[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 200480042869.7

[45] 授权公告日 2010 年 2 月 3 日

[11] 授权公告号 CN 100588174C

[22] 申请日 2004.4.26
[21] 申请号 200480042869.7
[86] 国际申请 PCT/EP2004/004379 2004.4.26
[87] 国际公布 WO2005/104452 英 2005.11.3
[85] 进入国家阶段日期 2006.10.26
[73] 专利权人 意大利电信股份公司
地址 意大利米兰
[72] 发明人 鲁斯阿诺·莱恩兹尼
恩佐·米恩格兹 玛斯莫·萨斯
安里克·斯卡罗尼 吉奥瓦尼·斯蒂
维尼西奥·维尔塞隆
[56] 参考文献
US2002/0114334A1 2002.8.22
US2003/0093526A1 2003.5.15
WO01/79992A2 2001.10.25
审查员 张仁杰
[74] 专利代理机构 中国国际贸易促进委员会专利商标事务所
代理人 李 玲

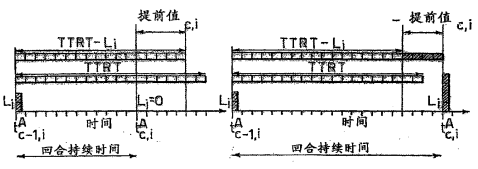
权利要求书 5 页 说明书 19 页 附图 6 页

[54] 发明名称

在同一个网络上调度同步和异步分组的方法和系统

[57] 摘要

一种调度信息包的多个流(SF，AF)利用服务资源的系统，其中，流包括速率得到保证的同步流(SF)和尽力异步流(AF)，异步流(AF)利用同步流(SF)未加以利用的资源的服务容量。 服务器(S)以连续的回合访问流(SF，AF)，首先访问同步流(SF)，接下来是访问异步流(AF)。 服务器(S)被配置为检测任何积压的同步流，当服务器(S)访问任何所述积压的同步流时，它允许积压的同步流在给定时间内利用资源，从而，所述同步流(SF)在每一个回合都具有保证的传输窗口。 当所述服务器(S)访问任何异步流(AF)时，确定自从所述服务器(S)对同一个异步流(AF)进行最后一次访问所消逝的时间，以及：－i)如果对所述异步流(AF)的访问在基于参考回合时间(TTRT)的预期的时间之后发生，则制止正在被访问的异步流(AF)为正在进行的回合利用资源，－ii)如果访问在基于参考回合时间(TTRT)的预期的时间之前发生，则允许正在被访问的异步流(AF)在相应的给定时间内利用资源。 相应的给定时间是服务器访问的提前值的函数，从而，该函数调节每一个异步流(AF)利用同步流(SF)未加以利用的资源的服务容量的状况。



1、一种调度信息包的多个流（SF，AF）利用服务资源的方法，所述多个流包括速率得到保证的同步流(SF)和尽力异步流(AF)，其中，所述异步流(AF)利用所述同步流(SF)未加以利用的所述资源的服务容量，该方法包括下列步骤：

- 提供服务器(S)，该服务器以连续的回合访问所述多个流（SF，AF），所述服务器(S)在每一个所述回合内首先访问所述同步流(SF)，接下来是访问所述异步流(AF)，

- 在多个所述同步流(SF)中检测任何积压的同步流*h*，

- 当所述服务器(S)访问任何所述积压的同步流*h*时，允许所述积压的同步流*h*在给定时间内利用所述资源，从而，所述同步流(SF)在每一个所述回合都具有保证的传输窗口，

确定参考回合时间(TTRT)，它表示所述服务器(S)完成对所述多个流（SF，AF）的一个回合的访问的预期的时间，

- 当所述服务器(S)访问任何所述异步流(AF)时，确定自从所述服务器(S)对同一个异步流(AF)进行最后一次访问所消逝的时间，以及；

- i)如果对所述异步流(AF)的访问在基于所述参考回合时间(TTRT)的预期的时间之后发生，则制止正在被访问的所述异步流(AF)为正在进行的回合利用所述资源，

- ii)如果对所述异步流(AF)的访问在基于所述参考回合时间(TTRT)的预期的时间之前的时间点发生，则允许正在被访问的所述异步流(AF)在相应的给定时间内利用所述资源，

所述方法的特征在于，所述相应的给定时间是服务器访问比预期提前的时间量的函数，从而，所述函数调节每一个所述异步流(AF)利用所述同步流(SF)未加以利用的所述资源的所述服务容量的状况，并且其中所述相应的给定时间按照比例因子 α_i 与所述时间量成比例。

2、根据权利要求 1 所述的方法，其特征在于，该方法包括所述

服务器(S)在以下时刻停止访问信息包的任何流（SF，AF），无论哪个首先发生：

- 当所述给定时间或相应的给定时间已经消逝时，或
- 当正在被访问的流已经清除了其积压时。

3、根据权利要求 1 所述的方法，其特征在于，该方法包括下列步骤：

- 确定所述同步流(SF)的同步带宽 H_h，以及
- 作为所述同步带宽 H_h 的函数，对所述给定时间设置上限。

4、根据权利要求 1 所述的方法，其特征在于，该方法包括下列步骤：

- 将表示在前面的回合中在访问所述异步流时由所述服务器(S)累积的延迟的值 L_i 与每一个所述异步流(AF)关联，
- 当所述服务器(S)访问所述异步流(AF)时，从所述参考回合时间(TTRT)中减去前面的回合的持续时间，以获得减法结果，以及
-从所述减法结果中减去所述值，如此，计算出当前回合的服务器访问比预期提前的时间量的值。

5、根据权利要求 4 所述的方法，其特征在于，该方法包括下列步骤：

- 如果所述时间量的值是负的，则将其绝对值存储在表示累积的延迟的所述值 L_i 中，以及
-制止正在被访问的所述异步流(AF)为正在进行的回合利用所述资源。

6、根据权利要求 4 所述的方法，其特征在于，该方法包括下列步骤：

- 如果所述时间量的值是正的，则将表示累积的延迟的所述值 L_i 重置为零，以及
-允许正在被访问的所述异步流(AF)在所述相应的给定时间内利用所述资源。

7、根据权利要求 1 所述的方法，其特征在于，它包括在范围(0,1]

内选择所述比例因子 α_i 的步骤。

8、根据权利要求 1 所述的方法，其特征在于，该方法包括为所述异步流(AF)选择相应的比例因子 α_i 的步骤。

9、根据权利要求 1 所述的方法，其特征在于，该方法包括为所述同步流(SF)确定相应的同步带宽 H_h 的步骤，其中，对应于所述同步带宽的时间间隔的总和低于或等于所述参考回合时间(TTRT)。

10、一种调度信息包的多个流（SF，AF）利用服务资源的系统，所述多个流包括速率得到保证的同步流(SF)和尽力异步流(AF)，其中，所述异步流(AF)利用所述同步流(SF)未加以利用的所述资源的服务容量，该系统包括服务器(S)，该服务器被配置为：以连续的回合访问所述多个流（SF，AF），所述服务器(S)在每一个所述回合内首先访问所述同步流(SF)，接下来是访问所述异步流(AF)，所述服务器(S)被配置为：

- 在多个所述同步流(SF)中检测任何积压的同步流*h*，

- 当所述服务器(S)访问任何所述积压的同步流*h*时，允许所述积压的同步流*h*在给定时间内利用所述资源，从而，所述同步流(SF)在每一个所述回合都具有保证的传输窗口，

