无线网络中优化TCP性能的网络编码方法研究

葛卫民 许文庆 朱海颖 李 娟 冉 放
(天津大学计算机科学与技术学院 天津 300072)

摘 要 网络编码的出现为改进网络的传输性能提供了新的方法。J. K. Sundararajan等人将网络编码技术与传输控制协议相结合,提出了基于网络编码的TCP/NC协议,其在改进无线网络中传统TCP的性能方面取得了明显效果。但该协议及其改进协议存在的数据传输和解码操作同步性问题会严重影响TCP/NC性能。提出的改进协议TCP/NCW根据解码时间调节解码窗口,以确保数据传输和解码操作同步,从而获得更好性能。运用排队论分析了TCP/NCW最优解码窗口的存在性。在NS2中的仿真结果表明,在静态场景和动态场景下TCP/NCW的吞吐率比TCP/Vegas和TCP/NC有显著提高,同时也具有较好的公平性。

关键词 无线ad hoc网络,网络编码,TCP/NC,最优窗口
中图法分类号 TP393 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.6.028

Research on Network Coding to Optimize Performance of TCP in Wireless Networks

GE Wei-min XU Wen-qing ZHU Hai-ying LI Juan RAN Fang
(School of Computer Science and Technology, Tianjin University, Tianjin 300072, China)

Abstract The emergence of network coding provides a new method for improving TCP performance in wireless network. J. K. Sundararajan et al. proposed a new protocol called TCP/NC based on network coding combining the network coding and transmission control protocol, which achieves remarkable results in improving the performance of TCP in wireless network. But the synchronism problem in data transmission and decoding operation may seriously affect the performance of TCP/NC and its modified protocol is not considered. To address this issue, a revised protocol TCP/NCW was proposed in this paper. We introduced the decoding window adjustment scheme based on TCP/NC. In TCP/NCW, the decoding window is adjusted according to the decoding time and will finally reach an optimal window size. This scheme can ensure the synchronization of data transfer and decoding operation. Therefore, this scheme can achieve better performance. We used the queuing theory to analyze the existence of the optimal decoding window of TCP/NCW. The simulation results with NS2 show that TCP/NCW achieves significant improvement in throughput compared with both TCP/Vegas and TCP/NC in different scenarios, without prejudice of fairness.

Keywords Wireless ad hoc network, Network coding, TCP/NC, Optimal window

1 引言

随着无线通信技术的快速发展,无线Ad hoc网络被广泛应用到军事、紧急救援和个人通信领域。它有动态网络拓扑、信道质量不可预测、节点能量有限等特点,因此造成分组丢失的原因通常不是网络拥塞。对于非拥塞引起的数据丢失,应该增加传输速率以克服链路的分组丢失,而传统TCP协议不能区分丢包原因,认为所有丢包都是因为网络拥塞和错误的启动拥塞控制机制,降低了传输速率,严重影响了吞吐率[1]。网络编码的出现为解决这个问题提供了一种新的有效途径。

网络编码最初是为了减少网络组播中的数据传输次数而提出的[2],此后引起了众多学者的关注和研究[3-6]。利用网络编码的纠删特性可以改善无线链路随机分组丢失对TCP的影响[3,7]。J. K. Sundararajan等人[8]提出了称为TCP/NC的编码传输方法,以一种简单而巧妙的方式将网络编码技术应用到现有的协议栈中,使得TCP协议在无线传输的情况下依然可以保持良好的实际吞吐率[8,9]。之后Minji Kim等人给出了TCP/NC协议的分析模型[10],通过理论分析说明了TCP/NC可以提高无线网络传输吞吐率。在此基础上,文献[11-13]通过分析冗余因子、编码窗口等参数对协议性能的影响,提出自适应冗余因子控制和编码策略等方法来进一步改进TCP/NC的性能 。但是,这些方法都没有考虑到解码操作与数据传输的同步性问题。当网络状况恶化使得网络传输延迟增大时,编码数据包进入接收端缓存的速率与TCP原始数据包进入发送端缓存的速率会严重不匹配,导致TCP/NC的性能下降,甚至会产生连接中断现象。

到稿日期:2014-07-09 返修日期:2014-09-13 本文受天津市软件产业发展基金项目:公交车辆嵌入式3G视频监控系统(2011E6-0009)资助。
葛卫民(1964—),男,博士,副教授,主要研究方向为无线网络协议性能与评价、移动计算和分布式数据库,E-mail:gewn@tju.edu.cn;许文庆(1991—),男,硕士生,主要研究方向为网络协议的仿真与在Linux系统下的实现;朱海颖(1989—),女,硕士生,主要研究方向为无线网络设计与仿真;李 娟(1985—),女,硕士,主要研究方向为无线网络设计与仿真;冉 放(1986—),男,硕士,主要研究方向为无线网络设计与仿真。

