第32卷 第19期 计 算 机 工 程 2006年10月
Vol.32 № 19 Computer Engineering October 2006
·网络与通信· 文章编号：1000—3428(2006)19—0120—03 文献标识码：A 中图分类号：TP393

大型数据的编码传输技术研究

李钟华[1,2]，李伟华[2]
(1. 江西财经大学信息管理学院，南昌 330013；2. 西北工业大学计算机学院，西安 710072)

摘 要：提出利用信息学编码理论中的线性纠错码 LECC 来改善当前计算机网络中进行大型数据传输时存在速度慢、可靠性低等问题。线性纠错码方法对数据进行分块冗余编码，在有损信道(如 Internet)上传输编码块。接收端只要接收到足够数量的编码包，就可解码出初始数据信息，无需反馈信道，减少包应答及丢失包重传的时间。实验结果表明，LECC 编码传输平均只要接收到比源数据包多 4%的编码包即可完成解码。对于大型文件传输，编解码及冗余包的传输所增加的负载比传统差错控制小，有效地提高了信道的可靠性及传输效率。
关键词：LECC 编码；差错控制；有损信道；编码传输

Research on Encoding Transmission of Big Data File

LI Zhonghua[1,2], LI Weihua [2]
(1. School of Information Technology, Jiangxi University of Finance & Economics, Nanchang 330013;
2. School of Computer Science, Northwestern Polytechnic University, Xi'an 710072)

【Abstract】This paper uses linear error correction code(LECC) of the informatics encoding theoretics to solve the problems of slowness and low reliability on transmission of big data file. LECC algorithm partitions the data into blocks and encodes it redundantly, then the encoded packages are transmitted in the erasure channel (Internet for example). If enough encoded packages are received, the receiver can decode the original data, regardless of which package is lost. It saves the time for package acknowledging and retransmitting. In the experiment, the receiver can finish decoding as long as it gets 4 percent encoded packages more on average in LECC transmission, the cost of encoding, decoding and the increased packages transmitting is less than the ARQ error control's in transmission of big data file. LECC algorithm improves the reliability of channel and the efficiency of data transmission.
【Key words】LECC Code; Error Control; Erasure Channel; Encoding Transmission

1 概述

企业间日益紧密的网络化协作及远程异地网络灾备的兴起，对快速高效的数据共享及传输提出更高的要求，大型数据的网络传输瓶颈问题日渐显露。Internet的传输问题研究结果表明，目前在Internet上传送数据，当考虑组播情况下的数据报的丢失概率时，一个发送者同时向 11 个地理位置分散的接收端发送同样的数据，在一个小时之内，每个接收端的数据报的平均丢失概率为 9.3%，同时有 46.5%的数据报曾经被一个或一个以上的接收端所丢失。传统的丢包问题处理采用自动重发请求(Automatic Repeat request, ARQ)方式[1]，ARQ 方式有简单、信元速率可控、对信道干扰的变化不敏感等优点。但它有一系列的缺点：(1)必须有一个反馈通道；(2)只能用在点对点的单一通信方式当中，不能实现一对多广播或组播通信方式；(3)如果信道干扰大，则系统经常处于反馈一重发状态，效率降低，不能连贯工作。ARQ在处理大时延系统时需要反馈信道来传送重传请求，必须有 2 个来回的时间(RTT)来完成一次重传请求，从而有较高的时延。在广播传输情况下，若每个接收端都随机地发出重传的请求时，常常会由于重传请求的增多而引起所谓的反馈拥堵现象，从而使传送受到限制甚至停止。这些缺陷使ARQ方式在大型数据传输上有很大限制，其浪费的传输资源是非常可观的。

当前寻求解决网速慢和可靠性低的办法，已经成为学术界的研究热点，信息理论[2]，特别是编码理论[3]，在网络传输的速度和可靠性上有了很大的提高。在编码理论中有许多优异的算法可以利用，特别是号称最佳码的RS码[4]，在处理小数据量时，效果很好。但由于其编解码时间与数据量呈平方关系，在大型数据传输中，编解码的消耗是不可容忍的。

针对以上问题，我们提出线性纠错码(Linear Error Correction Code, LECC)方法来保证大型数据的传输问题。LECC 方法对需传输的数据信息进行编码，然后发送所得到的编码包，接收端只要接收到足够数量的编码包，就可解码出初始数据信息，自行纠正出现的差错，无需反馈信道。且不管哪些数据包丢失了，接收到哪些信息包，信息包到达的次序，甚至有多少信息包丢失都并不重要，接收端不需要对数据包进行应答，有效地解决了信息包丢失或受到破坏等问题，提高了传输的效率和稳定性。

