文章编号:1001 - 9081(2005)04 - 0886 - 03

支持区分服务的自适应队列调度算法

刘 辉,夏汉铸,刘 翔
(重庆邮电学院 重庆信科设计有限公司,重庆 400065)
(lxll35@163.com)

摘 要:分析了在 DiffServ 模型下的 WRR 和 DWRR 调度算法,提出了一种基于 WRR 的改进的调度算法 AWRR(ADWRR),同时提出了 AWRR 调度算法的实现过程。该算法根据网络中各业务数据的实际流量动态的调整其对应的权值。因此 AWRR 不仅能提供 QoS 保证,而且还能根据该节点的实际负载状况,提供动态的带宽分配。

关键词:IP 区分服务;服务质量;加权轮循调度;自适应加权轮循
中图分类号: TP393 **文献标识码**:A

An adaptive queue scheduling mechanism for supporting Diffserv

LIU Hui, XIA Han-zhu, LIU Xiang
(*Chongqing Information Technology Designing CO., LTD, Chongqing University of Posts and Telecommunications, Chongqing* 400065, *China*)

Abstract: WRR and DWRR in the architecture of DiffSever was discussed. And based on WRR, an adaptive weighted round-robin (AWRR) and how to carry out this algorithm was presented. By this algorithm, different packets have different weight, and schedule packets according to this. So that AWRR not only can obtain the requirement of QoS, but also can according to the node's condition dynamically assigning the bandwidth and the bandwidth resource sharing.

Key words: IP DiffServ; quality of service; weighted round robin; adaptive weighted round robin

0 引言

传统的 Internet 只提供"尽力而为"(best_effort)的服务。随着 Internet 和各种 IP 业务的发展,尤其是音频、视频等多媒体业务的迅速增长,要求现在的网络不仅要提供传统的数据业务,还要提供对带宽、延迟要求较高的新业务的支持,即要求现有的 IP 网络对不同的业务实现区分,同时还要提供一定的 QoS 保证。为了解决这个问题,IETF 提出了两种服务模型:IntServ[1]和 DiffServ[2]。在 IntServ 模型下,每一个中间节点必须维持一定的状态信息,而且需要定时更新,所以在网络规模较大时,其扩展性就受到较大的限制。在 DiffServ 模型下,其大部分复杂的功能(分类、流量调节、标记等)在边缘节点完成,中间节点(核心节点)只需根据不同的服务类别采用不同的 PHB(Per-Hop Behavior,每一跳行为)转发分组。在 DiffServ 模型下中间节点如何在提供一定的 QoS 保证下调度分组,是当前的研究热点。虽然目前提出了许多能提供 QoS 保证的调度算法,但这些算法基本都不能根据网络负载实际状况地变化来动态的改变调度策略,这样必然会带来流量大的服务等级的数据包大量丢失。如何改进现有的调度算法,使其不仅能实现业务区分和提供一定的 QoS 保证,而且能动态的改变调度策略,在实际网络中就显得特别重要。

1 DiffServ 模型概述

IP DiffServ 模型主要包括边缘节点的流量调节器(Traffic Conditioner)和核心节点的 PHB。

边缘节点的流量调节器是根据流量调节协定(TCA)来进行流量调节的。当数据包到达 DiffServ 域的边缘节点时,根据 SLA(Service Level Agreements,服务等级协定:规定了分组分类、标记规则(重标记规则)、流量描述等),首先经过分类器,对数据包进行分类,再根据测量器的测量结果确定数据包标记(或重新标记)和整形(延迟和丢包处理),然后将该数据包发送出去。

根据到达数据包的 DSCP (Differentiated Services Codepoint)值,DiffServ 域的核心节点采用不同的 PHB 调度分组。现有的调度策略包括 First-In-First-Out (FIFO), Priority Queuing (PQ), Fair Queuing (FQ), Weighted Fair Queuing (WFQ), Weighted Round-Robin Queuing (WRR), Deficit Weighted Round-robin Queuing(DWRR)[3]。其中支持区分服务的队列调度算法主要包括:PQ、FQ、WFQ、WRR、DWRR。

