应用数学和力学，第9卷第9期(1988年9月
Applied Mathematics and Mechanics

应用数学和力学编委会编
重庆出版社出版

一个具有四元反馈的CTM型堆栈协议*

陈永义　尤传华

(兰州大学，1987年9月15日收到)

摘　要

我们分析了一类具有四元反馈的CTM (Capetankis-Tsybakov-Mikhailov) 型堆栈协议对於延迟存取的情形，求得了解除冲突期间的长度的数学期望的明显表达式；对於延迟和直接存取两种情形，我们利用数值计算求出其最大信道容量分别为0.4140和0.41445　包/时片．

一、引　言

近年来随机多路存取理论 (Random Multiple Access，简称RMA) 得到了很大的发展，其所以如此是因为随机多路存取是为多个用户分配信道的最有效的工具之一．这种信道的一个典型例子是人造卫星信道，其中卫星作为发射机应答器工作，而各个发送器(用户)产生的信号，通过此信道为许多接受器接收，这就产生了如何给各个用户分配信道的问题，而问题的核心则是解除冲突．

第一个随机多路存取的冲突解除算法 (Collision Resolution Alyorithm，简称CRA) 是在1970年提出的ALOHA算法(参看[1])．当用户较多，消息不长的，此算法比常规的时区分划多路存取法(Time DiVision Multiple Access简称TDMA)有效得多，因而ALOHA算法一直受到工程界的重视，但是若不加外部控制此算法是不稳定的，按此算法，终将使重新传输的用户数趋于无穷，而使传输率趋于零(参看[2])．1977年Capetanckis在英国(参看[2])，1978年Tsybakov和Mikhailov在苏联 (参看[3]) 各自独立地提出了类似的冲突解除算法(简称CTM-CRA)．当包的产生构成Poisson过程，且其到来率λ小于某个临界值时，此算法是稳定的．通常我们假设在同一时片内若有多于一个用户传输包就发生冲突，而该时片内传输的所有的包都失效 (无法正确传输)，在每个时片末各个用户都知道信道反馈，并根据反馈的历史传输其包．近年来不少文献从事于具有二元或三元反馈的CTM型协议的分析 (参看[4]～[9])．在[10]中考查了无反馈冲突信道，在[11]和[12]中考查了冲突包数为已知数的情形．

在本文中，我们分析了一种具有四元反馈的CTM型堆栈协议(Stack Protocol)．对于延迟存取情形，我们求出了解除冲突期的长度(CRI duration)的数学期望的明显表达式，通过数值计算，我们得知，对于延迟存取和直接存取两种情形，最大信道容量分别为0.4140和0.41445包/时片．

* 叶开沅推荐

二、说明和数学模型

〈一〉、假设信道是无错误的，有多个用户分配此信道，各个用户传输一定长度的消息（一个定长的消息称为一个包），每个时片中产生的新包的数目构成一个独立同分布的随机变量列，记为$\{X_i\}$（$i\geqslant1$），其共同的分布为具有参数λ的Polsson分布。

〈二〉、又假设时间是分片的，因而可看成离散的。每个时片等于传输一个包所需的时间，各个包仅在时片开头传输（亦即用户关于时片是同步的）。当两个或更多个用户同时传输其包时，称所传输的包发生冲突，并且没有一个包可被正确地接收。每个用户必须按照相同的算法再传输其冲突包，直至其包被成功地接收为止；并且各个用户均不能利用别的用户的活动性信息。

下面我们简单回顾一下非修正堆栈算法（此为二元反馈算法）和通常的修正堆栈算法（三元反馈）的主要规则。

A、非修正堆栈算法

〈一〉、〈二〉两点仍成立

每个活动用户维持一个“概念堆栈”（Conceptual Stack），当非活动用户变为活动时，用户进入堆栈中的0水平，此用户将在最近一个时片内传输，只要它处于0水平，总是如此传输。