-确定参考回合时间(TTRT)，它表示所述服务器(S)完成对所述多个流（SF，AF）的一个回合的访问的预期的时间，

- 当所述服务器(S)访问任何所述异步流(AF)时，确定自从所述服务器(S)对同一个异步流(AF)进行最后一次访问所消逝的时间，以及：

- i)如果对所述异步流(AF)的访问在基于所述参考回合时间(TTRT)的预期的时间之后发生，则制止正在被访问的所述异步流(AF)为正在进行的回合利用所述资源，

- ii)如果对所述异步流(AF)的访问在基于所述参考回合时间(TTRT)的预期的时间之前的时间点发生，则允许正在被访问的所述异步流(AF)在相应的给定时间内利用所述资源，

所述系统的特征在于，所述相应的给定时间是服务器访问比预

期提前的时间量的函数，从而，所述函数调节每一个所述异步流(AF)利用所述同步流(SF)未加以利用的所述资源的所述服务容量的状况，并且其中所述相应的给定时间按照比例因子α_i与所述时间量成比例。

11、根据权利要求 10 所述的系统，其特征在于，所述服务器(S)被配置为在以下时刻停止访问信息包的任何流（SF，AF），无论哪个首先发生：

- 当所述给定时间或相应的给定时间已经消逝时，或
- 当正在被访问的流已经清除了其积压时。

12、根据权利要求 10 所述的系统，其特征在于，所述服务器(S)被配置为：

- 确定所述同步流(SF)的同步带宽 H_h，以及
- 作为所述同步带宽 H_h 的函数，对所述给定时间设置上限。

13、根据权利要求 10 所述的系统，其特征在于，所述服务器(S)被配置为：

- 将表示在前面的回合中在访问所述异步流时由所述服务器(S)累积的延迟的值 L_i 与每一个所述异步流(AF)关联，
- 当所述服务器(S)访问所述异步流(AF)时，从所述参考回合时间(TTRT)中减去前面的回合的持续时间，以获得减法结果，以及
-从所述减法结果中减去所述值，从而计算出当前回合的服务器访问比预期提前的时间量的值。

14、根据权利要求 13 所述的系统，其特征在于，所述服务器(S)被配置为：

- 如果所述时间量的值是负的，则将其绝对值存储在表示累积的延迟的所述值 L_i 中，以及
-制止正在被访问的所述异步流(AF)为正在进行的回合利用所述资源。

15、根据权利要求 13 所述的系统，其特征在于，所述服务器(S)被配置为：

- 如果所述时间量的值是正的，则将表示累积的延迟的所述值

L_i重置为零，以及

-允许正在被访问的所述异步流(AF)在所述相应的给定时间内利用所述资源。

16、根据权利要求10所述的系统，其特征在于，所述服务器(S)被配置为使用位于范围(0,1]中的比例因子作为所述比例因子α_i。

17、根据权利要求10所述的系统，其特征在于，所述服务器(S)被配置为所述异步流(AF)使用相应的比例因子α_i。

18、根据权利要求10所述的系统，其特征在于，所述服务器(S)被配置为：为所述同步流(SF)确定相应的同步带宽H_h，其中，对应于所述同步带宽的时间间隔的总和低于或等于所述参考回合时间(TTRT)。

19、一种包括权利要求10到18中的任一权利要求所述的系统的通信网络。

在同一个网络上调度同步和异步分组的方法和系统

技术领域

本发明涉及用于调度资源，如通信网络中的信号处理和/或传输资源的利用的技术。

背景技术

过去十年，基于分组的网络技术的发展，以及这些技术的实例（即，因特网，作为全球通信基础架构）的广泛应用，正在挑战传统的网络服务模式（如原始的 TCP/IP 的尽力（best-effort）模式）。为了改善服务质量(QoS)，需要新的、替代的服务模式和业务管理方案。

在许多准备好在节点上某一个输出链路上进行传输的分组中，判断必须传输哪个分组，是由管理该输出链路的服务规则（或调度算法）作出的。因此，此组件在确定给定体系结构的 QoS 能力时起作关键的作用。

分组流可能需要尽力服务，即，没有具体服务质量(QoS)要求的服务，或者需要速率得到保证的服务，即，可保证分组有最小的发送速率。速率得到保证的业务和尽力业务具有非常不同的 QoS 要求：速率得到保证的业务要求最低速率，不管网络条件如何，而尽力业务则不需要这样的保证。

从用户的观点来看，需要速率得到保证的应用（例如，分组电话）在它们接收到高于所需的速率的速率的情况下可能不会改善它们的性能，而使用尽力服务的应用（例如，Web 浏览）可能会随着可用的容量增大而显著地改善它们的性能。

在过去几年，对于调度方法，花费了大量的精力进行研究。具体来说，为设计公平的队列服务规则，花费了很大的精力，即，近似于

“通用处理器共享(GPS)理想服务范例”的服务规则。

L.Lenzini,E.Mingozzi,G.Stea 所著的标题为“AUnifying Service Discipline for Providing Rate-Based Guaranteed and Fair Queuing Services Basedon the Timed Token Protocol”（2002 年 9 月，IEEE Transactions on Computers 第 51 卷，No.9,第 1011-1025 页，以及 WO-A-02/35777 和 WO-A-03/088605 的文章定义了分组交换网络的环境中的双类交换范例（或 DC 范例）。

简而言之，这样的方法致力于其容量是 C 的开关元件的输出链路。

传入的流被分为两组：前一组包括 R 速率得到保证的流 RG，每一个流都需要最低速率 r_i，$1 \le i \le R$，以便 $\sum_{i=1}^{R} r_i \le C$；后一组包括 B 尽力流 BE，每一个流都根据加权公平队列范例，竞争具有权重 w_j 的可用容量，$1 \le j \le B$。

图 1 显示了两组流。

假设 $B_R(t)$ 和 $B_B(t)$ 分别表示在时间 t 时积压的（即，在待服务的队列中等待的）速率得到保证的流和尽力流，而 $R_i(t)$ 是在时间 t 时被积压的流的瞬时服务速率，在任何时间 t，双类(DC)范例被定义为

$$R_i(t)=\begin{cases}\dfrac{r_i}{\sum_{j\in B_R(t)} r_j}\cdot C & i\in B_R(t),\ B_B(t)=\varnothing \\ r_i & i\in B_R(t),\ B_B(t)\neq\varnothing \\ \dfrac{w_i}{\sum_{j\in B_B(t)} w_j}\left(C-\sum_{j\in B_R(t)} r_j\right) & i\in B_B(t)\end{cases} \qquad (1)$$

简而言之，在 DC 范例中，当没有尽力流被积压时，速率得到保证的流基于它们的所需的速率公平地共享链路容量。相反，当有某些尽力流被积压时，被积压的速率得到保证的流受到约束，只使用为其配置的速率。而任何剩余的容量都在尽力流之间公平地共享。

这意味着，DC 范例是任务守恒的：显然(1)只能在其中可以同时服务于多个流的理想的液体流系统中成立。

在上文中引用的现有技术文档中所描述的调度配置定义了计时

令牌服务规则(TTSD)或分组计时令牌服务规则(WO-A-03/088605)。此引用的最后一个配置通过下列方式允许多个流共享资源:

-让同步流获得最小所需带宽;

