第 34 卷 第 13 期 计 算 机 工 程 2008 年 7 月
Vol.34 No.13 Computer Engineering July 2008
·网络与通信· 文章编号：1000—3428(2008)13—0092—03 文献标识码：A 中图分类号：TN915.02

基于消耗量的无状态核心节点调度算法

孙学康，顾畹仪
(北京邮电大学光通信与光波技术教育部重点实验室，北京 100876)

摘 要：随着多媒体应用的不断发展，未来的核心网络将负责提供大容量、无状态的数据调度，而流量控制等内容则由边缘节点来完成。基于上述思路提出一种与消耗量有关的无状态核心节点调度算法，对该算法下的数据调度性能进行仿真分析，结果显示与用户所缴费用相对应的消耗量和网络所提供的服务性能大致呈线性关系。

关键词：调度算法；无状态；token消耗量

Scheduling Algorithm of Stateless Core Nodes Based on Cost

SUN Xue-kang, GU Wan-yi
(Optic Telecommunication & Technology Key Lab of Ministry Education, Beijing University of Posts and Telecommunications, Beijing 100876)

【Abstract】As the development of the multi-media applications, it is considered to provide one kind of high rate and digital schema among stateless core nodes. The congestion control scheme is completed by edges nodes. A scheduling algorithm of stateless core nodes based on cost is proposed, and the simulation analysis is given according to this algorithm. Test result shows that the cost would be nearly linear with the service performance provided in the network.

【Key words】scheduling algorithm; stateless; token cost

1 概述

目前的因特网提供的是一种“尽力而为”的服务，使因特网处于无状态，其中的路由器根据配置策略以及所维护的路由表进行分组转发。这种“尽力而为”的服务无法满足各种多媒体应用需求，因而如何规划和改造现有的网络，使之适应业务的多样性和突发性的变化，是研究领域和业界亟待解决的问题。

调度机制的研究是其中非常重要的内容。所谓调度机制是指数据分组接受服务的次序。最简单的调度算法是先进先出(First In First Out, FIFO)法，也称为“去尾(droptail)”法。这种算法是按分组到达的顺序进行调度，它不支持区分服务。随后人们提出随机早期检测(Random Early Detection, RED)、加权公平排队(Weighted Fair Queue, WFQ)和逆差轮询算法(Deficit Round Robin, DRR)等算法[1]。它们均存在不足，故而出现了一些改进算法，如弱实时约束的加权公平队列(Worst-case WFQ, W2FQ)和开始时间公平队列(Self-clocked Fair Queuing, SFQ)算法，这些算法在实现高度公平的前提下所需的计算量相对较小[2-3]。

本文提出了一种基于消耗量的核心节点调度算法，它将与用户所缴费用相应的 token 消耗量参数引入算法之中，使用户向网络服务提供商所缴纳费用与其所获得的服务性能相关联，同时又能使核心节点流调度与流的状态无关，并且其性能近似于 WFQ。

2 基于 token 消耗量的网络调度策略

2.1 算法的适用环境

为了简化网络结构和节点功能，建议采用两级结构网络，即核心层和接入层网络。核心层网络由处于无状态的核心节点构成。接入层网络由处于流保持状态的边缘节点组成。在此每个流的准确速率、延时等状态信息插入其分组头中，当携带这些状态信息的分组经过核心网络时，核心路由器仅根据每个分组头所携带的描述标记来进行分组转发，而无需保存流信息。

由于在 IPv6 基本报头中使用了与 QoS 直接相关的服务元素，包括流和相应的流标签，因此可以根据实时多媒体业务的质量等级来确定所需的网络带宽。

2.2 基于 token 消耗量的两级网络调度策略

token 消耗量是用来描述每经过一个节点或链路所产生的消耗。该参数的大小与业务所需的带宽等因素有关。对一个两级网络而言，每一个边缘节点中均存放一个 token 消耗量维护表，表中描述了与其他各边缘节点间进行数据传输时所需的基本 token 消耗值。如无该信息，此边缘节点将通过发送探测分组以获取该信息。数据分组发送过程如图 1 所示。

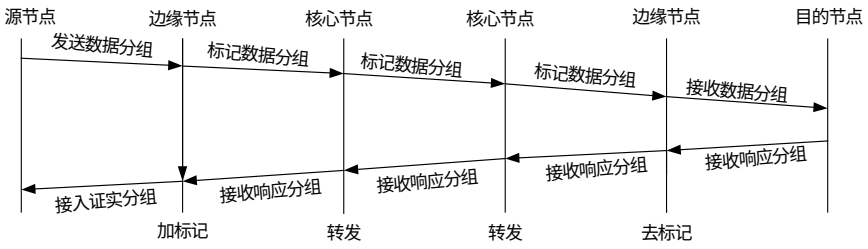

图 1 数据分组发送过程

基金项目：国家“863”计划基金资助项目(2006AA01Z246)
作者简介：孙学康(1959－)，女，副教授，主研方向：宽带通信网；顾畹仪，教授
收稿日期：2007-07-31 **E-mail：**sunxuekang6395@sina.com