2 WRR 和 DWRR

在 DiffServ 模型下,要实现业务区分,核心节点必须采用一定的队列调度策略。

WRR 调度算法[4,5]的基本思想就是给每一个子队列分配一个权值,然后根据权值来调度不同的子队列中的数据包。该算法的具体实现过程为每一个子队列 i 分配一个相应的权值 Wi(该权值是基于分组的个数),按轮循的方式调度,如果子队列 i 中的分组数大于 Wi,则该子队列调度 Wi 个分组,若子队列 i 中的分组数小于 Wi,则子队列中的分组全部被调度(空队列略过不调度),然后在转到下一个队列进行调度,如此轮流调度。对于 WRR 调度算法而言,在每一个分组(数据包)的大小一致时,能提供很好的公平性,但如果分组的大小不一致,就会对分组较小的队列带来不公平。为了克服这种不公平性,提出了 DWRR 调度算法[6],该算法给每一个队列分配的权值不是基于分组的个数,而是基于比特数,每一次调度剩余但又不够调度一个分组的比特数将累积到下一次可调

收稿日期:2004 - 09 - 02;**修订日期**:2004 - 11 - 23

作者简介:刘辉(1968 -),男,四川仪陇人,高级工程师,硕士研究生,主要研究方向:光网络传输; 夏汉铸(1974 -),男,湖北黄冈人,硕士研究生,主要研究方向:光互联网技术; 刘翔(1973 -),男,四川仪陇人,工程师,主要研究方向:光网络传输技术.

度的比特数。相对于 WRR 而言，DWRR 提供了一种更细粒度的带宽分配。

WRR 或 DWRR 调度算法的权值是固定分配的，如果某一服务等级的分组到达速率较大，而其他服务等级的分组到达比较平稳时，由于每一服务等级的存储容量有限，可能会导致到达速率较大的服务等级的分组大量丢失，而其他服务等级的数据存储单元大量空闲，造成不必要的分组丢失。但是如果采用存储单元的动态分配机制，可能会引起其他服务等级的分组丢失，不恰当的存储单元动态分配机制还可能会导致低优先级的数据饥饿现象。因此，如何改进现有的 WRR 和 DWRR 算法，不仅能实现业务区分，同时又能根据网络负载的实际状况实现动态的带宽资源分配，就显得特别重要。

3　AWRR 算法描述

由于权值固定分配的特性，WRR 或 DWRR 调度算法不能根据网络负载的实际状况，实时地调整各子队列的权值。AWRR 算法正是基于 WRR 或 DWRR 的这种缺陷提出的，为了简单起见，本文中提到的 AWRR 是 WRR 算法的改进算法，ADWRR 是 DWRR 算法的改进算法，如无特殊说明，一般用 AWRR 算法表示。AWRR 算法基本思想是根据测量器的测量结果来动态改变其相应队列 i 的权值 IWi。其功能框图如图 1所示。

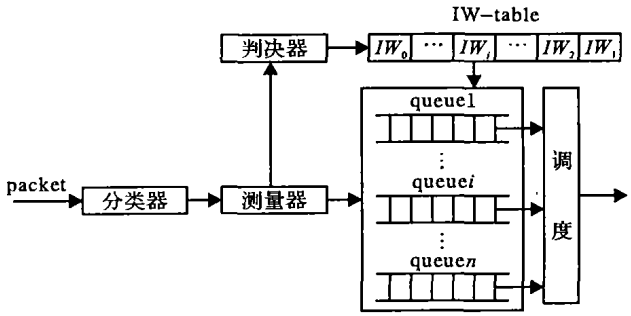

图 1　AWRR 功能框图

AWRR 功能框图中，每到达一个分组，首先根据分组的 DSCP 域，分类器对其进行分类，根据分类的结果测量对应服务类别的数据到达速率，同时将该分组插到相应的子队列中。将测量器的测量结果发送到判决器，判决器根据测量结果决定如何改变 IW-table 中各种服务类别的对应权值，从而影响调度器的调度行为。

3.1　变量说明

IWi：子队列 i 初始化权值，每一次调度完成，IWi 作为下一次子队列 i 的权值 Wi。

Wi：队列 i 实际对应的权值，对于子队列 i 而言，每次子队列调度一个数据包，Wi 减 1。若 $Wi = 0$，表示赋予子队列 i 的权值已调度完成，转到下一个子队列调度数据包。