2 线性纠错码 LECC

纠错传输技术对需传输的数据信息进行编码，每一编码块组成了原始信息块的一个随机子集，有关每个原始信息包的信息冗余被散布在一些编码信息包上，通过传输冗余信息来防止信息丢失和进行差错控制，无需向发送端传递反馈信息。纠错传输技术在大型数据传输领域及广播或组播传输中，具有极大的优势。我们提出的 LECC 编码方法是利用图论中

基金项目：国家“863”计划基金资助项目(2003AA142060)
作者简介：李钟华(1976－)，男，博士生，主研方向：网络信息安全；李伟华，博导
收稿日期：2005-10-11 **E-mail：**lzh_nwpu@sina.com

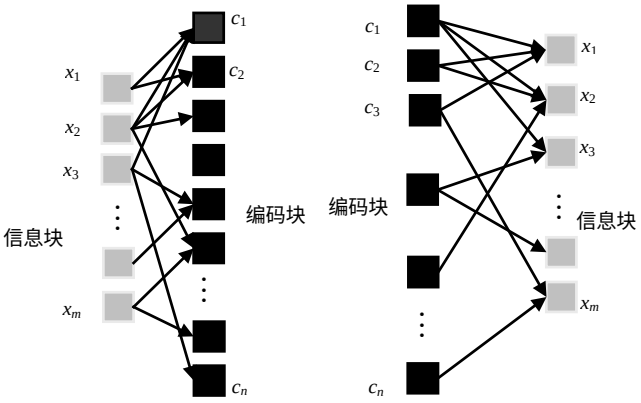
二分图相关知识和二进制数的 XOR 运算而得到的一种纠错编码方法，具有快速的编解码速率和高效的数据恢复效率。

对于一个长度为M的数据文件，首先对其按一定长度l进行分块，l可任意选择，但考虑到网络传输中IP碎片问题，一般选择l的大小略低于一个IP包的长度，因此源数据文件可被分为$m=M/l$ 信息块。信息块标记为$x_1,x_2,\ldots,x_m$，对$x_1,x_2,\ldots,x_m$进行冗余编码得到n个编码块，编码率 $\varepsilon=n/m$，编码块记为c_1, $c_2,\ldots,c_n$。这里，

$$\begin{cases} c_1 = x_{11} \oplus x_{12} \oplus \ldots \oplus x_{1d_1} \\ c_2 = x_{21} \oplus x_{22} \oplus \ldots \oplus x_{2d_2} \\ \cdots \\ c_n = x_{n1} \oplus x_{n2} \oplus \cdots \oplus x_{nd_n} \end{cases} \tag{1}$$

其中$1 \le d_i \le m$。

LECC 编码方法中的有关生成编码信息和利用编码信息来修复丢失了的原始信息的过程利用图的方式来表达，可以得到如图 1，其中图 1(a)表示编码过程，图 1(b)表示解码过程。



(a)编码过程　　(b)解码过程

图 1 用二分图表示的线性纠错码的编、解码过程

图 1 中，二分图 G 为含有 m 个左节点和 n 个右节点，左右节点之间的连接数不限制，m 个左节点分别代表 m 个源数据块，n 个右节点则表示了由 m 个左节点生成的所有编码数据块。

3 LECC 编码设计

由图 1 可知，LECC编码对初始信息块进行XOR运算，生成包含初始信息块冗余信息的编码信息块，实现信息的编码冗余传输。LECC编解码设计的过程就是构建图 1 的二分图的过程。图 1 中顶点c_i邻接的边的数量称为该顶点的度(Degree of Node)，如c_1的度为 3，c_2的度为 2。每一个编码块c_i通过以下方式由源文件$x_1x_2x_3\ldots x_m$产生，编码过程如下：

(1)编码块c_i在度分布 $\rho(d)$ 下随机选择度d_i；

(2)从$x_1,x_2,\ldots,x_m$中随机均匀选择d_i个不同的初始信息包 $x_{i1}, x_{i2}, ..., x_{id_i}$，计算

$$c_i = x_{i1} \oplus x_{i2} \oplus \ldots \oplus x_{id_i} \tag{2}$$

(3)输出编码数据包$\{c_i,d,(i_1,i_2,\ldots,i_d)\}$。

因此，各个编码包都独立于其他编码块，恢复过程只须关心编码包的数量，而无需在意具体哪些编码数据包丢失了。

编码块 c 的度概率分布 $\rho(d)$ 设计是 LECC 设计的关键部分，$\rho(d)$ 设计须保证图 1(a)的所有信息节点都有边连接(即 $\delta \to 0$)，而且须保证图 b 的解码过程可顺利完成。$\rho(d)$ 设计还制约着编解码效率。$\rho(d)$ 的设计目标如下：