在每个时片末，用户按如下程序决定在其堆栈中的位置（对各个用户使用同一程序，且各用户不能相互通告各自的堆栈状态）：

〈a〉、跟着一个非冲突的时片

堆栈水平0中的一个用户变为非活动的（至多一个这种用户，即系统宽度为1），而处于别的堆栈水平中的用户全将其水平降低1。

〈b〉、跟着一个冲突时片

所有处于水平i（$i\geqslant1$）的用户变到水平$i+1$；而0水平中的用户们分为两组，一组留在水平0，另一组进到水平1，没有一个用户知道水平1中究竟有多少个用户。这种分组法可看成掷一个硬币，上述决策构成一列独立的贝努里试验，其中留在水平0的概率为p。本文中假设p对所有用户都相同，并且关于时同是齐次的。记I为留在0水平中的用户数，显然I是一个随机变量，服从二项分布B（n, p），其中n为冲突用户数（又称为冲突重数）。

B、修正堆栈算法

当一个时片静止（不传输任何包）而且最近的一个非静止的时片产生一个冲突时，在紧接着的下一个时片中，并不是所有前面那些冲突用户（此时它们全处于水平1）都自动地被许可再传输（这必然产生一个冲突）而是让它们立即各掷一个硬币，以决定它们进到0水平或留在1水平，因此有可能减少一个注定无用的时片，同时，别的水平数大于1的用户不变其水平。

我们考查了一种具有四元反馈的多路存取堆栈算法，其规则如下：

〈一〉、〈二〉两点仍成立

在第k时片末，每个监听信道的用户知道一个公共的四元反馈随机变量$B(k)$，$B(k)\in\{0,1,2,m\}$。$B(k)=i$，$i=0,1$或2，如果在第k时片中传输的包的数目为i；$B(k)=m$，表示在第k时片中有三个或更多个传输的包。

当$B(k)=2$，$B(k+1)\geqslant 1$；或者$B(k)=m$，$B(k+1)\geqslant 2$时，我们的算法与非修正堆栈算法相同。

当$B(k)=2$，$B(k+1)=0$时，我们的算法与通常的修正堆栈算法是一致的。

但是，当$B(k)=m$，$B(k+1)=1$或0时，我们不是让所有先前的冲突用户都再传输（这必然产生冲突），而是让它们各掷一个硬币，以决定留0水平或去1水平，因而又可能挽救一些无用时片。

由上述规则我们可看出，按照我们的算法将比通常的堆栈算法和其修正算法得到更大的最大信道容量。

我们用L_n表示已知冲突重数为n时的解除冲突期的长度。L_n包含两部分：初始时片及其随后的时片，直至所有处于$i(i\geqslant 1)$，水平中的活动用户都回到原水平，而其原来空闲的堆栈水平也回到原状态L_n不等于在冲突后直到第n次成功传输为止的时片数，一方面由于在直接存取情形，在一个冲突解除前一般都有新包到来；另一方面，甚至当所有冲突包以及随后来的新包都被成功地传输后，仍然可能在别的活动用户的堆栈中有空闲水平，由于用户们相互不能通告堆栈水平，因此并不知道此情形，故只能将这种时片赋闲（参看[9]）

下面我们将讨论两种模型：

（a） 延迟存取协议

按此协议所有用户一直监听信道，因此当一个用户要传输一个包时，它就知道信道是闲着的抑或正处于解除冲突期，在前一种情形，此用户立即变为活动用户，而在后它一情形它必须等到冲突解除期过去了才进入其堆栈中的0水平。

（b） 直接存取协议

按此协议，非活动用户勿需连续监听信道，而仅当它们想传输消息时才监听信道，当一个用户得到一个包时，立即变为活动的，并按我们的解除冲突算法传输消息包。它可能正碰上一个闲着的时片，因此就成功地传输出去；也可能引起冲突，因而它就加入解除过程，如同此冲突开始一个解除冲突期一样。这种存取协议可望增大信道容量。

三、延迟存取协议情形的基本方程

为了方便，我们规定“退化”的解除冲突期L_0和L_1等于1。

按上述信道协议，我们可得到关于随机变量L_n的如下递推关系：

$$L_0=L_1=1 \tag{3.1}$$

$$L_2=1+\begin{cases} L_2 & (I=0) \\ L_I+L_{2-I} & (I>0) \end{cases} \tag{3.2}$$

$$L_n=1+\begin{cases} L_n & (I=0) \\ L_{n-1} & (I=1) \\ L_I+L_{n-I} & (I\geqslant 2,\ n\geqslant 3) \end{cases} \tag{3.3}$$