SWi：WRR 或 DWRR 算法分配给子队列 i 的固定权值。

ΔWi：表示子队列 i 相对于固定权值的增量，根据测量器的测量结果决定 ΔWi 的大小，每一次 ΔWi 的改变都影响 IWi 的值，即 $IWi = SWi + \Delta Wi$。

$Rminthi$：表示子队列 i 的速率下限门限值，其值可根据网络中各种业务流量的实际情况来决定。

$Rmaxthi$：表示子队列 i 的速率上限门限值，其值可根据网络中各种业务流量的实际情况来决定。

ΔWi、$Rminthi$ 和 $Rmaxthi$ 一起决定子队列 i 的权值变化情况。

3.2　确定 IWi

如图 1 所示：当一个分组到达时，分类器首先进行分类以便插入到相应的子队列中，然后测量器测量该数据所属类别的到达速率 $Rate$（由于测量器测得的数据应为一个给定的时间间隔内的平均值，为简单起见，本文中均用 $Rate$ 来表示测量器测得的平均值），判决器根据得到的该类别数据到达的速率 $Rate$ 进行如下动作：

1）如果该类别数据到达速率在某一个下限门限值以下（$Rate < Rminthi$），$\Delta Wi = 0$。

2）如果该类别数据到达速率在速率下限门限值和速率上限门限值之间（$Rminthi < Rate < Rmaxthi$），ΔWi 根据速率 $Rate$ 的变化线形增加：

$$\Delta Wi = \frac{Rate - Rminthi}{Rmaxthi - Rminthi}\max(\Delta Wi)$$

3）如果其到达速率大于上限门限值（$Rate > Rmaxthi$），$\Delta Wi = \max(\Delta Wi)$。

4）$IWi = SWi + \Delta Wi$。

对 WRR 调度算法，子队列 i 对应权的值为在一个轮循周期内子队列 i 可调度的分组数，所以 ΔWi 应为一个离散的值；而对 DWRR 调度算法，其权值为可调度的比特数，则 ΔWi 可线形增加。图 2、3 分别表示 AWRR、ADWRR 算法中 ΔWi 与速率的变化关系。

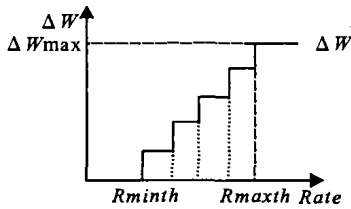

图 2　AWRR 的 ΔWi-$Rate$

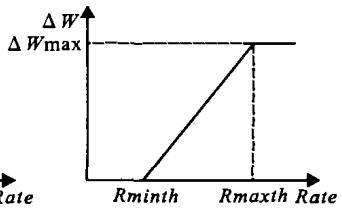

图 3　ADWRR 的 ΔWi-$Rate$

说明：

1）为了保证各业务区分，对于每一个子队列 i 的 $\max(\Delta Wi)$ 必须有一定的限制，以保证子队列 i 增加后的权重不会对其他的子队列造成较大的影响，即对于每一个子队列的 $\max(\Delta Wi)$ 都有一个上限。

2）对于每一个子队列的 $Rminth$ 和 $Rmaxth$ 可以选择不同的值。可以从以下两个方面来考虑：

① 基于权重：对于权重较大的子队列 i，可以使其 $Rminthi$ 和 $Rmaxthi$ 较小并使 $\max(\Delta Wi)$ 较大，这样可使 ΔWi 变化的较快，以能满足其数据速度较大的要求，如图 4 所示。

② 基于优先等级：对于优先级较大的子队列 i，可以使其 $Rminthi$ 和 $Rmaxthi$ 较小，并使 $\max(\Delta Wi)$ 较大，使 ΔWi 变化的较快，使优先级较高的业务能首先得到保证，如图 5 所示。

