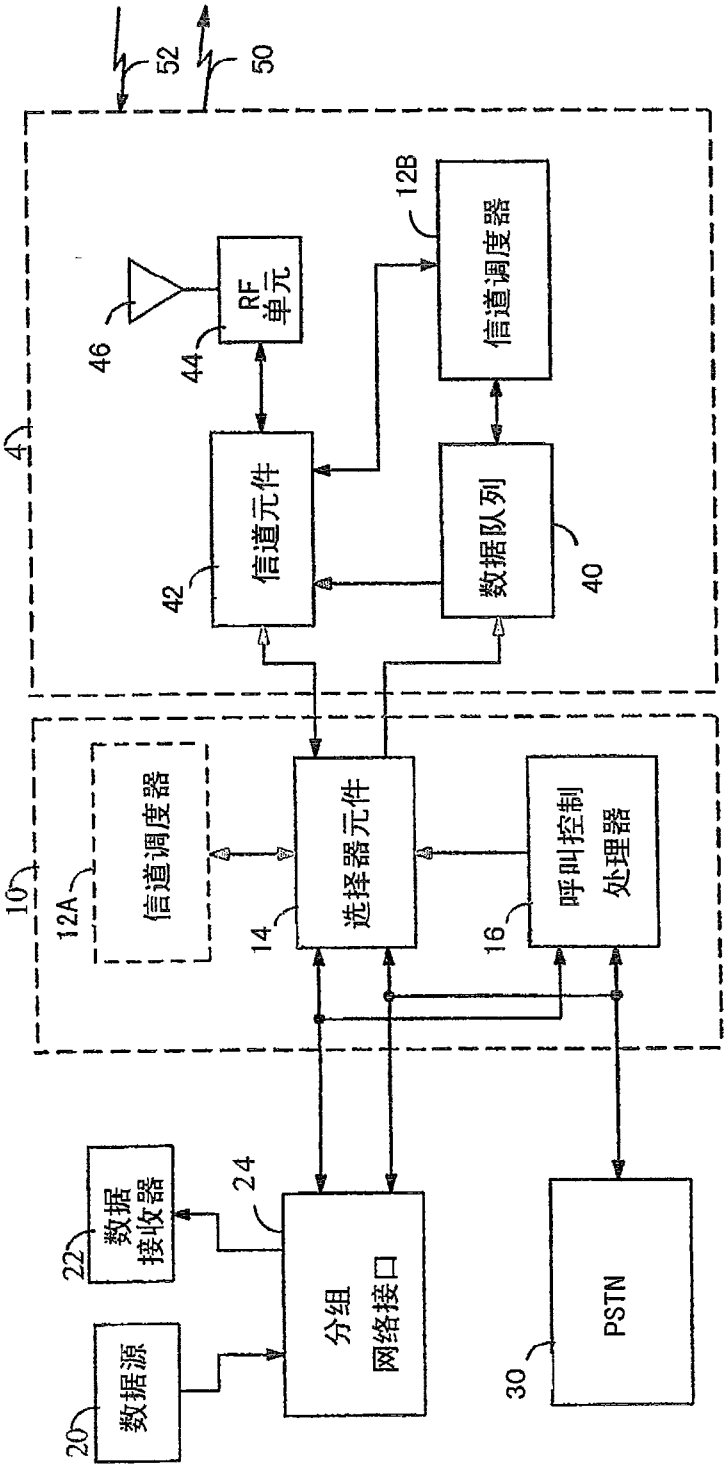

52
50
4
46
44
RF
单元
12B
信道调度器
42
信道元件
数据队列
40
10
12A
信道调度器
14
选择器元件
16
呼叫控制
处理器
22
数据
接收器
24
分组
网络接口
20
数据源
30
PSTN

图 2A

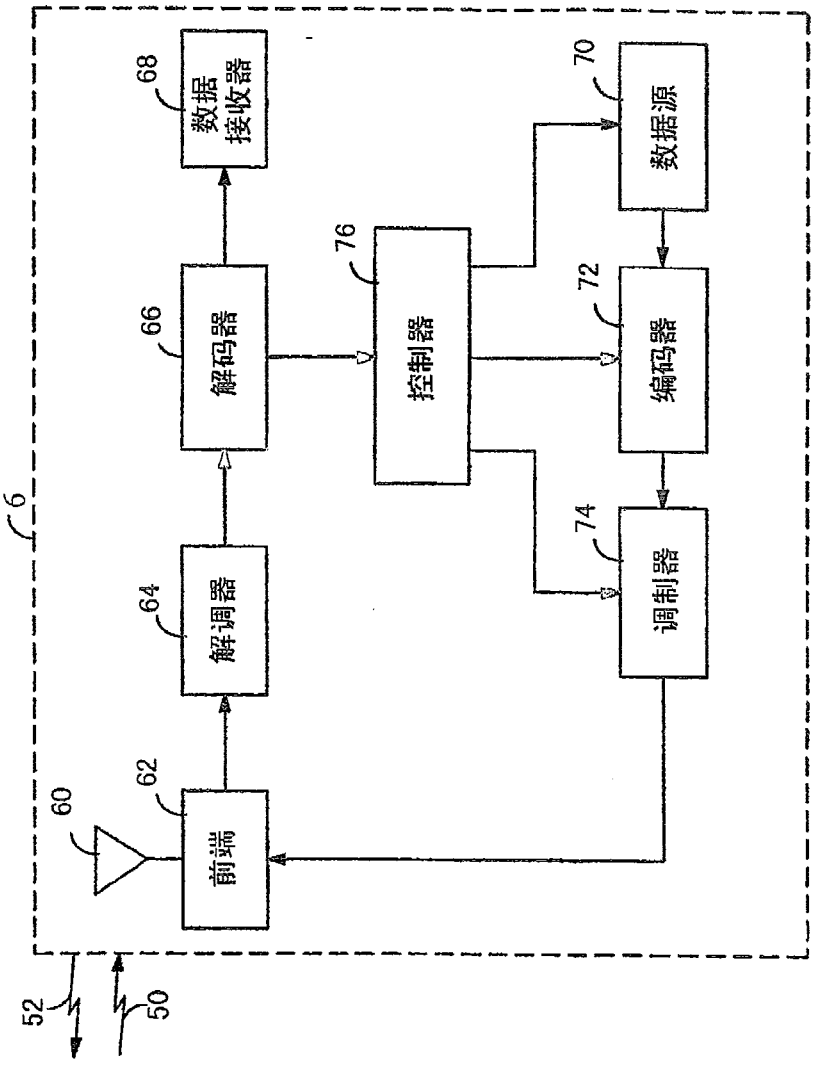


图 2B

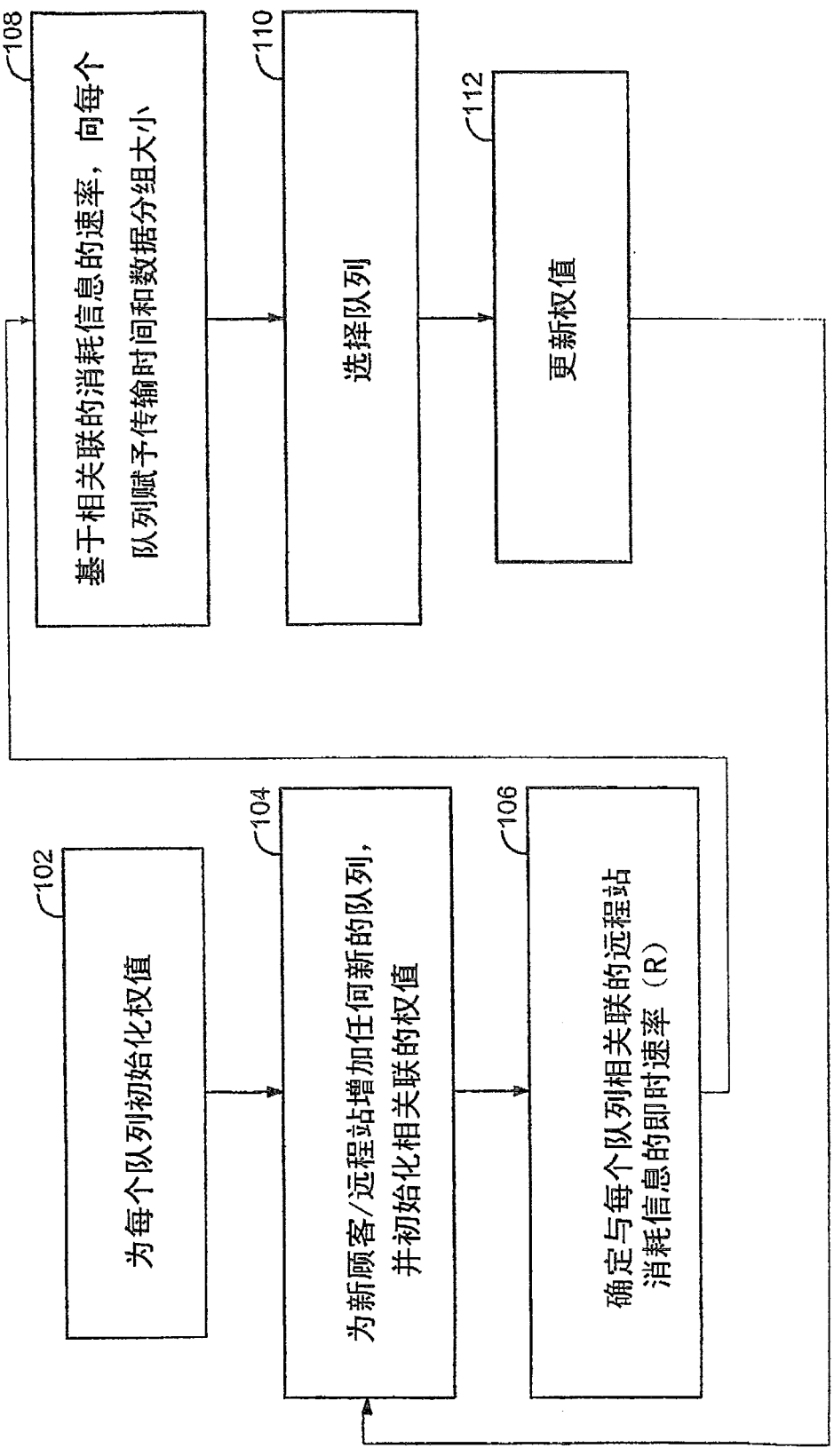

102
为每个队列初始化权值
104
为新顾客/远程站增加任何新的队列，并初始化相关联的权值
106
确定与每个队列相关联的远程站消耗信息的即时速率（R）
108
基于相关联的消耗信息的速率，向每个队列赋予传输时间和数据分组大小
110
选择队列
112
更新权值

