垃圾桶算法模型的理论分析

赵永祥,陈常嘉
(北京交通大学通信系,北京 100044)

摘 要: 本文在理论上给出了垃圾桶算法的数学模型,该模型能够从任意给定的系统参数设置,推导出垃圾桶系统的各个参数,并具有计算简单的特点. 本文通过将理论计算与仿真结果的比较,验证了本文所提出的模型确实是一个具有良好计算精度的实用化模型. 为了建立垃圾桶算法的数学模型,本文研究了早期随机丢弃 RED(Random early discard)系统的平均丢失率与平均队长的相互关系的理论问题,有效的解决了从平均丢弃率估计 RED 平均瞬时队长的问题.

关键词: TCP 友好流保护; 公平队列; RED; 缓存管理
中图分类号: TN393 **文献标识码**: A **文章编号**: 0372-2112 (2001) 10-1337-05

Theoretical Analysis of the Trash Algorithm

ZHAO Yong-xiang, CHEN Chang-jia
(Dept. of Communications, Northern Jiaotong Univ., Beijing 100044, China)

Abstract: A theoretical model of the trash algorithm is established in this paper based on our previous works of simulation study. In this model, all interested parameters of the Trash can be easily derived from given system settings. Simulation shows that this model is a useful practical model in its precision and simplicity. The relations between average queue lengths vs. loss rate of RED (Random early discard), are also studied and solved in this paper.

Key words: TCP friendly flow protection; fair queue; RED; buffer management

1 引言

传输控制协议(TCP, transmission control protocol)的端到端拥塞控制机制是 Internet 鲁棒性的一个极端重要的因素,它得以成立的一个基本假设是用户自觉地使用拥塞控制算法. 目前存在的大量应用却自觉或者不自觉的绕过或者不完全实现端到端拥塞控制机制,使得 Floyd 在文献[3]中重申端到端拥塞控制对于防止网络崩溃的极端重要性,指出需要在路由器实现一种鼓励使用端到端拥塞控制的机制,这种机制能够识别并惩罚非 TCP 友好流,并保护 TCP 友好流,本文简称这种机制为惩罚机制. 惩罚机制大致可分为三种类型:

(1) 具有流状态的惩罚机制,如文[4~6]所述,它们的主要特征是每个路由器需要保护一定的流状态,所以一般具有较好的惩罚能力,但复杂性高,难于实现,且可扩展性差.

(2) 核心无状态公平队列,如文[7,8]中所述,它们的主要特征是网络的核心部分可以在无流状态的条件下实现资源的公平分配,但前提是要求有一个功能强大的网络边缘,对用户做必要的入网控制.

(3) 完全无状态算法,这类算法的主要特征是不须要网络边缘的配合就可无状态实现公平的资源分配,但缺点是难以适合不同用户的不同要求,只能作到一般性的公平,这类算法在文[1,2,9,10]有介绍.

垃圾桶算法本身的性能仿真在文[2]已有介绍,垃圾桶算法与随机选择机制 CHOKe[10] (choose and keep responsive flows) 联合使用的性能仿真在文[1]中有介绍,本文的目的是在给出垃圾桶算法的一个简单数学模型,在目前关于无状态公平队列的研究中,已经建立较好数学模型的算法并不多,在 CHOKe[10] 的讨论中,作者只给出了一个开环的 M/M/1 模型,很难描述问题本身所固有的闭环控制机制,也很难反映 TCP 这种本质上是闭环工作的流的工作特征. 本文针对垃圾桶算法垃圾桶丢弃和 TCP 速率适配两个闭环控制环节,构造了较为精细的数学模型,较全面分析了垃圾桶控制能力(垃圾桶成分)、RED 丢弃概率、RED 平均队长、用户数据报 UDP (user datagram protocol) 吞吐量、TCP 吞吐量等参数之间的关系,给出了从给定系统参数(出口带宽、UDP 输入速率、TCP 个数和 RED 门限参数)直接得到垃圾桶控制能力(垃圾桶成分)、RED 丢弃概率、RED 平均队长、UDP 吞吐量、TCP 吞吐量等参数的精确方程组和近似方程组,近似方程组的求解非常简单,甚至在 Excel 的页面上就可简单完成,仿真表明在相当大的系统参数范围内,近似方程组的解与实际仿真结果匹配,是一个简便有效的实用模型.

