一种快速收敛的 RED 改进算法

A Rapid Convergence RED Improvement Algorithm

汪华斌　（惠州学院 计算机科学系 广东惠州 516007）
（中南大学 信息科学与工程学院 湖南 长沙 410083）
刘卫国　（中南大学 信息科学与工程学院 湖南 长沙 410083）

摘　要： 针对 RED 算法存在的不足，根据其算法设计思想，系统地研究了 IETF 推荐用于路由器队列管理的 RED 及 GentleRED 算法的性能，提出一种新的改进算法 RCRED。该算法的主要思想是当平均队列长度在最小门阀值和另一个阈值之间使丢包概率采用一种平滑的 n 次高阶函数收敛机制。采用 NS2 仿真分析的方法，通过大量仿真实验，结论表明 RCRED 算法在提高系统稳定性和可靠性、提高链路利用率、减少丢包率等网络性能上更有效。

关键词： 网络仿真 随机早期丢弃 主动队列管理 拥塞控制

1 引言

DropTail 是路由器最简单的分组丢弃方法，根据先来先服务（FCFS）的原则尽最大努力传送（Best Effort），直到路由器队列缓冲区队列满或溢出，新到达的分组全部被丢弃。DropTail 算法实现简单，一直被广泛地应用。但由于 Internet 中业务流具有突发性，易造成缓冲区队列满时所有连接的连续多个分组同时被丢弃，使源端同时进入慢启动阶段，导致全局同步，降低网络的性能。

主动队列管理（Active Queue Management，AQM）[1] 作为 Internet 端到端拥塞控制的一项增强手段，有效地克服了 DropTail 算法策略下的满队列、业务流的死锁以及全局同步问题。在现有的 AQM 方案中，链路的拥塞都是通过队列长度、注入网络的分组速率、缓冲区队列溢出与空白等网络特性，以及这些特性的结合来衡量的。

1999 年 S. Floyd 和 V. Jacobson 等提出来的 RED[2] 算法是最典型的 AQM 拥塞控制机制算法，能够提供比 DropTail 算法更佳的网络性能，目前已成为 IETF 推荐的唯一的 AQM 候选策略。RED 算法和 DropTail 算法相比，避免了由于连续丢包导致的全局同步现象，有效地提高链路带宽利用率和减小平均队列长度。但其仍具有参数设置复杂和不能有效估计拥塞的严重性等问题。随着 RED 应用的增多，研究者提出了许多形式的 RED 算法的改进策略，目前，Internet 上广泛使用的主动队列管理方案主要有 RED[2]、GentleRED[3]、SRED、DSRED、FRED 和 Adaptive RED 等。本文结合 RED 和 GentleRED 算法的主要思想，提出了一种改进的 RCRED 算法，以一种 n 次高阶函数来动态的调整分组丢弃概率，在平均队列接近最小门阀值 Qmin 时以较低概率丢弃分组，而当平均队列超过最大门阀值 Qmax 时继续以 n 次高阶函数修正的较高概率丢弃分组，有效地提高了链路的利用率和吞吐量，进一步提高网络的性能。

2 RED 及 GentleRED 算法描述

2.1 RED 算法描述

S. Floyd 等提出的随机早期检测 RED 算法，其基本思想是：通过检测路由器缓冲区的平均队列长度来判断是否会引起网络拥塞，并在平均队列长度达到最小门阀值 Qmin 时，随机的选择部分新到达的分组进行丢弃或者标记，并通知源端减小拥塞窗口，降低分组发送速率，从而缓解网络拥塞，原理如图 1。RED 算法中缓冲区队列是一个 FIFO 的分组队列。缓冲区队列的利用率通过队列的平均长度来反映，队列长度一般以分组的个数为单位。RED 算法有以下几个重要参数，平

