基于区分服务模型拥塞控制的改进 RED 算法

Modified RED Algorithm According To DiffServ

董世强　(山东大学计算机学院、山大华天公司 250061)
万建成　赵 刚　(山东大学计算机学院 250061)

摘要:随着网络中语音和视频的应用普及,网络服务质量(QoS)是近年来网络通信中研究的一个热点问题,主动队列管理算法中的 RED 算法有其独特有效的处理拥塞的方法。作者通过对 RED 算法的分析,结合 DiffServ 体制,提出了对于 PHB 组间带宽占用的公平性和分等级服务质量方面的改进,改进后的算法能更好的适应当今网络的现状。

关键词:服务质量 RED 算法 区分服务 尾丢弃策略 主动队列管理 拥塞控制

1 网络拥塞产生的原因及现状

网络中的拥塞问题日益严重,我们来分析一下产生拥塞的具体原因。由于当今的互联网络中的绝大部分是面向无连接的,当大量的分组进入网络中的路由器时,在路由器的转发能力范围内,它将尽其所能根据其路由协议和路由表将分组转发到下一条路由。当超过路由器本身的转发能力时,多余的分组只能通过路由器的缓冲区做缓冲处理,等到路由器空闲时,将缓冲区中的分组再转发出去,这都是正常的情况。可是,当路由器的缓冲区已经用尽仍有新的分组到来时,路由器无法处理而只能将到来的数据报丢弃,而被丢弃的分组会在源端重传而导致情况进一步恶化。最坏的情况是,路由器乃至整个网络完全瘫痪,几乎没有分组能顺利到达目的地,这就是网络拥塞(Congestion)。

为了解决拥塞问题,人们研究出许多拥塞控制的算法。根据拥塞控制算法的使用位置,可以将拥塞控制算法分为两大类:链路算法(Link Algorithm)和源算法(Source Algorithm).在源算法方面,目前使用最广泛的是 TCP 协议中的拥塞控制算法。在链路算法方面,目前的研究主要集中在"主动队列管理"(Active Queue Management)算法方面。AQM 算法的一个代表是 RED 算法。

2 RED 算法[1][2]

2.1 RED 算法的基本原理

1993 年,S. Floyd 和 V. Jacobson 提出了随机早期检测(Random Early Detection:RED)算法。

RED 算法的基本思想如图 1 所示,采用"指数加权滑动平均"(EWMA)的方法计算队列的平均长度,采用该方法的路由器在每个接口上只维持一个队列,每接收到一个分组就重新计算队列的平均长度,并把队列的平均长度与预先设定好的最大阀值和最小阀值比较,若平均队列长度小于最小阀值则不丢弃分组;若大于或者等于最小阀值而小于最大阀值时,则以一定的概率 Pb 丢弃到达的分组,其中 Pb 是平均队列长度的线性函数;当平均队列长度大于最大阀值时,则丢弃到达的分组。RED 算法的请求丢弃策略如图 2 所示。

```
for each packet arrival
calculate the average queue size avg
 if minth≤avg ≤maxth
   calculate probability pa
   with probability pa :
   mark the arriving packet
else if maxth≤ avg
mark the arriving packet
```

图 1　RED 算法基本思想

2.2 RED 算法的现状和不足

RED 算法已经被证实有如下优点：第一、无论对面向连接的流量（TCP）还是面向非连接的实时流量（UDP），有效的降低了端到端延时；第二、对各种流量通过充分利用缓冲区已阻止大量的连续丢包；第三、与 Drop – tail 相比，消除了阵发流量的较高丢包率，部分消除了全局同步的现象。

同时，RED 也有如下缺点：首先，在无法知晓进入路由器的流量结构和数量时 RED 参数难于设置，同时联机时也难于调节。其次，某些研究也表明平均队列长度并不总是拥塞发生的晴雨表。在 RED 中，丢包仅仅受队列长度所限，而与流经的流量数量无关，这就导致了没有几个连接流量就独占整个带宽，甚至连接流量虽然多所占用带宽缺少，从而出现类似于 Drop – Tail 队列管理机制的问题：对于各连接流量丢包不具有公平性和各 TCP 连接流量的同步问题。