收稿日期:2000-06-01;修回日期:2001-02-22

2 垃圾桶算法

垃圾桶算法:该算法的伪代码如图 1 所示,垃圾桶实际上是一个存储最近丢弃分组的分组头的表. 垃圾桶在初始化时,每丢弃一个分组就把此分组头按顺序记入称之为垃圾桶的表格,直到垃圾桶填满为止,实际上这个初始化过程可以省略. 算法将每个到达分组的分组头与随即从垃圾桶中取出的表项进行比较,如果属于同一个流(分组头相同),则丢弃到达分组,否则送入 RED 缓存器按照标准 RED 进行处理,RED 机制可能将分组保留,也可能将其丢弃,不论到达分组是因为何种原因被丢弃,算法都要求对垃圾桶进行刷新. 垃圾桶的刷新算法如下:从垃圾桶表随机选一项,将此表项中的分组头与被丢弃的分组头相比,如不相同则按某一个概率 β(这是本算法的一个可调整参数,称之为记忆概率)将被丢弃的分组头写入此表项.

```
const      N = dimension of trash;
IP_header trash(N); P, Q;
int        I = 0;
real       β;
up_date_trash(P){
  if ((I > N) and (random(1) < β)) trash(random(N)) = p;
  else initial_ _trash(P, I);
}
initial_ _trash(P, I);{
  trash(I + +) = P;
}
On arrival packet (P){
  M = random(N); Q = trash(M);
  if  (P = = Q){
    up_date_trash(P); drop P;
  } else{
    perform RED algorithm;
    if (RED drop P) up_date_trash(P);
  }
}.
```

图 1 垃圾桶算法伪代码

3 垃圾桶数学模型

这一节将针对整个惩罚系统进行建模和理论分析. 虽然这里的讨论只是针对输入的单个 UDP 流和多个 TCP 流这一情况的,但它可以容易地推广到多个 UDP 流的场合. 本节考虑的系统示于图 2,这是多个 TCP 流和一个恒比特率 CBR 业务竞争一个瓶颈链路的典型情况. 图中:C 为出口带宽;M 为输入的 TCP 流个数;$R_U = uC$ 为输入的 UDP 速率;$R_T = tC$ 为输入的 TCP 总速率;γ 为 RED 的平均丢弃率;Q 为 RED 的平均队长;$L_{\max}$、$L_{\min}$、max_p 为 RED 的基本参数,分别为最大门限、最小门限和最大门限左邻域丢弃概率的倒数;p 为垃圾桶中平均 TCP 流的个数与垃圾桶长度的比例.

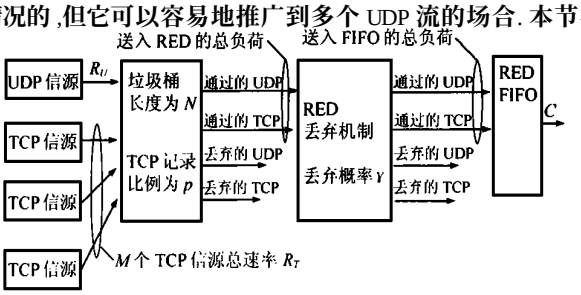


图 2 垃圾桶算法的惩罚机制

3.1 基本假定

本文采用了如下基本物理假设,它们有些不一定完全精确,但都基本能得到大多数人的认可.

(1) 垃圾桶中平均 TCP 流记录的个数比例等于系统丢弃 TCP 总速率在系统总丢弃速率中的比例:$p = (R_T - R_T^G)/(R_U - R_U^G + R_T - R_T^G)$,这里 R_U^G 为输出的 UDP 速率,R_T^G 为输出的 TCP 总速率.

(2) 一般来说对于不同的流特征,RED 的平均丢弃率略有不同[12],但在一般的讨论中仍可以认为 RED 对任何流的平均丢弃率均为 γ[14],则有:$R_U^G = (1 - \gamma)\, pR_U = \bar{\gamma} pR_U$ 和 $R_T^G = \bar{\gamma}(1 - p/M) R_U$.