均队列长度 Qavg、队列长度两个门阀值 Qmin、Qmax，系统设定的分组丢弃的最大概率为 Pmax 和计算平均队列长度权值 Wq。

当新分组到达路由器时，RED 算法先按低通滤波器计算平均队列长度，如公式(1)：

$$Qavg(new) = (1 - wq) * Qavg(olg) + wq * Qcur \quad (1)$$

其中 Qcur 为瞬间队列长度，Wq 为计算平均队列的权值，其大小决定 Qcur 对 Qavg 结果的影响程度。

如果平均队列长度 Qavg 小于最小门阈值 Qmin，则该分组进入队列；如果平均队列长度位于最小阈值 Qmin 和最大阈值 Qmax 之间，则通过下列公式(2)、(3)计算丢弃概率 P，并以 P 概率丢弃该分组，如果平均队列大于最大门阀值 Qmax，则丢弃该分组。计算分组丢弃概率 P 的公式如下：

$$pb = \begin{cases} 0 & Qavg \le Qmin \\ P\,max * \dfrac{Q\,avg - Qmin}{Qmax - Qmin} & Qmin < Qavg < Qmax \\ 1 & Qavg \ge Qmax \end{cases} \quad (2)$$

$$P = \frac{pb}{1 - count * pb} \quad (3)$$

从公式(2)可以看出，如果 Pmax 设置得太大或者网络流量负载较轻，则平均队列长度会靠近 Qmin；如果 Pmax 设置得太小或者网络流量负载较重，则平均队列长度会在 Qmax 附近。RED 算法依赖平均队列长度来判断是否会发生网络拥塞，然而平均队列长度受网络拥塞和控制参数的影响非常巨大，再者概率 P 不仅和 Qavg 有关，而且还和上次丢弃分组开始到现在连续进入队列的分组数量 Count 有关，如公式(3)所示，随着 Count 的增加，下一个分组被丢弃的可能性也在缓慢增加，这主要是为了均匀间隔的丢弃分组，以消除对突发流的偏见和全局同步现象。

2.2 GentleRED 算法描述

GentleRED 算法继承了 RED 算法的主要思想，当平均队列小于最大门阀值时，其丢弃概率的计算与 RED 算法完全一致，而当平均队列 Qavg 大于 Qmax 时，则不象 RED 算法一样丢弃概率由 Pmax 到 1 的跳跃，而是以另一种线性关系计算 P 的概率，然后以 P 概率来丢弃新到达路由器的分组，原理如图 1。其算法函数表示为公式(4)：

$$pb = \begin{cases} 0 & Qavg \le Q\,min \\ Pmax * \dfrac{Q\,avg - Q\,min}{Q\,max - Q\,min} & Q\,min < Qavg < Q\,max \\ (1 - P\,max) * \dfrac{Qavg - Q\,max}{Q\,max} + P\,max & Qmax \le Qavg < 2Q\,max \\ 1 & 2Q\,max \le Qavg \le Buffersize \end{cases} \quad (4)$$

3 改进的 RCRED 算法

RED 和 GentleRED 算法的丢弃概率与平均队列长度是一种线性关系，在最小阈值附近时进入网络的分组逐步增加，实际上并非处于一种严重拥塞状态，而 RED 和 GentleRED 以较高的丢包率丢弃分组，显然对提高网络性能没有提高，而当平均队列长度超过 Qmax 时，RED 算法丢弃所有新到达的分组，GentleRED 以线性关系缓慢提高分组的丢弃概率，这两种丢弃规则都较为偏激。所以本文引入基于队列长度的 n 次高阶函数，改进算法称为 RCRED，在最大门阀值 Qmax 处相交，此时三种算法的丢弃概率大小相同。因为 n 次高阶函数在队列长度接近最小阈值时丢包率比 RED 算法低，但是又不像增加最小阈值大小方法，使丢包率为零而失去对网络的拥塞控制，在最小阈值附近变化较为平滑，随着平均队列长度的增长缓慢增长。当平均队列长度接近或超过最大阈值时，其丢包概率的增长速度也随之加快，按 n 次高阶函数变化以更快的收敛速度实现分组丢弃概率变化的平滑化，快速退出网络拥塞状态，原理如图 1 所示。因此 RCRED 能够在链路相对空闲时，降低分组的丢弃概率，而在拥塞状态下迅速提高丢弃概率，从而增强了对网络拥塞的控制能力，提高网络的性能指标。