更确切地讲，L_n和I应分别记为$L_n(t)$和$I_n(t)$（对于第t时片中的n个冲突用户的解除冲突期而言），因此(3.1)，(3.2)和(3.3)应写为：

$$L_0(t)=L_1(t)=1$$

$$L_2(t)=1+\begin{cases} L_2(t+1),\text{ 若}I_2(t+1)=0\ (\text{即}B(t)=2, B(t+1)=0) \\ L_{I_2(t+1)}(t+1)+L_{2-I_2(t+1)}(t+1),\text{若} \\ \quad I_2(t+1)>0\ (\text{即}B(t)=2,\ B(t+1)>0), \end{cases}$$

$$L_n(t)=1+\begin{cases} L_n(t+1),\text{ 若}I_n(t+1)=0\ (B(t)\geqslant 3,\ B(t+1)=0) \\ L_{n-1}(t+1),\text{ 若}I_n(t+1)=1\ (B(t)\geqslant 3,\ B(t+1)=1) \\ L_{I_n(t+1)}(t+1)+L_{n-I_n(t+1)}(t+1),\text{ 若} \\ \quad I_n(t+1)\geqslant 2\ (B(t)\geqslant 3,\ B(t+1)\geqslant 2) \qquad (n\geqslant 3) \end{cases}$$

但是由于$L_n(t)$和$I_n(t)$的分布仅与n有关，与t无关(参看[13])。而下面我们又仅仅对它们的分布和数学特征感兴趣，因此在本文中我们仍用L_n和I_n(甚至更简单地用I)表示$L_n(t)$和$I_n(t)$而不会造成混淆。

定义$l_n=EL_n$，并在(3.1)和(3.2)中取数学期望，得到

$$l_0=l_1=1 \tag{3.4}$$

$$l_2=1+q^2l_2+\sum_{i=1}^{2}\binom{2}{i}p^iq^{2-i}(l_i+l_{2-i}) \tag{3.5}$$

其中$q=1-p$。

我们立即得到

$$l_2=1+\frac{2(1+p)-1-p^2}{1-p^2-q^2}=\frac{2+2pq-q^2}{1-p^2-q^2} \tag{3.6}$$

对于$n\geqslant 3$，我们有

$$L_n=1+L_I+L_{n-I}-(\mathrm{I}_{I=0}+\mathrm{I}_{I=1}) \tag{3.7}$$

其中

$$\mathrm{I}_A(\omega)=\begin{cases} 1,\ \omega\in A \\ 0,\ \omega\bar{\in} A \end{cases}$$

是A的示性函数。

在(3.7)中取数学期望，得到

$$l_n=1+\sum_{i=0}^{n}\binom{n}{i}p^iq^{n-i}(l_i+l_{n-i})-(q^n+npq^{n-1}) \tag{3.8}$$

为解(3.8)，我们定义l_n的指数母函数

$$A(z)=\sum_{n=0}^{\infty}l_nz^n/n! \tag{3.9}$$

在(3.8)中利用(3.9)，得

$$\sum_{n=3}^{\infty}l_nz^n/n!=\sum_{n=3}^{\infty}(1-q^n-npq^{n-1})\frac{z^n}{n!}+\sum_{n=3}^{\infty}\sum_{i=0}^{n}\binom{n}{i}p^iq^{n-i}(l_i+l_{n-i})\frac{z^n}{n!}$$