图 3

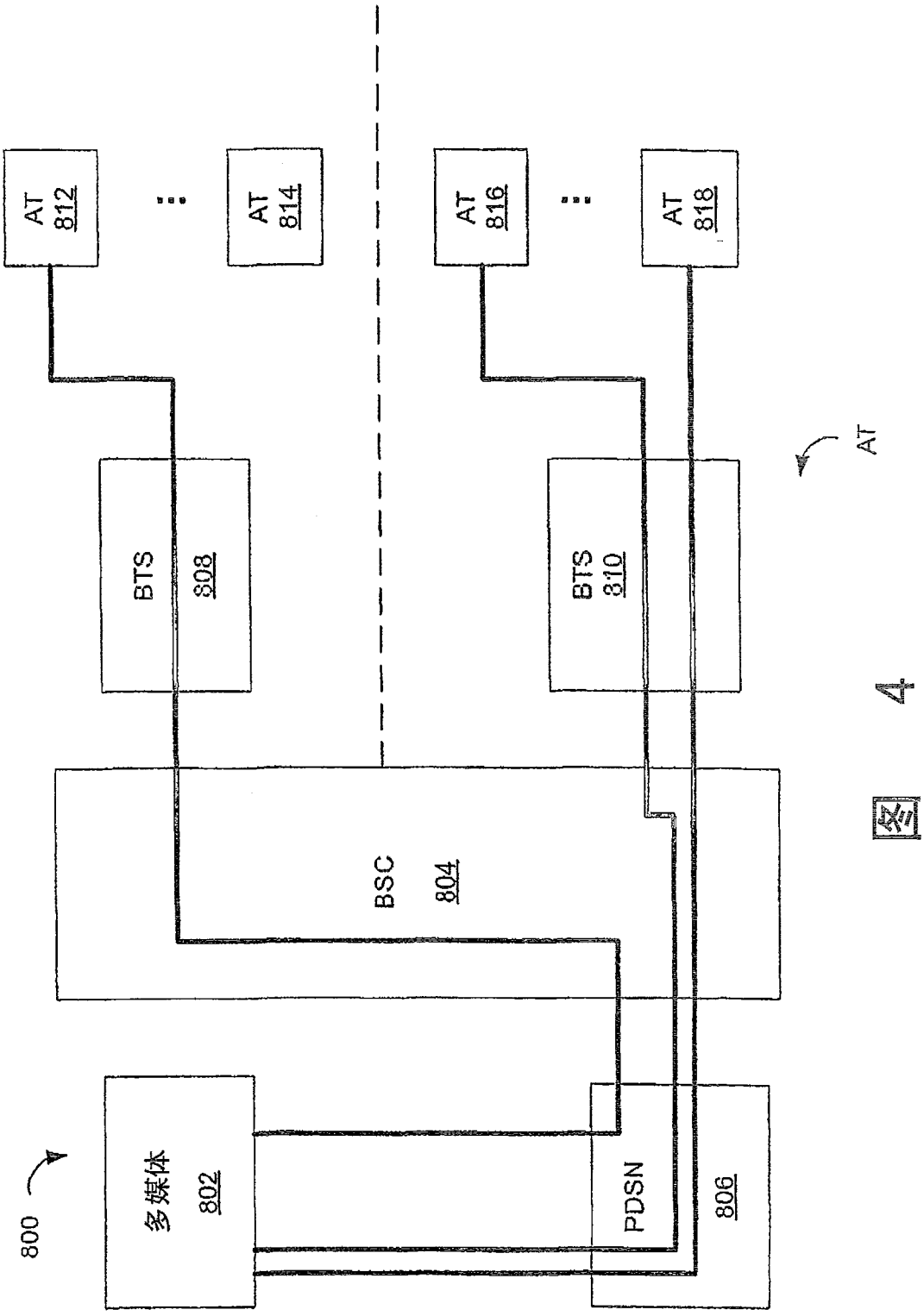

800
多媒体
802
PDSN
806
BSC
804
BTS
808
BTS
810
AT
812
AT
814
AT
816
AT
818
AT

图 4

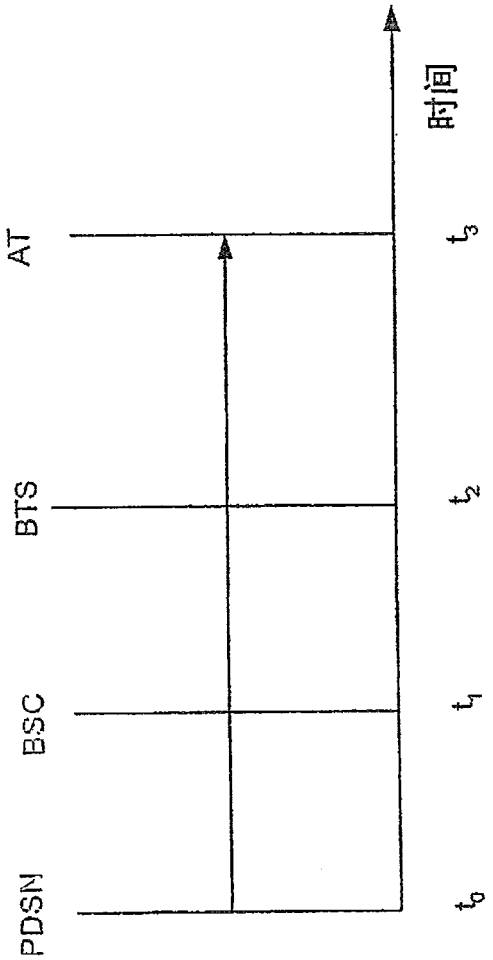


图 6

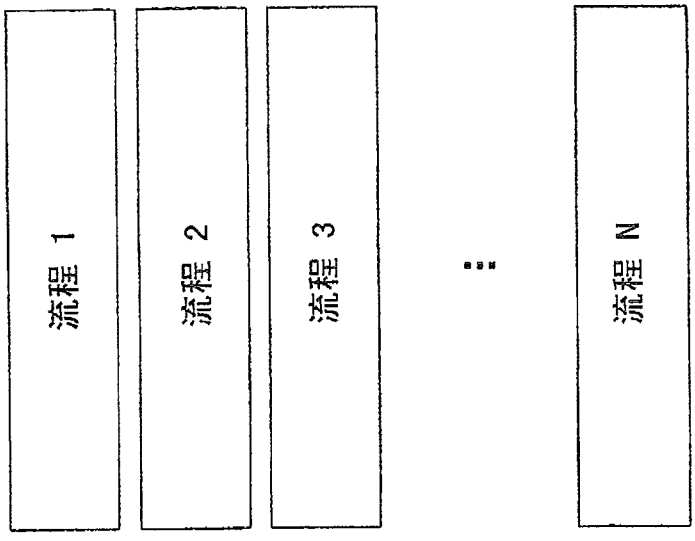


图 5

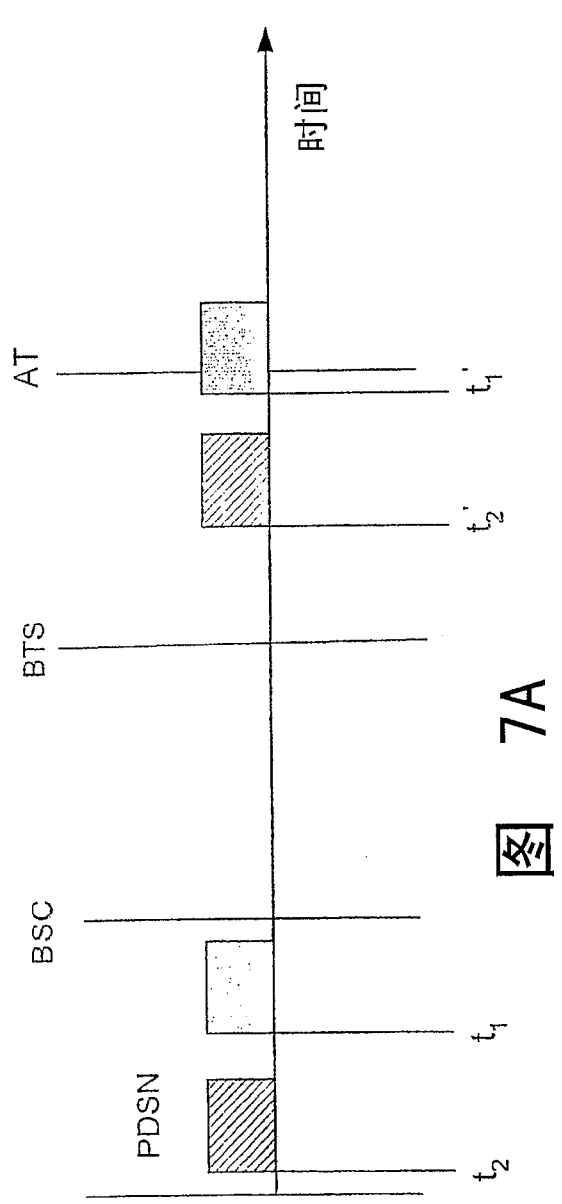

时间
AT
BTS
BSC
PDSN
t_1
t_2
t_1'
t_2'

图 7A

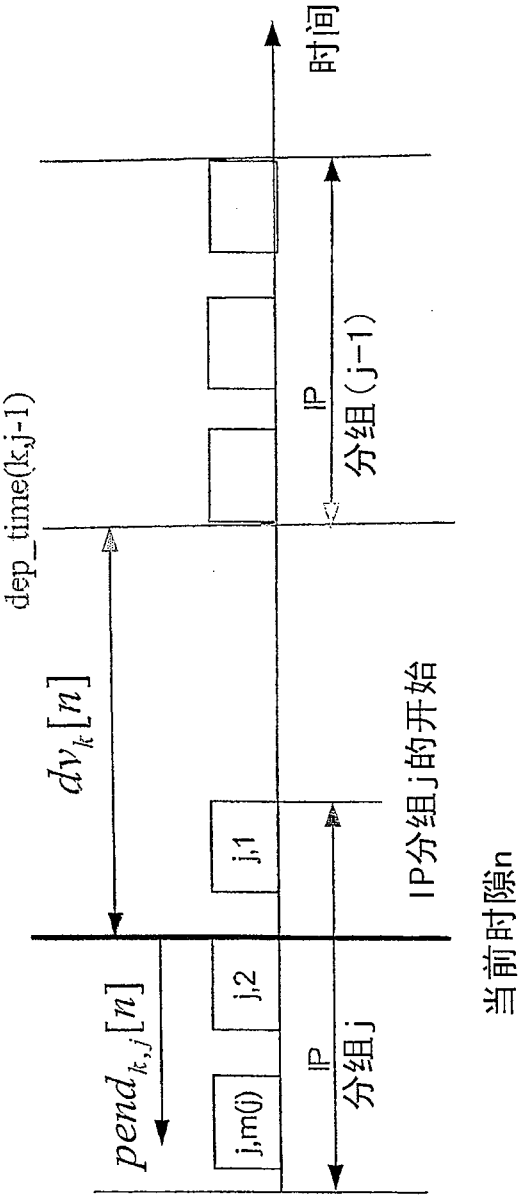

时间
dep_time(k,j-1)
$dv_k[n]$
$pend_{k,j}[n]$
IP
分组(j-1)
IP
分组j
j,1
j,2
j,m(j)
IP分组j的开始
当前时隙n

