Ad-hoc网络PSD拥塞控制算法

陈 亮[1,2] 张 宏[1]
(南京理工大学计算机科学与技术学院 南京 210094)[1] (南通纺织职业技术学院信息系 南通 226007)[2]

摘 要 神经元PID算法能较好地控制队列长度,但其神经元增益对被控对象的状态较为敏感,基于试凑和经验的设定往往使控制效果难以保证。基于TCP拥塞窗口加法增大、乘法减小原则和排队机制,推导出拥塞窗口与丢弃概率、队列长度的微分方程,再对方程进行线性化,获得Ad-hoc网络TCP/AQM控制系统模型。基于该模型,将递推计算修正功能引入神经元PID,设计了一种神经元自适应PSD的AQM。该算法可以在线调整神经元增益。NS仿真表明,在无线分组丢失、突发流及链路容量变化的Ad-hoc网络中,PSD队列管理性能优于神经元PID。

关键词 无线自组网,拥塞控制,主动队列管理,比例求和微分

中图法分类号 TP393 **文献标识码** A

PSD Congestion Control Algorithm in Ad-hoc Network

CHEN Liang[1,2] ZHANG Hong[1]
(School of Computer Science and Technology, Nanjing University Science and Technology, Nanjing 210094, China)[1]
(Information Department, Nantong Textile Vocational Technology College, Nantong 226007, China)[2]

Abstract Neuron PID algorithm can control queue length successfully, but its neuron gain is sensitive for controlled object state. So it is difficult to guarantee the control performance because the gain depends on experience and trial method. Congestion window size, loss probability and queue length differential equations were deduced based on TCP window additive-increase and multiplicative-decrease (AIMD) principle and queuing mechanism. TCP/AQM control model was obtained in Ad-hoc network through the equations linearization. Then it introduced recursion and modification gain to neuron PID based on the model. Finally, a neuron adaptive proportional summation differential (PSD) AQM was proposed. PSD algorithm can modify neuron gain dynamically. NS simulations demonstrate that PSD queue management performance is better than neuron PID under conditions of wireless packet loss, sudden flow and different link capacity.

Keywords Ad-hoc network, Congestion control, AQM, PSD

1 引言

Ad-hoc网络是由移动终端组成的多跳自组织对等网络,网络中的每个节点既是收发数据的主机,也可能作为转发数据的路由器[1]。无线节点多跳的多对一的通信方式、无线链路质量及外界对无线链路的干扰,都容易引起网络的局部或全局拥塞。由于传统TCP不能区分分组丢失原因,往往触发不必要的拥塞控制,从而导致TCP在Ad-hoc网络中性能低下。随着Ad-hoc网络规模越来越庞大,结构日趋复杂,仅仅依靠端到端的拥塞控制是不够的[2],由此考虑在Ad-hoc网络的中间节点引入主动队列管理(Active Queue Management, AQM)机制,通过动态管理并检测路由中的数据堆栈长度来预知可能产生的网络拥塞,并及时通知源端,使之能够提早采取措施,从而尽量避免更严重的数据丢失,实现对网络拥塞的监测和预防。

近年来人们利用经典控制理论的AQM算法[3]来提高网络系统的健壮性和稳定性。文献[4]显示在有线网络中应用广泛的Random Early Detection (RED)算法在Ad-hoc网络中具有不公平性。文献[5]将有线网络的PID-AQM应用于无线网络,但通过工程设计可以发现,PID等经典控制对被控对象的时变比较敏感。由于Ad-hoc网络参数时变,网络环境往往会超出控制器设计的范围,经典控制很难适应这种变化[6]。为了解决这些问题,文献[7]结合Smith预估器和Dahlin算法提出一种预测性的PI算法。文献[8]根据无线信道特性提出基于H_∞方法的AQM。文献[9]则设计了Ad-hoc网络的模糊AQM控制器。随着智能控制的发展,神经网络也被引入AQM设计中。文献[10]设计了一种神经元自适应PID智能控制器,它具有在线学习和自适应调整PID参数的功能,但其控制性能依赖于神经元增益的选取。一方面,增益的选取主要基于经验和实验。另一方面,由于不能自适应调整神经元增益K,当网络环境变化时,控制效果也不够理想,这是需要进一步改进的地方。

到稿日期:2010-07-16 返修日期:2010-11-15 本文受国家自然科学基金项目(60903027),江苏省自然科学基金项目(BK2007593),江苏省高校青蓝工程资助。

