自适应 PI 主动队列管理算法*

卢锡城, 张明杰+, 朱培栋

(国防科学技术大学 计算机学院,湖南 长沙 410073)

An Adaptive PI Active Queue Management Algorithm

LU Xi-Cheng, ZHANG Ming-Jie+, ZHU Pei-Dong

(School of Computer, National University of Defense Technology, Changsha 410073, China)

+ Corresponding author: Phn: +86-10-66241219, E-mail: canicula@263.net, http://www.nudt.edu.cn

Received 2003-11-17; Accepted 2004-06-10

Lu XC, Zhang MJ, Zhu PD. An adaptive PI active queue management algorithm. *Journal of Software*, 2005, 16(5):903–910. DOI: 10.1360/jos160903

Abstract: Active queue management (AQM) is a very active research area in networking. Compared with drop-tail, AQM can provide smaller average queue delay and higher bandwidth utilization. Although the performance of proportional integral (PI) controller is superior to that of random early detection (RED), its convergence speed is slow. This paper proposes an adaptive proportional integral (API) algorithm based on the original PI. API obtains load information by measuring the current packet-dropping rate, then sets PI parameters accordingly. Verified by using NS-2 simulations, API can achieve faster convergence speed and smaller queue oscillation than PI and PIP (proportional integral based series compensation and position feedback compensation) which is an improved algorithm of PI.

Key words: active queue management; proportional integral; adaptive; convergence speed; queue oscillation

摘 要: 主动队列管理是一个非常活跃的研究领域,相对于丢尾算法,AQM(active queue management)能够提供更短的平均队列延迟和更高的带宽利用率.虽然 PI(proportional integral)主动队列管理算法的性能优于 RED(random early detection)算法,但是 PI 算法的收敛速度比较慢.以 PI 算法为基础提出了一种自适应 PI 算法 API(adaptive proportional integral).API 通过实时测量链路的报文丢失率,获得当前的负载信息,然后动态设置 PI 算法中的有关参数.通过 ns-2 模拟表明,相对于 PI 及其改进算法 PIP(proportional integral based series compensation and position feedback compensation),API 具有更快的收敛速度和更小的队列抖动.

关键词: 主动队列管理;成比例积分;自适应;收敛速度;队列抖动

中图法分类号: TP393 文献标识码: A

因特网中大量存在的传输层协议是 TCP,如何针对 TCP 设计缓冲管理算法一直是人们研究的重点.由于丢

* Supported by the National Natural Science Foundation of China under Grant Nos.90104001, 90204005 (国家自然科学基金)

作者简介: 卢锡城(1946－),男,江苏靖江人,教授,博士生导师,中国工程院院士,主要研究领域为先进网络技术,高性能计算,并行与分布处理;张明杰(1974－),男,博士生,主要研究领域为网络服务质量,拥塞控制;朱培栋(1971－),男,博士,副教授,主要研究领域为网络路由,组播技术,高性能路由器.

尾算法的缺点,AQM(active queue management)[1]成为近几年来拥塞控制领域中的研究热点.Floyd 提出的RED(random early detection)[2]算法根据平均队列长度计算报文丢弃概率.自从 RED 提出以来,人们从理论和实验两方面对其性能进行了大量研究,同时对 RED 进行了许多改进.研究表明,RED 算法的性能和配置参数紧密相关,不正确的参数配置导致很大的队列抖动[3].从端系统的角度看,文献[4]通过实验表明,相对于其他主动队列管理算法,自适应 RED[5]算法对 Web 的性能改进最小.还需要指出的是,即使是自适应 RED 算法,它也是基于一种直觉的合理性,缺乏严谨的科学论证.

除了 RED 之外,研究人员还提出了自适应虚缓冲[6]和 REM[7]主动队列管理算法.这两种基于优化的方法更多地关注系统的稳态特性,而不是队列的瞬态性能.