(1)用于成功恢复所有初始信息的平均编码节点数尽可能小。即数据传输中要传输的冗余数据尽可能少，提高数据传输的效率；

(2)编码节点的平均度尽可能小。平均度的数量与 LECC 的编解码复杂度相关，平均度小，用于编解码的时间就少，LECC 传输算法在数据传输中效率越高；

LECC 中采用改进的 Soliton 概率分布：

$$\rho(d) = \frac{\omega(d) + \tau(d)}{\beta}$$

其中，

$$\omega(d) = \begin{cases} 1/m & d = 1 \\ \dfrac{1}{d(d-1)} & d = 2, \ldots, m \end{cases}$$

$$\tau(d) = \begin{cases} \dfrac{R}{m}\dfrac{1}{d} & d = 1, \ldots, m/R - 1 \\ \dfrac{R}{m}\ln(R/\delta) & d = m/R \\ 0 & d = m/R + 1, \ldots, m \end{cases}$$

$\beta = \sum_d (\omega(d) + \tau(d))$

$R = a \cdot m \ln(m/\delta)$，$a$ 为大于 0 的常数。

基于度分布 $\rho(d)$ 的 LECC 编码算法要解码 m 输入信息包所需的编码包数 n 为

$$n = m \cdot \sum_d (\omega(d) + \tau(d)) = m + \sum_{d=1}^{m/R-1} \frac{R}{d} + R\ln(R/\delta)$$

$$\le m + R \cdot H(m/R) + R \cdot \ln(R/\delta) = m + O(\sqrt{m} \cdot \ln^2(m/\delta))$$

每个编码节点的平均度 $\overline{d}$ 为

$$\overline{d} = \frac{\sum_d d \cdot (\omega(d) + \tau(d))}{\sum_d (\omega(d) + \tau(d))} \le \sum_d d \cdot (w(d) + \tau(d))$$

$$= \sum_{d=2}^{m+1} \frac{1}{d-1} + \sum_{d=1}^{m/R-1} \frac{R}{m} + \ln(R/\delta) \le H(m) + 1 + \ln(R/\delta)$$

$$= O(\ln(m/\delta))$$

因此，按照 LECC 编码算法编码生成的纠错码，如果初始数据含有 m 个输入包，平均经过 $O(\ln(m/\delta))$ 步异或运算产生一个编码数据包。编码数据包在不可靠信道上传输到接收端后，如果信道以一定的出错概率将某些信息元变换了，在接收端接收到 $n = m + O(\sqrt{m} \cdot \ln^2(m/\delta))$ 编码数据包后，就一定能够以 $1-\delta$ 的概率使得解码过程正常结束，并且整个解码过程只需要 $O(n \cdot \ln(m/\delta))$ 步异或运算即可完成。

4 LECC 解码过程

LECC的解码过程中，我们以图中校验信息元c_1为例，$c_1 = x_1 \oplus x_2 \oplus x_3$。如果在传输过程中，原始数据$x_3$丢失，而我们正确接收到了$x_1$、$x_2$和$c_1$，就可以利用异或加法本身的特点，得到：$x_3 = x_1 \oplus x_2 \oplus c_1$，从而将$x_3$修复。

由式(1)可知，若知道编码块的度及其邻居节点集，就可以构建出图 1(b)的解码二分图，解码器就可以迭代使用以下规则来恢复源信息块：

解码恢复规则 在图 1(b)中，如果至少 1 个编码块只有 1 个邻居(度 d=1)，则可直接恢复其对应的初始信息块。若已恢复的输入信息块还作为其它编码块的一个邻居，则将其从这些编码块的邻居中移除(即删除连接的边)，同时将对应的编码块度减 1，从而简化了二分图，继续找 d=1 的编码块，重复以上过程，直到恢复过程结束。

由解码恢复规则，LECC 算法的解码过程如下：

(1)寻找出一个编码节点c_i，c_i仅有一个初始信息节点x_j与之相连 (若找不到这样的编码节点，则解码终止)；

(2)恢复初始信息节点x_j，即$x_j=c_i$；

(3)通过异或(XOR)运算，把x_j加到所有与x_j有边相连的编码节点中；

(4)移除所有与x_j相连的边，简化二分图；

(5)重复以上过程，直到$\{x_j\}$完成解码恢复。

在 LECC 算法的解码迭代过程中，若步骤(1)解码终止时，解码二分图中还有左节点，则解码失败。接收端请求接收新的编码包，使解码过程继续下去，直到所有左节点都恢复，解码完成。