陈 亮(1982-),男,博士生,主要研究方向为无线自组网、网络拥塞控制,E-mail:njustchenliang@yahoo.com.cn;张 宏(1956-),男,教授,主要研究方向为信息安全理论与技术。

本文根据 TCP 拥塞窗口加法增大、乘法减小(additive-increase and multiplicative-decrease,AIMD)原则及排队机制,推导出拥塞窗口、队列长度与丢弃概率的微分方程,再对该方程进行线性化,获得 Ad-hoc 网络 TCP/AQM 控制系统模型,再将递推计算、动态修正增益 K 的方法引入控制器,组成具有自动调整增益的神经元 PSD (Proportional Summation Differential)控制器。仿真表明,PSD 算法在 Ad-hoc 网络 AQM 中获得了较好的控制效果。

2 Ad-hoc 网络 TCP/AQM 控制模型

变量定义:t ($t \geqslant 0$)为时间(秒);

$W(t)$为 TCP 拥塞窗口大小(分组);

$q(t)$为队列长度(分组);

q_0 为理想队列长度(分组);

C 为链路容量(分组/秒);

N 为负载(TCP 连接数);

RTT 为往返时间(秒),简记 $R(t)$;

用下标 i 代表第 i 个 TCP 流,则其往返时间 $R_i(t)=D_i+q(t)/C$,其中 D_i 为固定的传播延时;$p(t)$为拥塞分组丢弃概率;$p_{ul}(t)$为无线分组丢弃概率;记两者之和 $p_{total}(t)$;

α 为常系数($0<\alpha<1$);设 $B_i(t)$为第 i 个 TCP 流的发送速率 $B_i(t)=W_i(t)/R_i(t)$ (Mb/s);

变量的数学期望 $E[\phi]$简写作 $\bar{\phi}$。

注:TCP 窗口值=min{拥塞窗口值,接受窗口值}。由于研究网络拥塞,本文考虑性能瓶颈在中间节点,则可认为接受窗口足够大且变化很小。

2.1 TCP 窗口特性

TCP 拥塞控制原理如下:当拥塞窗口小于慢启动门限(ssthresh)时,拥塞窗口随每个 RTT 呈指数增长;当拥塞窗口超过门限值时,慢启动过程就停止了,此时进入拥塞避免阶段,拥塞窗口的增长变为线性增长,即一个 RTT 内所有包的确认包在相对应的计时器超时之前到达后拥塞窗口才增长 1 个。这个算法过程称为拥塞避免(Congestion Avoidance)。如果窗口内所有包的确认包均能在计时器超时之前到达,则拥塞避免阶段将一直持续;当超时(Time-out)现象发生时,ssthresh 设为拥塞窗口的一半,随后拥塞窗口被重新设定为 1;为了防止等待超时带来的性能下降,TCP 规定了快速重传(Fast-Retransmit),即源端收到 3 个 Duplicate ACK 则认为分组丢失,不等待时间结束立即重送。此后进入快速恢复(Fast-Recovery),拥塞窗口及 ssthresh 将被设定为现在拥塞窗口的一半(即系数 $\alpha=0.5$)。以上过程即为 TCP 拥塞窗口的 AIMD 原则。忽略慢启动,TCP 拥塞窗口的数学表达如下:

$W(t+RTT)=W(t)+1$	Congestion Avoidance Phase
$W(t+RTT)=(1-\alpha)W(t)$	After packet loss is detected (3 Duplicate ACK)
$W(t+RTT)=1$	After packet loss is detected (timer expires, Time-out)

由于有线网络中传输错误的概率很小,如果出现分组丢失,一般认为就是网络拥塞造成的,因而 TCP 认为发送方收到 3 个 Dup ACK 就是网络拥塞的标志。文献[11]就 TCP/AQM 有线模型做了很多基础性工作。但在 Ad-hoc 网络中,由于干扰、噪声和冲突也会出现分组丢失。如果这时启动拥塞算法,TCP 会错误地减小传输速率,使得 TCP 性能降低,这些都是 Ad-hoc 网络 TCP/AQM 研究要考虑的因素。由于无线自组网 TCP 流的丢包情况比较复杂,本文考虑将丢弃情况分为两类,即拥塞造成的分组丢失和无线信道造成的丢失。记一个 RTT 内拥塞丢失的分组次数为 $X(t)$,无线信道丢失的分组次数为 $Y(t)$,则系统整体的分组丢失为 $S(t)=X(t)+Y(t)$。

