区分服务网络基于覆盖的拥塞管理方案*

庞 斌[1+], 高 文[1,2,3]

[1](中国科学院 计算技术研究所,北京 100080)

[2](哈尔滨工业大学 计算机科学与工程系,黑龙江 哈尔滨 150001)

[3](中国科学院 研究生院,北京 100039)

An Overlay-Based Congestion Management Mechanism in Differentiated Services Networks

PANG Bin[1+], GAO Wen[1,2,3]

[1](Institute of Computing Technology, The Chinese Academy of Sciences, Beijing 100080, China)

[2](Department of Computer Science, Harbin Institute of Technology, Harbin 150001, China)

[3](Graduate School, The Chinese Academy of Sciences, Beijing 100039, China)

+ Corresponding author: Phn: 86-10-82649316, Fax 86-10-82649298, E-mail: binpang@ict.ac.cn

http://www.jdl.ac.cn

Received 2002-04-22; Accepted 2002-07-02

Pang B, Gao W. An overlay-based congestion management mechanism in differentiated services networks. *Journal of Software*, 2003,14(2):305~311.

Abstract: An overlay-based congestion management mechanism for assured forwarding in differentiated services (DiffServ) network is presented in this paper. In the proposed scheme, a control packet is sent from the ingress to the egress router every fixed time interval. The ingress router employs a simple additive increase and explicitly decrease algorithm to adjust the aggregate's sending rate according to the QoS (quality of services) information reflected from the egress routers. For the performance evaluation of the proposed scheme, the simulation of the (proportional) fairness of aggregates traffic and packet loss ratio is presented.

Key words: differentiated services; quality of services; congestion management; rate control; assured forwarding

摘 要: 提出一种面向区分服务网络确保转发的覆盖式拥塞管理方案,基本思想是利用控制分组在网络的入口和出口节点之间传递网络服务质量的状态信息,入口节点利用加性增加和显性降低的算法调节聚集通信量的发送速率.实验结果表明,与标准的区分服务网络相比,该方案能在聚集之间公平地分配带宽并能显著地降低分组丢失率.

关键词: 区分服务;服务质量;拥塞管理;速率控制;确保转发

中图法分类号: TP393 **文献标识码**: A

* Supported by the National Natural Science Foundation of China under Grant No.69983008 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2001AA112100 (国家高技术研究发展计划); the Knowledge Innovation Program in CAS under Grant No.KGCXZ-103 (中国科学院知识创新工程)

第一作者简介: 庞斌(1971－),男,山东临沂人,博士生,主要研究领域为计算机网络,多媒体通信技术.

为了在 Internet 上提供服务质量(quality of services,简称 QoS)保证,IETF 提出区分服务(differentiated services,简称 DiffServ)结构[1].在网络的边缘节点,DiffServ 根据与用户的协议标记 IP 分组头部的 DSCP(DiffServ code point)[2]字段并将分组划分为不同的通信量聚集;核心节点根据 DSCP 标记决定通信量聚集的逐节点行为(per hop behavior,简称 PHB)[3,4].与面向连接的集成服务(integrated services,简称 IntServ)[5]结构相比,DiffServ 不需要在核心路由器中保存每个传输流的状态和处理复杂的信令协议,因此具有良好的可扩展性,并被认为是下一代 Internet QoS 结构的基础.

现在 IETF 已完成加速转发(expedited forwarding,简称 EF)PHB[3]和确保转发(assured forwarding,简称 AF)PHB[4]的标准化工作.EF PHB 为用户提供低延迟、低抖动、低丢失率和保证带宽的服务.AF PHB 能提供比"尽力而为"服务更好的服务,即在网络发生拥塞时仍能为用户提供一定量的预约带宽.具体而言,DiffServ 网络的边缘节点负责标记和监视聚集的传输流.预约带宽以内的传输流被标记为 IN,而超过预约带宽的传输流被标记为 OUT.网络的核心节点利用主动队列管理算法(如 RIO[6])保护 IN 传输流,即在网络发生拥塞时,在丢弃 IN 分组之前,优先丢弃 OUT 分组.本文主要讨论 AF PHB.