-让异步流平均地共享剩余的带宽，即，未预定或空闲同步流临时未使用的带宽。

所引用的 Lenzini 等所著的文章表明，尽管是为令牌环网中的媒体访问控制的差别非常大的环境而设计的，计时令牌协议(TTP)仍提供了某些特点，使得它成为开发近似于 DC 范例的分组服务规则的良好的起点。具体来说，TTP 考虑了两种业务类别，即，同步和异步业务。在每一次对某一个节点进行令牌访问时，在有限的时间内可以传输同步通信，而异步业务传输时间适应令牌速度，以便使令牌之间的时间保持稳定。可以将同步服务用于传输实时业务，而 TTP 为每一个节点提供了相同的传输异步业务的机会。此外，异步业务还会有效地利用同步通信未使用的容量。

注意到这一点，计时令牌服务规则(TTSD)应用由 TTP 采用的相同规则来控制媒体访问，以便在一组流之间共享传输容量。具体来说，在 TTSD 中，可以保证为速率得到保证的（或同步）流 RG 提供最低速率，而所有尽力（或异步）流 BE 都平均地共享大多数剩余容量。如此，TTSD 也近似于 DC 范例的所有尽力流权重都具有相同值的特定实例。

由 A.Francini、F.M.Chiussi、R.T.Clancy、K.D.Drucker 和 N.E.Idirene 所著的标题为“Enhanced Weighted Round Robin Schedulers For Accurate Band width Distribution in Packet Networks”（2001 年 11 月，Computer Networks,第 37 卷，561-578 页）的文章提出了循环调度的框架。在该框架中，第一级别的服务器首先访问所有速率得到保证的流，然后，将控制权传递到管理尽力流的次服务器。给予次服务器的容量共享，根据速率得到保证的流的积压状态，在回合与回合之间是不同的，从而空闲的速率得到保证的流未使用的容量事实上由次服务器使用。然而，根据此现有技术的解决

方案，尽力流平均地共享可用容量。

J.Bennett,H.Zhang 所著的标题为“Hierarchical Packet Fair Queuing Algorithms”（1997 年 10 月，IEEE/ACM Transactionson Networking, Vol.5,No.5,第 675-689 页）描述了分层的 GPS(H-GPS)算法，该算法基于流体流服务规则。

可以将 DC 范例中实施的带宽共享与在 GPS 和 H-GPS 中所提供的带宽共享进行比较，假设一个简单的情况：两个速率得到保证的流 i,j,与尽力流 k 竞争可用带宽。让我们首先假设，当所有三个流都被积压时，根据任意选择的权重（适于为流 i 和 j 实施所需的保证）在 GPS 下调度三个流。当流 i 未被积压时，其带宽的一部分将转给流 j，即使流 k 被积压，这是不希望有的。

然后假设，在 H-GPS 下调度相同的三个流，在这种情况下，在选择树形结构（其叶子是被调度的流）时存在一些自由度。由于考虑了三个流，因此，给出了三个可能的情况：

a) 所有三个流都是同一个类的成员。可以轻松地看出，在此情况下，H-GPS调度器的行为与GPS调度器的行为相同，这已经得到处理。

b) 有两个流属于同一个类，而有一个流属于另一个类。在该情况下，给出了两个可能的子情况，分别在图2的左边和右边显示出来。在该图中，显示了三个流的分组队列。通过该图中的虚线箭头显示了带宽的重新分配情况。

b1)两个速率得到保证的流 i 和 j 都是同一个类的成员，而尽力流 k 属于另一个类（图 2，右边）。在该情况下，当流 i 未被积压时，其带宽可供其类别的其他流使用；在此具体情况下，只能供流 j 使用。

b2)速率得到保证的流(假设为 j)和尽力流 k 是同一个类的成员，而流 i 属于另一个类（图 2，左边）。在此情况下，当流 i 未被积压时，其带宽可供其他类别使用，如此，在流 j 和 k 之间分配。

申请人注意到，在两种情况下，当流 i 未被积压时，速率得到保证的流 j 获得多于其所需的速率的速率，即使流 k 被积压，这是不希

望有的。

在 I. Stoica, H. Abdel-Wahab, K. Jeffay 所著的"On the Duality between Resource Reservationand Proportional Share Resource Allocation," Proceedings of Multimedia Computing and Networking 1997, San Jose, CA, February 1997 年 2 月，SPIE Proceedings Series, Vol.3020, 207-214 页）中提出了在一个处理器上调度两组任务的统一的框架。尽管引用了不同于网络调度的环境，但是，此现有技术文档提出了应该近似于 DC 范例。为此，建议将动态权重管理策略与名为 EEVDF 的基于 GPS 的服务规则相结合。在此环境中，流仍基于权重来接收服务，但权重是随着流的积压状态而动态地改变的。

申请人注意到，权重管理策略需要在分组时间量程内重新计算整套权重。此外，由于使用虚拟时间函数来对分组进行分类，因此，重新计算权重也应该意味着对分组重新进行分类。当在平面的或在分层的框架中使用其他基于 GPS 的服务规则时，也会产生相同的问题。因此，通过基于 GPS 的服务规则的 DC 范例近似法在计算上看起来不是可行的解决方案。

美国专利申请 US2003/0103514 描述了基于信用的循环复用分组调度算法。然而，"基于信用的循环复用"是为提供公平的队列而设计的调度算法。申请人注意到，如此，上述算法没有考虑到在如 DC 范例所说明的不同业务类别（即，速率得到保证的和尽力）之间共享链路带宽的问题。

如 WO-A-01/24428 所描述的分层的优先次序循环复用(HPRR)将流分类为较少的业务类别。尽力业务被分为属于"默认"类别。根据速率有限的优先级调度（作为"亏损循环"调度的变体实现的）来服务于流类别。超过给定的 profile 的非默认类别的业务被当做尽力业务。尽管这样的调度方法在设计由每一个类别接收到的服务时具有某些灵活性，但是，它不允许默认类别使用由空闲的业务类别临时未使用的带宽。事实上，当某类别空闲时，一个回合的持续时间相应地缩小，从而每一个被积压的类别都接收到剩余的带宽的公平的份额。这与其

中只有允许尽力流访问剩余的带宽的 DC 范例完全不同。

作为“亏损循环”的扩展，提出了显式考虑了两种业务类别（即，延迟-关键（即，实时）业务和尽力业务）的调度规则，即 DRR.+（参见 M. Shreedhar and G. Varghese: "Efficient Fair Queueing Using Deficit Round-Robin", IEEE Transactionon Networking, Vol.4, No.3, 375-385 页, 1996 年 6 月)和 DRR++(参见 M. H. MacGregor and W.Shi: "Deficitsfor Bursty Latency-critical Flows:DRR++", Proceedings of the IEEE International Conference on Networks (ICON'00), Singapore, 2000 年 9 月 5 日-8 日）。

MacGregor 等人所著的论文显示了，前面引用的配置不适于调度多跳跃网络中的延迟-关键的业务，因为一旦累积了任何“爆炸性”，就会将它降级为尽力通信。第二种配置，即，DRR++，也负责爆炸性的延迟-关键的流。然而，可以轻松地发现，没有对 DRR++进行正式的分析，以便确定在什么条件下可以实施什么实时保证。

发明内容

前面的分析表明，现有技术的配置存在某些基本的限制。

具体来说，当同时调度了速率得到保证的流和尽力流时，上述的一些配置不能防止速率得到保证的流获得超过配置的最小带宽，如此使尽力流丧失了可能的更好的服务级别。在它们可以防止的情况下，它们不允许尽力流根据可选择的权重共享可用的带宽，或者在高速度环境中以不可行的调度开销的代价才能提供这样的功能。