有线 AQM 研究显示[12],链路容量 15Mb/s 且队长为 200 个分组时,拥塞分组丢弃概率 $p(t)$极小(低于 1%)。而根据 NS 仿真,链路容量 1Mb/s 而队长为 50 个分组时,$p(t)$不足 10%。无线丢失方面,文献[13]以无线误码率约 10^{-4} 为严重受干扰的信道,无线误码率小于 10^{-6} 为干扰很小的信道,无线误码率在 $10^{-6}\sim10^{-5}$ 之间为信道干扰的一般状态。本文取误码率为 6×10^{-6},每个分组 1000Byte(8000bit),由无线信道误码率 BER 与丢包率 $p_{ul}(t)$关系[14] $p_{ul}(t)=1-(1-BER)^{\text{packet length}}$,则无线丢弃概率 $p_{ul}(t)=5\%$。理论上无线丢失概率可达 100%。但对 Ad-hoc 网络拥塞研究而言,$p_{ul}(t)$也不能过大,否则会造成拥塞现象消失,队列管理失去意义。

假定 $p_{ul}(t)=5\%$而 $p(t)=10\%$,考虑最坏的丢失情况,即叠加两种丢失,则总丢弃概率 $p_{total}(t)$在 15%左右。在低带宽、小队列的 Ad-hoc 网络中,其 TCP 窗口 $W(t)$平均值很小,一般不到 4 个分组。设总丢包次数 $S(t)$服从二项分布($W(t)$,$p_{total}(t)$),则有一般式:

$$P\{S(t)=k\}=C^k_{W(t)}\,p^k_{total}(t)(1-p_{total}(t))^{W(t)-k}\quad k=0,1,2,\cdots$$

当 $W(t)=4$ packets 及 $p_{total}(t)=15\%$时,有:

$$P\{S(t)=0\}=C^0_4 0.15^0(1-0.15)^{4-0}=52\%$$

$$P\{S(t)=1\}=C^1_4 0.15^1(1-0.15)^{4-1}=37\%$$

可见在一个 RTT 中,拥塞及无线造成的分组丢失次数 $S(t)$是 1 个或 0 个的总概率达到 89%。而随着 $W(t)$减小到 3,此概率可达 94%。在工程上可假设 $S(t)$为区间[0,1]之间的随机数。

根据 2.1 节第一段分析的 TCP 特性,TCP 窗口值可写为

$$W(t+RTT)=(W(t)+1)(1-(X(t)+Y(t)))+(1+(1-\alpha)W(t))(X(t)+Y(t)) \quad (1)$$

为了转化微分形式,式(1)化简为:

$$W(t+RTT)-W(t)=\frac{RTT}{RTT}-\alpha W(t)(X(t)+Y(t)) \quad (2)$$

设时间[0,t]内拥塞造成的分组丢失数 $M_1(t)$服从丢失速率为 $\lambda_1(t)$的泊松分布[11],则有:

$$X(t)=M_1(t+RTT)-M_1(t) \quad (3)$$

设时间[0,t]内无线分组丢失数 $M_2(t)$也服从丢失速率为 λ_2 的泊松分布,则有:

$$Y(t)=M_2(t+RTT)-M_2(t) \quad (4)$$

将式(3)和式(4)代入式(2),考虑以微分形式表达,则有:

$$\mathrm{d}W(t)=\frac{1}{RTT}\mathrm{d}t-\alpha W(t)\mathrm{d}[M_1(t)+M_2(t)] \quad (5)$$

由 TCP 机制 AIMD 描述,取 $\alpha=0.5$,$R_i(t)$是 $q(t)$的函数,对第 i 个 TCP 流,有:

$$\mathrm{d}W_i(t)=\frac{\mathrm{d}t}{R_i(q(t))}-\frac{1}{2}W_i(t)[\mathrm{d}M_1(t)+\mathrm{d}M_2(t)] \quad (6)$$

对式(6)取数学期望,考虑 $E[\mathrm{d}M]=\mathrm{d}E[M]$及

$E[g(x)\mathrm{d}M]\approx E[g(x)]E[\mathrm{d}M]$，式(6)化为：

$$\mathrm{d}E[W_i(t)]=E[\frac{\mathrm{d}t}{R_i(q(t))}]-\frac{1}{2}E[W_i(t)](\mathrm{d}E[M_1(t)]+\mathrm{d}E[M_2(t)]) \tag{7}$$

数学上需要证明两个函数相互独立，才能将两个函数乘积的数学期望分解成函数各自期望的乘积，即 $E[g(x)\mathrm{d}M]=E[g(x)]E[\mathrm{d}M]$。本文中 $W_i(t)$ 与 $\mathrm{d}M$ 并不完全独立，但为了便于推导，做了以上近似。本文仿真表明，以上近似未对 TCP 窗口特性做出根本性改变。

