

# 一种 TCP 博弈模型的 Nash 均衡存在性分析与仿真

冯 坚,王书田,林日光

(海南广播电视大学,海南 海口 570105)

**摘 要:**在当今的 Internet 中,远程教学、大规模传输等“不响应拥塞”应用与日剧增,使得端系统有动机更改拥塞控制方案以满足自己的需要,这加重了 Internet 拥塞。一般希望通过研究用户非合作博弈行为的 Nash 均衡来规范自私的端用户行为。阐述了 TCP 拥塞博弈模型,主体为采取 TCP 协议的端用户,策略为 TCP 端用户的慢启动拥塞窗口递增参数。通过数学分析方法论证了网络方对 TCP 流没有额外的处罚时,TCP 博弈存在 Nash 均衡。同时,通过 Ns2 仿真工具验证了当路由器采用 Drop Tail 队列管理算法,端节点采取 TCP Tahoe 和 TCP Reno 时,TCP 博弈存在 Nash 均衡。这意味着 TCP 算法对 Internet 的稳定起着重要作用。

**关键词:**TCP 协议;拥塞;博弈论;Nash 均衡

**中图分类号:**TP391

**文献标识码:**A

**文章编号:**1673-629X(2009)11-0076-04

## Analysis and Simulation to Existence of Nash Equilibria in a TCP Game

FENG Jian, WANG Shu-tian, LIN Ri-guang

(Hainan Radio and TV University, Haikou 570105, China)

**Abstract:**In current Internet, the uncongestion reaction applications such as distance education, large scale transmissions are increasing, so the users have incentives to modify its congestion strategy to satisfy their own needs, this worsen Internet congestion situation. In order to control the selfish behaviors, Nash equilibrium theory of uncooperative game is studied. In this paper, a TCP game model is presented, the player is the network user that implemented by TCP algorithm, strategy is the slow start congestion window parameter. Through mathematics analysis, conclude that there is a Nash equilibrium in a TCP game when no punishments are given to the TCP flows. At the same time, through Ns2 simulation, validate that there is a Nash equilibrium in a TCP game when the router used Drop Tail and end user implemented by Tahoe or Reno. This means that TCP congestion algorithm play an important role in Internet stability.

**Key words:**TCP;congestion;game theory; Nash equilibria

## 0 引 言

TCP 中使用的拥塞控制算法已经成为保证互联网稳定性的重要因素<sup>[1]</sup>,它要求所有的端用户都是合作的,即使用相同的 TCP 算法来控制流量,由于 Internet 不能控制用户的流量,因而这种合作是非强制性的。随着网络业务的扩展,特别是大规模数据传输、远程教学等业务在 Internet 中的广泛应用,端系统有动机更改速率控制方案以满足网络应用的需要,甚至可以从软件上按自己的意图随意更改拥塞控制方案,这样引出了一个问题:为获得最大的收益(如占用可能多的带宽),自私的 TCP 端用户可能采取不同的拥塞策

略,这些 TCP 端用户能否收敛于一点。

博弈论的 Nash 均衡思想为该研究提供了坚实的数学基础。Nash 均衡是指对应的策略组合,使得任意一方单独改变自己的策略都无法提高自身的收益<sup>[2]</sup>。在拥塞博弈局势中,一个行动向量只有构成 Nash 均衡,才是在现实中可能稳定存在的结果<sup>[3]</sup>。文中探讨如果端用户是自私的,即端用户可以随意改变 TCP 慢启动拥塞参数来最大化自己的收益,TCP 博弈是否存在 Nash 均衡。

## 1 TCP 博弈模型

TCP 是基于闭环控制的算法,由四个核心子算法组成,即慢启动(slow start)、拥塞避免(congestion avoidance)、快速重传(fast retransmit)和快速恢复(fast recovery),经过十多年的发展,目前 TCP 协议主要有四个版本:TCP Tahoe、TCP Reno、TCP NewReno 和

收稿日期:2009-03-12;修回日期:2009-06-30

基金项目:海南省重点科研计划项目(HJ2007167)