5 LECC 编解码性能仿真测试

在 LECC 算法中，解码过程采用贪婪算法，在线实时地恢复初始数据包，即解码器每接收到一个编码数据包，就把它加入到解码二分图中，同时进行解码处理，图 2 为解码过程中，对于 m=10 000 的初始信息文件，接收到的编码数据包与已恢复的初始数据包的函数关系。从图中可以看出，LECC 编码算法在接收到大约初始数据包数量的编码包后，可解码恢复的初始数据包的数量是非常少的。在接收到大约 K=10 400 编码包后，解码恢复的数据包一下子从不到 10%上升到 100%。因此，LECC 编码算法的编码率为 4%。

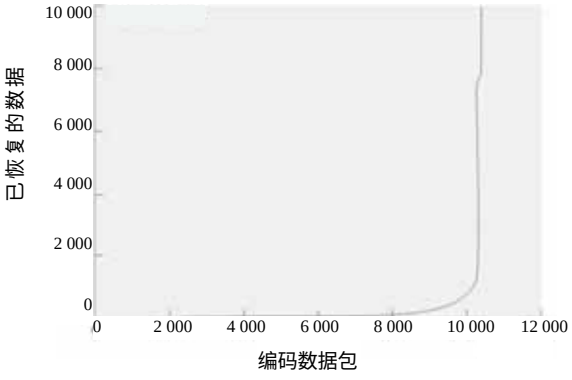


图 2 LECC 编码的解码性能

6 LECC 在大型数据传输中的应用

LECC 编码传输具有编码速度快、编码效率高、传输无需反馈信道等特性，在大型数据传输中具有很大的应用前景，可极大地提高恶劣传输环境(高网络工作负载或带宽攻击)下的传输效率及可靠性。

下面使用一个例子来进行分析说明，假定某公司有一个 200MB 的重要数据文件要发送到该公司在世界各地的子公司及合作伙伴。同时假设由于网络速率不同，平均下载速度为 56kbps，那么，若在传输过程中没有错误发生的情况下，整个文件传输完毕需 1h。若考虑在网络中会有 4%的出错概率下，使用 ARQ 差错控制方式，发送方每发送一个信息包，就必须接收到一个反馈信息报，当发生错误时，发送方将重新发送，这样整个过程中需要发送的数据量将是 1.04 倍，加上传输应答等待时间，整个过程中需要的时间将超过 1.04 倍。采用 LECC 编码传输方式时，不管网络的出错概率多大，其发送的数据量皆为 1.04 倍，且由于不需反馈应答，整个过程中需要的时间小于 1.04 倍。而且在该例子中，需要对多点进行文件传输，对于 ARQ 方式来说，网络上将需要发送大量的反馈信息，这将增加服务器的负担，同时有可能导致服务器崩溃的危险。

本例中对多点进行文件传输时，由于 LECC 编码算法产生的编码包相互独立，恢复过程只需关心编码包的数量，而不需要在意接收到的是哪些编码数据包，且无需反馈信道。反馈信道的带宽被解放出来，各接收方在接收数据的同时，可以利用反馈信道的带宽给其它接收方传送相异的编码包，大大降低发送方的工作量，提高传输的效率。

另外，LECC 编码传输在大型文件多路下载上也有很大的应用。众所周知，从一个严重超载的服务器或经过一个高度拥塞的线路上下载一个巨大文件，对于下载者的耐心是一个极大的考验。在大型文件的发布中，常用的方法是在许多地方设立多个镜像点，需要发送的文件在各个镜像点上保留拷贝。LECC 编码传输使得下载可同时从各个数据点获得编码包，最后只要收集到足够数量的编码包就可解码出原始数据信息，完成文件的下载。

综上，LECC 编码算法在大型文件传输中有很大的应用前景，特别是在丢包率高的信道上，LECC 传输更是显示出其优越性能。

7 结论

LECC 编码在有损信道上传输大型文件时有很大的优势，这种技术以软件方式实现，速度非常快，编码恢复原始数据所需的数据开销只占原始内容的 4%。LECC 编码技术是解决可靠内容传送问题的一种创新方式。采用 LECC 编码传输方式时，发生信息包丢失或受到破坏等问题时带来的影响很小，重传数据或对原始信息包增加特大冗余以补偿丢失的信息包的需要减至最小或完全消除，很好地提高了大型数据传输的效率及可靠性。另外，LECC 算法在一对多、多对一、多对多的数据传输中，由于 LECC 编码包的相异无差别特性，数据包可相互交换，极大地提高传输速度。

参考文献

1 张秀群. 信息传输基础与应用[M]. 北京: 电子工业出版社, 2005.

2 博 斯, 武传坤. 信息论、编码与密码学[M]. 北京: 机械工业出版社, 2005.

3 McEliece R J. The Theory of Information and Coding(Second Edition)[M]. Cambridge University Press, 2002.

4 Cipra B. The Ubiquitous Reed-Solomon Codes[J]. SIAM News, 1993, 26(1).