(3) 送入 RED 缓存的平均吞吐量 $\bar{\gamma}p$ 与缓存队长满足关系(其中 η 为某个参数):$\bar{\gamma}p/(1 - \bar{\gamma}p) = \eta Q$,这里 ρ 为送入 RED 的负荷;且当给定 RED 参数后,RED 平均队长仅是 RED 的平均丢弃率的函数:$Q = Q(\gamma)$,这个假定的讨论详见 3.2 节.

(4) 单个 TCP 流的吞吐量为[3,11]:$R_{tcp} = vB/RTT_{tcp}\sqrt{p_{tcp}}$,其中 RTT_{tcp}、p_{tcp} 和 B 分别为该 TCP 的往返时延、丢失率和分组长度.

3.2 RED 的平均队长和平均丢弃率间的关系

一般排队理论认为平均队列长度和系统丢失率均可以由排队法则和输入负荷量唯一确定,因此原则上说可以从系统丢失率计算出队列的平均长度,但对于有限缓存的 RED 系统,这个关系将非常复杂,目前尚无实用的理论结果[12]. 这里讨论的模型中又需要一个较为简洁便于处理的关于 RED 系统丢失与平均队列长度的关系,因此特别劈出一节来专门讨论这个问题. 据我们所知目前尚无这方面的理论结果报道,因此这节的内容和结论不仅仅解决了本文所讨论的问题,而且有它独立存在的意义,所以讨论也较为详细.

RED 可分为基本 RED[12] 和防连续丢失 RED[13,14] 两种,用仿真软件 ns 的语言,防连续丢失 RED 又可分为 wait = true 和 wait = false 两类(ns 仿真软件的默认值为 wait = true),大多数理论探讨中使用基本 RED 模型,而实际应用的多为仿连续丢失 RED 模型. RED 的瞬时丢弃率可以看成是一个由该时刻 RED 估计的平均队长 L,以及连续未丢弃计数 $Count$ 所确定的随机变量 $\Gamma(L, Count)$,对于不同类型的 RED 丢弃分组的法则示于图 3. 为了和基本参数中的 RED 平均瞬时队长 Q 区分,称 L 为估计队长,称它的平均值 l 为平均估计队长.

对于基本 RED 有:

$$E\{\Gamma^b(L)\} = \int_{L_{\min}}^{L_{\max}} \frac{L - L_{\min}}{(L_{\max} - L_{\min})\, max_p} p(dL) + Pr(L > L_{\max})$$

基本 RED 模型: $\Gamma^b(L) = \min\{1, \frac{(L - L_{min}^+)}{(L_{max} - L_{min})\, max_p}\}$

wait = false 模型: $\Gamma(L, Count) = \min\{1, \frac{\Gamma^b(L)}{1 - Count * \Gamma^b(L)}\}$

wait = true 模型: $\Gamma(L, Count) = \begin{cases} 0, & Count * \Gamma^b(L) < 1 \\ \frac{\Gamma^b(L)}{2 - Count * \Gamma^b(L)}, & 1 \le Count * \Gamma^b(L) < 2 \\ 1, & 2 \le Count * \Gamma^b(L) \end{cases}$

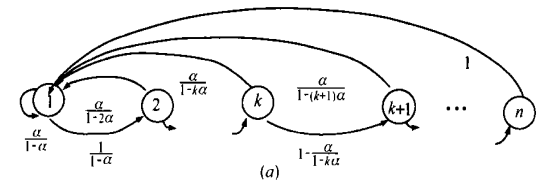
图 3　不同 RED 模型的丢弃分组法则

RED 的设计工作状态一般要求 $Pr(L > L_{max})$ 和 $Pr(L < L_{min})$ 均接近于 0,因此在多数应用中可假定基本 RED 的 l 与平均丢弃率的关系近似为:

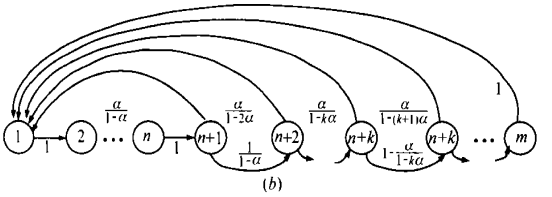
$$l = E\{L\} \approx (L_{max} - L_{min})\, max_p E\{\Gamma^b(L)\} + L_{min} = (L_{max} - L_{min})\, max_p \gamma + L_{min}$$

防连续丢失 RED 模型较为复杂,若进一步假定在一个连续不丢弃的时间段上,RED 的估计队长 L 不变,对于 wait = false 模型,在给定 L 时,令 $n = int^+\{[\Gamma^b(L)]^{-1}\}$,这里函数 $int^+\{x\}$ 是取大于等于 x 的最小整数,则连续未丢弃计数 $Count$ 构成如图 4(a) 所示的有限状态 $\{1, 2, \cdots, n\}$ 马尔可夫链,图中的 $\alpha = [\Gamma^b(L)]$,如果记 $p_i = Pr(Count = i)$ 是该链的稳态概率,则有: $p_k = (1 - k\alpha)\, p_1 / (1 - \alpha)$ 和 $p_1 = (1 - \alpha) / (1 - k\alpha)$. 对丢弃率求给定 L 下的条件期望为:

$$\Gamma(L) = \sum_{k=1}^{n} \frac{\alpha}{1 - k\alpha} p_k = \frac{2\alpha}{2 - (n+1)\alpha},$$



(a)



(b)

图 4　$count$ 状态转移图

将条件 $\alpha n \approx 1$ 和 $max_p \gg 1$ 时,将 $\Gamma^b(L) \ll 1$ 代入上式,有 $\Gamma(L) \approx 2\Gamma^b(L)$,因此得到 $\gamma = E\{\Gamma(L)\} \approx \frac{2(E\{L\} - L_{min})}{(L_{max} - L_{min})\, max_p}$, l 和平均丢弃率的关系可近似为 $l \approx 0.5(L_{max} - L_{min})\, max_p \gamma + L_{min}$.

对于 wait = true 模型,在给定 L 时,仍令 $\alpha = [\Gamma^b(L)]^{-1}$,但除定义 $n = int^+\{\alpha^{-1}\}$ 外,定义 $m = int^+\{2\alpha^{-1}\}$,则这时连续未丢弃计数 $Count$ 构成如图 4(b) 的有限状态 $\{1, 2, \cdots, m\}$ 马尔可夫链,为使分析简便,图中的状态转移概率中假定了 $\alpha n = 1$,稳态概率为:

$$p_i = \begin{cases} p_1, & k \le n \\ \frac{1 - (k - n)\alpha}{1 - \alpha} p_1, & k > n \end{cases};$$

其中, $p_1^{-1} = n + \sum_{k=1}^{m-n} \frac{1 - k\alpha}{1 - \alpha} = n + \frac{1}{1 - \alpha}\left(m - n - \frac{(m - n)(m - n + 1)\alpha}{2}\right)$,对丢弃率求给定 L 下的条件期望有:

$$\Gamma(L) = \sum_{k=1}^{m-n} \frac{\alpha}{1 - k\alpha} p_{n+k} = \sum_{k=1}^{m-n} \frac{\alpha}{1 - k\alpha} \times \frac{1 - k\alpha}{1 - \alpha} p_1 = \frac{\alpha(m - n)}{1 - \alpha} p_1,$$

将 $\alpha n \approx 1$ 和 $\alpha m \approx 2$ 带入有 $\Gamma(L) = \frac{2\Gamma^b(L)}{3(1 - \Gamma^b(L))} \approx \frac{2\Gamma^b(L)}{3}$, 因此 $\gamma = E\{\Gamma(L)\} \approx \frac{2(E\{L\} - L_{min})}{3(L_{max} - L_{min})\, max_p}$,所以 l 和平均丢弃率的关系又可表示为 $l \approx 1.5(L_{max} - L_{min})\, max_p \gamma + L_{min}$.