作者简介:冯 坚(1974-),男,海南琼海人,硕士,讲师,研究方向为网络协议。

TCP SACK<sup>[4,5]</sup>。依据博弈论模型的构成理论,文中 TCP 博弈模型的构成有局中人、策略和收益函数。

### 1) 局中人。

在 TCP 博弈中,局中人指采用类 TCP 协议的端用户,他们是一组基于闭环控制的协议,当探测到拥塞时,对拥塞作出响应。文中仅讨论端用户采取 Tahoe 和 Reno 的情形。

### 2) 策略。

假定 TCP 端用户是自私的,其目标是最大化自己的有效传输(GoodPut),为达到这一目标,每一端用户可以自由修改拥塞参数。在文中,假定所有的 TCP 流的  $\beta=0.5$  和  $\gamma=1/\text{cwnd}$ ,且在拥塞博弈中其值是固定的,所有 TCP 流可自由选择慢启动拥塞窗口递增参数  $\alpha$ ,以获得最大收益。

采取慢启动拥塞窗口递增参数作为博弈策略基于以下原因:慢启动算法是 TCP 拥塞算法的第一个阶段,自私的端用户会在博弈的初始阶段通过加大慢启动拥塞窗口递增参数以获得更多的带宽,且修改慢启动窗口函数能给用户的发送数据量带来几何级数的增长,更符合端用户的自私本性,因此自私的端用户倾向于加大慢启动拥塞窗口递增参数来最大化自己的收益。

### 3) 收益。

在 TCP 博弈中,收益指用户  $i$  在时间  $t$  内传输的实际字节数。

为方便研究,下面给出 TCP 博弈模型的一些限定条件。

(1)在 TCP 拥塞博中,仅讨论单瓶颈链路的情形,见图 1。所有的 TCP 流将通过相同的瓶颈链路。

(2)为分析 TCP 本身算法对博弈结果的影响,在 TCP 博弈中,博弈规则对流没有额外的处罚。TCP 博弈中,博弈规则是路由器中所采用的队列管理算法,Drop Tail 队列以相同的丢包概率处理分组,对任意 TCP 流没有额外的处罚,文中讨论队列管理算法为 Drop Tail 的情形。

(3)仅讨论对称博弈<sup>[6]</sup>的情形,即所有的 TCP 流采用相同的拥塞算法,具有相同的往返时间(RTT)。

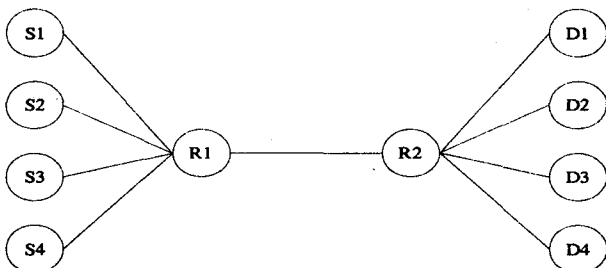


图 1 TCP 博弈瓶颈链路

## 2 TCP 博弈的 Nash 均衡

一般地,给定有  $n$  个 TCP 流,  $F_1, \dots, F_n$ , 流  $F_i$  的发送参数为  $\alpha_i$ , 支付为  $G_i$ , 下面给出拥塞控制博弈 Nash 均衡定义。

定义 1 在拥塞博弈  $G = [\{\alpha_i\}, \{G_i(\cdot)\}]$  中,如果任一用户的慢启动参数  $\alpha_i^*$ , 都是在给定其余用户行动  $\alpha_{-i}^* = (\alpha_1^*, \dots, \alpha_{i-1}^*, \alpha_{i+1}^*, \dots, \alpha_n^*)$  的情况下用户  $i$  的最佳策略,即:

$$G_i(\alpha_i^*, \alpha_{-i}^*) \geq G_i(\alpha_i, \alpha_{-i}^*), i \in [1, n]$$

则称  $\alpha^*$  构成  $G$  的一个 Nash 均衡。

Nash 均衡并不是在所有博弈中都存在的,文献 [6] 对 Nash 均衡的存在性做了深入的探讨,给出了 Nash 均衡存在性定理。

定理 1 在  $n$  人策略型博弈中,如果每个局中人的纯策略空间  $X_i$  是欧氏空间上的非空有界闭凸集,支付函数  $G_i(x_i)$  连续且对  $x_i$  是拟凹的,那么这一博弈中存在一个纯策略 Nash 均衡(Debreu, Glicksberg, Fan, 1952)。