申请人发现，为了有效而同时为速率得到保证的和尽力流负责，服务规则可以逼近替代的服务范例，双类(DC)范例。在 DC 范例中，给速率得到保证的流提供了固定的速率，该速率等于被请求的速率，不管网络条件如何，尽力流根据可选择的权重共享了所有的剩余容量（即，没有为速率得到保证的流预留的容量，或空闲的速率得到保证的流临时未使用的容量）。然后，基于 DC 范例的调度规则将最大化为尽力业务提供的服务，同时仍满足为速率得到保证的业务提供的保

证。

作为示例，在其中尽力流代表了发自（或发往）不同公司或部门的综合 TCP 业务的网络中，强制有权重的容量共享将是非常需要的。

此外，用户应该能够选择权重，以便尽力流获得可用的带宽的可预测的份额。例如，在多服务网络中，可以为背景业务（如从机器发往机器的业务，非交互式的应用）分配非常低的带宽份额，而可以为人与机器的交互式应用预留较高的份额。

如此，本发明的目标是克服现有技术配置的缺点，同时满足上文中所概述的需求。根据本发明，这样的目标是通过具有随后的权利要求中阐述的特征的方法来实现。本发明还涉及对应的系统，相关的通信网络和可加载到至少一个计算机的存储器中的计算机程序产品，并包括软件代码部分，用于当产品在计算机上运行时，执行本发明的方法的步骤。如这里所使用的，对“这样的计算机程序产品”的引用相当于对“计算机可读的介质”的引用，该介质包含用于控制计算机系统以协调本发明的方法的性能的指令。对“至少一个计算机”的引用显然强调了本发明的系统以分布式的/模块的方式实现。

基本上，本发明的优选实施例是调度多个信息分组的流利用服务资源的方法，多个流包括速率得到保证的同步流和尽力异步流，其中，异步流利用同步流未加以利用的资源的服务容量。服务器以连续的回合访问流，首先在每一个回合内访问同步流，接下来是访问异步流。

下文中的术语“访问”是指在服务周期或回合中，将调度系统的注意力给予特定用户（例如，队列、节点或任务）的操作/事件，为了它能够获得所分配的那部分资源。

当服务器访问任何所述积压的同步流时，它允许积压的同步流在给定时间内利用资源，从而，同步流在每一个所述回合都具有保证的传输窗口。确定参考回合时间，它表示服务器完成对流的回合的访问的预期的时间。当服务器访问任何所述异步流时，确定自从服务器对同一个异步流进行最后一次访问所消逝的时间，以及：

- i)如果访问在基于参考回合时间的预期的时间之后发生，则制

止正在被访问的异步流为正在进行的回合利用资源，

- ii)如果访问在基于参考回合时间的预期的时间之前发生，正在被访问的异步流在相应的给定时间内利用资源，该相应的给定时间是服务器访问的提前值（earliness）的函数（例如，成比例），从而，该函数调节每一个异步流利用同步流未加以利用的资源的服务容量的状况。

优选情况下，可以选择相应的调节异步流利用服务容量的函数。

在特别优选实施例中，通过所选择的比例因子，例如，在范围(0,1]中（即，排除 0，但可能包括 1）使分配异步流的相应的给定时间与提前值成比例。如此，可以为各个异步流选择相应的比例因子。

如此，这里所描述的配置实现了通用化计时令牌服务规则（下面称为 GTTSD），即，用于在多个客户端之间调度共享资源的一般规则。

这样的配置可以用于在分组交换网络（如因特网）中在多个分组流之间调度路由器或交换机的输出链路。虽然这里是在该特定环境中描述的，然而，GTTSD 也适用于多个其他环境中，包括（但不仅限于）在处理器中调度任务。

这里所描述的 GTTSD 调度配置近似于 DC 范例。事实上，它显式和同时考虑了两类分组流：同步（即，速率得到保证的）和异步的（即，尽力）。此外，它允许用户指定应该将可用带宽的多大的份额给予不同的尽力流。给同步流提供了最小的可保证的速率，而根据可选择的权重，在异步流之间共享同步流不严格地需要的传输容量。

GTTSD 配置的带宽共享属性可以用于，例如：

- 通过在每一个多媒体（例如，声音）流中强制最小的可保证的速率来供应多媒体流；

- 给尽力业务提供最大可能的服务级别，如此，最大化它们的响应性。

这里所描述的 GTTSD 调度配置允许用户选择权重(呈现 α 因子）的形式，以便尽力流获得可预测的可用带宽份额。此外，GTTSD 为

任意数量的速率得到保证的流预留了固定的绝对量的带宽，不管是否有竞争。

此外，GTTSD 近似于 DC 范例。如此，使速率得到保证的流没有预定的或没有立刻使用的任何带宽可以供尽力流（根据它们的 α 因子，共享带宽）使用。如此，异步流可以根据可选择的权重共享剩余的带宽。

附图说明

现在将参考下面的附图，只作为示例，对本发明的优选实施例进行描述，其中：

-图 1 和 2 一般涉及现有技术已经在前面描述了，

-图 3 是这里所描述的系统的体系结构的方框图，

-图 4 是根据这里所描述的配置的系统的可能操作的图表，以及

-图 5 和 6 是代表涉及这里所描述的配置的操作数据的图表。

具体实施方式

虽然本详细描述将专门讲述交换节点中的分组调度，但是，注意，GTTSD 也可以用于各种环境中，即，需要在多个请求实体之间分配共享资源时（例如，操作系统中的 CPU）。

GTTSD 概括了 TTSD 配置，这在前面已经进行了描述。

下面将考虑以流的形式组织的输入业务。流是在输出链路上转发的可区别的业务流。用于将业务分类为流的机制不构成本申请的主题。

刚刚提供的定义概括性足够大，可以匹配 IntServ 的流，和 DiffServ 的 BA（BehaviorAggregate-在链路的特定方向具有涉及 DiffServ 的字段的相同标记的分组的集合）。

GTTSD 对两种流进行管理，即，同步流和异步流，它们按照固定的顺序循环地进行传输。现在基于这样的假设对 GTTSD 进行描述和分析：此配置对一组 N_S 同步流和一组 N_A 异步流进行管理，两组分

别被表示为 SF 和 AF。我们假设每一个流都将其业务排列为单独的队列。

图 3 代表了所考虑的系统模型。

在一个回合（或循环）内，首先考虑同步流 SF，接下来是异步流 AF。当服务器 S 访问积压的同步流 h 时，后者被允许在一段时间内传输其业务，该时间由其同步带宽 H_h 设置上限，这在流初始化时选择。一旦此时间间隔已经消逝，或当流已经清除了其积压时，无论哪个首先发生，服务器停止访问流 h。因此，同步流 SF 在每一个回合都具有保证的传输窗口。当服务器 S 访问异步流 AF 时，它计算自从对同一个流进行最后一次访问所消逝的时间。如果访问在预期的时间之后发生，则制止该流为正在进行的回合传输其业务。否则，允许传输其业务；在该情况下，服务器为从该流传输业务可以花费的时间间隔与服务器访问的提前值成比例。

否则-当服务器访问任何异步流时，基于参考（预期的）回合时间确定由服务器对同一个异步流进行最后一次访问所消逝的时间，以及：

-i)如果访问在基于参考回合时间的预期的时间之后发生，则制止正在被访问的异步流为正在进行的回合利用资源，

-ii)如果访问在预期的时间之前发生，则允许异步流在与服务器访问的提前值成比例的时间间隔内利用资源。

当访问正好在预期的时间进行时，上文考虑的两个策略是一致的。在该情况下，服务器访问没有发生提前，异步流可能（概念上地）被允许利用资源的时间与等于零提前值成比例，如此，也为零。这在实践中对应于制止正在被访问的所述异步流为正在进行的回合利用资源。