$$=e^{z}-e^{qz}-pz-(1-q^{2})\frac{z^{2}}{2}-pze^{qz}+pz+pqz^{2}+e^{qz}A(pz)$$

$$+e^{pz}A(qz)-2-2z-[q^{2}(1+l_{2})+4pq+p^{2}(l_{2}+1)]\frac{z^{2}}{2}$$

由此又得到

$$A(z)-e^{qz}A(pz)-e^{pz}A(qz)=e^{z}-e^{qz}(1+pz)-1-z$$

$$+\frac{z^{2}}{2}[l_{2}-2pq-1+q^{2}-(p^{2}+q^{2})(l_{2}+1)]$$

$$=e^{z}-e^{qz}(1+pz)-1-z+z^{2}pq \tag{3.10}$$

记$B(z)=e^{-z}A(z)$，我们得到更简便的方程：

$$B(z)-B(pz)-B(qz)$$

$$=1-e^{-pz}(1+pz)+e^{-z}(pqz^{2}-1-z) \tag{3.11}$$

令(3.11)中两端z^n的系数相等，得

$$b_{0}=1 \tag{3.12}$$

$$b_{2}=\frac{2pq+p^{2}+1}{2(1-p^{2}-q^{2})} \tag{3.13}$$

$$b_{n}=\frac{(-1)^{n}}{n!(1-p^{n}-q^{n})}[p^{n}(n-1)+n(n-1)pq+n-1]\qquad(n\geqslant3) \tag{3.14}$$

同时又有

$$b_{1}=B'(0)=0$$

注意到

$$l_{n}=n!\sum_{k=0}^{n}\frac{b_{k}}{(n-k)!}\qquad(n\geqslant3)$$

我们得到，对于$n\geqslant3$

$$l_{n}=1+\sum_{k=2}^{n}\binom{n}{k}\frac{(-1)^{k}(k-1)(1+p^{k}+kpq)}{1-p^{k}-q^{k}} \tag{3.15}$$

再注意到二项式系数$\binom{n}{k}=0$（若$k>n$），我们即知(3.15)对于一切$n\geqslant0$成立.

四、延迟存取协议的数值结果

注意到方程(3.10)和(3.11)关于p和q不是对称的，易知l_n并不在$p=1/2$取最小值. 事实上，对于l_2我们可知在$p=\sqrt{2}-1$达到最小值$3+\sqrt{2}$，而对于$p=1/2$却有$l_2=4.5$. 对于l_3，$p=0.3979$给出最小值6.2944，而对于$p=1/2$，$l_3=6.5$. 因此可知，不存在单个的p值使所有的l_n都达到最小值. 利用数值计算得如下结果（表1）：

表 1

n	p_n^*	l_n	n/l_n	n	p_n^*	l_n	n/l_n
2	0.4142	4.4142	0.4531	17	0.3742	40.3547	0.4213
3	0.3979	6.2944	0.4766	18	0.3741	42.7871	0.4207
4	0.3782	8.7166	0.4589	19	0.3740	45.2197	0.4202
5	0.3703	11.1459	0.4486	20	0.3739	47.6525	0.4197
6	0.3693	13.5822	0.4418	25	0.3740	59.8167	0.4179
7	0.3708	16.0215	0.4369	30	0.3741	71.9808	0.4168
8	0.3726	18.4599	0.4334	35	0.3742	84.1443	0.4160
9	0.3739	20.8961	0.4307	40	0.3742	96.3066	0.4153
10	0.3746	23.3303	0.4286	45	0.3742	108.4729	0.4149
11	0.3749	25.7631	0.4270	50	0.3742	120.5836	0.4147
12	0.3749	28.1951	0.4256	55	0.3742	132.7156	0.4144
13	0.3748	30.6268	0.4245	60	0.3742	144.8348	0.4143
14	0.3746	33.0585	0.4235	65	0.3742	156.9442	0.4142
15	0.3745	35.4904	0.4226	70	0.3742	169.0699	0.4140
16	0.3743	37.9224	0.4219	75	0.3742	181.1551	0.4140

表中p_n^*(表示冲突重数为n时，使l_n最小的p)是一个振幅递减的振荡序列。当$n\gg 1$时，p_n^*在0.3742左右摆动，且对充分大的n，n/l_n很接近于0.4140，由是即知最大信道容量 $\lambda_{\max}\approx 0.4140$（参看[5]，[13]）。

