

一种适用于实时多媒体业务的随机早期检测算法

安智平, 张德运, 赵东平, 高磊

(西安交通大学电子与信息工程学院, 710049, 西安)

摘要: 针对随机早期检测算法 (RED) 可能连续丢弃同一数据流分组的问题, 提出了一种适用于实时多媒体业务的主动队列管理算法. 在网络未发生拥塞时, 该算法以一定的概率丢弃到达的分组. 在丢弃分组时要根据瞬时丢包率判断该数据流最近的分组丢弃情况, 如果最近丢包率比较高则放弃丢弃, 避免连续丢弃该数据流的分组, 以保证多媒体应用的服务质量. 在网络拥塞时, 丢弃部分数据流的分组, 避免了因拥塞造成的大部分多媒体应用同时中断. 实验结果表明, 不论网络是否拥塞, 所提算法都能为实时多媒体应用提供较好的服务质量.

关键词: 主动队列管理; 随机早期检测; 拥塞; 服务质量

中图分类号: TP393 **文献标识码:** A **文章编号:** 0253-987X(2004)10-1061-04

Real2Time Enhanced Random Early Detection Algorithm for Multimedia Service

An Zhiping, Zhang Deyun, Zhao Dongping, Gao Lei

(School of Electronics and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China)

Abstract: Focusing on the problem that the random early detection (RED) may drop packets from the same data flow packets continuously, an active queue management algorithm which is applicable for the real time multimedia service is proposed. The algorithm drops the arriving packets with certain probability when the congestion has not been occurred yet. In dropping packets, the recent situation of packets dropped is judged according to instantaneous loss rate. It endeavors to avoid dropping packets from the data flow continuously to ensure the QoS of the multimedia applications if the recent dropping packets rate is high. When network is congested, all packets of the part of flows are dropped to avoid most of application being interrupted due to packet loss. The experiments indicate that the proposed algorithm provides better QoS in real2time multimedia service, whether the network is congested or not.

Key words: active queue management; random early detection; congestion; quality of service

解决网络阻塞的一种方法就是在路由器中引入主动队列管理 (AQM) 机制, 以便为所有的连接提供一个公平高效的服务^[1]. 随机早期检测 (RED)^[2] 是指, 在检测到路由缓冲队列发生拥塞时以一定的概率丢弃数据包, 以避免拥塞和全局同步, 避免突发流量数据包全部丢弃^[3].

实时多媒体应用一般能容忍一定的丢包率, 如果超过了这个范围, 应用就无法再正常运行^[4]. RED 算法如果按照相同概率丢弃所有数据流的分组数据包, 会导致短时间内大部分业务的服务质量同时下

降, 甚至业务中断, 造成大量的带宽浪费^[5]. 为了解决以上问题, 本文基于 RED 算法提出了一种适用于实时多媒体业务的主动队列管理算法 (MMRED).

1 RED

RED 算法^[1] 是通过检测路由器的平均队列长度 A 来获得网络拥塞的信息, 并以一定的概率丢弃数据包的方式来通知发送方. 该算法中的数据包丢弃概率是 A 的函数, 它通过将 A 限制在一定范围来避免突发性丢弃大量的数据包. 将 A 与最小阈值

收稿日期: 2003-11-30. 作者简介: 安智平 (1975~), 男, 博士生; 张德运 (联系人), 男, 教授, 博士生导师.