IETF 虽然完成了 DiffServ 体系结构和 PHB 的定义,但 DiffServ 网络仍然需要额外控制功能(如接纳控制、带宽代理和拥塞控制等)的支持,才能在网络中提供真正意义的服务质量保证.在所有这些控制功能中,拥塞的控制和避免具有重要的作用,这是由于:(1) 多媒体实时应用(如视频会议和 VoIP)通常采用 UDP 传输协议,会产生大量的无答复传输流.这些传输流在网络发生拥塞时不能适当地调整自己的发送速率,因此会抢占 TCP 连接的带宽,造成网络资源的不公平分配;(2) 传统的基于窗口的端到端 TCP 拥塞控制[7]不能适应 DiffServ 网络环境.TCP 拥塞控制机制以分组丢弃作为拥塞发生的信号,当发送端确认发出的数据分组已丢失时,通过减少拥塞窗口的方法降低发送速率,以此缓解拥塞.但随着网络规模的不断扩大,从网络节点发生拥塞(分组丢弃)到发送端降低流量需要较长的周期,而同时发送端仍以高速率发送数据,就会造成更大规模的拥塞,因此不能完全依靠所有的发送端控制拥塞.

现有 DiffServ 网络的拥塞管理(即同时包括拥塞恢复和拥塞避免机制[8])方案主要面向公平带宽分配和预防分组丢失.在带宽分配方面,Nandy 等人[9]提出一种聚集流量控制算法,它根据控制 TCP 连接(control TCP connection)的分组丢失来调节聚集的发送速率;文献[10]提出一种新型的聚集标记器,主要解决各聚集之间按比例公平共享多余的可用带宽问题.所谓的按比例公平共享是指聚集获得的网络可用带宽与该聚集的保证信息速率(committed information rate,简称 CIR)成正比[11].这些方案能保证聚集之间(按比例)公平共享网络带宽和提高资源的利用率,但是存在以下不足:(1) 分组丢失不可避免;(2) 没有考虑聚集内各传输流之间的带宽分配问题.在预防分组丢失方面,文献[12]提出一种基于边缘节点的拥塞管理方案,它利用核心节点的反馈和边缘节点的速率控制来减少分组丢失率.Harrison 和 Kalyanaraman 提出利用边到边(edge-to-edge)的通信量控制功能来提供无丢失的 TCP 服务[13].这些方案都是利用网络的边缘节点实现覆盖式拥塞控制和降低丢失率,但是没有考虑到在 DiffServ 网络环境下聚集或传输流之间的带宽分配问题.

为解决以上问题,本文提出一种新型的面向 DiffServ 网络的拥塞管理方案——基于覆盖的拥塞管理(DiffServ overlay-based congestion management,简称 DSOCM).在 DSOCM 方案中,网络入口节点每隔一定的时间向对应的出口节点发送 QoS 控制分组.当收到出口节点的反馈信息之后,入口节点根据 QoS 信息调节聚集的发送速率.

与传统的端到端控制相比,DSOCM 具有一些优点.(1) DSOCM 独立于特定的传输层协议.在核心网络中,DSOCM 在聚集通信量的水平上提供拥塞控制,不关心数据分组的传输控制协议类型.(2) DSOCM 能对将要发生的拥塞作出快速响应.采用 DSOCM 的网络,边缘节点对拥塞的响应时间与一个 DiffServ 域中边缘节点之间的往返时间有关,因此响应时间远远小于端到端控制方案.(3) DSOCM 能降低网络的分组丢失率.DSOCM 通过信令协议在边缘节点间交换 QoS 状态信息.这样,在核心网络的瓶颈链路发生拥塞之前,边缘节点就降低了发送速率,从而减少分组的丢失率.(4) DSOCM 不需要在核心节点中保存每个传输流的状态信息.核心节点利用支持 ECN[14]的 RIO 队列管理算法检测将要发生的拥塞和随机标记到达分组的方法通知边缘节点.

本文第 1 节阐述 DSOCM 基本机制和系统结构.第 2 节给出 DSOCM 的主要组成部分和算法,包括决策算

法、增加/降低算法和决策周期.第 3 节通过实验评价 DSOCM 的性能.第 4 节总结全文并指出进一步研究方向.