五、直接存取协议情形的基本方程

仿延迟存取协议情形，我们不难得到L_n的如下关系：

$$L_0=L_1=1$$

$$L_2=1+\begin{cases} L_2, & X+I=0\ (\text{即}B(t)=2,\ B(t+1)=0) \\ L_{I+X}+L_{2-I+Y}, & X+I>0\ (\text{即}B(t)=2,\ B(t+1)>0) \end{cases}$$

$$L_n=1+\begin{cases} L_n, & X+I=0\ (\text{即}B(t)\geqslant 3,\ B(t+1)=0) \\ L_{n+X-1}, & X+I=1\ (\text{即}B(t)\geqslant 3,\ B(t+1)=1) \\ L_{I+X}+L_{n-I+Y}, & X+I\geqslant 2\ (\text{即}B(t)\geqslant 3,\ B(t+1)\geqslant 2) \end{cases}$$

其中 I为立即再传输的用户数，X为立即再传输的时片中新来的包数，Y为$I+X$个用户的解除冲突期后紧接着的时片中新来的包数。

按照上面的规定，I是服从二项分布$B(n,\ p)$的随机变量，其中n为冲突重数。X和Y都服从Poisson分布$P(\lambda)$。X，Y，I相互独立。

注意到

$$\begin{aligned} L_2&=1+1_{X+I=0}L_2+1_{X+I>0}(L_{I+X}+L_{2+Y-I}) \\ &=1+L_{I+X}+L_{2-I+Y}+1_{X=0,I=0}(L_2-1-L_{2+Y}), \\ L_n&=1+L_{I+X}+L_{n-I+Y}+1_{X=0,I=0}(L_n-1-L_{n+Y}) \\ &\quad +1_{X=1,I=0}(L_n-1-L_{n+Y})+1_{X=0,I=1}(L_{n-1}-1-L_{n-1+Y}) \qquad (n\geqslant 3). \end{aligned}$$

在(3.1)，(3.4)，(3.5)两端取数学期望，得

$$l_0=l_1=1,$$

$$l_2=1+\sum_{k=0}^{\infty}\sum_{l=0}^{\infty}\sum_{i=0}^{2}\binom{2}{i}p^i q^{2-i}(l_{i+k}+l_{2-i+l})\ \frac{\lambda^k\lambda^l}{k!\,l!}\ e^{-2\lambda}$$

$$+q^2e^{-\lambda}\left(l_2-1-\sum_{l=0}^{\infty}e^{-\lambda}\frac{\lambda^l}{l!}l_{2+l}\right),$$

$$l_n=1+\sum_{k=0}^{\infty}\sum_{l=0}^{\infty}\sum_{i=0}^{n}\binom{n}{i}p^iq^{n-i}(l_{i+k}+l_{n-i+l})\frac{\lambda^k\lambda^l}{k!l!}e^{-2\lambda}$$

$$+q^ne^{-\lambda}(1+\lambda)\left(l_n-1-\sum_{l=0}^{\infty}e^{-\lambda}\frac{\lambda^l}{l!}l_{n+l}\right)$$

$$+npq^{n-1}e^{-\lambda}\left(l_{n-1}-1-\sum_{l=0}^{\infty}e^{-\lambda}\frac{\lambda^l}{l!}l_{n-1+l}\right)\qquad(n\geqslant 3).$$

记$a_n=l_n/n!$ $(n\geqslant 0)$，可得

$$a_0=a_1=1 \tag{5.1}$$

$$a_2=\frac{1}{2}+e^{-\lambda}\left[\sum_{k=0}^{\infty}\sum_{i=0}^{2}\frac{p^iq^{2-i}\lambda^k(i+k)!}{i!(2-i)!k!}a_{i+k}\right.$$

$$\left.+\sum_{l=0}^{\infty}\sum_{i=0}^{2}\frac{p^iq^{2-i}\lambda^l(2-i+l)!}{i!(2-i)!l!}a_{2-i+l}+q^2\left(a_2-\frac{1}{2}-\sum_{l=0}^{\infty}e^{-\lambda}\frac{\lambda^l(2+l)!}{2\cdot e!}a_{2+l}\right)\right] \tag{5.2}$$