图 7B

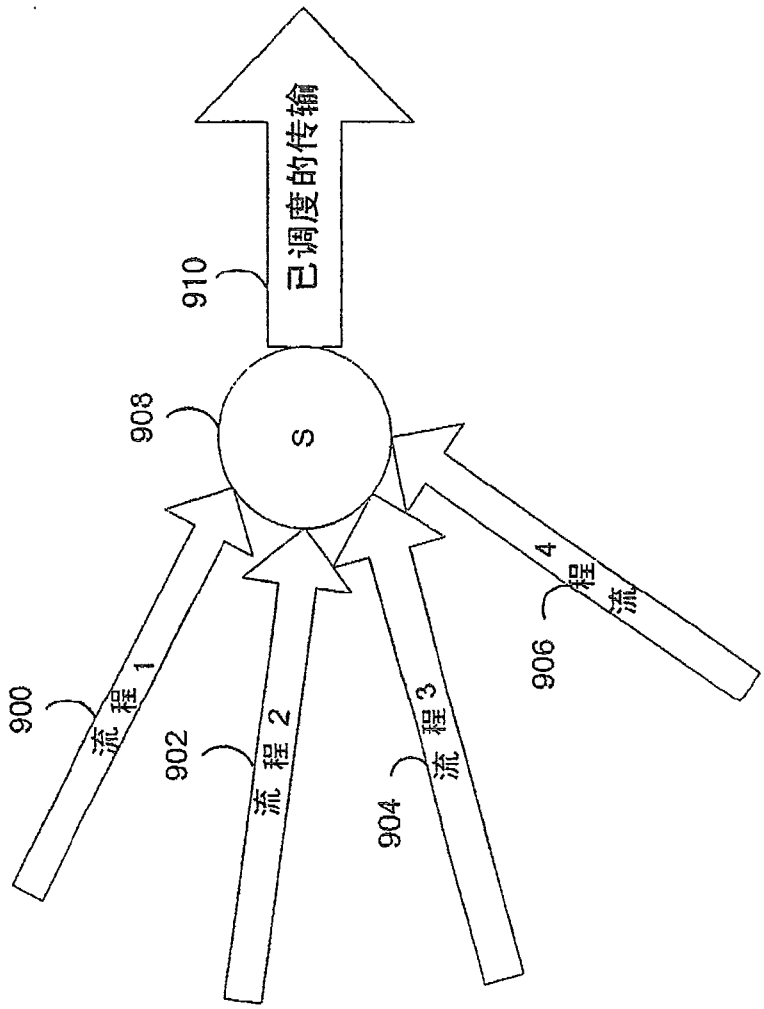


图 8

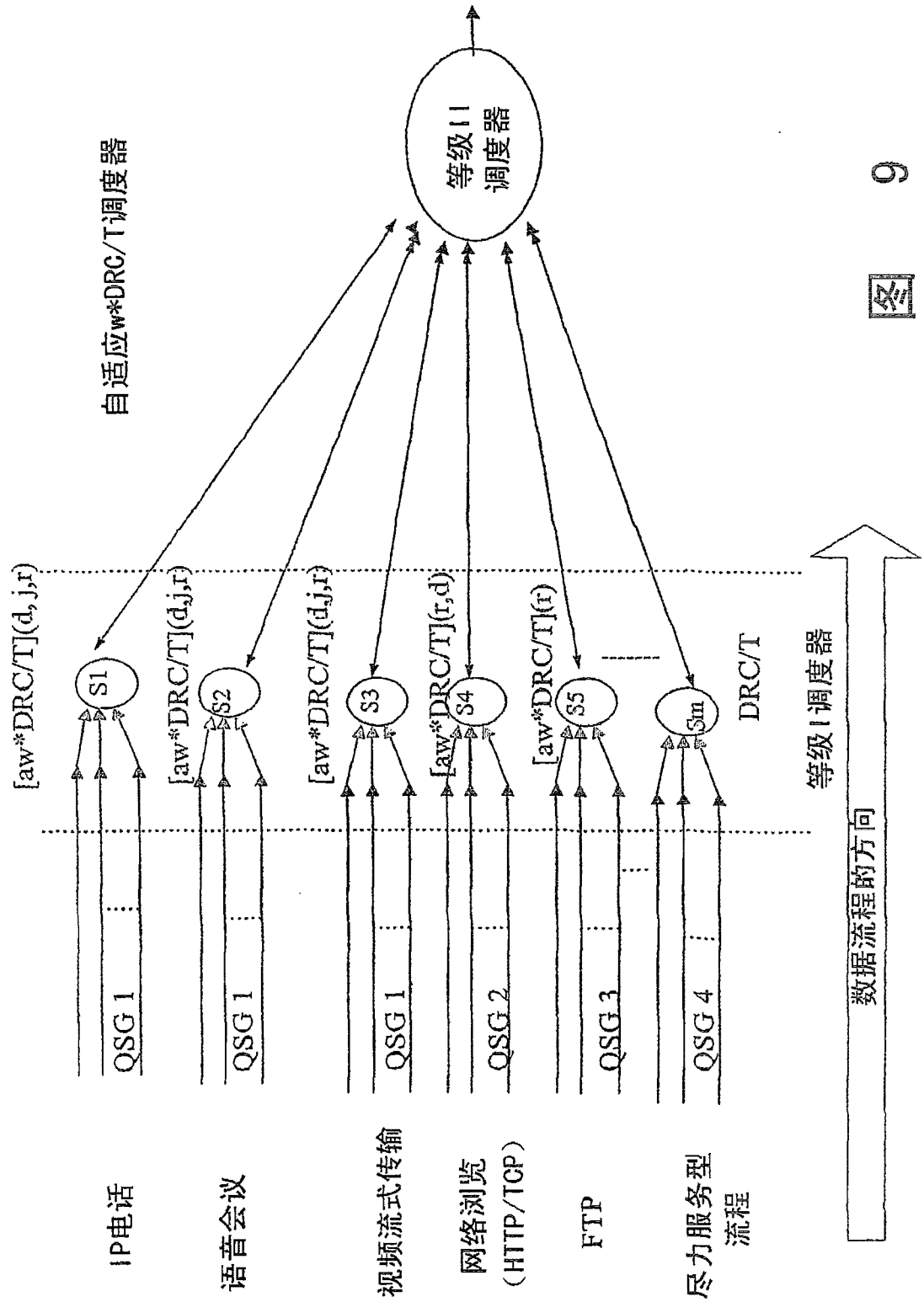

自适应w*DRC/T调度器
等级II
调度器
[aw*DRC/T](d,j,r)
[aw*DRC/T](d,j,r)
[aw*DRC/T](d,j,r)
[aw*DRC/T](r,d)
[aw*DRC/T](r)
S1
S2
S3
S4
S5
Sm
QSG 1
QSG 1
QSG 1
QSG 2
QSG 3
QSG 4
IP电话
语音会议
视频流式传输
网络浏览
（HTTP/TCP）
FTP
尽力服务型
流程
DRC/T
等级I调度器
数据流程的方向