本文针对 TCP/NC 中存在的解码操作与数据传输的同步性问题，提出了一种自适应调整解码窗口的改进协议——TCP/NCW；通过理论分析和仿真说明了 TCP/NCW 最优解码窗口的存在性；重新设计了网络编码层发送端和接收端的算法以及网络编码层缓存区管理、编解码窗口的管理策略；使用 NS-2 软件在不同场景下分别对 TCP/Vegas、TCP/NC 和 TCP/NCW 协议进行了仿真，验证了 TCP/NCW 协议的优越性。

2 TCP/NC 协议基本原理及其存在的问题

2.1 TCP/NC 协议基本原理[8]

TCP/NC 协议主要思想是在 TCP 层和 IP 层之间引进一个网络编码(NC)层。TCP 发送端产生的数据包不再以单个数据包的形式发送，而是在发送端的 NC 层按一定的冗余比例进行随机线性编码[14]后以线性组合形式发往接收端。传输过程中传输路径上的中间节点不参与编解码操作。接收端每收到一个新的编码包就将系数向量作为新的一行存入解码矩阵，并将信息向量保存在解码缓存区中。通过高斯消元法将解码矩阵化为最简型(阶梯型)后，矩阵主元所在列对应的数据包称为可见包，并将解码出的原始数据包提交给 TCP 接收端，如图 1 所示。这样可以保持原有 TCP 的语义基本不变，使 TCP/NC 具有良好的兼容性。TCP 发送端产生的数据包被视为有限域上的向量，记为 $P_1,P_2,P_3,\cdots,P_k,\cdots$。假设 n_k 为第 k 个数据包到达缓存区时缓存区中数据包个数，W_m 为编码窗口大小，网路中实际传输的编码包可表示为式(1)：

$$X_k=\sum_{i=k-m_k+1}^{k}\alpha_i P_i,k=1,2,3,\cdots \tag{1}$$

式中，$m_k=\min(n_k,W_m)$，α_i 是同一有限域上的随机数，向量 $(a_{k-mk+1},\cdots,a_k)$ 记为 a_k，称为系数向量，X_k 称为信息向量。

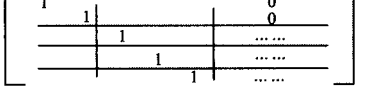

图 1 解码矩阵

2.2 编解码的同步性问题

为了使编码和解码操作顺利进行，TCP/NC 协议要求编码包进入接收端解码缓存的速率与 TCP 原始数据包进入发送端的速率相匹配。而在实际的网络传输中，由于传输路径的不同，不同数据包可能经历不同的传播和队列延迟，这种传输的不同步会严重影响解码速度。例如，当数据包 p_1 在接收端成为可见包时，接收端就会发送一个 p_1 的 ACK 给发送端，但由于传播延迟，发送端不能立即收到 ACK。所以在此之前，发送端继续发送含 p_1 的编码包。并且可能不断有新原始包从 TCP 发送端传递到编码缓存中，那么新编码组合就会不断产生，比如，p_1、p_2 组合包，p_1、p_2 和 p_3 组合包，p_1、p_2、p_3 和 p_4 组合包等。只有当发送端收到 p_1 的 ACK 后，才不会再发送含有 p_1 的编码包。而由于 TCP/NC 协议独特的“看见”策略，每一个新的编码包到达接收端都会带来一个新的可见包。在这种情况下，如果原始数据包进入编码缓存区的速率与编码包进入解码缓存区的速率不一致，将会导致解码矩阵很长时间内都不能满秩而变得过于庞大，所有的已解码和未解码的数据包都将在解码缓存中等待，甚至引起数据错乱。另外，规模增大的矩阵运算，必定会增加计算开销。当网络状况好时，TCP/NC 性能较好；但当网络状况恶化时，这些因素都会造成其性能的急剧下降。

3 TCP/NCW 协议

3.1 最优解码窗口思想

TCP/NC 协议中，接收端在解码矩阵满秩时应用高斯消元法求解原始数据包，但传播延时带来的不同步问题，会使得解码速率大幅下降。为此，提出了改进协议 TCP/NCW，在 TCP/NC 的基础上加入了根据解码时间调整解码窗口大小使之达到最优解码窗口的策略。解码窗口实际上是一个矩阵，用于保存编码包的系数向量以及进行解码操作。设解码窗口为 W，为了使数据包的解码与传输同步，每一轮的编解码操作都根据上一次解码的等待时间调整一次 W，使之动态地达到一个最优值。同时通过对发送端发包速率的控制，确保解码窗口达到 W 时解码矩阵进入满秩状态，消除传播延时带来的影响，使每一轮解码操作都能正确的进行。