一旦计算出的时间间隔已经消逝，或当流已经清除了其积压时，无论哪个首先发生，服务器停止访问流。因此，异步流在一个回合中没有保证的传输窗口；相反，它们的传输窗口应该适应服务器访问的速度，回合与回合之间是不同的。

按如下方式确定在单一的回合中允许传输异步流的传输时间。

必须选择参考回合持续时间 *TTRT*（目标令牌循环时间）。下面将详细描述选择 TTRT 的规则。每一个异步流 i 都与变量 L_i（延迟值）关联，该变量记录了在前面的回合中累积的延迟。服务器的第 c 次对异步流 i 的访问的时间由 $t^A_{c,i}$ 来表示。当服务器访问异步流时，它从 *TTRT* 中减去前面的回合的持续时间（即，从该流被最后一次访问开始的时间），并从该结果中减去 L_i，如此，计算出该流的当前回合的提前值；即：

$$earliness_{c,i} = TTRT - L_i - \left(t^A_{c,i} - t^A_{c-1,i}\right) \quad (2)$$

如果提前值是负的（意味着，服务器访问实际在预期的时间之后发生），则将其绝对值存储成 L_i 变量，对于正在进行的回合，不允许流传输其业务，因为这可能会过度地延迟随后的服务器对同步流的访问。如果提前值是正的，则将 L_i 重置为零，允许该流在某一个时间间隔内进行传输。这样的时间间隔，叫做该流的异步带宽，并表示为 $a_{c,i}$，按如下方式进行计算：

$$a_{c,i} = \alpha_i \cdot earliness_{c,i} \quad (3)$$

其中，α_i 是流的 α 因子，包括在(0,1]中，在流初始化时选择。这里所描述的 GTTSD 配置，概括了 TTSD 配置，后者是前者的特定实例，其中 $\alpha_i = 1$，$i = 1 \ldots N_A$。如此，在 TTSD 中，对于所有异步流，异步带宽始终等于提前值。图 4 显示了在 GTTSD 中如何计算提前值的图形表示。显示了早期的服务器访问（左）和晚期的服务器访问（右）的时间。由 GTTSD 为计算对异步流进行访问时的提前值而执行的算法与没有同步业务的 TTP 节点执行的算法相同。

由于 GTTSD 任务守恒，因此，一个回合的持续时间是可以改变的。然而，在供应同步流之后，该算法尝试通过从异步流传输业务来填充大部分预期的 TTRT 时间间隔（只要有被积压的流）。因此，积压的异步流有权使用没有为同步流预留的或者由空闲的同步流临时未使用的大部分容量。在 GTTSD 中，同步带宽的总和（不管名称如何，都被表达为对应于带宽的时间间隔）受到下列不等式的约束（协

议约束）：

$$\sum_{h\in S} H_h \le TTRT \quad (4)$$

下面显示了在 GTTSD 服务器访问的示范性伪代码。为了确保完整性，也显示了流初始化过程。

```
Sync_Flow_Init()
 h=new_sync_flow_id;
 select_synchronous_bandwidth (Hh);
 add_new_flow_at_the_end (h);

 Async_Flow_Init ()
 i=new_async_flow_id;
 Select_alpha_factor (αi);
 Li = 0;
 last_timei= start_of_curr_round;
 add_new_flow_at_the_end (i);
Sync_Flow_Visit (synchronous flow h)
B = current_backlog(h);
Transmit (min (B,Hh));

Async_Flow_Visit (asynchronous flow i)
t = current_time;
earliness = TTRT - Li - (t-last_timei);
if ( earliness > 0 )
{ Li = 0;
  ai= αi*earliness;
  B = current_backlog(i);
  Transmit (min (B, ai)); }
else Li = - earliness;
last_timei = t;

GTTSD_revolution ()
 for (h=1 to Ns) Sync_Flow_Visit (h);
 for (i=1 to Na) Async_Flow_Visit (i);
```

报告的示范性伪代码显示了，在 GTTSD 下，在每一个服务器访问时要执行的操作的数目为 O(1)。

GTTSD 的属性

具体来说，可以证明，同步和异步流的稳定状态速率是：

$$R_i=\begin{cases} C\cdot\dfrac{N_A^{eff}+1}{N_A^{eff}\cdot TTRT+\sum_{h\in S}H_h}\cdot H_i, & i\in S \\ C\cdot\dfrac{TTRT-\sum_{h\in S}H_h}{N_A^{eff}\cdot TTRT+\sum_{h\in S}H_h}\cdot\alpha_i, & i\in A \end{cases} \qquad (5)$$

其中，N_A^{eff}表示α因子的总和，即，$N_A^{eff}=\sum_{j=1}^{N_A}\alpha_j$。

同步流稳定状态速率不是恒定的，因为它们取决于同步流的数目和N_A^{eff}参数。然而，可以直接看出，随着后者增大，同步速率逼近最小值$C\cdot H_h/TTRT$。

因此，提供给异步流的服务给同步流速率设置了上限。另一方面，每一个异步流都接收可用容量的与其α因子成比例的带权重的份额。当N_A^{eff}比较高时，(5)可以改写为：

$$R_i^*\cong\begin{cases} C\cdot H_i/TTRT, & i\in S \\ C\cdot\dfrac{\alpha_i}{N_A^{eff}}\left(1-\sum_{h\in S}H_h/TTRT\right), & i\in A \end{cases} \qquad (6)$$

注意，当没有异步流处于活动时（即$N_A^{eff}=0$），(5)产生：

$$R_i=\frac{H_i}{\sum_{h\in S}H_h}\cdot C,\ i\in S$$

因此，当没有异步流被积压时，稳定状态容量共享实现了DC范例，当异步流被积压时，则逼近它。随着异步流容量需求增大，逼近越强。

图5显示了GTTSD（上面的等式5）中的以及在理想情况下的（上面的等式6）针对N_A^{eff}以及对于各种同步负载$\Sigma_{h\in S}H_h$的规一化同步和异步速率之间的比率R_i/R_i^*。该图显示了当N_A^{eff}为10的量级时，实现了DC范例的良好逼近（在10%内）。

尽管很大，但是具有可调的稳态带宽共享可能不足以在调度器上实施可控制的行为。事实上，在现实的网络环境中，流上的积压条件可能变化得相当快；结果，调度器可以在始终与稳态条件有很大不同的条件下操作。此外，瞬时的带宽共享也可能显著地偏离稳态的带宽共享，如此在异步流之间产生很大的不公平性。那么，我们需要比较

详细地理解在瞬时状态下异步流如何共享它们的带宽。

在瞬时状态下，每一个异步带宽都通过阻尼振荡朝其稳态值收敛。如此，在瞬时状态下，异步流不能永久地使用多于（少于）其稳态带宽。一个简单的示例说明了这种情况。

考虑 2x2 系统，其中，$\alpha_1=1.0\ \alpha=[1.0,0.7]$，$\alpha_2=0.7$，$TTRT-\sum_{h\in S}H_h=50ms$。稳态异步带宽是 $a_1\cong 18.51ms$，$a_2\cong 12.96ms\ \mathbf{a}^{(\infty)}\cong[18.51ms,12.96ms]^T$。图 6 显示了 2 个异步带宽的演变围绕稳态值振荡。

α 因子的选择

如上所述，异步流 AF 根据它们的 α 因子共享它们的带宽。然而，α 因子越高，它们的总和 N_A^{eff} 就越大，因此，比率 R_i/R_i^* 就靠近 1。如此，α 因子应该被规一化，以便最大的 α 因子等于 1。在现实的环境中，动态地创建并破坏流。