图 9

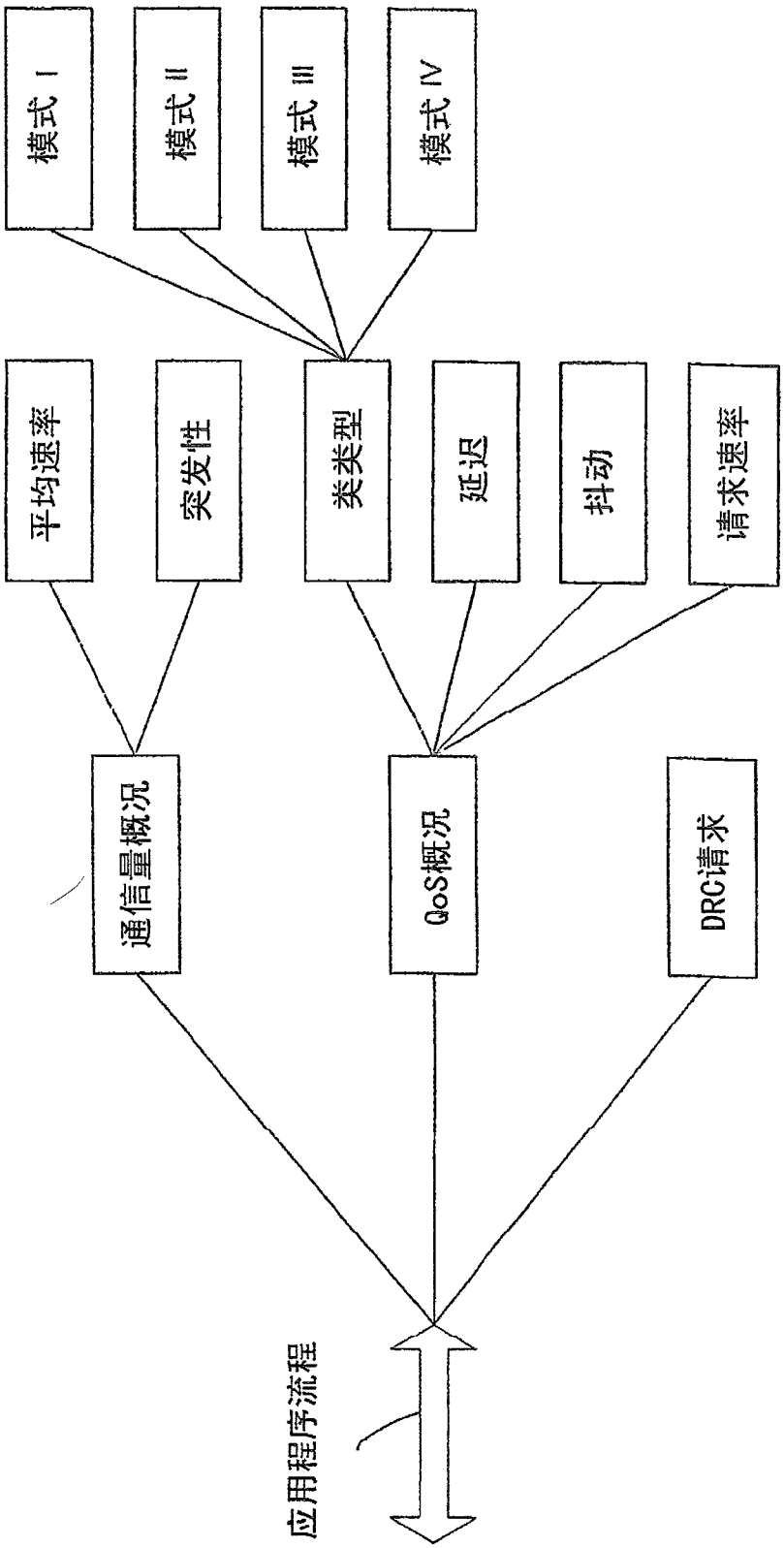

模式 I
模式 II
模式 III
模式 IV
平均速率
突发性
类类型
延迟
抖动
请求速率
通信量概况
QoS概况
DRC请求
应用程序流程

图 10

模式	QoS要求	QoS参数
I	高延迟敏感性	速率、延迟、抖动
II	中等延迟敏感性	速率、平均延迟
III	对速率敏感	速率
IV	无指定QoS	无

图 11

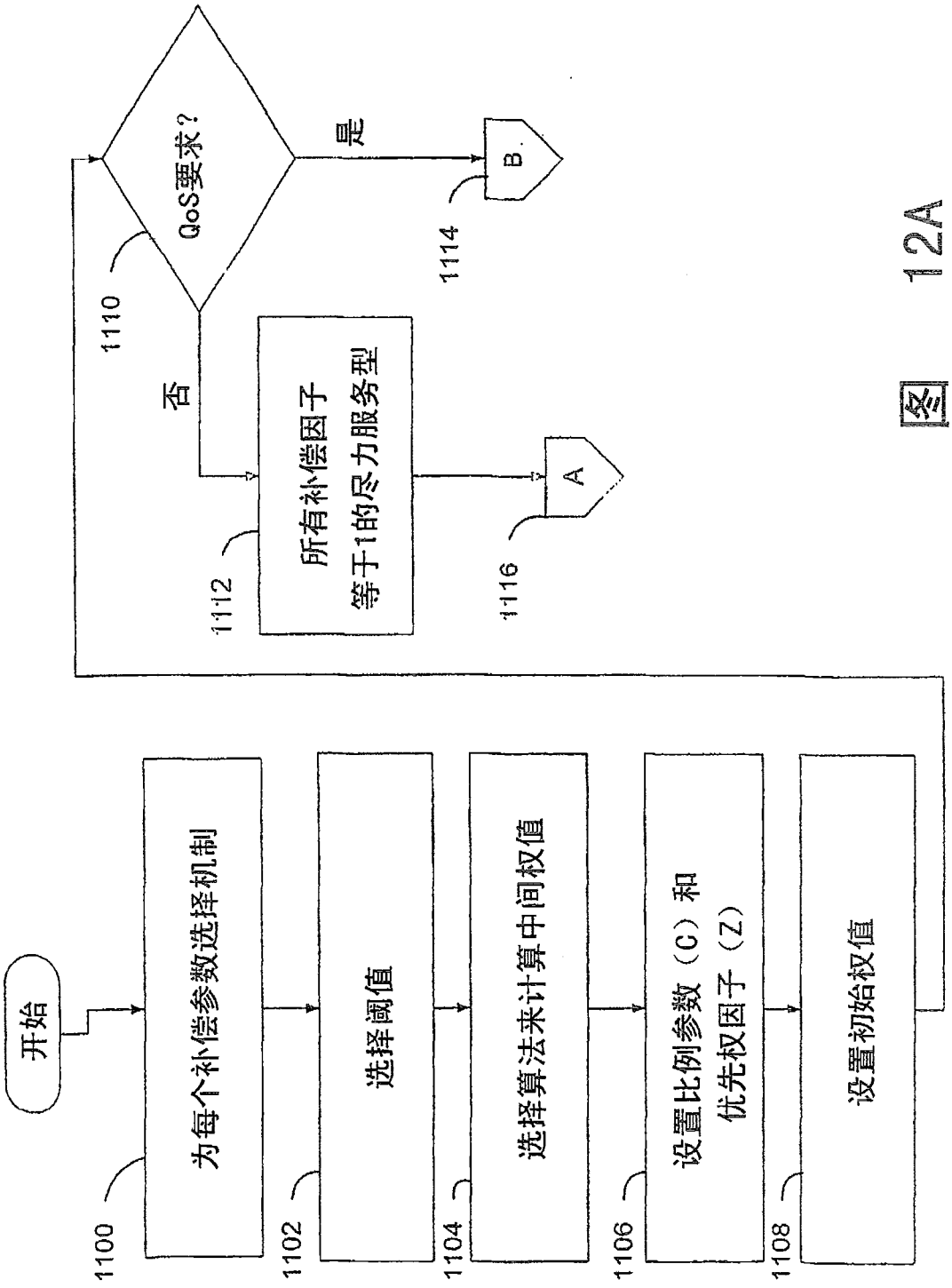

开始
为每个补偿参数选择机制
选择阈值
选择算法来计算中间权值
设置比例参数（C）和
优先权因子（Z）
设置初始权值
QoS要求？
否
是
所有补偿因子
等于1的尽力服务型
A
B
1100
1102
1104
1106
1108
1110
1112
1114
1116

图 12A

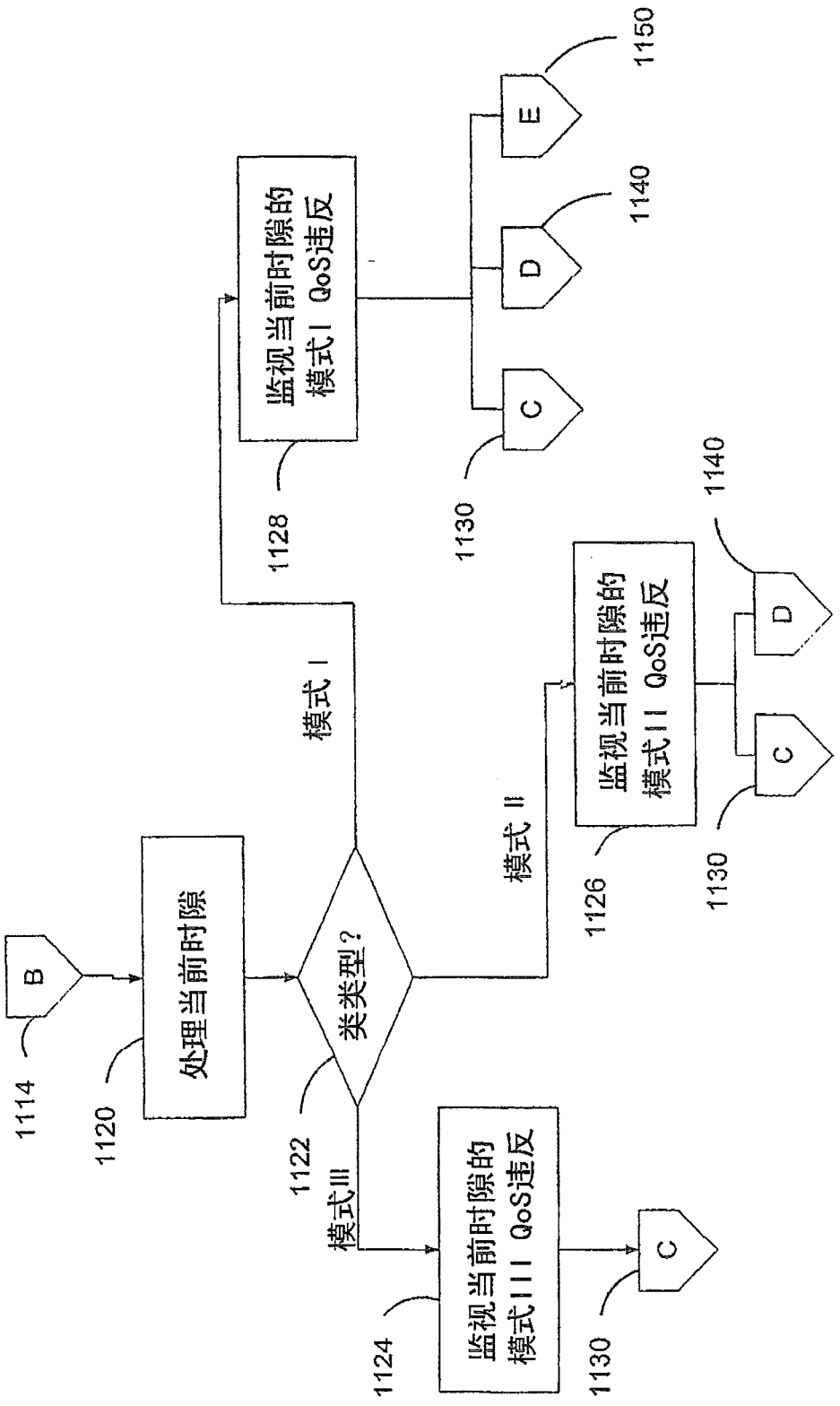

B
1114
处理当前时隙
1120
类类型？
1122
模式 I
模式 II
模式III
监视当前时隙的
模式I QoS违反
1128
监视当前时隙的
模式II QoS违反
1126
监视当前时隙的
模式III QoS违反
1124
C
1130
D
1140
E
1150

图 12B

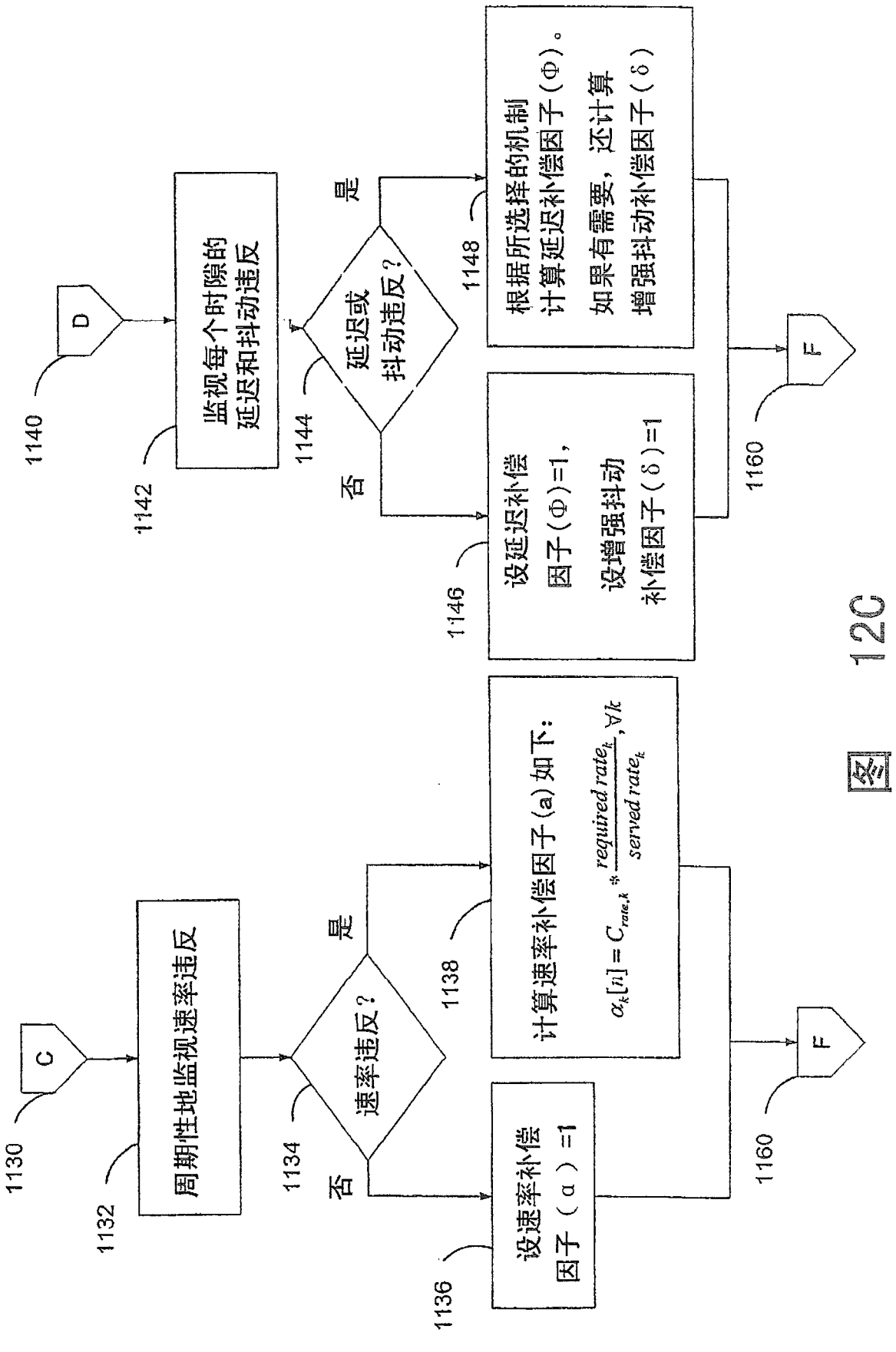

C
1130
周期性地监视速率违反
1132
速率违反？
1134
是
否
计算速率补偿因子(a)如下：
$\alpha_k[i] = C_{rate,k} * \frac{required\ rate_k}{served\ rate_k}, \forall k$
1138
设速率补偿因子（α）=1
1136
F
1160
D
1140
监视每个时隙的延迟和抖动违反
1142
延迟或抖动违反？
1144
是
否
根据所选择的机制计算延迟补偿因子(Φ)。
如果有需要，还计算增强抖动补偿因子(δ)
1148
设延迟补偿因子(Φ)=1，
设增强抖动补偿因子(δ)=1
1146
F
1160

