

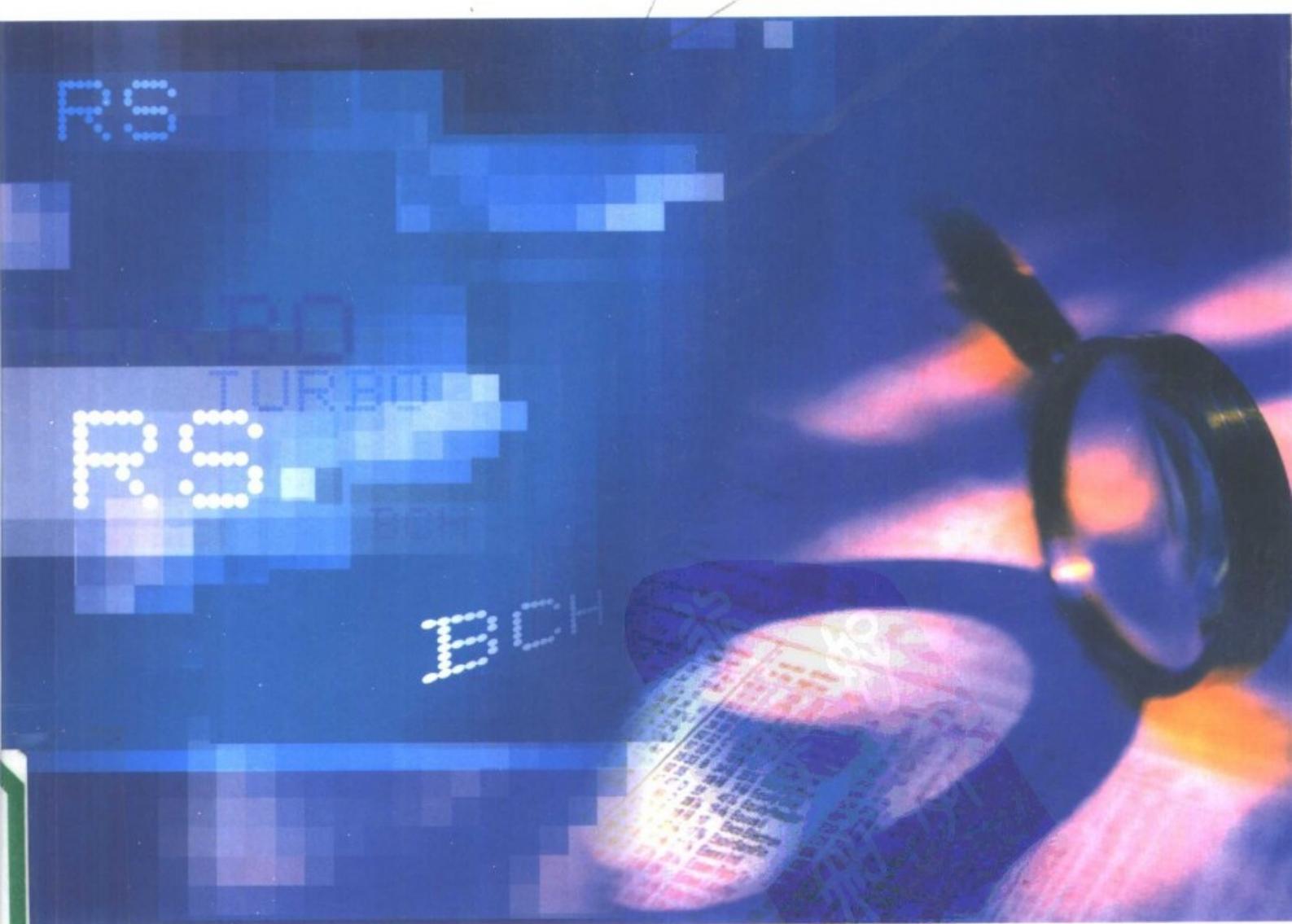


高等 学 校 通 信 类 系 列 教 材

纠错码——原理与方法

(修订版)

□ 王新梅 肖国镇 编著



西安电子科技大学出版社
[http:// www.xdph.com](http://www.xdph.com)

21世纪高等学校通信类系列教材

纠 错 码

—原理与方法

(修订版)

王新梅 肖国镇 编著

西安电子科技大学出版社

2001



内容简介

纠错码是一门新的差错控制技术，目前已广泛应用于各种通信系统和计算机系统中。本书着重阐述纠错码的基本原理和各种编、译码方法。全书共分十三章。前九章介绍各种线性分组码（如循环码、BCH 码、RS 码、不等保护能力码和代数几何码等）的基本原理和必要的数学基础，还介绍了各种实用的编、译码技术和方法。后四章介绍卷积码的基本概念以及代数译码和概率译码的方法和 Turbo 码。全书对材料的阐述循序渐进；在内容上，既要有必要的数学基础，又着重于物理概念的解释；在每章之后均附有习题。本书可作为高等学校本科生、研究生的教材和参考书，也可作为从事通信、计算机等领域中工程技术人员的参考书。



序

提高信息传输的可靠性和有效性，始终是通信工作所追求的目标。纠错码是提高信息传输可靠性的一种重要手段。它已经历了 40 年的历史，在此期间有了很大的进展。

伴随着信息时代的到来以及微电子技术的飞速发展，今天纠错码已不再单纯是一个理论上探讨的课题了，它已成为一门标准技术而被广泛采用。在通信领域中，CRC 校验已成为 CCT 对各类线路传输建议中必不可少的一部分；在移动通信中，纠错码被广泛用于模拟体制的信令传输及数字体制的整个传输，以提高传输的可靠性和节省珍贵的频谱资源；在卫星通信中，纠错码技术已成为用来降低对高功放的要求和减少地球站天线孔径的尺寸的经济可靠的方法，VSAT 和 USAT 的兴起，都是和纠错码技术的应用有关的；在电话网上的数据传输中，纠错码、差错控制技术已是使高速数据传输(9.6 kbit/s 以上的数据率)成为现实的关键技术。纠错码技术还广泛应用于计算机存贮和运算系统中，此外，纠错码技术还应用于超大规模集成电路(VLSI, ULSI)设计中，以提高集成电路芯片的成品率，降低芯片的成本。

因此，纠错码已不再是致力于专门研究的专业人员才应掌握的一门学科，而成为从事通信、计算机、电子系统工程的有关工程技术人员都必需掌握的一门技术。纠错码的内容十分丰富，涉及的领域亦很广，很需的数学知识较多、较深，为此，应在大学本科生及研究生中开设纠错码课程，而实施纠错码教学的关键在于有一本好教材。西安电子科技大学的王新梅教授、肖国镇教授编著的《纠错码——原理与方法》一书正是适应这一需要的一本很好的专著。

西安电子科技大学是我国最早从事纠错码研究工作的基地之一，在纠错码发展第一阶段的后期，即 60 年代初，西电就系统地进行了这方面的研究工作，30 年来锲而不舍。本书的作者及王育民、梁传甲等教授带领一批青年学者在这一方面做出了不少贡献，并与国际上编码学界有着广泛的联系与交流，在科研与教学方面也积累了丰富的经验，正如作者在本书的前言中所述，本书可以说是编码讨论班的集体结晶。

尽管国内外在纠错码方面已出版了相当多的著作，但本书吸收了各家之长而有其显著的特色。本书对材料的选择和问题的阐述循序渐进，在内容上既有必要的数学证明又着重于物理概念的解释，内容新颖、全面、系统，并反映了纠错码理论的最新成果，贯彻了理论与实际相结合的原则。相信本书的出版，将对纠错码的普及与提高起到积极的推动作用。

值此岁暮之际，我谨向充满团结、友谊、进取精神的西电信息论编码集体祝贺他们所取得的丰硕成果，并预祝他们今后为我们的祖国做出更大贡献。

陈太一

1991 年 12 月 25 日

于北京雁栖湖畔

前　　言

1948 年香农(Shannon)在他的开创性论文“通信的数学理论”中，首次阐明了在有扰信道中实现可靠通信的方法，提出了著名的有扰信道编码定理，奠定了纠错码的基石。自此以后汉明(Hamming)、斯列宾(Slepian)、普兰奇(Prange)等人在 50 年代初，根据香农的思想，给出了一系列设计好码和有效译码的方法。以后，纠错码受到了越来越多的通信和数学工作者，特别是代数学家的重视，使纠错码无论在理论还是在实际中都得到飞速发展。

迄今，纠错码已有 40 年的历史，其发展过程大致分以下几个阶段。

50 年代至 60 年代初，主要研究各种有效的编、译码方法，奠定了线性分组码的理论基础；提出了著名的 BCH 码编、译码方法以及卷积码的序列译码；给出了纠错码的基本码限；还出版了纠错码的第一本专著。这是纠错码从无到有得到迅速发展的年代。

自 60 年代至 70 年代初，这是纠错码发展过程中最为活跃的时期。不仅提出了许多有效的编译码方法，如门限译码、迭代译码、软判决译码和卷积码的维特比(Viterbi)译码等。而且注意到了纠错码的实用化问题，讨论了与实用有关的各种问题，如码的重量分布、译码错误概率和不可检错误概率的计算、信道的模型化等，所有这些问题的研究为纠错码的实用打下了坚实基础。在此期间，以代数方法特别以有限域理论为基础的线性分组码理论已趋成熟。

70 年代初至 80 年代，这是纠错码发展史中具有极其重要意义的时期。在理论上以戈帕(Goppa)为首的一批学者，构造了一类 Goppa 码，其中一类子码能达到香农在信道编码定理中所提出的码——香农码，所能达到的性能，这在纠错码历史上具有划时代意义。在这期间大规模集成电路和微机的迅速进展，为纠错码的实用打下了坚实的物质基础，因而与实用相关的各种技术及有关问题得到了极大关注，并在实用中取得了巨大成功。如美国在 70 年代初发射的“旅行者”号宇宙飞船中，成功地应用了纠错码技术，使宇宙飞船在极其遥远的距离(30 亿公里)，向地面传回了天王星、海王星等星体的天文图片，发现了天王星的 9 个卫星和光环以及海王星的 6 个卫星和光环等许多极其宝贵的资料。若不应用纠错码，这些成就的取得是不可想象的。应当指出，在此期间利用 FFT 技术，从频谱观点研究纠错码，受到了特别重视，使得很多熟悉信号处理技术但不熟悉有限域理论的工程师们，能够较快地掌握纠错码理论，并能熟练地应用于实际中，从而为纠错码在各类通信系统中的广泛使用，起到了极好的推动作用。

自 80 年代初以来，戈帕等从几何观点讨论分析码，利用代数曲线构造了一类代数几何码。在这些码中，某些码的性能达到了香农码所能达到的性能。由于代数几何码是一类范围非常广的码，在理论上已证明它具有优越的性能，因而一开始就受到了编码理论工作者，特别是代数几何学家的重视，使代数几何码的研究得到了非常迅速的进展，取得了许多成果。现在，代数几何码的研究方兴未艾。

目前，利用纠错码降低各类数字通信系统以及计算机存贮和运算系统中的误码率，提高通信质量，延长计算机无故障运行时间等，在美国等西方国家中已作为一门标准技术而

广泛采用。而且纠错码技术还应用于超大规模集成电路设计中，以提高集成电路芯片的成品率，降低芯片的成本。不仅如此，自70年代末以来，纠错码技术已开始渗透到很多领域。利用纠错码中的许多编、译码原理和方法，与通信系统中的其它有关技术相结合，得到了令人惊喜的结果。例如，纠错码与调制技术相结合所产生的TCM技术，已作为国际通信中标准技术而推广使用。又如纠错码与密码相结合，可以构造出一类既能加密、签名，又具有纠、检错功能的密码系统；纠错码与信源编码相结合的结果，使得通信系统更为有效与可靠。不仅如此，纠错码中的许多译码思想和方法，与神经元网络中的能量函数有密切关系，纠错码中的许多译码技术，可以用来解决神经元网络中的一些问题。因此，可以预料，随着科学的进步和实际的需要，纠错码理论必将进一步发展，它的应用范围必将进一步扩大。

本书共分三部分。第一部分介绍学习纠错码所必需的数学基础，包括第二、四章和第八章的前一部分。这部分内容介绍了群、环、域、格、线性空间和矩阵等的基本概念，详细讨论了有限域的性质和构造，以及代数几何的基本知识。这一部分内容也可以作为一门基础课（约3学分）单独讲述。第二部分包括第一、三章及第五章至第九章的内容。这部分介绍了线性分组码的基本概念和主要码类，如：汉明码、循环码、BCH码、RS码、不等保护能力码和代数几何码等的基本原理与构造方法。这部分还介绍了各种译码方法如：最小汉明距离译码、捕错译码、大数逻辑译码、迭代译码、频域译码和软判决译码等。第二部分是本书的主要内容。第三部分包括第十章至第十二章，主要介绍卷积码的基本概念及其有关的译码方法，如大数逻辑译码、维特比译码和序列译码等，并讨论了编码与调制相结合的TCM技术。第二、三部分内容是纠错码的核心（约需3~4学分）。

纠错码的内容非常丰富，涉及的领域较广，所需的数学知识较多、较深。由于篇幅所限，本书不可能详细介绍所有内容，仅讨论纠错码理论中比较基本和重要的，并在实际中用得较多的各种码的编译码原理和方法。

本书对材料的选择和问题的阐述循序渐进，在内容上既有必要的数学证明，又着重于物理概念的解释。内容比较新颖、全面和系统，并力求反映出纠错码理论的最近成果。全书中有大量的各种最佳码表，以便读者选择应用。每章后都附有习题，以便读者加深对内容的了解和掌握各种编译码方法。书中有*号的章节为较深或较新的内容，可视情况选读。

本书第十三章由孙蓉同志编写，第八章由肖国镇同志编写，其余各章均由王新梅同志编写。

作者衷心地感谢我们的老师陈太一教授和胡征教授。早在60年代初，他们领导的信息论和编码讨论班上，不仅为年青一代讲授了当时最新的科学知识，把我们带入了信息论这一领域。而且，他们所提倡的那种自由争论的学术气氛和民主、平等的学术环境，使当时作为学生和讨论班成员的第一作者终身难忘，并深刻地影响了作者以后的工作和发展方向。

在70年代初、中期，由王育民、梁传甲、顾慰文和尹克震等教授以及我们一起组成的编码和密码讨论班上，共同经历了风风雨雨的几年，在那动荡、压抑的年代编写了几套讲义。其中《纠错编码讲义》比较全面系统地收集了当时最新的研究成果。文革后，在此讲义基础上，并参考了当时美国夏威夷大学林舒（S. Lin）教授和科斯特洛（J. Costello）教授合著

的《差错控制编码：基础和应用》一书初稿中的内容，于 1981 年由第一作者编写出版了《纠错码》讲义，作为我校研究生和有关专业学生的教材。根据近十年教学实施的经验，以及纠错码的最近成果，对该讲义进行了补充、修改和删节编写了本书。因此，可以说本书是编码讨论班的集体结晶。

我们要特别感谢近几年来编码和密码讨论班中的所有老师和研究生们，特别是王育民教授和梁传甲教授。在这个温暖的集体中，我们不仅共同学习，讨论最新的成果、理论和方法，相互交流学术思想和观点，而且相互鼓励与支持，使我们在精神上得到了极大的鼓舞，谨以此书献给该集体的所有同志们。

此外，我们还要感谢关心该书出版的美国 Lehigh 大学的曾开明(K. M. Tzeng)教授、总参 57 所的宋国文总工程师以及西南交通大学的靳蕃教授和所有其他同志们。对吉筮琴工程师的大力支持与鼓励，以及细心、认真负责地抄写该书的全稿，表示深切谢意。

最后，我们还要特别感谢西安电子科技大学出版社和本书的责任编辑，在出版经费非常紧张的艰难时刻，大力支持本书的出版，并认真细致地编辑，使本书能顺利地与读者见面。

由于作者水平有限，错误遗漏在所难免，恳请读者批评指正。

本书得到国家自然科学基金的资助。

作 者
1991 年 2 月
于西安电子科技大学

目 录

第一章 纠错码的基本概念

| | |
|-----------------------------|----|
| § 1.1 数字通信系统的组成及信道模型 | 1 |
| § 1.2 差错控制系统和纠错码分类 | 5 |
| § 1.3 最大似然译码和纠错码的基本概念 | 7 |
| § 1.4 信道编码定理..... | 13 |
| 参考文献 | 16 |

第二章 代数初步

| | |
|------------------------|----|
| § 2.1 整数的一些基本知识..... | 17 |
| § 2.2 群和格的基本概念..... | 25 |
| § 2.3 环与域的基本概念..... | 30 |
| § 2.4 子群、正规子群和商群 | 32 |
| § 2.5 子格与划分..... | 36 |
| § 2.6 线性空间和矩阵..... | 38 |
| 习题 | 50 |
| 参考文献 | 51 |

第三章 线性分组码

| | |
|--|----|
| § 3.1 线性分组码的基本概念..... | 52 |
| § 3.2 码的一致校验矩阵与生成矩阵..... | 54 |
| § 3.3 伴随式与标准阵列及其它译码..... | 59 |
| * § 3.4 线性码的覆盖半径 | 65 |
| § 3.5 由一个已知码构造新码的简单方法..... | 67 |
| * § 3.6 用多个已知码构造新码的方法 | 70 |
| § 3.7 线性码的重量分布与译码错误概率..... | 73 |
| § 3.8 线性码的纠错能力..... | 79 |
| * § 3.9 不等保护能力线性分组码 | 83 |
| * § 3.10 纠非对称、单向错误及 t -EC/AUED 码 | 89 |
| 习题 | 97 |
| 参考文献 | 99 |

第四章 多项式环与有限域

| | |
|--------------------------------|-----|
| § 4.1 子环与理想 | 101 |
| § 4.2 多项式剩余类环 | 103 |
| § 4.3 循环群 | 113 |
| § 4.4 有限域(Galois 域)的乘法结构 | 117 |
| § 4.5 有限域的加法结构 | 120 |

| | |
|---|-----|
| § 4.6 有限域的代数结构与多项式的因式分解 | 130 |
| * § 4.7 迹与对偶基 | 137 |
| * § 4.8 孙子定理(中国剩余定理) | 141 |
| 习题..... | 143 |
| 参考文献..... | 144 |
| 第五章 循环码 | |
| § 5.1 循环码与理想 | 145 |
| § 5.2 由生成多项式的根定义循环码 | 152 |
| * § 5.3 幂等多项式和最小循环码 | 157 |
| § 5.4 缩短循环码与准循环码 | 159 |
| * § 5.5 平方剩余码 | 162 |
| § 5.6 多项式及域元素运算电路 | 165 |
| § 5.7 循环码的编码电路 | 174 |
| * § 5.8 循环码的谱表示与 MS 多项式 | 178 |
| * § 5.9 序列线性复杂度与勃拉哈特(Blahut)定理 | 183 |
| 习题..... | 188 |
| 参考文献..... | 189 |
| 第六章 循环码的译码 | |
| § 6.1 循环码译码的一般原理 | 190 |
| § 6.2 捕错译码 | 197 |
| § 6.3 大数逻辑译码原理 | 206 |
| § 6.4 大数逻辑可译码的构造 | 213 |
| § 6.5 软判决译码的基本原理 | 220 |
| § 6.6 码字错误概率最小的软判决译码 | 229 |
| 习题..... | 239 |
| 参考文献..... | 240 |
| 第七章 BCH 码与 Goppa 码 | |
| § 7.1 BCH 码的描述及其距离限 | 242 |
| § 7.2 二进制 BCH 码及其扩展 | 251 |
| § 7.3 Reed-Solomon(RS)码 | 259 |
| § 7.4 BCH 码的一般译码方法 | 268 |
| § 7.5 BCH 码的迭代译码算法 | 277 |
| * § 7.6 BCH 码的纠错纠删译码 | 291 |
| § 7.7 BCH 码的频域译码 | 293 |
| § 7.8 超 BCH 限译码 | 295 |
| § 7.9 Goppa 码的一般描述 | 298 |
| § 7.10 Goppa 码的扩展及其它特殊子类 | 304 |
| * § 7.11 交替码(Alternant 码)和 GBCH 码 | 309 |
| * § 7.12 交替码的欧几里德译码算法 | 313 |

| | |
|-----------------------------|-----|
| 习题 | 317 |
| 参考文献 | 318 |
| 第八章 代数几何码 | |
| § 8.1 代数几何的研究对象 | 319 |
| § 8.2 仿射空间与仿射变换 | 320 |
| § 8.3 射影空间与射影变换 | 323 |
| § 8.4 在有限域上的仿射曲线与射影曲线 | 324 |
| § 8.5 RS 码与 Goppa 码 | 325 |
| § 8.6 代数几何码的构成 | 329 |
| § 8.7 代数曲线中的一些重要概念 | 331 |
| § 8.8 Riemann—Roch 定理 | 335 |
| § 8.9 椭圆曲线码 | 338 |
| 习题 | 339 |
| 参考文献 | 340 |
| 第九章 纠突发错误循环码 | |
| § 9.1 基本码限 | 341 |
| § 9.2 纠单个错误循环码的构造 | 344 |
| § 9.3 纠定段(字节)突发错误码 | 352 |
| § 9.4 交错码与乘积码 | 356 |
| § 9.5 组合信道纠错码 | 361 |
| * § 9.6 级联码与贾斯特逊(Justesen)码 | 363 |
| § 9.7 纠突发错误码的译码 | 369 |
| 习题 | 375 |
| 参考文献 | 376 |
| 第十章 卷积码基础 | |
| § 10.1 卷积码的基本概念 | 378 |
| § 10.2 卷积码的矩阵和多项式描述 | 380 |
| § 10.3 伴随式计算与一般译码 | 394 |
| § 10.4 误差传播 | 398 |
| § 10.5 卷积码的树图描述和距离特性 | 402 |
| * § 10.6 卷积码的状态图表示和码的重量分布 | 409 |
| 习题 | 414 |
| 参考文献 | 415 |
| 第十一章 纠随机错误与纠突发错误卷积码 | |
| § 11.1 卷积码的大数逻辑译码 | 419 |
| § 11.2 非系统卷积码的大数逻辑译码 | 426 |
| § 11.3 纠突发错误卷积码的基本概念 | 428 |
| § 11.4 交错码 | 430 |
| * § 11.5 岩垂(Iwadare)码 | 431 |

| | |
|--------------------------------|-----|
| § 11.6 扩散卷积码..... | 435 |
| * § 11.7 加拉格尔(Gallager)码 | 439 |
| 习题..... | 441 |
| 参考文献..... | 442 |

第十二章 卷积码的概率译码

| | |
|--------------------------------------|-----|
| § 12.1 Viterbi(VB)译码算法的基本原理和实现 | 443 |
| § 12.2 Viterbi 译码算法的性能 | 452 |
| § 12.3 适用于 VB 译码算法的码和删余码 | 455 |
| § 12.4 序列译码——Fano 译码算法 | 466 |
| * § 12.5 序列译码——ST 译码算法 | 476 |
| * § 12.6 序列译码的性能 | 479 |
| * § 12.7 适用于序列译码的码 | 483 |
| § 12.8 调制与卷积码的结合(TCM 技术) | 487 |
| 习题..... | 502 |
| 参考文献..... | 503 |

第十三章 Turbo 码

| | |
|---------------------------------------|-----|
| § 13.1 Turbo 码的提出 | 505 |
| § 13.2 Turbo 码编码器的组成 | 506 |
| § 13.3 Turbo 码的译码 | 507 |
| § 13.4 Turbo 码的分量码、交织器与性能限 | 517 |
| § 13.5 Turbo 码在实际通信系统(3GPP)中的应用 | 528 |
| 习题 | 531 |
| 附录 Turbo 码不同译码算法的比较 | 532 |
| 参考文献..... | 533 |

第一章 纠错码的基本概念

本章主要介绍纠错码在数字通信系统中所处的地位，信道模型以及纠错码的某些基本概念和信道编码定理。

§ 1.1 数字通信系统的组成及信道模型

一、数字通信系统的组成

通信的目的是要把对方不知道的消息及时可靠地(有时还须秘密地)传送给对方，因此，要求一个通信系统传输消息必须可靠与快速，在数字通信系统中可靠与快速往往是一对矛盾。若要求快速，则必然使得每个数据码元所占的时间缩短、波形变窄、能量减少，从而在受到干扰后产生错误的可能性增加，传送消息的可靠性减低。若要求可靠，则使得传送消息的速率变慢。因此，如何较合理地解决可靠性与速度这一对矛盾，是正确设计一个通信系统关键问题之一。通信理论本身(包括纠错码)也正是在解决这对矛盾中不断发展起来的。

所有数字通信系统如通信、雷达、遥控遥测、数字计算机的存贮系统和内部运算以及数字计算机之间的数据传输等，都可归结成如图 1-1 所示的模型。

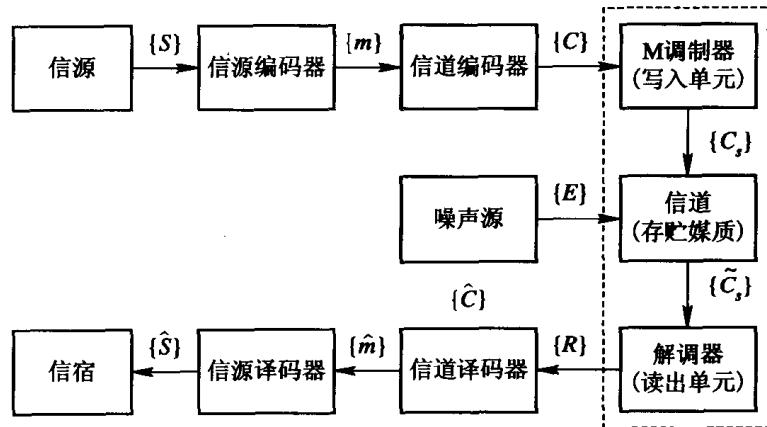


图 1-1 数字通信系统模型

图中，信源编码器是把信源发出的消息如语言、图像、文字等转换成为二进制(也可转换成为多进制)形式的信息序列，并且为了使传输有效，还去掉了一些与传输信息无关的多余度(有时为了保密，信源编码器后还可接上加密器)。为了抗击传输过程中的各种干

扰，往往要人为地增加一些多余度，使其具有自动检错或纠错能力，这种功能由图中的信道编码器即纠错编码器完成。发射机(调制器)的功用是把纠错码送出的信息序列通过调制器变成适合于信道传输的信号。数字信号在信道传输过程中，总会遇到各种干扰而使信号失真，这种失真信号传输到接收端的接收机，进行解调，变成二进制(或多进制)信息序列。由于信道干扰的影响，该信息序列中可能已有错误，经过信道译码器即纠错码译码器，对其中的错误进行纠正，再通过信源译码器(及解密器)恢复成原来的消息送给用户。

我们关心的是图中的信道编、译码器即纠错编、译码器两个方框，为了研究方便，将上述模型再进一步简化成图 1-2 所示的模型。在此模型中，信源是指原来的信源和信源编码器，其输出是二(多)进制信息序列。信道是包括发射机、实际信道(或称传输媒质)和接收机在内的广义信道(又称编码信道)，它的输入是二(多)进制数字序列，输出一般也是二(多)进制数字序列，而图中的信宿可以是人或计算机。

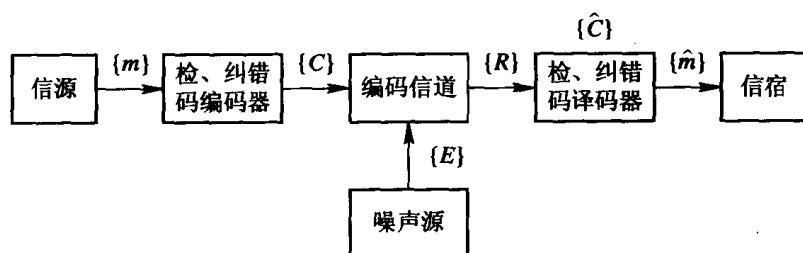


图 1-2 数字通信系统简化模型

二、信道模型

现在以图 1-2 的模型来讨论二进制数字序列通过该系统时所发生的情况。设从信源送出字母 A，它的二进制序列为 11000，以基带信号传送，经发射机调制后，送往信道的已调信号如图 1-3 所示。由于信道的干扰，从信道输出端的信号产生了失真，如图 1-4 所示。这些失真信号送入接收机进行判决时，由于第一、二、四、五码元的波形失真不大，容易正确地判为 1、1 和 0、0；但对第三个码元来说，由于失真严重而难于判决。这时有以下三种判决方法：一是勉强作出是 0 还是 1 的判决，即所谓硬判决；另一种是对该码元暂且不作判决，而输出一个未知或待定的信号“x”，称其为删除符号；第三种方法是输出一种有关该码元的信息，例如关于 0 和 1 的后验概率或似然函数，这种作法称为软判决。当然软判决的性能较好，但实现起来较复杂。