在每一轮编解码操作中需要固定解码窗口，但是解码窗口的选择并不是一个固定值，需要动态选择一个适合当前网络状况的最优值来完成每一轮的编解码操作。

当 W 很小时，比如 1，为确保解码矩阵可解，发送方的编码包只能包含一个原始包，这时 TCP/NC 协议相当于传统的 TCP 协议，唯一区别是 TCP/NC 数据是以只包含一个原始包的编码包形式发送，接收端要解码。此时不仅没利用网络编码优势，还增加了编解码时间负载，网络性能显然比传统 TCP 更差。

当 W 很大时，接收方接收到编码包会在解码矩阵中保存编码向量，当解码窗口满秩时，所有原始包都能被解码出。但如果解码窗口非常大，解码矩阵需很长时间才能满秩，所有解码和未解码的数据包都会在解码窗口中等待，吞吐率严重下降，另外，大规模矩阵操作增加了计算负载。因此，当窗口非常大时网络性能也很低。

根据以上分析可知，恰当选择 W 与网络性能的提升有直接关系。选择最优窗口 W 不仅能获得更好的吞吐率，而且能降低解码时间，提高网络性能。

3.2 最优解码窗口的存在性分析

编码窗口变化需要与解码窗口的解码速率一致，因此对最优编码窗口的分析可以转化为分析最优解码窗口。下面利用排队论分析最优解码窗口的存在性。

选取无线 Ad hoc 网络中典型的带反馈的链式拓扑结构为系统模型，如图 2 所示。为在图 2 场景下分析发送端队列的长度，作如下假设：链路带宽固定，且对发送端已知，网络负载低于链路带宽，各个节点的缓存能力无限，每个数据包在链路上成功发送的概率为 μ，链路丢包率为 $1-\mu$，时间单位为时间片。数据源产生数据包的过程遵循参数为 λ 的伯努利过程。发送端在每个时间片开始时发送产生的随机线性编码包，忽略传输延迟和编码数据包负载。每个新编码包都能使一个未被看见的数据包被看见。

数据包在源端的停留时间包括在缓存队列的等待时间和等待 ACK 确认时间。发送端缓存区中原始数据包个数和被接收端看见的数据包个数之间的差值可以看作符合 G/G/1 队列。

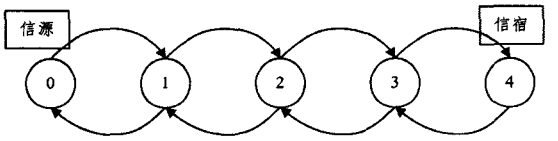

图 2　链式拓扑结构

当链路负载因子 $\rho<1$ 时，马尔科夫链是正常往返的，服从稳态分布：$\pi_k=(1-\alpha)\alpha^k, k\geqslant 0$，其中 $\alpha=\lambda(1-\mu)/(\mu(1-\lambda))$。稳态时队列的平均长度如式(2)所示，数据包在缓存队列里等待发送的平均时间如式(3)所示。

$$E(Q)=\sum_{k=0}^{\infty}k\pi_k=(1-\mu)\frac{\rho}{1-\rho} \tag{2}$$

$$E(T_1)=\frac{(1-\mu)}{\mu(1-\rho)} \tag{3}$$

由于每个时间片数据包在链路上成功发送的概率为 μ，因此这段时间的均值就是 $1/\mu$。因此数据包平均在发送端停留的总时间如式(4)所示，发送端缓存区中的平均数据包数如式(5)所示。

$$E(T)=\frac{(1-\mu)}{\mu(1-\rho)}+\frac{1}{\mu}=\frac{(1-\mu)+(1-\rho)}{\mu(1-\rho)} \tag{4}$$

$$E(N)=E(T)*\lambda=\rho\frac{(1-\mu)+(1-\rho)}{(1-\rho)} \tag{5}$$

当编码窗口的大小不小于编码缓存中的数据包数目时，才能通过编码来掩盖发送过程中可能产生的所有丢包，此外为了使网络编码有意义，编码窗口的大小应该大于等于 2，因此有式(6)，式中 W 是编码窗口大小：

$$W\geqslant\max\left(2,\left\lceil\rho\frac{(1-\mu)+(1-\rho)}{1-\rho}\right\rceil\right) \tag{6}$$

编解码时间会影响 TCP 的往返时间。编码模块主要负载是随机线性编码包的生成。在有限域 GF(256)上，每个随机线性组合的产生需要做 LW 次乘法操作和 $L(W-1)$ 次加法操作，其中 L 是数据包的长度，W 是编码窗口的大小。注意：对于每个数据包，这种编码操作平均要进行 R 次。

在接收端的解码模块中，主要操作是高斯消元法，复杂度为 $O(LW)$，所有操作都是在 LW 个字节的数据上进行的，编码技术所带来的时间总开销是 $O(L^2W^2)$，据此可以推断编码窗口最优值如式(7)所示，与解码窗口大小相同。

$$W=\max\left(2,\left\lceil\rho\frac{(1-\mu)+(1-\rho)}{1-\rho}\right\rceil\right) \tag{7}$$