文献[8]建立了 TCP/AQM 控制论模型,从而可以使用控制理论研究主动队列管理算法.Hollot 对建立的控制论模型进行了合理的线性化处理,提出了 PI(proportional integral)算法[9].PI 使用瞬时队列长度计算报文丢弃概率,这一点和 RED 不同,理论分析和模拟都表明,PI 算法比 RED 算法具有更小的队列抖动,从而在保证高带宽利用率的前提下,为端用户提供更小的延迟抖动.但是 PI 算法中,参数是固定设置的,固定的参数设置导致 PI 算法在较小的目标队列长度情况下收敛速度较慢,为了改进 PI 算法的收敛特性,文献[10]提出了 PIP(proportional integral based series compensation and position feedback compensation)算法,PIP 算法在 PI 算法的基础上引入了位置反馈补偿,使得系统具有更快的收敛速度.

本文针对 PI 算法的缺点,提出了自适应 PI 算法 API(adaptive proportional integral),API 算法通过实时测量链路的报文丢失率,动态地调整 PI 算法的配置参数,从而保证 PI 算法具有更快的收敛速度和更小的队列抖动.

本文第 1 节介绍 TCP/AQM 控制理论模型.第 2 节阐述 API 算法设计.第 3 节通过实验验证 API 算法的有效性.第 4 节总结全文并指出下一步的工作.

1 TCP/AQM 控制理论模型

以控制论为基础的 TCP/AQM(PI)系统,如图 1 所示.

在图 1 中,q_0 是目标队列长度,p 是报文丢失率.如果系统的负载为 N,连接的 RTT(round-trip time)为 R,则 PI 模块可以表示为 $G_1(s)=\frac{1+\tau s}{Ts}$ (τ,T 是待确定的参数),AQM dynamic 模块可以表示为 $G_2(s)=\frac{K_m e^{-sR}}{(T_1 s+1)(T_2 s+1)}$,其中 $K_m=\frac{(RC)^3}{4N^2}$, $T_1=\frac{R^2C}{2N}$, $T_2=R$.整个系统的开环传递函数为

$$G(s)=\frac{1+\tau s}{Ts}\frac{K_m e^{-sR}}{(T_1 s+1)(T_2 s+1)} \tag{1}$$

文献[9]给出了当 $R\le R_{\max}, N\ge N_{\min}$ 时系统的稳定性条件,PI 算法根据式(2)计算报文丢弃概率.

$$p(k)=(a-b)(q(k)-q_0)+b(q(k)-q(k-1))+p(k-1) \tag{2}$$

在式(2)中,$p(k)$是在 k 时刻的报文丢弃概率,$q(k)$是在 k 时刻的瞬时队列长度,a,b 是 τ和 T 的函数.在 PI 算法中,参数 a,b 是固定配置好的,不能随网络动态变化而灵活设置,因而导致 PI 算法收敛较慢,这一点在文献[10]中通过实验进行了验证,本文的实验也证明了这一点.

PIP[10]算法在 PI 基础上引入了位置反馈补偿,如图 2 所示.在图 2 中,要求$|G_2(j\omega)G_c(j\omega)|>>1$.PIP 算法得到的开环带宽比 PI 算法的要大,从而获得更快的收敛速度.虽然 PIP 优于 PI 算法,但是 PIP 算法的稳定性条件也是固定的.

如果 PI 算法的参数能够随着网络的动态变化而灵活设置,那么就可以取得更好的队列特性,本文提出的自适应 PI 算法 API 在这方面进行了努力,下面具体介绍 API 算法的理论基础和实现.