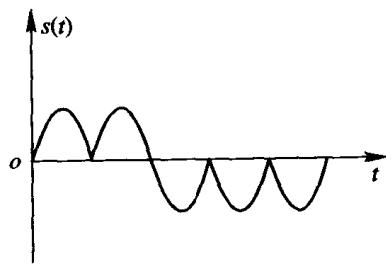


图 1-3 11000 发送的已调信号波形

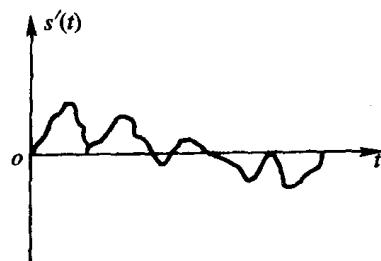


图 1-4 接收端收到的失真信号波形

在二进制硬判决情况下，信道可用图 1-5 所示的简单模型表示。图中， p_{01} 和 p_{10} 分别是 0 错成 1 和 1 错成 0 的概率，称信道转移概率。该信道的信道转移概率矩阵可用

$$\mathbf{P} = \begin{bmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{bmatrix} = \begin{bmatrix} 1 - p_{01} & p_{01} \\ p_{10} & 1 - p_{10} \end{bmatrix}$$

描述。如果 $p_{01} = p_{10} = p_e$ ，则称这种信道为**二进制对称信道**，简称**BSC**。否则，称为**不对称信道**。若 p_{01} 或 p_{10} 等于零，则称为**Z 信道**。通常 BSC 是一种**无记忆信道**，所以也称**随机信道**，它说明数据序列中出现的错误彼此无关。

如果信道的输入是二进制符号，而输出是离散的 $q (q = p^m \geq 2)$ 进制符号，如图 1-6 所示，且 $p(i|0) = p(q-1-i|1)$ ， $i = 0, 1, \dots, q-1$ ，则这种信道称为**离散无记忆信道 (DMC)**，显然 BSC 是 DMC 的一种特殊情况。DMC 的信道转移概率矩阵

$$\mathbf{P} = \begin{bmatrix} p(0|0) & p(1|0) & \cdots & p(q-1|0) \\ p(0|1) & p(1|1) & \cdots & p(q-1|1) \end{bmatrix}$$

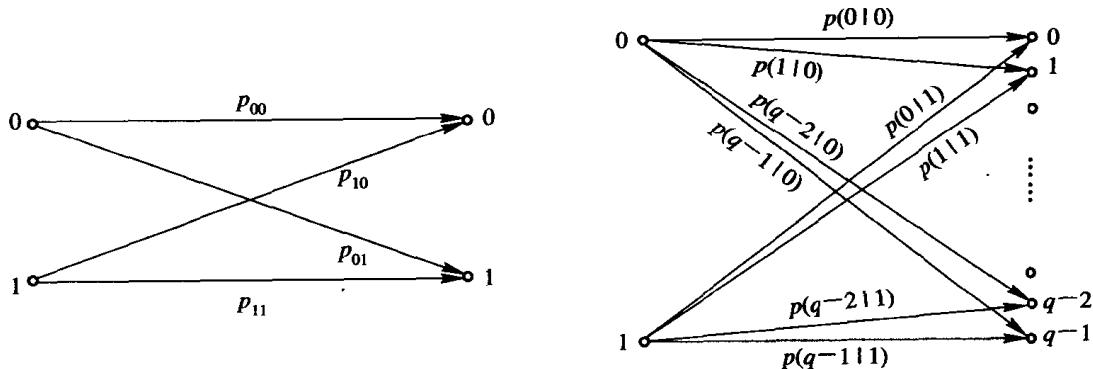


图 1-5 二进制信道

图 1-6 DMC

在作删除判决情况下，信道可用图 1-7 所示的模型表示，称为**二进制删除信道**，简称**BEC**，一般它也是对称信道。图中， p_e 为信道的转移概率， q 为删除概率，在有删除处理情况下，信道的转移概率 p_e 一般很小，可忽略，因此把图 1-7 所示的模型用图 1-8 代替，称为**二进制纯删除信道**。以后所说的 BEC 都是指这种信道。应当指出，当码元作删除处理时，它在序列中的位置是已知的，仅不知其值是 0 还是 1，故对这种 BEC 信道的纠错要比 BSC 信道容易。

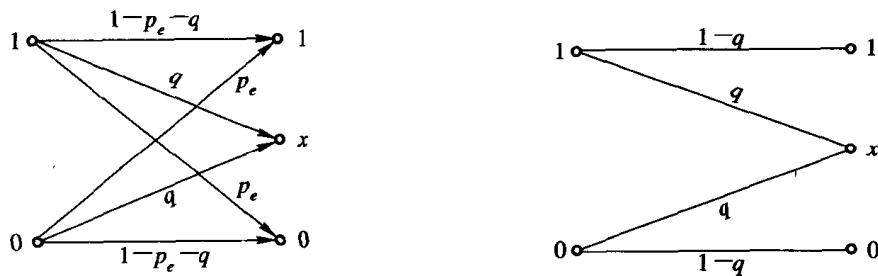


图 1-7 二进制删除信道

图 1-8 二进制纯删除信道

上述三种信道模型只是为了讨论问题方便而简化成理想的情况，它们表达了某些实际信道传送信号的主要特征。例如，卫星信道或深空信道，可近似看成是 BSC。但有很多实际信道如高频、散射、有线等信道，由于各种干扰所造成的错误，往往不是单个地而是成群成串地出现的，也就是一个错误的出现，往往引起其前后码元的错误(突发错误)，表现为错误之间的相关性。产生这种错误的信道称有记忆信道或突发信道。在计算机存贮系统中，磁带的缺陷或读写头接触不良所引起的错误，也属于这种类型。但由于实际信道干扰的复杂性，所引起的错误往往不是单纯的一种，而是两种错误形式并存，只不过有的信道以某种错误形式为主罢了。像这种随机错误与突发错误并存的信道，称为组合信道或复合信道。有关这些信道的模型请参阅[5]。作为检错与纠错用的抗干扰码，必须针对这几类信道，设计能纠正随机错误或纠正突发错误的码，或者设计既能纠正随机错误又能纠正突发错误的码。我们将首先讨论 BSC 和 BEC(包括多进制)的纠错编码问题，而后讨论纠正突发错误码，最后讨论组合信道的纠错码。

由于目前在信道中传输或计算机内部运算的数据序列，大部分是二进制数字序列，因此以后主要讨论二进制数字通信中的纠错码，当然这些纠错码往往可以推广到 q (素数或素数的幂)进制情况，这将在以后具体讨论时予以说明。二进制情况下，序列之间 0、1 两个符号按下列规则进行运算：

| 模 2 相加 | | | 模 2 相乘 | | |
|----------|---|---|-----------|---|---|
| \oplus | 0 | 1 | \otimes | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 1 |

为了简便，今后用 + 和 \times 表示模 2 相加和相乘。在模 2 情况下，加和减是一回事。

三、错误图样

设发送的是 n 个码元长的序列 $C: (c_{n-1}, c_{n-2}, \dots, c_1, c_0)$ ，通过信道传输到达接收端(纠错码译码器输入端)的序列为 $R: (r_{n-1}, r_{n-2}, \dots, r_1, r_0)$ 。由于信道中存在干扰， R 序列中的某些码元可能与 C 序列中对应码元的值不同，也就是说产生了错误。由于二进制序列中的错误不外乎是 1 错成 0 或 0 错成 1，因此，如果把信道中的干扰也用二进制序列 $E: (e_{n-1}, e_{n-2}, \dots, e_1, e_0)$ 表示，则相应有错误的各位 e_i 取值为 1，无错的各位取值为 0，而 R 就是 C 与 E 序列模 2 相加的结果，我们称 E 为信道的错误图样或干扰矢量。例如，发送序列 $C: (1111100000)$ ，收到的序列 $R: (1001010000)$ ，第二、三、五、六位产生了错误，因此信道的错误图样 E 的二、三、五、六位取值为 1，其它各位取值为 0，即 $E: (0110110000)$ 。用式子可表示成：

$$\begin{array}{r} \text{发送序列 } C: 1111100000 \\ \oplus \quad \text{错误图样 } E: 0110110000 \\ \hline \text{接收序列 } R: 1001010000 \end{array}$$

即 $R = C + E$ ，或 $E = R - C$ 。

信道干扰所造成的错误可在序列中的任一位出现，且也可以在 n 长序列中同时出现一位、二位……，甚至 n 位错误。因此，若发送的 C 序列长为 n ，则信道中可能产生的错误图

样 E 共有 2^n 种。

若为突发信道，则在错误图样 E 中，第一个 1 与最后一个 1 之间的长度称为**突发长度**，其图样称为**突发图样**。在该例中，突发图样是(11011)，突发长度为 5。

§ 1.2 差错控制系统和纠错码分类

一、差错控制系统分类

在数字通信系统中，利用纠错码或检错码进行差错控制的方式大致有以下几类：

(1) 重传反馈方式(ARQ)。应用 ARQ 方式纠错的通信系统如图 1-9 所示。发送端发出能够发现(检测)错误的码，接收端收到通过信道传来的码后，在译码器根据该码的编码规则，判决收到的码序列中有无错误产生，并通过反馈信道把判决结果用判决信号告诉发端。发端根据这些判决信号，把接收端认为有错的消息再次传送，直到接收端认为正确接收为止。

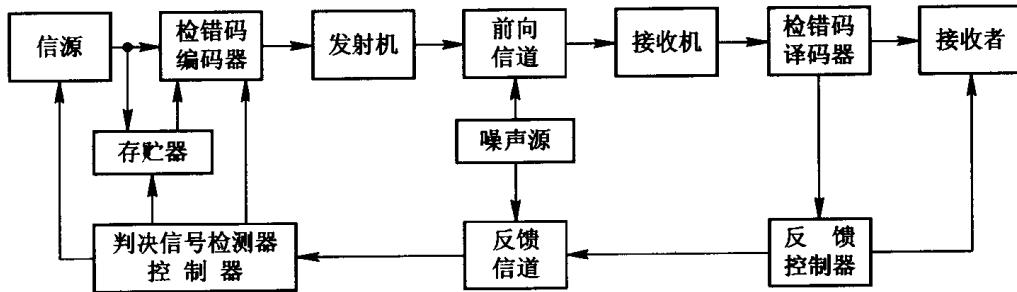


图 1-9 ARQ 通信系统

从上可知，应用 ARQ 方式必须有一反馈信道，一般较适用于一个用户对一个用户(点对点)的通信，且要求信源能够控制，系统收发两端必须互相配合、密切协作，因此这种方式的控制电路比较复杂。由于反馈重发的次数与信道干扰情况有关，若信道干扰很频繁，则系统经常处于重发消息的状态，因此这种方式传送消息的连贯性和实时性较差。该方式的优点是：编译码设备比较简单；在一定的多余度码元下，检错码的检错能力比纠错码的纠错能力要高得多，因而整个系统的纠错能力极强，能获得极低的误码率；由于检错码的检错能力与信道干扰的变化基本无关，因此这种系统的适应性很强，特别适应于短波、散射、有线等干扰情况特别复杂的信道中。

(2) 前向纠错方式(FEC)。利用前向纠错方式进行差错控制的数字通信系统如图 1-2 所示。发送端发送能够被纠错的码，接收端收到这些码后，通过纠错译码器不仅能自动地发现错误，而且能自动地纠正接收码字传输中的错误。这种方式的优点是不需要反馈信道，能进行一个用户对多个用户的同播通信，译码实时性较好，控制电路比 ARQ 的简单。其缺点是译码设备比较复杂，所选用的纠错码必须与信道的干扰情况相匹配，因而对信道的适应性较差。为了要获得比较低的误码率，往往必须以最坏的信道条件来设计纠错码，故所需的多余度码元比检错码要多得多，从而使编码效率很低。但由于这种方式能同播，特别适用于军用通信，并且随着编码理论的发展和编译码设备所需的大规模集成电路成本

的不断降低，译码设备有可能做得越来越简单，成本越来越低，因而在实际的数字通信中逐渐得到广泛应用。

(3) 混合纠错方式(HEC)。这种方式是发送端发送的码不仅能够被检测出错误，而且还具有一定的纠错能力。接收端收到码序列以后，首先检验错误情况，如果在纠错码的纠错能力以内，则自动进行纠错。如果错误很多，超过了码的纠错能力，但能检测出来，则接收端通过反馈信道，要求发端重新传送有错的消息。这种方式在一定程度上避免了 FEC 方式要求用复杂的译码设备和 ARQ 方式信息连贯性差的缺点，并能达到较低的误码率，因此在实际中的应用越来越广。

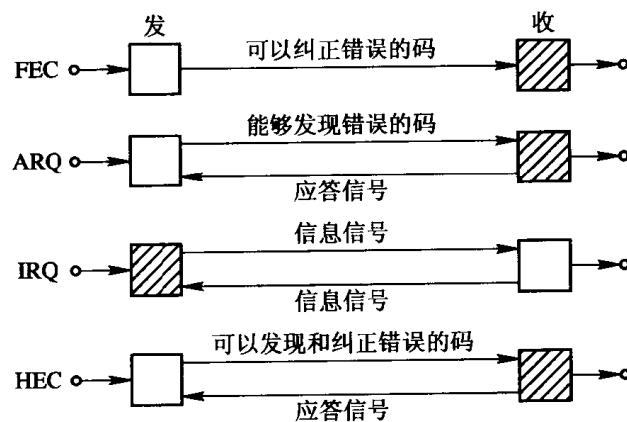


图 1-10 差错控制的基本方式

除了上述三种主要方式以外，还有所谓**狭义信息反馈系统(IRQ)**。这种方式是接收端把收到的消息原封不动地通过反馈信道送回发送端，发送端比较发送的与反馈回来的消息，从而发现错误，并且把传错的消息再次传送，最后达到使对方正确接收消息的目的。

为了便于比较，我们把上述几种方式用图 1-10 所示的框图表示。图中，有斜线的方框表示在该端检出错误。在实际系统设计中，如何根据实际情况选择哪种差错控制方式是一个比较复杂的问题，由于篇幅所限，这里不再讨论，有兴趣的读者可参阅[5]。

二、纠错码的分类

上述各种差错控制系统中所用到的码，不外乎是能在译码器自动发现错误的**检错码**，或者不仅能发现错误而且能自动纠正错误的**纠错码**，或者能纠正删除错误的**纠删码**。但这三类码之间没有明显区分，以后将看到，任何一类码，按照译码方法不同，均可作为检错码、纠错码或纠删码来使用。除了上述的划分方法以外，通常还按以下方式对纠错码进行分类：

(1) 按照对信息元处理方法的不同，分为分组码与卷积码两大类。

分组码是把信源输出的信息序列，以 k 个码元划分为一段，通过编码器把这段 k 个信息元按一定规则产生 r 个校验(监督)元，输出长为 $n = k - r$ 的一个码组。因此每一码组的校验元仅与本组的信息元有关，而与别组无关。分组码用 (n, k) 表示， n 表示码长， k 表示信息位。

卷积码是把信源输出的信息序列，以 k_0 个(k_0 通常小于 k)码元分为一段，通过编码器

输出长为 n_0 ($\geq k_0$) 一段的码段。但是该码段的 $n_0 - k_0$ 个校验元不仅与本组的信息元有关，而且也与其前 m 段的信息元有关，称 m 为 **编码存储**。因此卷积码用 (n_0, k_0, m) 表示。

(2) 根据校验元与信息元之间的关系分为线性码与非线性码。若校验元与信息元之间的关系是线性关系(满足线性叠加原理)，则称为**线性码**；否则，称为**非线性码**。

由于非线性码的分析比较困难，实现较为复杂，故今后我们仅讨论线性码。

(3) 按照纠正错误的类型可分为**纠正随机(独立)错误的码**、**纠正突发错误的码**和**纠正同步错误的码**，以及**既能纠正随机错误又能纠正突发错误的码**。

(4) 按照每个码元取值来分，可分为**二进制码**与 **q 进制码** ($q = p^m$, p 为素数, m 为正整数)。

(5) 按照对每个信息元保护能力是否相等可分为**等保护纠错码**与**不等保护(UEP)纠错码**。除非特别说明，今后讨论的纠错码均指等保护能力的码。

此外，在分组码中按照码的结构特点，又可分为**循环码**与**非循环码**。为了清楚起见，我们把上述分类用图 1-11 表示。

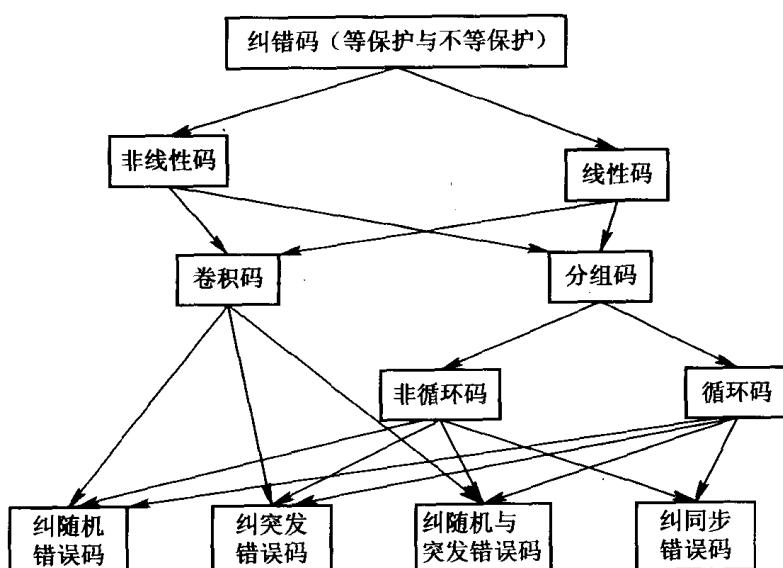


图 1-11 纠错码分类

§ 1.3 最大似然译码和纠错码的基本概念

一、基本定义

利用纠错码进行差错控制的数字通信系统如图 1-12 所示，由此可如下定义分组码。

定义 1.3.1 分组码是对每段 k 位长的信息组，以一定规则增加 $r=n-k$ 个校验元，组成长为 n 的序列： $(c_{n-1}, c_{n-2}, \dots, c_1, c_0)$ ，称这个序列为**码字(码组、码矢)**。在二进制情况下，信息组总共有 2^k (q 进制为 q^k) 个，因此通过编码器后，相应的码字也有 2^k 个，称这 2^k 个码字集合为 (n, k) 分组码。

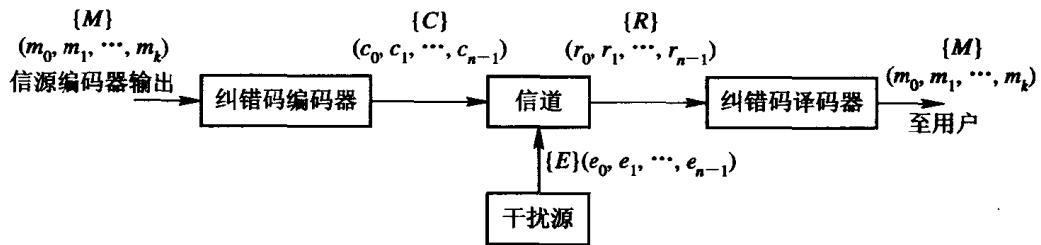


图 1-12 利用分组码的数字通信模型

n 长序列的可能排列总共有 2^n 种(每一 n 长序列称为 n 重)，而 (n, k) 分组码的码字集合只有 2^k 种。所以，分组码的编码问题就是定出一套规则，以便从 2^n 个 n 重中选出 2^k 个码字，不同的选取规则就得到不同的码。我们称被选取的 2^k 个 n 重为许用码组，其余的 $2^n - 2^k$ 个为禁用码组。

称 $R=k/n$ 为码率，表示 (n, k) 分组码中，信息位在码字中所占的比重。 R 是衡量分组码有效性的一个基本参数。

图 1-13 是一个 $(2, 1, 2)$ 卷积码编码器。若输入的信息序列以 $k_0=1$ 个码元分段输入，则输出以 $n_0=2$ 个码元为一段输出，如输入的信息序列 $M=(1\ 1\ 0\ 1\ 0\ 0)$ ，输出的码序列为 $C=(11, 10, 10, 00, 01, 11, 00, \dots)$ 。可知随着信息元的不断输入，输出的是一个半无限长的码序列，由此可如下定义卷积码。

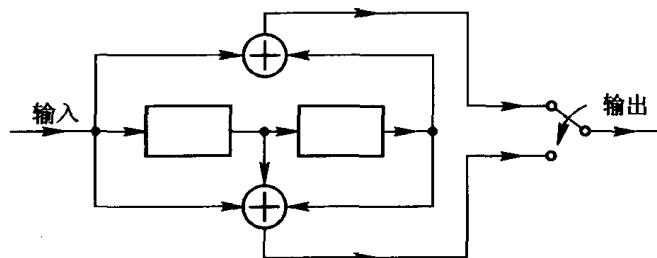


图 1-13 一个 $(2, 1, 2)$ 卷积码编码器

定义 1.3.2 (n_0, k_0, m) 卷积码是对每段 k_0 长的信息组以一定的规则增加 $r_0=n_0-k_0$ 个校验元，组成长为 n_0 的码段。 n_0-k_0 个校验元不仅与本段的信息元有关，且与前 m 段的信息元有关，当信息元不断输入时，输出的码序列是一个半无限长序列。

(n_0, k_0, m) 卷积码的码率 $R=k_0/n_0$ 。与分组码的码长 n 相对应，在卷积码中称 $n_c=n_0(m+1)$ 为**编码约束长度**，说明 k_0 个信息元从输入编码器到离开时在码序列中影响的码元数目，如图 1-13 中 $(2, 1, 2)$ 卷积码的 $n_c=6$ 。

二、最大似然译码

由图 1-2 可知，信道输出的 R 是一个二(或 q)进制序列，而译码器的输出是一个信息序列 M 的估值序列 \hat{M} 。

译码器的基本任务就是根据一套译码规则，由接收序列 R 给出与发送的信息序列 M 最接近(最好是相同)的估值序列 \hat{M} 。由于 M 与码字 C 之间存在一一对应关系，所以这等

价于译码器根据 \mathbf{R} 产生一个 \mathbf{C} 的估值序列 $\hat{\mathbf{C}}$ 。显然，当且仅当 $\hat{\mathbf{C}}=\mathbf{C}$ 时， $\hat{\mathbf{M}}=\mathbf{M}$ ，这时译码器正确译码。

如果译码器输出的 $\hat{\mathbf{C}} \neq \mathbf{C}$ ，则译码器产生了错误译码。之所以产生错误译码是由于：信道干扰很严重，超过了码本身的纠错能力；其次，由于译码设备的故障（这点本书不予讨论）。当给定接收序列 \mathbf{R} 时，译码器的条件译码错误概率定义为

$$P(E|\mathbf{R}) = P(\hat{\mathbf{C}} \neq \mathbf{C}|\mathbf{R})$$

所以译码器的错误译码概率

$$P_E = \sum_{\mathbf{R}} P(E|\mathbf{R})P(\mathbf{R})$$

$P(\mathbf{R})$ 是接收 \mathbf{R} 的概率，与译码方法无关，所以译码错误概率最小的最佳译码规则是使

$$\begin{aligned} \min P_E &= \min_{\mathbf{R}} P(E|\mathbf{R}) = \min_{\mathbf{R}} P(\hat{\mathbf{C}} \neq \mathbf{C}|\mathbf{R}) \\ \min P(\hat{\mathbf{C}} \neq \mathbf{C}|\mathbf{R}) &\Rightarrow \max P(\hat{\mathbf{C}} = \mathbf{C}|\mathbf{R}) \end{aligned} \quad (1.3.1)$$

因此，如果译码器对输入的 \mathbf{R} ，能在 2^k 个码字中选择一个使 $P(\hat{\mathbf{C}}_i = \mathbf{C}|\mathbf{R}) (i=1, 2, \dots, 2^k)$ 最大的码字 \mathbf{C}_i 作为 \mathbf{C} 的估值序列 $\hat{\mathbf{C}}$ ，则这种译码规则一定使译码器输出错误概率最小，称这种译码规则为**最大后验概率译码**。

由贝叶斯公式

$$P(\mathbf{C}_i|\mathbf{R}) = \frac{P(\mathbf{C}_i)P(\mathbf{R}|\mathbf{C}_i)}{P(\mathbf{R})}$$

可知，若发端发送每个码字的概率 $P(\mathbf{C}_i)$ 均相同，且由于 $P(\mathbf{R})$ 与译码方法无关，所以

$$\max_{i=1,2,\dots,2^k} P(\mathbf{C}_i|\mathbf{R}) \Rightarrow \max_{i=1,2,\dots,2^k} P(\mathbf{R}|\mathbf{C}_i) \quad (1.3.2)$$

对 DMC 而言

$$P(\mathbf{R}|\mathbf{C}_i) = \prod_{j=1}^n P(r_i|c_{ij}) \quad (1.3.3)$$

这里码字 $\mathbf{C}_i = (c_{i1}, c_{i2}, \dots, c_{in})$, $i=1, 2, \dots, 2^k$ 。

一个译码器的译码规则若能在 2^k 个码字 \mathbf{C} 中选择某一个 \mathbf{C}_i 使式(1.3.2)成为最大，则这种译码规则称为**最大似然译码(MLD)**， $P(\mathbf{R}|\mathbf{C})$ 称为似然函数，相应的译码器称为最大似然译码器。由于 $\log_b x$ 与 x 是单调关系，因此式(1.3.2)与式(1.3.3)可写成

$$\max_{i=1,2,\dots,2^k} \log_b P(\mathbf{R}|\mathbf{C}_i) = \max_{i=1,2,\dots,2^k} \sum_{j=1}^n \log_b P(r_i|c_{ij}) \quad (1.3.4)$$

称 $\log_b P(\mathbf{R}|\mathbf{C})$ 为对数似然函数或似然函数。对于 DMC 信道，MLD 是使译码错误概率最小的一种最佳译码准则或方法，但此时要求发端发送每一码字的概率 $P(\mathbf{C}_i) (i=1, 2, \dots, 2^k)$ 均相等，否则 MLD 不是最佳的。在以后的讨论中，都认为 $P(\mathbf{C}_i)$ 均近似相等。

为了进一步描述码的性能和译码器如何进行最佳译码，我们还必须介绍两个重要的基本概念。

三、汉明(Hamming)距离与重量

定义 1.3.3 两个 n 重 \mathbf{x}, \mathbf{y} 之间，对应位取值不同的个数，称为它们之间的**汉明距离**，

用 $d(x, y)$ 表示。

例如, 若 $x: (10101)$, $y: (01111)$, 则 $d(x, y) = 3$ 。

定义 1.3.4 n 重 x 中非零码元的个数, 称为它的汉明重量, 简称重量, 用 $w(x)$ 表示。

例如, 若 $x: (10101)$, 则 $w(x) = 3$ 。若 $y: (01111)$, 则 $w(y) = 4$, 等等。

定义 1.3.5 (n, k) 分组码中, 任两个码字之间距离的最小值, 称为该分组码的最小汉明距离 d_0 , 简称最小距离

$$d_0 = \min_{x, y \in (n, k)} \{d(x, y)\}$$

例如 $(3, 2)$ 码, $n=3$, $k=2$, 共有 $2^2=4$ 个码字: $000, 011, 101, 110$, 显然 $d_0=2$ 。

d_0 是 (n, k) 分组码的另一个重要参数。它表明了分组码抗干扰能力的大小。以后将看到: d_0 越大, 码的抗干扰能力越强, 在同样译码方法下它的译码错误概率越小。由上可知, R 和 d_0 是 (n, k) 分组码的两个最重要参数。纠错编码的基本任务之一就是构造出 R 一定、 d_0 尽可能大的码, 或 d_0 一定、 R 尽可能高的码。下面用几个具体例子说明码的 R 、 d_0 以及译码错误概率之间的关系。