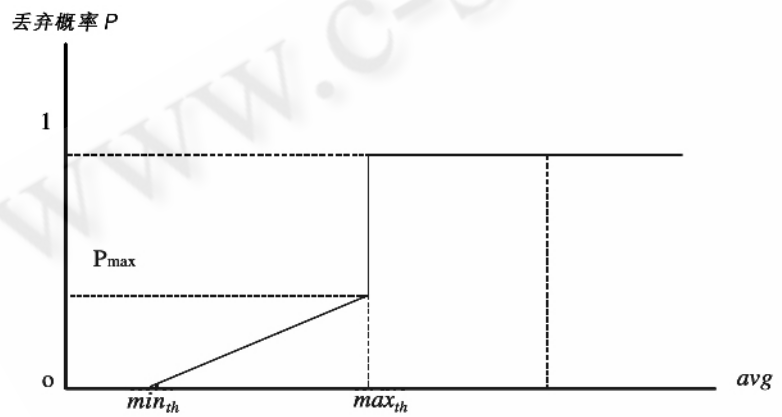

图 2　RED 算法的请求丢弃策略

2.3 已经存在的 RED 算法的改进技术[5][6]

RED 算法的变种有 2 类，传统的 RIO（RED with In/Out）使用与 RED 算法相同的机制，但它对于 IN 包和 OUT 包各配置一套参数。当流量符合制定策略的限制，它标记为 IN 包，若超过制定策略的限制，则标记为 OUT 包。RIO 有两套参数：一套是（min_in，max_in，max_in_p），用来计算 In 包的标记率；另一套是（min_out，max_out，max_out_p），用来计算 Out 包的标记率。可以通过选择这两套参数使得 RIO 增大 Out 包的丢包率，从而保护 In 包。一般来说，min_in > = max_out，max_out_p > max_in_p，并且 In 包的丢包率依赖于 In 包的平均队长 avg_in，而 Out 包的丢包率依赖于总的平均队长 avg_Out。这样，一旦 RIO 检测到早期拥塞，首先丢弃 Out 包，并且直到 Out 包全丢了，如果继续拥塞，才开始丢弃 In 包。

RIO – C 简单扩展为三种丢弃优先级或颜色，在 RIO – C 算法中，对于不同颜色的包计算不同的队列，在该方案中，一种颜色的包的平均队列尺寸可通过将它的平均队列加到较低优先级的平均队列上计算得到。例如，绿颜色包的平均尺寸仅使用绿颜色包；黄颜色包的平均尺寸将使用黄颜色和绿颜色包；红颜色包的平均尺寸使用红、黄、绿颜色包。该方案隐含着较高丢弃优先级的包队列长度必然高于较低丢弃优先级的包队列长度。

3 DiffServ 体系中的拥塞控制技术[3][4]

区分服务模型为每个用户提供一系列的服务类型，并通过 IETF 推荐的几类 PHB 组来实现，例如快速转发（EF）、确保转发（AF）以及尽力而为（BE）转发方式。其中 BE 对服务质量没有特别要求，可以采用原有的 RED 算法进行拥塞控制。EF 类用于提供端到端的低丢失率，低时延，低抖动，确保带宽的服务。EF 类理论上不会有拥塞发生。于是，只有 AF 类需要拥塞控制。

确保服务（Assured Service：AS）的初衷是：在网络拥塞的情况下仍能保证用户拥有一定量的预约带宽，使用户摆脱在单一尽量做好时无法把握自己实际占有带宽量的无奈窘况；着眼点是带宽与丢失率，不涉及延迟、抖动。服务原则是：无论是否拥塞，保证用户占有预约的最低限量的带宽；当网络负载较轻而有空闲资源时，用户也可以使用更多的带宽。则用户最终实际得到的带宽分为两部分，预定最小保证值以及与其他 AS 流或 BE 流竞争剩余资源获得的额外带宽。但与 PS 对带宽的严格承诺不同，AS 定位于统计型保证，这样可以提高资源利用率并降低价格，但也弱化了 AS 的质量保证，而更多的是一种优化服务（Better Service）。

4 在 DiffServ 模型下对 RED 算法的改进

4.1 DiffServ 模型的公平性讨论

在资源共享的环境中,一定会有各共享者之间的公平性问题。具体到 Diffserv,在区域边界微流将聚合为流聚集,之后在区域内 PHB 的处理对象是流聚集而非微流,因而同一流聚集内的各微流实质上在共享预留资源。Diffserv 中的公平性质就是属于同一流聚集的各微流能享受同等待遇。包括:

(1) 资源总量充足时各微流能充分享用其预约资源,达到预期性能。

(2) 有额外资源并允许竞争时各微流能平均分配或按比例分配额外资源。

(3) 资源总量不足时,各微流能按预约资源比例获得相应的降级服务。

公平性问题的研究目标是,改进服务实现机制,消除各种流特性差异对公平性的影响。

4.2 RED 算法改进的方法说明

首先,RED 算法没有分组优先级的概念,不能适应各种用户的不同要求;其次,当拥塞发生时也无法保证对不同 PHB 组间数据流的公平性。因此需要对算法作改进以适应区分服务的系统模型。