3.3　TCP/NCW 协议算法

根据最优解码窗口的思想以及存在性分析，编码端和解码端控制算法如下：

I)NC 层发送端：

1)初始化：$W_m=1$，*TX SERIAL NUM*$=0$，*fraction*$=0$（W_m 为编码窗口大小，*fraction* 为冗余因子 R 的小数部分，*TX SERIAL NUM* 为传输序列号）。

2)等待状态：如果以下任何事件发生，则按对应方法处理；否则，继续等待状态。

·事件 1　收到来自 TCP 发送端的数据包：

A)如果这个包是用于连接管理的控制包，直接将其提交给 IP 层，并返回等待状态。

B)如果这个包是一个新的数据包：

a)将其放入编码缓存等待队列中。

b)将等待队列中的数据包移至编码窗口直到窗口达到 W 或者等待队列中没有数据包。

c)令 $R=Re$（Re 为预设的冗余系数），$R+=Fraction$。并重复 i)、ii) 操作$\lfloor R\rfloor$次。

i)*TX SERIAL NUM*$+=1$，产生一个随机线性编码包，添加 NC 层编码头部，包括编码系数向量、参与编码的原始数据包序号、*TX SERIAL NUM* 以及 *Base*（*Base* 是当前存在时间最久的未被 ACK 的原始数据包的序号数），将其发送给 IP 层。

ii)为以后计算 RTT，按 *TX SERIAL NUM* 记录当前发送时间。

d)置 *Fraction* 为原 R 的小数部分，R 为原 R 的整数部分。

e)返回到等待状态。

·事件 2　收到来自接收方的 ACK 确认：

A)从 ACK 头部取下 W、*PREV SERIAL NUM* 和 *Flag*（*PREV SERIAL NUM* 是触发前一个 ACK 的传输序列号）。

B)根据 *PREV SERIAL NUM* 计算 RTT。

C)如果 ACK 带回满秩信息（即 *Flag*$=1$），则清空编码窗口；否则，根据该 ACK 管理编码缓存区。

D)更新 $W_m=W$，返回等待状态。

II)NC 层接收端：

1)初始化：令最优解码窗口 $W=1$，解码矩阵的秩 *Rank*$=0$，等待时间 $T=T0$（$T0$ 是 $W=1$ 时的平均解码时间），$S=$ *Threshold*（*Threshold* 取经验值），*Flag* $=0$。

2)等待状态：如果以下任何事件发生，则按对应方法处理；否则，继续等待状态。

·事件 1　来自 TCP 接收端的 ACK 到达：

A)如果此 ACK 是用于连接管理的控制包，直接将其提交给 IP 层，返回等待状态。

B)否则，根据 *Base* 和 ACK 管理解码缓存区，删除以后不再需要的已解码包，然后忽略这个 ACK，返回等待状态。

·事件 2　来自发送端的编码包到达：

A)获取编码包的编码头部，取出编码系数向量、*TX SERIAL NUM* 和 *Base*。

B)将编码系数向量作为一个新行添加到解码矩阵中，执行高斯消元法识别新的可见包。

C)如果有新的可见包，令 *Rank* $=$*Rank*$+1$，如果 *Rank*$=W$，则令 *Flag* $=1$。

D)产生一个新的 TCP/NC 协议下的 ACK。ACK 头部信息中包含 W、*Flag* 和 *PREV SERIAL NUM*。

E)将编码包数据部分添加到解码缓存区中，执行与系数向量对应的高斯消元操作。若有数据包被解码出，则将其提交给 TCP 接收端。

F)如果 *Flag* $=1$，记录解码时间 t，令 *Rank*$=0$。

a)如果 $t\leqslant 2T$，作如下处理：

·如果 $W\times 2\leqslant S$，令 $W=W\times 2$，$T=T\times 2$；

·如果 $W\times 2>S$，令 $W=W+1$。

b)如果 $t>2T$，令 $W=W/2$，$T=T/2$。

c)返回等待状态。

·事件 3　解码窗口等待编码包超时：

如果由于网络环境导致解码窗口等待编码包超时，则计

算此时解码矩阵的维度 n，将当前解码窗口大小缩减为 n，即令 $W=n$。返回等待状态。

根据 NC 层发送端和接收端算法，编码窗口 W_{en} 一定小于解码窗口 W。对 W 控制的同时可以自然地限制 W_{en}，在解决数据传输和解码操作同步性问题的同时，也限制了编码头部的大小及编解码的复杂度。此外，根据以上算法可知，TCP/NCW 中解码窗口可以根据解码时间自行动态调整窗口大小到一个最优的值，以适应网络环境的变化，这是 TCP/NC 无法做到的。