尽管讨论在 GTTSD 调度器中建立异步流的可能的接口不在本申请的范围之内，但是，可以假设，异步流 i 通过指定权重 w_i 来竞争可用的带宽，如在 GPS 中那样。在该情况下，可以按如下方式选择其 α 因子：

$$\alpha_i=\frac{w_i}{\max\limits_{j=1,\ldots,N_A}\left(w_j\right)} \tag{7}$$

显而易见，这确保了至少一个 α 因子等于 1。另一方面，当该组异步流变化时（例如，当破坏具有最高的权重的流时），这可能需要重新计算 max 的值。每一个流创建/破坏，根据最差情况的实现成本 $O(\log N_A)$，以软件执行此操作。然而，我们发现，这样的操作相对于分组传输的时标以大得多的时标进行，因此，我们预期，此计算不会产生很大的开销。

扩大 α 因子会增大异步流在每一个回合上可以利用的带宽。如下一节所说明的，这也会最小化当传输有限长度的分组时带宽共享中的失真。

在存在分组的情况下的性能评估

在前面的几节中报告的分析基于这样的假设：每一个流（无论是

同步还是异步）都可以完全利用它根据伪代码有权利用的传输时间。在分组交换网络中，业务是以分组的形式传输的，这些分组需要作为整体来传输。分组使得在每一次服务器访问时正好填充给定传输时间，除非在一个节点中考虑分组重新分段，这看起来好像不是可行的解决方案。

结果，在存在分组的情况下，可以改变前面几节中概述的属性，特别是在它们的传输时间可与（同步或异步）带宽相比的情况下。

在存在分组的情况下评估 GTTSD 性能要求定义分组管理策略，即，决定当在当前服务器访问中没有足够的剩余的传输时间时在什么条件下供应分组的规则。

在定义适当的分组管理策略之后，可以考虑重负载条件下的 GTTSD 调度器的操作，并将由实验产生的平均速率与通过实例化（上面的等式 5）所获得的结果进行比较。也可以确定当系统的负载不太大时流共享容量的方式。

在本节的其余部分，我们用 τ_i 表示流 $i \in S \cup A$ 的最大分组传输时间，用 τ^S 和 τ^A 分别表示所有同步和异步流上的最大分组传输时间。

分组管理策略

由于为同步和异步业务定义了不同的传输过程，因此，可以为两种业务类型分别定义分组管理策略。在本说明书的前言部分中讨论的由 Lenzini 等人作出的著作/文档已经处理了有关同步流所存在的分组管理的问题。

只要涉及异步流，下面的简单和公知的分组管理策略可以被视为起点：

预测：不允许启动分组传输，除非有足够的时间完成它；

溢出：允许启动分组传输，只要有某些剩余的访问时间。

当分组传输时间相对于所有异步流的平均异步带宽可以忽略时，两个策略都适用。

相反，当平均异步带宽可与某些流的最大分组传输时间相比时，则它们无效。

具体来说，根据预测策略，具有较小的α因子的流有“挨饿”的危险，因为它们可能从不会有足够的异步带宽来传输“行标题（head-of-line）”分组。

另一方面，根据溢出策略，具有较小α因子的流容易被过度地满足，如此，在异步流之间产生不公平性。

由于τ参数的存在，异步分组管理策略也可以对同步流速率保证和对协议约束具有轻微的影响。如此，可以应用下列异步分组管理策略：当在当前服务器访问过程中没有足够的时间传输分组时，如果剩余的传输时间大于或等于分组传输时间的一半，则传输分组。

此策略简单，并最小化了在每一个回合上根据伪代码计算出的异步带宽和所利用的传输时间之间的绝对差。在避免饥饿和参数的更宽的范围内的不公平性的意义上，此策略比预测和溢出策略更有效。根据此异步分组管理策略，在下列关系中通过设置$\tau=\tau^A/2$可以获得同步流的最差情况速率保证

$$\overline{R_i}=\frac{N_A+1}{N_A\cdot TTRT+\sum_{h\in S}H_h+\tau^S+\tau}\cdot H_i \qquad (8)$$

其中，τ是由于异步分组的传输的最大溢出，假设下列协议约束成立：

$$\sum_{h\in S}H_h+\tau^S+\tau\le TTRT \qquad (9)$$

不管在异步流上选择什么α因子，表达式(8)成立。

作为普通设计过程的应用，那些精通本技术的人员将基于本说明书中公开的一般原理，轻松地为异步流确定进一步的分组管理规则，例如，选择确定性或概率性的，有状态的或无状态的等等。

在重负载条件下的性能评估

为了洞察在存在分组的情况下（上面的等式5）的适用性的限制，可以模拟重负载条件下的GTTSD调度器操作，并将实验结果与通过分析所获得的结果进行比较。

例如，可以在下列情况下模拟GTTSD调度器：由1个同步流和6个异步流共享具有10Mbps输出链路的IP交换节点，所有的流始终被积压。我们为六个异步流选择下列α因子：$\alpha_{1,2}=0.25$，$\alpha_{3,4}=0.5$，

$\alpha_{5,6}$=1.0。同步流的带宽被选为 *TTRT* 的不变分数 γ，即，$H_1=\gamma \cdot TTRT$，所有流上的最大分组长度是 1500 字节，如此，$\tau^5=\tau^4$=1.2ms。分组长度取自表 1 中的经验分布，代表了在因特网路由器中所实际测量的值。

实验结果显示了，遍历因特网的分组的长度不是均匀分布的。某些分组长度（具体来说，28、40、44、48、52、552、576、628、1420 和 1500 字节）出现的概率相对来说比较大一些。除上面所列的那些分组长度之外的分组长度可以广泛地分成三个范围：40 到 80 个字节，80 到 576、576 到 1500。通过假设分组长度在那些范围内均匀分布，整个范围的概率等于 1.6%，9.6%，17.7%，可以获得正确的类似于因特网的分组长度分布。表 1 中概述了此情况。

分组长度	百分比
28	0.08%
40	3.51%
44	0.22%
48	0.24%
52	0.45%
552	1.10%
576	16.40%
628	1.50%
1420	10.50%
1500	37.10%
从 40 到 80 的范围	1.60%
从 40 到 576 的范围	9.60%
从 576 到 1500 的范围	17.70%

表 1-经验分组长度分布

我们将同步负载从 γ=0.1 改变到 γ=0.8，并考虑具有各种 *TTRT* 值的情况。此外，假设分组长度在此间隔和后面的间隔中是均匀分布的。

为了洞察在存在分组的情况下 *TTRT* 如何影响容量共享，对于每一个流r_i^S，r_i^A，$1 \le i \le 6$，*TTRT* 从 3 到 20ms 之间变化，当 γ=0.3，考虑速率样本的平均值（速率样本是通过对在 500ms 的时间间隔接收到的字节数进行平均而收集的）。还考虑通过实例化获得的稳定状态速率（上面的等式 5）。正如所料，随着 *TTRT* 增大，模拟结果重叠了通过等式 5 计算出的稳定状态速率。随着 *TTRT* 缩小，在异步流中逐渐地产生了不公平性；具体来说，具有较小的 α 因子的流不能获得所希望的速率，在具有较高的 α 因子的异步流之间共享了大部分速率，还有部分被重定向到同步流。