图 12C

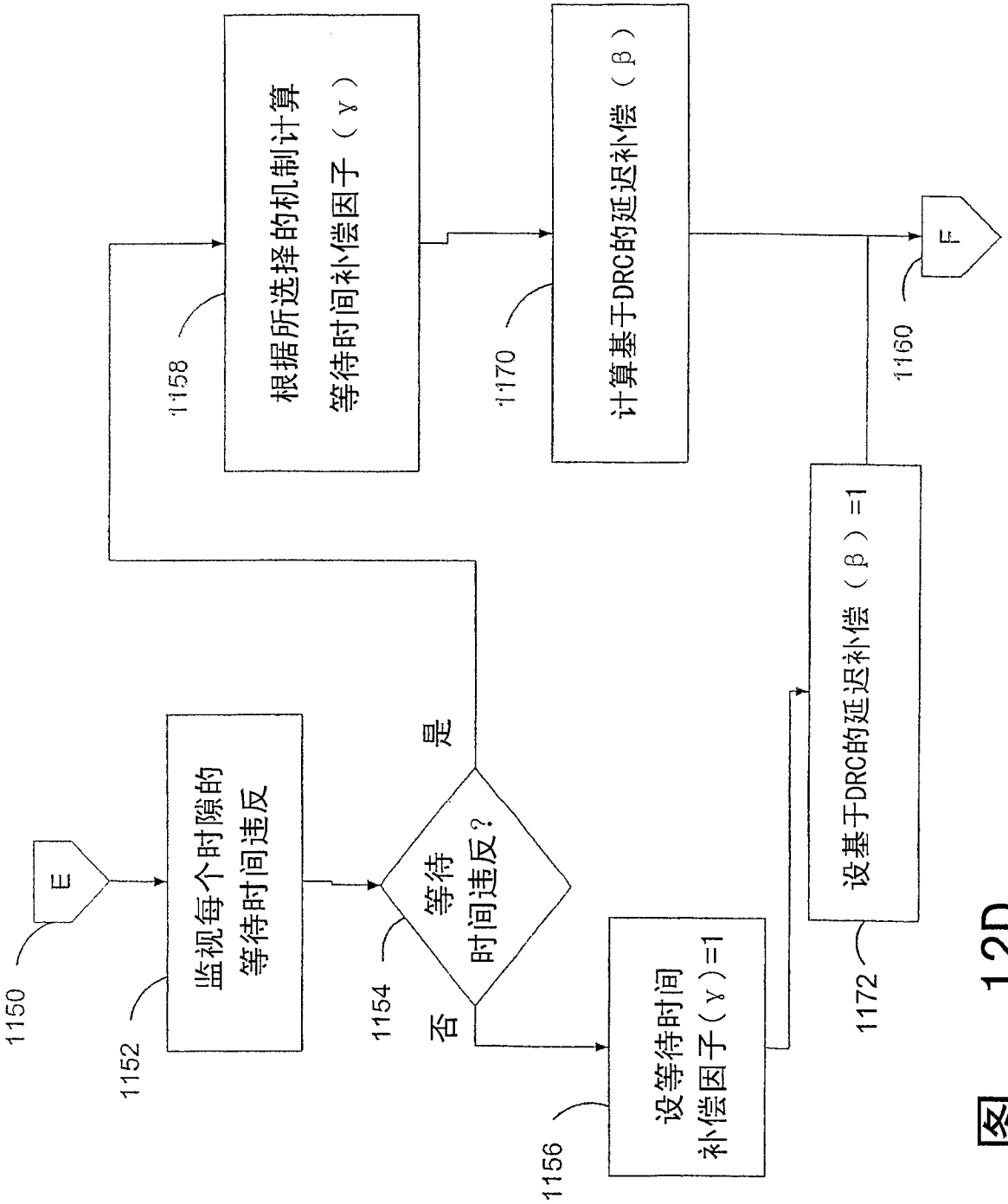

E
1150
监视每个时隙的等待时间违反
1152
等待时间违反?
1154
是
否
根据所选择的机制计算等待时间补偿因子（γ）
1158
设等待时间补偿因子（γ）=1
1156
计算基于DRC的延迟补偿（β）
1170
设基于DRC的延迟补偿（β）=1
1172
F
1160

图 12D

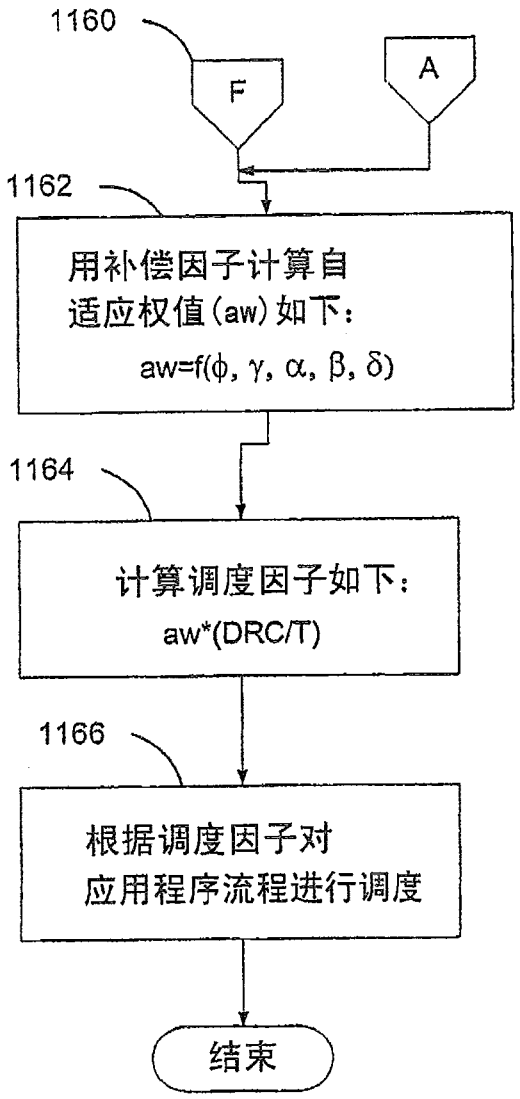

1160
F
A
1162
用补偿因子计算自
适应权值(aw)如下：
aw=f(ϕ, γ, α, β, δ)
1164
计算调度因子如下：
aw*(DRC/T)
1166
根据调度因子对
应用程序流程进行调度
结束

图 12E

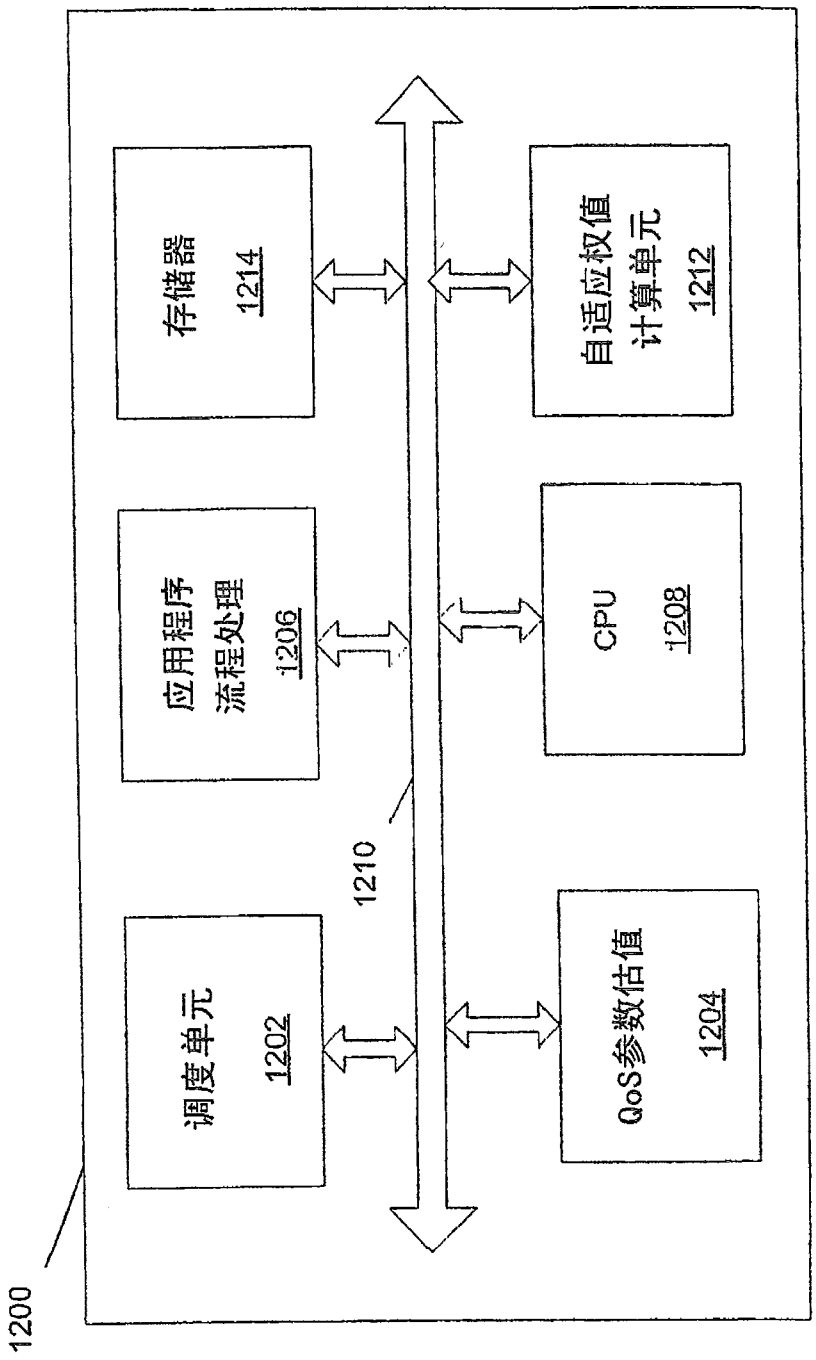

1200
调度单元
1202
应用程序
流程处理
1206
存储器
1214
1210
QoS参数估值
1204
CPU
1208
自适应权值
计算单元
1212

图 13

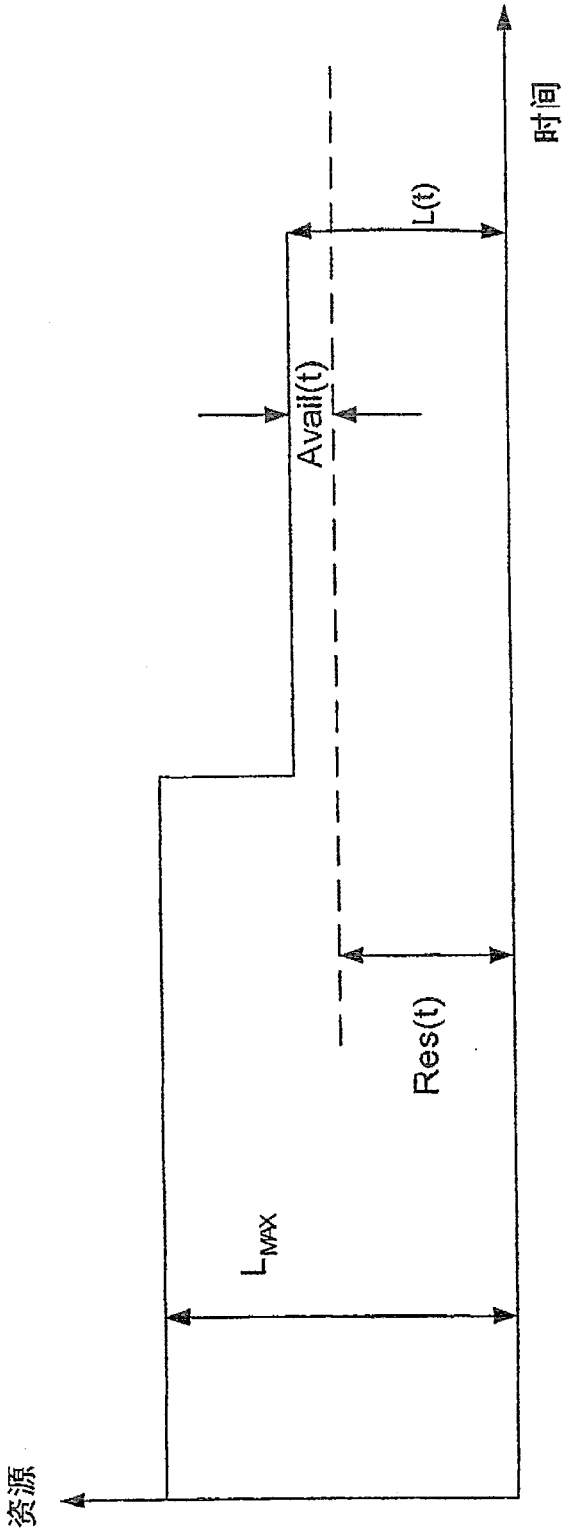

资源
时间
L(t)
Avail(t)
Res(t)
L_{MAX}
$1 \le AS_{min\_prespecified} \le AS(t) \le AS_{max\_prespecified} < \infty$
L(t) = 对希望在时间t保留的带宽的估计
Avail(t)=maximum(L(t) –Res(t), 0)