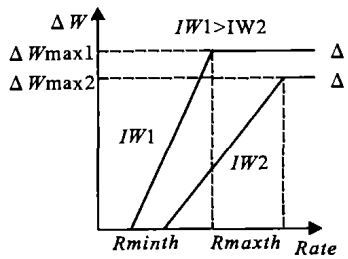

图 4　基于权重的 ΔWi-$Rate$

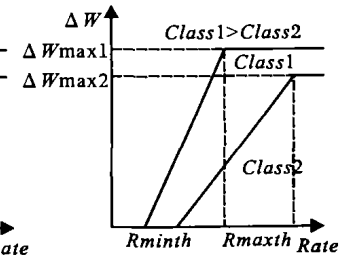

图 5　基于优先级的 ΔWi-$Rate$

3.3　AWRR 调度算法实现（基于 WRR 的 AWRR 调度算法）

如 3.2 节的描述，每一次改变的 IWi 的值都写到 IW-table 表中，而调度器每完成一次循环后，读出 IW-table 表中各对应的 IWi 作为下一次的各对应的子队列 i 的权值，然后根据各对应的子队列 i 的权值按 WRR 调度算法来调度每一个子队列的数据包。

1）初始化 IW-table 表。根据 IW-table 中的对应权值，对各子队列的权值进行赋值）：$Wi = IWi(i = 1,2\cdots n)$；

2）当有分组到达时，根据测量器的测量结果影响判决器

决定如何改变 ΔWi，并将判决器的判决结果写入到 *IW-table* 表中；

3）从IW-table表读取权值，$Wi = IWi + \Delta Wi(i = 1,2,\cdots,n)$；

4）根据得到的每一个子队列 i 的权值 Wi，按WRR算法的调度规则调度每一个子队列中的数据包，每调度一个数据包 Wi 的值减1，若 $Wi = 0$ 或子队列中没有数据包可调度，跳到下一个子队列 $(i + 1)$ 调度。

5）轮流调度结束转到步骤3）。

如果调度器采用ADWRR调度算法，其算法与AWRR调度算法基本一致。它们之间的区别在于：① 权值 Wi 是指每次最多可调度的比特数；② 在以上算法步骤3中 $Wi = Wi + IWi$。

4 仿真结果分析

为了简单起见，在仿真中只有一种业务的流量发生改变，其他业务的流量保持不变的情况。仿真的网络拓扑结构如图6，瓶颈链路带宽为2Mbps，source0 提供1Mbps 的 BE 业务的背景流量，source2 提供 0.4Mbps 的 EF 业务的背景流量，source1 提供流量可变的 AF 业务。在 WRR 和 AWRR 算法下各业务 BE、AF、EF 的初始权值比为 2∶4∶8。

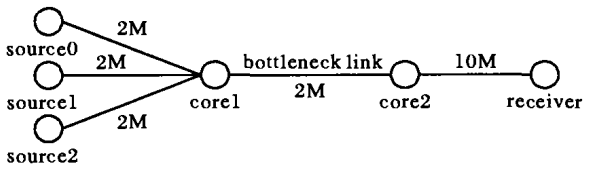

图6 仿真网络拓扑结构图

在 WRR 算法和 AWRR 算法下，AF 业务的丢包率与 AF 业务的负载关系如图7所示。

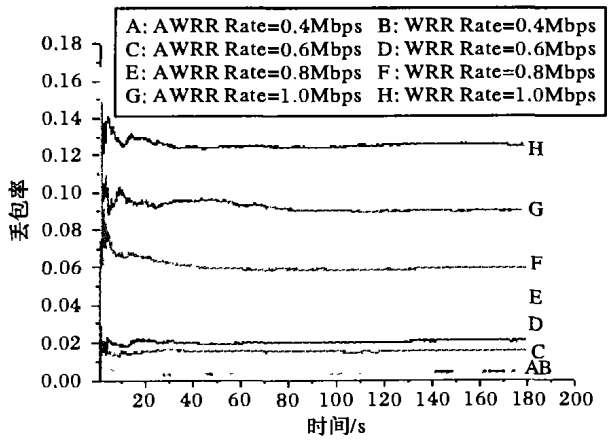

图7 WRR 和 AWRR 下 AF 的丢包率随负载的变化

由图7可以看出：在 AF 业务流量超过一定的范围时（大于0.6Mbps），采用 AWRR，可以明显的减少其丢包率，如：AF 业务的流量为 0.8 和 1.0Mbps，图7中曲线 E、F、G、H。