由泊松过程性质 $E[M_j(t)]=\lambda_j t$ $(j=1,2)$，另外考虑 $E[f(x)]\approx f(E[x])$，代入式(7)，得：

$$\mathrm{d}E[W_i(t)]=\frac{\mathrm{d}t}{R_i(E[q(t)])}-\frac{1}{2}E[W_i(t)](\lambda_1+\lambda_2)\mathrm{d}t \tag{8}$$

根据泊松过程定义

$\lambda_1=p(t-R_i(t))\overline{B_i}(t-R_i(t))$，$\lambda_2$ 可同理获得，则整理后得：

$$\frac{\mathrm{d}\overline{W_i}(t)}{\mathrm{d}t}=\frac{1}{R_i(\bar{q}(t))}-\frac{1}{2}\overline{W_i}(t)[p(t-R_i(t))+p_{ul}(t-R_i(t))]\frac{\overline{W_i}(t-R_i(t))}{R_i(\bar{q}(t-R_i(t)))} \tag{9}$$

2.2 瓶颈节点队列特性

考虑第 i 个 TCP 流在稳态下离散时刻源端的发送分组数 $A(k)$，$\{k=0,1,2,\cdots\}$，$a(k)$ 为离散间隔内发送数，则进入队列的分组即为：

$$a(k)=A(k)-A(k-1)$$

由 Lindley 方程：

$$q(k+1)=\max(q(k)+a(k+1)-C,0) \tag{10}$$

将上式转化为连续时间上的函数：

$$q(t+\Delta t)=\max(q(t)+a(t+\Delta t)-C,0) \tag{11}$$

考虑 $a(t+\Delta t)\approx\overline{B_i}(t)\Delta t$，则 Ad-hoc 网络瓶颈节点队列的微分形式可改写为：

$$\frac{\mathrm{d}\bar{q}(t)}{\mathrm{d}t}=-C+\sum_{i=1}^{N}\frac{\overline{W_i}(t)}{R_i(q(t))} \tag{12}$$

2.3 控制系统建模

保留物理意义不变，省略数学期望上标，联立式(9)和式(12)，Ad-hoc 网络 TCP/AQM 微分模型如下：

$$\begin{cases}\dfrac{\mathrm{d}W_i(t)}{\mathrm{d}t}=\dfrac{1}{R_i(t)}-\dfrac{W_i(t)}{2}\dfrac{W_i(t-R_i(t))}{R_i(t-R_i(t))}[p(t-R_i(t))+p_{ul}(t-R_i(t))]=f\\ \dfrac{\mathrm{d}q(t)}{\mathrm{d}t}=-C+\sum_{i=1}^{N}\dfrac{W_i(t)}{R_i(t)}=g\end{cases} \tag{13}$$

在稳定工作点附近，对式(13)用小信号线性化处理[15]，假设 TCP 连接数和链路容量是常数，取 (W,q) 为状态，p 为输出，可求得工作点：

$$\dot{W}=0\Rightarrow W_0^2(p_0+p_{ul0})=2$$

$$\dot{q}=0\Rightarrow W_0=\frac{R_0C}{N}$$

$$R_0(t)=D+q_0/C$$

记 $W_R=W(t-R_0)$，$q_R=q(t-R_0)$，

$p_R=p(t-R_0)$，$p_{ulR}=p_{ul}(t-R_0)$

对式(13)的函数形式，在工作点分别求 f 和 g 的偏微：

$$\frac{\partial f}{\partial W}=-\frac{W_0}{2R_0}(p_0+p_{ul0});\frac{\partial f}{\partial W_R}=-\frac{W_0}{2R_0}(p_0+p_{ul0});$$

$$\frac{\partial f}{\partial q}=-\frac{1}{R_0^2C};\frac{\partial f}{\partial q_R}=\frac{1}{R_0^2C};\frac{\partial f}{\partial p_R}=-\frac{R_0^2C}{2N^2};$$

$$\frac{\partial f}{\partial p_{ulR}}=-\frac{R_0^2C}{2N^2};\frac{\partial g}{\partial q}=-\frac{1}{R_0};\frac{\partial g}{\partial W}=\frac{N}{R_0}$$