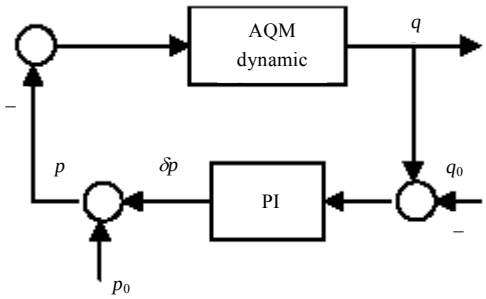


Fig.1 TCP/AQM control system (PI)

图 1 TCP/AQM 控制系统(PI)

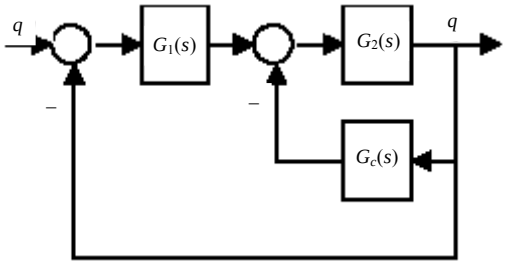


Fig.2 TCP/AQM control system (PIP)

图 2 TCP/AQM 控制系统(PIP)

2 API 算法设计

2.1 根据丢失率推测链路负载信息

文献[11]分析了 TCP 的吞吐量,得到如下表达式:

$$T=\frac{1}{H(R,p)} \tag{3}$$

其中 $H(R,p)=R\sqrt{\frac{2p}{3}}+t_{RTO}\left(3\sqrt{\frac{3p}{8}}\right)p(1+32p^2)$,$T$ 的单位为报文/秒,R 是 RTT 大小,t_{RTO} 是连接超时值,p 是报文丢失率.一般 $t_{RTO}\approx 5R$ [12],则式(3)变形为

$$T=\frac{1}{h(p)R} \tag{4}$$

其中 $h(p)=\sqrt{\frac{2p}{3}}+\left(15\sqrt{\frac{3p}{8}}\right)p(1+32p^2)$.

如果当前 TCP 连接的数目为 N,每条连接的 RTT 为 R,链路带宽为 C,那么由式(4)可知,$T=\frac{C}{N}=\frac{1}{h(p)R}$,从而可以导出 N,R 和 p 的关系式:

$$N=\alpha R \tag{5}$$

其中 $\alpha=\left(\sqrt{\frac{2p}{3}}+\left(15\sqrt{\frac{3p}{8}}\right)p(1+32p^2)\right)C=h(p)C$.

由式(5)可知,给定 N,R,可以确定链路的丢失率 p;反过来,知道链路的丢失率 p,可以推测负载 N 和 R.定义集合 $\Omega_p=\{\langle N,R\rangle \mid N-h(p)CR=0\}$,$\Omega_p$ 是所有丢失率为 p 时的 N,R 组合.

2.2 调整参数定理

这一节介绍 API 算法参数调整的理论依据.

定理 **1**. 当系统负载为 N,RTT 为 R 时,如果 $\omega_g=\frac{\beta}{\sqrt{T_1T_2}}$,$\tau\geq\frac{1}{\omega_g}$,$T\geq\frac{K_m\tau}{1+\beta^2}$,$\beta\leq\frac{\pi/4}{\sqrt{2h(p)}}$,那么系统(1)是稳定的.

证明:整个系统的开环传递函数 $G(s)=\frac{1+\tau s}{Ts}\frac{K_me^{-sR}}{T_1T_2s^2+(T_1+T_2)s+1}$,则系统的频率响应为

$$|G(j\omega)|=\frac{\sqrt{1+\tau^2\omega^2}}{T\omega}\frac{K_m}{\sqrt{(1-T_1T_2\omega^2)^2+(T_1+T_2)^2\omega^2}},$$

$$\left|G\left(j\omega_g\right)\right| \approx \frac{\tau}{T}\frac{K_m}{\sqrt{\left(1-T_1T_2\omega_g^2\right)^2+\left(T_1+T_2\right)^2\omega_g^2}}$$
$$=\frac{\tau}{T}\frac{K_m}{\sqrt{\left(1-\beta^2\right)^2+\left(T_1+T_2\right)^2\omega_g^2}}$$
$$\le \frac{\tau}{T}\frac{K_m}{\sqrt{\left(1-\beta^2\right)^2+4\beta^2}}$$
$$=\frac{\tau}{T}\frac{K_m}{1+\beta^2}\le 1.$$