(1)当需进行数据发送时，源节点首先向与之相连接的边缘节点发送数据分组。该分组中包括源地址、目的地址和业务等级等状态信息。

(2)边缘节点负责完成流识别、速率估算和加标记等工作。标记内容包括当前剩余token总量和该流token消耗量基值N_{ci}的倍数N。N值的大小由业务速率决定。当边缘节点收到来自其下属源节点数据分组后，根据现有token消耗量维护表中所存放的历史信息、所需网络带宽和业务等级，确定与目的节点相连接的边缘节点之间进行数据传输时所需的token消耗量初始值，并为发送分组赋值(所赋值可以大于、等于或小于所预计的token总消耗量)。此时当前剩余token总量等于token初始值与出链路所需token值之差，然后以保持流状态的方式向与之连接的核心节点发送数据分组。

这里需要说明的是，由于不同等级的业务，其token消耗量不同，因此$N_{ci}=mN_c$。其中，m是反映业务等级的系数；N_c是单位流量下的token消耗量基值。

(3)当核心节点收到上述信息时，首先根据配置策略和当前网络状态，计算出该节点和出链路的 token 消耗量，然后将其从分组所携带的 token 量中扣除，并进行标记更新，再向下一跳进行数据分组的转发。

(4)每经过一个核心节点均按步骤(3)进行标记更新操作，直至与目的节点相连接的边缘节点。

(5)由该边缘节点去除标记等额外信息，将分组送至目的节点。数据流传送完毕后将自动产生一个接收响应分组，并将其送往与之相连接的边缘节点，由该节点向与源节点相连的边缘节点发送接收响应分组，最后由此边缘节点向源节点发送一个接入证实分组以表示对端正常接收。接收响应分组中还包含分组传送过程中所剩余的 token 消耗量。这样与源节点相连的边缘节点可根据接收响应分组中所携带的该信息估计出分组的平均 token 消耗量，并与 token 消耗量维护列表中存放的对应参数进行对比，必要时更新该参数。

(6)当网络发生拥塞，源节点无法正常收到接入证实分组，超时后便根据拥塞控制策略决定如何降低发送数据流速率。当网络拥塞解除时，则重复步骤(1)。

当源端的边缘节点所赋 token 总量小于所预计的 token 总量时，此分组到达某节点时，所剩余 token 量等于 0 或小于该节点为其提供服务所需的消耗量。网络则将为其提供尽力而为的服务，可见此时的服务质量将有所下降。

3 基于消耗量无状态核心节点调度算法

3.1 算法模型

该算法是一种基于WFQ系列的改进算法[4]，它一方面消除了WFQ系列算法中对流的依赖性，既能提供数据流间的区分，又能使其性能近似地达到WFQ系列的性能；另一方面能在流上体现出服务质量与用户所缴费用之间的差别。在链路上采用多个RED队列加权轮询的调度策略作为区分调度的基础，以此适应不同媒体业务的延时要求。

基于消耗量的无状态核心节点调度算法的处理过程包括流速率及token消耗速率检测、带宽分配、丢弃概率的计算和更新标记 4 项内容。从图 2 中可以看出，由多个边缘节点发送的分组首先由参量统计与提取模块接收。在该模块中将根据所接收分组标记中所指示的参数N和分组的业务等级来确定所需的相应token消耗量NN_{ci}，并将结果分别送入token消耗量速率估算、token消耗量调协和丢包统计模块。M-RED队列模块根据业务等级对来自参量统计与提取模块的数据流进行排序。不同业务服务等级所排的队列不同，所需支付的token消耗量不同。token消耗量速率估算模块负责进行token消耗量速率计算，并将结果送至带宽分配和发送模块。带宽分配模块的功能是进行基于token量的带宽分配计算，其结果被送入丢包统计模块。该模块将根据token消耗量速率和带宽估算结果计算出预计的丢包概率，并以此控制M-RED队列模块和token调协模块。在M-RED队列模块中将根据数据流的丢包率进行排序，然后采用多队列轮询方式将数据分组发往标记模块。在标记模块中进行当前剩余token量的更新，然后将更新后的数据分组存放在缓存器中。