图 14

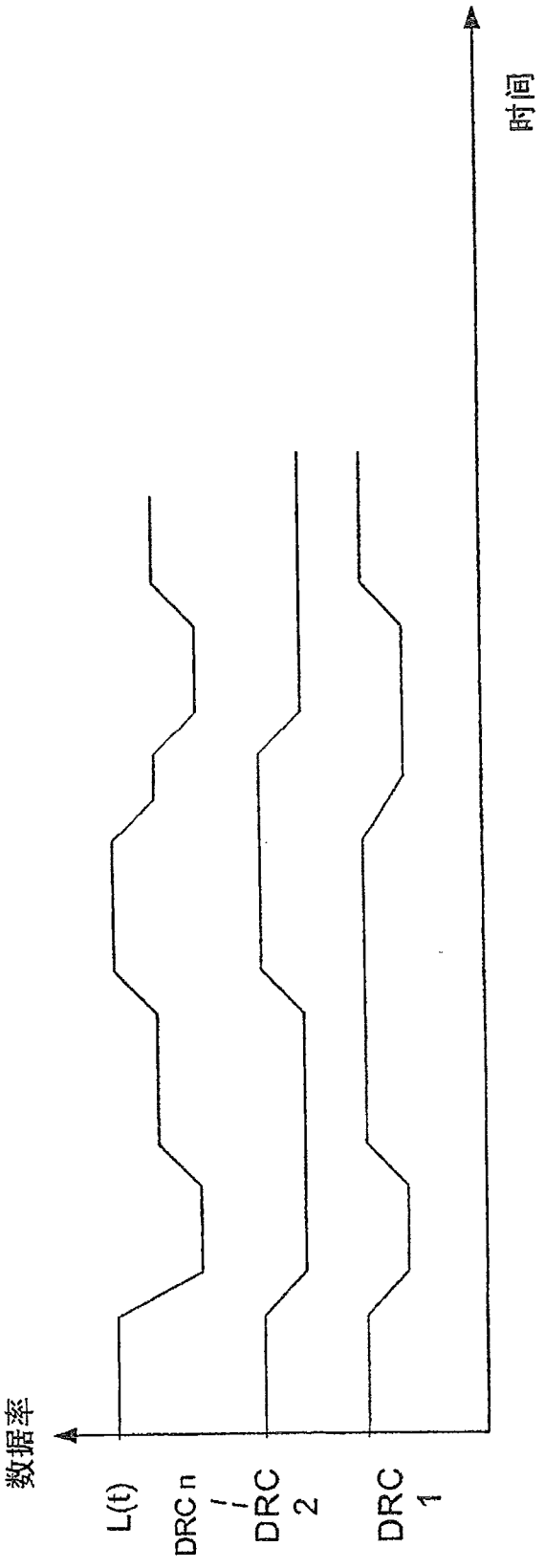


图 15

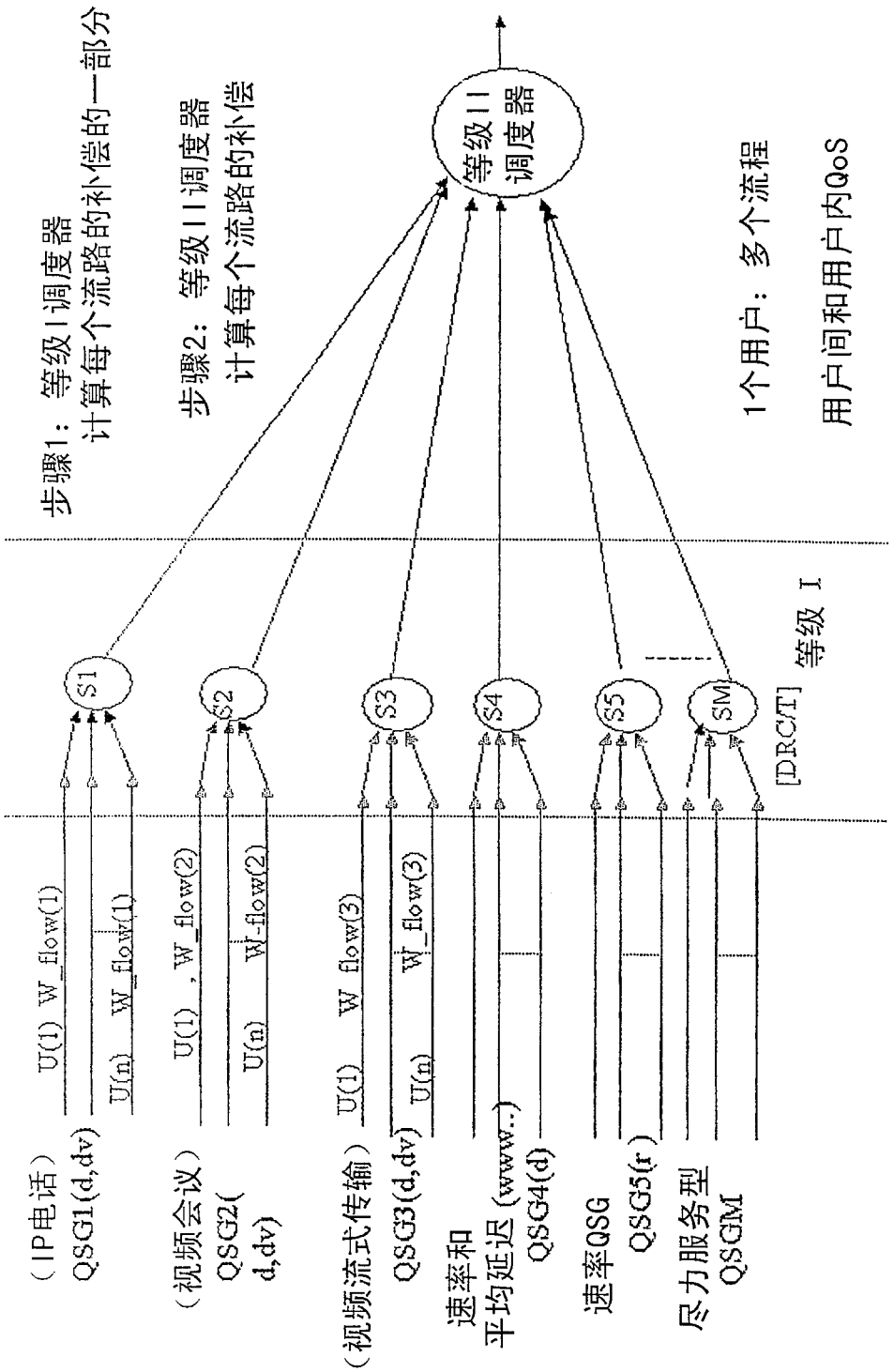

步骤1：等级I调度器
计算每个流路的补偿的一部分
步骤2：等级II调度器
计算每个流路的补偿
1个用户：多个流程
用户间和用户内QoS
[DRC/T]
等级 I
等级II
调度器
S1
S2
S3
S4
S5
SM
（IP电话）
QSG1(d,dv)
（视频会议）
QSG2(
d,dv)
（视频流式传输）
QSG3(d,dv)
速率和
平均延迟 (wwww...)
QSG4(d)
速率QSG
QSG5(r)
尽力服务型
QSGM
U(1) W_flow(1)
U(n) W_flow(1)
U(1) , W_flow(2)
U(n) W-flow(2)
U(1) W flow(3)
U(n) W_flow(3)

图 16

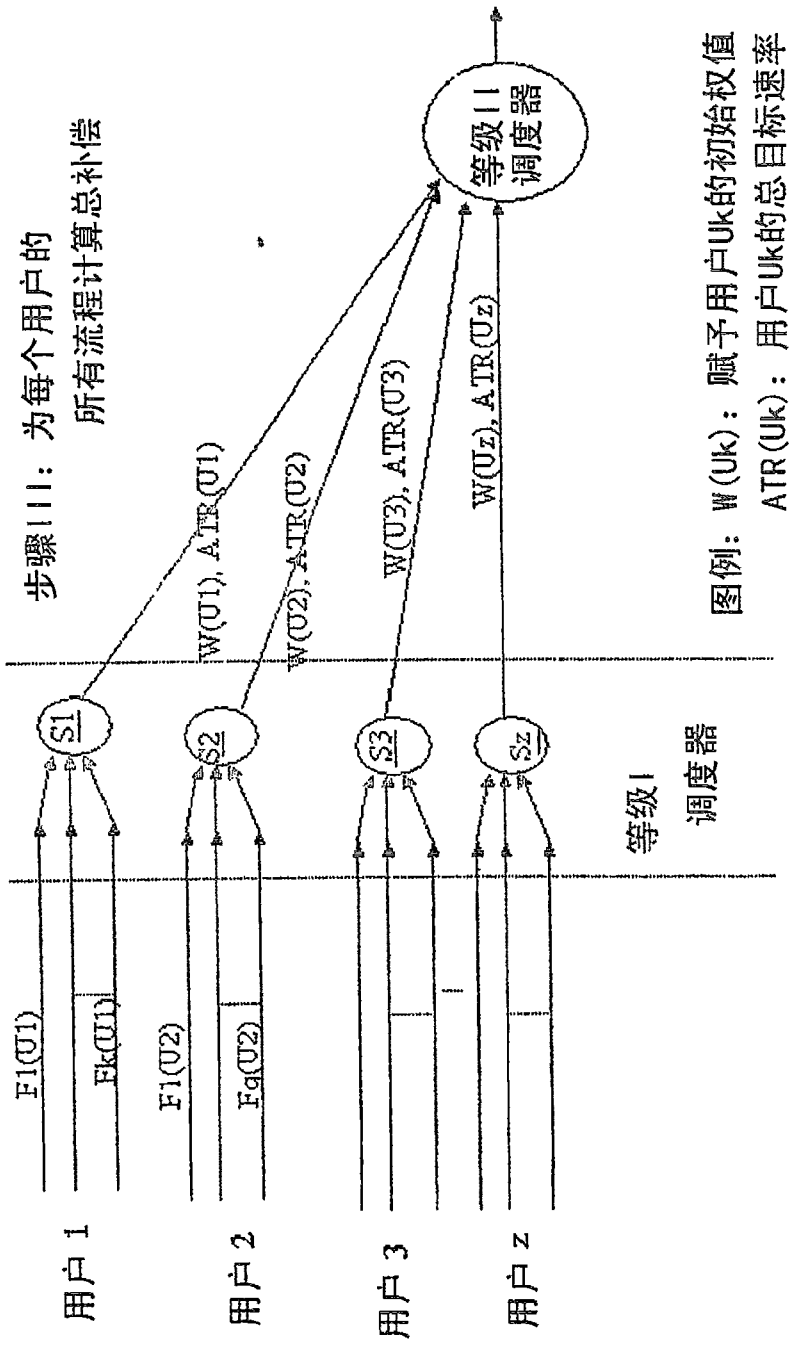

步骤III：为每个用户的
所有流程计算总补偿
W(U1), ATR(U1)
W(U2), ATR(U2)
W(U3), ATR(U3)
W(Uz), ATR(Uz)
等级II
调度器
S1
S2
S3
Sz
等级I
调度器
F1(U1)
Fk(U1)
F1(U2)
Fg(U2)
用户 1
用户 2
用户 3
用户 z
图例：W(Uk)：赋予用户Uk的初始权值
ATR(Uk)：用户Uk的总目标速率

