第34卷 第20期 计 算 机 工 程 2008年10月
Vol.34 No.20 Computer Engineering October 2008
·网络与通信· 文章编号：1000—3428(2008)20—0091—02 文献标识码：A 中图分类号：TP393

基于RED算法的非线性拥塞控制

李金东，马东堂，李 卫，王 杉
(国防科技大学电子科学与工程学院，长沙 410073)

摘 要：由于RED算法是采用丢包率随平均队列长度线性变化的方法，因此导致网络在拥塞并不严重的时候丢包率较大，在拥塞比较严重的时候丢包率较小，拥塞控制能力较低。该文提出非线性平滑算法通过对RED算法的丢包率函数进行非线性平滑，在最小阈值时丢包率增长速度比较小，在最大阈值时丢包率增长速度比较大，有效地控制了平均队列长度，具有较好的拥塞控制能力。NS2仿真结果表明该算法对丢包率、端到端时延、吞吐量以及时延抖动等性能均有较明显的提高。

关键词：主动队列管理；拥塞控制；随机早期检测

Non-linear Congestion Control Based on RED Arithmetic

LI Jin-dong, MA Dong-tang, LI Wei, WANG Shan
(College of Electronic Science and Engineering, National Univ. of Defense Technology, Changsha 410073)

【Abstract】Because RED adopts the way that drop packet ratio varies following average queue length, as a result, drop packet ratio is high when the network congestion is not serious and drop packet ratio is low when the network congestion is serious, so the congestion ability is not effective. This paper puts forward a non-linear congestion control arithmetic for this shortcoming, and the arithmetic can control average queue length effectively by non-linear control to the drop packet function. The non-linear RED arithmetic has made a visible improvement on drop packet ratio, time delay, througput, time delay jitter by NS2 simulation, and result proves the arithmetic is effective.

【Key words】active queue management; congestion control; random early detection

1 概述

随着Internet网络的发展，网络拥塞[1]已成为当前研究的热点，基于主动队列管理的RED算法[2]自从1993年被Floyd等人提出以来已经作为RFC2309推荐的AQM惟一候选算法。但由于该算法对参数设置敏感，在不同的网络环境中需要不同的参数设置[3]，这个先天性缺点导致了该算法并没有在实际应用中被广泛应用。后来很多人对RED算法进行了研究和改进，1999年Wu-chang Feng等人提出了参数自配置RED算法[4]，但该算法没有克服参数敏感性的缺点；2001年Floyd等人又提出了参数自适应RED算法[5]，但该算法的自适应能力不强，当遇到复杂网络环境时该算法的性能低于RED算法；2002年Feng Wuchang等人又提出了基于丢包和链路空闲事件的BLUE算法[6]，该算法虽然实现较为简单，丢包率较小，但时间延迟较大，使拥塞控制机制变得比较迟钝。本文在对RED算法研究的基础上，提出了非线性拥塞控制算法，通过采用该算法对丢包率函数进行控制，以此控制平均队列长度在期望值附近，以期达到低的丢包率和高的吞吐量，通过NS仿真可以看出该算法具有较好的拥塞控制能力，验证了算法的有效性。

2 算法描述和分析

2.1 随机早期检测算法

RED算法使用指数平均算法来估算路由器上的平均队列长度，得到的估算结果被用来与两个阈值进行比较，当估算得到的平均队列长度小于最小缓冲阈值时，判断此时网络没有发生拥塞现象，所有到达的分组都被插入到当前队列，进行正常的分组转发；当估算得到的平均队列长度大于最大缓冲阈值时，所有的到达分组都被标记。当所有的被标记分组都被丢弃或者所有的流的发送源端都能对分组标记做出合理的反应时，RED算法设计保证了平均队列长度不会一直超出最大缓冲阈值很多。当平均队列长度的估算值介于最小缓冲阈值与最大缓冲阈值之间时，所有的分组被标记的概率记为P_a，这一丢弃概率P_a是队列平均长度估算值avg_L的一个函数。计算公式如下：