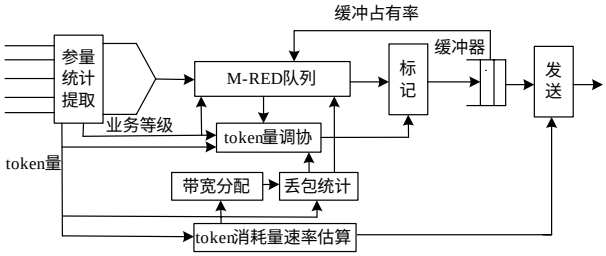

图 2　无状态核心节点功能框图

该算法可根据缓冲器占有率来调整队列长度。当业务量达到某种程度时，可通过丢包来缓解网络拥塞，但这种操作将直接影响业务质量，因此在 token 量调协模块中依据数据流的业务等级进行合理的调整。同样，当聚合流的速率大于链路最大容量时，会出现丢包的现象，因此也需对所消耗的 token 量作出调整。发送模块将根据 token 量估算速率来进行数据发送。具体估算算法如下：

(1)token 消耗量速率估算

token 消耗量速率估算是根据前一时刻所接收数据分组的 token 消耗量估算速率来预测当前即将到达数据分组的 token 消耗量估算速率。本算法是采用指数平滑来对 token 消耗量估算速率进行计算。其计算式如下：

$$v_{\text{token}i}^{\ n}=(1-\mathrm{e}^{-\frac{T_i^k}{K}})\frac{N\cdot N_{Ci}^{\ k}}{T_i^k}+\mathrm{e}^{-\frac{T_i^k}{T_i^k}}v_{\text{token}i}^{\ n-1} \tag{1}$$

其中，$v_{\text{token}i}^{\ n}$和$v_{\text{token}i}^{\ n-1}$分别代表第 i 流在 n 时刻和 n-1 时刻数据分组 token 消耗量的估算速率；T_i^k 代表第 i 流第 k 和第 k-1 个数据分组到达的时间差；K 为常量；N_{Ci} 代表第 i 流的 token 消耗量基值，通常一个节点的 token 消耗量是 N_{Ci} 的 N 倍，N 可取 0 或任意正整数。当数据分组的当前剩余 token 量小于当前节点所需 token 消耗量时，当前数据分组所能支付的 token 消耗量为 0。此时的 token 总量不足，因此由该跳开始网络只能为其提供尽力而为的服务。

(2)带宽分配

由前面的假设可知，链路所提供的带宽是以所付出的 token 量的多少为依据的，例如实时性要求高的交互电视业务，为保证其业务质量所需提供的传输带宽较宽，所需消耗的 token 量也较大。其定义如下：

$$B_{\text{token}i}^{\ n}=B_{\text{token}i}^{\ n-1}(\frac{C}{v_{\text{token}}}) \tag{2}$$

其中，$B_{\text{token}i}^{\ n}$和$B_{\text{token}i}^{\ n-1}$分别表示 n 和 n-1 时刻第 i 个数据流的分配带宽；C 为链路带宽；v_{token} 为聚合流速率，数值上等于各数据流速率之和。

(3)丢包概率的计算

$$p_i=\max(0,1-\frac{N\cdot B_{\text{token}}}{v_{\text{token}i}}) \tag{3}$$

其中，B_{token} 表示数据流的平均带宽；v_{token_i} 为第 i 数据流的 token 消耗量平均估算速率。

3.2 基于 NS2 的仿真结果分析

(1)基于消耗量的无状态核心节点调度算法

当采用不同调度算法时，网络性能会发生变化，因而利用NS2 模拟建立一个简单拓扑结构。如图 3 所示，C为核心节点，e_1和e_d为边缘节点。其中，e_1分别与源节点S_1, S_2相连接；e_d与目的节点D相连接。

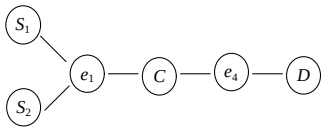

图 3　网络拓扑结构

为了能够直接对 2 个数据流所携带的 token 量进行控制，同时又能降低程序操作的复杂度，在 NS2 中采用直接对数据包赋 token 值的方法，采用 CBR 流方式来设定 token 量与流速率的比例关系。

首先假设流S_1(图表中用$flow_1$表示)的token消耗量是流S_2($flow_2$)的 2 倍，并且S_2的速率为 1 Mb/s时所需的token量为 1，各链路带宽均为 10 Mb/s速率的带宽。在链路的延时设置上，S_1, S_2分别与e_1间的链路延时设置为 0.01 ms，e_1, e_d与C间的链路延时均设置为 0.1 ms，e_d与D间的链路延时设置为 0.01 ms。其中，MRED采用 2 队列，队列长度限制为 100。从输出跟踪文件中可提取到相关数据，其结果如图 4 所示。