在以上条件不变的情况下，AF 业务提供的负载为 0.6Mbps，分别采用 AWRR 和 WRR 算法时各种业务丢包率如图8、图9所示。

从图8和图9可以看出：在只有一种业务的数据流量增加，其他业务的数据流量基本保持不变时，如采用 WRR 算法 EF、AF、BE 的丢包率分别为 1.1%、6%、17.4%，总的丢包率为 10%；而采用 AWRR 算法 EF、AF、BE 的丢包率分别为 1.1%、4%、18.6%，总的丢包率为 10%。可见：采用 AWRR 算法能明显的减少该业务的丢包率，而且还减少了总的丢包率；同时不会对其他业务带来较大的影响，特别是对高等级的业务（如 EF）。

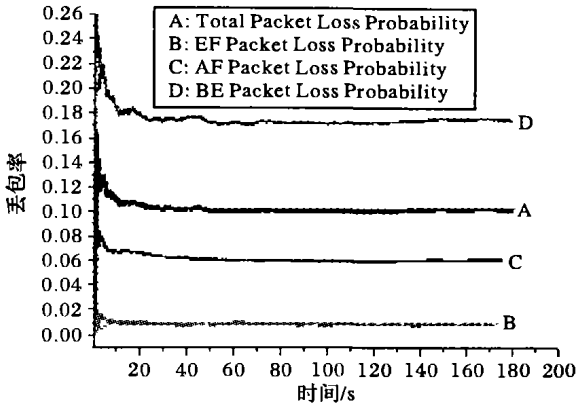

图8 AWRR 算法的各种业务丢包率

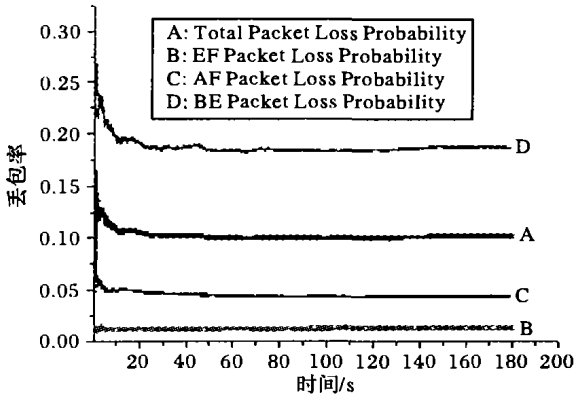

图9 WRR 算法的各种业务丢包率

在实际网络中，如果有两种或两种以上业务的流量发生改变时，可能会对流量平稳的业务带来一定的负面影响。由于该算法是通过业务的流量变化来动态的改变其相应的权值，如果有多种业务的流量增加时，那么这些业务的相应的权值也相应的增加。这样，可能会带来流量平稳的业务的丢包率增加。AWRR 算法采用限制 max（ΔWi）来尽量减少这种负面影响，由于限制了任何一种业务可增加的权值的最大值，从而达到了限制该业务对其他业务流的影响。同时由于在该算法中引入了测量器，可以根据测量器的测量结果，采用一定的策略（如：接入控制）来限制各种不同业务的最大流量，也可以达到减少负面影响的效果。

参考文献：

[1] BRADEN R, CLARK D, SHENKER S. Integrated Services in the Internet Architecture: An Overview, IETF RFC 1633[S]. 1994.

[2] BLACK D, BLAKE S, CARLSON M, *et al.* An Architecture for Differentiated Services, IETF RFC 2475[S]. 1998.

[3] SEMERIA C . Supporting Differentiated Service Classes : Queue Scheduling Disciplines [EB/OL]. http://www. juniper. net/solutions/literature/white_papers/200020. pdf, 2001.

[4] BENNETT J , ZHANG H . Hierarchical Packet Fair Queuing Algorithm[A]. Proceedings of ACM SIGCOMM'96[C]. 1996. 675 - 689.

[5] BENNETT J , ZHANG H . W2FQ : Worst _ case Fair Weighted Fair Queueing[A]. Proceeding of IEEE INFOCOM'96[C]. 1996. 120 - 128.

[6] SHEEDHAR M, VARGHESE G. Efficient Fair Queue Using Deficit Round Robin[A]. Proceedins of ACM. SIGNCOMM'95[C]. 1995, 25 (4): 231 - 242.

[7] ZHENG W. Internet Qos: Architectures and Mechanisms for Quality of Service[M]. Morgan Kaufmann, 2001.

[8] KALEVI K. Differentiated Services for the Internet[M]. New Riders Publishing, 1999.