图 17

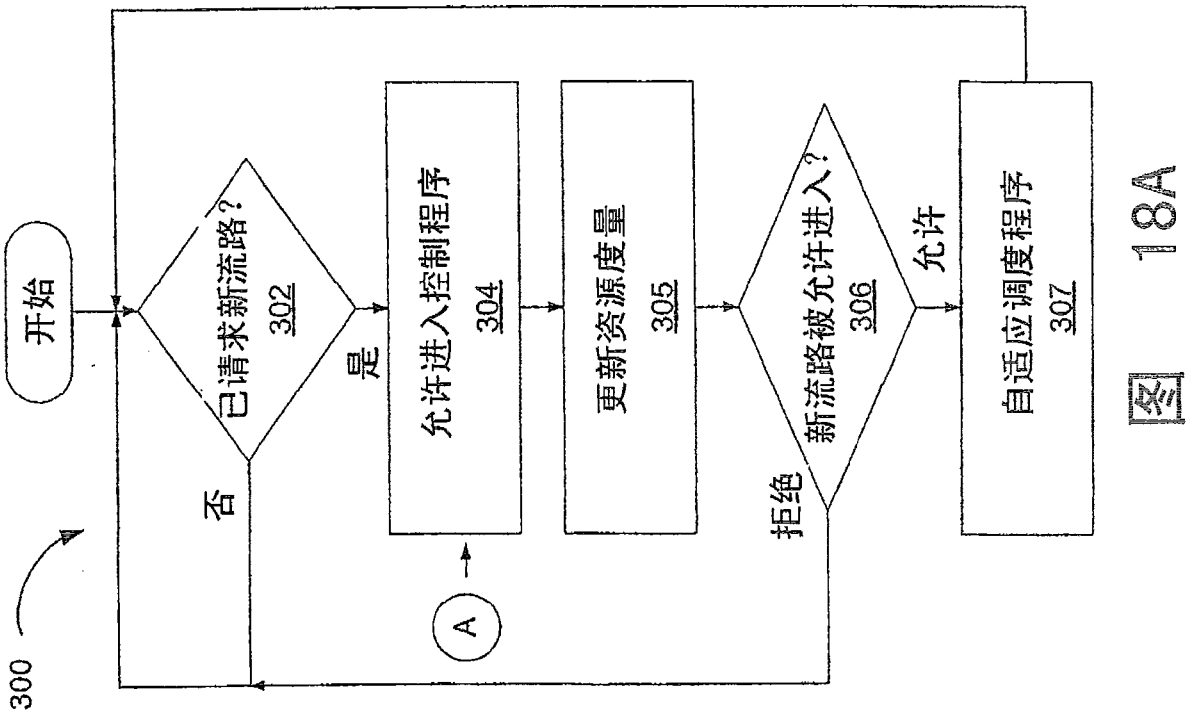

300
开始
已请求新流路?
302
否
是
允许进入控制程序
304
A
更新资源度量
305
新流路被允许进入?
306
拒绝
允许
自适应调度程序
307

图 18A

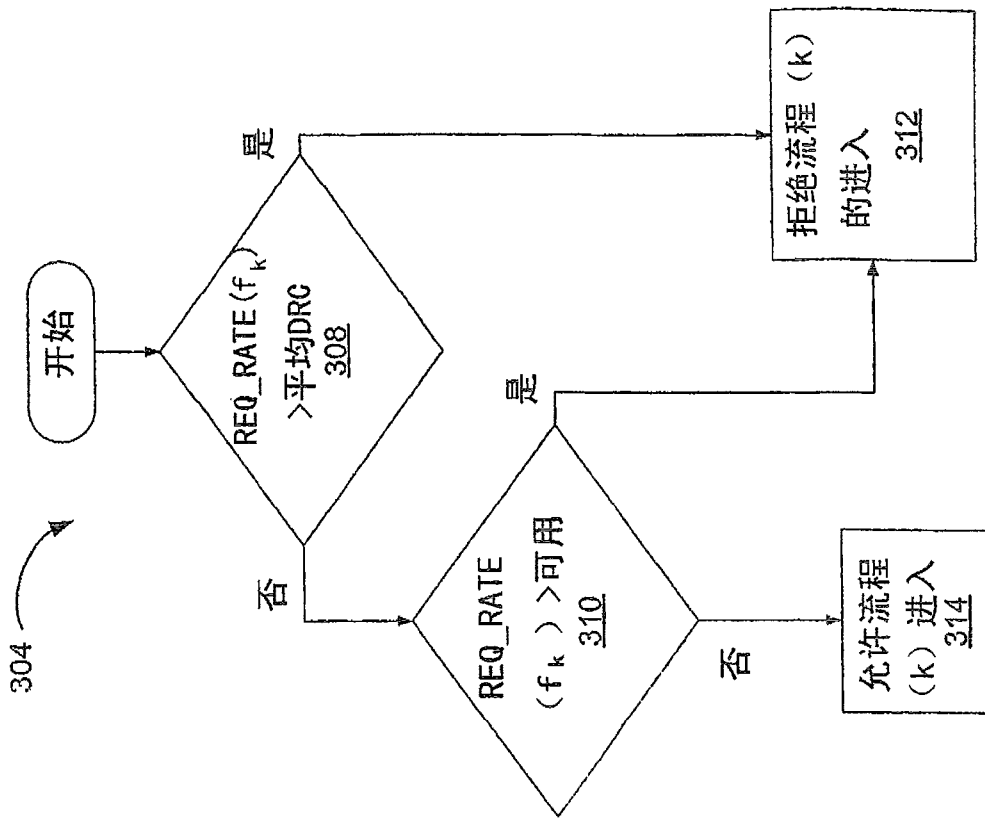

304
开始
REQ_RATE(f_k)
>平均DRC
308
是
否
拒绝流程（k）
的进入
312
REQ_RATE
（f_k）>可用
310
是
否
允许流程
（k）进入
314

图 18B

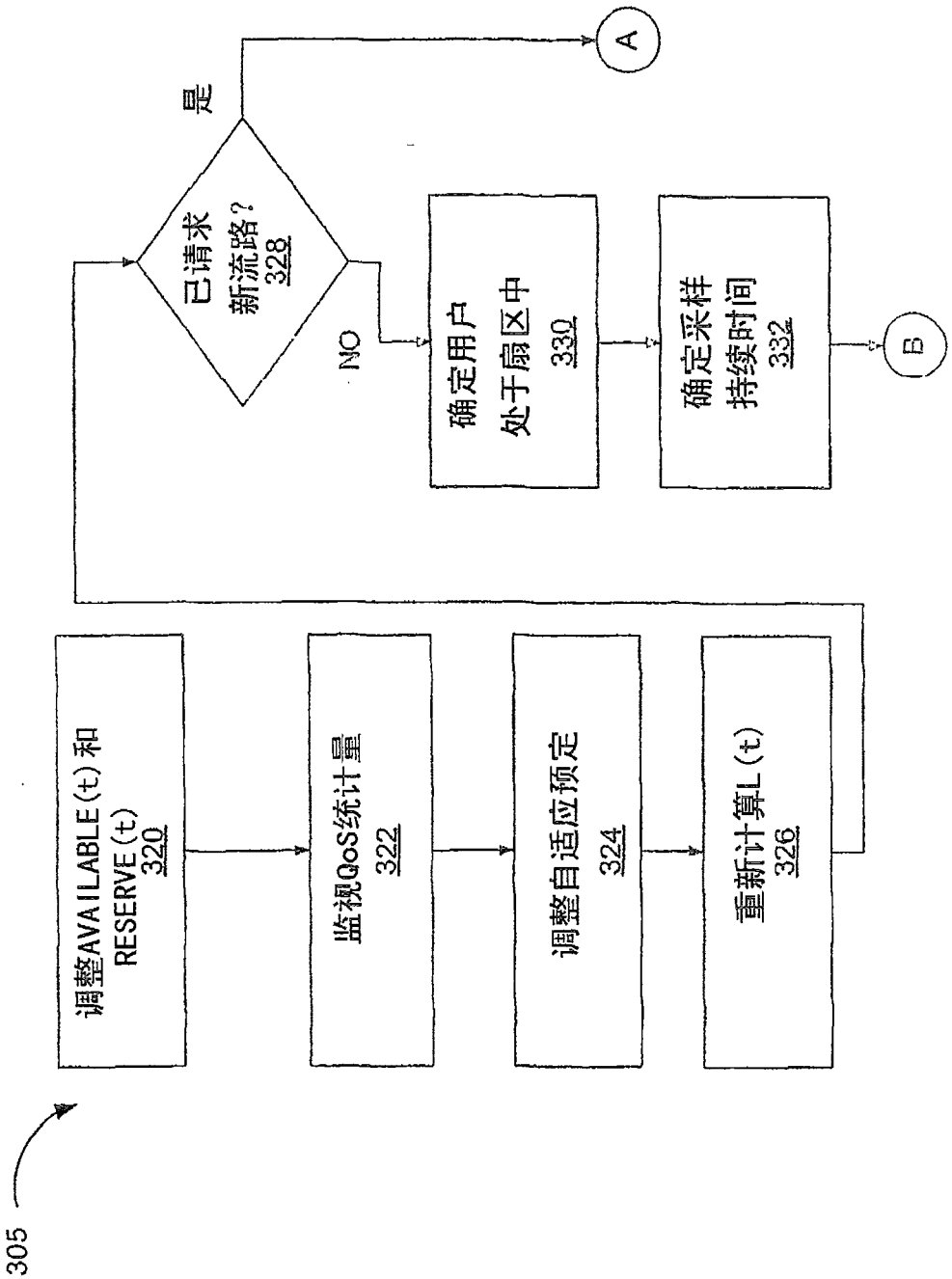

305
调整AVAILABLE(t)和RESERVE(t)
320
监视QoS统计量
322
调整自适应预定
324
重新计算L(t)
326
已请求新流路?
328
是
NO
A
确定用户处于扇区中
330
确定采样持续时间
332
B