例 1.1 重复码 重复码是 $k=1$ 的 $(n, 1)$ 码, 它的编码规则(即在 2^n 个 n 重中挑选 $2^k=2^1$ 个码字的规则)是 $(n-1)$ 个校验元是信息元的重复, 设信息元为 $c_n=1$, 则校验元 $c_i=c_{n-1}$ ($i=0, 1, 2, \dots, n-2$)。由于信息组只有 $2^k=2$ 组: 0 和 1, 因此相应的许用码字只有两个($00\cdots 0$)和($11\cdots 1$)。设它们通过 BSC 传输, 信道的转移概率为 p_e 。

通常情况下, $p_e \leqslant 0.5$, 因此在传输中没有错误的可能性比出现一个错误的可能性大, 出现一个错误的可能性比出现两个错误的大, 等等。也就是说信道错误图样 E 中, 出现重量最轻的图样可能性最大。

$$P(w(E) = 0) > P(w(E) = 1) > P(w(E) = 2) > \dots$$

或

$$(1 - p_e)^n > p_e(1 - p_e)^{n-1} > p_e^2(1 - p_e)^{n-2} > \dots$$

由 $E=R-C$ 及式(1.3.2)可知, MLD 译码器寻求可能出现的错误图样, 就是由接收到的 R 在 2^k 个码字集中, 寻求与 R 的汉明距离最小的码字 C_i , 为最可能发送的码字而接收, 这就是**最小汉明距离译码**(或称**最近邻区译码**)。(试证明: 在 DMC 中, 最小汉明距离译码就是 MLD。)在重复码情况下这种译码方案就是根据收到序列中 0 和 1 的多少, 来判断信息组是 0 还是 1。若接收序列中 1 的个数大于 $n/2$, 则判为 1; 否则, 判为 0。这种译码方案就是**大数准则译码**。

当 n 是奇数时, 按照这种大数准则译码方法, 译码器总可以很快地作出是 0 还是 1 的判决。当 n 是偶数时可能出现以下情况, 即接收序列中“1”的个数和“0”的个数刚好相等, 此时若按大数准则无法作出判断, 造成**译码失败**, 这种译码称为**不完备译码**。而 n 为奇数情况下的译码(即译码器一定作出是哪一个信息组的判决)称为**完备译码**。译码失败只表明译码器无法给出明确判断, 但并不等于译码错误, 而仅表明接收序列中存在有错误。此时, 如果我们与 ARQ 系统相结合, 则可以利用此信息要求对方重发该组信息, 直到译码器作出明确判决为止。下面几个具体例子说明重复码的抗干扰能力。

(1) $(1, 1)$ 码。这种码 $n=1$, $k=1$, $d=1$, $R=1$, 显然无任何抗干扰能力。设 BSC 中的 $p_e=0.1$, 则这种码的误码率仍为 0.1。

(2) (2, 1)重复码。该重复码的两个许用码字是(00)和(11), $d_0=2$, $R=1/2$ 。若用不完备译码(并与 ARQ 结合)则译码错误概率为 $p_e^2=10^{-2}$ 。也就是说只有当(00)错成(11)或(11)错成(00)时, 才造成译码错误。而(01)和(10)不是许用码字, 不会造成译码错误。因此, 这个码能发现传输中的一个错误, 但不能自动纠正。

(3) (3, 1)重复码。显然, 它的两个码字是(000)和(111), $d_0=3$, $R=1/3$ 。设发的是(000), 若收到的是(001)或(010)或(100), 则根据大数译码准则正确地判为(000), 信息组为0。若收到的是(011), (101), (110), (111)中之一, 则造成译码错误, 错判为信息组是1。因此, 该码若用完备译码能纠正序列中的一个错误, 此时译码错误概率

$$p = 1 - Q = 1 - [(1 - p_e)^3 + 3p_e(1 - p_e)^2] = 2.8 \times 10^{-2}$$

也就是误码率由 0.1 减至 2.8×10^{-2} 。

若该码不用作纠错, 而采用不完备译码用作检错, 则可以发现两个错误, 与 ARQ 系统结合后, 译码错误概率减至 $p_e^3=10^{-3}$ 。

(4) (4, 1)重复码。显然, 该码的 $d_0=4$, $R=1/4$ 。由于 n 是偶数, 故只能采取不完备译码。它的纠错能力如下:

① 能纠正一个错误同时发现两个错误。若发送的是(0000), 则错一个时, 根据大数准则可正确判断发送的是0信息组。若错两个变成(0011), (1100), (1010), (0101), (1001), (0110)之一时, 则译码器无法作出判决, 而指出发生了两个错误。仅在变成(1110), (0111), (1011), (1101)和(1111)时, 译码器才作出错误译码。因此, 这时的译码错误概率

$$p = p_e^4 + 4p_e^3(1 - p_e) = 3.6 \times 10^{-4}$$

② 若仅用来检错, 则可检测 $e=d_0-1=3$ 个错误。显然, 若发送的是(0000), 则仅在变成(1111)时, 才产生错误译码, 而其它情况均能正确译码或发现错误。因此, 此时的译码错误概率 $p=p_e^4=10^{-4}$ 。

(5) (5, 1)重复码。(5, 1)码的 $d_0=5$, $R=1/5$ 。它的纠错能力如下:

① 能纠正两个随机错误, 这时的译码错误概率

$$p = 1 - [(1 - p_e)^5 + \binom{5}{1}p_e(1 - p_e)^4 + \binom{5}{2}p_e^2(1 - p_e)^3] = 1 - 0.993 = 7 \times 10^{-3}$$

因此, 传送信息组0、1的错误概率从0.1降至千分之七, 约降低了两个量级。

② 若仅用来检错, 则能发现 $e=d_0-1=4$ 个错误。若与 ARQ 系统结合进行纠错, 则译码错误概率大约为 $p=p_e^5=10^{-5}$ 。

由上面有关重复码的一系列例子可以看到:

(1) 随着码长 n 的增加, 重复码的 $d_0=n$ 越来越大, 抗干扰能力越来越强, 即 $d/n=1$, 误码率也越来越小, 但码率 $R=1/n$ 却越来越低, 并随着 n 的增加而趋近于零。

(2) 在用同样的(n, k)码下, 应用非完备译码并与 ARQ 系统相结合, 译码器所给出的误码率比完备译码所产生的要低得多。

通过上面这些例子还可看出, (n, k)分组码的最小距离 d_0 (今后简称 d)与纠错能力有如下关系。

定理 1.3.1 任一(n, k)分组码, 若要在码字内:

(1) 检测 e 个随机错误, 则要求码的最小距离 $d \geq e+1$;

- (2) 纠正 t 个随机错误, 则要求 $d \geq 2t+1$;
- (3) 纠正 t 个随机错误, 同时检测 $e (\geq t)$ 个错误, 则要求 $d \geq t+e+1$ 。
- (4) 纠正 t 个错误和 ρ 个删除, 则要求 $d \geq 2t+\rho+1$ 。

证明

(1) 由图 1-14(a) 可知, 若 C_1 发生了 e 个错误变为 C'_1 , 则 $d(C_1, C'_1) = e$, 设 $e = d - 1$, 则 $d(C'_1, C_2) = 1$, 故 $C'_1 \neq C_2$, 因此译码器不会将 C'_1 错判成 C_2 , 检测到 $e = d - 1$ 个错误。

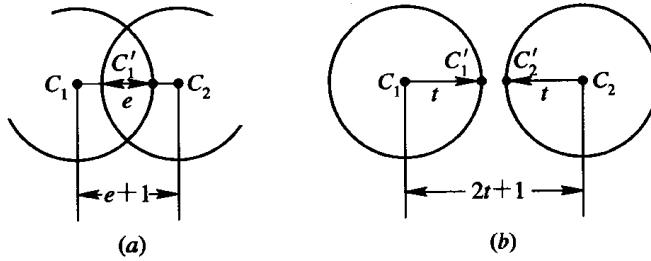


图 1-14 纠错码纠错能力的几何解释

(2) 设 C_1 与 C_2 是 (n, k) 码中任两码字距离之最小者, 且为 $2t+1$ 。则 C_1 错了 t 个错误以后变成 C'_1 , 它们之间的距离 $d(C_1, C'_1) = t$, 但 $d(C'_1, C_2) = t+1$ 。 $d(C'_1, C_2) > d(C_1, C'_1)$, 如图 1-14(b) 所示, 所以译码器可以根据它们之间的距离的大小正确译码, 从而纠正 t 个错误。

(3) 这里所指的同时, 是当错误个数 $\leq t$ 时, 该码能纠正 t 个错; 当错误个数大于 t 而小于 e 时, 则码能发现 e 个错误。由(1)和(2)的证明可直接得到结论(3), 请读者自行证明。 ■

由此定理可知, 一个距离为 d 的分组码, 至多能纠正 $t = \lfloor (d-1)/2 \rfloor$ ($\lfloor x \rfloor$ 是 x 的整数部分) 个错误。该定理确定了码的纠错能力与它的距离之间的关系, 是纠错码理论中最基本的定理之一。

例 1.2 奇偶校验(监督)码 奇偶校验码是只有一个校验元的 $(n, n-1)$ 分组码。

设给定 $k = n-1$ 位的二进制信息码组为: $m_{k-1}, m_{k-2}, \dots, m_1, m_0$, 则按如下规则完成码中一个码字 $(c_{n-1}, c_{n-2}, \dots, c_1, c_0)$ 的编码: $c_{n-1} = m_{k-1}, c_{n-2} = m_{k-2}, \dots, c_2 = m_1, c_1 = m_0$, 而一个校验元

$$c_0 = m_{k-1} + m_{k-2} + \dots + m_1 + m_0$$

或

$$\begin{aligned} m_{k-1} + m_{k-2} + \dots + m_1 + m_0 + c_0 &= 0 \\ c_{n-1} + c_{n-2} + \dots + c_1 + c_0 &= 0 \end{aligned} \quad (1.3.5)$$

该式保证每个码字中“1”的个数为偶数, 所以称这种校验关系为奇偶校验。由于分组码中的每一个码字均按同一规则构成, 故称这种分组码为一致校验码。显然, 码中的码字数目 $M = 2^k = 2^{n-1}$ 。

(1) $(2, 1)$ 奇偶校验码。此码的 $n=2, k=1$, 按 $c_1 + c_0 = 0$ 求出校验元 $c_0 = c_1$, 所以两个许用码组是 00 和 11, 此时 $d=2, R=1/2$, 能发现一个错误。若在 $p_e=0.1$ 的 BSC 中传送, 则利用不完备译码并与 ARQ 相结合, 可使误码率大约减至 $p=10^{-2}$ 。

(2) (3, 2)码。此码有 $2^2=4$ 个信息组, 由式(1.3.5)求得校验元, 得到相应的 4 个码字为: 000, 011, 101, 110。由此看出, 这 4 个信息组就是在 8 个二进制三重中, 把重量为偶数的 4 个三重挑选为许用码字, 重量为奇数的其它 4 个为禁用码组。显然, 该码的 $d=2$, $R=\frac{2}{3}$, 也只能发现码元中的一个错误。译码误码率大约为 $p=\binom{3}{2} p_e^2 (1-p_e)=2.7 \times 10^{-2}$ 。

(3) (4, 3)码。该码的 8 个码字是按照式(1.3.5)的要求, 在 16 个二进制四重中挑选出来的, 其重量均为偶数。该码的 $d=2$, $R=3/4$, 故只能检测码字中的一位错误。其译码误码率约为 $p=\binom{4}{2} p_e^2 (1-p_e)^3 + p_e^4 \approx 5 \times 10^{-2}$ 。 ■

由上看出, $(n, n-1)$ 奇偶校验码。当它的 $n \rightarrow \infty$ 时, $R \rightarrow 1$, 但 $d=2$, $d/n \rightarrow 0$, 译码误码率 $p \rightarrow p_e$, 接近未编码时的情况。即随着码长的增加, 抗干扰能力接近于零。

上述 $(n, 1)$ 和 $(n, n-1)$ 码, 随着码长 n 的增加, 前者 $R \rightarrow 0$, 后者抗干扰能力接近于零, 或 $d/n \rightarrow 0$, 都不理想。那么是否存在有一种码, 随着 n 的增加。其纠错能力和传信率都保持一定呢? 换言之, 在 R 一定时, 随着 $n \rightarrow \infty$, $d/n > 0$, 从而使 $p \rightarrow 0$ 的码是否存在? 1949 年香农(Shannon)的信道编码定理对此作了肯定的回答。

§ 1.4 信道编码定理

信道编码定理 每个信道具有确定的信道容量 C , 对任何小于 C 的码率 R , 存在有速率 R 码长为 n 的分组码及 (n_0, k_0, m) 卷积码, 若用最大似然译码, 则随着码长的增加其译码错误概率 p 可任意小, 即

$$p \leq A_b e^{-nE_b(R)}$$

和

$$p \leq A_c e^{-(m+1)n_0 E_c(R)} = A_c e^{-n_c E_c(R)} \quad (1.4.1)$$

式中, A_b 和 A_c 为大于 0 的系数, $E_b(R)$ 和 $E_c(R)$ 为正实函数, 称为**误差指数**, 它与 R 、 C 的关系如图 1-15 所示。图中, C_1 、 C_2 为**信道容量**, 且 $C_1 > C_2$ 。由信息论的基本知识可知, 在高斯白噪声信道时, 信道容量

$$C = W \log_2 \left[1 + \frac{P_s}{WN_0} \right] (\text{bit/s}) \quad (1.4.2)$$

式中, W 是信道所能提供的带宽, $P_s = E_s/T$ 是信号功率, E_s 是信号能量, T 是分组码信号的持续时间即信号宽度, P_s/W 是单位频带的信号功率, N_0 是单位频带的噪声功率, $P_s/(WN_0)$ 是信噪比。

由式(1.4.1)和图 1-15 可看出, 信道容量 C 、码长 n 和错误概率 p 之间的转换关系。为了满足一定误码率 p 的要求, 可用以下两类方法实现。

一是增加信道容量 C , 从而使 $E(R)$

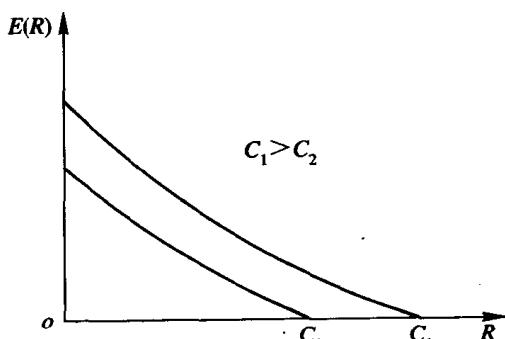


图 1-15 $E(R)$ 与 R 的关系

增加。由 C 的表示式(1.4.2)可知, 增加 C 的方法可以采用如加大系统带宽或增加信噪比的方法来达到。例如, 采用调频、调相等宽带调制方法; 增加发射机的功率; 应用高增益天线; 采用分集接收及低噪声器件等方法。这些措施是从根本上改善信道、增加信道容量、减少误码率的方法, 是通信设计工作者经常采用的传统方法。

另一种方法是在 R 一定下, 增加分组码长 n (也就是增加分组信号持续时间 T), 可使 p 随 n 的增加而呈指数下降。但由于码长 n 的增加, 当 R 保持一定时, 可能发送的码字数 2^k 指数增加, 从而增加了译码设备的复杂性。这种方法就是信道编码定理所指出减少误码率的另一方向, 它为通信设计工作者提供了一条新的途径。下面通过几个具体例子说明 R 保持一定时, 随着 n 的增加, 可使 p 下降。

例 1.3 设有一个随机产生二进制序列的信源和一个转移概率 $p_e = 0.1$ 的 BSC。如果不编码, 则传送消息的误码率仍为 0.1。现以 $R=1/2$ 的码率进行编码, 考察一下编码效果。

首先, 按 $k=2$ 分组, 则编出码的长度 $n=4$, 构成一个(4, 2)分组码。编码规则如下: 设两个信息元为 c_3, c_2 , 两个校验元为 c_1, c_0 , 则

$$\begin{aligned}c_1 &= c_2 \\c_0 &= c_3 + c_2\end{aligned}$$

编出的 4 个许用码组为:(0000), (1001), (0111), (1110)。其余 12 个四重为禁用码组。

译码按表 1-1 所示的译码表进行。译码表的第一行是许用码组, 禁用码组则在其它行, 列于许用码组之下, 接收到的四重落在哪一列就相应地译为最上面一行所对应的信息码。每一种可能接收的四重, 在译码表中仅出现一次。

表 1-1 (4, 2)码译码表

| 信息组 | 0 0 | 1 0 | 0 1 | 1 1 |
|------------------|------|------|------|------|
| 码 字 | 0000 | 1001 | 0111 | 1110 |
| 禁 用 码 组 | 1000 | 0001 | 1111 | 0110 |
| 0100 | 1101 | 0011 | 1010 | |
| 0010 | 1011 | 0101 | 1100 | |

由译码表可知, 若发送码字在传送过程中前三位的任一位发生错误, 接收的四重必位于该码字所在列中, 因而正确译码。因此正确译码概率

$$Q = (1 - p_e)^4 + 3p_e(1 - p_e)^3 = 0.879$$

码字的错误译码概率

$$P = 1 - Q = 0.12$$

由此可得译码后的误码率

$$p = 1 - \sqrt{0.879} = 0.063$$

可以看到, 编码使误码率由 0.1 下降到 0.063。

如果保持传信率 $R=1/2$ 不变, 把信息组的长度 k 加大到 3, 构成一个(6, 3)码。设信息组为 c_5, c_4, c_3 , 相应地 3 个校验元为 c_2, c_1, c_0 , 按下述校验规则求校验元:

$$c_2 = c_3 + c_4$$

$$c_1 = c_3 + c_5$$

$$c_0 = c_4 + c_5$$

得到 8 个码字及其译码表，如表 1-2 所示。由该译码表知，发送码字的六位码元中，任一位出现错误均可纠正。第一位和第四位同时出现错误时也能正确译码。故三位信息组完全正确译出的概率为

表 1-2 (6, 3) 码编译码表

| 消息 | m_1 000 | m_2 001 | m_3 010 | m_4 011 | m_5 100 | m_6 101 | m_7 110 | m_8 111 |
|-------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| 码字 | 000000 | 001110 | 010101 | 011011 | 100011 | 101101 | 110110 | 111000 |
| | 000001 | 001111 | 010100 | 011010 | 100010 | 101100 | 110111 | 111001 |
| | 000010 | 001100 | 010111 | 011001 | 100001 | 101111 | 110100 | 111010 |
| 有一位错误 的字 | 000100 | 001010 | 010001 | 011111 | 100111 | 101001 | 110010 | 111100 |
| | 001000 | 000110 | 011101 | 010011 | 101011 | 100101 | 111110 | 110000 |
| | 010000 | 011110 | 000101 | 001011 | 110011 | 111101 | 100110 | 101000 |
| | 100000 | 101110 | 110101 | 111011 | 000011 | 001101 | 010110 | 011000 |
| 有两位错误 的字 | 100100 | 101010 | 110001 | 111111 | 000111 | 001001 | 010010 | 011100 |

$$Q = (1 - p_e)^6 + 6p_e(1 - p_e)^5 + p_e^2(1 - p_e)^4 = 0.8923$$

相应的信息元误码率

$$p = 1 - \sqrt[3]{0.8923} = 0.037$$

由此可见，随着 n 的增加，在保持 $R=1/2$ 不变时，可使误码率越来越小，这正是信道编码定理指出的结果。但也必须看到，随着 n 的增加，其译码设备的复杂性成指数增加。我们研究纠错编码的意义就在于：保持一定的信息传输效率条件下，通过编译码来降低误码率以实现可靠通信，并且要求译码器尽可能简单。

应用纠错码后，若仍要求传输信息的速率不变，则必然使信道的带宽 W 增加。设原来的信息传输速率为 600 bit/s，如用 $R=1/2$ 码，则要求信道传输速率为 1200 bit/s，一般要求信道带宽增加一倍（通常增加 $(n-k)/k$ 倍）。因此，纠错码主要应用于功率受限而带宽不太受限的信道中。

在极限情况 $n \rightarrow \infty$ 时，要求带宽 $W \rightarrow \infty$ 。根据计算，此时只要求信噪比 $E_b/N_0 > -1.6$ dB，就可实现高斯白噪声信道下的无误传输。这就是带宽无限高斯白噪声信道的极限传输能力，称为 Shannon 限。图 1-16 给出了未编码的 PSK 与应用各类纠错码后，误码率与信噪比之间的关系曲线。由此图看到：应用 $R=1/2$ 的 (24, 12) 分组码后，在误码率为 10^{-5} 时比未编码的大约可节省 3 dB 的功率（称编码增益）；而用 $R=1/2$ ，约束度 N 为 8 的卷积码软判决维特比(Viterbi)译码，在误码率为 10^{-5} 时，大约可节省 5 dB 的功率。一般情况下，应用纠错码后，大约能得到从零点几到几个 dB 程度不等的编码增益。

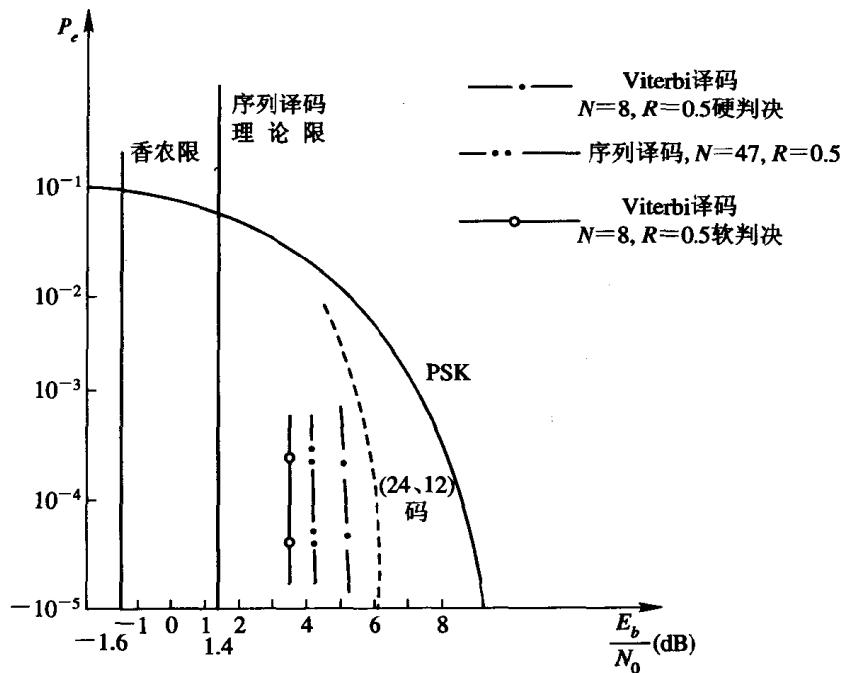


图 1-16 各种码的性能比较

参 考 文 献

- [1] C. E. Shannon, "A mathematical theory of communication", B. S. T. J., 27, pp. 379—423, 625—656, July 1948.
- [2] S. Lin and D. J. Costello, Error Control Coding: Fundamentals and Applications. Prentice-Hall, 1983.
中译本 王育民, 王新梅译:《差错控制编码: 基础和应用》, 人民邮电出版社, 1986.
- [3] R. G. Gallager, Information Theory and Reliable Communication, Wiley, New York, 1968.
- [4] 王新梅:《纠错码浅说》, 人民邮电出版社, 1976.
- [5] 王新梅:《纠错码与差错控制》, 人民邮电出版社, 1989.

第二章 代数初步

分组码是建立在码的代数结构基础上的。本章介绍的代数中的某些有关基本知识是学习以后各章所必需的。这一章首先介绍数论中某些最基本的知识，然后介绍群、格、环、域的基本概念，最后简单介绍线性空间和矩阵的一些基本知识。

§ 2.1 整数的一些基本知识

一、基本概念

这一节介绍的一些有关整数的基本概念，如初等运算、欧几里德(Euclid)运算、同余等，它们在编码理论中有着广泛的应用。

在数列： $\cdots, -2, -1, 0, 1, 2, 3, \cdots$ 中的数是整数，其有 ∞ 多个，它分为负整数和正整数两部分，0 和正整数又称自然数。按照它们能否被其它整数除尽，又分为素数和合数。

定义 2.1.1 一个大于 1 的正整数，只能被 1 和它本身整除，而不能被其它整数整除，这样的正整数称为素数(或称质数)。

例如，2, 3, 5, 7, 9, 11, 13, 17, 19, \cdots 都是素数，共有 ∞ 多个。

定义 2.1.2 一个正整数除了能被 1 和本身整除以外，还能被另外的正整数整除，这样的正整数称为合数。

例如，4, 6, 8, 9, \cdots 都是合数，也有 ∞ 多个。

因此，全体正整数又可分为：1、全体素数和全体合数三部分。

定义 2.1.3 设 a, b 是整数， $b \neq 0$ ，如果有一个整数 c ，使得 $a = bc$ ，则 a 叫做 b 的倍数， b 称为 a 的因数或约数。

由 $a = bc$ 可知， b 能整除 a ，或 a 能被 b 整除，我们用 $b | a$ 表示之。若 b 不能整除 a ，则用 $b \nmid a$ 表示。

定义 2.1.4 如果一个正整数 a 有一个因数 b ，而 b 又是素数，则称 b 是 a 的素因数。

例如， $30 = 5 \times 6$ ，5 是素数，所以称 5 是 30 的素因数，而 6 则不是 30 的素因数。

定理 2.1.1 任何正整数 a 均可表示成素因数之乘积，若不考虑这种乘积的顺序，则这种表示是唯一的，即

$$a = p_1^{r_1} p_2^{r_2} \cdots p_r^{r_r} \quad (2.1.1)$$

式中， p_1, p_2, \cdots, p_r 为素数， r_1, r_2, \cdots, r_r 是正整数。

例如，60 可以表示成 $5^1 \cdot 2^2 \cdot 3^1$ 。 $12 = 3^1 \cdot 2^2, \cdots$ 。该定理的证明可参阅有关数论的书籍^[1]。

二、整数的初等运算及性质

整数可作加、减、乘、除四则运算。对加、乘运算有如下性质：

1° 有封闭性。对所有整数 a, b, c, d 有 $a+b=c, ab=d$ 。

2° 结合律成立

$$a + (b + c) = (a + b) + c$$

$$a \cdot (b \cdot c) = (a \cdot b) \cdot c \quad (\cdot 表示乘法)$$

3° 加法有恒等元 0。即对任意整数 a 有： $0+a=a$ 。

4° 加法有逆元。对每一整数 a , 存在有唯一逆元 $(-a)$, 使 $(-a)+a=0$ 。

5° 乘法消去律。任意三个整数 a, b, c , 若 $c \neq 0$, 则由 $ac=bc$, 可得到 $a=b$ 。

6° 乘法有单位元 1。即对任意整数 a 有： $1 \cdot a=a$ 。

7° 加法和乘法之间有如下分配律：

$$a \cdot (b+c) = a \cdot b + a \cdot c$$

8° 关于加法和乘法的如下交换律分别成立：

$$a+b=b+a \quad a \cdot b=b \cdot a$$

此外, 还满足以下三条最基本的逻辑规则:

(1) 自反律, $a=a$;

(2) 对称律, 若 $a=b$, 则 $b=a$;

(3) 传递律, 若 $a=b, b=c$, 则 $a=c$ 。

所有其它性质诸如负负得正、乘法单位元唯一、加法消去律, 以及 $a+x=b$ 有唯一解等, 都可由上述性质和逻辑规则推出。例如, $a+x=b$ 有唯一解, 现证明如下:

取 $x=(-a)+b$, 则由 8°、4° 可得

$$a+x=a+(-a)+b=((a)+(-a))+b=0+b=b$$

可知, $x=(-a)+b$ 是 $a+x=b$ 的解。若 y 是它们的另一个解, 则由 $a+x=b$, 可得 $a+y=b$; 由传递律得 $a+x=a+y$; 由加法消去律得 $x=y$; 由此证明解是唯一的。可见, 对加法而言, $a+x=b$ 不仅有解, 且有唯一解; 但对乘法而言, $a \cdot x=b$ 不一定有解, 也就是说除 1 以外, 在整数范围内, 乘法不一定有逆元, 但有如下的欧几里德除法(又称带余除法)。