RCRED 算法的丢弃概率修正描述为公式(5)：

$$pb = \begin{cases} 0 & Qavg \le Q\,min \\ P\,max * \left(\dfrac{Qavg - Q\,min}{Qmax - Qmin}\right)^n & Q\,min < Qavg < (Q\,max - Q\,min)\sqrt[n]{1/P\,max} + Qmin \\ 1 & Qavg \ge (Q\,max - Q\,min)\sqrt[n]{1/P\,max} + Qmin \end{cases} \quad (5)$$

公式(5)中，当 n=2 时，要使得改进算法的另一个阀值在 Qmax 和 2Qmax 之间，Pmax 可取 0.2。而当 n=3 或者更大时，$Q\,max < (Q\,max - Q\,min)\sqrt[n]{1/P\,max} + Q\,min < 2Q\,max$，则取 Pmax=0.1，本文中采用 n=3，Pmax=0.1。

4 仿真及网络性能比较分析

4.1 仿真模型

为了验证 RCRED 算法的性能，作者进行了大量的

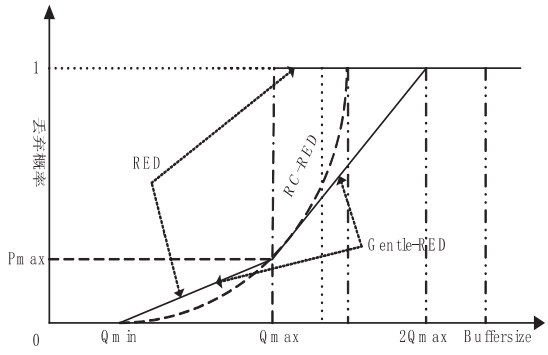


图 1　丢弃概率与平均队列长度的关系

仿真实验，并与 RED 算法（为方便描述，下文称为 Original RED）及 GentleRED 算法进行了性能比较和评价。本文采用的仿真环境为 Winxp + Cygwin + NS2. 30[4]。仿真采用的拓扑结构如图 2 的哑铃模型。在 R1 ~ R2 路由器之间的瓶颈链路，为各业务流共同竞争的资源，带宽和延迟分别为 10Mb 和 10ms，路由器 R1 缓冲区为 100 个分组，最小门阀值 Qmin = 10 和最大门阀值 Qmax = 30，要求大于 2Qmax，以便更好的仿真验证 RCRED 算法的性能。其他链路带宽和延迟均为 20Mb 和 1ms。TCP1 ~ TCPn 作为 TCP 源端，TCPsink1 ~ TCPsinkn 作为 TCP 接收端，UDP1 ~ UDPn 作为 UDP 源端，NULL1 ~ NULLn 作为 UDP 接收端，仿真时间为 20s。

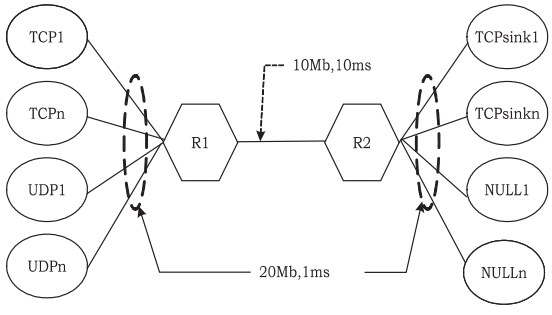


图 2　仿真实验拓扑图

4.2 仿真结果及分析

由于当前 Internet 中约 90% ~95% 以上的数据为 TCP 流[5]，考虑到 UDP 和 TCP 流的不同机制，采用一个 UDP 流作为背景，多个 TCP 流参与竞争的模式。为了使实验更好进行，且数据更真实有效，仿真使用突发流进行实验，在实验中 Original RED 和 GentleRED 算法及其改进算法 RCRED 均采用最小门阀值 Qmin = 0. 1 * ql 和最大门阀值 Qmax = 0. 3 * ql，ql 为路由器缓冲队列，权重 Wq = 0. 002，最大丢弃概率 Pmax = 0. 1[6]。

