广西科学院学报
Journal of Guangxi Academy of Sciences
2010,26(4):426～428
Vol. 26, No. 4 November 2010

基于 NS2 软件验证随机早期检测算法 RED 的优越性*

Validation of Random Early Detection Algorithm's Superiority Based on NS2

孙栋栋,王玉斌,马争先,张 净
SUN Dong-dong, WANG Yu-bin, MA Zheng-xian, ZHANG Jing

(桂林电子科技大学计算机科学与工程学院,广西桂林 541004)
(School of Computer Science and Engieering, Guilin University of Electronic Technology, Guilin, Guangxi, 541004, China)

摘要:在构造 RED 算法的网络拓扑模型和建立该模型的有限状态机的基础上,用目前主流的网络模拟软件 NS2 进行仿真,得到当前队列大小、平均队列大小和 TCP 数据源窗口大小的仿真图形,通过对仿真图形的比较和分析,说明 RED 算法的高效性和优越性。

关键词:队列管理 RED 形式化 有限状态机 NS2
中图法分类号:TP393 **文献标识码**:A **文章编号**:1002-7378(2010)04-0426-03

Abstract: Based on constructing the network topology model and establishing the finite state machine of the model, RED algorithm is simulated the current mainstream NS2 network simulation software. The simulation graphs of the current queue size, average queue size and TCP data source window size are obtained. The comparison and analysis of the simulation graphs shows the efficiency and superiority of RED algorithm.

Key words: queue management, RED, formal, finite state machine, NS2

以网络协议 IP(Internet Protocol)为基础的 Internet 呈爆炸式增长,使 Internet 的流量急剧增加,由此引发的网络拥塞已经成为制约网络发展和应用的关键因素。由 Jacobson[1] 提出的 TCP 流量控制策略,奠定了端到端流量控制的基础。随着网络规模越来越大,仅仅依靠 TCP 拥塞控制机制来提高网络的服务质量是远远不够的,网络中的中间节点也必须加入到网络拥塞的控制当中来,于是出现了许多基于路由器的拥塞控制策略,通常也称之为队列管理机制。

当前队列管理机制可以分为被动式队列管理和主动式队列管理这两大类。去尾(DropTail)算法是一种被动队列管理算法,它不需要选择丢弃的分组,而是在系统没有空闲缓冲资源时丢弃新到达的分组,它的最大优点是简单。而最著名、应用最广的随机早期检测算法 RED (Random Early Detection)属于主动式队列管理。RED 队列管理算法通过监控路由器输出端口队列的平均长度来探测拥塞,一旦发现接近拥塞,就随机选择连接来发出拥塞通知,使它们在队列溢出导致丢包之前减小发送窗口,降低数据发送速率,从而缓解网络拥塞[2]。目前,RED 算法已经被广泛使用在网络队列管理中来提高系统的综合性能[3]。RED 算法与 DropTail 算法相比有两个好处:一是队列缓冲总是预留了一定的缓冲空间以便更好的处理突发事件;二是保持较短队列长度,这样可以更好地支持实时应用。RED 算法还有很多变种和改进版本,如 ARED[4]、SRED、FRED 等等。

在验证 RED 算法的优越性时,文献[2]从延迟和公平性这两个属性进行分析研究,文献[5]从丢包率和稳定性等属性进行分析研究。而本文是在构造

收稿日期:2010-08-25
修回日期:2010-10-09

作者简介:孙栋栋(1987-),男,硕士研究生,主要从事计算机网络应用研究。

*广西自然科学基金项目(桂科自 0991242)资助。

RED 算法的网络拓扑模型和建立该模型的有限状态机的基础上,用目前主流的网络模拟软件 NS2 进行仿真,得到当前队列大小、平均队列大小和 TCP 数据源窗口大小的仿真图形,通过对仿真图形的比较和分析,说明 RED 算法的高效性和优越性。

1 网络拓扑模型及其有限状态机构建

为了用网络模拟软件 NS2(Network Simulator,version2)[2]动态演示队列管理机制 RED 的模拟过程,采用单瓶颈链路拓扑结构(图 1)[5]进行模拟仿真。其中,瓶颈链路是 n2 与 n3 之间的链路,S(1)~S(2)是 3 个数据源发生的节点,n3 为目的节点,本文主要考察的是 n2 和 n3 之间链路上的队列。

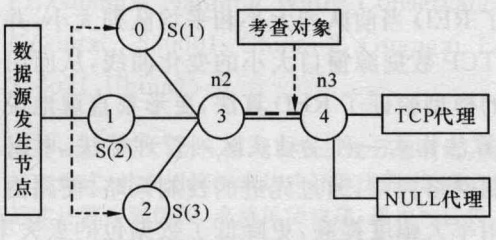

图 1 队列管理模拟的网络拓扑模型

1.1 网络协议的形式化工具

有限状态机 FSM(Finite State Machine)是一个五元组[6] $M=(Q,\sum,\delta,q_0,F)$ 其中

(1)$Q=\{q_0,q_1,\cdots,q_n\}$ 是有限状态集合,在任一确定的时刻,*FSM* 只能处于一个确定的状态 q_i;