三、欧几里德除法

定理 2.1.2(欧几里德除法) 设 b 是正整数, 则任意正整数 $a>b$ 皆可唯一地表示成

$$a = qb + r \quad 0 \leqslant r < b \tag{2.1.2}$$

证明 设用 $\lfloor x \rfloor$ 表示实数 x 的整数部分, 则下述不等式显然成立:

$$\lfloor x \rfloor \leqslant x < \lfloor x \rfloor + 1 \tag{2.1.3}$$

现在以 b 为单位度量 a , 取 $q=\lfloor a/b \rfloor$, 由式(2.1.3)得

$$\lfloor a/b \rfloor \leqslant a/b < \lfloor a/b \rfloor + 1$$

或

$$0 \leqslant a - b\lfloor a/b \rfloor < b \tag{2.1.4}$$

令

$$r = a - b\lfloor a/b \rfloor = a - bq \tag{2.1.5}$$

由式(2.1.4)可得

$$0 \leq r < b$$

把式(2.1.5)进行移项可得

$$a = bq + r$$

下面证明唯一性。若还有另一表达式

$$a = b q_1 + r_1 \quad 0 \leq r_1 < b$$

代入式(2.1.2)得

$$\begin{aligned} a - a &= b(q_1 - q) + (r_1 - r) \\ 0 &= b(q_1 - q) + (r_1 - r) \end{aligned}$$

或

$$-b(q_1 - q) = r_1 - r$$

所以 b 能整除 $r - r_1$, 也就是说 $r - r_1$ 是 b 的倍数, 但因 r, r_1 均小于 b , 故 $r - r_1 < b$, 因此只有 $r = r_1, q = q_1$ 。■

四、最大公约数与欧几里德算法

定义 2.1.5 同时除尽 a, b, \dots, l (不全为 0) 的正整数, 称为 a, b, \dots, l 的公约数, 其中的最大者称为最大公约数, 用 (a, b, \dots, l) 或 $\text{GCD}(a, b, \dots, l)$ 表示。若 $(a, b, \dots, l) = 1$, 则称 a, b, \dots, l 互素。

例 4, 8, 24 中, 2、4 都可同时除尽 4, 8, 24, 所以都是它们的公约数, 但 4 在 2、4 两个数中是最大的, 所以 $(4, 8, 24) = 4$ 。而 $(7, 6, 8) = 1$, 所以 7、6、8 为互素。

下面介绍两两互素的概念。

若 a, b, \dots, l 中每一对数的最大公约数均为 1, 则称 (a, b, \dots, l) 中的数为两两互素。

例如 $(7, 6, 8) = 1$, 7、6、8 三个数为互素, 但不是两两互素。因为 $(7, 6) = 1$, $(6, 8) = 2$ 。而 $(7, 6, 5) = 1$, $(7, 6) = (6, 5) = (5, 7) = 1$, 故 7、6、5 三个数是两两互素。

求两个正整数的最大公约数, 除了用因子分解法外, 也可以用欧几里德除法逐步进行, 下面定理给出了这点。

定理 2.1.3 给定两正整数 a, b , 且 $a > b$, 若

$$a = bq + r \tag{2.1.6}$$

则 $(a, b) = (b, r)$ 。

证明 设 $A_{a,b}$ 是 a, b 所有公约数的集合, $A_{b,r}$ 是 b, r 所有公约数的集合。若 $c | a, c | b$, 则由式(2.1.6)可看出 $c | r$ 。这说明如果 c 是 a, b 的公约数, 则一定得出 c 是 b, r 的公约数。由此可知 $A_{b,r}$ 集合一定包含或等于 $A_{a,b}$ 集合, 即

$$A_{a,b} \subseteq A_{b,r} \tag{2.1.7}$$

反之, 若 $c | r, c | b$, 则由式(2.1.6)可得 $c | a$ 。因此, 若 c 是 r, b 的公约数, 则一定可知 c 必是 a, b 的公约数, 所以

$$A_{a,b} \supseteq A_{b,r} \tag{2.1.8}$$

由式(2.1.7)和式(2.1.8)可得

$$A_{a,b} = A_{b,r}$$

即 $A_{a,b}$ 集合与 $A_{b,r}$ 集合完全相同, 因此它们当中的最大值也必相等, 故 $(a, b) = (b, r)$ 。■

例 2.1 求 $(150, 42) = ?$

解 由欧几里德除法可知:

$$\begin{array}{ll} 150 = 3 \times 42 + 24 & 0 < 24 < 42 \\ 42 = 1 \times 24 + 18 & 0 < 18 < 24 \\ 24 = 1 \times 18 + 6 & 0 < 6 < 18 \\ 18 = 3 \times 6 & \end{array}$$

所以, 由定理 2.1.3 可知:

$$(150, 42) = (42, 24) = (24, 18) = (18, 6) = 6$$

可把上述运算简化如下:

$$\begin{array}{ll} \overbrace{(150, 42)}^3 \text{ 余 } 24 & 150 = 3 \cdot 42 + 24 \\ \overbrace{(42, 24)}^1 \text{ 余 } 18 & 42 = 1 \cdot 24 + 18 \\ \overbrace{(24, 18)}^1 \text{ 余 } 6 & 24 = 1 \cdot 18 + 6 \\ \overbrace{(18, 6)}^3 = 6 & \end{array}$$

若求 a, b, c 的最大公约数, 则可先求 (a, b) , 再求 (a, b) 与 c 的最大公约数 $((a, b), c)$ 。

例 2.2 求 $12, 6, 3$ 的最大公约数。

解 先求 $(12, 6) = 6$, 再求 $(6, 3) = 3$ 。所以 $(12, 6, 3) = 3$ 。

我们不但可以用上述方法求得两个正整数的最大公约数, 而且也可将最大公约数化成原来两个数的线性组合, 即

$$(a, b) = Aa + Bb$$

如上例中 $(150, 42) = 6$ 可以化成

$$6 = A \cdot 150 + B \cdot 42$$

的形式。这一点可以从例 2.1 表示式中倒数第一个式子往上推得到:

$$\begin{aligned} (150, 42) &= 6 = 24 - 1 \cdot 18 = 24 - 1 \cdot (42 - 1 \cdot 24) \\ &= 2 \cdot 24 - 1 \cdot 42 = 2(150 - 3 \cdot 42) - 42 \\ &= 2 \cdot 150 - 6 \cdot 42 - 42 = 2 \cdot 150 - 7 \cdot 42 \\ &= 2 \cdot 150 + (-7) \cdot 42 \end{aligned}$$

由此可推出如下的一般情况。

定理 2.1.4(欧几里德算法) 给定任意正整数 a, b , 必存在有整数 A, B 使

$$(a, b) = Aa + Bb$$

证明 由上面的例子看出, 必须建立余式之间的递推关系。令 $a=r_{-2}, b=r_{-1}$, 且设 $0 < b < a$, 即 $0 < r_{-1} < r_{-2}$ 。反复进行欧几里德除法便得到下述的一系列关系式:

$$\begin{array}{ll} r_{-2} = a_0 r_{-1} + r_0 & 0 < r_0 < r_{-1} \\ r_{-1} = a_1 r_0 + r_1 & 0 < r_1 < r_0 \\ r_0 = a_2 r_1 + r_2 & 0 < r_2 < r_1 \\ \vdots & \vdots \end{array}$$

$$\begin{array}{ll}
r_k = a_{k+2}r_{k+1} + r_{k+2} & 0 < r_{k+2} < r_{k+1} \\
\vdots & \vdots \\
r_{n-3} = a_{n-1}r_{n-2} + r_{n-1} & 0 < r_{n-1} < r_{n-2} \\
r_{n-2} = a_nr_{n-1} + r_n & 0 = r_n < r_{n-1}
\end{array}$$

由此可知，该一系列关系式中的余数 r ，构成单调下降序列：

$$r_n < r_{n-1} < \cdots < r_{k+1} < r_k < \cdots < r_2 < r_1 < r_0 < r_{-1} < r_{-2}$$

因此，上述除法过程不可能无止境地进行下去，而必然进行到某一个 n 而结束，于是 $r_n = 0$ 。

由定理 2.1.3 可知：

$$(r_{-2}, r_{-1}) = (r_{-1}, r_0) = (r_0, r_1) = \cdots = (r_{n-1}, r_n) = r_{n-1}$$

现在我们找出 b_k 和 B_k 序列，使

$$r_{n-1} = b_k r_k + B_k R_k \quad k = -2, -1, 0, 1, 2, \dots, n-1 \quad (2.1.9)$$

在此，令 $R_k = r_{k-1}$ ，由式(2.1.9)可知

$$r_{k-2} = a_k r_{k-1} + r_k \quad r_k = r_{k-2} - a_k r_{k-1}$$

所以

$$r_k = R_{k-1} - a_k r_{k-1}$$

因此

$$\begin{aligned}
b_k r_k + B_k R_k &= b_k (R_{k-1} - a_k r_{k-1}) + B_k R_{k-1} \\
&= (B_k - a_k b_k) r_{k-1} + b_k R_{k-1} \\
&= b_{k-1} r_{k-1} + B_{k-1} R_{k-1}
\end{aligned} \quad (2.1.10)$$

这里

$$b_{k-1} = B_k - a_k b_k \quad B_{k-1} = b_k \quad (2.1.11)$$

注意到 $r_{n-1} = 1 \cdot r_{n-1} + 0 \cdot R_{n-1}$ ，取 $b_{n-1} = 1$ ， $B_{n-1} = 0$ ，利用逆推关系式(2.1.10)，便可得到下述逆推关系：

$$\begin{aligned}
r_{n-1} &= 1 \cdot r_{n-1} + 0 \cdot R_{n-1} \\
&= b_{n-1} r_{n-1} + B_{n-1} R_{n-1} \\
&= b_{n-2} r_{n-2} + B_{n-2} R_{n-2} \\
&\vdots \\
&= b_{-1} r_{-1} + B_{-1} R_{-1} = b_{-1} r_{-1} + B_{-1} r_{-2} \\
&= B_{-1} \cdot a + b_{-1} \cdot b = b_0 \cdot a + b_{-1} \cdot b
\end{aligned} \quad (2.1.12)$$

因此

$$(a, b) = r_{n-1} = b_0 \cdot a + b_{-1} \cdot b = Aa + Bb \quad (2.1.13)$$

由式(2.1.11)的逆推关系，由初始值 $b_{n-1} = 1$ ， $b_n = B_{n-1} = 0$ 出发，便可最终得 $b_0 = A$ ， $b_{-1} = B$ 。 ■

仍以 $(150, 42) = 6$ 为例说明，在此 $r_{-2} = 150$ ， $r_{-1} = 42$ ，并且 $a_0 = 3$ ， $r_0 = 24$ ， $a_1 = 1$ ， $r_1 = 18$ ， $a_2 = 1$ ， $r_2 = 6$ ， $a_3 = 3$ ， $r_3 = 0$ ，因此 $n=3$ ，由式(2.1.11)可计算：

$$\begin{array}{ll}
b_{k-1} = b_{k+1} - a_k b_k & \\
b_{-1} = b_1 - a_0 b_0 & b_0 = b_2 - a_1 b_1 \\
b_1 = b_3 - a_2 b_2 & b_2 = b_4 - a_3 b_3
\end{array}$$

由 $b_{n-1}=b_2=1$, $b_n=b_3=0$ 出发求得:

$$\begin{aligned} b_1 &= b_3 - a_2 b_2 = 0 - 1 \cdot 1 = -1 \\ b_0 &= 1 - 1 \cdot (-1) = 1 + 1 = 2 \\ b_{-1} &= -1 - 3 \cdot 2 = -1 - 6 = -7 \end{aligned}$$

所以, $(150, 42) = 2 \cdot 150 + (-7) \cdot 42$ 。

上面介绍的欧几里德除法与欧几里德算法, 在编码理论中起着很重要的作用, 今后要经常用到。

下面我们讨论最大公约数的其它性质。(这些性质的证明, 由于比较简单, 读者可自己求证, 或参考有关书籍。)

- 1° 若 $r|a$, $r|b$, 则 $r|(a, b)$ 。
- 2° 设 m 为任意正整数, 则 $(ma, mb) = m(a, b)$ 。
- 3° 设 $r|a$, $r|b$, 则 $(a/r, b/r) = (1/r)(a, b)$ 。
- 4° 若 $(a, b) = 1$, 则 $(ac, b) = (c, b)$ 。

五、最小公倍数

定义 2.1.6 设 a, b 为任意两个正整数, 若有一整数 M 使 $a|M$, $b|M$, 则称 M 是 a, b 的公倍数, 其中最小的正公倍数称为最小公倍数, 记为 $[a, b]$ 或 $\text{LCM}(a, b)$ 。

例 设 $a=2$, $b=3$ 。显然, $6, 12, 18, 24, \dots$ 共有 ∞ 多个, 它们都是 2、3 的公倍数, 而 6 是其中最小的, 所以

$$[2, 3] = 6 \quad \text{或} \quad \text{LCM}(2, 3) = 6$$

可用如下方法求最小公倍数: 把 a, b 进行素因子分解, 观察它们有什么不同的素因子, 挑出次数最高的共同素因子项和不同素因子项, 作它们的乘积, 即为最小公倍数。

例 2.3 求 $[108, 28] = ?$ $[198, 240, 360] = ?$

解 (1) 因为 $108 = 2^2 \times 3^3$, $28 = 2^2 \times 7$, 所以

$$[108, 28] = 2^2 \times 3^3 \times 7 = 756$$

(2) 因为 $198 = 2 \times 3^2 \times 11$, $240 = 2^4 \times 3 \times 5$, $360 = 2^3 \times 3^2 \times 5$, 所以

$$[198, 240, 360] = 2^4 \times 3^2 \times 11 \times 5 = 7920$$

下面介绍最小公倍数的性质。

1° a, b, c 中若两两互素, 则 $[a, b, c] = abc$ 。

2° a, b 的最小公倍数除尽 a, b 的一切公倍数, 若 m 是 a, b 的公倍数, 则 $[a, b] | m$ 。所以 $m = q[a, b]$ 。

3° 若 a, b 都是正整数, 它们的最大公约数为 (a, b) , 而最小公倍数为 $[a, b]$, 则

$$ab = a, b \quad (2.1.14)$$

证明 3° ab 是 a 和 b 的公倍数, 所以由性质 2° 可得 $[a, b] | ab$, 或 $ab = q[a, b]$ 。现在只要证明 q 是最大公约数 (a, b) 即可。

要证明 $q = (a, b)$, 只要证明 a, b 的一切公约数都除尽 q 。为此首先证明 q 是公约数, 然后再证明它是最大公约数。

由 $ab = q[a, b]$ 可得:

$$[a, b] = ab/q \quad a/q = [a, b]/b \quad b/q = [a, b]/a \quad (2.1.15)$$

显然 $[a, b]/b$ 与 $[a, b]/a$ 都是正整数，因此 $a/q, b/q$ 也是正整数。所以 $q|a, q|b, q$ 是 a, b 的公约数。

下面证明 q 是最大公约数。设 g 是 a, b 的另一公约数， $g|a, g|b$ 。令 $m=ab/g$ ，由于 g 是公约数，所以 $a/g, b/g$ 都是正整数。由 $m=a\times b/g=a/g\times b$ 可知， m 是 a, b 的公倍数，由性质 2° 可知

$$[a, b] | m \quad \text{或} \quad m/[a, b] = t$$

t 是一个正整数。又由式(2.1.15)有

$$m/[a, b] = ab/(g[a, b]) = ab/(g(ab/q)) = q/g = t$$

所以 $q=gt$ 。因为 g 是 a, b 的公约数，而 gt (当 t 取不同值时) 是 a, b 的任一公约数，任一公约数能除尽 q ，所以 q 是最大公约数。即 $q=(a, b)$ ，因此

$$ab = q[a, b] = (a, b)[a, b]$$

我们可以利用性质 3° 以及欧几里德除法，求出最小公倍数。

例 2.4 求 $[24\ 871, 3\ 468] = ?$

解 先利用欧几里德除法求出 $(24\ 871, 3\ 468)$:

$$\begin{array}{ll} 24\ 871 = 3\ 468 \times 7 + 595 & 3\ 468 = 595 \times 5 + 493 \\ 595 = 493 + 102 & 493 = 102 \times 4 + 85 \\ 102 = 85 + 17 & 85 = 17 \times 5 \end{array}$$

所以 $(24\ 871, 3\ 468) = 17$ 。利用性质 3° 可得

$$[24\ 871, 3\ 468] = (24\ 871 \times 3\ 468)/17 = 5\ 073\ 684$$

如果要求两个以上正整数的最小公倍数，则可先求两个数的最小公倍数，然后再求此最小公倍数与另一数的最小公倍数，依次进行，就可求出任意几个正整数的最小公倍数。

例 2.5 求 $[513, 135, 3\ 114] = ?$

解 $(513, 135) = 27$

$$[513, 135] = 513 \times 135/27 = 2\ 565$$

再求 $[2\ 565, 3\ 114]$ ， $(2565, 3114) = 9$ ，所以

$$[2\ 565, 3\ 114] = 2\ 565 \times 3\ 114/9 = 887\ 490$$

因此

$$[513, 135, 3\ 114] = 887\ 490$$

六、同余和剩余类的概念

定义 2.1.7 若两整数 a, b 被同一正整数 m 除时，有相同的余数：

$$a = q_1m + r, \quad b = q_2m + r \quad 0 \leq r < m$$

则称 a, b 关于模 m 同余，记为

$$a \equiv b \pmod{m}$$

例 2.6 $25 \equiv 4 \pmod{7}$

$$16 \equiv 2 \pmod{7}$$

由同余概念可将全体整数加以分类，把余数相同的归一类，即由

$$a = gm + r \quad 0 \leq r < m$$

可得到 a 的余项 r 。由于 r 可取 $0, 1, 2, \dots, m-1$ 中的任一个，因此 r 共有 m 个值，也就是

有 m 个剩余类。可知对所有的整数，必属于这 m 个剩余类中一个，今后用 \bar{r} 表示余数为 r 的剩余类。

例如： $m=7$ ，则对全体整数可如下划分：

$$\begin{aligned}\bar{r}=\bar{0} & \cdots, -14, -7, 0, 7, 14, 21, \dots \\ \bar{r}=\bar{1} & \cdots, -13, -6, 1, 8, 15, 22, \dots \\ \bar{r}=\bar{2} & \cdots, -12, -5, 2, 9, 16, 23, \dots \\ \bar{r}=\bar{3} & \cdots, -11, -4, 3, 10, 17, 24, \dots \\ \bar{r}=\bar{4} & \cdots, -10, -3, 4, 11, 18, 25, \dots \\ \bar{r}=\bar{5} & \cdots, -9, -2, 5, 12, 19, 26, \dots \\ \bar{r}=\bar{6} & \cdots, -8, -1, 6, 13, 20, 27, \dots\end{aligned}$$

所以，可把全体整数按模7分成七类： $\bar{0}, \bar{1}, \dots, \bar{6}$ 。一般而言，若模数为 m ，则全体整数可按模 m 划分成 m 类， $\bar{0}, \bar{1}, \dots, \bar{m-1}$ ，或用 $\{0\}, \{1\}, \dots, \{m-1\}$ 表示之。称这样的一类为模 m 的同余或剩余类。

如同整数一样，剩余类之间可以定义加法和乘法运算。

$$\bar{a} + \bar{b} = \overline{a+b}, \quad \bar{a} \cdot \bar{b} = \overline{a \cdot b} \pmod{m}$$

例如， $m=7$ ，则

$$\bar{0} + \bar{1} = \overline{1+0} = \bar{1}, \quad \bar{2} \cdot \bar{4} = \overline{2 \cdot 4} = \bar{8} = \bar{1} \pmod{7}$$

定理2.1.5 若 $a_1 \equiv b_1 \pmod{m}$, $a_2 \equiv b_2 \pmod{m}$ ，则

$$(1) a_1 \pm a_2 \equiv b_1 \pm b_2 \pmod{m}$$

$$(2) a_1 \cdot a_2 \equiv b_1 \cdot b_2 \pmod{m}$$

证明

(1) 由定义可知：

$$a_1 = q_1 m + r \quad b_1 = q_2 m + r$$

所以

$$a_1 - b_1 = (q_1 - q_2)m + 0 \equiv 0 \pmod{m}$$

或

$$a_1 - b_1 = mt_1$$

同理可得： $a_2 - b_2 = mt_2$ 。因此

$$(a_1 \pm a_2) - (b_1 \pm b_2) = m(t_1 \pm t_2) \equiv 0 \pmod{m}$$

所以

$$a_1 \pm a_2 \equiv b_1 \pm b_2 \pmod{m}$$

(2) 由

$$(a_1 - b_1) = mt_1 \quad (a_2 - b_2) = mt_2$$

$$a_1 = mt_1 + b_1 \quad a_2 = mt_2 + b_2$$

可知

$$\begin{aligned}a_1 a_2 &= (mt_1 + b_1)(mt_2 + b_2) = m^2 t_1 t_2 + mt_1 b_2 + mt_2 b_1 + b_1 b_2 \\ &= m(mt_1 t_2 + t_1 b_2 + t_2 b_1) + b_1 b_2 = mt + b_1 b_2\end{aligned}$$

所以

$$a_1 a_2 - b_1 b_2 = mt \equiv 0 \pmod{m}$$

$$a_1 a_2 \equiv b_1 b_2 \pmod{m}$$

■

由上看到剩余类中任一 \bar{a} 类的性质并不随 $\{a\}$ 中所取代表元的不同而不同。如上例中 $m=7$, $\{1\}$ 这一类所代表的整数是 $\cdots -13, -6, 1, 8, 15, 22, \cdots$, 可以用 $\{8\}$ 也可以用 $\{-6\}$ 代表这一类, 但为了简单起见, 今后我们均用 $0, 1, \dots, m-1$ 这几个数作为模 m 的 m 个剩余类的代表元, 即 $\{0\}, \{1\}, \dots, \{m-1\}$ 。

可以证明模 m 的同余类中, m 个元素对于加法和乘法具有和整数相同的性质, 即具有 8 条性质和 3 条逻辑公理。

§ 2.2 群和格的基本概念

近世代数是纠错码理论的数学基础, 它主要关心代数运算本身的性质和规律, 被作为运算对象——元素, 可以是整数、多项式、矩阵等等。但它们对乘法、加法等运算具有共同的性质。为了了解什么是近世代数, 我们首先介绍一些最基本的概念, 如代数系统和代数运算。

定义 2.2.1 满足一定规律或定律的系统称为代数系统。且在该系统中有:

- (1) 有一群元素 a, b, c, \dots , 构成一个集合;
- (2) 在元素集合中有一等价关系;
- (3) 在集合中定义了一个或数个运算, 通过运算建立起元素之间的关系;
- (4) 有一组假定。

我们下面所要介绍的群、格、环、域等都是代数系统。

如图 2-1 所示, A, B 代表由元素组成的两个不同集合。以下的运算都称为代数运算:

$A+B$: 代表属于 A 和 B 的所有元素的集合;

AB : 代表既属于 A 又属于 B 的元素的全体;

$A-B$: 代表属于 A 不属于 B 的元素全体;

$A=B$: 集合 A 的每一成员都是 B 中的成员, 反之亦然。即 $A \supseteq B, B \supseteq A$, 因此 $A=B$ 。式中, “ \supseteq ”表示“包含”的意思(“ \subset ”表示“被包含”)。

$a \in A$: 表示 a 元素属于集合 A 。“ \in ”表示属于的意思。

定义 2.2.2 G 为非空集合(至少有一个元素), 假如在 G 内任何两个有序元素 a, b , 都有 G 内的一个元素 c 与之对应, 则称在 G 内定义了一个代数运算“ \circ ”, 记为 $a \circ b=c$ 。

必须注意: 在不同代数系统中, 代数运算有的能交换, 有的不能交换。即在有的代数系统中 $a \circ b=b \circ a$, 但在某些代数系统中 $a \circ b \neq b \circ a$ 。

一、同态与同构

同态与同构是说明两个元素集合在代数性质或代数结构上的一种等价关系。它们在研究群、环、域的性质时很有用。

定义 2.2.3 设 A 和 B 是两个集合, 如果存在某个对应法则 φ , 使对任一 $a \in A$, 都能

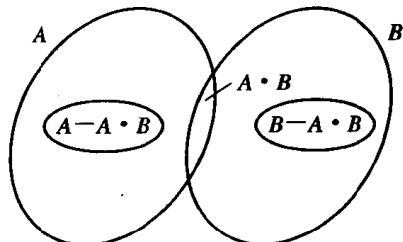


图 2-1 两个元素集合的关系图

唯一确定 B 中一个元素 b 与之对应，则称 φ 是 A 到 B 的一个单值映射，记为

$$\varphi: a \longrightarrow b \quad \text{或} \quad a \xrightarrow{\varphi} b \quad \text{或} \quad \varphi(a) = b$$

称 $\varphi(a)$ 或 b 是 a 在 φ 之下的像，称 a 是原像。

定义 2.2.4 设 φ 是集合 A 到集合 B 的单值映射，如果对任一 $b \in B$ ，必然存在有 $a \in A$ ，使 $b = \varphi(a)$ ，则称 φ 是 A 到 B 的满射，或说 φ 是 A 到 B 上的映射。

定义 2.2.5 如果对集合 A 中不同的元素，在 φ 之下在集合 B 中有不同的像，即当且仅当 $a_1 = a_2$ 时， $\varphi(a_1) = \varphi(a_2)$ ，则称 φ 是 A 到 B 的单射。如果 φ 既是满射又是单射，就称为双射或一一对应。

在双射或一一对应下， B 中任一元素 b 必是 A 中某一元素 a 的像，且 a 是 b 的唯一的原像，特别当 A 和 B 都是有限集合时，所含元素的个数必相等。

定义 2.2.6 设 φ 是集合 A 到 A 自身的映射，则称 φ 是 A 中的变换。 A 到 A 的满射称为满变换，单射称为单变换，一一映射称为一一变换或置换。如果一种变换 τ ，它保持 A 中任一元素不变， $a \xrightarrow{\tau} a$ ，对所有 $a \in A$ ，则这种变换 τ 称为恒等变换或恒等置换。显然，恒等变换必为一一变换。

例如， x 为实数时，则 x 与 $\log_b x$ 之间就是存在有一一映射的关系。

定义 2.2.7 设 φ 是集合 A 到 B 的映射，如果它满足条件

$$\varphi(a_1 \circ a_2) = \varphi(a_1) \circ \varphi(a_2) \quad a_1, a_2 \in A, \varphi(a_1), \varphi(a_2) \in B$$

则称 φ 是 A 到 B 的同态映射，集合 A 与 B 同态。如果同态映射 φ 又是单射，则称为同构映射，集合 A 与 B 同构。若 φ 是 A 到自身的同构映射，则称为自同构。

由上定义可知 A 与 B 同构，它们元素之间必存在有一一对应关系，但同态却不一定有这种关系。例如，实数集合 $\{x\}$ 与对数集合 $\{\log_b x\}$ 之间是同构映射，而整数集合与剩余类集合 $\{\bar{0}, \bar{1}, \dots, \bar{m}\}$ 之间是同态映射。

二、群

定义 2.2.8 设 G 是非空集合，并在 G 内定义了一种代数运算，若满足下述公理：

- (1) 有封闭性。对任意 $a, b \in G$ ，恒有 $a \circ b \in G$ 。
- (2) 结合律成立。对任意 $a, b, c \in G$ ，有 $(a \circ b) \circ c = a \circ (b \circ c)$ 。
- (3) G 中有一恒等元 e 存在，对任意 $a \in G$ ，有 $e \in G$ ，使 $a \circ e = e \circ a = a$ 。
- (4) 对任意 $a \in G$ ，存在有 a 的逆元 $a^{-1} \in G$ ，使 $a \circ a^{-1} = a^{-1} \circ a = e$ 。

则称 G 构成一个群。

上述定义中， G 的运算“ \circ ”可以是通常的乘法或加法。若为乘法，则恒等元称为单位元；若为加法，则恒等元记为 0 ； a 的逆元记为 $-a$ 。群中元素的个数，称为群的阶。若群中元素个数有限，称为有限群；否则，称无限群。

例 G1： 整数全体，按通常加法构成群，这是一个无限群。但通常的乘法不构成群。

例 G2： 偶数全体，对加法构成群（无限群），对乘法不构成群。

例 G3： 实数全体对加法构成群。除 0 元素以外的实数全体对普通乘法也构成群，此时群的单位元 $e=1$ 。这两个群均为无限群。

例 G4： 所有 n 阶矩阵的全体 M ，对矩阵的加法构成群，单位元为 0 矩阵。对矩阵乘

法, 若为满秩矩阵, 则构成群; 否则不是。

例 G5: 模 m 的剩余类全体: $\bar{0}, \bar{1}, \dots, \bar{m-1}$, 在模 m 加法运算下构成群。如 $m=4$, 在模 4 加法运算下构成群, 此时的加法运算如下表所示。称该表为模 4 加法表, 该群是阶为4的有限群。但在模 4 乘法运算下, 并不构成群。

| + | $\bar{0}$ | $\bar{1}$ | $\bar{2}$ | $\bar{3}$ |
|-----------|-----------|-----------|-----------|-----------|
| $\bar{0}$ | $\bar{0}$ | $\bar{1}$ | $\bar{2}$ | $\bar{3}$ |
| $\bar{1}$ | $\bar{1}$ | $\bar{2}$ | $\bar{3}$ | $\bar{0}$ |
| $\bar{2}$ | $\bar{2}$ | $\bar{3}$ | $\bar{0}$ | $\bar{1}$ |
| $\bar{3}$ | $\bar{3}$ | $\bar{0}$ | $\bar{1}$ | $\bar{2}$ |