对于$n\geqslant 3$，有

$$a_n=\frac{1}{n!}+e^{-\lambda}\left[\sum_{k=0}^{\infty}\sum_{i=0}^{n}\frac{p^iq^{n-i}\lambda^k(i+k)!}{i!(n-i)!k!}a_{i+k}\right.$$

$$+\sum_{l=0}^{\infty}\sum_{i=0}^{n}\frac{p^iq^{n-i}\lambda^l(n-i+l)!}{i!(n-i)!l!}a_{n-i+l}$$

$$+q^n(1+\lambda)\left(a_n-\frac{1}{n!}-\sum_{l=0}^{\infty}e^{-\lambda}\frac{\lambda^l(n+l)!}{l!n!}a_{n+l}\right)$$

$$\left.+npq^{n-1}\left(\frac{1}{n}\cdot a_{n-1}-\frac{1}{n!}-\sum_{l=0}^{\infty}e^{-\lambda}\frac{\lambda^l(n-1+l)!}{l!n!}a_{n-1+l}\right)\right] \tag{5.3}$$

这不是递推方程，我们尚无现成方法求出精确解，但我们可用截断法求无穷维线性方程组(5.1)、(5.2)及(5.3)的近似解。因为我们的算法比通常的堆栈算法少一些无用时片，因而其l_n必不大于通常堆栈算法的$l_n=O(n)$（参看[6]），故可期望$a_n=O(1/(n-1)!)$，此下降速度使截断法很稳定。

我们不难将方程组(5.1)，(5.2)和(5.3)写成如下更方便的形式：

$$a_0=a_1=1,$$

$$(1-q^2e^{-\lambda})a_2-\sum_{k=0}^{\infty}\sum_{i=0}^{2}\frac{p^iq^{2-i}\lambda^k(i+k)!}{i!(2-i)!k!}e^{-\lambda}a_{i+k}$$

$$-\sum_{l=0}^{\infty}\sum_{i=0}^{2}\frac{p^iq^{2-i}\lambda^l(2-i+l)!}{i!(2-i)!l!}e^{-\lambda}\cdot a_{2-i+l}+q^2\sum_{l=0}^{\infty}\frac{\lambda^l(2+l)!}{l!2!}e^{-2\lambda}\cdot a_{2+l}$$

$$=(1-q^2e^{-\lambda})/2,$$

对$n\geqslant 3$，有

$$[1-q^n(1+\lambda)e^{-\lambda}]a_n-pq^{n-1}e^{-\lambda}a_{n-1}-\sum_{k=0}^{\infty}\sum_{i=0}^{n}\frac{p^iq^{n-i}\lambda^k(i+k)!}{i!(n-i)!k!}e^{-\lambda}a_{i+k}$$

$$-\sum_{l=0}^{\infty}\sum_{i=0}^{n}\frac{p^iq^{n-i}\lambda^l(n-i+l)!}{i!(n-i)!l!}e^{-\lambda}a_{n-i+l}+q^n(1+\lambda)\sum_{l=0}^{\infty}\frac{\lambda^l(n+l)!}{l!n!}e^{-2\lambda}\cdot a_{n+l}$$

$$+npq^{n-1}\sum_{l=0}^{\infty}\frac{\lambda^l(n-1+l)!}{l!n!}e^{-2\lambda}a_{n-1+l}$$

$$=\frac{1}{n!}-\frac{1}{n!}q^n(1+\lambda)e^{-\lambda}-\frac{1}{(n-1)!}pq^{n-1}e^{-\lambda}$$

记上述方程组中第n个方程的a_m的系数为M_{nm}，则可将上述方程组写成如下形式：

$$M\vec{a}=\vec{f} \tag{5.4}$$

其中矩阵$M=(M_{nm})_{n,m\geqslant 2}$；列向量$\vec{a}=(a_2,\ a_3,\cdots)^T$，$\vec{f}=(f_2,f_3,\cdots)^T$。

我们不难得知，$\vec{f}$的分量为：

$$f_2=(1-q^2e^{-\lambda})/2,$$

$$f_n=\frac{1}{n!}-\frac{1}{n!}q^n(1+\lambda)e^{-\lambda}-\frac{1}{(n-1)!}pq^{n-1}e^{-\lambda}\quad (n\geqslant 3)$$

M的分量为：

$$M_{22}=1-q^2e^{-\lambda}+q^2e^{-2\lambda}-\sum_{i=0}^{2}\frac{p^iq^{2-i}\lambda^{2-i}2!}{i!(2-i)!(2-i)!}e^{-\lambda}-\sum_{i=0}^{2}\frac{p^iq^{2-i}\lambda^i2!}{i!(2-i)!i!}e^{-\lambda}$$