T_{min} 、最大阈值 T_{max} 比较,可以计算出数据包丢弃概率

$$P(A) = \begin{cases} 0 & A < T_{min} \\ P_{max} \frac{A - T_{min}}{T_{max} - T_{min}} & T_{min} \leq A < T_{max} \\ 1 & A \geq T_{max} \end{cases} \quad (1)$$

式中: A 为采用指数加权移动平均算法得到的平均队列长度; T_{min} 和 T_{max} 分别为最小和最大阈值; P_{max} 是队列平均长度为最大阈值时的分组数据包丢弃概率。

2 MMRED 算法

2.1 瞬时丢包率的定义

定义1 每到达一个分组数据包,瞬时丢包率

$$L_{t+1} = \begin{cases} L_t(1 - \alpha) & \text{如果该分组数据包未被丢弃} \\ L_t(1 - \alpha) + \alpha & \text{如果该分组数据包被丢弃} \end{cases} \quad (2)$$

式中: L_{t+1} 是最新的瞬时丢包率; L_t 是前次分组到达时的瞬时丢包率, $L_0 = 0$; α 是瞬时丢包率的计算权值,取值范围应在 $(0, 1)$ 之间. 当 $\alpha = 0.15$ 时,瞬时丢包率可简化为

$$L_{t+1} = \begin{cases} \frac{L_t}{2} & \text{如果该分组数据包未被丢弃} \\ \frac{L_t}{2} + \frac{1}{2} & \text{如果该分组数据包被丢弃} \end{cases} \quad (3)$$

也就是说,每到达一个分组数据包,原瞬时丢包率衰减 $1/2$,因此丢包率可以一个二进制小数 $0.X_n X_{n-1} X_{n-2} \dots$ 来表达. 根据二进制数的特性,可以得到下述定理.

定理1 对于一个数据流,如果瞬时丢包率为 L_t ,那么从最后一次丢弃分组数据包后,连续未丢弃的分组数据包个数为

$$x = \lceil \log_{1/2}(L_t) \rceil - 1 \quad (4)$$

由此可以看出,取 $\alpha = 0.15$ 时,根据瞬时丢包率可以判断过去一段时间内的丢包情况.

2.2 短时平均丢包率

对于实时应用,本算法保留了两个丢包率阈值 (L_{min} 和 L_{max}). 丢包率低于 L_{min} 时,实时应用运行良好;丢包率高于 L_{max} 时,应用无法运行. 当丢包率介于 L_{min} 和 L_{max} 之间时,系统能够运行,用户尚且能够接受. 若采用短时平均丢包率 L 作为数据流的平均

丢包率,则有下述定义.

定义2 数据流 K 的短时平均丢包率 L 是指,数据流 K 从最后一次丢弃分组数据包至今的平均丢包率.

根据瞬时丢包率,利用定理1可以获得最近连续未丢弃的分组数据包数,因此可以用瞬时丢包率获得短时平均丢包率.

定理2 如果数据流 K 的瞬时丢包率为 $L_t(K)$,那么它的短时平均丢包率为

$$L(K) = \frac{1}{\lceil \log_{1/2}(L_t(K)) \rceil} \quad (5)$$

对于数据流 K ,如果 $L(K) \leq L_{min}$,称 K 处于良好状态,如果 $L_{min} < L(K) < L_{max}$,称 K 处于临界状态,如果 $L(K) \geq L_{max}$,称 K 处于失效状态.

2.3 算法描述

实时应用 RED 的数据包丢弃概率如图1所示,其算法根据队列平均长度 A 分为如下3个阶段.

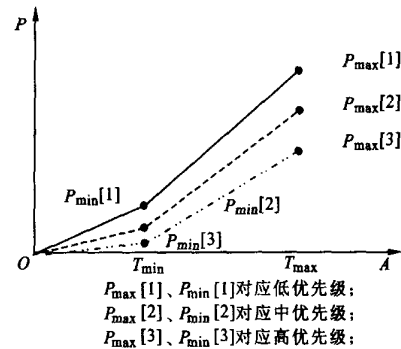


图1 实时应用 RED 的数据包丢弃概率示意图

(1) 在宽松丢弃阶段 ($A < T_{min}$), 队列的长度较短,尚未发生拥塞,因此以较小的概率

$$P = P_{min} \frac{A}{T_{min}} \quad (6)$$

丢弃分组数据包.