线性化式(13)，其中的变量用平衡工作点的值加增量表示：

$$\begin{cases}\delta\dot{W}(t)=\dfrac{\partial f}{\partial W}\delta W(t)+\dfrac{\partial f}{\partial W_R}\delta W_R(t)+\dfrac{\partial f}{\partial q}\delta q(t)+\dfrac{\partial f}{\partial q_R}\delta q_R+\dfrac{\partial f}{\partial p_R}\delta p_R+\dfrac{\partial f}{\partial p_{ulR}}\delta p_{ulR}\\ \delta\dot{q}(t)=\dfrac{\partial g}{\partial W}\delta W(t)-\dfrac{\partial g}{\partial q}\delta q(t)\end{cases} \tag{14}$$

式中，$\delta W=W-W_0$；$\delta q=q-q_0$；$\delta p=p-p_0$；$\delta p_{ul}=p_{ul}-p_{ul0}$。

另外忽略时间延迟 $t-R$ 对队列长度 q 的相关性，并假设它的值固定为 $t-R_0$，另一方面保留动态参数中往返时间对队列长度的相关性，由此可得简化的动态方程，再对式(14)进行拉普拉斯变换，得：

$$\begin{cases}s\cdot\delta W(s)=-\dfrac{N}{R_0^2C}(1+\mathrm{e}^{-sR_0})\delta W(s)-(1-\mathrm{e}^{-sR_0})\dfrac{1}{R_0^2C}\delta q(s)-\dfrac{R_0C^2}{2N^2}\mathrm{e}^{-sR_0}(\delta p(s)+\delta p_{ul}(s))\\ s\cdot\delta q(s)=\dfrac{N}{R_0}\delta W(s)-\dfrac{1}{R_0}\delta q(s)\end{cases} \tag{15}$$

将式(15)转换为控制框图，则特性如图 1 所示。

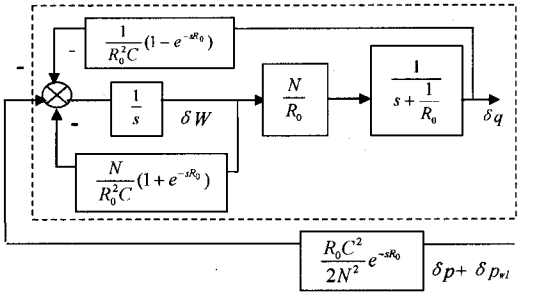

图 1 线性化 TCP 连接方框图

为了分析方便，对图 1 的框图进行合并整理。当拥塞控制稳定时，拥塞窗口 W 大于 1，可忽略高频性能的影响，即 $\mathrm{e}^{-sR_0}\approx1$。线性化后的 Ad-hoc 网络 TCP 结构如图 2 右侧虚线框中所示，其中 $P_0(s)=\dfrac{C^2/(2N)}{(s+2N/(CR_0^2))(s+1/R_0)}$，为被控对象非时滞部分，$\mathrm{e}^{-sR_0}$ 为延时环节，p_{ul} 为无线分组丢失的等效扰动，p 为拥塞的分组丢弃概率，q 为队列长度，q_0 为参考队列，$P_0(s)\mathrm{e}^{-sR_0}$ 由图 1 整理所得；图 2 左侧的点虚线框是引入的控制器，其结构在下节讨论。

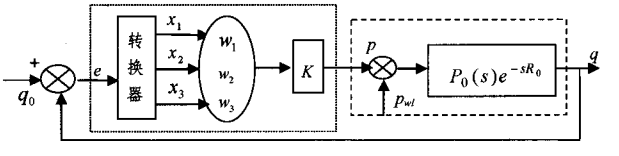

图 2 神经元 PID 控制系统结构框图

3 神经元 PSD 算法

首先分析文献[10]中利用神经元实现的自适应 PID 控制，其结构框图如图 2 所示。

$$\begin{cases}p(k)=p(k-1)+K\dfrac{\sum_{i=1}^{3}w_i(k)x_i(k)}{\sum_{i=1}^{3}w_i(k)}\\ w_i(k+1)=w_i(k)+d_ie(k)p(k)x_i(k)\end{cases} \tag{16}$$