一般而言, 若群中的运算是加法, 则简称为**加群**; 若为乘法则简称为**乘群**。如例 G1、G2、G3 是加群, 例 G4 是乘群。

若群 G 中, 对任何 $a, b \in G$, 有 $a \circ b = b \circ a$, 则称 G 为**交换群、可换群或阿贝尔群 (Abel 群)**。

上述一系列群中, 除例 G4 中的矩阵在乘法运算下不是阿贝尔群外, 其余例中都是可换群。下面定理给出了群的一些重要性质。

定理 2.2.1 群 G 中恒等元是唯一的、群中每个元素的逆元也是唯一的。

证明 先证恒等元唯一。设有两个恒等元 $e, e' \in G$, 则根据恒等元性质可知: $e \circ e' = e$, $e \circ e' = e'$, 所以 $e \circ e' = e = e'$ 。

下面再证明逆元的唯一性。设 $g \in G$, 它若有两个逆元 $g_1^{-1}, g_2^{-1} \in G$, 则

$$g_1^{-1} = e \circ g_1^{-1} = (g_2^{-1} \circ g) \circ g_1^{-1} = g_2^{-1} \circ (g \circ g_1^{-1}) = g_2^{-1} \circ e = g_2^{-1} \quad \blacksquare$$

定理 2.2.2 若 $a, b \in G$, 则 $(a \circ b)^{-1} = b^{-1} \circ a^{-1}$ 。

证明 $(a \circ b) \circ (b^{-1} \circ a^{-1}) = a \circ (b \circ b^{-1}) \circ a^{-1} = a \circ e \circ a^{-1} = a \circ a^{-1} = e$

由此可知: $(b^{-1} \circ a^{-1})$ 是 $(a \circ b)$ 的逆元, 因此 $(a \circ b)^{-1} = b^{-1} \circ a^{-1}$ 。 ■

推论 2.2.1 若 $a, b, c, \dots, f \in G$, 则

$$(a \circ b \circ c \circ \dots \circ f)^{-1} = f^{-1} \circ \dots \circ c^{-1} \circ b^{-1} \circ a^{-1}$$

定理 2.2.3 给定 G 中任意两个元素 a 和 b , 下述方程 $a \circ x = b$ 和 $y \circ a = b$ 在 G 中有唯一解。

证明 $a \circ x = b$ 的解是 $x = a^{-1} \circ b \in G$ 。因为

$$a \circ a^{-1} \circ b = e \circ b = b$$

而 $y \circ a = b$ 的解是 $y = b \circ a^{-1}$ 。因为

$$b \circ a^{-1} \circ a = b \circ e = b$$

下面证明解的唯一性。若在 $a \circ x = b$ 的方程中, 除了 $x = a^{-1} \circ b$ 外, 还有另外一个解 x_1 , 使 $a \circ x_1 = b$, 则把上式两边乘以 a 的逆元 a^{-1} , 因而有

$$a^{-1} \circ a \circ x_1 = a^{-1} \circ b$$

由此可得

$$e \circ x_1 = x_1 = a^{-1} \circ b = x$$

同理可证 $y \circ a = b$ 方程解的唯一性。 ■

推论 2.2.2 群 G 中消去律成立。即由 $a \circ x = a \circ y$ 可推得 $x = y$ 。

证明 在 $a \circ x = a \circ y$ 方程两边乘以 a^{-1} , 则

$$\begin{aligned} a^{-1} \circ a \circ x &= a^{-1} \circ a \circ y \\ e \circ x &= e \circ y \\ x &= y \end{aligned}$$

■

定义 2.2.9(半群) 一非空集合 S , 若元素之间定义了一种运算“ \circ ”, 且满足:

- (1) 封闭性成立。对任何 $a, b \in S$, 恒有 $a \circ b \in S$ 。
- (2) 结合律成立。对任何 $a, b, c \in S$, 有 $a \circ (b \circ c) = (a \circ b) \circ c$ 。

则称 S 为半群。

定义 2.2.10(Monoid 弱群) 一非空集合 M , 若元素之间定义了一种运算“ \circ ”, 且满足:

- (1) 封闭性成立。对任何 $a, b \in M$, 恒有 $a \circ b \in M$ 。
- (2) 结合律成立, 即 $a \circ (b \circ c) = (a \circ b) \circ c$ 。
- (3) 有恒等元 e 。对任何 $a \in M$, 有 $a \circ e = e \circ a = a$ 。

则称 M 是一个 Monoid 或弱群。

例如, 整数在乘法运算下就是一个 Monoid, 而不是一个群。正整数在加法运算下是一个半群, 因为无恒等元。

以后在不致引起错误的情况下, 为了简单, 我们把元素集合中的运算“ \circ ”省略不写, 即把 $a \circ b$ 写作 ab 。

三、置换群与对称群

正如前面所说的置换是有限集合 A 到自身的一一映射, 例如有限集合 $A = \{1, 2, 3\}$, 其中一个置换 φ_1 是

$$\varphi_1(1) = 1, \varphi_1(2) = 2, \varphi_1(3) = 3; \text{ 即 } 1 \rightarrow 1, 2 \rightarrow 2, 3 \rightarrow 3.$$

显然, φ_1 是一个恒等置换。另两个置换分别是:

$$\varphi_2(1) = 2, \varphi_2(2) = 3, \varphi_2(3) = 1; \text{ 即 } 1 \rightarrow 2, 2 \rightarrow 3, 3 \rightarrow 1$$

$$\varphi_3(1) = 3, \varphi_3(2) = 1, \varphi_3(3) = 2; \text{ 即 } 1 \rightarrow 3, 2 \rightarrow 1, 3 \rightarrow 2$$

它们可以表示成:

$$\varphi_1 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}, \varphi_2 = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}, \varphi_3 = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$$

或简写成:

$$\varphi_1 = (1, 2, 3), \varphi_2 = (2, 3, 1), \varphi_3 = (3, 1, 2)$$

置换之间的运算“ \circ ”如下:

$$\varphi_2 \circ \varphi_3 = (2, 3, 1) \circ (3, 1, 2) = (1, 2, 3) = \varphi_1$$

即先进行 φ_2 置换, 然后再进行 φ_3 置换, 可知 φ_2, φ_3 互为逆元。一般而言

$$\varphi_i \circ \varphi_j = \varphi_j \circ \varphi_i \quad (2.2.1)$$

在上述置换运算下, 3 个置换元素集 $S = \{\varphi_1, \varphi_2, \varphi_3\}$ 组成了 1 个三阶有限群, 像这种由置换元素 φ 组成的有限群称为置换群。

由置换定义可知, 3 个元素的所有置换有 $3! = 6$ 种, 除了上面 3 个置换外, 还有以下 3

个：

$$\varphi_4 = (1, 3, 2), \varphi_5 = (3, 2, 1), \varphi_6 = (2, 1, 3)$$

可以检验这6个置换组成的集合 $S_3 = \{\varphi_1, \varphi_2, \dots, \varphi_6\}$ ，在上述的置换运算下，组成一个群，这是一个六阶有限群。

定义 2.2.11 由 n 个元素组成的有限集合 $A = \{1, 2, \dots, n\}$ 到自身的所有置换有 $n!$ 个，这 $n!$ 个置换组成的集合 $S_n = \{\varphi_1, \varphi_2, \dots, \varphi_{n!}\}$ ，在置换运算下，是一个 $n!$ 阶有限群，称它为对称群。

可知对称群一定是置换群，但置换群不一定是对称群，如3个元素的例中，共有 $3! = 6$ 种置换，其中 $\{\varphi_1, \varphi_2, \varphi_3\}$ 组成了一个置换群，但不是对称群，仅只有 $S_3 = \{\varphi_1, \varphi_2, \varphi_3, \dots, \varphi_6\}$ 集合才称为对称群。

对称群其实是 n 边形几何图形的对称变换所组成的群，如3个元素 $A = \{1, 2, 3\}$ 所组成的对称群 S_3 就是三角形3个顶点的对称变换，如图 2-2 所示。可以证明任何一个有限群都同构于某一置换群。

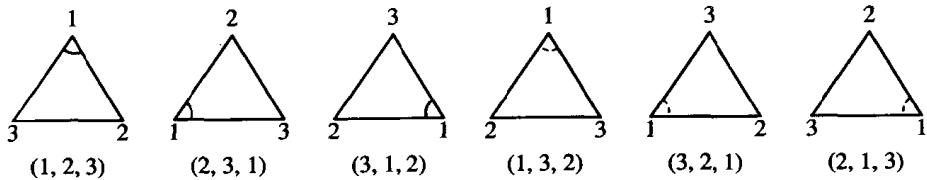


图 2-2 3 个元素组成的对称群

四、格(Lattice)

格是一类加群，它的元素是某一集合中的离散点。

定义 2.2.12 若加群的元素是欧几里德(欧氏)空间中离散点(n 重，矢量)的集合，则这种加群称为格。格中每个元素称为格点。

实数(R) n 维欧氏空间 R^n 中离散点的集合 $\{(a_1, a_2, \dots, a_n); a_i \in R\}$ 在 n 维矢量相加运算下，组成了一实数格 Λ 。它其实就是在矢量相加和相减运算下的一个线性空间。

复数(C) n 维欧氏空间 C^n 中的离散点集合 $\{(a_1 + ib_1, \dots, a_n + ib_n); a_i, b_i \in R\}$ ，在复数相加和相减运算下，也是一个格，称为复数格 C^n 。

整数(Z) n 维欧氏空间 Z^n 中离散点的集合 $\{(Z_1, Z_2, \dots, Z_n); Z_i \in Z\}$ 在矢量相加和相减运算下组成了一整数格 Z^n 。

格是欧氏空间上点集组成的群，因此除了有与群一样的代数性质外，还有空间上的以下几何性质：

(1) 最小均方距离 $d_m^2(\Lambda)$ 。格 Λ 中格点之间的最小均方距离定义为

$$d_m^2(\Lambda) = \min_{a, b \in \Lambda} \|a - b\|^2 \quad (2.2.2)$$

式中， $a = (a_1, a_2, \dots, a_n)$ 和 $b = (b_1, b_2, \dots, b_n)$ 是 Λ 中的两个格点。 $\|x\|^2$ 表示格点 $x = (x_1, x_2, \dots, x_n)$ 的范数，计算如下：

$$\|x\|^2 = (x_1^2 + x_2^2 + \dots + x_n^2) \quad (2.2.3)$$

所以

$$\|\mathbf{a} - \mathbf{b}\|^2 = ((a_1 - b_1)^2 + (a_2 - b_2)^2 + \cdots + (a_n - b_n)^2)$$

格是加群，必满足封闭性，因此格的最小均方距离就等于格中非零格点的最小范数。如在整数格 \mathbf{Z}^n 中， $d_m(\mathbf{Z}^n) = 1$ 。

(2) 格的基本体积 $V(\Lambda)$ 。格 Λ 的基本体积(体积)定义为每一格点在 n 维空间中所占的体积，或者是每一单位体积中格点数目的倒数。

如在 \mathbf{Z}^n 格中，可以把 n 维整数空间划分为每一边均为 1 的 n 维立方体，每一立方体中只含一个格点，故 $V(\mathbf{Z}^n) = 1$ 。

除了以上的几何性质外，实数格 Λ 还有以下一些结构上的性质：

(1) 标量乘(数乘)。若 r 是任何一个实数，而 $\lambda \in \Lambda$ ，则集合 $\{r\lambda, r \in \mathbf{R}\}$ 是格，也就是在格 Λ 中矢量 λ 的所有标量乘 $r\lambda$ 组成了一个格。

(2) 正交变换。若 T 是 N 维空间中的任一标量正交变换集， $\lambda \in \Lambda$ ，则集合 $\{T\lambda; T \in T\}$ 是一个格。即在格 Λ 中，矢量 λ 的所有 $T\lambda$ 组成了一个格。显然，该性质是性质(1)的推广。

(3) 笛卡儿(Cartesian)积。 Λ 与自身的 M —折笛卡儿积是集合 $\{(\lambda_1, \lambda_2, \dots, \lambda_M); \lambda_i \in \Lambda\}$ 。可知笛卡儿积的集合中每一元素是 Mn 维的一个格点，这种格用 Λ^M 表示。

例如， \mathbf{Z}^n 格是全体整数集合 \mathbf{Z} 与自身的 n —折笛卡儿积。对任何 r 和 n ， $r\mathbf{Z}^n$ 格是 \mathbf{Z}^n 格的一种标量乘变种。

与编码关系最密切的一种标量正交变换是旋转运算 R_2 ，用以下的 2×2 阶矩阵定义：

$$R_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

所以， $R_2\mathbf{Z}^2$ 是 \mathbf{Z}^2 的一种变形，它是 \mathbf{Z}^2 旋转 45° 得到的，也就是格点的每个分量旋转 45° 。旋转运算的结果使格点的范数成为偶数。

我们也可定义一个 $2n$ 维的旋转运算，如：

$$R_4 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

也就是在 $2n=4$ 维空间中，对每一格点($2n$ 重)的每对坐标(分量)旋转 45° 。 R_{2n} 运算的结果总使格点的范数成为偶数。

§ 2.3 环与域的基本概念

一、环

定义 2.3.1 非空元素集合 R 中，若定义了两种代数运算加和乘，且满足下述公理：

- (1) 集合 R 在加法运算下构成阿贝尔群；
- (2) 乘法有封闭性，对任何 $a, b \in R$ ，有 $ab \in R$ ；
- (3) 乘法结合律成立，且如和乘之间有分配律，即对任何 $a, b, c \in R$ ，有 $a(b+c) = ab+ac$ 和 $(b+c)a = ba+ca$ ，则称 R 是一个环。

由上述定义可知，环是在集合中定义了两种运算的代数系统。必须注意的是，在环 R 中乘法无恒等元，即无单位元，当然更无逆元存在。若环有单位元存在，则称它为有单位元环。

若环 R 对乘法满足交换律，即对任何元素 $a, b \in R$ ，恒有 $ab = ba$ ，则称此环为可换环或交换环。

例 R1 全体整数构成环，用 \mathbf{Z} 表示。

例 R2 全体偶数构成环。

例 R3 某一整数 m 的倍数全体构成环，如 3 的倍数全体 $\dots, -3, 0, 3, 6, 9, \dots$ ，构成一个环。

例 R4 模整数 m 的全体剩余类构成环，称此环为剩余类环，用 Z_m 表示。如模 $m=7$ 所构成的全体剩余类： $\bar{0}, \bar{1}, \bar{2}, \bar{3}, \bar{4}, \bar{5}, \bar{6}$ 构成环 Z_7 ，且为可换环。

例 R5 实系数多项式全体构成环。

例 R6 n 阶方阵全体构成环。

下面定理说明了环中的一些性质。

定理 2.3.1 任何 $a, b \in R$ ，有

(1) $a0 = 0a = 0$ ；

(2) $a(-b) = (-a)b = -ab$ 。

除了以上性质外，环中还有许多较特殊的性质。

(1) 环中可以有零因子。设 $a, b \in R$ ，且 $a \neq 0, b \neq 0$ ，若 $ab = 0 \in R$ ，则 a, b 为零因子，称有零因子的环为有零因子环。

例 R7 模 $m=6$ 的剩余类环 Z_6 有 6 个元素： $\bar{0}, \bar{1}, \bar{2}, \bar{3}, \bar{4}, \bar{5}$ 。 $\bar{3} \neq \bar{0}, \bar{2} \neq \bar{0}$ ，但 $\bar{2} \cdot \bar{3} = \bar{6} = \bar{0}$ 。所以 $\bar{2}, \bar{3}$ 是模 6 剩余类环的零因子，该环称有零因子环。

在有零因子环中乘法消去律不成立。如上例中： $\bar{2} \cdot \bar{3} = \bar{0}, \bar{0} \cdot \bar{3} = \bar{0}, \bar{2} \cdot \bar{3} = \bar{0} \cdot \bar{3}$ ，由于 $\bar{2} \neq \bar{0}$ ，因此等式两边的 $\bar{3}$ 不能消去。

但在无零因子环中，乘法消去律成立，这种环称为整环 (domain)，如全体整数组成的环。

(2) 有单位元的，每个非零元素有逆元的、非可换的环称为除环 (商域、斜域或 Skew 域)。

例如，所有 n 阶实数满秩矩阵全体和全 0 矩阵构成除环。

二、域

除上面所讲的群、格和环以外，域在编码理论中起着关键作用。域是定义了两种代数运算的系统。

定义 2.3.2 非空元素集合 F ，若在 F 中定义了加和乘两种运算，且满足下述公理：

(1) F 关于加法构成阿贝尔群。其加法恒等元记为 0。

(2) F 中非零元素全体对乘法构成阿贝尔群。其乘法恒等元(单位元)记为 1。

(3) 加法和乘法间有如下分配律：

$$a(b + c) = ab + ac$$

$$(b + c)a = ba + ca$$

则称 F 是一个域。

由此可见域是一个可换的、有单位元的、非零元素有逆元的环，且域中一定无零因子。若在 F 中有零因子，则由 $a, b \in F, a \neq 0, b \neq 0$, 得出 $ab = 0$, 因此 $0 = a^{-1}0 = a^{-1}(ab) = (a^{-1}a)b = b$, 这与原来假设 $b \neq 0$ 有矛盾，因此 $ab \neq 0$ 。

例 F1 有理数全体、实数全体、复数全体对加法、乘法都分别构成域，分别称**有理数域**、**实数域**和**复数域**。且这 3 个域中的元素个数有无限多个，所以称它们为**无限域**。

今后称域中元素的个数为域的**阶**。元素个数有限的域称**有限域**，用 $GF(q)$ 或 F_q 表示 q 阶有限域。有限域也称为**伽罗华(Galois)域**，在编码理论中起着非常重要的作用。

例 F2 0、1 两个元素按模 2 加和模 2 乘构成域。该域中只有两个元素，记为 $GF(2)$ 或 F_2 。

定理 2.3.2 设 p 为素数，则整数全体关于模 p 的剩余类： $\bar{0}, \bar{1}, \bar{2}, \dots, \bar{p-1}$ ，在模 p 运算下（模 p 相加和相乘），构成 p 阶有限域 $F_p(GF(p))$ 。

证明 由前面已知，模 m 整数（ m 不一定为素数）剩余类集合构成交换环 Z_m ，现在只需证明当 $m=p$ 为素数时，非 0 元素有逆元即可。1 为单位元，因为 p 为素数，因此任何小于 p 的数 a 和 p 均互素。所以，由欧几里德算法可知：

$$(a, p) = 1 = Aa + Bp$$

在等式两边对 p 取模，则

$$1 \equiv Aa \pmod{p}$$

所以

$$\bar{1} = \bar{A}\bar{a}$$

也就是剩余类中任一元素 a 均有逆元 $a^{-1} = \bar{A}$ 。 ■

例 F3 以 $p=3$ 为模的剩余类全体： $\bar{0}, \bar{1}, \bar{2}$ 构成一个三阶有限域 $GF(3)$ ，它们的模 3 加法和乘法运算表如下所示：

| $+$ | $\bar{0}$ | $\bar{1}$ | $\bar{2}$ | \times | $\bar{0}$ | $\bar{1}$ | $\bar{2}$ |
|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| $\bar{0}$ | $\bar{0}$ | $\bar{1}$ | $\bar{2}$ | $\bar{0}$ | $\bar{0}$ | $\bar{0}$ | $\bar{0}$ |
| $\bar{1}$ | $\bar{1}$ | $\bar{2}$ | $\bar{0}$ | $\bar{1}$ | $\bar{0}$ | $\bar{1}$ | $\bar{2}$ |
| $\bar{2}$ | $\bar{2}$ | $\bar{0}$ | $\bar{1}$ | $\bar{2}$ | $\bar{0}$ | $\bar{2}$ | $\bar{1}$ |

§ 2.4 子群、正规子群和商群

一、子群

定义 2.4.1 若群 G 的非空子集 H 对于 G 中定义的代数运算也构成群，则称 H 为群 G 的**子群**。

例如，偶数全体构成的群是全体整数所构成加群的一个子群。

每个群一定有两个子群，群自己和由一个恒等元所构成的群，称这两个子群为**假子群**或**平凡子群**。除这两个假子群以外，所有其它子群称为**真子群**。

定理 2.4.1 群 G 的非空子集 H 为 G 的子群的充要条件是：

- (1) 若 $a \in H$, $b \in H$, 则 $ab \in H$;
- (2) 若 $a \in H$, 则 a 的逆元 $a^{-1} \in H$.

证明 若 H 为子群, 则(1)、(2)自然成立。反之, 由(1)可知, H 关于 G 的代数运算封闭。由(2)知, H 中的每一元素均有逆元。因为 H 是 G 的子集, 所以 G 中的结合律在 H 中同样适用, 又因 $a \in H$, $a^{-1} \in H$ 及 $aa^{-1}=e \in H$ (由(1)条件), H 中有单位元, 故 H 是一个群。 ■

可把上述定理中的两个条件归并成一个条件。

定理2.4.2 H 是 G 的子群的充要条件是: 对任何 $a, b \in H$, 恒有 $ab^{-1} \in H$ 。

证明 若 H 是 G 的子群, 则上述条件自然成立。反之, 假定上述条件成立, 现在要证 H 是 G 的子群。

因为 H 为非空子集。所以必有 $a \in H$, 再令 $b=a$, 由假设条件可得 $aa^{-1} \in H$, 而 $aa^{-1}=e$, 所以 $e \in H$, H 中有单位元存在。

其次, 由 $a \in H$, $e \in H$, 由条件可推出 $ea^{-1} \in H$, 而 $ea^{-1}=a^{-1}$, $a^{-1} \in H$, 故 H 中的每一元素 a 均有逆元 a^{-1} 存在。

最后, 由 $a \in H$, $b \in H$ (从而 $b^{-1} \in H$), 再根据条件可知 $a(b^{-1})^{-1}=ab \in H$, 所以在 H 中封闭性成立。由此可知, H 满足群的公理, 所以 H 是一子群。 ■

二、陪集

若 G 中有子群 H , 则可用 H 把 G 划分成等价类, 如下所示:

设群 G 的元素是: $g_1, g_2, g_3, g_4, \dots$;

子群 H 的元素是: h_1, h_2, h_3, \dots ;

| | | | | |
|----------------|----------|----------|----------|-----|
| $h_1 = e$ | h_2 | h_3 | \cdots | 子群 |
| $g_1h_1 = g_1$ | g_1h_2 | g_1h_3 | \cdots | 左陪集 |
| $g_2h_1 = g_2$ | g_2h_2 | g_2h_3 | \cdots | 左陪集 |
| \vdots | \vdots | \vdots | \vdots | |
| $g_nh_1 = g_n$ | g_nh_2 | g_nh_3 | \cdots | 左陪集 |

↑
陪集首

定义2.4.2 设 H 是群 G 的子群, g 是 G 中的任一元素, 将 g 左(右)乘 H 中的每一元素, 便得到 $gh(hg)$ 的元素集合, 记 $gH(Hg)$, 称之为它的子群 H 在群 G 中的一个左(右)陪集, 称 e, g_1, g_2, \dots, g_n 为陪集首。

由定义可知, 若 $g=e \in H$, 则 $eH=H$ 。因此子群本身就是一个左(右)陪集。同样, 若 $b \in gH$, 即 $b=gh(h \in H)$, 则 $bH=ghH=g(hH)=gH$ 。这表明左(或右)陪集 gH 可由其中任一元素唯一确定。

从上可知, 陪集其实就是把 G 中的元素按子群 H 划分成等价类, 在同一陪集中都含有一个共同的元素 g_i (陪集首)。

在阿贝尔群中, 由于对任何 $g, h \in G$, $gh=hg$, 因此左陪集等于右陪集。

例如, 全体整数构成一个加群, 以 m 为倍数的整数全体是其中的一个子群, 所以可以按此子群把全体整数划分陪集。如 $m=5$, 则陪集表如下:

| | | | | | | | |
|-----------|---|---|----|----|-----|----------|---|
| 子群 H : | 0 | 5 | -5 | 10 | -10 | \cdots | 0 |
| $1 + h$: | 1 | 6 | -4 | 11 | -9 | \cdots | 1 |
| $2 + h$: | 2 | 7 | -3 | 12 | -8 | \cdots | 2 |
| $3 + h$: | 3 | 8 | -2 | 13 | -7 | \cdots | 3 |
| $4 + h$: | 4 | 9 | -1 | 14 | -6 | \cdots | 4 |

由此表可看出共有 5 个陪集。从这里看到，把全体整数按 $m=5$ 倍数的子群划分陪集，其实就是按模 5 把全体整数划分成 5 个剩余类： $\bar{0}, \bar{1}, \bar{2}, \bar{3}, \bar{4}$ 。在一般情况下得到 $\bar{0}, \bar{1}, \bar{2}, \bar{3}, \bar{4}, \dots, \bar{m-1}$ 个剩余类。

下面我们讨论陪集的性质。

引理 2.4.1 群 G 的两个元素 g_a, g_b 属于子群 H 的同一陪集的充要条件是 $g_a^{-1}g_b \in H$ 。

证明 若 g_a, g_b 属于 H 的同一陪集，则

$$g_a = g_i h_j \quad g_b = g_i h_k \quad j \neq k$$

g_i 为该陪集的陪集首。由此

$$g_a^{-1}g_b = (g_i h_j)^{-1} g_i h_k = h_j^{-1} g_i^{-1} g_i h_k = h_j^{-1} e h_k = h_j^{-1} h_k \in H$$

反之，若 $g_a^{-1}g_b \in H$ ，现证 g_a, g_b 属于 H 的同一陪集中。

$g_a^{-1}g_b \in H$ ，所以 $g_a^{-1}g_b = h' \in H$ ，两边同乘 g_a 得 $g_b = g_a h'$ 。因此，若设 $g_a = g_i h_i$ ，则 $g_b = g_i h_i h' = g_i h_i$ ，因为 $h_i \in H$ ，所以 g_a 与 g_b 都在以 g_i 为陪集首的陪集中。 ■

定理 2.4.3 两个陪集要么相等要么不相交。

证明 若两个陪集 g_1H 与 g_2H 相交，则必定有公共的元素 c ，即 H 存在有元素 h_1, h_2 ，使 $g_1h_1 = g_2h_2 = c$ ，则 $g_1h_1h_1^{-1} = g_2h_2h_1^{-1}$ ， $h_2h_1^{-1} = h$ ， $g_1 = g_2h$ 。所以， $g_2^{-1}g_1 = h \in H$ ，由引理 2.4.1 可知， g_2, g_1 必在同一陪集中，所以两个陪集完全相等。 ■

前面已谈到过，群中元素的个数可以是无限的，也可以是有限的。例如，所有整数全体所构成的加法群，其元素个数为无限的，是一无限群。若元素个数为有限的，则称为有限群，且称元素的个数为有限群的阶。同样，我们称子群中元素的个数为子群的阶。

由上面的讨论可以得出如下结论：

(1) 有限群 G 可以按子群 H 划分成有限个互不相交的陪集，且各陪集都具有相同的元素个数，即等于子群 H 的阶。

(2) 在划分陪集的以下阵列(称 Slepian 阵)中，若仅有 j 个陪集：

$$\begin{array}{cccccc} H: & 1 & h_1 & h_2 & \cdots & h_n \\ a_2H: & a_2 & a_2h_1 & a_2h_2 & \cdots & a_2h_n \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ a_{j-1}H: & a_{j-1} & a_{j-1}h_1 & a_{j-1}h_2 & \cdots & a_{j-1}h_n \end{array}$$

则 G 中的所有元素都在此阵列中，无一遗漏。若有一元素 $b \in G$ 不在该阵列中，则我们可再作陪集 bH ，若它与上述某一陪集重合，则 b 含于该陪集中，否则便得到第 $j+1$ 个陪集，而这与仅有 j 个陪集的假设相矛盾。因此， G 中所有元素都在该阵列中。所以，若 G 的阶为 N ， H 的阶为 n ，有 j 个陪集，则 $N = jn$ 。由此得到以下的定理。