AF PHB 组族包含几个相互独立的 AF PHB 组,每组中 M 个 PHB 分属 M 个相对丢弃优先级。目前的定义 N=4, M=3。根据资源预留规格,各 DS 节点为每个 AF PHB 组预留一定量资源,以保证 AF 组对应的流在任何时候能获得预约最小带宽。DS 节点还应保证在同一 AF 组内,低丢弃优先级流聚集的丢失率应小于高丢弃优先级流;即拥塞时,节点应尽量避免低丢弃优先级流中的分组被丢弃,而更多的丢掉高丢弃优先级流中的分组。

目前有许多关于 RED 算法的改进研究,对于在 DiffServ 模型下,某些算法只是对优先级进行改进,例如三色 RED 算法能区分 3 个不同的服务优先级,而没有结合区分服务结构体系提出完整地解决方法。下面进行的改进是针对区分服务体系中的确保服务 AF 模型,在区别不同业务服务优先级的同时考虑到各个 PHB 分组占用网络带宽的公平性保证。

改进的 RED 算法如下:

(1) DiffServ 体制中的 AF 模型一共有 4 个 PHB 组,进入路由器的每个微流按照平均分配的原则,依次分配给 4 个组 PHB Group 1、PHB Group 2、PHB Group 3、PHB Group 4。

(2) 如果 4 个组在缓冲区中的包平均长度加起来未超过预先设定的最小值 MIN,则继续接受新来的包;否则,转入丢包处理过程。

(3) 在路由器中加入一个比较器,比较 4 个 PHB 组在缓冲区中占用的字节数,如果其中任何一组 PHB 占用的字节数已经超过总缓冲区的 25%,并且另外的 PHB 组仍然有新的分组到来需要使用缓冲区时,则不再给这组 PHB 分配缓冲区,直到其所占用的缓冲区低于 25%。

(4) 当缓冲区资源紧张时,如果一个 PHB 组甲占到 25% 以上的缓冲区而另一 PHB 组乙没有占到 25%,并且低于 25% 的分组仍然需要缓冲区时,则丢掉甲的分组来满足乙的要求,直至甲占用的缓冲区低于 25%。

(5) 每个 PHB 组内采用三色标记器 RED 控制算法,保证组内不同业务流享受到不同的优先级。

(6) 在同一个 PHB 组内,按优先级的由高到低的次序为:绿、黄、红。维持三个虚队列,queue_green, queue_yellow, queue_red。

(7) 计算平均队列长度时,采用如下方法:计算 queue_green 的平均队列 avg_queue_green 时,只计算所有 Green 分组的总长度;计算 queue_yellow 的平均队列长度时,令 avg_queue_yellow = queue_yellow + queue_green/2;计算 queue_red 的平均队列长度时,令 avg_queue_red = queue_red + queue_yellow/2 + queue_green/2;

(8) PHB 组内部的每个虚队列的分组丢弃策略仍然遵从 RED 算法,这样,在三种颜色队列长度相同的情况下,分组丢弃概率有如下关系:p_green < p_yellow < p_red. 如图 3 所示。

4.3 改进后的算法实现

```
For each packet arrival
{
  if ( buffer_phb[1] + buffer_phb[2] + buffer_phb[3]
 + buffer_phb[4]) < MIN_sum
    {
      if buffer_phb[i] >= 25% * buffer_sum &&
packet of group[j] <> null
        block[i];
      else
```

```
        accept[i];
    }
  else
    drop the packet;
}
```