相角频率特性:

$$\varphi\left(\omega_g\right)=-90^\circ+\mathrm{arctg}\left(\tau\omega_g\right)-\mathrm{acrtg}\frac{\left(T_1+T_2\right)\omega_g}{1-T_1T_2\omega_g^2}-\omega_gR.$$

因为 $\omega_gR=\frac{\beta R}{\sqrt{T_1T_2}}=\beta R\sqrt{\frac{2\alpha R}{R^3C}}=\beta\sqrt{\frac{2\alpha}{C}}=\beta\sqrt{2h(p)}\le\frac{\pi}{4}=45^\circ$,

所以 $\varphi\left(\omega_g\right)\ge -90^\circ+45^\circ-90^\circ-45^\circ=-180^\circ$.

由 Nyquist 稳定性判据可知系统是稳定的. □

定理 2. 对于 N,R 变化的系统而言,当丢失率为 p 时,取 $\omega_g=\min\left\{\frac{\beta}{\sqrt{T_1T_2}}\middle|\left\langle N,R\right\rangle\in\Omega_p\right\}$, $\beta\le\frac{\pi/4}{\sqrt{2h(p)}}$, $\tau\ge\frac{1}{\omega_g}$,

$T\ge\max\left\{\frac{K_m\tau}{\sqrt{\left(1-T_1T_2\omega_g^2\right)^2+\left(T_1+T_2\right)^2\omega_g^2}}\middle|\left\langle N,R\right\rangle\in\Omega_p\right\}$,则 $\forall\left\langle N,R\right\rangle\in\Omega_p$,系统稳定.

证明:对于 $\forall\left\langle N,R\right\rangle\in\Omega_p$,令 $\omega_g'=\frac{\beta}{\sqrt{T_1T_2}}$,则 $\omega_g\le\omega_g'$,系统的频率响应为

$$\left|G\left(j\omega\right)\right|=\frac{\sqrt{1+\tau^2\omega^2}}{T\omega}\frac{K_m}{\sqrt{\left(1-T_1T_2\omega^2\right)^2+\left(T_1+T_2\right)^2\omega^2}},$$
$$\left|G\left(j\omega_g\right)\right|\approx\frac{\tau}{T}\frac{K_m}{\sqrt{\left(1-T_1T_2\omega_g^2\right)^2+\left(T_1+T_2\right)^2\omega_g^2}}$$
$$\le\frac{\sqrt{\left(1-T_1T_2\omega_g^2\right)^2+\left(T_1+T_2\right)^2\omega_g^2}}{K_m}\frac{K_m}{\sqrt{\left(1-T_1T_2\omega_g^2\right)^2+\left(T_1+T_2\right)^2\omega_g^2}}=1.$$

相角频率特性:

$$\varphi\left(\omega_g\right)=-90^\circ+\mathrm{arctg}\left(\tau\omega_g\right)-\mathrm{acrtg}\frac{\left(T_1+T_2\right)\omega_g}{1-T_1T_2\omega_g^2}-\omega_gR$$
$$\ge-90^\circ+\mathrm{arctg}\left(\tau\omega_g\right)-\mathrm{acrtg}\frac{\left(T_1+T_2\right)\omega_g}{1-T_1T_2\omega_g^2}-\omega_g'R\ge-180^\circ.$$

由 Nyquist 稳定性判据可知 $\forall\left\langle N,R\right\rangle\in\Omega_p$ 系统稳定. □

定理 3. 对于 N,R 变化的系统而言,当报文丢失率为 p,最大的 RTT 为 $R_{\max}$ 时,取 $\omega_g=\frac{\pi/4}{R_{\max}}$, $\tau\ge\frac{1}{\omega_g}$,

$T\ge\max\left\{\frac{K_m\tau}{\sqrt{\left(1-T_1T_2\omega_g^2\right)^2+\left(T_1+T_2\right)^2\omega_g^2}}\middle|\left\langle N,R\right\rangle\in\Omega_p\right\}$,则 $\forall\left\langle N,R\right\rangle\in\Omega_p$,系统稳定.