格拉朗日(Lagranges)定理 有限群的子群的阶数，一定是整个群的阶数的因子。

例如, 模 $m=9$ 的剩余类全体: $\bar{0}, \bar{1}, \dots, \bar{8}$, 对模 9 加法运算构成一个群, 而 $\bar{0}, \bar{3}, \bar{6}$ 是它的一个子群, 以此子群划分陪集:

| | | | | |
|---------|---|-----------|-----------|-----------|
| H | : | $\bar{0}$ | $\bar{3}$ | $\bar{6}$ |
| $1+H$: | | $\bar{1}$ | $\bar{4}$ | $\bar{7}$ |
| $2+H$: | | $\bar{2}$ | $\bar{5}$ | $\bar{8}$ |

共有 3 个陪集。子群的阶为三, 是群的阶(等于九)的因子。

三、正规子群

定义 2.4.3 设 H 是 G 的一个子群, 若对每一个 $a \in G$, 恒有 $aH = Ha$, 则称 H 是 G 的一个正规子群或不变子群。

该定义表明, 群 G 的正规子群 H 是这样的一个子群: 对每一个 $a \in G$, 都有 $aH = Ha$ 。因此, 阿贝尔群的每一个子群都是正规子群。

定理 2.4.4 H 是 G 的正规子群的充要条件是: 对每一个 $a \in G$, 恒有 $a^{-1}ha \in H$, $h \in H$ 。

证明 若 H 是 G 的正规子群, 则由定义可知:

$$H = He = H(aa^{-1}) = Ha(a^{-1}) = aHa^{-1}$$

立即得到

$$aha^{-1} \in H \quad h \in H$$

反之, 若 $a^{-1}ha \in H$, 现证明 H 是正规子群。

因为对每一个 $h \in H$, 都有 $a^{-1}ha \in H$, 因此

$$a^{-1}Ha = H$$

两边左乘 a , 便得

$$aa^{-1}Ha = aH$$

所以

$$Ha = aH$$

例如, 全体整数在加法下构成群, 其中整数 m 的一切倍数所构成的一个子群 M , 就是正规子群。

定理 2.4.5 若 H 为群 G 的正规子群, 则 H 的全体陪集必构成群。

证明 首先, 定义陪集之间的乘法运算。设 $\bar{a}=aH$, $\bar{b}=bH$ 是 H 的两个陪集, 我们规定: 以 ab 为代表的元的陪集 $\bar{ab}=(ab)H$ 定义为陪集 aH 与 bH 之积, 即 $\bar{ab}=\bar{a}\bar{b}$ 。为表明这一规定的合理性, 我们必须证明, 如此规定的陪集 $(ab)H$ 并不因从陪集 aH 与 bH 中所选取代表元的不同而改变, 即它仅与陪集 aH 与 bH 自身有关, 而与所选的代表元是什么无关。事实上, 若分别从 aH 和 bH 中另选两个代表元, 设为 a_1 和 b_1 , 则可令

$$\begin{aligned} a_1 &= ah_1 & b_1 &= bh_2 & h_1h_2 &\in H \\ a_1b_1 &= ah_1bh_2 & & & & \end{aligned}$$

由于 H 是正规子群, 故 $bH=Hb$, 因而 $h_1b \in Hb=bH$, 所以可将 h_1b 写为

$$h_1b = bh_3 \quad h_3 \in H$$

因此

$$a_1b_1 = a(h_1b)h_2 = a(bh_3)h_2 = (ab)(h_3h_2) = abh$$

式中, $h=h_3h_2 \in H$ 。这表明 $a_1b_1 \in abH$, 故

$$(a_1b_1)H = abH$$

其次, 证明结合律成立。事实上, 由我们的规定可知:

$$\begin{aligned} (\bar{a}\bar{b})\bar{c} &= \bar{a}\bar{b}\bar{c} = \bar{a}\bar{b}\bar{c} \\ \bar{a}(\bar{b}\bar{c}) &= \bar{a}\bar{b}\bar{c} = \bar{a}\bar{b}\bar{c} \end{aligned}$$

所以

$$(\bar{a}\bar{b})\bar{c} = \bar{a}(\bar{b}\bar{c})$$

下面证明有单位元存在, 设 e 为群 G 的单位元, 则 $\bar{e}=eH=H$, 即为陪集集合的单位元, 因为 $\bar{e}\bar{a}=\bar{e}\bar{a}=\bar{a}$ 。

最后, 我们证明每一个陪集 $\bar{a}=aH$ 均有逆元 $\bar{a}^{-1}=a^{-1}H$ 。事实上

$$\bar{a}\bar{a}^{-1} = aHa^{-1}H = (aHa^{-1})H$$

因为 H 是正规子群 $aH=Ha$, 所以上式可写成

$$\bar{a}\bar{a}^{-1} = (Haa^{-1})H = H \cdot H = H = \bar{e}$$

所以, H 的全体陪集构成群。 ■

例如, 以 \mathbf{Z} 代表全体整数构成的加群, M 是以 m 的一切倍数所构成的一个正规子群。若 \mathbf{Z} 按 M 划分的陪集: $M, 1+M, 2+M, \dots, (m-1)+M$ 必构成一个群。如 $m=5$, 则全体整数按 5 的一切倍数构成的正规子群, 进行如下划分:

| | | | | | | |
|----------------|---|---|----|----|-----|-----|
| $\bar{0}=M:$ | 0 | 5 | -5 | 10 | -10 | ... |
| $\bar{1}=M+1:$ | 1 | 6 | -4 | 11 | -9 | ... |
| $\bar{2}=M+2:$ | 2 | 7 | -3 | 12 | -8 | ... |
| $\bar{3}=M+3:$ | 3 | 8 | -2 | 13 | -7 | ... |
| $\bar{4}=M+4:$ | 4 | 9 | -1 | 14 | -6 | ... |

由定理 2.4.5 可知, M 的 5 个陪集: $\bar{0}, \bar{1}, \bar{2}, \bar{3}, \bar{4}$ 关于剩余类加法构成群, 它是 5 阶子群。恒等元为 $\bar{0}$, 元素之间的运算如下表所示。

| + | $\bar{0}$ | $\bar{1}$ | $\bar{2}$ | $\bar{3}$ | $\bar{4}$ |
|-----------|-----------|-----------|-----------|-----------|-----------|
| $\bar{0}$ | $\bar{0}$ | $\bar{1}$ | $\bar{2}$ | $\bar{3}$ | $\bar{4}$ |
| $\bar{1}$ | $\bar{1}$ | $\bar{2}$ | $\bar{3}$ | $\bar{4}$ | $\bar{0}$ |
| $\bar{2}$ | $\bar{2}$ | $\bar{3}$ | $\bar{4}$ | $\bar{0}$ | $\bar{1}$ |
| $\bar{3}$ | $\bar{3}$ | $\bar{4}$ | $\bar{0}$ | $\bar{1}$ | $\bar{2}$ |
| $\bar{4}$ | $\bar{4}$ | $\bar{0}$ | $\bar{1}$ | $\bar{2}$ | $\bar{3}$ |

这样, 我们从陪集的角度, 把模 m 的剩余类构成一个阿贝尔群, 它是 \mathbf{Z} 的子群。

定义 2.4.4 设 H 是群 G 的正规子群, 于是把 H 的全体陪集所构成的群称为 G 关于 H 的商群, 记为 G/H 。

因此模 m 的剩余类群也可写成 \mathbf{Z}/M , 可知模 m 的剩余类群与商群 \mathbf{Z}/M 同构。

§ 2.5 子格与划分

格是加群, 因此在格中也有子格、陪集划分等概念。

一、子格

定义 2.5.1 若格 Λ 中的部分格点集合 Λ' 满足格的条件，则称 Λ' 为 Λ 的子格。

由此可知，子格其实就是 Λ 中的子群。例如 \mathbf{Z}^2 是一个格，则 $\mathbf{R}\mathbf{Z}^2$ 就是其中的一个子格。它就是二维整数平面上格点范数为偶数的点的集合，如图 2-3 中的黑点·集合。

又如 m 是任一整数，则 m 的整数倍所组成的格 $m\mathbf{Z}$ 就是 \mathbf{Z} 格的子格。它类似于整数加群中所有 m 的整数倍是整数加群的子群。因此子格 $m\mathbf{Z}$ 与模 m 的剩余类加群同构。可知 $2\mathbf{Z}$ 子格就是分量的范数为偶数的格点集合所组成，它与整数加群中的偶整数子群同构，也与模 2 剩余类加群同构。

二、划分(分割)

一个格 Λ 的陪集用 $\Lambda + \mathbf{C}$ 表示，这里 $\lambda \in \Lambda$ ，而 \mathbf{C} 是一个 n 重（类似于陪集首）。可知 $\Lambda + \mathbf{C}$ 这一陪集其实就是在 n 维欧氏空间中， Λ 平移一个 \mathbf{C} 。与定理 2.4.2 相同，若两个格点的差在 Λ 中则两个格点处在同一陪集。

在格 Λ 中，若以 Λ 的子格 Λ' 对 Λ 划分等价类，则称为按模 Λ' 对 Λ 进行划分（分割），用 $\Lambda' + \mathbf{C}$ 表示。因此每一个格均包含一个划分，用 Λ/Λ' 表示。与群一样， Λ/Λ' 也组成一个群。划分的阶数也就是 Λ/Λ' 中等价类的数目，用 $|\Lambda/\Lambda'|$ 表示。例如 $|\mathbf{Z}^2/\mathbf{R}\mathbf{Z}^2| = 2$ ，如图 2-3 所示。可知按 Λ' 划分的每一等价类，其实就是 Λ' 的一个陪集，从几何上讲就是 Λ' 的一种平移。

用 $[\Lambda/\Lambda']$ 表示 Λ' 划分 Λ/Λ' 的陪集表示。因此，格 Λ 中的每一格点 λ 可唯一地表示成 $\lambda = \lambda' + \mathbf{C}$ ， $\lambda \in \Lambda'$ ， $\mathbf{C} \in [\Lambda/\Lambda']$ 是等价类的代表元，也就是陪集表示中的陪集首。所以， $\lambda' = \lambda - \mathbf{C}$ ，与定理 2.4.2 的意义相同。称

$$\Lambda = \Lambda' + [\Lambda/\Lambda'] \quad (2.5.1)$$

为格 Λ 的陪集分解。如 \mathbf{Z}^2 按 $\mathbf{R}\mathbf{Z}^2$ 的陪集分解，可表示为

$$\mathbf{Z}^2 = \mathbf{R}\mathbf{Z}^2 + [\mathbf{R}\mathbf{Z}^2(10)]$$

相应于图 2-3 中黑点与白点集合。在 \mathbf{Z}^n 格中， $m\mathbf{Z}^n$ 是子格，它的陪集表示是

$$\mathbf{Z}^n = m\mathbf{Z}^n + [\mathbf{Z}^n/m\mathbf{Z}^n] \quad (2.5.2)$$

如果在子格 Λ' 中还含有子格 Λ'' ，则按 Λ'' 又可对 Λ' 进行划分，如果这种子格的子格一直存在的话，则这种划分可一直继续下去，得到一个划分链 $\Lambda/\Lambda'/\Lambda''/\dots$ ，其中后一子格是前一子格的子格（即 $\Lambda \supseteq \Lambda' \supseteq \Lambda'' \supseteq \dots$ ）。如 $\mathbf{Z}/2\mathbf{Z}/4\mathbf{Z}/\dots$ 就是一个无限的双向（每次划分只有两个陪集）划分链，它就是 \mathbf{Z} 格的陪集分解链。陪集分解链的概念在篱笆（Trellis）码的设计中起重要作用。

三、某些重要的格

这里简单介绍一下在编码理论中起着重要作用的一些格。

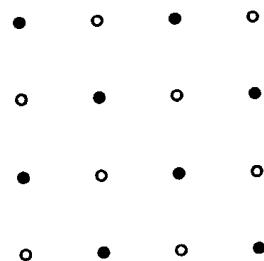


图 2-3 \mathbf{Z}^2 格和它的子格 $\mathbf{R}\mathbf{Z}^2$

(1) D_4 (Schlafli)格。 D_4 格是四维整数格，它由四维整数空间中偶范数的格点组成。因此，它是 \mathbf{Z}^4 的一个子格，其陪集分解表示为

$$D_4 = R\mathbf{Z}^4 + [R\mathbf{Z}^4 + (1010)]$$

也就是它由两对坐标均为偶范数的整数四重，与两对坐标为奇范数的整数四重联合。 D_4 格中格点之间的最小均方距离 $d_m^2(D_4) = 2$ 。可以证明它与距离为 2 的(4, 3)奇偶校验码相对应。

(2) 二进制格。二进制格是最有用的一类格，它与二进制分组码密切相关。

定义 2.5.2 一个 n 维实数格 Λ ，对某一整数 m ，若有一个整数格 $2^m \mathbf{Z}^n$ 作为子格，则称格 Λ 为二进制格， m 称为格的 2-层次。

可知 $\mathbf{Z}^n / \Lambda / 2^m \mathbf{Z}^n$ 是一个二进制格的陪集分解链。最有用的二进制格是 $m=1$ 和 $m=2$ 的格 $2\mathbf{Z}^n$ 和 $2^2 \mathbf{Z}^n$ ，分别称为模 2 格与模 4 格。由此可知，与(4, 3)码对应的 D_4 格就是一个模 2 格，其实所有 $(n, n-1)$ 奇偶校验码都是模 2 格，称为 D_n (Checkerboard)格。模 2 格的最小均方距离不大于 4，因为它有 $(2, 0, \dots, 0)$ 作为格点。

E_8 格 是另一个非常重要的模 2 格，它是八维实数空间中的一个子格，其最小均方距离 $d_m^2(E_8) = 4$ ，它与以后将讲到的距离为 4 的(8, 4)RM 码相对应。 E_8 格也称 Gosset 格。

里奇(Leech)格 Λ_{24} 是一个很重要的二十四维模 4 格，它的 $d_m^2(\Lambda_{24}) = 8$ ，它与以后讲到的扩张 Golay 码密切相关。

有关格及格与编码的详细关系请参阅文献[5]。

§ 2.6 线性空间和矩阵

一、线性空间

1. 线性空间和子空间 由一般的初等数学可知，平面上二维矢量的全体构成一个二维的矢量空间。空间中，三维矢量的全体构成三维矢量空间，现加以推广引入一般的线性空间概念。

定义 2.6.1 如果域 F 上的 n 重元素集合 V 满足下述条件时：

(1) V 关于加法构成阿贝尔群。

(2) 对 V 中任何元素 v 和 F 中任何元素 c ， $cv \in V$ 。我们称 V 中元素 v 为矢量(向量)， F 中元素 c 为纯量或标量，称乘 c 运算为数乘。

(3) 分配律成立，对任何 $u, v \in V, c, d \in F$ 恒有：

$$c(u + v) = cu + cv \quad (c + d)v = cv + dv$$

(4) 若 $c, d \in F, v \in V$ ，有：

$$(cd)v = c(dv) \quad 1 \cdot v = v \quad 1 \in F$$

则称 V 是域 F 上的一个 n 维线性空间或矢量空间，一般用 V_F^n 表示。

上述定义的线性空间，已作了两方面的推广：一是维数不限于二维或三维，而可以是任何 n 维；二是域不限于实数域，而可以是任何域。

例 L1 实数域 R 上的 n 重数组全体： $\{(a_1, a_2, \dots, a_n); a_i \in R\}$ 组成一线性空间 V_R^n 。

例 L2 GF(2)上的 n 重数组全体： $\{(a_1, a_2, \dots, a_n); a_i \in GF(2)\}$ 是一线性空间 V_2^n 。

例 L 3 系数取自实数域 R 上的次数小于 n 次的多项式全体: $\{(f_0 + f_1x + \dots + f_{n-1}x^{n-1}); f_i \in R\}$ 也组成一个线性空间。

上述例中的两个数组相加是对应的分量相加, 数乘 c 是对每位分量乘以 c , 这将在后面更详细讲到。

定义 2.6.2 若子集 $V_1 \in V$, 且满足线性空间的条件, 则称 V_1 是 V 的一个子空间。

例 L 4 例 L 1 中所有偶数分量 ($a_i = 2b \in R$) 所组成的线性空间就是实数 n 维数组线性空间中的一个子空间。

例 L 5 例 L 2 中的第 i 个分量 $a_i = 0$ 的 n 重数组全体是它的一个子空间。

例 L 6 平面上通过原点的任一条直线是平面上二维矢量空间中的一个子空间。

由上述定义可知, 要确定一个集合是否是子空间, 只要检验以下两点即可: (1)这个元素集合是否构成阿贝尔加群; (2)数乘是否封闭。

下面我们讨论域 F 上的线性空间中各个元素之间的运算规则以及线性空间中的其它重要概念, 特别是线性相关、线性无关等概念, 以及如何构成线性空间和线性子空间等问题。

定义 2.6.3 域 F 上的矢量 v_1, v_2, \dots, v_k 的线性组合定义为

$$u = b_1v_1 + b_2v_2 + \dots + b_kv_k \quad b_i \in F$$

例如线性空间 V : $\{(a_1a_2a_3); a_i \in GF(3)\}$ 是一个三重数组。设 v_1 是 (012) , v_2 是 (011) , v_3 是 (002) , $b_1=2$, $b_2=0$, $b_3=1$, 则

$$u = 2(012) + 0(011) + 1(002) = (021) + (000) + (002) = (020)$$

定理 2.6.1 线性空间 V 中矢量 v_1, v_2, \dots, v_k 的所有线性组合所构成的集合 S 是 V 的子空间。

证明 只要证明 S 中矢量加法和数乘封闭即可。令:

$$w = b_1v_1 + b_2v_2 + \dots + b_kv_k \quad b_i \in F, v_i \in V$$

$$u = c_1v_1 + c_2v_2 + \dots + c_kv_k \quad c_k \in F$$

是 S 中的任两个矢量, 则

$$\begin{aligned} w + u &= (b_1 + c_1)v_1 + (b_2 + c_2)v_2 + \dots + (b_k + c_k)v_k \\ &= d_1v_1 + d_2v_2 + \dots + d_kv_k \quad (b_i + c_i) = d_i \in F \\ aw &= ab_1v_1 + ab_2v_2 + \dots + ab_kv_k \\ &= e_1v_1 + e_2v_2 + \dots + e_kv_k \quad ab_i = e_i \in F \end{aligned}$$

仍在集合 S 中, 因此 S 是线性空间。 ■

定义 2.6.4 设 v_1, v_2, \dots, v_k 是线性空间 V 中的一组非全零矢量, 当且仅当存在有一组不全为零的纯量 c_1, c_2, \dots, c_k ($c_i \in F$; $i=1, 2, \dots, k$) 使

$$c_1v_1 + c_2v_2 + \dots + c_kv_k = 0$$

成立时, 则称这组矢量 **线性相关**。否则, 称这组矢量 **线性无关** (或称 **线性独立**)。

例 L 7 判断 $GF(2)$ 上的三重: $(010), (100), (001)$ 是否线性相关, 关键是看能否找到 3 个不全为 0 的数 $c_1, c_2, c_3 \in GF(2)$, 使

$$c_1(010) + c_2(100) + c_3(001) = (000)$$

成立。显然找不到, 故这 3 个矢量线性独立。

例 L 8 $GF(2)$ 上的三重: $(010), (100), (110)$ 是否线性相关?

$$1(010) + 1(100) + 1(110) = (000)$$

所以这 3 个矢量线性相关。

定理 2.6.2 线性空间中, 非零矢量 v_1, v_2, \dots, v_k 为线性相关的充要条件是: 存在有一个矢量 v_k , 它可以表示为其余矢量的线性组合。

该定理的证明比较简单, 读者可自行证明。下面介绍张成的概念。

定义 2.6.5 线性空间 V 中的每一矢量, 如果可以由其中的一组矢量集 S' 中的矢量线性组合生成, 则我们说 S' 张成了矢量空间 V 。

例 L9 GF(3)上二重数组: (00), (10), (01), (20), (21), (11), (12), (22), (02)组成一个矢量空间 V 。在其中挑选(11), (20)两个矢量组成子集 S' , 显然, 通过这两个矢量的线性组合, 可以生成 V 中的所有 9 个矢量。例如:

$$(21) = 1(11) + 2(20) = (11) + (10) = (21)$$

所以(21)、(01)这两个矢量的线性组合张成了线性空间 V 。

又例如挑(21)、(01)矢量组成子集 S' , 显然, 通过它们的线性组合也能生成 V 中的所有矢量。例如:

$$(21) = 1(21) + 0(01) = (21)$$

$$(12) = 2(21) + 0(01) = (12)$$

因此, 在 V 中可以存在有很多组的矢量子集, 它们都能张成整个矢量空间。下面我们讨论这些都能张成同一空间的矢量组有什么特点。

引理 2.6.1 如果 k 个矢量的集合 v_1, v_2, \dots, v_k 张成了含有 m 个线性独立的矢量 u_1, u_2, \dots, u_m 的空间 V , 则 $k \geq m$ 。

证明 因为 v_1, v_2, \dots, v_k 张成 V , 而 u_1, u_2, \dots, u_m 在 V 中, 所以

$$u_1 = a_1 v_1 + a_2 v_2 + \dots + a_k v_k \quad a_i \in F$$

且总有一个 $a_i \neq 0$ 。设 $a_1 \neq 0$, 则

$$v_1 = a_1^{-1} u_1 - a_1^{-1} a_2 v_2 - \dots - a_1^{-1} a_k v_k$$

所以 v_1 可用 u_1, v_2, \dots, v_k 的线性组合表示。因此, V 可以由 $u_1, v_2, v_3, \dots, v_k$ 张成。由于 $u_2 \in V$, 所以 u_2 也可以由 u_1, v_2, \dots, v_k 的线性组合表示, 即

$$u_2 = b_1 u_1 + b_2 v_2 + \dots + b_k v_k \quad b_i \in F$$

式中, 至少有一个 b_i 不为 0, 设 $b_2 \neq 0$, 则 v_2 也可由 $u_1, u_2, v_3, \dots, v_k$ 的线性组合表示, 即

$$v_2 = b_2^{-1} u_2 - b_2 b_2^{-1} u_1 - \dots - b_2^{-1} b_k v_k$$

所以, $u_1, u_2, v_3, \dots, v_k$ 可以张成整个线性空间。由于 u_1, u_2, \dots, u_m 是线性独立的, 因此在

$$u_j = c_1 u_1 + c_2 u_2 + \dots + c_{j-1} u_{j-1} + c_j v_j + \dots + c_k v_k$$

中, 至少存在有一个不为 0 的系数 c_j , 使上述用 u_j 替换 v_j 的过程一直能进行下去, 直到全部 v_i 用完为止, 由于每一步只替换一个 v , 因此 $m \leq k$ 。 ■

定理 2.6.3 如果两组线性独立矢量集合张成同一空间, 则在每一组集合中含有相同的矢量数。

证明 设一个矢量集合中矢量数为 m , 另一个为 k , 则由引理 2.6.1 知 $m \leq k$ 和 $k \geq m$, 所以 $m = k$ 。 ■

该定理说明张成同一空间的集合, 含有相同的独立矢量的数目, 在例 L9 中, (11)、(20)与(21)、(01)这两组矢量集都能张成 GF(3)上的二重矢量组的全体, 且这两组矢量集

中的矢量均是线性独立，独立矢量数均是 2。

定义 2.6.6 在任何线性空间中，能张成该空间的线性独立矢量的集合称为该线性空间的基底。而称这组线性独立矢量的数目为该线性空间的维数。若基底中矢量的个数为有限，则称为有限维数线性空间，否则，称无限维数线性空间。

在例 L9 中 GF(3) 上二重数组构成的线性空间维数为 2，所以称为二维空间。它的基底是(11), (20)或(21), (01)等。下面定理给出了如何寻找矢量空间的基底。

定理 2.6.4 如果 V 是 k 维线性空间，则 V 中任意 k 个线性独立的矢量是 V 的基底。

证明 用反证法。设 v_1, v_2, \dots, v_k 是 V 中 k 个线性独立矢量，如果它们不能张成整个空间 V ，则 V 中必定还有一个矢量 v ，不能由 v_1, v_2, \dots, v_k 的线性组合表示，所以 v, v_1, v_2, \dots, v_k 共 $k+1$ 个矢量是线性独立的，但这和 V 是 k 维线性空间的假设相矛盾。所以， v_1, v_2, \dots, v_k 必张成整个空间 V 。 ■

推论 2.6.1 如果线性空间 $V_1 \subset V_2$ 中，且二者有相同的维数，则 $V_1 = V_2$ 。

证明 V_1 的基底必含在 V_2 中，且是 V_2 中的 k 个线性独立的矢量（设 V_1 和 V_2 的维数为 k ）。所以，这 k 个线性独立的矢量不仅张成 V_1 ，也必张成 V_2 。反之也一样， V_2 中的 k 个线性独立矢量也必张成 V_1 ，故 $V_1 = V_2$ 。 ■

2. n 维数组的内积(点积)和正交由前面知，域 F 上的 n 维数组是 n 重： (a_1, a_2, \dots, a_n) , $a_i \in F$ 。若我们如下定义 n 维数组之间的相加和数乘运算：

$$\begin{aligned} + : \quad a+b &= (a_1, a_2, \dots, a_n) + (b_1, b_2, \dots, b_n) \\ &= (a_1+b_1, a_2+b_2, \dots, a_n+b_n) \quad a_i, b_i \in F \end{aligned}$$

$$\text{数乘} : \quad ca = c(a_1, a_2, \dots, a_n) = (ca_1, ca_2, \dots, ca_n) \quad c \in F, a_i \in F$$

则 n 维数组全体所构成的 n 维线性空间，它的自然基底是：

$$(10\cdots 0), (010\cdots 0), (0010\cdots 0), \dots, (00\cdots 01)$$

定义 2.6.7 两个 n 维数组 a, b 的内积(或点积)表示为

$$\begin{aligned} a \cdot b &= (a_1, a_2, \dots, a_n) \cdot (b_1, b_2, \dots, b_n) \\ &= a_1b_1 + a_2b_2 + \cdots + a_nb_n \quad a_i, b_i \in F \end{aligned}$$

它是一个纯量。

如果两个矢量 a, b 的内积 $a \cdot b = 0$ ，则 a, b 矢量互为正交。显然

$$a \cdot b = b \cdot a \quad a \cdot (b+c) = a \cdot b + a \cdot c$$

3. 线性结合代数

定义 2.6.8 域 F 上的有限维线性空间 A ，若元素之间定义了乘法，且有如下性质：

(1) 乘法封闭。对每一个 $a, b \in A$ ，恒有 $ab \in A$ 。

(2) 乘法结合律成立。对每一个 $a, b, c \in A$ 恒有 $(ab)c = a(bc)$ 。

(3) 分配律成立。

$$\begin{aligned} ① \quad a(\alpha b + \beta c) &= \alpha(ab) + \beta(ac) \quad \alpha, \beta \in F, a, b, c \in A \\ ② \quad (\alpha b + \beta c)a &= \alpha(ba) + \beta(ca) \end{aligned}$$

则称 A 是一个线性结合代数。它的阶数定义为它作为线性空间时的维数。如果 A 关于乘法有逆元(0元除外)，则称 A 为可除代数。

从上面定义看来，线性结合代数非常像一个环。

例 L10 若我们定义 n 维数组 (a_1, a_2, \dots, a_n) 、 (b_1, b_2, \dots, b_n) 的相乘为

$$(a_1, a_2, \dots, a_n)(b_1, b_2, \dots, b_n) = (a_1b_1, a_2b_2, \dots, a_nb_n) \quad a_i, b_i \in F$$

则 $GF(p)$ 上的 n 维数组全体组成一个线性结合代数。如 $GF(3)$ 上的二维数组全体构成了一个线性结合代数。如 p 为素数，则 $GF(p)$ 上的 n 维数组是一个可除代数。如 $GF(3)$ 上的二维数组全体就是一个可除代数。

二、矩阵

从环 R 中选出 $m \times n$ 个元素 a_{ij} ($i=1, 2, \dots, m$; $j=1, 2, \dots, n$)，按一定次序排列成如下 m 行 n 列的方阵：

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

称为环 R 上的一个 $m \times n$ 阶矩阵，用 $A_{m \times n}$ 或 $[a_{ij}]_{m \times n}$ (有时以简写 A 或 $[a_{ij}]$) 表示之。

在今后学习的纠错码论中，矩阵内的第 i 行第 j 列元素 a_{ij} 一般取自域上，今后我们仅讨论域上的矩阵。下面我们首先介绍矩阵的基本运算，然后介绍行空间与列空间的概念。