1 系统结构

1.1 基本机制

DSOCM 机制涉及一个 DiffServ 域的入口和出口节点,分别作为 QoS 控制分组的发送方和接收方.发送方每隔时间 τ 向接收方发送一个控制分组.控制分组的作用是在入口和出口节点之间交换 QoS 信息.接收方收到控制分组后,计算 IN 和 OUT 分组的输出速率、已接收的 IN 和 OUT 分组的个数以及 ECN 被标记的 IN 和 OUT 分组的个数.出口节点将这些 QoS 参数写到控制分组的相应字段中,并将控制分组发回入口节点.我们假设在 DiffServ 域中控制分组具有最高的优先级且不会被网络节点丢弃.当入口节点收到返回的控制分组后,根据分组中的 QoS 参数调节聚集的发送速率.

1.2 RIO算法

网络核心节点的队列管理算法是 RIO,它可以看成是两个 RED 算法的组合,分别对应 IN 和 OUT 分组.RIO 算法的主要参数由两组类似 RED[15]的阈值参数组成:⟨minth_in, maxth_in, maxp_in⟩和⟨minth_out, maxth_out, maxp_out⟩,分别用来计算 IN 和 OUT 分组的丢弃概率.在一般情况下,为了在丢弃 IN 分组之前先丢弃 OUT 分组,OUT 分组的队列长度的上下限要低于 IN 分组.此外,在计算平均队列长度时,OUT 分组是基于队列中所有的(IN 和 OUT)分组,而 IN 分组只根据 IN 分组的个数.

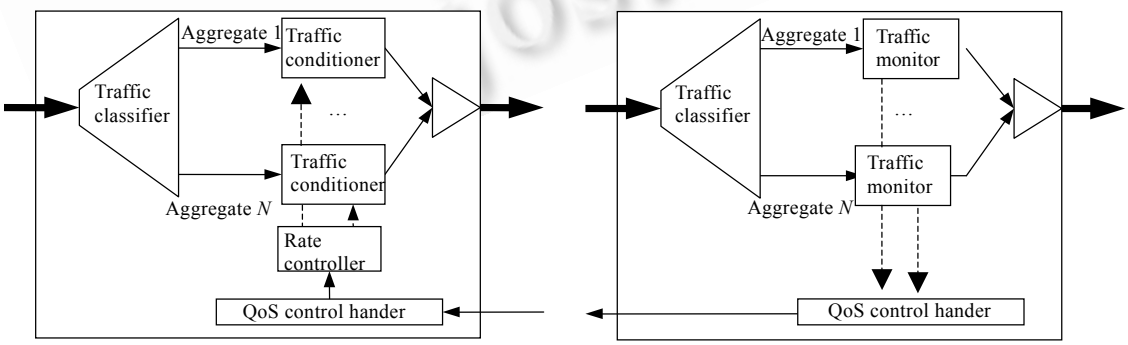
在本文中,我们假设传输路径上的每个节点都支持 ECN.当网络节点检测到拥塞将要发生时,标记 IN 或 OUT 分组头部的 ECN 位.

1.3 入口节点

入口节点的结构如图 1(a)所示.在入口节点,到达的分组首先根据分组包头的 DSCP 标记将分组分为不同的聚集.通信量调节器负责监视聚集的速率,并通过整形器保证聚集的发送速率不会超过预约带宽.超过预约带宽的通信量将被丢弃或重新标记为低优先级分组.速率控制器根据收到的 QoS 控制分组调节通信量调节器的输出速率.

1.4 出口节点

出口节点的结构与入口接点类似(如图 1(b)所示).到达的分组经过分类后进入通信量监视器.通信量监视器的功能是:(1) 利用 TSW(time sliding window)计量器[6]估计 IN 和 OUT 分组的到达速率;(2) 记录到达的 IN 和 OUT 分组的数量;(3) 记录到达的带 ECN 标记的 IN 和 OUT 分组的数量.



(a) Ingress node (b) Egress node

(a) 入口节点 (b) 出口节点

Fig.1 System structure

图 1 系统结构

2 核心算法

设计一个有效的拥塞管理方案一般需要 3 个核心算法[16]:决策算法、增加/降低算法和决策周期.本节还将讨论如何在一个聚集内部实现公平带宽分配的问题.