式中，

$x_1(k)=e(k)=q_0(k)-q(k)$，

$x_2(k)=\Delta e(k)=e(k)-e(k-1)$，

$x_3(k)=\Delta^2 e(k)=e(k)-2e(k-1)+e(k-2)$。

i 分别为1,2,3，即 d_1 为积分学习速率 η_I，d_2 为比例学习速率 η_P，d_3 为微分学习速率 η_D。

分析式(16)算法可以发现，神经元增益 K 是系统最敏感的参数。K 值增大或减小，相当于比例、积分及微分3项系数同时增大或减小，所以 K 值的选择非常重要；另一方面，恒定的 K 值也不容易适应被控过程时变的 Ad-hoc 网络。由 Marsik 和 Strejc 提出的无需辨识的自适应控制算法，其机理是根据过程误差的几何特性建立性能指标，从而形成自适应 PSD 控制规律[16]。该方法无需辨识过程参数，只要在线检测过程的期望输出和实际输出，具有简单和可实现性。将自适应神经元增益算法与神经元 PID 结合起来，组成神经元自适应 PSD 控制器，解决增益 K 的自动调整问题，其结构框图仍如图2所示。神经元 PSD 算法如下：

$$\begin{cases}\Delta p(k)=K(k)\sum_{i=1}^{3}\dfrac{w_i(k)}{\sum_{i=1}^{3}|w_i(k)|}x_i(k)\\ w_1(k+1)=w_1(k)+\eta_I z(k)p(k)(e(k)+\Delta e(k))\\ w_2(k+1)=w_2(k)+\eta_P z(k)p(k)(e(k)+\Delta e(k))\\ w_3(k+1)=w_3(k)+\eta_D z(k)p(k)(e(k)+\Delta e(k))\end{cases}\tag{17}$$

式中，$\Delta p(k)$ 为控制器输出增量；$K(k)$ 为神经元增益；$z(k)=q_0(k)-q(k)$，其他定义同式(16)。

工程经验表明，神经元增益 K 值大，则快速性较好，但超调量大，甚至可能使系统不稳定；K 值选择过小，则会使系统的响应速度变慢。当被控对象时延增大时，K 值必须减小，以保证系统稳定。设 $T_v(k)=\overline{|\Delta e(k)|}/|\Delta^2 e(k)|$，据此设计动态增益 $K(k)$ 为：

$$\begin{cases}\text{sign}(e(k))=\text{sign}(e(k-1)),\ K(k)=K(k-1)+VK(k-1)/T_v(k-1)\\ \text{sign}(e(k))\neq\text{sign}(e(k-1)),\ K(k)=0.75K(k-1)\end{cases}\tag{18}$$

式中，$T_v(k)=T_v(k-1)+L^*\,\text{sign}[|\Delta e(k)|-T_v(k-1)|\Delta^2 e(k)|]$，$0.025\leqslant V\leqslant 0.05$，$0.05\leqslant L^*\leqslant 0.1$

当第 k 步的队列长度和理想队列长度的差 $e(k)$ 与第 $(k-1)$ 步队列长度和理想队列长度的差 $e(k-1)$ 同号时，说明响应尚未达到理想值，$K(k)$ 需单调微量递增，以使响应较快接近理想值；而控制误差符号相异时，说明响应已越过理想值，$K(k)$ 需下降为 $K(k-1)$ 的75%，以防增益过大，使系统失稳。

4 仿真研究

4.1 MATLAB 仿真

Ad-hoc 网络参数设置为 $R_0=0.1$s，$C=1$Mb/s $=125$ packets/s (1000Byte/packet)，$N=30$，$p_{ul}(t)=5\%$，自适应 PSD 控制器的神经元增益初始值 K 为 0.02，$L^*=0.05$，$V=0.025$，神经元 PID 算法 K 值恒定为 0.02，两者学习速率都为 $\eta_P=0.40$，$\eta_I=0.35$，$\eta_D=0.40$。由 MATLAB 仿真，两者单位阶跃响应曲线如图3所示。可以发现，神经元 PID 算法性能很大程度上依赖增益 K 的选取。增益 K 设定不当，会使系统震荡严重，而 PSD 控制器尽管 K 初值也设定为 0.02，但能动态调整增益 K，可以在更大的增益范围内稳定系统。其动态增益变化如表1所列。

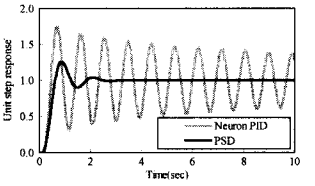

图3 PSD 与神经元 PID 算法单位阶跃响应曲线

表1 自适应 PSD 算法神经元增益 K 变化表

时间/s	0	0.65	1.25	1.50	1.95	2.00	2.50
神经元增益	0.02	0.014	0.013	0.014	0.012	0.012	0.012