(2)$\sum=\{\delta_1,\delta_2,\cdots,\delta_m\}$ 是有限输入字符集合,在任一确定的时刻,*FSM* 只能接收一个确定的输入 δ_j;

(3)$\delta:Q\times\sum\rightarrow Q$ 是状态转移函数,如果在某一确定的时刻,*FSM* 处于某一状态 $q_i\in Q$,并接收一个输入字符 $\delta_j\in\sum$,那么下一时刻将处于一个确定的状态 $q'=\delta(q_i,\delta_j)\in Q$;

(4)$q_0\in Q$ 是初始状态,FSM 由此态开始接收输入;

(5)$F\subset Q$ 是终结状态集合,FSM 在到达终态后不接收输入。

1.2 RED 网络拓扑的形式化定义

用有限状态机对具体的 RED 的网络拓扑结构图 2 进行形式化描述,使其更具体、更精确。

该网络拓扑结构的有限状态机 $M=(\{q_0,q_1,q_2,q_3,q_4\},\{0,1\},\delta,q_3,\{q_4\})$,其中 q_0,q_1,q_2 是 3 个数据源发生节点的形式化定义;q_3 和 q_4 链路结点的形式化定义;0 代表没有输入,1 代表有输入;δ 是状态转移函数;q_3 是接收数据的初态,q_4 是终态。状态转移函数 δ 的定义如下:$\delta(q_0,1)=q_3,\delta(q_0,0)=q_0;\delta(q_1,1)=q_3,\delta(q_1,0)=q_1;\delta(q_2,1)=q_3,\delta(q_2,0)=q_2;\delta(q_3,1)=q_4,\delta(q_3,0)=q_3;\delta(q_4,1)=q_4,\delta(q_4,0)=q_4$。

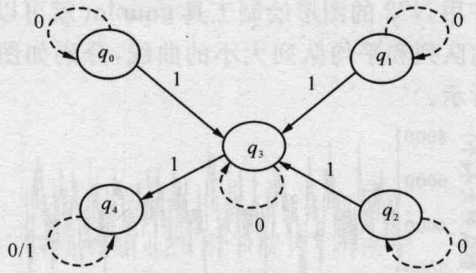

图 2 网络拓扑结构的有限状态机

2 NS2 仿真及结果分析

根据该模型和有限状态机的状态转移函数,编写 Tcl 脚本语言,在网络模拟软件 NS2 中进行模拟,得到如图 3 所示队列管理算法 RED 的 Nam 动画演示。

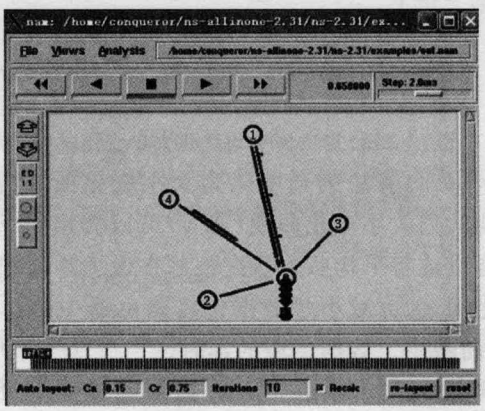

图 3 队列管理算法 RED 的 Nam 动画演示

从模拟结果的 Trace 文件中可绘制出 RED 队列即时和平均队列大小变化曲线,也可绘制出 RED 队列管理中 TCP 数据源的窗口大小变化曲线。

由于 RED 的 Trace 结果有些不同,所以需要对 red-queue. tr 文件中的数据进行预处理,即按照 Trace 文件中不同标记将平均队列大小和当前队列大小的记录区分开来。运行如下 grep 命令:

```
$ grep "a" red-queue. tr>ave. tr
$ grep "Q" red-queue. tr>cur. tr
```

运行命令后在同一文件夹下就会生成 ave. tr 和 cur. tr 这两个文件。

再利用 GNU 开发的 gawk 语言从 ave. tr 和 cur. tr 这两个文件中提取出我们需要的数据,编写一段程序代码放在 ave. awk 和 cur. awk 这两个文件中,再

运行如下 gawk 命令：

$ gawk-f ave. awk ave. tr＞ave

$ gawk-f cur. awk cur. tr＞cur

运行命令后在同一文件夹下就会生成只含我们需要数据的 ave 和 cur 这两个文件，然后针对这两个文件用 NS2 的图形绘制工具 gnuplot 就可以绘制出当前队列和平均队列大小的曲线，分别如图 4 和图 5 所示。