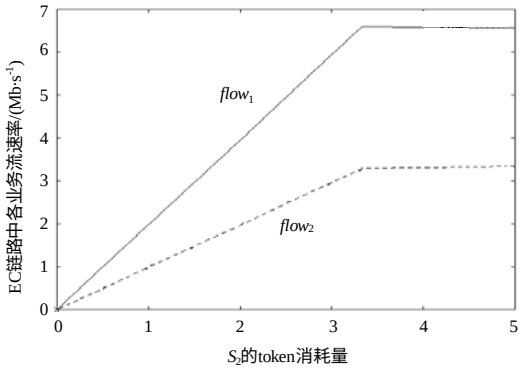

图 4　各流带宽分配与S_2的token消耗量的关系

可以看出，当各数据流通过边缘接入节点汇聚到核心节点C时，$flow_1$与$flow_2$近似成 2 倍的关系。由链路总带宽限制，当S_2的token消耗量大于 3.3 时，会出现丢包的现象，此时各流所分配的瞬时带宽如图 5 所示(实线与虚线分别表示S_2=5 时$flow_1$与$flow_2$的平均带宽)。尽管所提供的带宽相对理想情况均有所下降，但为各数据分组所提供的带宽与各流拥有的token消耗量之间近似呈线性关系，如图 6 所示。

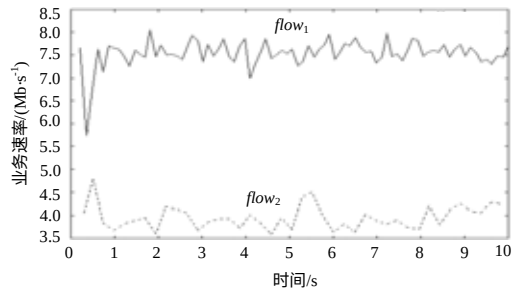

图 5　S_2=5 时各流带宽的瞬时分配关系

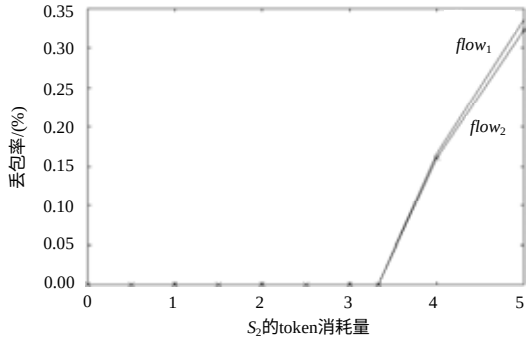

图 6　丢包率与 $S2$ 的 token 消耗量之间的关系

(2)调度算法的比较

在传统的 Internet 网络中可供使用的调度算法有多种，不同的调度算法对网络性能影响不同。为了正确评价本文所提出的基于 token 消耗量的无状态核心节点调度算法，笔者又进行了 FIFO-Droptail, DRR 和 SFQ 等模式下的网络端到端的试验仿真。

表 1 给出了试验环境及条件与前面所述的条件相同的情况下各算法的技术比较结果。从表 1 可见，本算法的实施并没有对网络的端到端延时性能产生不良影响，并且还有一定的改进。这是因为在该算法中采用多队列排序，并对队列长度进行了控制，从而降低了数据分组的平均排队时间。

表 1　各算法CBR流的技术比较(S_2=3 时)

算法	平均延时/s	丢包率/(%)
Droptail-FIFO	0.069 745	0.006 077
DRR	0.083 357	0.001 954
SFQ	0.069 757	0.007 370
MRED-tdiffserv	0.042 498	0.007 765

4　结束语

调度算法是下一代互联网中的一个关键的内容。本文提出了一种基于消耗量的无状态核心节点调度算法。与传统的算法相比，本算法既可以在快速、大容量的网络环境下提供数据流之间服务等级的区分，又能根据网络流量分布情况自动在流上体现用户所缴费用的差别。

由于该算法是针对两级结构的网络提出来的，因此在核心节点上算法的时间和空间复杂度与业务流量无关，这一特点正与大容量的网络特性相吻合，能够更好地满足 QoS 的要求。

参考文献

[1] Altman E, Jimenez T. Simulation Analysis of Red with Short Lived TCP Connections[J]. Computer Networks, 2004, 44(5): 631-641.

[2] Parehk A K, Gallager R G. A Generalized Processor Sharing Approach to Flow Control in Integrated Service Networks: Single-node Case[J]. IEEE/ACM Transactions on Networking 1993, 1(3): 344-357。

[3] Lee J F, Sun Yeali, Chen Mengchan. On Maximum Rate Control of Weighted Fair Scheduling for Transactional Systems[C]//Proc. of the 24th IEE International Real-time Systems Symposium. [S. l.]: IEEE Press, 2003: 335-344.

[4] Stoica I. Stateless Core: A Scalable Approach for Quality of Service in the Internet[D]. Pittsburgh, USA: Carnegie Mellon University, 2000.