4 仿真评价

4.1 仿真环境

本文根据文献[8]提出的 TCP/NC 思想及算法，实现了 TCP/NC 协议，并将 TCP/NC 代码放在仿真软件 NS-2.34 相应目录下，makefile 该文件，将其嵌入到 NS 中。在 TCP/NC 的基础上，本文按照上节算法改进协议实现了 TCP/NCW，以同样方式放入 NS 中，以便利用该软件进行仿真验证。下文通过仿真验证了最优解码窗口的存在性，并在不同场景下对 TCP/Vegas、TCP/NC 和 TCP/NCW 协议进行了性能比较。具体仿真环境设置如下。

仿真环境 Linux 操作系统，NS-2.34 仿真软件。环境配置如下：物理层无线信号传播 Two Ray Ground 模型，Mac 层 IEEE 802.11b 协议，无线信道带宽 2Mbps，通过 Mac 层数据包出错的概率来模拟丢包率 p，设丢包服从均匀分布，冗余因子 R 的选择和 TCP/NC 一样，设为 $1/(1-p)$。链路缓存队列 drop-tail 模型，可存储的最大数据包数量为 200。网络层静态场景下使用 NOAH 路由协议，手动设置静态路由表。动态场景下，使用 AODV 协议。传输层选择 TCP/Vegas，接收方的最大通知窗口设为 100 个数据包。应用层使用持续比特流业务 FTP 协议，每个数据包为 1500 字节。仿真拓扑中 $N0-N8$ 代表无线节点，节点接收半径为 250m，间距为 150m。仿真时间为 1000s，所有的仿真结果都是取 10 次运行得到的结果的平均值。

4.2 最优窗口大小存在性验证

本文在无线场景下仿真 TCP/NCW，验证最优窗口的存在性。仿真拓扑结构如图 3 所示，节点 N0 与 N6 通信，带宽 2Mbps，数据包长度 1kB，丢包率 p 为 0.25，按照参数为 0.7 packet/(8L/B) 的泊松过程不断产生数据包，即 $\lambda=0.7$。吞吐率随解码窗口大小变化的仿真结果如图 4 所示。

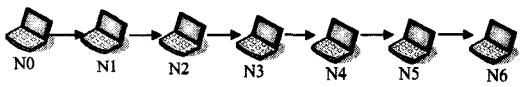

图 3 单一流的级联网络拓扑结构

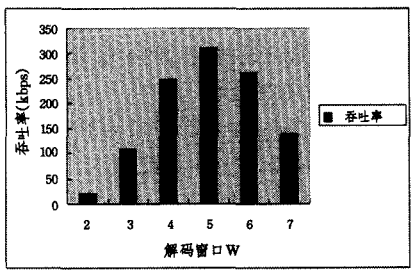

图 4 吞吐率 VS 解码窗口大小

从图 4 看出，吞吐率随着解码窗口不同有显著变化。当解码窗口小于 5 时，吞吐率随着解码窗口变大而增大；当解码窗口为 5 时，吞吐率最大。将理论参数设置为与仿真参数相同时，代入式(7)得最优值 5。其结果与仿真结果相同，验证了最优解码窗口的存在性。

4.3 不同场景下 TCP/Vegas、TCP/NC、TCP/NCW 协议性能比较

1)无线链式拓扑单条数据流场景

为评价 TCP/Vegas、TCP/NC 和 TCP/NCW 协议在不同丢包率下的吞吐率和平均解码时间性能，针对图 3 所示的拓扑结构，采用 4.1 节的仿真参数对 3 种协议进行了仿真，结果如图 5 和图 6 所示。

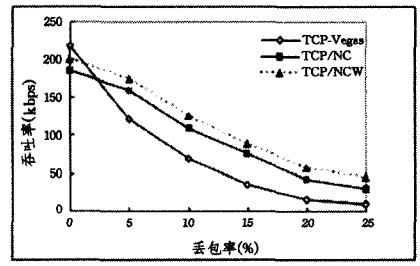

图 5 单一流级联网络的吞吐率比较

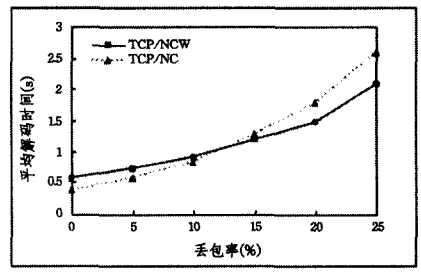

图 6 单一流级联网络的平均解码时间比较

从图 5 看出，3 种协议吞吐率都随着 p 的增大而下降。当 p 小于 2%时，TCP/Vegas 的吞吐率略高于 TCP/NC 和 TCP/NCW。但是随着 p 增长，网络中非拥塞丢包大量增加，TCP/NC 和 TCP/NCW 比 TCP/Vegas 更好。其验结果显示，TCP/NCW 的吞吐率要比 TCP/NC 高出 20%左右。