2.1 决策算法

决策算法的作用是根据 QoS 控制分组的参数决定聚集速率的调节方向(增加或降低).在 DSOCM 中,核心节点负责监视输出队列和随机标记到达分组的 ECN 位.因此,本方案的拥塞控制是在核心节点的队列长度到达一定阈值时予以触发的.决策算法如下所示:

入口接点每隔时间 τ 收到由出口节点返回的 QoS 控制分组

if N_S= N_R then

  if EN_R^{IN} >0 then

    decrease the sending rate of IN- and OUT-packets

  else if EN_R^{OUT} >0 then

    decrease the sending rate of OUT-packets

  else

    increase the sending rate

else

  decrease the sending rate of IN- and OUT-packets

符号 N_S 表示入口节点发送分组的个数,N_R 表示出口节点接收分组的个数,EN_R^{IN} 表示 IN 分组中被标记 ECN 的分组的个数,EN_R^{OUT} 表示 OUT 分组中被标记 ECN 的分组的个数.

2.2 速率增加/降低算法

在拥塞管理方案中,如何根据当前网络资源状况决定增加和降低发送速率是一个重要的问题.在入口节点,我们利用一个简单的算法来增加聚集的发送速率 R_S:

$$R_S = R_S \times (1+\alpha),\quad 0<\alpha<1.$$

DSOCM 根据 QoS 控制分组的参数降低 IN 或 OUT 分组的发送速率:

if EN_R^{OUT} >0 then

$$R_S = R_R^{IN} + R_R^{OUT} \times \beta,\ 0<\beta<1$$

R_R^{IN} 和 R_R^{OUT} 分别是 IN 和 OUT 分组在出口节点的输出速率.

if EN_R^{IN} >0 or $N_S \neq N_R$ then

$$R_S = R_R^{IN} \times \gamma,\ 0<\gamma<1.$$

2.3 决策周期

决策周期是指改变聚集发送速率的时间间隔.在 DSOCM 中,当入口节点收到由出口节点返回的 QoS 控制分组后,开始按照决策函数和增加/降低函数改变速率.因此,决策周期由发送控制分组的时间间隔 τ 决定.

2.4 聚集内的带宽分配

现有的 DiffServ 拥塞管理方案主要集中在保证聚集间的公平性等问题上,而忽视了聚集内各 TCP 连接间的加权公平速率分配.对一个包含 N 个 TCP 连接的聚集,第 i 个 TCP 连接的加权公平分配速率可由下式得到:

$$r_i = \frac{w_i}{\sum_{j=1}^{N} w_j} \times R_S,$$

其中 w_i 是第 i 个 TCP 连接的权值,R_S 是聚集的发送速率.

在 DSOCM 中,为了减少分组丢失和 TCP 连接发送速率的波动,我们没有采用在入口节点直接丢弃分组的方法来降低 TCP 连接的发送速率,而是采用直接窗口调整的方法[17].当入口节点收到第 i 个 TCP 连接的应答分

组时,按下式得到新的广告窗口:

$$W_i^{new} = r_i \times RTT_i ,$$

其中 r_i 是第 i 个 TCP 连接的加权公平分配速率,RTT_i 是往返时间.入口节点根据下式设置应答分组的窗口字段:

$$W_i = \min(W_i^{new}, W_i^{ad}) ,$$

其中 W_i^{ad} 是窗口字段的初始值.

对 UDP 传输流,我们直接利用加权公平分配速率调节通信量调节器中整形器的速率.

3 性能评价

本节将利用网络仿真器 NS-2[18]评价 DSOCM 的性能.实验用的网络拓扑结构如图 2 所示,它由 4 个边缘节点(E1,E2,E3,E4)和两个核心节点(C1,C2)构成.边缘节点与主机(s1~s20 和 d1~d20)直接相连. 节点 C1 和 C2 之间链路的队列管理算法是 RIO, IN 和 OUT 分组的参数分别为⟨15,30,0.02⟩和⟨5,15,0.1⟩.经过边缘节点 E1 和 E2 的聚集通信量由 TCP 或 UDP 传输流构成.分组长度为 500 字节.在实验中,我们将比较 DSOCM 和标准 DiffServ 的性能.在标准 DiffServ 网络中,边缘节点用令牌桶调节和标记传输流.

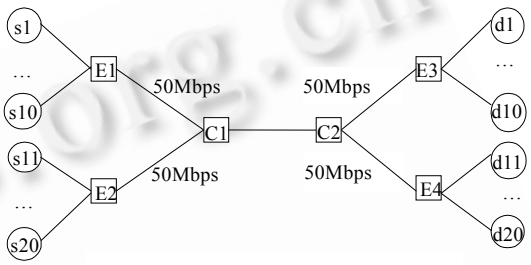


Fig. 2 Network topology

图 2 网络拓扑结构

负载比率 r 的定义如下:

$$r = \frac{BW_{\text{request}}}{BW_{\text{reserve}}} ,$$

其中 BW_{request} 是通信量请求的带宽,BW_{reserve} 是通信量规定中的预约带宽.当负载比大于 1 时,超出预约带宽的分组被标识为 OUT 分组.