矩阵中的行数 m 与列数 n 通常情况并不相等。如 $m=n$ ，则称为方阵， n 称为方阵的阶。

1. 矩阵的转置、相等、相加和相乘 $m \times n$ 阶矩阵 A 的转置矩阵 A^T (或 A') 表示为

$$A^T = [a_{ij}]^T = [a_{ji}]$$

它是把 A 矩阵的第一行变成 A^T 矩阵的第一列，第二行变成第二列等等而得到的。若 A 为 $m \times n$ 阶矩阵，则 A^T 为 $n \times m$ 阶矩阵。

两个 $m \times n$ 阶矩阵 $[a_{ij}]$ 和 $[b_{ij}]$ 相等，定义为它们对应位的元素相等，即

$$a_{ij} = b_{ij} \quad i = 1, 2, \dots, m; j = 1, 2, \dots, n$$

两个 $m \times n$ 阶矩阵 $[a_{ij}]$ 和 $[b_{ij}]$ 相加，定义为它们对应位元素相加，即

$$\begin{aligned} & \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mn} \end{bmatrix} \\ &= \begin{bmatrix} a_{11} + b_{11} & a_{12} + b_{12} & \cdots & a_{1n} + b_{1n} \\ a_{21} + b_{21} & a_{22} + b_{22} & \cdots & a_{2n} + b_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} + b_{m1} & a_{m2} + b_{m2} & \cdots & a_{mn} + b_{mn} \end{bmatrix} \end{aligned}$$

$m \times n$ 阶矩阵 $[a_{ij}]$ 和域元素 $c \in F$ 相乘之积定义为

$$c \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} = \begin{bmatrix} ca_{11} & \cdots & ca_{1n} \\ \vdots & & \vdots \\ ca_{m1} & \cdots & ca_{mn} \end{bmatrix}$$

$m \times n$ 阶矩阵 $[a_{ij}]$ 和 $n \times k$ 阶矩阵 $[b_{ij}]$ 相乘定义为

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} b_{11} & \cdots & b_{1k} \\ \vdots & & \vdots \\ b_{n1} & \cdots & b_{nk} \end{bmatrix} = \begin{bmatrix} c_{11} & \cdots & c_{1k} \\ \vdots & & \vdots \\ c_{m1} & \cdots & c_{mk} \end{bmatrix}$$

式中

$$c_{ij} = \sum_{l=1}^n a_{il} b_{lj}$$

是 $[a_{ij}]$ 矩阵的第 i 行与 $[b_{ij}]$ 矩阵的第 j 列的内积，因此要两个矩阵能相乘，第一个矩阵的列数与第二个矩阵的行数必须相等，否则不能相乘。

例 M 1 实数域上的两个矩阵相乘

$$\begin{bmatrix} 1 & 2 & 0 & 1 & 1 \\ 0 & 1 & 0 & 2 & 0 \\ 1 & 0 & 1 & 0 & 2 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 2 & 0 \\ 0 & 2 \\ 2 & 1 \\ 2 & 0 \end{bmatrix} = \begin{bmatrix} 9 & 2 \\ 6 & 2 \\ 5 & 3 \end{bmatrix}$$

应注意的是，在相乘或相加运算中，矩阵中元素之间的运算必须按域的运算规则进行。

若两个矩阵 $[a_{ij}]$ 、 $[b_{ij}]$ 为 $n \times n$ 阶矩阵，则可以交换相乘，但结果一般不相同，即

$$[a_{ij}] [b_{ij}] \neq [b_{ij}] [a_{ij}]$$

例 M 2 在 GF(2) 上的两个方阵相乘，例如

$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

而

$$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}$$

显然

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} \neq \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}$$

也就是说，矩阵相乘不满足交换律。

下面介绍分块矩阵概念。

如果我们把矩阵的每一行（或每一列）看成一个 n （或 m ）维数组，或者行（列）矢量，则可把一个 $m \times n$ 阶矩阵 $[a_{ij}]$ 表示如下：

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ a_{21} & \cdots & a_{2n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} = \begin{bmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \vdots \\ \mathbf{a}_m \end{bmatrix} \quad \mathbf{a}_i = (a_{i1} \ a_{i2} \ \cdots \ a_{in})$$

或

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ a_{21} & \cdots & a_{2n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} = [\mathbf{a}_1 \ \mathbf{a}_2 \ \cdots \ \mathbf{a}_n], \quad \mathbf{a}_j = \begin{bmatrix} \mathbf{a}_{1j} \\ \mathbf{a}_{2j} \\ \vdots \\ \mathbf{a}_{mj} \end{bmatrix}$$

像这种表示方法称为分块矩阵的一种表示。因两个矩阵 $[a_{ij}]$ 和 $[b_{jk}]$ 相乘也可表示成

$$[\mathbf{a}_{ij}] [\mathbf{b}_{jk}] = \begin{bmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \vdots \\ \mathbf{a}_m \end{bmatrix} [\mathbf{b}_1 \ \mathbf{b}_2 \ \cdots \ \mathbf{b}_k]$$

根据矩阵相乘的规则，显然只有 $[a_{ij}]$ 矩阵中行矢的维数等于 $[b_{jk}]$ 中列矢的维数时，相乘才有意义，否则不能相乘。

矩阵不仅可以按列或按行进行分块，也可以按照需要，以几行几列分块。如

$$\begin{array}{|c c c|c c|} \hline a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ \hline a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ \hline \cdots & & & & \\ \hline a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ \hline a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ \hline \end{array} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix}$$

可按阵中虚线所示划分成4块。在分块矩阵表示下，两个矩阵 $[a_{ij}]$ 和 $[b_{jk}]$ 的相乘可写成

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} \\ \mathbf{B}_{21} & \mathbf{B}_{22} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{11}\mathbf{B}_{11} + \mathbf{A}_{12}\mathbf{B}_{21} & \mathbf{A}_{11}\mathbf{B}_{12} + \mathbf{A}_{12}\mathbf{B}_{22} \\ \mathbf{A}_{21}\mathbf{B}_{11} + \mathbf{A}_{22}\mathbf{B}_{21} & \mathbf{A}_{21}\mathbf{B}_{12} + \mathbf{A}_{22}\mathbf{B}_{22} \end{bmatrix}$$

根据矩阵乘法规则，必须保证分块矩阵中每一块矩阵相乘有意义，即只有 \mathbf{A}_{ij} 的列数等于 \mathbf{B}_{jk} 的行数时，这两个分块才能相乘。

由上面定义可知，所有 $m \times n$ 阶矩阵全体构成 F 上的 mn 维的一个线性空间（每一矩阵看成一个元素）。所有 n 阶方阵全体构成了域 F 上的 n^2 阶线性结合代数。

2. 矩阵的秩

定义 2.6.9 $m \times n$ 阶矩阵 A 的行(列)空间是以 A 的行(列)作为矢量所张成的空间，它是 $n(m)$ 维矢量空间中的一个子空间。行(列)空间的维数叫做行(列)秩，它等于行(列)空间中的线性无关的最大行(列)数(即基底中的矢量数)。

上面定义了矩阵的行(列)秩概念，而矩阵的秩定义为：矩阵中不等于零的子式(矩阵的子行列式)的最大阶数，或定义为行(列)秩。今后我们主要用后者的定义计算矩阵的秩。

例 M3 在 $GF(2)$ 上的以下矩阵：

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

由行矢：(10001), (01010), (10101) 张成的行空间共有 8 个矢量：(10001), (01010), (10101), (11011), (00100), (11111), (01110) 及 (00000)。显然，这 3 个行矢是该行空间的一组基底，所以是五维空间的一个三维子空间。

由列矢(101), (010), (001), (010), (101), 张成的列空间也有 8 个矢量: (101), (010), (001), (110), (111), (100), (011), (000)。其中(101), (010), (001)是一组基底。故列秩为 3。由此可知, 矩阵的秩为 3。

3. 初等行运算 矩阵的初等行运算定义为:

(1) 交换矩阵任意两行(或两列)。例如把以下矩阵

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

交换第一行与第二行, 变为

$$\begin{bmatrix} a_{21} & a_{22} & \cdots & a_{2n} \\ a_{11} & a_{12} & \cdots & a_{1n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

这相当于用以下的 $m \times m$ 阶矩阵

$$I = \begin{bmatrix} 0 & 1 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

左乘 A 矩阵。

若称主对角线元素均为 1, 其余元素均为 0 的矩阵

$$I_{m \times m} = \begin{bmatrix} 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

为 $m \times m$ 阶单位方阵, 则交换矩阵 $A_{m \times n}$ 的第 j 和第 i 行, 就是把 $I_{m \times m}$ 矩阵交换第 j 行和第 i 行后左乘 $A_{m \times n}$ 矩阵得到的。而交换矩阵 $A_{m \times n}$ 的第 j 列和第 k 列, 就是把 $I_{n \times n}$ 矩阵交换第 j 列和第 k 列后右乘 $A_{m \times n}$ 得到的。

例 M4 交换以下矩阵的第二与第三行

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

为

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

(2) 以非 0 元素 $c \in F$, 乘以任一行。如矩阵 $A_{m \times n}$

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

用 $c \in F$ 乘以矩阵 $A_{m \times n}$ 的第二行，则为

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ ca_{21} & ca_{22} & \cdots & ca_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

这相当于进行以下运算：

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & c & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ ca_{21} & ca_{22} & \cdots & ca_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

由此可知，如矩阵 $A_{m \times n}$ 的第 i 行要乘以 c ，相当于 $I_{m \times m}$ 的第 i 行的 1 换成 c ，然后左乘 $A_{m \times n}$ 矩阵。

(3) 任意第 j 行的 c 倍加至第 k ($\neq j$) 行。这种运算，相当于把 $I_{m \times m}$ 阶单位方阵的第 j 行的 1 换成 c ，与第 k 行相加后，左乘 $A_{m \times n}$ 矩阵。例如：

$$\begin{aligned} & \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ c & 0 & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \\ &= \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ ca_{11} + a_{31} & ca_{12} + a_{32} & ca_{13} + a_{33} & ca_{14} + a_{34} \end{bmatrix} \end{aligned}$$

定义 2.6.10 完成初等行运算的矩阵称为初等矩阵。

由此定义可知，初等行运算的逆也是同类型的初等行运算，得到的矩阵也是初等矩阵。例如：

$$\begin{aligned} & \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -c & 0 & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ ca_{11} + a_{31} & ca_{12} + a_{32} & ca_{13} + a_{33} & ca_{14} + a_{34} \end{bmatrix} \\ &= \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \end{aligned}$$

就是(3)例中的逆运算，而

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c^{-1} & 0 & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ ca_{21} & ca_{22} & \cdots & ca_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

就是(2)例中的逆运算，而这些例中的左乘 $A_{m \times n}$ 矩阵的方阵，例如：

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -c & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ c & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & c & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & \cdots & & & 1 \end{bmatrix}$$

等均称为初等矩阵。

定理 2.6.5 如果一个矩阵可以由另一矩阵逐次施行初等运算得到，则该二矩阵有同样的行空间，称这两个矩阵为等价矩阵。

证明 在(1)、(2)初等运算中定理显然成立。现检验(3)，设矩阵 A_3 是 A 作第(3)种行运算得到的，它的行空间为 V_3 ， A 的行空间为 V 。由于 A_3 中改变了的行仅是 A 中相应行的线性组合，所以 A_3 中行的线性组合就是 A 中行的线性组合，故 $V_3 \subset V$ 。但是 A 可由 A_3 作逆变换得到，所以 A 的行空间 V 在 A_3 的行空间 V_3 内， $V \subset V_3$ ，由此得到 $V_3 = V$ 。 ■

推论 2.6.2 初等行运算中，矩阵的秩不变。

上面所述的 3 条初等行运算和定理 2.6.5，同样也适用于初等列运算。今后，初等行运算或初等列运算统称为矩阵的初等运算。

4. 梯形标准(典型)阵 为了很快地得到矩阵的秩，可以通过初等运算把矩阵化成梯形标准阵或梯形典型阵。

定义 2.6.11 如下形式的矩阵称为梯形标准阵：

1° 非 0 行自左算起，第一个非 0 元素为 1。

2° 包含这一个首项的，列的其余元素均为 0。

3° 下一行的首项元素，在上一行首项元素的右边，所有 0 行均在非 0 行的下面。

例 M5 如下两个矩阵：

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad \text{和} \quad \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

都是梯形典型阵。

显然，任一矩阵的梯形标准阵是唯一的，梯形标准阵的非 0 行均线性无关，所以梯形标准阵中非 0 行的数目，就是行空间的维数，也是矩阵的秩。

例 M6 求 GF(2) 上矩阵

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

的秩。

把该矩阵进行初等行运算化成梯形典型阵

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \xrightarrow{\text{第1行与第2行交换}} \begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \xrightarrow{\text{第2行加第3行为第3行}} \begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

$$\left[\begin{array}{ccccc} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{array} \right] \xrightarrow{\text{第3行与第4行相加为第4行}} \left[\begin{array}{ccccc} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{array} \right]$$

由此可知，该矩阵只有三行线性独立，故矩阵的秩是 3。

推论 2.6.3 任何矩阵的秩等于其等价梯形标准阵中非 0 行的数目。

5. 奇异与非奇异矩阵

定义 2.6.12 n 阶方阵的每行若为线性独立，则称该方阵为**非奇异(或称满秩)矩阵**；否则，称为**奇异(非满秩)矩阵**。

显然， $n \times n$ 阶单位矩阵 $I_{n \times n}$ 是一个非奇异的梯形标准阵，该阵也称为**恒等矩阵**，用 E 或 I 表示之。显然，任何 n 阶矩阵与单位方阵相乘后，其积仍为原矩阵。所以，单位方阵相当于乘法运算中的恒等元(单位元)。

对任一方阵 A ，若有 $AA^{-1}=E$ ，则 A^{-1} 为 A 的**左逆阵**。若 $AA^{-1}=E$ ，则称 A^{-1} 为 A 的**右逆阵**。

定理 2.6.6 每个非奇异方阵都有一左逆阵和右逆阵，它是初等矩阵的乘积，且左逆阵等于右逆阵。

证明 因非奇异的 $n \times n$ 阶方阵 A ，可用初等行运算化成主对角线元素均为 1 的梯形典型阵(单位方阵) E ，而每次初等行运算可用各初等矩阵 E_i 左乘 A 完成，即

$$E_k E_{k-1} \cdots E_1 A = E$$

所以

$$E_k E_{k-1} \cdots E_1 = A^{-1}$$

同理可证右逆阵。由于非奇异 n 阶方阵全体对乘法构成一个非交换的群，由群中逆元唯一性可知，左逆阵等于右逆阵。 ■

定理 2.6.7 如果 A 是 $n \times m$ 阶矩阵， S 是 $n \times n$ 阶非异方阵，则 SA 与 A 有同样的行空间。

证明 SA 的行是 A 行的线性组合，所以 SA 的行空间 V_{SA} 含在 A 的行空间 V_A 中， $V_{SA} \subset V_A$ 。由于 S 是非异方阵， S^{-1} 存在，所以

$$S^{-1}(SA) = (S^{-1}S)A = EA = A$$

由此可见 A 的行是 SA 行的线性组合，故 $V_{SA} \supseteq V_A$ ，因此 $V_{SA} = V_A$ 。 ■

定理 2.6.8 若 V_1 是 n 维空间 V 的子空间，则和 V_1 中每一个 n 维矢量均正交的所有矢量，构成 V 的另一子空间 V_2 ，称 V_2 为 V_1 的**零化空间(零空间、成零空间)**或解空间。

证明 要证明 V_2 是子空间只要证明

1° 封闭性成立。对每一个 $v \in V_1$, $u_1, u_2 \in V_2$ ，由假设可知恒有： $v \cdot u_1 = v \cdot u_2 = 0$ (矢量的正交就是内积为 0)。所以

$$v \cdot (u_1 + u_2) = v \cdot u_1 + v \cdot u_2 = 0$$

故 $(u_1 + u_2) \in V_2$ 中。

2° 数乘封闭。

$$v \cdot (cu_1) = c(vu_1) = 0 \quad c \in F$$

所以 $cu_1 \in V_2$ 中。由 1°、2° 可知 V_2 是子空间。 ■

例 M 7 GF(2)上五维线性空间 V_5 是 32 个元素: (10000), (01000), ..., (11111) 组成。

在 V_5 中挑出一个三维子空间 $V_{5,3}$, 它的基底是: (10000), (00100), (00010)。由此 $V_{5,3}$ 由 8 个元素组成, $V_{5,3}$: (10000), (00100), (00010), (10100), (10010), (00110), (10110), (00000)。

在 V_5 中另外再挑出 4 个矢量, 组成另一个二维子空间 $V_{5,2}$, 它的基底为 (01000), (00001), 则 $V_{5,2}$: (01000), (00001), (01001), (00000)。

可以检验 $V_{5,3}$ 中的每一矢量均与 $V_{5,2}$ 中的每一矢量正交, 所以 $V_{5,2}$ 是 $V_{5,3}$ 的零空间; 反之, $V_{5,3}$ 也是 $V_{5,2}$ 的零空间。(这点将在后面证明。)

定理 2.6.9 若矢量 u 和张成 V_1 的矢量集 v_1, v_2, \dots, v_k 中的每一矢量正交, 则 u 在 V_1 的零空间中。

证明 由假设可知, V_1 中的任一矢量 v , 都由 v_1, v_2, \dots, v_k 矢量集张成, 所以

$$v = \alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_k v_k \quad \alpha_i \in F$$

而

$$u \cdot v = \alpha_1 u \cdot v_1 + \alpha_2 u \cdot v_2 + \dots + \alpha_k u \cdot v_k = \alpha_1 0 + \alpha_2 0 + \dots + \alpha_k 0 = 0$$

所以, u 在 V_1 的零空间中。 ■

定理 2.6.10 若 n 维空间 V 的子空间 V_1 的维数为 k , 则 V_1 的零空间的维数为 $n-k$ 。

该定理的证明需要用到有关线性方程组解的知识, 这里不再证明。有兴趣的读者可参阅有关线性代数方面的书籍。

定理 2.6.11 设 V_1, V_2 均是 n 维线性空间 V 的子空间, 且 V_1 是 V_2 的零空间, 则 V_2 也是 V_1 的零空间。

证明 设 V_2 是 n 维空间中的一个 k 维子空间, 则由上面定理可知, 它的零空间 V_1 有维数为 $n-k$ 。反之, V_1 零空间的维数必为 k , 因 V_2 是含在 V_1 的零空间中且有维数为 k , 故 V_2 是 V_1 的零空间。 ■

所以, 若 M_1 和 M_2 是两个 n 列矩阵, 且 $M_1 M_2^T$ 等于全为 0 矩阵, 则 M_2 的行空间必含在 M_1 的零空间中; 反之亦然。若 M_1 的行秩与 M_2 的行秩之和为 n , 则 M_1 的行空间必是 M_2 的零空间; 反之亦然。

6. 范德蒙(Vandermonde)矩阵与范德蒙行列式

定义 2.6.13 下列矩阵

$$M_v = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \vdots & \vdots & & \vdots \\ x_1^{n-1} & x_2^{n-1} & \cdots & x_n^{n-1} \end{bmatrix}$$

称为范德蒙矩阵, 而其相应的行列式称为范德蒙行列式。

定理 2.6.12 只要 M_v 矩阵中所有 x_1, x_2, \dots, x_n 元素不为 0, 且互不相同, 则范德蒙矩阵是一满秩矩阵。

可用归纳法证明, 有兴趣的读者可参阅有关代数书籍。

范德蒙矩阵在编码理论中起着很重要的作用, 这将在以后看到。

到此为止我们简单地讨论了学习编码所需的一些基本数学知识：群、格、环、域、线性空间和矩阵的概念。其它所需的知识，将在以后各章介绍。

习 题

1. 求下列各数之间的最大公约数：

(1) (51 425, 13 310)，并用欧几里德算法表示成 $(a, b) = aA + bB$ 的形式。

(2) (353 430, 530 145, 165 186)。

2. 求下列各数之间的最小公倍数：

[391, 493]，[1 965, 1 834, 30 261]

3. 证明最大公约数的 4 条性质。

4. 求下列同余式的解 x ：

$$15^{15} \equiv x \pmod{17} \quad 3x \equiv 10 \pmod{29}$$

5. 令 G 是由一切数对 (a, b) 所构成的集合， a, b 为有理数，且 $a \neq 0$ 。问 G 对于乘法

$$(a_1, b_1)(a_2, b_2) = (a_1 a_2, a_2 b_1 + b_2)$$

是否构成群，为什么？

6. 全体非负整数集合，在通常的加和乘运算下是否构成群？

7. 构造 $GF(5)$ 的乘法和加法表。

8. 求 $GF(17)$ 上 15 元素的逆元。

9. 证明仅有一个三元素的群，仅有两个四元素的群。这些群是否为可换群？

10. $\{0, 1, 2, 3\}$ 集合在模 4 运算下是否构成乘群或加群？若是，找出其中所有真子群。

11. $(1, -1, i, -i)$ 对乘和加是否构成群？里面有真子群吗？

12. 二进制四重全体的集合能否构成群，是可换群吗？若是，找出其中一个四阶子群，并构造陪集表。

13. 证明定理 2.3.1。

14. 求出环 Z_{16} 和 Z_{24} 的全部零因子。

15. 若假定实数集合是一个域，下列实数子集中哪一个是域，为什么？

(a) 全体形如 $a+b\sqrt{3}$ 的数，此处 a, b 是有理数。

(b) 全体形如 $a+b\sqrt[3]{5}$ 的数，此处 a, b 是有理数。

(c) 全体不是整数的有理数。

16. 试找出能张成 $GF(3)$ 上的三维空间的两组基底。

17. 设 S 是所有三维矢量 (x, y, z) 所组成的空间 V_3 中的一个集合，且分量之间的运算满足结合律，确定 S 是否是子空间，若是，则决定其维数。

18. 把下述 $GF(3)$ 上的矩阵：

$$\begin{bmatrix} 1 & 2 & 0 \\ 2 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

表示成初等矩阵之积，并化成梯形标准阵，计算它的秩。

19. 试用初等运算把下述 $GF(2)$ 上的矩阵：

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

化成梯形标准阵，并计算它的秩。

20. 设 GF(2)上以下两个矩阵：

$$G = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \quad H = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

证明 G 的行空间是 H 的零空间，反之亦然。

21. 令 S_1 和 S_2 是 V 的子空间，证明与 S_1 和 S_2 正交的空间也是 V 的子空间。

22. 证明定理2.6.12。

参 考 文 献

- [1] G. 伯克霍夫, S. 麦克莱恩:《近世代数概论(上册)》(王连祥、许广善译), 人民教育出版社, 1979.
- [2] W. W. Peterson, Error-Correcting Codes, MIT Press, 1961.
- [3] 陈景润:《初等数论》,科学出版社,1978.
- [4] 肖国镇:《代数编码引论》,西北电讯工程学院,1975.
- [5] G. D. Forney, "Coset codes—Part I ,Part II ", IEEE Trans. on IT, No. 5, pp. 1123—1187, 1988.
- [6] 张禾瑞、郝炳新:《高等代数》,高等教育出版社, 1958.

第三章 线性分组码

线性分组码是分组码中最重要的一类码，它是讨论各类码的基础。虽然这类码的概念比较简单，但却非常重要，特别是有关码的生成矩阵 G 和校验矩阵 H 的表示，以及它们之间的关系，而 H 与纠错能力之间的关系则更为重要。本章分以下几部分：线性分组码的基本概念、码的 G 矩阵和 H 矩阵、伴随式计算与标准阵、汉明码和不等纠错能力保护(UEP)码，最后简单讨论码纠错能力的上下限及非对称信道与单向信道纠错码。

§ 3.1 线性分组码的基本概念

第一章第三节已叙述了分组码的某些重要概念，如分组码的表示、码率、距离、重量等。如果我们把每一码字看成是一个 n 维数组或 n 维线性空间中的一个矢量，则可以从线性空间的角度，比较深入地讨论线性分组码。

一个 $[n, k]$ 线性分组码，是把信息划成 k 个码元为一段（称为信息组），通过编码器变成长为 n 个码元的一组，作为 $[n, k]$ 线性分组码的一个码字。若每位码元的取值有 q 种（ q 为素数幂），则共有 q^k 个码字。 n 长的数组共有 q^n 组，在二进制情况下，有 2^n 个数组。显然， q^n 个 n 维数组（ n 重）组成一个 $GF(q)$ 上的 n 维线性空间。如果 q^k （或 2^k ）个码字集合构成了一个 k 维线性子空间，则称它是一个 $[n, k]$ 线性分组码。

定义 3.1.1 $[n, k]$ 线性分组码是 $GF(q)$ 上的 n 维线性空间 V_n 中的一个 k 维子空间 $V_{n,k}$ 。

由于该线性子空间在加法运算下构成阿贝尔群，所以线性分组码又称为群码。

为简单起见，今后若没有特别说明，所说的分组码均指线性分组而言，且用 $(c_{n-1}, c_{n-2}, \dots, c_1, c_0)$ 表示 $[n, k]$ 码的一码字，其中每一分量 $c_i \in GF(q)$ 。

例 3.1 $n=7, k=3$ 的 $[7, 3]$ 线性分组码的 8 个码字和信息组如表 3-1 所示。这 8 个码字在模 2 加法运算下构成一个阿贝尔加群。 ■

由于线性分组码是分组码的一类，因此第一章中有关分组码的参数，如码率 $R=k/n$ 、码字的距离与码的最小距离、码字的重量等定义，以及说明最小距离与纠错能力之间关系的定理 1.3.1，对线性分组码均适用，这里不再赘述。

显然， R 和 d 是分组码的两个最重要的参数，因此今后我们用 $[n, k, d]$ （或 $[n, k]$ ）表示线性分组码。而用 (n, M, d) 表示码字数目为 M 的任何

表 3-1 $[7, 3]$ 码的码字表

| 信息组 | 码字 |
|-----|---------|
| 000 | 0000000 |
| 001 | 0011101 |
| 010 | 0100111 |
| 011 | 0111010 |
| 100 | 1001110 |
| 101 | 1010011 |
| 110 | 1101001 |
| 111 | 1110100 |

码，此时码率 $R = n^{-1} \log_2 M$ 。

$[n, k, d]$ 分组码是一个群码，因此若码字 $C_1 \in [n, k, d]$ 、 $C_2 \in [n, k, d]$ ，则由群的封闭性可知，码字 C_1 与 C_2 之和 $C_1 + C_2 \in [n, k, d]$ ，即 $C_1 + C_2$ 也必是 $[n, k, d]$ 分组码的一个码字。所以，两码字 C_1 和 C_2 之间的距离 $d(C_1, C_2)$ 必等于第三个码字 $C_1 + C_2$ 的汉明重量。

如例 3.1 中的两个码字：(1010011)，(1101001)，它们之间的距离是 4，它就是(0111010)码字的重量，即

$$d(C_1, C_2) = w(C_1 + C_2)$$

因此，一个 $[n, k, d]$ 分组码的最小距离必等于码中非零码字的最小重量，由此可得如下定理。

定理 3.1.1 $[n, k, d]$ 线性分组码的最小距离等于非零码字的最小重量。

$$d = \min_{C_i \in [n, k]} w(C_i)$$

下面定理给出了线性分组码码字重量之间的重要关系和主要性质。

定理 3.1.2 GF(2)上 $[n, k, d]$ 线性分组码中，任何两个码字 C_1, C_2 之间有如下关系：

$$w(C_1 + C_2) = w(C_1) + w(C_2) - 2w(C_1 \cdot C_2) \quad (3.1.1)$$

或

$$d(C_1, C_2) \leq w(C_1) + w(C_2) \quad (3.1.2)$$

式中， $C_1 \cdot C_2$ 是两个码字的内积。

证明 设 $C_1 = (c_{1,0}, c_{1,1}, \dots, c_{1,n-1})$

$$C_2 = (c_{2,0}, c_{2,1}, \dots, c_{2,n-1})$$

且令

$$\delta c_{1,i} = \begin{cases} 0 & c_{1,i} = 0 \\ 1 & c_{1,i} = 1 \end{cases}$$

$$\delta c_{2,i} = \begin{cases} 0 & c_{2,i} = 0 \\ 1 & c_{2,i} = 1 \end{cases}$$