从图 6 看出，随着 p 的增长，TCP/NC 和 TCP/NCW 协议的平均解码时间也都逐渐增长，但是 TCP/NC 上升趋势要大于 TCP/NCW。当 p 大于 10%后，TCP/NC 的平均解码时间明显大于 TCP/NCW，主要原因在于，当 p 较小时，传播和排队延迟对网络性能带来的影响较小，数据传输和解码操作不同步的问题不明显，解码时间没有太大的区别。但随着 p 增加，由于 TCP/NC 的同步协调能力较差，解码矩阵会由于不同步问题一直增大，从而延长了解码时间，而 TCP/NCW 由于对解码窗口进行控制，解码时间相比 TCP/NC 大大减少。

2)无线链式拓扑两条数据流场景

仿真拓扑结构如图 7 所示。N0 和 N1 分别与 N7 和 N8 进行通信。3 种协议吞吐率及公平性随丢包率变化的仿真结果如图 8 和图 9 所示。

从图 8 看出，当 p 小于 4%时，TCP/Vegas 的吞吐率高于 TCP/NC 和 TCP/NCW；而随着 p 变大后，TCP/NC 和 TCP/

NCW 协议的吞吐率优势也越来越明显，并且 TCP/NCW 的吞吐率一直高于 TCP/NC。

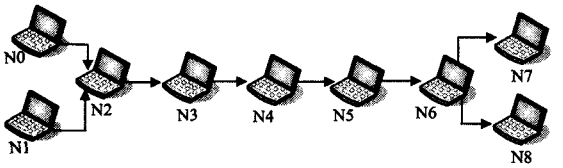

图 7　两个流的级联网络拓扑

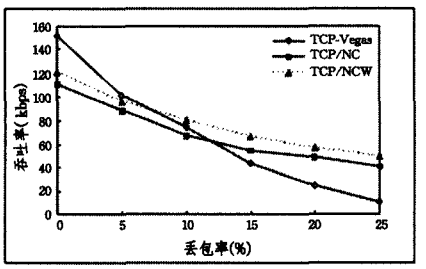

图 8　两个流级联网络吞吐率比较

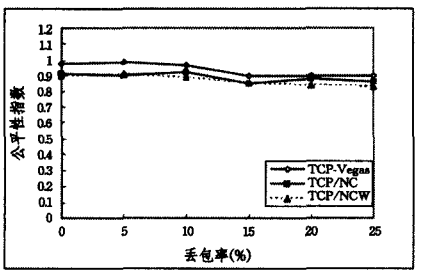

图 9　公平性指数 VS 数据丢包率

在比较吞吐率的同时，本文也比较了 3 个协议的公平性，其中计算公平性索引 FI 的计算公式来自于文献[15]，表达式如式(8)所示。

$$FI=\frac{(\sum_{i=1}^{n}x_i)^2}{n\sum_{i=1}^{n}x^2 i} \tag{8}$$

式中，n 为业务流个数，x_i 是第 i 个业务流的吞吐率。当 $FI=1$ 时，带宽分配最公平，FI 越小公平性越差。3 种协议的公平性随 p 的变化情况如图 9 所示。

可以看出，当 p 小于 25%时，TCP/Vegas 的 FI 值在0.9 到 1 的范围内变化，而 TCP/NC 和 TCP/NCW 的 FI 值在0.8 到 0.92 之间变化。这说明在相同条件下，TCP/NC 和 TCP/NCW 的公平性相对 TCP/Vegas 略差，但仍是可接受的。