(2) 在严格丢弃阶段 ($T_{min} \leq A < T_{max}$), 队列的长度有所增长,预示着可能会发生拥塞,此时以概率

$$P = P_{min} + (P_{max} - P_{min}) \frac{A - T_{min}}{T_{max} - T_{min}} \quad (7)$$

丢弃分组数据包. 其中, P_{min} 表示队列平均长度为 T_{min} 时的分组数据包丢弃概率, P_{max} 表示队列平均长度为 T_{max} 时的分组数据包丢弃概率. 对于不同优先级别的应用,它们的取值不同. P_{max} 的取值仍然参照 RED 算法推荐的数值,即 $P_{min} = \frac{P_{max}^{[2]}}{2}$.

(3) 在拥塞阶段 ($A \geq T_{max}$), 应该大量丢弃分组

数据包. 先从丢包率较高的数据流中集中丢弃分组数据包, 如果仍然发生拥塞, 再丢弃那些丢包率较高的数据流, 直到拥塞缓解. 这样能够避免同时大量丢弃所有数据流的分组数据包, 从而避免所有的应用同时失效.

为了避免短时间内连续丢弃而导致的数据流失效, 在丢弃某个分组数据包前, 首先检查数据流状态, 如果是临界状态或失效状态, 就尽可能不丢弃该分组数据包.

在宽松丢弃阶段, 如果数据流处于临界状态或失效状态, 就暂时不丢弃该分组数据包, 而把需要丢弃的分组数据包个数加 1. 在严格丢弃阶段, 如果数据流处于失效状态, 就暂时不丢弃该分组数据包, 而把需要丢弃的分组数据包个数加 1. 当某个数据流应该丢弃的分组数据包个数大于 0 时, 在下次分组数据包到达时, 如果按照概率未丢弃, 则选择丢弃该分组数据包, 并把应丢弃的分组数据包个数减 1. 丢弃过程与正常情况下按照一定概率的丢弃过程相同.

在拥塞阶段, 如果要丢弃的分组数据包所属数据流处于临界状态或失效状态, 则丢弃该分组数据包, 并标记该数据流状态为失效, 该数据流以后的所有分组数据包都会被丢弃. 为了避免大量的应用同时失效, 以标记失效概率

$$P_m = \begin{cases} \frac{A}{3S}f(K) & \text{当 } K \text{ 为高优先级时} \\ \frac{2A}{3S}f(K) & \text{当 } K \text{ 为中优先级时} \\ \frac{A}{S}f(K) & \text{当 } K \text{ 为低优先级时} \end{cases} \quad (8)$$

的丢弃状态为失效的数据流. 其中, S 为缓冲队列的最大尺寸, 函数 $f(K)$ 的值依赖于数据流 K 的状态. 如果数据流 K 的状态为临界, $f(K) = 1/2$; 如果数据流 K 的状态为失效, $f(K) = 1$.

3 实验与分析

3.1 实验环境

实验环境如图 2 所示. 客户端 1~客户端 4 分别模拟一个多媒体业务终端向客户端 6 发送数据, 客户端 5 用来产生背景流量. Linux 路由器的转发能力限制为 2 Mb/s, 缓冲队列的缓冲区大小为 500 包, 最大和最小阈值分别为 200 和 100. 在实验中, 假设该多媒体业务应用的丢包率低于 10%, 效果良好, 丢包率低于 20% 时用户能够接受, 丢包率超过 20% 则用户无法忍受.

计算平均队列长度时, 权值取自 Floyd 推荐的

01002^[2]. 由于 P_{max} 的推荐值为 011, 因此本文中低优先级 $P_{max} = 011$, $P_{min} = 0105$, 而中优先级 $P_{max} = 0108$, $P_{min} = 0104$.