由于 RED 的平均对长估计是无偏的,平均瞬时队长和平均估计队长相等,另一方面 RED 的平均队长(估计的或所瞬时的)不会超过 L_{max},基于上面分析,可以得到 RED 平均瞬时队长和平均丢弃率的关系:

$$Q(\gamma) = \min\{\theta_{red}(L_{max} - L_{min})\, max_p \gamma + L_{min}, L_{max}\} \quad (1)$$

这里 θ_{red} 是一个与 RED 类型有关的修正参数,对于基本 RED、wait_false 和 wait_true 的 RED 分别为 1,0.5 和 1.5.

3.3　TCP 方程

根据基本假设 4 计算 TCP 流输入速率,必须能够估算 TCP 流的往返时延 RTT_{tcp} 和丢失概率 p_{tcp}. RTT_{tcp} 由两部分组成:本节点的排队时延和不包括本节点排队时延的其他时延 τ. 本节点排队时延可估计为 QB/C,所以 $RTT_{tcp} = \tau + QB/C$,在下面的分析中假定 $\tau \approx 0$,则 $RTT_{tcp} \approx QB/C$,在文中的模型中,TCP 的丢失概率 p_{tcp} 包括垃圾桶和 RED 机制的全部丢弃,所以 $p_{tcp} = p/M + (1 - p/M)\gamma = \gamma + \bar{\gamma} p/M$. 如果假定所有 M 个 TCP 均有相同的参数,则有

$$R_T/M = vBC/QB \sqrt{\gamma + \bar{\gamma} p/M}, \text{即}$$

$$t = vM/Q \sqrt{\gamma + \bar{\gamma} p/M} \quad (2)$$

3.4　基本方程组

基于基本假设可以得到 $1 - p = (u - \bar{\gamma} pu)/(u + t - \bar{\gamma} p)$,解得 $p = (t - \bar{\gamma} p)/(u + t - \bar{\gamma}\bar{p} + \bar{\gamma} u)$,由于本文的仿真中 TCP 的总归一化输入速率 t 会小于 1,所以不能使用简单的一阶近似 $\bar{\gamma}\bar{p} = 1$,这样会导致 $p \le 0$,转而采用较为复杂的排队近似 $\bar{\gamma}\bar{p} = \eta Q/(\eta Q + 1) = 1 - 1/(\eta Q(\gamma) + 1)$. 为了简化下面推导时的符号,记 $f(\gamma) = 1/(\eta Q(\gamma) + 1)$,将 $\bar{\gamma}\bar{p} = 1 - f(\gamma)$ 代入 p 的表达式有

$p = (t - 1 + f(\gamma))/(\gamma u + t - 1 + f(\gamma))$,再将上式和 $\rho = pu + t(1 - p/M)$ 代入 $\bar{\gamma}\bar{p} = 1 - f(\gamma)$ 得: $\bar{\gamma} t + \frac{\bar{\gamma}(u - t/M)(t - 1 + f(\gamma))}{\gamma u + t - 1 + f(\gamma)} = 1 - f(\gamma)$,化简后为:

$$ut\gamma^2 + \frac{M - 1}{M} t(t - 1 + f(\gamma))\gamma - (u - \frac{t}{M} + t - 1 + f(\gamma))(t - 1 + f(\gamma)) = 0$$

这样就得到了一组描述系统的方程组:

$$\begin{cases} p = \frac{t - 1 - f(\gamma)}{\gamma u + t - 1 + f(\gamma)} \\ ut\gamma^2 + \frac{M - 1}{M} t(t - 1 + f(\gamma))\gamma - (u - \frac{t}{M} + t - 1 + f(\gamma))(t - 1 + f(\gamma)) = 0 \\ t = \frac{vM}{Q(\gamma)\sqrt{\gamma + \frac{\bar{\gamma}(t - 1 + f(\gamma))}{M(\gamma u + t - 1 + f(\gamma))}}} \end{cases} \quad (3)$$

式中 $f(\gamma)$ 由所使用的 RED 类型和参数确定,满足 $(1-f(\gamma))/\eta f(\gamma)=Q(\gamma)=\min\{\theta_{red}(L_{\max}-L_{\min})\,max_p\gamma+L_{\min},L_{\max}\}$,$\theta_{red}$ 取 1、0.5、1.5 分别对应基本 RED、wait = false 和 wait = true 时的 RED.