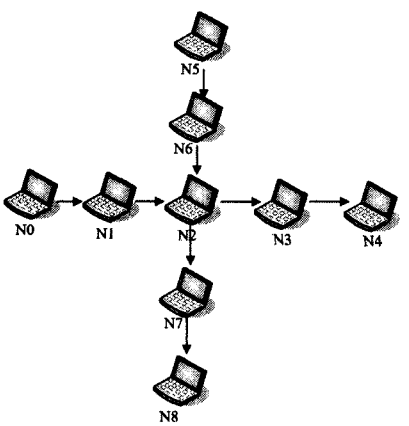

图 10　两个流的交叉网络拓扑

3)两条数据流的交叉网络场景

仿真拓扑结构如图 10 所示。N0 和 N5 分别与 N4 和 N8 通信。3 种协议在不同丢包率下吞吐率比较的仿真结果如图 11 所示。

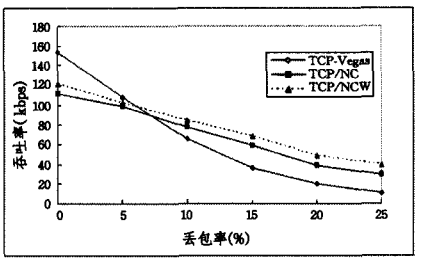

图 11　两个流的交叉网络吞吐率比较

从图 11 可以看出，随着 p 的增加，3 种协议性能都降低。当 p 小于 5% 时，TCP/Vegas 的吞吐率比 TCP/NC 和 TCP/NCW 高些。但是随着 p 的增长，网络中非拥塞丢包大量增加，TCP/NC 和 TCP/NCW 比 TCP/Vegas 更好。实验结果显示，TCP/NCW 的吞吐率要比 TCP/NC 高出 10%左右。

4)3 条数据流的格状网络场景

仿真拓扑结构和数据流的路径如图 12 所示。3 种协议在不同丢包率下的吞吐率比较仿真结果如图 13 所示。

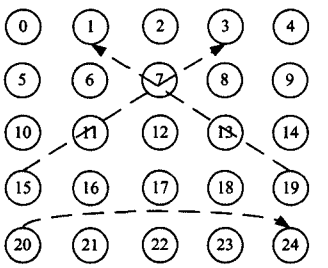

图 12　3 条数据流的格状网络的拓扑

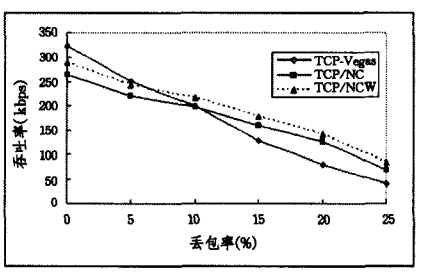

图 13　3 个流格状网络的吞吐率比较

从图 13 看出，随着 p 的增加，3 种协议性能都降低。当 p 小于 5%时，TCP/Vegas 的吞吐率比 TCP/NC 和 TCP/NCW 高。但是随着 p 的增长，网络中非拥塞丢包大量增加，TCP/NC 和 TCP/NCW 比 TCP/Vegas 更好。实验结果显示，TCP/NCW 的吞吐率要比 TCP/NC 高出 15%左右。

以上 4 个场景的仿真实验中，在 p 较小时，TCP/Vegas 的吞吐率都比 TCP/NC 和 TCP/NCW 高。因为此时无线网络丢包现象较少，丢包主要原因是拥塞，TCP/Vegas 能及时对拥塞丢包进行响应，而 TCP/NC 和 TCP/NCW 的优势则在于处理非拥塞丢包。但是随着 p 的增加，TCP/NC 和 TCP/NCW 都比 TCP/Vegas 更好。因为此时网络中非拥塞丢包大量增加，TCP/Vegas 协议错误地调度拥塞处理机制，降低发送速率，导致其吞吐率急剧下降。而 TCP/NC 和 TCP/NCW

协议采用网络编码技术，利用编码信息的冗余和节点的计算能力，掩盖了链路中的丢包，从而避免了调度拥塞处理机制，使吞吐率大幅提升。又由于 TCP/NCW 对解码窗口进行控制，使得编码包进入接收端缓存的速率与 TCP 原始包进入发送端缓存的速率一致，提高了 TCP/NC 的性能，使得吞吐率更高。

5)动态拓扑场景

除了静态场景外，在动态场景下对 3 种协议进行了仿真。该场景在一个 1000m×500m 的矩形区域内设置了 50 个无线节点，分别在 4 种不同平均节点移动速度(1m/s、5m/s、10m/s、20m/s)下进行了仿真。在不同移动速度下 3 种协议的吞吐率比较如图 14 所示。

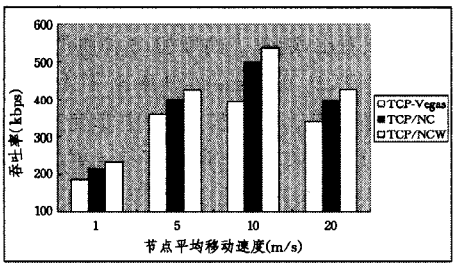

图 14　吞吐率 VS 平均节点速度

从图 14 看出，TCP/NC 和 TCP/NCW 协议在吞吐率上相比 TCP/Vegas 协议均有提高，在 4 种不同移动速度下 TCP/NC 的吞吐率分别提高了 14.5%，10%，26%和 16.4%，TCP/NCW 分别提高了 25.8%，17.1%，35%和 25%。这是因为在动态场景下，对于节点移动所造成的丢包，TCP/Vegas 协议仍错误地进行拥塞控制，造成网络性能下降。而 TCP/NC 和 TCP/NCW 协议采用网络编码技术，避免了错误调度拥塞处理机制，使吞吐率大幅提升，又由于 TCP/NCW 对解码窗口进行控制，确保了数据传输和解码操作的一致性，提高了 TCP/NC 的性能，从而提升吞吐率。