定理 1 给出存在纯策略 Nash 均衡的充分条件(并非必要条件),其中拟凹性是相当严格的条件。

## 3 TCP 博弈 Nash 均衡存在性分析

本节探讨,当端用户可自由选择慢启动拥塞窗口递增参数  $\alpha$  时, TCP 能否依靠自身的拥塞算法保证 Internet 的稳定, TCP 流博弈是否存在 Nash 均衡。首先分析 TCP 业务流的性质,进而证明 TCP 博弈 Nash 均衡的存在性。

### 3.1 TCP 流的性质分析

Shenker 从效用函数性质的角度,将 IP 业务分成两类:弹性业务和非弹性业务<sup>[7]</sup>,其区分方法已被计算机网络界普遍接受。

传统的 IP 网业务如 WWW、FTP、E-mail 都是弹性业务,它们对网络拥塞有很强的适应能力,对 QoS 的需求可以有很大的弹性变化,较少的资源分配不会导致应用决定性的失败。对于弹性业务,IP 网络一般采用 TCP。

在 TCP 博弈中,当网络对任意流没有额外的处罚时,用户  $i$  的收益函数表示为:  $G_i(\alpha_i, Y) = u_i(\alpha_i) - D_i(Y)$ 。  $Y$  为此时链路的资源使用率,  $Y = f(\alpha_1, \alpha_2, \dots, \alpha_n)$ ,  $u_i(\alpha_i)$  为用户  $i$  采取  $\alpha_i$  的策略时的效用,  $D_i(Y)$  为链路上产生拥塞时,对分组的时延和丢弃所造成的拥塞成本。

随着用户发送速率的增加,网络资源的使用率越高,因而有公式(1)。

$$\frac{\partial Y}{\partial \alpha_i} \geq 0, \frac{\partial^2 Y}{\partial \alpha_i^2} \leq 0 \quad (1)$$

用户在发送分组数据时,用户占用的带宽越大,获得的效用越高,但随着资源利用率的增高,分组的平均延迟急剧增大,拥塞成本会急剧提高,用户的效用会随之降低<sup>[7,8]</sup>,因而有公式(2)。

$$\frac{\partial u_i}{\partial \alpha_i} \geq 0, \frac{\partial^2 u_i}{\partial \alpha_i^2} \leq 0 \quad (2)$$

随着链路使用率的增高,分组的平均延迟急剧增大<sup>[9,10]</sup>,因而有公式(3)。

$$\frac{\partial D_i}{\partial Y} \geq 0, \frac{\partial^2 D_i}{\partial Y^2} \geq 0 \quad (3)$$

由公式(1)、(2)和(3)推导出公式(4)。

$$\frac{\partial^2 G_i}{\partial \alpha_i^2} = \frac{\partial^2 u_i}{\partial \alpha_i^2} - \frac{\partial^2 Y}{\partial \alpha_i^2} \frac{\partial^2 D_i}{\partial Y^2} < 0 \quad (4)$$

可见,用户收益  $G_i(\alpha_i)$  是  $\alpha_i$  的凹函数。

### 3.2 TCP 流的性质仿真

仿真实验也验证了用户收益  $G_i(\alpha_i)$  是  $\alpha_i$  的凹函数。仿真实验采用 Ns2<sup>[11]</sup> 平台,网络拓扑结构如图 2。节点 R1 和 R2 之间的链路队列为 Drop Tail,队列容量为 10,链路带宽为 1Mbps,延迟为 20ms,在 R1 之上建立 TCP(Tahoe)代理,其数据源为 ftp,当代理每收到一个 ACK,cwnd 值增加  $\alpha_i$ 。实验中,ftp 的运行时间为 100 秒,分别计算  $\alpha_i$  值为 1 至 14 时,在 sink 端收到的数据量  $G_i(\alpha_i)$ 。仿真数据见表 1,收益单位为 Mbyte,仿真数据模拟如图 3,由此可见用户收益  $G_i(\alpha_i)$  是  $\alpha_i$  的凹函数。