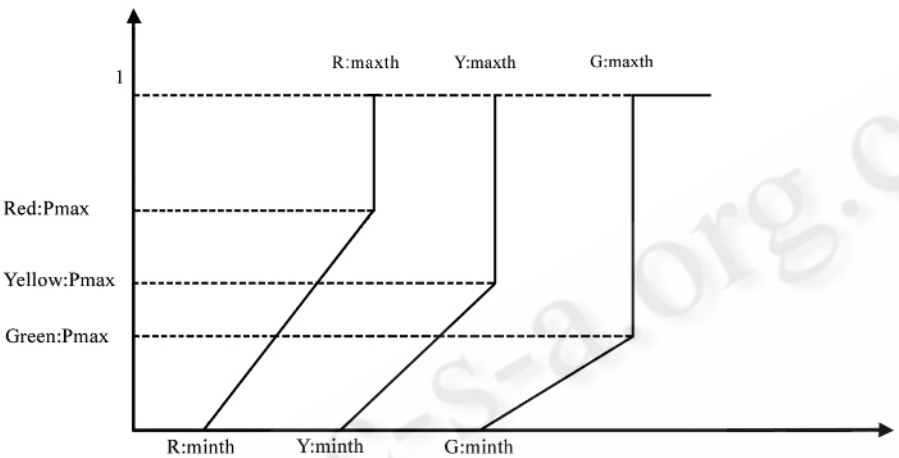

图 3　三色 RED 算法丢弃策略

```
Accept[i]
{
 if it is a Green packet
    {
    avg_queue_green = queue_green;
    if minth < avg_green < maxth
      calculate probability p_green;
      drop the packet with probability p_green;
    else if avg_green > maxth
      drop this packet;
    }
else
  if it is a Yellow packet
    {
    avg_queue_yellow = queue_yellow + queue_
green/2 ;
      if minth < avg_yellow < maxth
        calculate probability p_yellow;
      drop the packet with probability p_yellow;
  else if avg_yellow > maxth
      drop this packet;
      }
else
  {
  avg_queue_red = queue_red + queue_yellow/2 +
queue_green/2;
  if minth < avg_red < maxth
        calculate probability p_red;
        drop the packet with probabili-
ty p_red;
    else if avg_red > maxth
      drop this packet;
    }
```

4.4 改进后的算法效果分析说明

上述算法，能够在保证组内优先级的同时，保证组间的业务流享受到相对公平的待遇。如果资源空闲时，所有的组均能得到充足的缓冲区；如果资源紧张并且 4 个组都有数据需要缓冲时，平均分配缓冲区；如果仅仅其中一个或两个 PHB 组有数据时，则它们可以使用所有的空闲缓冲区，也不会造成缓冲区的浪费。改进后的算法弥补了现有算法在组间公平性方面的不足。

对于组内分成 3 个优先级，在文献[6]的基础上，对于平均队列长度的计算方法作了改进。原有的三色算法在计算优先级队列的平均长度时，用队列的长度加上较低丢弃优先级队列的长度。这样能保证在任何时候高丢弃优先级颜色的队列平均长度比低丢弃优先级颜色的队列平均长度要长，从而保证其绝对高的丢弃概率。然而，这对于黄、红两种颜色来说，显得尤其的不公平，即优先级差别悬殊，并不利于实际中的业务应用。比如绿、黄、红三色分别代表实际网络中的语音、视频、和数据，如果只考虑到语音的绝对优先级而对视频和数据的优先级考虑很少，则不能满足网络中这类用户的基本需求。因此，改进后的算法在计算平均队列长度时，如步骤(7)中所示，保证了三种颜色在队列长度相同的情况下，p_green < p_yellow < p_red，并且丢弃优先级平滑上升，真正满足实际网络中的业务要求。

(下转第 37 页)

（上接第 33 页）

5 结束语

网络的拥塞控制是一项系统工程，需要网络中的源节点和目的节点的共同配合，需要中间节点路由器和交换机的算法改进，本文所讨论的问题只是针对于中间节点路由器得缓冲区管理算法，结合主动网络技术，在网络即将拥塞时通过临近节点及时做发送速率的调整极其重要，而不能仅仅依靠源发送节点的速率控制，这样可以快速有效消除 RED 算法的全局同步问题。下一步要做的工作的一部分就是如何将中间节点的拥塞控制通相邻节点联系互动，同时兼顾源节点的拥塞反馈，有效的控制网络中的流量，使网络资源的利用达到理想水平，满足众多的网络应用要求。

参考文献

1 Sally Floyd and Van Jacobson, Random Early Detection Gateways for Congestion Avoidance, IEEE/ACM Transactions on Networking, 1999, 8:458 – 472.

2 Bonald T, May M and Bolot J. Analytic evaluation of RED performance [c] IEEE Incom'99, SanFrancisco, CA, November 1999 47 – 54.

3 Elloumi O, De Cnodder S, and Pauwels K. Usefulness of Three Drop Prededences in Assured , Forwarding Service. Internet Draft, draft – elloumi – diffserv – threevstwo – 00. txt, July 1999.

4 Icon Stoica ,Hui Zhang. Providing Guaranteed Services without Per Flow Management. In , Proceedings of ACM SIGCOMM, 1999.

5 D Clark, W Fang. Exiplicit Allocation of Best Effort Packet Delivery Service[J],IEEE/ACM , Transactions on Networking, 1998; 4(6).

6 王止戈、郑枫、杨珉、高传善，一种区分服务模型的分组丢弃算法，计算机工程与应用，175 – 177，2003 年 4 月。