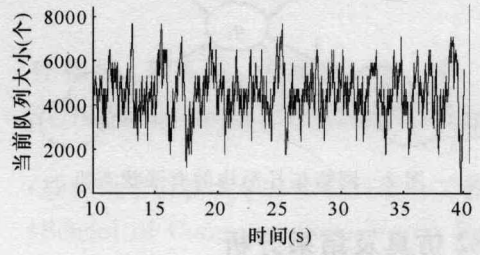

图 4 RED 算法的当前队列大小变化曲线

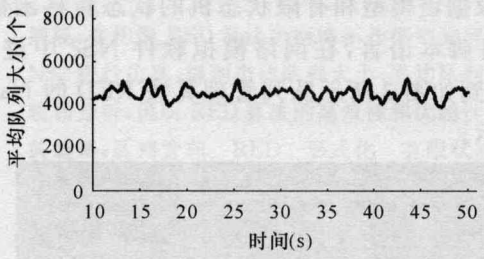

图 5 RED 算法的平均队列大小变化曲线

从 RED 算法的当前队列大小变化曲线(图 4)和平均队列大小变化曲线(图 5)可以看出，RED 算法能有效地将平均队列长度控制在最大阈值和最小阈值之间，大大减小了数据包在路由器中的排队时间，从而减少了数据传输的延时。

同样可以绘制出 TCP 数据源窗口变化曲线如图 6 所示。从图 6 分析得知，采用 RED 算法时，队列大小平均在 6 左右，也就是说，没有出现队列满的情况，而几个不同的 TCP 数据源的数据丢包时间也没有同步特性，即其丢包并不是等到队列满以后，而是根据 RED 计算出的阈值来进行判定。

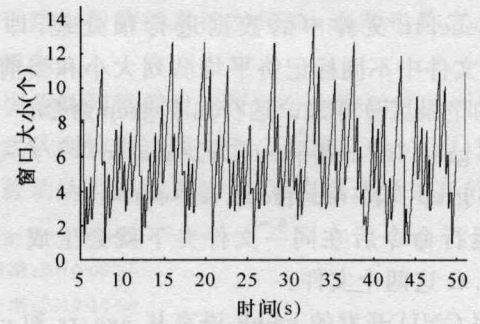

图 6 TCP 数据源的窗口大小变化曲线

3 结束语

传统的端到端拥塞控制机制很难满足现有网络服务的需求，于是出现了中间节点参与控制。中间节点的功能增强，有效地解决了网络拥塞问题[7,8]。目前，主动队列管理算法是重要的拥塞控制增强机制，其作为控制网络拥塞的主要实现方法越来越受到人们的关注。

在综合考虑节点链路接入和仿真模拟简化的基础上，本文一是构建网络模型时选取了 5 个节点，并利用有限状态机将其精确、全面地描述出来，再根据该网络拓扑模型和有限状态机的状态转移函数，编写 Tcl 脚本语言；二是用 NS2 进行模拟仿真，着重比较了 RED 当前队列大小和平均队列大小，并且绘制出 TCP 数据源窗口大小的变化曲线，从而用一种简单的模型验证了 RED 算法，更形象直观地说明了 RED 算法作为一种主动式队列管理算法，可以有效地抑制网络拥塞，通过先进的控制策略，使路由队列的利用率大幅度提高，更降低了数据包的丢失率，从而提高了网络的传输性能。

参考文献：

[1] Jacobson V, Karels M. Congestion avoidance and control[M]. Proc Acm Sigcomm, Stanford, CA, USA, 314-329.

[2] 城新年. 基于 NS2 的路由器算法 Droptail 和 RED 的分析与比较[J]. 计算机工程与科学，2007，29(6)：24-25.

[3] 任丰原，林闯，王福豹. RED 算法的稳定性：基于非线性控制理论的分析[J]. 计算机学报，2002，25(12)：1302-1307.

[4] Floyd S, Gummadi R, Shenker S. Adaptive RED: an algorithm for increasing the robustness of RED's active queue management [EB/OL]. Under Submission, 2005.

[5] 王建新，荣亮，肖雪峰. 几种主动队列管理算法的仿真及性能评估[J]. 计算机工程，2007，33(3)：128-130.

[6] 古天龙. 软件开发的形式化方法[M]. 北京：高等教育出版社，2003：24-26.

[7] REN Feng-yuan, LIN Chuang, LIU Wei-dong. Congestion control in IP network[J]. Chinese Journal of Computers, 2003, 26(9): 1025-1034.

[8] 蔡安妮，孙景鳌. 多媒体通信技术基础[M]. 北京：电子工业出版社，2000.

(责任编辑：邓大玉)