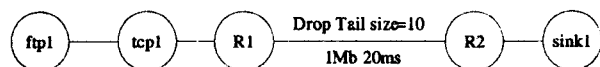


图 2 TCP 凹性仿真链路

表 1 TCP 凹性验证数据

G(4)	G(5)	G(6)	G(7)	G(8)	G(9)	G(10)
11.7	11.7	11.9	11.8	10.4	10.4	10.3

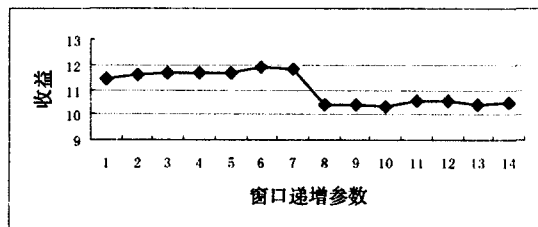


图 3 TCP 凹性验证

### 3.3 TCP 博弈 Nash 均衡存在性证明

定理 2 在 TCP 博弈中,网络方对参与者没有实施额外的处罚时,TCP 博弈存在 Nash 均衡。证明:网络方对参与者没有实施额外的处罚时,TCP 端用户的

收益可表示为  $G_i(\alpha_i, Y) = u_i(\alpha_i) - D_i(Y)$ 。

对收益表达式两边求导得最优化一阶条件  $\frac{\partial G_i}{\partial \alpha_i} = \frac{\partial u_i}{\partial \alpha_i} - \frac{\partial Y}{\partial \alpha_i} \frac{\partial D_i}{\partial Y} = 0, (i = 1, \dots, n)$ , 即  $\frac{\partial u_i}{\partial \alpha_i} = \frac{1}{k} \frac{\partial D_i}{\partial Y}, (i = 1, \dots, n)$ 。

这  $n$  个一阶条件定义了  $n$  个反应函数  $\alpha_i^* = \alpha_i(\alpha_1, \dots, \alpha_{i-1}, \alpha_{i+1}, \dots, \alpha_n), (i = 1, \dots, n)$ 。

若 Nash 均衡存在,则  $n$  个反应函数的交叉点就是 Nash 均衡。由公式(4)和定理 1 知这个交点(即 Nash 均衡)一定存在,证毕。

在当前 Internet 中,Drop Tail 在路由器中应用最为广泛,Tail Drop 是队尾丢弃算法。Internet 中传统的路由器队列按照“先进先出”的规则(FIFO)处理到来的数据报,由于队列长度总是有限的,因此当队列已满时,以后到达的数据报都将被丢弃,这就叫做尾部丢弃策略(Tail Drop Policy),它是状态无关的,对流没有任何额外的处罚措施,因此依据定理 1 有以下推论。

推论 1 在 TCP 博弈中,路由器采取 Drop Tail 队列时,TCP 博弈存在 Nash 均衡。

以上定理表明:在 TCP 博弈中,增加慢启动拥塞窗口递增参数  $\alpha$  有正负两方面的效应。正的效应是这个分组对于用户的价值,负的效应是这个分组产生的拥塞使得其它被发送的分组价值下降,用户单方增加发送速率并不总能获得更大的收益,网络中存在 Nash 均衡。可见,TCP 可依靠自身的拥塞控制机制实现 Nash 均衡,这也是过去 Internet 保持稳定的重要原因。

## 4 TCP 博弈 Nash 均衡存在性仿真

本节利用 Ns2 仿真工具验证推论 1,当路由器采取 Drop Tail 队列,端用户可自由选择慢启动拥塞窗口递增参数  $\alpha$  时,TCP 流博弈存在 Nash 均衡。

### 4.1 TCP 博弈 Nash 均衡仿真方法

本实验采用 Ns2 网络仿真工具,采用图 4 的仿真链路。路由器主动队列管理算法为 Drop Tail。源端至 R1 的链路为 2M,10ms,R2 至目的端的链路为 2M,10ms,R1 至 R2 的链路为 0.4M,20ms,源端的应用为 ftp。Nash 均衡存在性的验证算法见算法 1。