在实验中根据图 2 所示实验拓扑图设计由多个 TCP 连接突发流和一个 UDP 连接作为背景流的情况，TCP 突发流采用在 0 – 10s 之间随机开始，在 11 – 20s 之间随机结束。利用该模型对三种算法分别进行仿真，仿真结果如表 1、表 2：

表 1　三种算法的丢包率对比

TCP 连接	OriginalRED	GentleRED	RCRED
N = 1	0. 000000%	0. 000000%	0. 000000%
N = 5	0. 286434%	0. 286434%	0. 181050%
N = 10	0. 750302%	0. 750302%	0. 591872%
N = 15	1. 090711%	1. 090711%	1. 040667%
N = 20	1. 692186%	1. 728774%	1. 593527%
N = 25	1. 927553%	2. 093394%	1. 970421%
N = 30	2. 245196%	2. 262180%	2. 384150%
N = 35	2. 620361%	2. 767664%	2. 657057%
N = 40	3. 121836%	3. 029390%	3. 061158%
N = 45	3. 312794%	3. 508395%	3. 346538%
N = 50	3. 535127%	3. 860090%	3. 440933%

表 1 的数据表明，原始的 OriginalRED 和 GentleRED 在节点数为 1、5、10、15 时，其丢包率大小相同，改进的 RCRED 算法的丢包率较小，此时网络处于无拥塞或轻度拥塞，平均队列 Qavg 小于 Qmax，而当平均队列 Qavg 大于 Qmax 时，OriginalRED 算法丢包率最小，RCRED 算法次之，GentleRED 最大，如图 3：

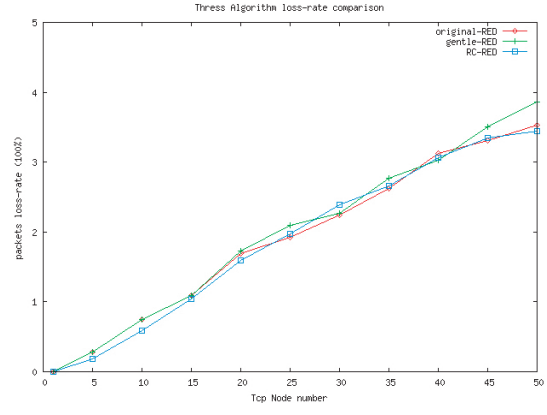


图 3　三种算法丢包率对比

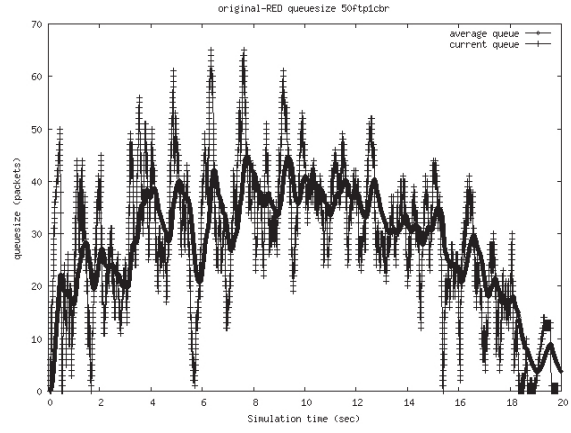
表 2　三种算法的链路利用率对比

TCP 连接数	OriginalRED	GentleRED	RCRED
N = 1	48.3677%	48.3677%	48.3677%
N = 5	82.9606%	82.9606%	83.8958%
N = 10	90.2734%	90.2734%	91.0085%
N = 15	95.2014%	95.2014%	95.3195%
N = 20	94.7518%	95.0365%	95.3984%
N = 25	94.7992%	95.1482%	95.3856%
N = 30	94.6270%	95.2019%	95.1845%
N = 35	98.0688%	88.7880%	97.7394%
N = 40	97.9082%	98.0619%	97.8381%
N = 45	98.0181%	97.8149%	98.0979%
N = 50	90.4536%	98.0248%	96.7813%