假设 α 因子和最大分组长度是输入数据，则异步带宽取决于 *TTRT*（直接）和 γ（相反地）。

根据经验法则，以这样的方式选择 *TTRT*，以便每一个流实际接收到的速率与通过应用等式 5 计算出的速率没有很明显的差别。具体来说，假设 *m* 是具有最小 α 因子的异步流，$\overline{\tau_m}$是平均长度分组的传输时间。那么，*TTRT* 应该不小于

$$TTRT^* = \frac{1}{1-\gamma}\cdot\left(\frac{\overline{\tau_m}}{2}\cdot\frac{1+\sum_{j=1}^{N_A}\alpha_j}{\alpha_m}+\tau^S\right) \qquad (10)$$

即，通过使流 *m* 的稳态异步带宽等于其平均分组传输时间的一半而获得该值。应用于异步流的分组配置策略允许传输分组，只要剩余的传输时间大于或等于分组长度的一半。

一般而言，可以通过考虑下列情况来执行 TTRT 的选择：TTRT 是预期的回合持续时间。因此，它越低，调度的响应速度就越快。可以预期的是，TTRT 的较低的值为速率得到保证的流产生较低的延迟限制。另一方面，当 TTRT 可以与平均和最大分组传输时间相比较时，在异步流上可能会产生不公平性，因为异步带宽可能有时候不足够大，难以传输单个分组。作为经验法则，可以计算出并采用应该避免不公平性的对 TTRT 的下限。

利用动态变化的负载进行性能评估

已经预先调查了当所有流经常地被积压时，由 GTTSD 共享容

量。此外，还可以显示，积压的异步流的对之间的相对速率比率随着其他流的积压状态的改变而保持不变。为了这样做，模拟了具有与前面所描述的相同的流集合的节点；使用了相同的 α 因子（即，$\alpha_{1,2}$=0.25，$\alpha_{3,4}$=0.5，$\alpha_{5,6}$=1.0），而 *TTRT*=20ms，γ=0.3。

通过改变 i)同步流上的提供的负载，ii)异步流 No.5 上提供的负载，而其他的保持始终积压，执行两组实验。

在第一组实验中，在重负载条件下，在同步流中提供的负载在从 0.1 到 1.2 的稳定状态速率之间变化，这在 0.35 到 4.6Mbps 之间。

实验显示了，不管同步流上的提供的负载如何变化，速率比 r_i^A/r_6^A $1\le i\le 6$ 都保持不变，并等于相应的 α 因子比率 α_i/α_6。

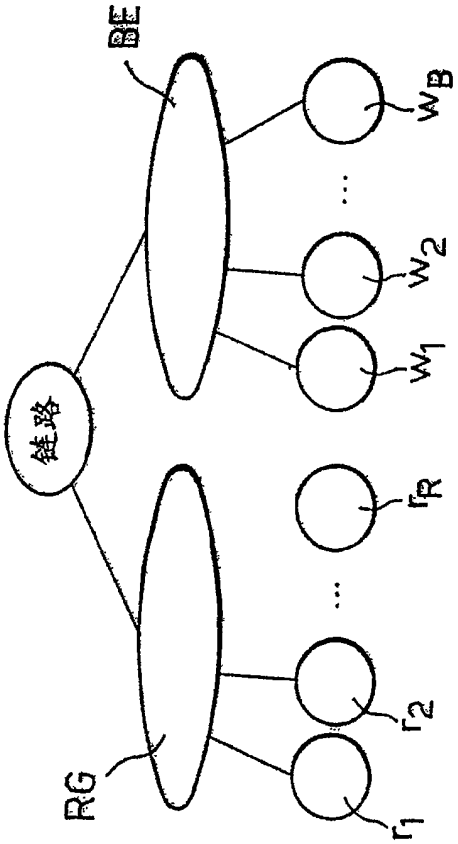
在第二组实验中，在重负载条件下，在异步流 5 中提供的负载在从 0.1 到 1.2 的稳定状态速率之间变化，这在 0.2 到 2.3Mbps 之间。积压的异步流共享了没有被负载不足的异步流使用的大部分容量，而只有小得多的部分被同步流（其速率实际从 3.5 到 3.7Mbps 之间变化，即，小于 6%）利用。不管负载不足的异步流上提供的负载如何变化，对于其他流，速率比 r_i^A/r_6^A 保持不变，并等于相应的 α 因子比 α_i/α_6。通过改变其他参数，如在所有流上均匀的异步流上的分组长度分布、所有流上的双模态、混合的和 TTRT 值，不断地重复实验。