M_{2m}，$m\geqslant 3$，由下式计算：

$$M_{2m}=-\sum_{i=0}^{2}\frac{p^iq^{2-i}\lambda^{m-i}m!}{i!(2-i)!(m-i)!}e^{-\lambda}-\sum_{i=0}^{2}\frac{p^iq^{2-i}\lambda^{i+m-2}m!}{i!(2-i)!(i+m-2)!}e^{-\lambda}$$

$$+q^2\frac{\lambda^{m-2}m!}{(m-2)!2!}e^{-2\lambda}$$

对于$n\geqslant 3$，我们可区分如下四种情形：

(a) $2\leqslant m<n-1$

$$M_{nm}=-\sum_{i=0}^{m}\frac{p^iq^{n-i}\lambda^{m-i}m!}{i!(n-i)!(m-i)!}e^{-\lambda}-\sum_{i=n-m}^{n}\frac{p^iq^{n-i}\lambda^{i+m-n}m!}{i!(n-i)!(i+m-n)!}e^{-\lambda}$$

(b) $m=n-1$

$$M_{n\,n-1}=-\sum_{i=0}^{n-1}\frac{p^iq^{n-i}\lambda^{n-i-1}(n-1)!}{i!(n-i)!(n-i-1)!}e^{-\lambda}-\sum_{i=1}^{n}\frac{p^iq^{n-i}(n-1)!\lambda^{i-1}}{i!(n-i)!(i-1)!}e^{-\lambda}$$

$$-pq^{n-1}(1-e^{-\lambda})e^{-\lambda},$$

(c) $m=n$,

$$M_{nn}=-\sum_{i=0}^{n}\frac{p^iq^{n-i}\lambda^{n-i}n!}{i!(n-i)!(n-i)!}e^{-\lambda}-\sum_{i=0}^{n}\frac{p^iq^{n-i}\lambda^in!}{i!(n-i)i!}e^{-\lambda}$$

$$+1-q^n(1+\lambda)e^{-\lambda}+q^n(1+\lambda)e^{-2\lambda}+npq^{n-1}\lambda e^{-2\lambda}$$

(d) $m>n$

$$M_{nm}=-\sum_{i=0}^{n}\frac{p^i q^{n-i}\lambda^{m-i}m!}{i!(n-i)!(m-i)!}e^{-\lambda}-\sum_{i=0}^{n}\frac{p^i q^{n-i}\lambda^{i+m-n}m!}{i!(n-i)!(i+m-n)!}e^{-\lambda}$$

$$+q^n(1+\lambda)\frac{\lambda^{m-n}m!}{(m-n)!n!}e^{-2\lambda}+npq^{n-1}\frac{\lambda^{m-n+1}m!}{(m-n+1)!n!}e^{-2\lambda}$$

六、直接存取协议的数值结果

由于$a_n=O(1/(n-1)!)$下降速度很快，用截断法解方程组(5.4)时，截断方程的阶数N取成适中的值，比如15～30即可，所得结果相差甚微。由最大信道容量的定义我们知道，对固定的p，增加到来率λ，直至l_n变为无穷时的λ即为对应该p的最大信道容量$\lambda_{\max}$（实际计算时，当截断方程组的解由正解变为负解时，即表示方程组没有非负有限解，此时的λ即为$\lambda_{\max}$)。我们先对p,λ用步长为0.01的等分法计算，求出近似的p的最优值及相应的$\lambda_{\max}$，从而确定p的最优区间为[0.35,0.40]，再用优选法逐步求得最优p值及相应的$\lambda_{\max}$的精密值，这也是下表中p，λ的值的有效数字的位数不相同的原因。

表 2

p	$\lambda_{\max}$	p	$\lambda_{\max}$
0.35	0.413	0.40	0.4141
0.36	0.4141	0.41	0.413
0.37	0.4143	0.45	0.410
[0.377,0.382]	0.41445	0.50	0.40
0.39	0.4143	0.55	0.39