证明:由定理 2 可知,只需证明 $\omega_g=\min\left\{\frac{\beta}{\sqrt{T_1T_2}}\middle|\langle N,R\rangle\in\Omega_p\right\}=\frac{\pi/4}{R_{\max}}$.

取 $\beta=\frac{\pi/4}{\sqrt{2h(p)}}$,由定理 2 可知,

$$\begin{aligned}\omega_g&=\min\left\{\frac{\beta}{\sqrt{T_1T_2}}\middle|\langle N,R\rangle\in\Omega_p\right\}\\&=\frac{\pi/4}{\sqrt{2h(p)}}\min\left\{\frac{1}{\sqrt{T_1T_2}}\middle|\langle N,R\rangle\in\Omega_p\right\}\\&=\frac{\pi/4}{\sqrt{2h(p)}}\min\left\{\frac{\sqrt{2h(p)}}{R}\middle|\langle N,R\rangle\in\Omega_p\right\}\text{ 因为 }\left(T_1(N,R)=\frac{R^2C}{2N}=\frac{R^2C}{2\alpha R}=\frac{R}{2h(p)}\right)\\&=\frac{\pi}{4}\min\left\{\frac{1}{R}\middle|\langle N,R\rangle\in\Omega_p\right\}=\frac{\pi/4}{R_{\max}}.\end{aligned}$$ □

定理 3 说明了为什么 API 算法比 PI 和 PIP 算法具有更大的开环带宽.例如,当 C=3750, $R_{\max}=0.22$, $N_{\min}=60$ 时,PI 给出的开环带宽为 0.66(rad/s),PIP 算法给出的开环带宽为 1.7(rad/s),定理 3 给出的开环带宽为 3.57(rad/s).开环带宽越大,系统的收敛速度越快.当 $N<N_{\min}$ 时,由 PI 和 PIP 算法给出的稳定性条件可能使系统不稳定,如果按定理 3 进行参数调整,仍可以使系统处于稳定状态.

2.3 API算法设计

根据定理 3 我们设计了自适应算法 API,API 由两部分组成:初始化部分和参数调节部分,下面分别加以介绍.

2.3.1 API 算法初始化过程 initAPI

API 维护两个简单的数组 $API_\tau[30]$(参数τ)、 $API_T[30]$(参数T),每一个元素与一个丢失率 p 相对应.初始化过程就是初始化这两个数组.初始化过程如下:

```
for (i=0;i<30;i++){
    p=i/100;
    if (p==0)p=0.001;
    根据定理 3 计算 API_τ[i], API_T[i];
}
```

在 initAPI 过程中,p 表示当前的报文丢失率.最大的报文丢失率设置为 30%.网络管理员也可以根据具体的操作环境来设置最大的报文丢弃概率.

2.3.2 API 参数调整过程 updateAPI

```
每隔 adaptiveInterval 时间:
    计算当前的报文丢失率 p;
    index=(int)(p×100);
    τ = API_τ[index];
    T = API_T[index];
    更新 PI 算法中对应的系数 a,b.
```

在 updateAPI 过程中,*adaptiveInterval* 是算法的执行周期,具体实现时取 1s.PI 算法中参数 a,b 与 τ,T 的关系是 $a=\frac{\tau}{T}+\frac{1}{T\cdot S}$ (S 是 PI 算法采样队列长度的频率), $b=\frac{\tau}{T}$.

关于丢失率 p 的计算可以有两种方法:

指数平滑均值(**EWMA**)法.

$p=(1-k)p+kP$, k 是加权因子,P 是上一时间间隔(长度为 *adaptiveInterval*)的报文丢失率.加权因子越大,算法对负载变化的响应速度就越快.

直接使用上一时间间隔的报文丢失率.

$p=P$,相当于 k=1 时的 EWMA 方法.

updateAPI 使用了第 2 种方法.