3.1 TCP和UDP聚集之间的公平性

实验用到两种聚集通信量:聚集 1 由 UDP 传输流构成;聚集 2 由 3 个 TCP 连接构成.核心节点 C1 和 C2 之间的带宽是 5Mbps.当聚集 1 的速率从 1Mbps 增大到 5Mbps 时,各聚集获得的网络带宽如图 3 所示(标记为“DSOCM Agg1”和“DSOCM Agg2”的曲线).

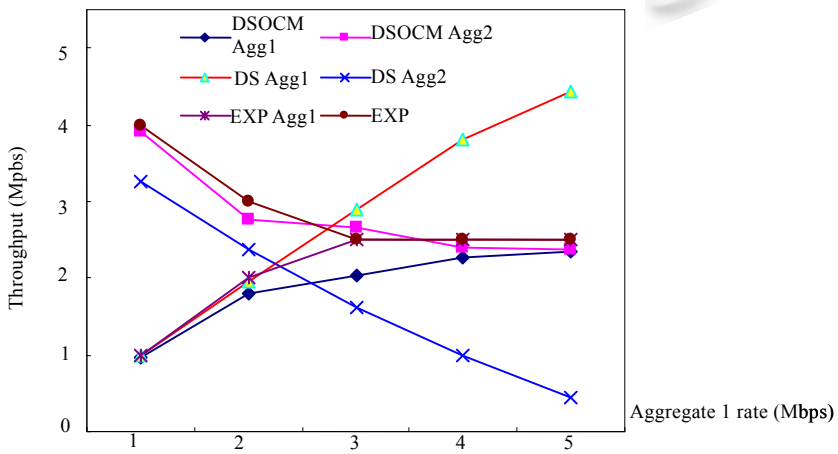


Fig.3 The fairness between TCP and UDP aggregates

图 3 TCP 和 UDP 聚集之间的公平性

结果显示,采用 DSOCM 的 DiffServ 网络,网络带宽基本上公平地分配给 TCP 和 UDP 聚集,接近理想情况

下的期望值(标记为“EXP Agg1”和“EXP Agg2”的曲线).当采用标准的 DiffServ 网络时(标记为“DS Agg1”和“DS Agg2”的曲线),随着 UDP 聚集发送速率的增大,TCP 聚集的带宽不断减少.

3.2 聚集之间按比例公平分配带宽

在一个按比例共享系统中,一个聚集获得的带宽与它的 CIR 有关.文献[11]指出,聚集获得的带宽与 RTT、分组丢弃概率、分组长度和 CIR 有关.在这里,我们只考虑 CIR 对聚集带宽的影响.

实验用到两种聚集通信量:聚集 1 和聚集 2 都由 10 个 TCP 连接构成.核心节点 C1 和 C2 之间的带宽是 10Mbps.聚集 1 的 CIR 从 2Mbps 增大到 8Mbps,聚集 2 的 CIR 保持为 2Mbps.各聚集获得的带宽见表 1.由表 1 可以看出,在供应水平从 30%增大到 100%的情况下,DSOCM 能保证在聚集之间按比例公平地分配带宽,同时总的带宽利用率保持在 93%以上.

Table 1 Bandwidth allocation among aggregates

表 1 聚集间的带宽分配

Provision level (%)	Target rate (Mbps)		Achieved rate (Mbps)		Link goodput (Mbps)
	Agg1	Agg2	Agg1	Agg2	
100	8	2	8	1.58	9.58
90	7	2	7.47	2.06	9.53
80	6	2	7.34	2.4	9.74
70	5	2	7.1	2.6	9.7
60	4	2	6.7	3.1	9.8
50	3	2	5.9	3.9	9.8
40	2	2	5.0	4.8	9.8
30	1	2	2.9	6.4	9.3

3.3 分组丢失率

表 2 给出在相同网络环境下 DSOCM 和标准 DiffServ 结构的分组丢失率的比较.由表 2 可以看出,随着负载比率的增大,DSOCM 的丢失率明显低于标准 DiffServ 结构.由于 DSOCM 能利用拥塞状态反馈信息调整边缘节点的发送速率,因此当负载比率到达 1.7 时,核心节点的分组丢失率仍然限制在 0.23%,比标准 DiffServ 结构的分组丢失率低约 40%.