兹将计算结果列于表2。

由上表可见，当p在[0.377，0.382]之间变化时，$\lambda_{\max}$都为0.41445，有相同的五位有效数字。但$p=0.3787$时，其l_n最小，因此p的最优值应为0.3787。

我们可注意到p在[0.36，0.40]之间变化时，$\lambda_{\max}$的变化不超过1‰，可见在此区间里，$\lambda_{\max}$对p并不太敏感。同时，计算表明，截断方程的阶数N适中时，$\lambda_{\max}$、l_n对N也不太敏感，比如$N=20$与$N=30$时，相应于$p=0.3787$(及0.390等)的$\lambda_{\max}$都是0.41445，相应的l_n也很接近。

总之，对于本文提出的协议，当$p=0.3787$时达到最大信道容量0.41445，这比普通直接存取协议（三元反馈）提高了5%（参看[6]）。

致谢 法国国家信息与自动化研究所Guy Fayolle博士建议著者研究随机多路存取的问题，并一起作过多次有益的讨论，特此致谢。

参 考 文 献

[1] Abramson, N., The ALOHA system—Another alternative for computer communications, *Proc. AFIPS Conf. Fall Joint Comp. Conf.*, **37** (1970), 281—285.

[2] Capetanakis, J. I., Tree algorithms for packet broadcast channels, *IEEE Trans. Inf. Th.*, **IT-25**, 5 (1979), 505—515.

[3] Tsybakov, B. S. and V. A. Mikhailov, Free synchronous packet access in a broadcast channel with feedback, *Proc. of Inf. Trans.*, **14** (1978), 32—59.

[4] Mehravari, N. and T. Berger, Poisson multiple-access contention with binary feedback, *IEEE Trans. Inf. Th.*, **30**, 5 (1985), 745—751.

[5] Hofri, M., Stack algorithms for collision-detecting channels and their analysis: A limited survey, *Proceedings of the International Seminar*, Paris, France, edited by F. Baccelli and G. Fayolle, Springer-Verlag, January (1983).

[6] Fayolle, G. and M. Hofri, On the capacity of a collision resolution channel under stack-based collision resolution algorithms, Technion, Haifa, Israel, Rep. 327, Oct. (1983).

[7] Fayolle, G., P. Flajolet, M. Hofri, and P. Jacquet, Analysis of a stack algorithm for random multiple-access communication, *IEEE Trans. INF. Th.*, March, **IT-31**, 2 (1985), 244—254.

[8] Mathys, P. and P. Flajolet, Q-ary collision resolution algorithms in random-access systems with free or blocked channel access, *IEEE Trans. Inf. Th.* **IT-31**, 2, March (1985), 217—243.

[9] Fayolle, G., P. Flajolet and M. Hofri, On a functional equation arising in the analysis of a protocol for a multi-access broadcast channel, *Internal Rep.*, 131, INRIA, France, Apr. (1982).

[10] Massey, J. L. and P. Mathys, The collision channel without feedback, *IEEE Trans. Inf. Th.*, **IT-31**, 2, March (1985), 192—204.

[11] Tsybakov, B. S., Resolution of conflict with known multiplicity, *Problem Peredach. Inform.*, **16**, 2, Apr.-June (1980), 69—82. (in Russian)

[12] Georgiadis, L. and P. Papantoni-Kazakos, A collision resolution protocol for random access channel with energy detectors, *IEEE Trans. Com.* **Com-30**, 11, (1982), 2413—2420.

[13] Tsybakov, B. S., Survey of USSS contributions to random multiple-access communications, *IEEE Trans. Inf. Th.*, **TH-31**, 2, March (1985).

A Stack Protocol of CTM Type with Quartet Feedback

Chen Yong-yi   You Chuan-hua

(*Lanzhou University, Lanzhou*)

Abstract

In this paper we analysed a stack protocol of the CTM (Capetanakis-Tsybakov-Mikhailov) type with quartet feedback. We obtained the explicit expression of the expectation of CRI (Collision Resolution Interval) duration for the delayed access case. By means of numerical calculation we gave respectively the maximal capacities of channel of 0.4140 and 0.41445 packets/slot for both the delayed case and immediate cases.