算法 1 给定  $n$  个流, $\alpha_i^j$  为在第  $j$  轮中流  $i$  的  $\alpha$  值, $G_i(\alpha)$  为流  $i$  采取  $\alpha$  策略时的收益, $\alpha^*$  为 Nash 均衡解。在每一轮中,每一流的运行时间为 100 秒。

Step1  $j = 1$ ,对于前  $n - 1$  个流,给定  $\alpha_i^j = 1, i \in [1, n - 1]$ ,求出使得  $G_n$  值最大的  $\alpha_n'$ ,另  $\alpha^{(j, \text{best})} = \alpha_n'$ ,如果  $\alpha^{(j, \text{best})} = 1$ ,则  $\alpha^* = 1$ ,运行 Step4;否则运行 Step2。

Step2  $j = j + 1, \alpha'_i = \alpha^{(j-1, \text{best})}, i \in [1, n - 1]$ , 求出使得  $G_n$  值最大的  $\alpha'_n$ , 另  $\alpha^{(j, \text{best})} = \alpha'_n$ , 如果  $\alpha^{(j, \text{best})} = \alpha^{(j-1, \text{best})}$ , 则  $\alpha^* = \alpha^{(j, \text{best})}$ , 运行 Step4, 否则运行 Step3。

Step3 运行 Step2。

Step4 退出。

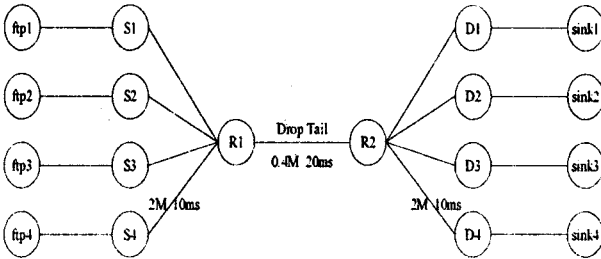


图 4 TCP 仿真链路

#### 4.2 仿真过程

在仿真中,第一步需修改 TCP 算法中慢启动阶段拥塞窗口递增参数,这一步的修改是属于 C++ 和 OTcl 编程的编译、配置层次。在 Ns2 中,需要重新设计 TCP 的拥塞算法,当端用户每收到一个 ACK 时,如果拥塞窗口 ( $\text{cwnd}$ ) < 拥塞阈值 ( $\text{ssthresh}$ ),拥塞窗口  $\text{cwnd}$  不是增加 1,而是增加  $\text{slow\_}$ ,因此需要在 Ns2 中修改  $\text{tcp.cc}$  程序,源代码如图 5 所示;第二步需在 OTcl 中创建网络的拓扑结构和仿真元素;第三步从仿真结果中统计各 TCP 流的收益。

```
void TcpAgent::opencwnd()
{
    double increment;
    if (cwnd < ssthresh_)
    {
        /*慢启动拥塞窗口递增参数 slow_*/
        cwnd_ += slow_;
    }
    else
    {
        /* 线性递增 */
        double f;
        switch (wnd_option_)
        {
            case 0:
                if (++count_ >= cwnd_)
                {
                    count_ = 0;
                    ++cwnd_;
                }
                break;
            case 1:
                .....
        }
    }
}
```

图 5 TCP 源代码

#### 4.3 仿真结果分析

1)端用户采用 Tahoe 时的仿真结果如表 2,收益单位为 Mbyte。当前三个 TCP 流的慢启动拥塞窗口递增参数  $\alpha_1 = \alpha_2 = \alpha_3 = 1$  时,第四个 TCP 流的慢启动拥塞窗口递增参数为  $\alpha_4 = 14$  时能获得最大的收益;当

前三个 TCP 流的慢启动拥塞窗口递增参数  $\alpha_1 = \alpha_2 = \alpha_3 = 14$  时,第四个 TCP 流的慢启动拥塞窗口递增参数为  $\alpha_4 = 8$  时能获得最大的收益;当前三个 TCP 流的慢启动拥塞窗口递增参数  $\alpha_1 = \alpha_2 = \alpha_3 = 8$  时,第四个 TCP 流的慢启动拥塞窗口递增参数为  $\alpha_4 = 9$  时能获得最大的收益;当前三个 TCP 流的慢启动拥塞窗口递增参数  $\alpha_1 = \alpha_2 = \alpha_3 = 9$  时,第四个 TCP 流的慢启动拥塞窗口递增参数为  $\alpha_4 = 9$  时能获得最大的收益。可见, Nash 均衡解为  $\alpha^* = 9$ 。