3.5 对基本方程组的简化

上面给出的方程组虽然在原则上可解,但最终要导致一个非常高次的代数方程,计算相当复杂,所以有必要进行近似和化简. 由于 TCP 的参与,一般正常工作负荷时,RED 的丢弃率不会太大且队长一般也不会太短,因此可以据此对基本方程组做进一步简化,简化的条件是 $f(\gamma)\ll 1$ 和 $\gamma\ll 1$,于是得

$$Q(\gamma)=1/\eta\approx vM/\sqrt{h\gamma} \tag{4}$$

求解方程得到 γ 后,可用下面步骤直接得到其他所要求的各个结果.

(1) 从 $Q(\gamma)=\theta_{red}(L_{\max}-L_{min})\,max_p\gamma+L_{\min}$ 解得 $Q(\gamma)$,若 $Q(\gamma)>L_{\max}$ 则令 $Q(\gamma)=L_{\max}$ 并将其代入式(4);

(2) 从 $f(\gamma)=1/(\eta Q(\gamma)+1)$ 解得 $f(\gamma)$;

(3) 从 $f(\gamma)-\Delta_t\approx\dfrac{(M-1)f^2(\gamma)+Mu\gamma^2}{Mu-1}$ 解得

$$t=1-\Delta_t\approx 1-f(\gamma)+\frac{(M-1)f^2(\gamma)+Mu\gamma^2}{Mu-1};$$

(4) 从 $p=\dfrac{t-1+f(\gamma)}{\gamma u+t-1+f(\gamma)}$ 解得 p;

(5) $R_U^G=(1-\gamma)puC$; $R_T^G=(1-\gamma)(1-p/M)/C$.

4 通过仿真结果对理论模型的验证

本文对垃圾桶系统建立的理论模型是比较全面的,几乎涵盖了从任意的系统设置导出所有可能的其他系统参数,因此通过仿真对理论模型的验证是一个十分繁重的工作,由于篇幅的限制,不可能在这里反映所有的仿真结果,所有说明也只能限制在可懂的最小范围,几乎没有讨论. 总的来说,仿真结果与理论计算的差异不大,支持文中的理论模型. 本文的所有仿真工作采用 ns[13] 进行,仿真网络的结构如图 5. 图中 m 个 TCP 与 n 个 UDP 共享一个 $R1$ 与 $R2$ 之间的链路. $R1$ 与 $R2$ 之间的链路速率为 1Mbps, $R1$ 与 $R2$ 之间的链路的缓冲区管理策略就是带有垃圾桶的 RED,主机到路由器之间的链路速率均为 10Mbps,所有链路时延均设为 1ms,端到端时延主要由排队时延决定,TCP 的最大拥塞窗口设为 300,TCP 的应用为文件传输协议 FTP,UDP 以固定速率 rKbps 发送分组,分组长

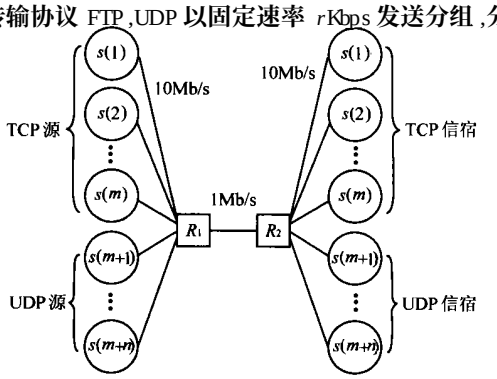


图 5 单链路网络

度设为 1000 字节. 显然,图 5 所示网络与图 2 的理论模型一致. 在下面所有的仿真工作中均做如下配置:FIFO 长度设为 300,垃圾桶长度为 20,垃圾桶的记忆概率取 0.5. 仿真时间取 240 秒,各个参数从 20 秒开始测量,$max_p=20$,并在计算理论值时采用了简化的公式来计算 γ,并取 $\eta=1$,$v=\sqrt{3/4}$.