4.2 NS 仿真

Ad-hoc 网络节点 $S_1, S_2, S_3, \cdots, S_N$ 通过瓶颈节点 G 向目标节点 D 发送数据，应用层运行 FTP 服务，拓扑结构如图4所示。所有节点链路容量默认为 1 Mb/s，考虑无线丢弃为分散性遗失，无线丢包率为 $p_{ul}(t)$，瓶颈节点 G 接口队列长度为 100Packets，理想队列长度设为 50Packets。节点 G 的队列管理采用神经元 PID 及 PSD 算法，控制器参数均同 4.1 节中的设置，仿真时间 100s。

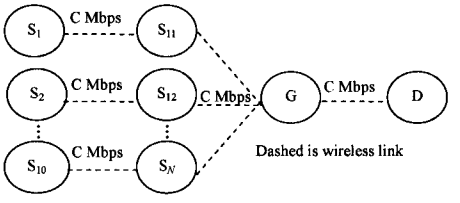

图4 Ad-hoc 网络拓扑结构

仿真1 无线分组丢失对算法的影响

负载 $N=20$，链路容量 $C=1$Mb/s，无线丢包率 $p_{ul}(t)=5\%$，如图5所示，神经元 PID 算法出现了较大的震荡，而 PSD 算法则较好地稳定了队列长度。由于无线丢包的存在，队列初期增长受到较明显的抑制，队列到达峰值需要的时间较长。

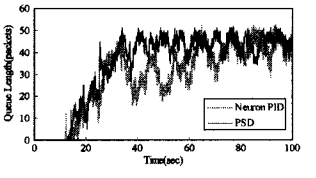

图5 无线丢失下 Neuron PID 与 PSD 队列长度曲线

仿真2 突发业务流的冲击对算法的影响

链路容量 $C=1$Mb/s，无线丢包率 $p_{ul}(t)=0\%$，初始时刻0秒有10个 FTP 业务源启动($N=10$)，在 50s 时另有 10 个 FTP 业务源启动($N=20$)，实验结果如图6所示。当引入突发业务流时，神经元 PID 算法在 50s 以后受其影响较大，队列长度出现较大的震荡，而 PSD 可以较好地处理突发业务流的冲击。

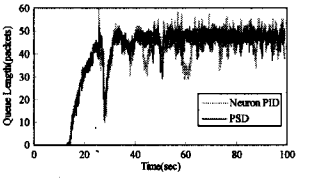

图6 突发业务流的冲击下 Neuron PID 与 PSD 队列长度曲线

(下转第 69 页)

11 DCF with variable packet length[J]. IEEE Communications Letters,2004,8(3):186-188

[20] Blefari-Melazzi N,Detti A,Habib I,et al. TCP Fairness Issues in IEEE 802. 11 Networks[J]. Problem Analysis and Solutions Based on Rate Control. IEEE Transactions on Wireless Communications,2007,6(4):1346-1355

[21] Tickoo O,Sikdar B. Modeling Queueing and Channel Access Delay in Unsaturated IEEE 802. 11 Random Access MAC Based Wireless Networks[J]. IEEE/ACM Transactions on Networking,2008,16(4):878-891

[22] Xiao Y, Du H, Cao Z, et al. Active queue management for differentiated network[C]//Proceedings of IET International Conference on Wireless Mobile and Multimedia Networks(ICWMMN 2006). Hangzhou,China,2006:357-361

[23] Xiao Y,Kim K. Congestion control of Differentiated service network[J]. Chinese Journal of Electronics,2010,19(1):113-118

[24] Mao P-X,Zhang N,Xiao Y,et al. The QoS of the Edge Router Based on Diffserv/MPLS[C]//5th International Conference on Wireless Communications, Networking and Mobile Computing (WiCom '09). Sept. 2009:1-4

[25] Lu L Y,Xiao Yang,Woo S,et al. Network congestion control algorithm with cooperation between edge and core routers[C]//11th International Conference on Advanced Communication Technology(ICACT 2009)Volume 01. Feb. 2009:621-625

[26] 郭振清,肖扬. CDMA 系统粒子群多用户检测算法[J]. 信号处理,2007(6):806-809

(上接第 48 页)

仿真 3　链路容量时变对算法的影响

$N=20$，$p_{ut}(t)=0$，初始时刻所有节点链路容量 $C=1$Mb/s。当 50s 时，瓶颈节点 G 的链路容量减至 0. 1Mb/s，60s 时，链路容量恢复为 1Mb/s。实验结果如图 7 所示，PSD 算法比神经元 PID 更好地控制了因链路容量骤变出现的队列震荡，可取得更好的控制效果。