3 模拟验证

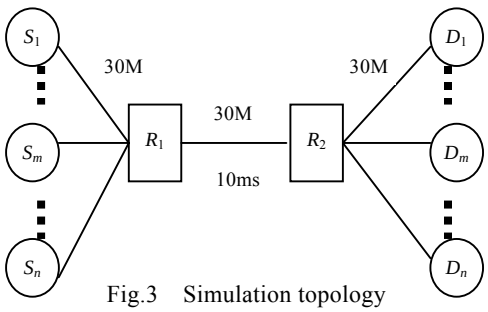


Fig.3 Simulation topology

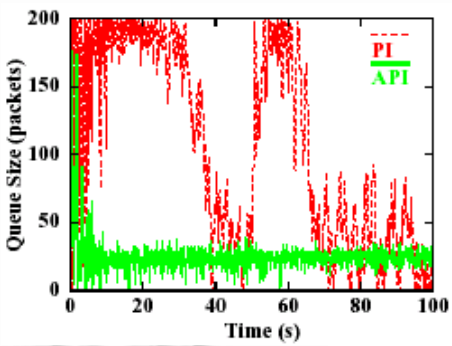
图 3 模拟拓扑

本文使用 NS-2[13]模拟器对 API 算法进行了验证,同时与 PI 和 PIP 算法进行了比较.模拟拓扑如图 3 所示,其中 S_1~S_n 为发送节点,D_1~D_n 为接收节点,R_1,R_2 为路由器,S_i→R_1 和 R_2→D_i 的传输延迟为[10ms,50ms]之间平均分布的随机延迟,R_1→R_2 的链路构成瓶颈.R_1 上分别运行 API,PI 和 PIP 算法.目标队列长度设置为 25 个报文.

实验中,TCP 使用 Reno,报文大小为 1000 字节;CBR 报文大小为 500 字节,报文发送间隔为 0.5s;HTTP 短连接的平均页面请求大小为 10K 字节.

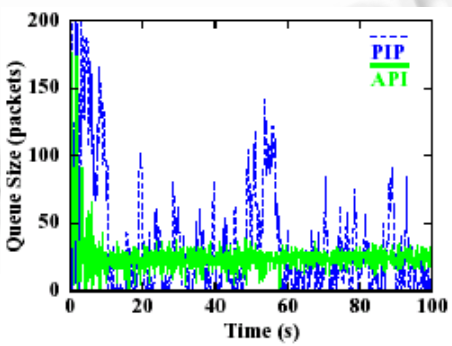
实验 **1**.

实验 1 比较了 3 种算法的响应速度.实验过程:在 0s~10s 之间启动 200 条 FTP 连接;然后在 50s~60s 之间再启动 200 条 FTP 连接.总的模拟过程持续 100s.瞬时队列长度随时间变化的过程如图 4(a)和图 4(b)所示.从图 4(a)可以看出,当 FTP 连接数从 0 增加到 200 时,PI 算法的队列长度从 0 攀升到最大值,然后缓慢收敛到目标队列长度.当 FTP 连接数目在 50s 增加时, PI 算法的队列长度又会出现较大的抖动.从图 4(b)可以看出,PIP 算法的队列长度能够很快地收敛到目标值,在队列长度稳定之后,PIP 和 PI 算法具有相似的抖动.API 不仅具有较快的收敛速度,而且具有比 PIP 更小的队列抖动.