结束语　本文针对 TCP/NC 协议数据传输和解码操作的同步性问题，提出了改进协议 TCP/NCW 和最优解码窗口的思想，并给出了最优解码窗口的选择策略。通过理论分析和仿真说明了最优解码窗口的存在性，并在不同场景下对 TCP/Vegas、TCP/NC 和 TCP/NCW 协议进行了仿真。仿真结果表明，虽然在各个场景下 TCP/Vegas 、TCP/NC 和 TCP/NCW 的吞吐率都随着 p 的增加而下降，但是 TCP/NC 和 TCP/NCW 的下降趋势明显小于 TCP/Vegas，并且 p 越大，TCP/NC 和 TCP/NCW 的优势越明显，甚至在静态场景下，当 p 达到 25%时，TCP/NC 和 TCP/NCW 的吞吐率比 TCP/Vegas 高出 50%～100%，而且也具有很好的公平性。在动态场景下，节点移动成为丢包的主要原因，TCP/NC 和 TCP/NCW 的性能更好。同时，从全部结果来看，TCP/NCW 协议的吞吐率比 TCP/NC 平均提高了 20%，解码时间也显著小于 TCP/NC，说明了最优解码窗口策略的有效性。在接下来的工作中，将对 TCP/NC 和 TCP/NCW 做全面的建模分析，比较两种算法的传输性能，找出进一步的优化方向。

参 考 文 献

[1] Hanbali A A, Altman E, Nain P. A survey of TCP over ad hoc networks[J]. IEEE Communications Surveys & Tutorials, 2005,7(3):22-36

[2] Ahlswede R, Cai N, Li S-Y R, et al. Network information flow [J]. IEEE Trans. Inform. Theory, 2000,46,(4):1204-1216

[3] Yeung R W, Li S-Y, Cai N, et al. Theory of network coding Foundations and Trends[M]. Now Publishers, 2006:241-381

[4] 蒲保兴，杨路明，王伟平. 线性网络编码的导出与扩展[J]. 软件学报，2011,23(3):558-571
Pu Bao-xing, Yang Lu-ming, Wang Wei-ping. The export and expansion of linear network coding[J]. Journal of Software, 2011,23 (3):558-571

[5] 郝琨，金志刚. 一种最小化编码节点的网络编码优化算法[J]. 电子与信息学报，2011,33(2):260-265
Hao Kun, Jin Zhi-gang. A algorithm optimized of network coding to minimize the coding nodes[J]. Journal of Electronics & Information Technology, 2011,33(2):260-265

[6] 俞立峰，杨琼，于娟，等. 防窃听攻击的安全网络编码[J]. 计算机应用研究，2012,29(3):813-818
Yu Li-feng, Yang Qiong, Yu Juan, et al. A secure network coding which could prevent eavesdropping attacks[J]. Application Research of Computers, 2012,29(3):813-818

[7] Fragouli C, Le Boudec J Y, et al. Network coding: an instant primer[J]. ACM SIGCOMM Computer Communications Review, 2006,36(1):63-68

[8] Sundararajan J K, Shah D, Medard M, et al. Network coding meets TCP [OL]. http://adsabs. harvard. edu/abs/2009arxiv0908. 1564s

[9] Sundararajan J K, Jakubczak S, Medard M, et al. Interfacing network coding with TCP: an implementation [OL]. http://adsabs. harvard. edu/abs/2009arxiv0908. 1564s

[10] Sundararajan J K, Kim M, M'edard M, et al. Modeling Network Conded TCP Throughput: A simple Model and its Validation [OL]. http://arxiv. org/pdf/1008/0420. pdf

[11] Zhang H, Yu W, Wu C, et al. Self-adaptive Scheme to Adjust Redundancy for Network Coding with TCP[M]// Computer Engineering and Technology. Springer Berlin Heidelberg, 2013:81-91

[12] Wu C, Zhang H, Yu W, et al. Self-adaptive Retransmission for Network Coding with TCP[M]// Advanced Parallel Processing Technologies. Springer Berlin Heidelberg, 2013:396-407

[13] Medina-Ruiz H J, Kieffer M, Pesquet-Popescu B. An Adaptive Redundancy Scheme for TCP with Network Coding[C]// Proceedings of the IEEE International Symposium on Network Coding. 2012:1-6

[14] Li S-Y, Yeung R, Cai N. Linear network coding [J]. IEEE Transactions on Information Theory, 2003,49(2):371-381

[15] Pilosof S, Ramjee R, Raz D, et al. Understanding TCP fairness over wireless LAN[C]// Proc. Annual Joint Conference of the IEEE Computer and Communications Societies. CA, 2003, 2: 863-872