$$avg_L(t_{n+1}) = (1-\omega_q)avg_L(t_n) + \omega_q \cdot q(t_n) \quad (1)$$

$$P_a = \max_p(avg_L - \min_{th})/(\max_{th} - \min_{th}) \quad (2)$$

其中，式(1)为计算平均队列长度公式；式(2)为计算丢包率公式。$avg_L(t_n)$，ω_q，$q(t_n)$，$\min_{th}$，$\max_{th}$，$\max_p$，P_a的含义分别为平均队列长度、加权系数、瞬时队列长度、最小缓冲阈值门限、最大缓冲阈值门限、最大丢包率、丢包率。平均队长和丢包率的关系如图1(直线1)所示。

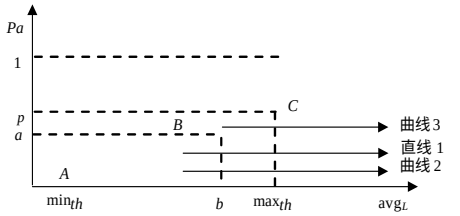


图1 平均队长和丢包率的关系曲线

基金项目：“十一五”通信装备预研基金资助项目(11001030107)；国家部委科技重点实验室基金资助项目

作者简介：李金东(1977－)，男，硕士研究生，主研方向：移动异构网络中的主动队列管理；马东堂，副教授、博士；李 卫、王 杉，博士

收稿日期：2007-11-20 **E-mail：**qdljd2008@sina.com

由图 1(直线 1)可以看出当平均队列长度介于最小门限和最大门限之间时，丢包率随平均队列长度的增加而线性增加。

2.2 非线性平滑算法(Non-Linear RED, NLRED)

拥塞控制的目的就是保持网络中的节点具有较高的吞吐量和较低的端到端延时，但是在现实网络传输过程中端到端延时和吞吐量总是对立的，如何在两者中维持一个平衡值是本文的研究方向。RED 算法的缺点在于当平均队列长度在最小门限附近时，容易产生较高的丢包率，而此时网络并不处于严重拥塞状态，需要较低的丢包率；当平均队列长度在最大门限附近时，容易产生较低的丢包率，而此时网络已经处于严重的拥塞状态，需要较高的丢包率。而非线性平滑算法有效地解决了这个问题，其数学模型如下：

直线 1 的函数为

$$P_a=\max_p(avg_L-\min_{th})/(\max_{th}-\min_{th}) \quad avg_L\in(\min_h,\max_{th}) \tag{3}$$

A, B, C 分别为直线 1 上的 3 点，A 点坐标为 $(\min_{th},0)$，B 点坐标为 $(2\min_{th},\frac{\min_{th}\bullet\max_p}{\max_{th}-\min_{th}})$，$C$ 点坐标为 $(\max_{th},\max_p)$。设曲线 2 和曲线 3 的二次函数方程为

$$P_a=A_1avg_L^{\ 2}+B_1avg_L+C_1 \tag{4}$$

$$P_a=A_2avg_L^{\ 2}+B_2avg_L+C_2 \tag{5}$$

其中，设 A 点为曲线 2 的顶点，曲线 2 的对称轴方程为

$$-\frac{B_1}{2A_1}=\min_{th} \tag{6}$$

C 点为曲线 3 的顶点，曲线 3 的对称轴方程为

$$-\frac{B_2}{2A_2}=\max_{th} \tag{7}$$

将 A, B 点坐标代入式(4)，结合式(6)得

$$P_a=\frac{\max_p}{\min_{th}(\max_{th}-\min_{th})}(avg_L-\min_{th})^2 \quad avg_L\in(\min_{th},\ 2\min_{th}) \tag{8}$$

将 B, C 点坐标代入式(5)，结合式(7)得

$$P_a=\frac{\max_p}{(\max_{th}-\min_{th})(2\min_{th}-\max_{th})}(avg_L-\min_{th})^2+\max_p \quad avg_L\in(2\min_{th},\ \max_{th}) \tag{9}$$