(a) Queue evolution of Exp 1 (PI vs. API)

(a) 实验 1 队列变化(PI vs. API)



(b) Queue evolution of Exp 1 (PIP vs. API)

(b) 实验 1 队列变化(PIP vs. API)

Fig.4 Simulation results of Exp.1

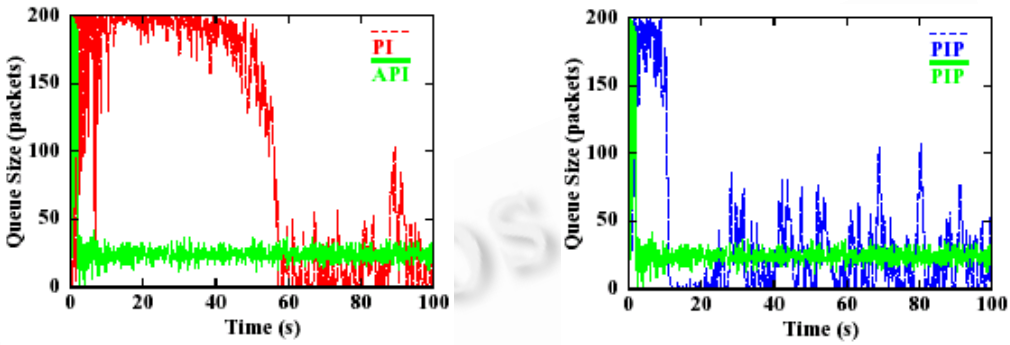
图 4 实验 1 模拟结果

实验 **2**.

实验 2 验证了 API 在重负载情况下的性能.实验过程:在 0s~10s 之间启动 500 条 TCP 连接.模拟结果如图 5(a)和图 5(b)所示.从图 5(a)可以看出,在重负载情况下,PI 算法的收敛速度很慢;从图 5(b)可以看出,虽然 PIP 算法的收敛速度比 PI 算法快,但是在 10s~30s 之间,PIP 算法出现了队列下溢现象.在重负载情况下,API 算法的性能不仅明显优于 PI,而且优于 PIP 算法.

实验 **3**.

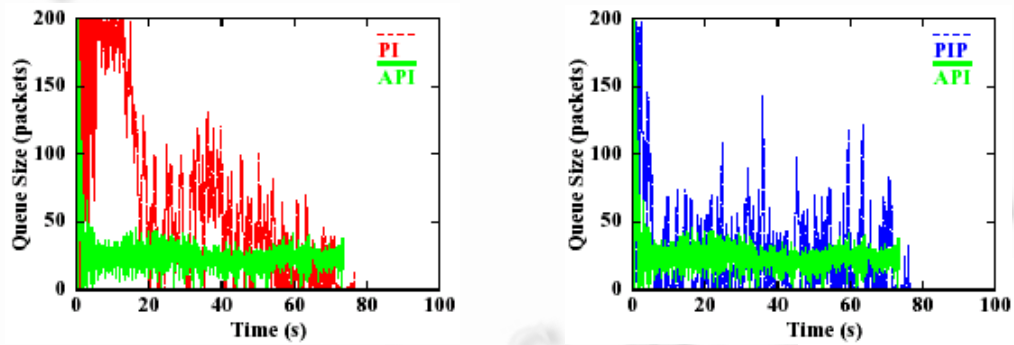
实验 3 评估了 API 算法在混合连接类型情况下的性能.具体的模拟过程:在 0s~10s 之间,启动 150 条 FTP 连接,在 150s 时停止这些连接;在 20s~30s 之间,启动 100 条 CBR 连接,在 120s 时停止这些连接;在 40s~50s 时启动 200 条 HTTP 连接;在 60s~70s 之间再启动 70 条 FTP 连接,这些连接在 140s 时停止;在 80s~90s 之间启动 100 条 CBR 连接,在 160s 时停止这些连接.模拟结果如图 6(a)和图 6(b)所示.从实验 3 可以看出,在有短连接(HTTP)和 UDP(CBR)连接存在的情况下,API 算法仍然具有较快的收敛速度和较小的队列长度抖动,性能优于 PI 和 PIP 算法.