表 2 Tahoe 仿真结果

$\alpha_1 \alpha_2 \alpha_3$	$G_4(7)$	$G_4(8)$	$G_4(9)$	$G_4(10)$	$G_4(14)$	$G_4(15)$
1	1.6	1.6	1.7	0.6	2.1	0.4
14	0.7	2.2	1.8	1.3	2	1.9
8	1.2	1.4	2.2	1.5	2.1	2.3
9	1.2	1.2	2.2	1.4	1.7	1.8

2)端用户采用 Reno 时仿真结果如表 3,收益单位为 Mbyte。当前三个 TCP 流的慢启动拥塞窗口递增参数  $\alpha_1 = \alpha_2 = \alpha_3 = 8$  时,第四 TCP 流的慢启动拥塞窗口递增参数为  $\alpha_4 = 8$  时能获得最大的收益。可见, Nash 均衡解为  $\alpha^* = 8$ 。

表 3 Reno 仿真结果

$\alpha_1 \alpha_2 \alpha_3$	$G_4(3)$	$G_4(4)$	$G_4(5)$	$G_4(7)$	$G_4(8)$	$G_4(9)$	$G_4(10)$
1	1.6	1.7	0.3	1	0.5	0.3	0.8
4	1.2	1.2	1.2	1.3	1.3	1.6	1.2
9	1.3	1.3	1.4	1.4	1.7	1.2	1.5
8	1.9	1.5	1.4	0.9	2.1	1.1	1

#### 5 结束语

文中建立起策略为慢启动窗口递增参数的 TCP 博弈模型,通过分析 TCP 流的性质,认为用户收益是慢启动窗口递增参数的凹函数,进而得出结论:1)在 TCP 博弈中,网络方对参与者没有实施额外的处罚时, TCP 博弈存在 Nash 均衡。2)在 TCP 博弈中,主体为采用 Tahoe 或 Reno 算法的端用户,策略为慢启动拥塞窗口递增参数,规则为路由器采用的 Drop Tail 的队列管理算法时,网络中存在 Nash 均衡。以上结论表明,在 TCP 博弈中, TCP 能依靠自身的拥塞控制算法获得 Nash 均衡。

#### 参考文献:

- [1] Floyd S, Fall K. Promoting the Use of End-to-End Congestion Control in the Internet[J]. IEEE/ACM Transactions on Networking, 1999, 7(4):458-472.
- [2] 张 怡,刘高嵩,李章华,等.基于博弈论的 P2P 系统分析[J].计算机技术与发展,2007,17(8):26-27.
- [3] 冯 坚.状态无关主动队列管理算法博弈的 Nash 均衡

BSD 的 mbuf 类似,是一个真实数据包在内存中的序列化表示。标签列表是附加信息的容器,主要用于携带模拟专用信息如流标号、跨层信息等。公共接口函数提供了包头、数据和标签在 Packet 中的添加、移除和读取功能,以及 Packet 本身的分段重组和序列化反序列化功能。如果一个 Packet 被转移到仿真环境或者真实网络环境中,只需将标签字段剥落,然后把字节缓冲中的内容直接拷贝到真实的数据包中就可以了。

Packet 的字节缓冲基于 Linux 的 skb 或 BSD 的 mbuf 来实现,存储 Packet 在内存中的序列化内容。该序列化的表示与真实网络数据包按比特位匹配,意味着缓冲区的内容就是一个真实的网络数据包,不同的是,NS-3 中 Packet 的数据部分可以是任意位的 0 填充而没有真实的内存分配给它。基于此,Packet 的分段和重组变得非常自然而且支持与仿真环境和现实网络的移植。