以上式子中，$2\min_{th}<\max_{th}$。

平均队长和丢包率的关系曲线如图 1(曲线 2 和曲线 3)所示。由图 1 可以看出，当平均队列长度介于最小门限和 2 倍最小门限时，丢包率按照曲线 2 轨迹变化，其值随平均队列长度的增加缓慢增加，当平均队列长度大于最小门限值 2 倍时，丢包率按照曲线 3 轨迹变化，其值随平均队列长度的增加快速增加。这样通过对丢包率进行非线性平滑增强了对网络拥塞的调节能力，提高了网络的资源利用率。

3 算法的仿真和分析

3.1 仿真模型与参数设计

仿真拓扑采用双哑铃结构，如图 2 所示。

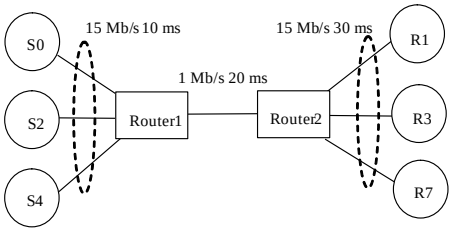
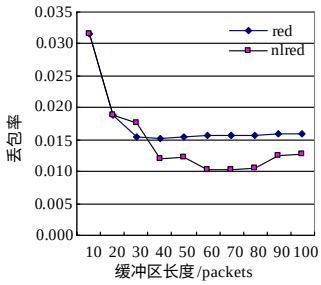


图 2 仿真拓扑结构

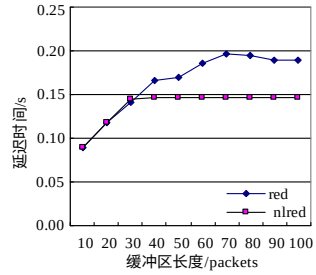
在图 2 中，S0, S2, S4 为发送源节点，R1, R3, R7 为接收节点。每个源节点和路由器 Router1 之间的带宽为 15 Mb/s，时延为 10 ms。Router1 和 Router2 之间的瓶径带宽为 1 Mb/s, 时延为 20 ms。Router2 与每个接收节点间的带宽为 15 Mb/s，时延为 30 ms。Router1~Router2 的队列长度按 10 packets ~ 100 packets 变化。仿真时间为 500 s，包长为 1 000 Byte。RED 参数设置：$\min_{th}=10$，$\max_{th}=30$，$\max_p=0.1$，$\omega_q=0.003$，$\min_{th},\max_{th},\max_p,\omega_q$ 参数的意义分别为：最小门限，最大门限，最大丢包率，权值。

3.2 仿真结果及分析

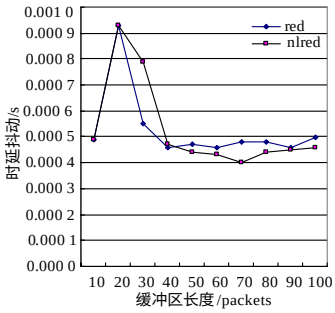
图 3 为 RED 算法和 NLRED 算法的性能仿真数据曲线图。图 3(a)~图 3(d)分别为丢包率、端到端平均时延、时延抖动、吞吐量随缓冲区队列长度变化的曲线图。由图 3(a)~图 3(c)可以看出，当缓冲区队列长度大于 30 packets 时，NLRED 算法在丢包率、端到端平均时延、时延抖动等方面都优于 RED 算法，由图 3(d)可以看出 NLRED 算法的吞吐量略高于 RED 算法。



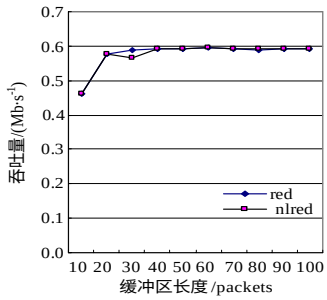
(a)丢包率随缓冲区长度变化曲线



(b)平均时延随缓冲区长度变化曲线



(c)时延抖动随缓冲区长度变化曲线



(d)吞吐量随缓冲区长度变化曲线

图 3 算法性能仿真曲线

(下转第 95 页)