(a) Queue evolution of Exp 2 (PI vs. API)
(a) 实验 2 队列变化(PI vs. API)

(b) Queue evolution of Exp 2 (PIP vs. API)
(b) 实验 2 队列变化(PIP vs. API)

Fig.5 Simulation results of Exp.2
图 5 实验 2 模拟结果



(a) Queue evolution of Exp 3 (PI vs. API)
(a) 实验 3 队列变化(PI vs. API)

(b) Queue evolution of Exp 3 (PIP vs. API)
(b) 实验 3 队列变化(PIP vs. API)

Fig.6 Simulation results of Exp.3
图 6 实验 3 模拟结果

综合以上 3 个实验可以看出,与 PI 和 PIP 算法相比,API 具有更好的队列特性,包括更快的收敛速度和更小的队列抖动.

4 结束语及下一步工作

本文基于 PI 算法提出了一种自适应的主动队列管理算法 API,通过分析和模拟表明,API 算法能够适应动态变化的网络环境,与 PI 和 PIP 算法相比,具有更快的收敛速度和更小的队列抖动.文中通过丢失率估计负载的方法具有一定的普适性.在 API 算法中,唯一需要配置的参数是 R_{max},由于网络环境不同,很难知道 R_{max} 的数值,因此可以使用一定的策略探测 R_{max} 值,从而使 API 能够适应更加复杂的网络环境,这是我们下一步要做的工作.

References:

[1] Braden B, Clark D, Crowcroft J, Davie B, Deering S, Estrin D, Floyd S, Jacobson V, Minshall G, Partridge C, Peterson L, Ramakrishnan K, Shenker S, Wroclawski J, Zhang L. Recommendations on queue management and congestion avoidance in the Internet. RFC2309, Internet Engineering Task Force, 1998.

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

[3] Hollot CV, Misra V, Towsley D, Gong W. A control theoretic analysis of RED. In: Ammar M, ed. Proc. of the IEEE INFOCOM. Anchorage: IEEE Communications Society, 2001. 1510−1519.

[4] Le L, Aikat J, Jeffay K, Smith FD. The effects of active queue management on Web performance. In: Proc. of the ACM SIGCOMM 2003. Karlsruhe, 2003. 265−276. http://www.cs.unc.edu/~jeffay/papers/SIGCOMM-03.pdf

[5] Floyd S, Gummadi R, Shenker S. Adaptive RED: An algorithm for increasing the robustness of RED's active queue management. 2001. http://www.icir.org/˜floyd

[6] Kunniyur S, Srikant R. A time scale decomposition approach to adaptive ECN marking. In: Ammar M, ed. Proc. of the IEEE INFOCOM. Anchorage: IEEE Communications Society, 2001. 1330−1339.

[7] Athuraliya S, Low S, Li VH, Yin QH. REM: Active queue management. IEEE Network, 2001,15(3):48−53.

[8] Misra V, Gong WB, Towsley D. Fluid-Based analysis of a network of AQM routers supporting TCP flows with an application to RED. In: Proc. of the ACM SIGCOMM 2000. Stockholm, 2000. 151−160. http://gaia.cs.umass.edu/fluid/

[9] Hollot CV, Misra V, Towsley D, Gong W. On designing improved controllers for AQM routers supporting TCP flows. In: Ammar M, ed. Proc. of the IEEE INFOCOM. Anchorage: IEEE Communications Society, 2001. 1726−1734.

[10] Zhang HY, Liu BH, Dou WH. Design of a robust active queue management algorithm based on feedback compensation. In: Proc. of the ACM SIGCOMM 2003. Karlsruhe, 2003. 277−285. http://portal.acm.org/citation.cfm?id=863987

[11] Padhye J, Firoiu V, Towsley D, Kurose J. Modeling TCP throughput: A simple model and its empirical validation. In: Oran D, ed. Proc. of the ACM SIGCOMM. Vancouver: ACM Press, 1998. 303−314.

[12] El-Gendy MA, Shin KG. Equation-Based packet marking for assured forwarding services. In: Proc. of the IEEE INFOCOM 2002. New York, 2002. 845−854. http://kabru.eecs.umich.edu/papers/publications/2002/mgendy_infocom02.pdf

[13] Ns-2 Network simulato. 2001. http://www.isi.edu/nsnam/ns