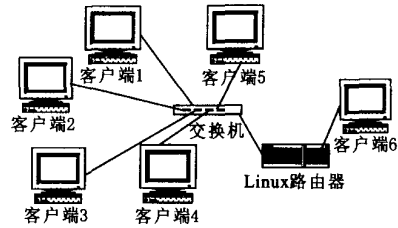


图 2 实时业务主动队列管理算法实验环境

采用的多媒体业务数据流量为 40 帧/s, 每一帧数据的大小按平均值为 1 000 B 的指数分布, 该业务的大约平均流量为 $10^3 \times 8 \times 40 = 312 \times 10^5$ b/s, 背景流量按平均值为 640 kb/s 的指数分布. 客户端 1~客户端 5 分别从第 11 s、第 31 s、第 51 s、第 71 s 和第 91 s 开始发送数据, 到第 110 s 时停止. 在本次实验中, 客户端 1、客户端 2、客户端 4 和客户端 5 的数据流的优先级均取中级优先, 客户端 3 的数据流取低级优先, 客户端 5 的背景流量随机地取中级优先和低级优先.

3.2 实验结果与讨论

图 3 是分别采用 RED 和 MMRED 算法时, 客户端 1~客户端 4 在 0~110 s 期间每一秒内被丢弃的分组数据包个数. 实验结果见表 1 和表 2. 从表 1 中可以看出, 采用 RED 算法时, 随着数据流的增加, 丢包率逐渐上升. 在 71~90 s 期间, 4 个客户端同时发送数据(没有背景流量)时, 丢包率大约在 8% 左右, 业务能够良好地进行. 在 91~110 s 期间, 4 个客户端同时发送数据, 客户端 5 也发送背景流量, 客户端 1~客户端 4 的数据流丢包率上升, 丢包数在 10 个左右, 丢包率大约为 25%, 超出了应用可以忍受的范围, 导致应用失效. 丢包率上升的原因是由于数据流量过大, 超过了路由器的转发限制.

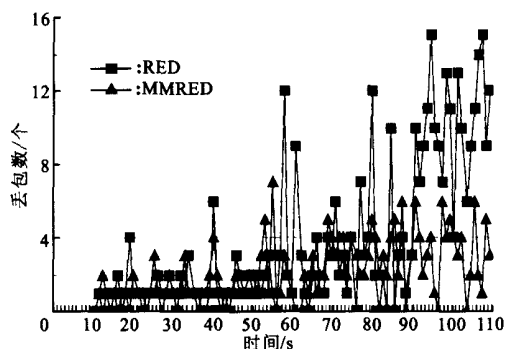
表 1 RED 与 MMRED 在 0~110 s 期间每一秒丢弃的分组数据包个数

| 时间 /s | 丢包率/包 s ⁻¹ | | | | | | | |
|--------|-----------------------|------|-------|------|-------|-------|-------|------|
| | U1 | | U2 | | U3 | | U4 | |
| | R | M | R | M | R | M | R | M |
| 10~30 | 1105 | 0195 | - | - | - | - | - | - |
| 31~50 | 1135 | 1135 | 1190 | 1195 | - | - | - | - |
| 51~70 | 2165 | 2115 | 3115 | 2125 | 2160 | 2160 | - | - |
| 71~90 | 3135 | 3120 | 3115 | 2185 | 4150 | 3175 | 4135 | 3100 |
| 91~110 | 9180 | 3125 | 10115 | 3165 | 11130 | 37115 | 10140 | 3180 |

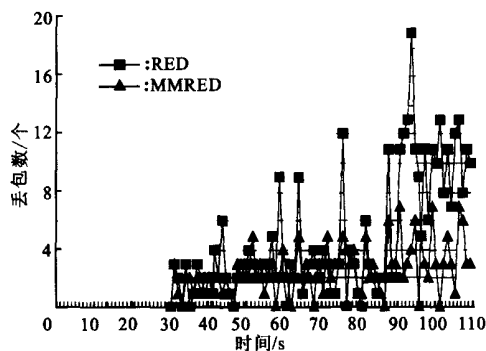
注: U1~U4 表示客户端 1~客户端 4; R 和 M 分别表示 RED 算法和 MMRED 算法.