4.1 不同 UDP 速率下的理论与仿真结果

仿真中将图 4 的 TCP 数设为 32,UDP 数设为 1. UDP 以 rMbps 发送. RED 的最小门限 $L_{\min}$ 设为 0,最大门限 $L_{\max}$ 设为 200. 对于 wait = true、wait = false 及基本 RED 三种 RED 类型,图 6 绘出了在不同 UDP 到达速率下的 γ、RED 平均队长、UDP 吞吐量和 TCP 吞吐量的理论值及实际仿真测量值. 从图中可以看出理论值与实际仿真值吻合得非常好.

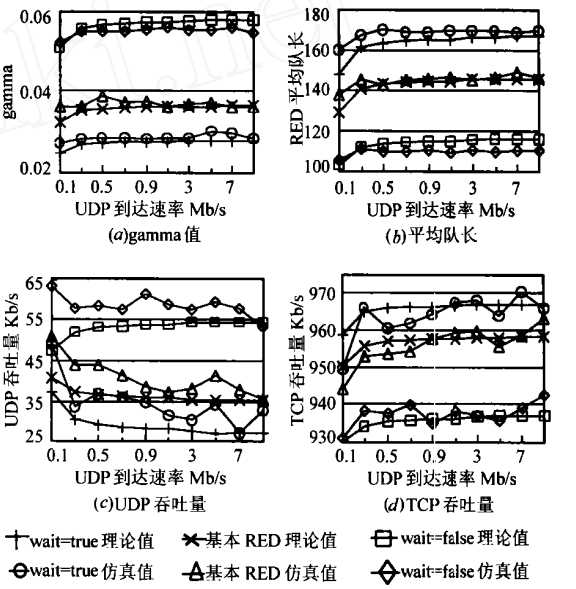


图 6 不同 UDP 速率下的仿真

4.2 不同 L_max 下的理论与仿真结果

仿真中将 UDP 速率固定为 1.5Mbps,$L_{\min}$ 为 0,TCP 数目取 32,变化 $L_{\max}$. 限于篇幅,本文只给出 wai = false 时的结果,如图 7 所示. 图 7 给出了在不同 L_max 下的 TCP、UDP 吞吐量,理论值与实际仿真值也非常接近.

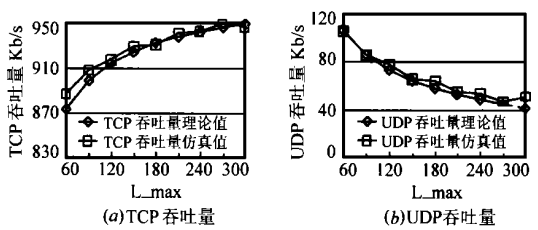


图 7 wait = false 时不同 L_max 的仿真

4.3 不同 L_min 时理论与仿真结果

变化 $L_{\min}$,并保持 $L_{\max}=2L_{\min}$,UDP 速率固定为 1.5Mbps,TCP 数目取 32. 图 8(a)、(b) 分别给出 wait = true 时的 gamma 和 UDP 吞吐量. 在图 8 中,直接理论值指直接使用 RED 的队长估计,而不做 $Q(\gamma)>L_{\max}$ 判断所得的结果,理论值指严格

按照 3.5 节中的办法所得的结果. 当 L_{min}较小时(30,60),直接使用 RED 的队长估计失效(估计出的队长分别为 99 和 134,大于 L_{max}的 60 和 120),硬做将导致 γ 估计失效,最终导致 UDP 吞吐量的估计误差较大. 因此这时要采用 3.5 节中的办法将 $Q(\gamma) = L_{max}$重新计算,所得的结果误差相对上一个方法要小一些. 由此可见,本文的模型只有在 RED 工作在正常的条件下才有效,当 RED 工作在极限条件时需要寻找新的模型.

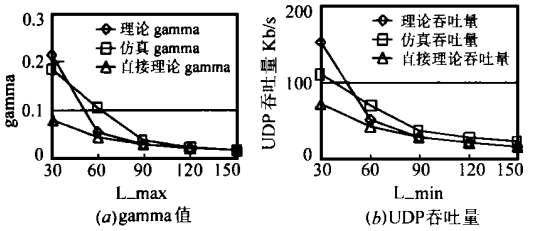


图 8　不同 L_min 的仿真

4.4　不同 TCP 数目下理论与仿真结果

本仿真变化 TCP 的数目. 固定 $L_{max} = 200$, $L_{min} = 0$, UDP 速率为 1.5Mbps. 图 9(a)、(b)分别给出了 wait = false 时的 gamma 和 UDP 吞吐量.