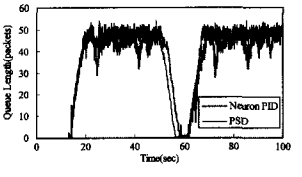

图 7　链路容量时变下 Neuron PID 与 PSD 队列长度曲线

对比实验的相关性能参数如表 2 所列，表中 Neuron PID 简写为 N-PID。可以看出，相较神经元 PID 控制，PSD 算法具有较小的静态误差和超调。而当队列基本稳定后，面对突发干扰，队列震荡幅度也小于神经元 PID，综合性能较好。另外，与有线网络不同[3]，Ad-hoc 网络传输数据前各节点需要互相通信并路由，会花费一定的时间，这是其调节时间远大于有线网络的原因。

表 2　Neuron PID 与 PSD 性能比较

	无线分组丢失		突发业务流		链路容量时变	
	PSD	N-PID	PSD	N-PID	PSD	N-PID
超调/%	2	2	22	2	6	2
调节时间/s	34	34	26	26	22	22
稳态误差/pac	18	11	5	4	7	6
稳态 max/pac	52	51	58	52	53	54
稳态 min/pac	16	27	29	29	28	38

结束语　由于神经元 PID 的神经元增益对其控制性能影响较大，而增益的配置又具有盲目性，基于 Ad-hoc 网络 TCP/AQM 模型，本文设计了一种动态修正神经元增益的自适应 PSD 控制器。与神经元 PID 相比，PSD 算法具有可调参数选择范围大、自适应能力强的特点。仿真结果表明，在无线丢失、突发流及链路容量骤变的时变 Ad-hoc 网络中，PSD 的 AQM 控制器具有较好的动态与静态性能，优于神经元 PID 算法。

参 考 文 献

[1] Promkotwong D,Sornil O. A Mesh-based QoS Aware Multicast Routing Protocol[C]//Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing. Volume 3, Qingdao: IEEE Computer Soc,2007:1046-1051

[2] 李千目,戚涌,许满武,等. 基于模糊神经网络的移动组网拥塞预测[J]. 计算机科学,2006,33(10):37-39

[3] 侯萍,王执铨. 基于混沌优化的大时滞网络拥塞控制算法[J]. 系统仿真学报,2009,21(17):5486-5497

[4] Xu Kai-xin,Mario G,Qi Lan-tao,et al. TCP Unfairness in Adhoc Wireless Networks and a Neighborhood RED Solution[J]. Wireless Networks,2005,11(4):383-399

[5] 续欣,汤凯,马刈非. 无线误码信道上的拥塞控制策略[J]. 通信学报,2004,25(12):8-13

[6] 李千目,许满武,严悍,等. 一种网络拥塞预测新方法[J]. 系统仿真学报,2006,18(8):2101-2104

[7] 钱艳平,李奇,刁翔. 预测 PI 时滞网络拥塞控制算法设计及性能分析[J]. 控制理论与应用,2006,23(2):161-168

[8] Zheng Feng, Nelson J. An H_∞ approach to congestion control design for AQM routers supporting TCP flows in wireless access networks [J]. Computer Networks, 2007, 51 (6): 1684-1704

[9] Natshed E, Adznan K S, et al. Intelligent Reasoning Approach for Active Queue Management in Wireless Ad-hoc Networks [J]. International Journal of Business Data Communications and Networking,2007,3(1):16-35

[10] Sun Jin-sheng,Chan Sammy,Ko King-Tim,et al. Neuron PID: A Robust AQM Scheme[C]//Proceeding of ATNAC 2006. Melbourne:IEEE,2006:259-262

[11] Misra V,Gong W B,Towsley D. Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED[C]//Proceedings of the ACM SIGCOMM. Stockholm: ACM Press,2000:151-160

[12] 任丰原,林闯,任勇,等. 大时滞网络中的拥塞控制算法[J]. 软件学报,2003,14(3):503-511

[13] Chatzimisios P, Vitsas V, Boucouvalas A C. DIDD backoff scheme: An enhancement to IEEE 802. 11 DCF under burst transmission errors[C]//Proceedings of the IEEE Sarnoff 2006 Symposium. Nassau Inn in Princeton:IEEE,2006:27-28

[14] Kumar A, Manjunath D, Kuri J. Wireless networking[M]. Burlington:Morgan Kaufmann Pub,2008:75-76

[15] Bigdeli N, Haeri M. AQM controller design for networks supporting TCP vegas: A control theoretical approach[J]. ISA Transactions,2008,47(1):143-155

[16] Gong C K,Chen D J. Neuron PSD Control for Piezoelectric Micro Displacement System[J]. Journal of Physics,2006,48(1):1107-1111