Packet 的标签列表用于存储用户定制的、仅用于模拟研究的附加信息。与字节缓冲不同的是,标签列表是一个 TagData 类型数据结构的链表。NS-3 允许每个 Packet 包含一个标签列表,每个标签列表可以存储任意规模的由用户提供的数据结构集合。虽然标签列表的规模可以是任意的,但是其中的每个标签只能有一个实例,由其类型唯一标识,且每个标签最大只能有 16 字节。

#### 4 结束语

作为开源网络模拟器 NS-2 的替代者,NS-3 没有采用 NS-2 的架构,而是进行了全新的设计和实现,在实用性、兼容性、易操作性、可扩展性等方面有突出的表现。虽然 NS-3 刚刚发布,在功能模块上与 NS-2 相比还有很大差距,但是由于 NS-3 极其贴近现实的设计和强大的可扩展性操作,使得在进行新的课题研究的时候,可以很方便地建立网络模型并进行

代码实现,真正的将易操作性和可扩展性结合了起来。

文中对 NS-3 进行了全面的介绍,并结合项目经验对体系结构和主要模块进行了深入剖析,为 NS-3 及相关课题的后续研究和学习提供了有力的支持。

#### 参考文献:

- [1] 雷 擎,王行刚. 计算机网络模拟方法与工具[J]. 通信学报,2001,22(9):84-90.
- [2] Henderson T R, Floyd S, Riley G F. NS-3 Project Goals [C]//Proceedings of the Workshop of Network Simulation. Pisa, Italy:[s. n.],2006.
- [3] 李 越,钱德沛,何 莹,等. 网络仿真器 NS 问题分析及改进方案[J]. 系统仿真学报,2005,17(11):2832-2836.
- [4] Ye Qiang, MacGregor M H. Combining Petri Nets and ns-2: A Hybrid Method for Analysis and Simulation[C]//Proceedings of the 4th Annual Communication Networks and Services Research Conference (CNSR'06). Moncton, New Brunswick:[s. n.], 2006.
- [5] Riley G F. The Georgia Tech Network Simulator[C]//Proceedings of the ACM SIGCOMM 2003 Workshops. New York, NY, USA:[s. n.], 2003.
- [6] Lacage M, Henderson T R. Yet another network simulator [C]//In WNS2006: Proceeding from the ACM 2006 workshop on NS-2: the IP network simulator. New York, NY, USA:[s. n.], 2006.
- [7] Henderson T R, Lacage M, Riley G F. Network Simulations with the ns-3 Simulator[C]//Proceedings of the ACM SIGCOMM'08. Seattle, Washington:[s. n.], 2008.
- [8] Ns-3 developers. ns-3 Tutorial[EB/OL]. 2008-10-07. <http://www.nsnam.org/docs/tutorial/index.html>.
- [9] Ns-3 developers. ns-3 Reference Manual[EB/OL]. 2008-10-07. <http://www.nsnam.org/docs/manual.html>.
- [10] Henderson T. ns-3 tutorial slides[C]//Proceedings of the International Conference on Simulation Tools and Techniques 2008(Simutools 2008). Mercure Marseille, France:[s. n.], 2008.
- [11] [J]. 计算机技术与发展,2007,17(7):127-130.
- [4] Jacobson V. Congestion Avoidance and Control[J]. ACM Computer Communication Review,1988,18(4):314-329.
- [5] Stevens W. RFC2001-TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms[DB/OL]. 2005-04[2006-12-10]. <http://www.faqs.org/rfcs/rfc2001.html>.
- [6] 黄 涛. 博弈论教程[M]. 北京:首都经济贸易大学出版社,2004.
- [7] Shenker S. Fundamental design issues for the future Internet [J]. IEEE journal on selected areas in communication - ns, 1995,13(7):1176-1188.
- [8] Varian H R. Microeconomic Analysis[M]. 北京:经济科学出版社,2001.
- [9] Odlyzko A M. Paris Metro Pricing: The minimalist differentiated services solution[C]//in 1999 Seventh International Workshop on Quality of Service. [s. l.]:IEEE, 1999:159-161.
- [10] Stallings W. High-speed Networks: TCP/IP and ATM Design Principles[M]. 北京:电子工业出版社,1999.
- [11] 柯志享. 计算机网络实验——以 NS2 模拟工具实作[M]. 台北:学业行销股份有限公司,2005.

(上接第 79 页)