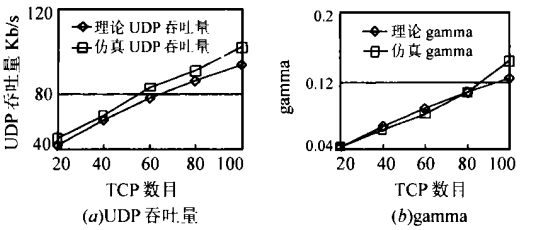


图 9　wait = false 时不同 TCP 数目的仿真

5　结论

本文分析了 wait = true、wait = false 及基本 RED 这三种情况下丢失概率与平均队长之间的关系,以此为基础分析了有一个非 TCP 友好流时垃圾桶算法的性能,最后给出了理论及仿真结果,仿真结果说明理论分析的有效性. 关于垃圾桶算法在多个 UDP 下的仿真结果在文[1]给出,垃圾桶算法在多个 UDP 下的建模工作正在进行.

参考文献:

[1] 赵永祥,陈常嘉. 一种对非 TCP 友好流的实用无状态惩罚算法[J]. 通信学报,2001,22(8):100 - 107.

[2] 赵永祥,陈常嘉. 纯垃圾桶算法:一种对非 TCP 友好流的实用无状态惩罚算法[J]. 铁道学报,2001,23(1):66 - 71.

[3] Floyd S, Fall K. Promoting the use of end-to-end congestion control in the internet [J]. IEEE/ACM Transactions on Networking, August 1999.

[4] Floyd S, Fall K. Router mechanisms to support end-to-end congestion control [J]. LBL Technical report, February 1997.

[5] D Lin, Morris. Dynamics of random early detection [A]. In Proc of ACM SIGCOMM [C], September 1997.

[6] B Suter, T V Lakshman, D Stiliadis, A choudhury. Design consideration for supporting TCP with per-flow queueing [A]. Proc. INFOCOM [C], March 1998.

[7] I stoics, S Shenker, H Zhang. Core-stateless. Fair queue: a scalable architeture to approximate fair bandwith allocation in high speed networks [A]. In Proceeding of ACM SIGCOMM [C], September 1998.

[8] Zhiruo Cao, Zheng Wang, Ellen Zegum. Rainbow fair queueing: fair bandwidth sharing without per-flow state [A]. Proc. INFOCOM [C], March 2000.

[9] Ott T, Lakshman T, Wong L. SRED: stabilized RED [A]. Proc. INFOCOM [C], March 1999.

[10] Rong Pan, Balaji Prabhakar, Konstantinos Psounis. CHOKe a stateless active queue management scheme for approximating fair bandwidth allocation [A]. Proc. INFOCOM [C], March 2000.

[11] Jitendra Padhye, Victoe Firoiu, Don Towsley, Jim Kurose. Modeling TCP throughtput: a simple model and its empirical validation [A]. In Proceeding of ACM SIGCOMM [C], September 1998.

[12] M May, T Bonald, J-C Bolot. Analytic evaluation of RED performance [A]. Proc. INFOCOM [C], March 2000.

[13] ns Network Simulator (version 2.1b5), 1999, URL: http://www-mash.cs.berkeley.edu/ns/

[14] S Floyd, V Jacobson. Random early detection gateways for congestion avoidance [A]. IEEE/ACM Trans. Networking [C], 1993, 1: 397 - 413.

作者简介:

赵永祥　男. 1970 年 7 月出生于云南昆明. 现于北方交通大学通信系攻读博士学位,主要研究方向:调度及缓存管理,拥塞控制. Email: zhaoyongxiang@263.net.

陈常嘉　男. 1949 年 10 月 27 日出生于澳门. 北方交通大学教授,博士生导师,主要研究领域:调度及缓存管理,拥塞控制,网络路由算法等.