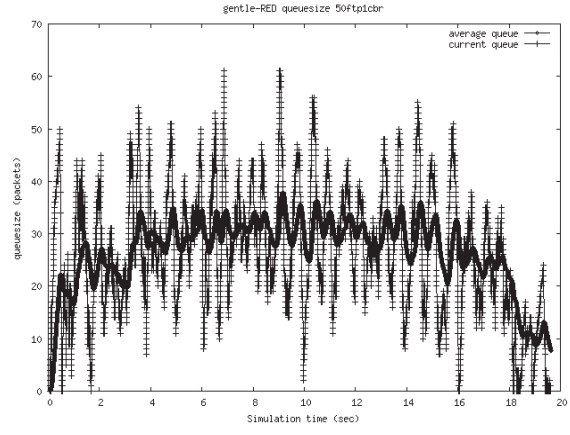
表 2 仿真数据表明 RCRED 算法在轻度拥塞和中度拥塞情况下，链路利用率比 OriginalRED 和 GentleRED 算法稍大，而在重度拥塞时三种算法的链路利用率相差不大。在 RCRED 算法策略下，当平均队列长度在 Qmin 附近时以较低的丢弃概率丢弃分组，而当平均队列长度在大于 Qmax 时才以 n 次高阶函数快速增长逼近 1。在正常情况下，RCRED 算法下分组进入队列的数量增加，分组在队列中的排队时间也随之增长，尽管如此，实验结果表明改进的 RCRED 算法在路由器的缓冲区队列长度变化较 OriginalRED 和 GentleRED 算法稳定（如图 4 所示），振荡较弱，稳定性较好，RCRED 算法平均队列较 OriginalRED 算法稍大，有效提高了缓冲区的利用率，网络吞吐量也稍有提高。

5　结论

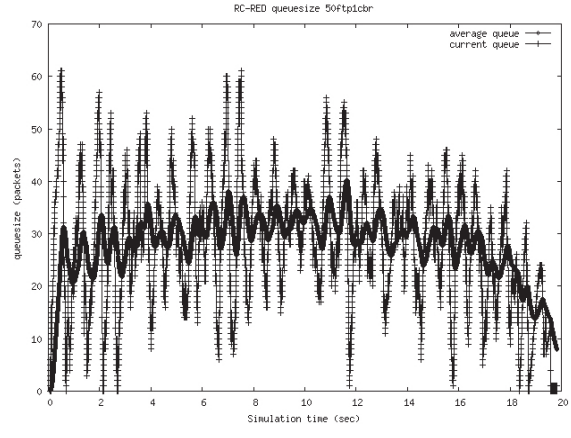
本文分析了 OriginalRED 和 GentleRED 算法的原理，以及算法实现算法的主要策略，阐述了其不足之处，结合原有算法的主要思想，提出的一种非线性 n 次高阶函数作为快速收敛的分组丢弃概率的 RCRED 算法。对于高阶函数的 n 次幂的取值不同，仿真结果稍有不同，作者通过 n 次（n = 3）函数修正结论表明：RCRED 算法在提高链路利用率、缓冲区利用率和网络吞吐量及减少丢包率等多种性能上比 OriginalRED 和 GentleRED 算法更优，在解决网络业务流突发的问题上，RCRED 算法比 OriginalRED 和 GentleRED 算法更稳定和可靠。



(a) OriginalRED



(b) GentleRED



(c) RCRED

图 4　50 个 TCP 连接下队列长度变化

（下转第 71 页）

(上接第 65 页)

参考文献

1 Braden B, Clark D. Recommendation on queue management and congestion avoidance in the internet. Request for Comments (RFC) 2309. http://www.ietf.org/rfc, 2003-02-15.

2 Floyd S, Jacobson V. Random early detection gateways for congestion avoidance. IEEE/ACM Transaction on Networking. August 1993, 1(4): 397-413.

3 Floyd Sally. Recommendation on using the "gentle_" variant of RED. http://www.icir.org/floyd/red/gentle.html.

4 UCN/LBL/VINT. Network Simulator - NS2. http://www-mash.cs.berkeley.edu/ns.

5 Thompson K, Miller GJ, Wilder R. Wide Area Internet Traffic Patterns and Characteristics. IEEE Network, 1997, 11(6): 10-23.

6 Floyd S. RED: Discussions of Setting Parameters. http://www.icir.org/floyd/REDparameters.txt, November 1997.