表2 RED和MMRED丢包率标准差

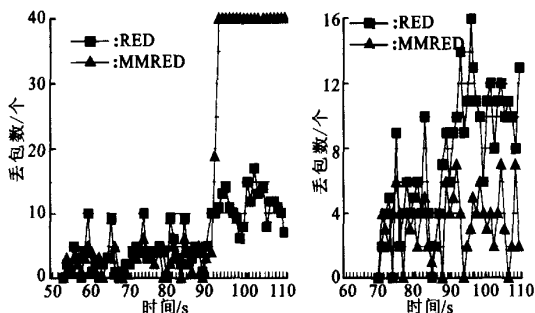
| 时间 /s | 丢包率标准差 | | | | | | | |
|--------|--------|------|------|------|------|------|------|------|
| | U1 | | U2 | | U3 | | U4 | |
| | R | M | R | M | R | M | R | M |
| 10~30 | 0194 | 0183 | - | - | - | - | - | - |
| 31~50 | 1142 | 0193 | 1152 | 0189 | - | - | - | - |
| 51~70 | 2196 | 1179 | 2128 | 1141 | 2176 | 1154 | - | - |
| 71~90 | 3127 | 1140 | 3120 | 1163 | 2152 | 1189 | 2192 | 1184 |
| 91~110 | - | 1189 | - | 2118 | - | - | - | 2104 |



(a) 客户端 1



(b) 客户端 2



(c) 客户端 3

(d) 客户端 4

图3 数据流每秒钟丢包个数对比

采用MMRED算法时,没有背景流量时的丢包率与采用RED算法没有明显区别.在91~110s期间,客户端5发送背景流量,客户端3的分组被全部丢弃,而客户端1、客户端2、客户端4的丢包率与71~90s期间没有背景流量时的丢包率接近.

因此可以得出,在网络负载较重时,RED算法基本上丢弃了所有数据流的分组数据包,这将导致大部分数据流的丢包率超过了可以忍受的限度,使得应用大量失效,MMRED算法通过丢弃部分数据流的分组数据包,从而为其他数据流节省了带宽.

从表2可以看出,采用MMRED算法时,其丢包个数的标准差明显小于RED算法,这表明MMRED算法的丢包率比较稳定.从图3中还可以看出,采用RED算法时,分组数据包的丢包率可能在某些时刻超过10%甚至超过20%,这样就会造成实时多媒体业务的暂时中断.采用MMRED算法时,除了在91~100s期间客户端3的数据流分组数据包丢包率超过20%,其余的数据流分组数据包丢包率从未超过20%.

4 结论

本文在RED算法的基础上提出了一种适用于实时多媒体应用程序的主动队列管理算法MMRED.该算法通过瞬时丢包率来判断每一个数据流最近的分组数据包丢弃情况,并且尽量避免连续丢弃分组数据包以保持实时多媒体业务的服务质量.在网络拥塞时,MMRED算法丢弃已经失效的所有分组数据包,为其他应用节省了带宽,避免了大部分应用同时失效.实验结果表明,无论网络是否发生拥塞,本文所提算法的性能比RED算法的性能更好.

参考文献:

- [1] Aweya J. An adaptive buffer management mechanism for improving TCP behavior under heavy load [A]. International Conference on Communications, Helsinki, Finland, 2001.
- [2] Floyd S, Jacobson V. Random early detection gateways for congestion avoidance [J]. IEEE/ACM Transactions on Net2 working, 1993, 1(4): 397 - 413.
- [3] Stallings W. High2speed networks and Internets: performance and quality of service [M]. 2nd ed. Beijing: China Machine Press, 2002. 485 - 491.
- [4] 温 斌. VOIP:IP 语音技术 [M]. 北京:机械工业出版社, 2000.
- [5] Aweya J, Ouellette M, Montuno D Y. A linear system analy2 sis of RED [J]. Computer Communications, 2002, 25(18): 1 736 - 1 750.

(编辑 苗 凌)