所获得的结果与前面所描述的那些一致。如此，积压的异步流共享取决于它们的 α 因子的带宽，不管其他（同步和异步的）流的积压状态中的动态变化如何。

这还适用于利用异步流的不同分组化策略的可能的替代实施例和/或网络处理器上的 GTTSD 的可能的实现方式。

因此，在不违背本发明的基本原理的情况下，在不偏离随后的权利要求中所定义的本发明的范围的情况下，细节和实施例相对于通过示例所描述的内容可以有显著的不同。

图1



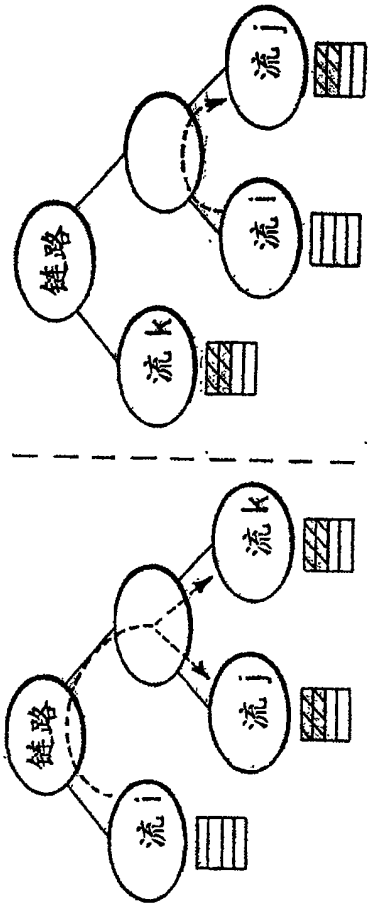


图 2

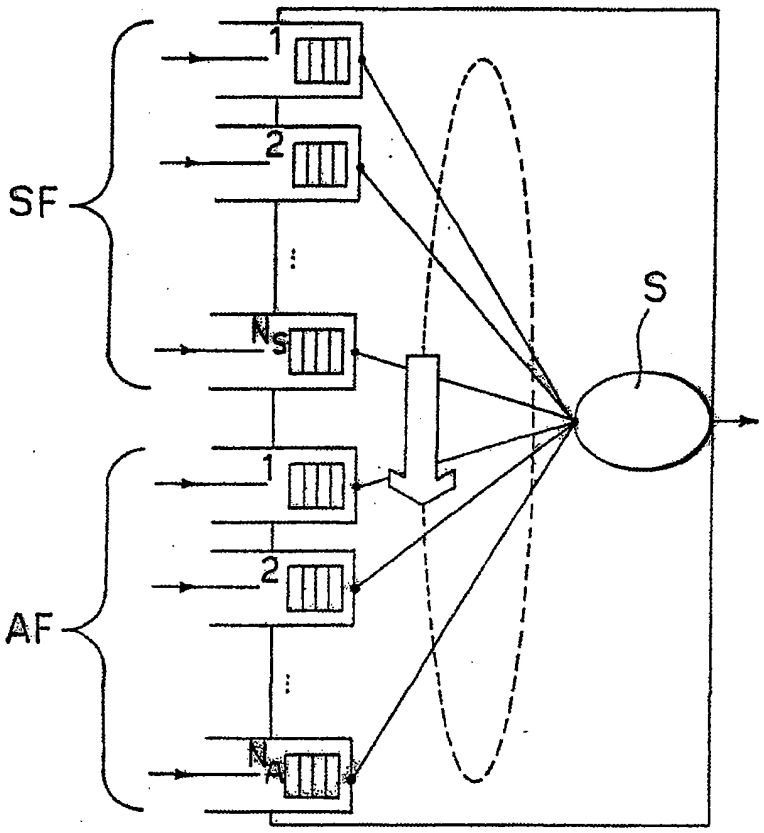


图 3

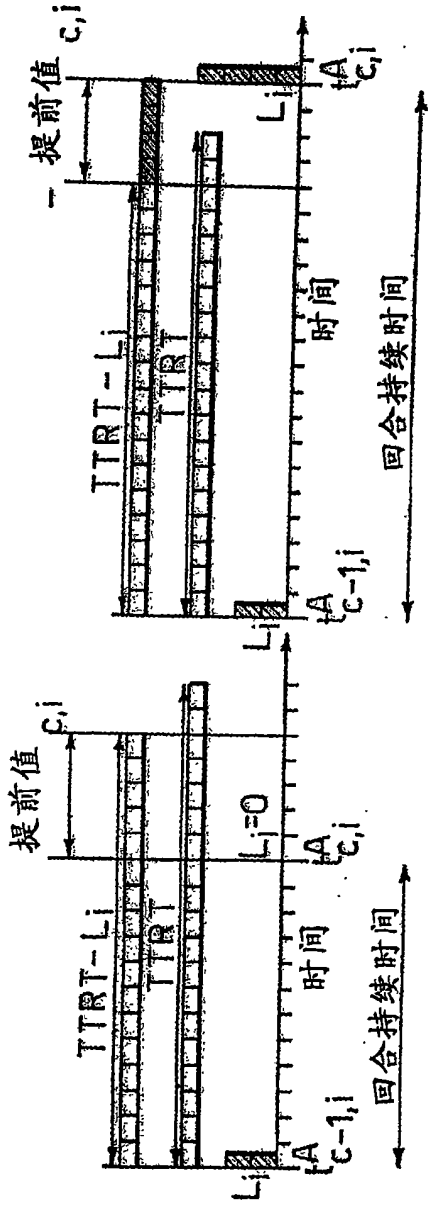

- 提前值 c,i
TTRT - Li
TTRT
Li
tA c,i
时间
tA c-1,i
回合持续时间
提前值 c,i
TTRT - Li
TTRT
Li=0
Li
tA c,i
时间
tA c-1,i
回合持续时间

图 4

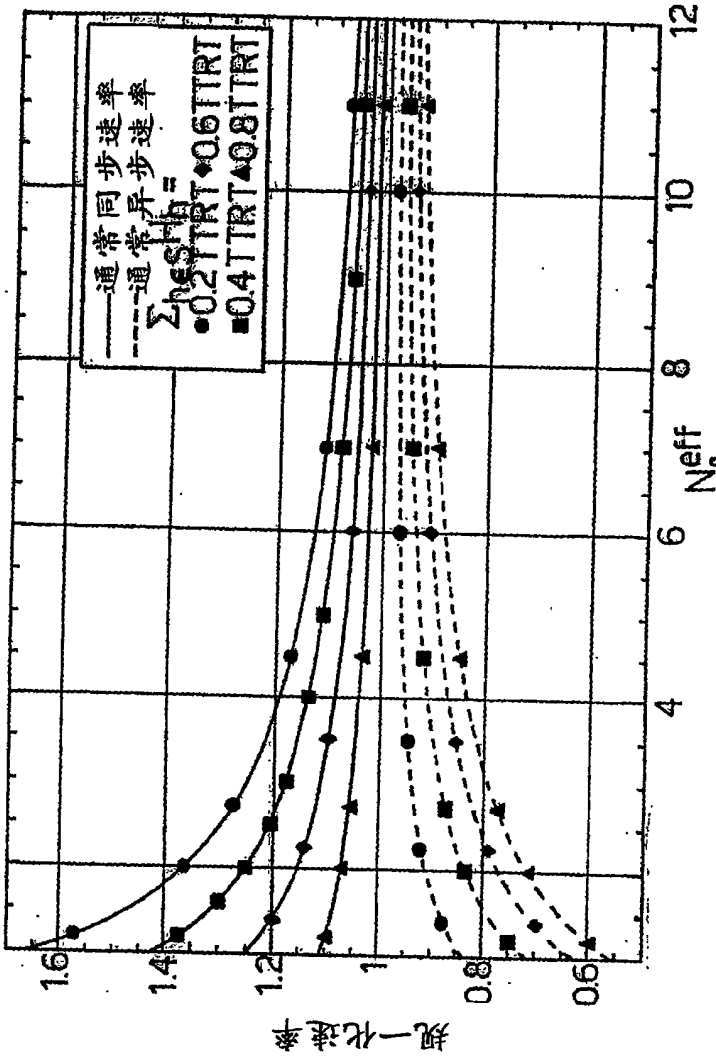

规一化速率
1.6
1.4
1.2
1
0.8
0.6
4
6
8
10
12
N_A^{eff}
通常同步速率
通常异步速率
$\Sigma_{h\in S_{TH}}$=
•0.2TTRT•0.6TTRT
■0.4TTRT▲0.8TTRT

图 5

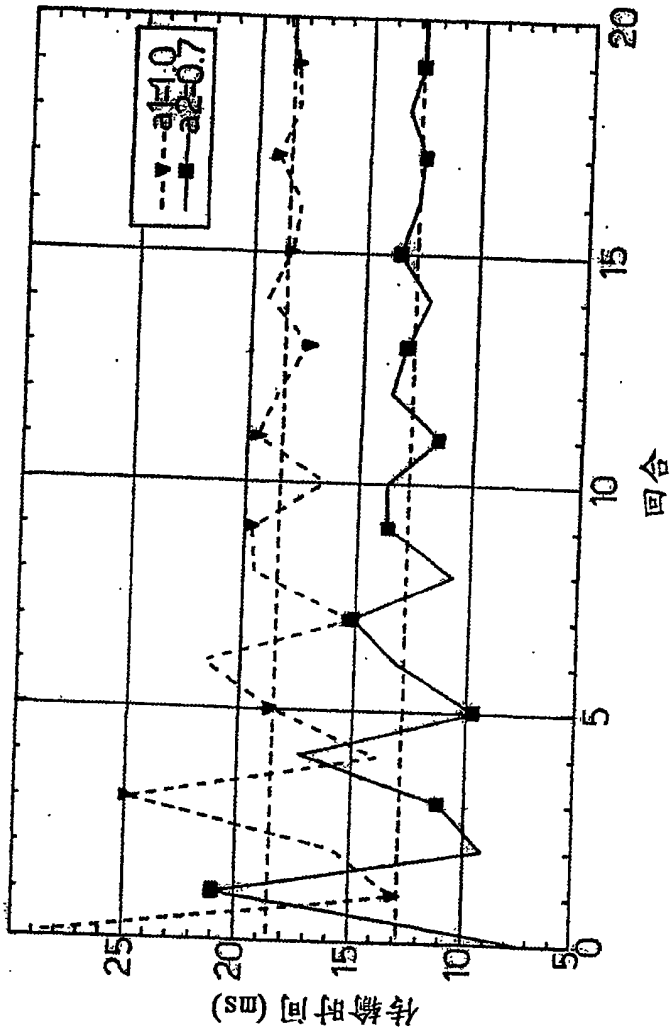


图6