图 18C

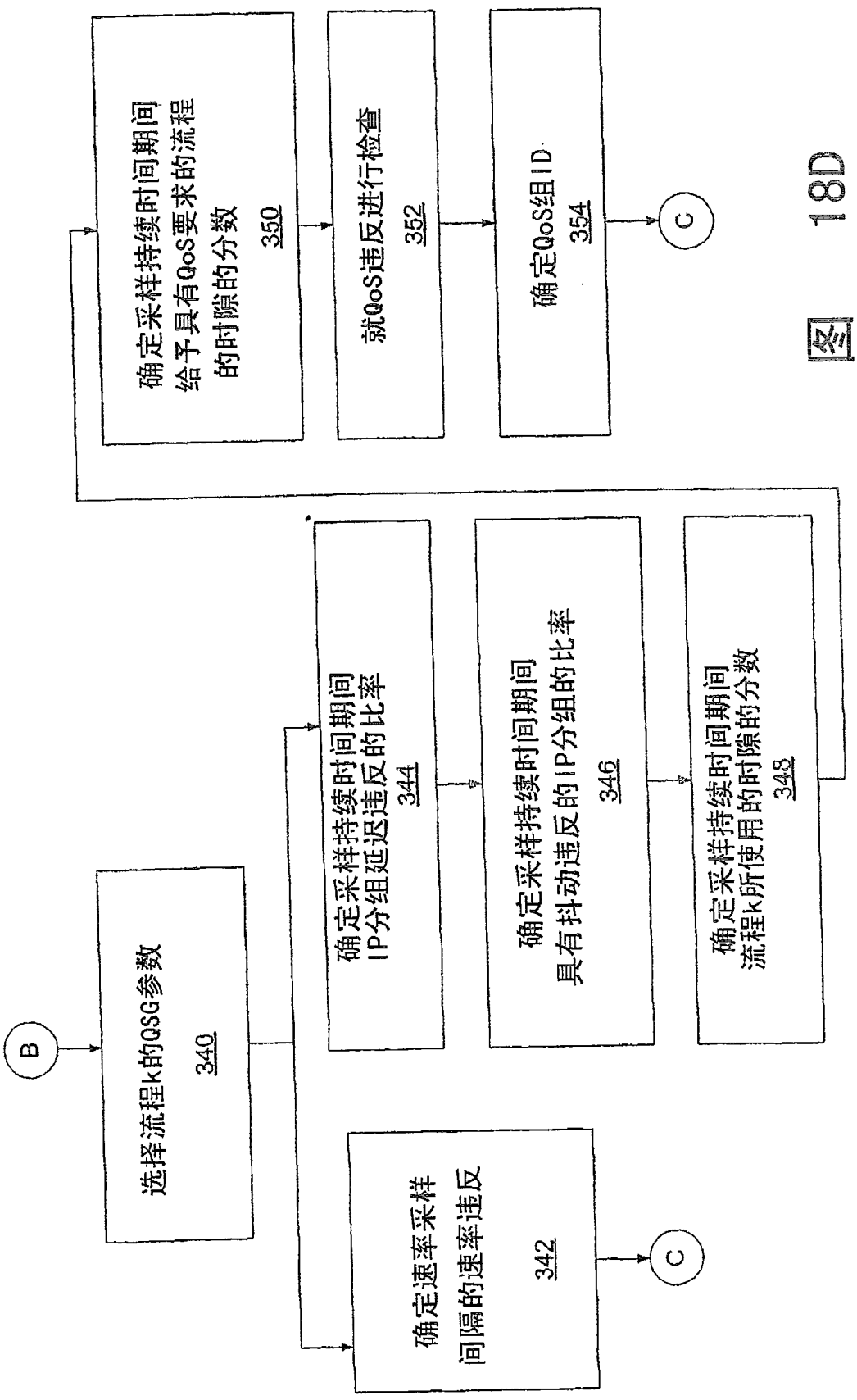

B
选择流程k的QSG参数
340
确定速率采样
间隔的速率违反
342
C
确定采样持续时间期间
IP分组延迟违反的比率
344
确定采样持续时间期间
具有抖动违反的IP分组的比率
346
确定采样持续时间期间
流程k所使用的时隙的分数
348
确定采样持续时间期间
给予具有QoS要求的流程
的时隙的分数
350
就QoS违反进行检查
352
确定QoS组ID
354
C

图 18D

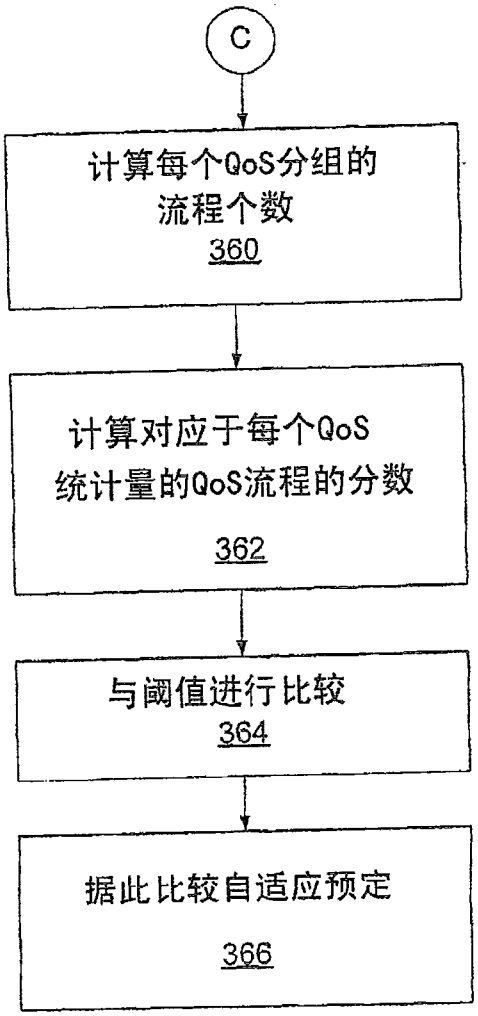

C
计算每个QoS分组的流程个数
360
计算对应于每个QoS统计量的QoS流程的分数
362
与阈值进行比较
364
据此比较自适应预定
366

图 18E

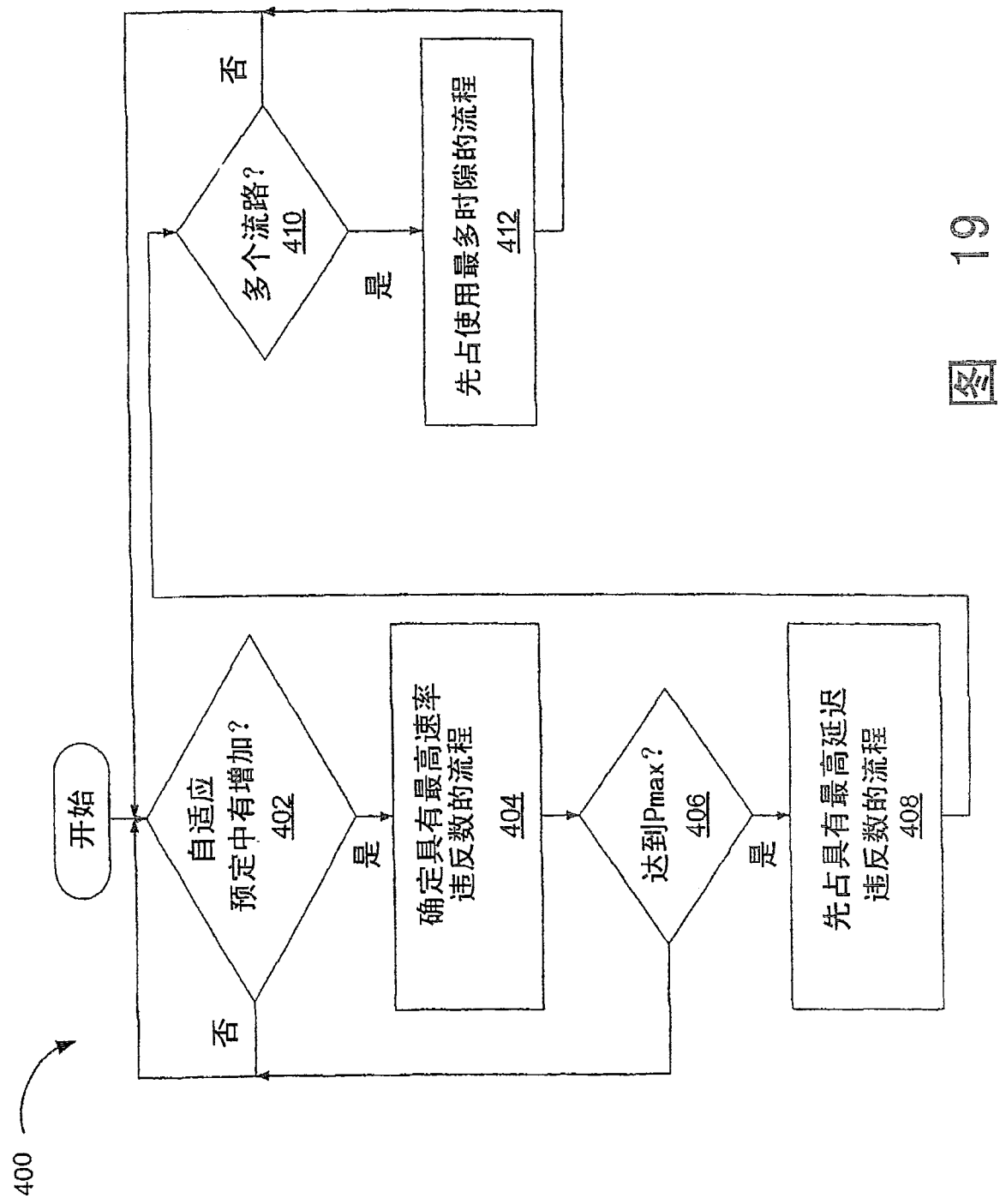

400
开始
自适应
预定中有增加？
402
否
是
确定具有最高速率
违反数的流程
404
达到Pmax？
406
是
先占具有最高延迟
违反数的流程
408
多个流路？
410
否
是
先占使用最多时隙的流程
412

图 19

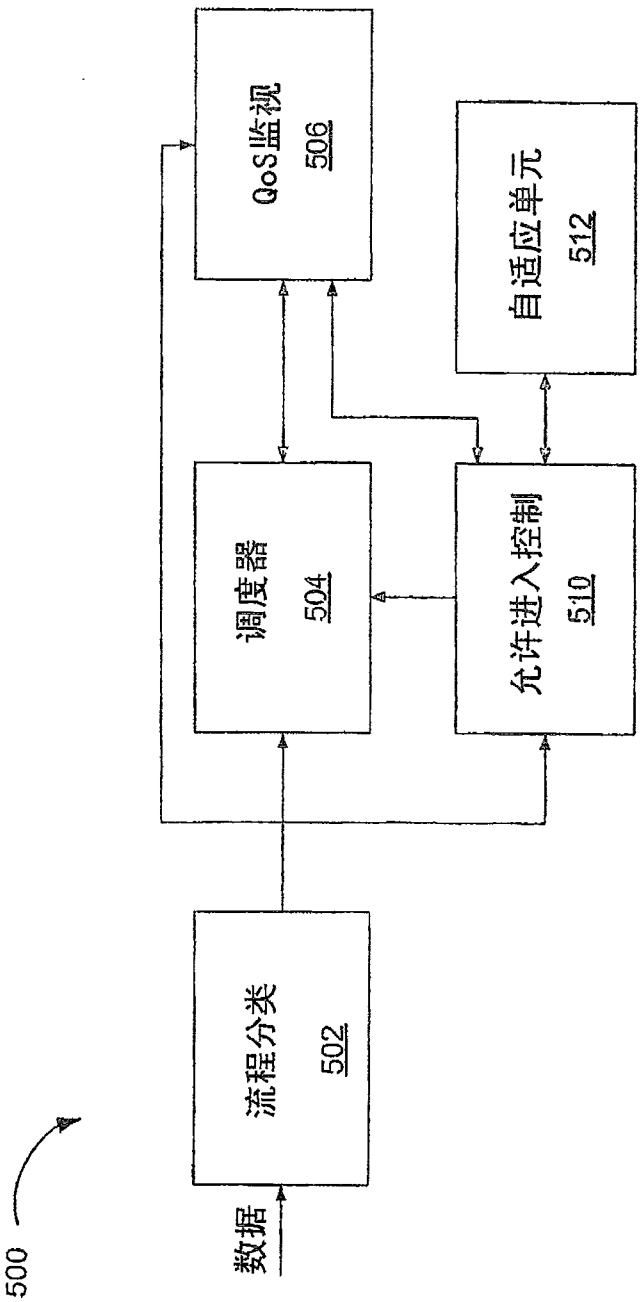


图 20