Table 2 Packet loss ratio

表 2 分组丢失率

Load ratio	DSOCM (%)	DIFFSERV (%)
1.1	0	9
1.2	0	16
1.3	0.04	23
1.4	0.06	28
1.5	0.19	33
1.6	0.22	37
1.7	0.23	41

4 结 论

本文提出一种新型的面向区分服务网络的覆盖式拥塞管理方案,它利用网络边缘节点间的信息交换和聚集速率控制的方法保证聚集间的公平性,并可降低分组丢失率.该方案包括 3 个主要组成部分:决策算法、增加/降低算法和决策周期.实验结果表明,与标准的区分服务网络相比,该方案能在聚集之间(按比例)公平地分配带宽,并能显著地降低分组丢失率.

在今后的工作中,我们将考虑其他因素(如分组长度、传输流的个数和丢失率等)对聚集带宽分配的影响.

References:

[1] Blake S, Black D, Carison M, Davies E, Wang Z, Weiss W. An architecture for differentiated services. IETF RFC 2475, 1998.

[2] Nichols K, Blake S, Baker F, Black D. Definition of the differentiated services field (DS Field) in the IPv4 and IPv6 headers. IETF RFC 2474, 1998.

[3] Jacobson V, Nichols K, Poduri K. An expedited forwarding PHB. IETF RFC 2598, 1999.

[4] Heinanen J., Baker F, Weiss W, Wroclawski J. Assured forwarding PHB group. IETF RFC 2597, 1999.

[5] Braden R, Clark D, Shenker S. Integrated services in the Internet architecture. IETF RFC 1633, 1994.

[6] Clark D, Fang W. Explicit allocation of best-effort packet delivery service. IEEE/ACM Transactions on Networking, 1998,6(4): 362~373.

[7] Jacobson V. Congestion avoidance and control. ACM Computer Communication Review, 1988,18(4):314~329

[8] Jain R. Myths about congestion management in high speed networks. Internetworking: Research and Experience, 1992,3(3): 101~113.

[9] Nandy B, Ethridge J, Lakas A, Chapman A. Aggregate flow control: improving assurance for differentiated services network. In: Sengupta B, ed. Proceedings of the IEEE INFOCOM. Anchorage: IEEE Communications Society, 2001. 1340~1349.

[10] Su H, Atiquzzaman M. ItswTCM: a new aggregate marker to improve fairness in DiffServ. In: Chang FR, ed. Proceedings of the IEEE GLOBECOM. San Antonio, Texas, USA: IEEE Communications Society, 2001. 1841~1846.

[11] Baines M, Seddigh N, Nandy B, Pieda P, Devetsikiotis M. Using TCP models to understand bandwidth assurance in a differentiated services network. In: Chang FR, ed. Proceedings of the IEEE GLOBECOM. San Antonio: IEEE Communications Society, 2001. 1800~1805.

[12] Chiruvolu G, Charcranoon S. An efficient edge-based congestion management for a differentiated services domain. In: Chou W, ed. Proceedings of the IEEE ICCCN. Miami: IEEE Communications Society, 2000. 75~80.

[13] Harrison D, Kalyanaraman S. Edge-to-Edge traffic control for the Internet. Technical Report, RPI ECSE Network Laboratory ECSE_NET-2000-I, 2000. http://networks.ecse.rpi.edu/~harrisod/index2.html.

[14] Floyd S. TCP and explicit congestion notification. ACM Computer Communication Review, 1994,24(10):8~23.

[15] Floyd S, Jacobson V. Random early detection gateways for congestion avoidance. IEEE/ACM Transactions on Networking, 1993, 1(4):397~413.

[16] Jain R. A delay-based approach for congestion avoidance in interconnected heterogeneous computer network. ACM Computer Communication Review, 1989,19(5):56~71.

[17] Kapoor R, Casetti C, Gerla M. Core-Stateless fair bandwidth allocation for TCP flow. In: Glisic SG, ed. Proceedings of the IEE ICC. Helsinki: IEEE Communications Society, 2001. 146~150.

[18] NS Simulator. http://www.isi.edu/nsnam/ns.