对所有 $i=0, 1, \dots, n-1$ ，则

$$w(C_1) = \sum_{i=0}^{n-1} \delta c_{1,i}$$

$$w(C_2) = \sum_{i=0}^{n-1} \delta c_{2,i}$$

$$w(C_1 + C_2) = \sum_{i=0}^{n-1} \delta(c_{1,i} + c_{2,i})$$

注意到 $C_1 + C_2$ 是对应位分量的模 2 和， $\delta c_{1,i} + \delta c_{2,i}$ 是算术和，因此

$$\delta(c_{1,i} + c_{2,i}) = \begin{cases} 0 & c_{1,i} = 0, c_{2,i} = 0 \\ 1 & c_{1,i} = 0, c_{2,i} = 1 \\ 1 & c_{1,i} = 1, c_{2,i} = 0 \\ 0 & c_{1,i} = 1, c_{2,i} = 1 \end{cases}$$

$$\delta c_{1,i} + \delta c_{2,i} = \begin{cases} 0 & c_{1,i} = 0, c_{2,i} = 0 \\ 1 & c_{1,i} = 0, c_{2,i} = 1 \\ 1 & c_{1,i} = 1, c_{2,i} = 0 \\ 2 & c_{1,i} = 1, c_{2,i} = 1 \end{cases}$$

比较两式显然有

$$\delta(c_{1,i} + c_{2,i}) = \delta c_{1,i} + \delta c_{2,i} - 2\delta c_{1,i}c_{2,i}$$

式中

$$\delta c_{1,i}c_{2,i} = \begin{cases} 1 & c_{1,i} = 1, c_{2,i} = 1 \\ 0 & \text{其它} \end{cases}$$

于是

$$\sum_{i=0}^{n-1} \delta(c_{1,i} + c_{2,i}) = \sum_{i=0}^{n-1} \delta c_{1,i} + \sum_{i=0}^{n-1} \delta c_{2,i} - 2 \sum_{i=1}^{n-1} \delta c_{1,i}c_{2,i}$$

$$w(\mathbf{C}_1 + \mathbf{C}_2) = w(\mathbf{C}_1) + w(\mathbf{C}_2) - 2w(\mathbf{C}_1 \cdot \mathbf{C}_2)$$

由前知 $d(\mathbf{C}_1, \mathbf{C}_2) = w(\mathbf{C}_1 + \mathbf{C}_2)$, 因此由上式立刻可得

$$d(\mathbf{C}_1, \mathbf{C}_2) \leq w(\mathbf{C}_1) + w(\mathbf{C}_2)$$

推论 3.1.1 GF(2)上线性分组码任3个码字 $\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3$ 之间的汉明距离, 满足以下三角不等式

$$d(\mathbf{C}_1, \mathbf{C}_2) + d(\mathbf{C}_2, \mathbf{C}_3) \geq d(\mathbf{C}_1, \mathbf{C}_3) \quad (3.1.3)$$

证明 设码字 $\mathbf{C}_a = \mathbf{C}_1 + \mathbf{C}_2, \mathbf{C}_b = \mathbf{C}_2 + \mathbf{C}_3$, 由式(3.1.2)可知: $w(\mathbf{C}_a + \mathbf{C}_b) = w(\mathbf{C}_1 + \mathbf{C}_2 + \mathbf{C}_2 + \mathbf{C}_3) = w(\mathbf{C}_1 + \mathbf{C}_3) = d(\mathbf{C}_1, \mathbf{C}_3) \leq w(\mathbf{C}_a) + w(\mathbf{C}_b) = w(\mathbf{C}_1 + \mathbf{C}_2) + w(\mathbf{C}_2 + \mathbf{C}_3)$ 。所以

$$d(\mathbf{C}_1, \mathbf{C}_3) \leq d(\mathbf{C}_1, \mathbf{C}_2) + d(\mathbf{C}_2, \mathbf{C}_3)$$

定理 3.1.3 任何 $[n, k, d]$ 线性分组码, 码字的重量或全部为偶数, 或者奇数重量的码字数等于偶数重量的码字数。

请读者自行证明该定理。

§ 3.2 码的一致校验矩阵与生成矩阵

一、码的校验矩阵与生成矩阵

$[n, k, d]$ 分组码的编码问题就是在 n 维线性空间 V_n 中, 如何找出满足一定要求的, 有 2^k 个矢量组成的 k 维线性子空间 V_{n-k} 。或者说, 在满足给定条件(码的最小距离 d 或码率 R)下, 如何从已知的 k 个信息元求得 $r=n-k$ 个校验元。这相当于建立一组线性方程组, 已知 k 个系数, 要求 $n-k$ 个未知数, 使得到的码恰好有所要求的最小距离 d 。

上例中的 $[7, 3, 4]$ 码, 若 c_6, c_5, c_4 代表 3 个信息元, 则 c_3, c_2, c_1, c_0 这 4 个校验元, 可由以下线性方程组求得:

$$\begin{cases} 1 \cdot c_3 = 1 \cdot c_6 + 0 \cdot c_5 + 1 \cdot c_4 \\ 1 \cdot c_2 = 1 \cdot c_6 + 1 \cdot c_5 + 1 \cdot c_4 \\ 1 \cdot c_1 = 1 \cdot c_6 + 1 \cdot c_5 + 0 \cdot c_4 \\ 1 \cdot c_0 = 0 \cdot c_6 + 1 \cdot c_5 + 1 \cdot c_4 \end{cases} \quad (3.2.1)$$

或

$$\begin{cases} c_6 + c_4 + c_3 = 0 \\ c_6 + c_5 + c_4 + c_2 = 0 \\ c_6 + c_5 + c_1 = 0 \\ c_5 + c_4 + c_0 = 0 \end{cases} \quad (3.2.2)$$

上述运算均为模 2 运算，因此当 3 个信息元确定后，就可由上述线性方程组求得 4 个校验元。

例 3.2 $c_6=1, c_5=0, c_4=1$, 求 c_3, c_2, c_1, c_0 。

由上述线性方程组可知：

$$\begin{aligned}c_3 &= c_6 + c_4 = 1 + 1 = 0 \\c_2 &= c_6 + c_5 + c_4 = 1 + 0 + 1 = 0 \\c_1 &= c_6 + c_5 = 1 + 0 = 1 \\c_0 &= c_5 + c_4 = 0 + 1 = 1\end{aligned}$$

由此得到的码字为：(1010011)。 ■

可以检验例 3.1 中的 [7, 3, 4] 码的 8 个码字均满足式(3.2.1)和式(3.2.2)。若用矩阵形式表示这些线性方程组，则式(3.2.1)或式(3.2.2)可表示为：

$$\left[\begin{array}{cccccc|cccc} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{array} \right] \begin{bmatrix} c_6 \\ c_5 \\ c_4 \\ c_3 \\ c_2 \\ c_1 \\ c_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \mathbf{0}^T$$

或

$$\begin{bmatrix} c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_0 \end{bmatrix} \left[\begin{array}{cccc} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array} \right] = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix} = \mathbf{0}$$

称上式中的 4 行 7 列矩阵为 [7, 3, 4] 码的一致校验矩阵，通常用 \mathbf{H} 表示，该码的

$$\mathbf{H} = \left[\begin{array}{ccccccc} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{array} \right]$$

一般情况下，任何一个 $[n, k, d]$ 码的 \mathbf{H} 矩阵可表示为

$$\mathbf{H} = \left[\begin{array}{cccccc|c} h_{1, n-1} & h_{1, n-2} & \cdots & h_{1, 0} \\ h_{2, n-1} & h_{2, n-2} & \cdots & h_{2, 0} \\ \vdots & \vdots & & \vdots \\ h_{n-k, n-1} & h_{n-k, n-2} & \cdots & h_{n-k, 0} \end{array} \right] \quad (3.2.3)$$

它是一个 $(n-k) \times n$ 阶矩阵。由此 \mathbf{H} 矩阵可以很快地建立码的线性方程组：

$$\begin{bmatrix} h_{1, n-1} & h_{1, n-2} & \cdots & h_{1, 0} \\ h_{2, n-1} & h_{2, n-2} & \cdots & h_{2, 0} \\ \vdots & \vdots & & \vdots \\ h_{n-k, n-1} & h_{n-k, n-2} & \cdots & h_{n-k, 0} \end{bmatrix} \begin{bmatrix} c_{n-1} \\ c_{n-2} \\ \vdots \\ c_0 \end{bmatrix} = \mathbf{0}^T \quad (3.2.4)$$

或

$$\begin{bmatrix} c_{n-1} & c_{n-2} & \cdots & c_0 \end{bmatrix} \begin{bmatrix} h_{1, n-1} & h_{2, n-1} & \cdots & h_{n-k, n-1} \\ h_{1, n-2} & h_{2, n-2} & \cdots & h_{n-k, n-2} \\ \vdots & \vdots & & \vdots \\ h_{1, 0} & h_{2, 0} & \cdots & h_{n-k, 0} \end{bmatrix} = \mathbf{0} \quad (3.2.5)$$

简写为

$$\mathbf{H} \cdot \mathbf{C}^T = \mathbf{0}^T \quad (3.2.6)$$

或

$$\mathbf{C} \cdot \mathbf{H}^T = \mathbf{0} \quad (3.2.7)$$

可知 \mathbf{H} 矩阵的每一行代表一个线性方程组的系数，它表示求一个校验元的线性方程。因此任何一个 $[n, k, d]$ 码的 \mathbf{H} 矩阵必须有 $n-k$ 行，且每行必须线性独立。若把 \mathbf{H} 的每一行看成一个矢量，则这 $n-k$ 个矢量必然张成了 n 维线性空间中的一个 $n-k$ 维子空间 $V_{n, n-k}$ 。

由于 $[n, k, d]$ 码的每一码字必须满足式(3.2.6)或式(3.2.7)，即它的每一码字必然在由 \mathbf{H} 矩阵的行所张成的 $V_{n, n-k}$ 空间中的零空间中。由定理 2.6.10 可知， $V_{n, n-k}$ 的零空间必然是一个 k 维子空间 $V_{n, k}$ ，而这正是 $[n, k, d]$ 码的码字集合全体。所以， $V_{n, n-k}$ 与 $[n, k, d]$ 码的每一码字均正交，也就是 \mathbf{H} 矩阵的每一行与它的码的每一码字的内积均为 0。

$[n, k, d]$ 分组码的 2^k 个码字组成了一个 k 维子空间，因此这 2^k 个码字完全可由 k 个独立矢量所组成的基底而张成。设基底为

$$\left\{ \begin{array}{l} \mathbf{C}_1 = (g_{1, n-1}, g_{1, n-2}, \dots, g_{1, 0}) \\ \mathbf{C}_2 = (g_{2, n-1}, g_{2, n-2}, \dots, g_{2, 0}) \\ \vdots \\ \mathbf{C}_k = (g_{k, n-1}, g_{k, n-2}, \dots, g_{k, 0}) \end{array} \right.$$

若把这组基底写成矩阵形式，则有

$$\mathbf{G} = \begin{bmatrix} g_{1, n-1} & g_{1, n-2} & \cdots & g_{1, 0} \\ g_{2, n-1} & g_{2, n-2} & \cdots & g_{2, 0} \\ \vdots & \vdots & & \vdots \\ g_{k, n-1} & g_{k, n-2} & \cdots & g_{k, 0} \end{bmatrix} \quad (3.2.8)$$

$[n, k, d]$ 码中的任何码字，都可由这组基底的线性组合生成，即

$$\mathbf{C} = \mathbf{m} \cdot \mathbf{G} = [m_{n-1} \ m_{n-2} \ \cdots \ m_{n-k}] \begin{bmatrix} g_{1, n-1} & g_{1, n-2} & \cdots & g_{1, 0} \\ g_{2, n-1} & g_{2, n-2} & \cdots & g_{2, 0} \\ \vdots & \vdots & & \vdots \\ g_{k, n-1} & g_{k, n-2} & \cdots & g_{k, 0} \end{bmatrix} \quad (3.2.9)$$

式中， $\mathbf{m} = (m_{n-1}, m_{n-2}, \dots, m_{n-k})$ 是 k 个信息元组成的信息组。因此，若已知信息组 \mathbf{m} ，通过式(3.2.9)可求得相应的码字，称式(3.2.8)的 \mathbf{G} 为 $[n, k, d]$ 码的生成矩阵。

如表 3-1 中的 [7, 3, 4] 码，可从它的 8 个码字中任意挑出 $k=3$ 个线性无关的码字：(1001110), (0100111), (0011101) 作为码的生成矩阵的行，则

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix} \quad (3.2.10)$$

若已知信息组 $m=(011)$ ，则相应的码字为

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix} = (0 \ 1 \ 1 \ 1 \ 0 \ 1 \ 0)$$

显然，一个矢量空间的基底可以不止一个，因此作为码的生成矩阵 G 也可以不止一种形式。但不论哪一种形式，它们都生成相同的矢量空间，即生成同一个 $[n, k, d]$ 码。

G 中的每一行及其线性组合均为 $[n, k, d]$ 码的一个码字，所以由式 (3.2.6) 和式 (3.2.7) 可知

$$G \cdot H^T = \mathbf{0} \quad (3.2.11)$$

或

$$H \cdot G^T = \mathbf{0}^T \quad (3.2.12)$$

说明由 G 与 H 的行生成的空间互为零空间。

二、对偶码

$[n, k, d]$ 码是 n 维线性空间中的一个 k 维子空间 $V_{n,k}$ ，由一组基底即 G 的行张成。由前可知，它的零空间必是一个 $n-k$ 维的线性子空间 $V_{n,n-k}$ ，并由 $n-k$ 个独立矢量张成。由式 (3.2.11) 和式 (3.2.12) 可知，这 $n-k$ 个矢量就是 H 矩阵的行。因此，若把 H 矩阵看成是 $[n, n-k, d]$ 码的生成矩阵 G ，而把 $[n, k, d]$ 码的 G 看成是它的校验矩阵 H ，则我们称由 G 生成的 $[n, k, d]$ 码 C 与由 H 生成的 $[n, n-k, d]$ 码 C^\perp 互为对偶码。相应地，称 $V_{n,k}$ 与 $V_{n,n-k}$ 空间互为对偶空间。由此可如下定义对偶码。

定义 3.2.1 设 C 是 $[n, k, d]$ 码，则它的对偶码 C^\perp 是

$$C^\perp \triangleq \{x \in V_{n,(n-k)}; \text{ 对所有 } y \in C \text{ 使 } x \cdot y = 0\}$$

式中， $x \cdot y$ 为 x 与 y 的内积。

如例 3.1 中的 $[7, 3, 4]$ 码，它的对偶码必是 $[7, 4, 3]$ 码，其生成矩阵 $G_{[7,4]}$ 就是 $[7, 3, 4]$ 码的校验矩阵 $H_{[7,3]}$ 。

$$G_{[7,4]} = H_{[7,3]} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

由此生成矩阵，已知信息组 m ，根据式 (3.2.9) 就可求得 $[7, 4, 3]$ 码的 16 个码字。

若一个码的对偶码就是它自己，即 $C=C^\perp$ 则称 C 码为自对偶码。显然，自对偶码必定是 $[2m, m, d]$ 形式的分组码。如 $[2, 1, 2]$ 重复码就是一个自对偶码。如果自对偶码的最小距离 d 是 4 的倍数，则称为双偶自对偶码，可以证明双偶自对偶码的码长 n 必是 8 的整数倍。

三、系统码

定义 3.2.2 若信息组以不变的形式在码组的任意 k 位(通常在最前面: $c_{n-1}, c_{n-2}, \dots, c_{n-k}$)中出现的码称为**系统码**, 否则为**非系统码**。

系统码的一种结构形式如图 3-1 所示。表 3-1 中所列的 [7, 3, 4] 码就是系统码形式。显然, 系统码的信息位与校验位很容易区分开, 所以这种码也称**可分码**。

由于系统码的码字前 k 位是原来的信息组, 故由式(3.2.8)可知, G 矩阵左边 k 列必组成一个单位方阵 I_k , 因此系统码的生成矩阵通常为

$$G = [I_k P] \quad (3.2.13)$$

| | |
|----------|------------|
| k 位信息位 | $n-k$ 位校验位 |
|----------|------------|

图 3-1 系统码的一种结构形式

式中, P 是 $k \times (n-k)$ 阶矩阵。如果信息组不在码字的前 k 位, 而在码字的后 k 位, 则 G 矩阵中的 I_k 方阵在 P 矩阵的右边。因为 G 与 H 矩阵所组成的空间互为零空间, 所以与式(3.2.13)相应的 H 矩阵为

$$H = [-P^T I_{n-k}] \quad (3.2.14)$$

式中, $-P^T$ 是一个 $(n-k) \times k$ 阶矩阵, 它是 P 矩阵的转置, “-”号表示 $-P^T$ 阵中的每一元素是 P 阵中对应元素的逆元, 在二进制情况下, 仍是该元素自己。显然, 由此得到的 H 满足

$$G \cdot H^T = [I_k P] \begin{bmatrix} -P \\ I_{n-k} \end{bmatrix} = 0$$

通常, 我们称式(3.2.13)与式(3.2.14)中的 G 和 H 矩阵为码的**典型(标准)生成矩阵**和**典型校验矩阵**。如 [7, 3, 4] 码的典型生成矩阵

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix} = [I_3 P]$$

相应地, 典型校验矩阵由式(3.2.14)为

$$H = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} = [-P^T I_4]$$

系统码的编码相对而言较为简单, 且由 G 可以方便地得到 H (反之亦然), 容易检查编出的码字是否正确。同时, 对分组码而言, 系统码与非系统码的纠错能力完全等价。因此, 今后若无特别声明, 仅讨论系统码形式。

四、缩短码

在某些情况下, 如果不能找到一种比较合适的码长或信息位个数, 则可把某一 $[n, k, d]$ 码进行缩短, 以满足要求。

在 $[n, k, d]$ 码的码字集合中，挑选前*i*个信息位数字均为0的所有码字，组成一个新的子集。由于该子集的前*i*位信息位均取0，故传输时可以不送它们，仅只要传送后面的*n-i*位码元即可。这样该子集组成了一个 $[n-i, k-i, d]$ 分组码，称它为 $[n, k, d]$ 码的**缩短码**。由于缩短码是*k*维子空间 V_{n-k} 中取前*i*位均为0的码字组成的一个子集，显然该子集是 V_{n-k} 空间中的一个*k-i*维的子空间 V_{n-k-i} ，因此 $[n-i, k-i, d]$ 缩短码的纠错能力至少与原 $[n, k, d]$ 码相同。

如例3.1中的 $[7, 3, 4]$ 码，若挑出第一个信息位均为0的码字：(0000000)，(0011101)，(0100111)，(0111010)，则可组成一个 $[6, 2, 4]$ 缩短码：(000000)，(011101)，(100111)，(111010)。

缩短码的**G**矩阵只要在原 $[n, k, d]$ 码的**G**矩阵中，去掉左边*i*列和上边*i*行即可。如上例中， $[6, 2, 4]$ 缩短码的**G**矩阵可由 $[7, 3, 4]$ 码的**G**矩阵(式(3.2.10))去掉第一行及左边第一列得到

$$G_{[6, 2]} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

而**H**矩阵，只要在原码的**H**矩阵中去掉第一列即可。如该例的**H**矩阵为

$$H_{[6, 2]} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$[n-i, k-i]$ 缩短码是 $[n, k]$ 码缩短*i*位得到的，因而码率*R*比原码要小，但纠错能力不一定比原码强。因此总的看来，缩短码比原码的性能要差。

§ 3.3 伴随式与标准阵列及其它译码

一、伴随式(校正子)

本节讨论如何译码。设发送的码字 $C=(c_{n-1}, c_{n-2}, \dots, c_1, c_0)$ ，通过有扰信道传输，信道产生的错误图样 $E=(e_{n-1}, e_{n-2}, \dots, e_1, e_0)$ 。接收端译码器收到的*n*重为 $R=(r_{n-1}, r_{n-2}, \dots, r_1, r_0)$ ， $R=C+E$ ， $r_i=c_i+e_i$ ， $c_i, r_i, e_i \in GF(q)$ 或 $GF(2)$ 中。译码器的任务就是从收到的*R*中得出 \hat{C} ，或者由*R*中解出错误图样 \hat{E} ，从而得到 $\hat{C}=R-\hat{E}$ ，并使译码错误概率最小，或使 \hat{C} 尽可能接近于 C 。

$[n, k, d]$ 码的每一码字 C ，都必须满足式(3.2.6)或式(3.2.7)。因此，收到*R*后用该两式之中的任一式进行检验：

$$R \cdot H^T = (C + E) \cdot H^T = C \cdot H^T + E \cdot H^T = E \cdot H^T \quad (3.3.1)$$

若 $E=0$ ，则 $R \cdot H^T=0$ ， $E \neq 0$ 则 $R \cdot H^T \neq 0$ 。说明 $R \cdot H^T$ 仅与错误图样有关，而与发送的是什么码字无关。令

$$S = R \cdot H^T = E \cdot H^T \quad \text{或} \quad S^T = H \cdot R^T = H \cdot E^T \quad (3.3.2)$$

称为接收矢量*R*的**伴随式(或校正子)**。因此伴随式完全由*E*决定，它充分地反映了信道的干扰情况。译码器的主要任务就是如何从*S*中得到最像*E*的错误图样 \hat{E} ，从而译出

$$\hat{C} = R - \hat{E}.$$

由前知, $[n, k, d]$ 码的校验矩阵

$$H = \begin{bmatrix} h_{1, n-1} & h_{1, n-2} & \cdots & h_{1, 1} & h_{1, 0} \\ h_{2, n-1} & h_{2, n-2} & \cdots & h_{2, 1} & h_{2, 0} \\ \vdots & \vdots & & \vdots & \vdots \\ h_{n-k, n-1} & h_{n-k, n-2} & \cdots & h_{n-k, 1} & h_{n-k, 0} \end{bmatrix} = [h_{n-1} \ h_{n-2} \ \cdots \ h_1 \ h_0]$$

式中, h_{n-i} 为 H 矩阵的第 i 列, 它是一个 $n-k$ 重列矢量。设

$$\begin{aligned} E &= (e_{n-1}, e_{n-2}, \dots, e_1, e_0) \\ &= (0, \dots, e_{i_1}, 0, \dots, e_{i_2}, 0, \dots, e_{i_3}, 0, \dots, e_{i_t}, 0, \dots, 0) \end{aligned}$$

第 i_1, i_2, \dots, i_t 位有错, 则

$$\begin{aligned} S &= E \cdot H^T = [0 \ \cdots \ e_{i_1} \cdots \ e_{i_2} \cdots \ e_{i_t} \ 0 \ \cdots \ 0] \begin{bmatrix} h_{n-1} \\ h_{n-2} \\ \vdots \\ h_1 \\ h_0 \end{bmatrix} \\ &= e_{i_1} h_{i_1}^T + e_{i_2} h_{i_2}^T + \cdots + e_{i_t} h_{i_t}^T \end{aligned} \quad (3.3.3)$$

说明 S 是 H 矩阵中相应于 $e_{i_j} \neq 0$ ($j=1, 2, \dots, t$) 的那几列 h_{n-i_j} 的线性组合, 由于 h_{n-i_j} 是 $n-k$ 重列矢量, 故 S 也是一个 $n-k$ 重的矢量 (s_1, s_2, \dots, s_{n-k})。若没有错误, 所有 $s_i=0$, 则 S 是一个零矢量。

如 $[7, 3, 4]$ 码, 设发送码字 $C=(1110100)$, $E=(1100000)$, 收到的 $R=(0010100)$, 由式(3.3.2)可知:

$$S^T = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{bmatrix}$$

它是 H 矩阵第一列与第二列之和。

若错误图样 $E=(0010000)$, 则

$$S^T = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

它是 H 矩阵的第三列。

若错误图样 $E=(0010100)$, 则

$$S^T = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

它是 \mathbf{H} 矩阵第三列与第五列之和。

由上面的一系列计算看出, 若错误图样中只有一个分量非零, 其它均为 0, 则 S^T 不为 $\mathbf{0}$, 且是 \mathbf{H} 矩阵相应的列。因此, 当在 n 长的码字中, 发生在不同位置上的单个错误, 就得到不同的非 $\mathbf{0}$ 伴随式(因为 \mathbf{H} 的每列均不同), 从而由这些不同的伴随式可求得不同的错误图样 E , 因而该 $[7, 3, 4]$ 码能纠正单个错误。若发生两个错误, 则 S^T 也不为 $\mathbf{0}$, 但不能纠正。如上面例中第一、第二位发生错误与第三、第五位发生错误, 所得到的伴随式均相同。因此, 由此伴随式不为 $\mathbf{0}$, 译码器只能判决传输有错($E \neq \mathbf{0}$), 但不能判定是由哪几位错误引起的。可见, 该码能纠正一个错误, 同时发现两个错误, 这是很显然的, 因为该码的最小距离为 4。

从上面分析看出, 一个 $[n, k, d]$ 码要纠正 $\leq t$ 个错误, 则要求 $\leq t$ 个错误的所有可能组合的错误图样, 都应该有不同的伴随式与之对应。这等效于: 若

$$(0, \dots, 0, e_{i_1}, e_{i_2}, \dots, e_{i_t}, 0, \dots, 0) \neq (0, \dots, 0, e_{j_1}, e_{j_2}, \dots, e_{j_t}, 0, \dots, 0)$$

则要求:

$$e_{i_1}\mathbf{h}_{i_1} + e_{i_2}\mathbf{h}_{i_2} + \dots + e_{i_t}\mathbf{h}_{i_t} \neq e_{j_1}\mathbf{h}_{j_1} + e_{j_2}\mathbf{h}_{j_2} + \dots + e_{j_t}\mathbf{h}_{j_t} \quad (3.3.4)$$

或

$$e_{i_1}\mathbf{h}_{i_1} + e_{i_2}\mathbf{h}_{i_2} + \dots + e_{i_t}\mathbf{h}_{i_t} - e_{j_1}\mathbf{h}_{j_1} - e_{j_2}\mathbf{h}_{j_2} - \dots - e_{j_t}\mathbf{h}_{j_t} \neq \mathbf{0}^T \quad (3.3.5)$$

由此可得到如下重要结论。

结论 一个 $[n, k, d]$ 线性分组码, 若要纠正 $\leq t$ 个错误, 则其充要条件是 \mathbf{H} 矩阵中任何 $2t$ 列线性无关。由于 $d=2t+1$, 所以也相当于要求 \mathbf{H} 矩阵中 $d-1$ 列线性无关。

由此结论可得到以下重要定理。

定理 3.3.1 $[n, k, d]$ 线性分组码有最小距离等于 d 的充要条件是, \mathbf{H} 矩阵中任意 $d-1$ 列线性无关。

证明 先证明必要性。即码有最小距离为 d , 证明 \mathbf{H} 中的任意 $d-1$ 列线性无关。

用反证法。若 \mathbf{H} 中某一 $d-1$ 列线性相关, 则由线性相关定义可知:

$$c_{i_1}\mathbf{h}_{i_1} + c_{i_2}\mathbf{h}_{i_2} + \dots + c_{i(d-1)}\mathbf{h}_{i(d-1)} = \mathbf{0}^T$$

式中, $c_{ij} \in GF(q)$, \mathbf{h}_{ij} 是 \mathbf{H} 矩阵的列矢量。现作一个码字 C , 它在 $i_1, i_2, \dots, i_{(d-1)}$ 位处的值分别等于 $c_{i_1}, c_{i_2}, \dots, c_{i(d-1)}$, 而其它各位取值均为 0, 所以得到的码字 C 是: $(0, \dots, c_{i_1}, 0, \dots, 0, c_{i_2}, 0, \dots, 0, c_{i(d-1)}, 0, \dots, 0)$, 由此

$$\mathbf{H} \cdot C^T = c_{i_1}\mathbf{h}_{i_1} + c_{i_2}\mathbf{h}_{i_2} + \dots + c_{i(d-1)}\mathbf{h}_{i(d-1)} = \mathbf{0}^T$$

故 C 是一个码字, 而 C 的非 0 分量个数只有 $d-1$ 个, 这与码有最小距离为 d 的假设相矛盾, 故 \mathbf{H} 中的任意 $d-1$ 列必线性无关。

下面证明: 若 \mathbf{H} 中任意 $d-1$ 列线性无关, 则 $[n, k, d]$ 码有最小距离为 d 。

若 \mathbf{H} 中任意 $d-1$ 列线性无关，则 \mathbf{H} 中至少需要 d 列才能线性相关。我们将能使 \mathbf{H} 中某些 d 列线性相关的列的系数，作为码字中对应的非 0 分量，而码字的其余分量均为 0，则该码字至少有 d 个非 0 分量，故 $[n, k, d]$ 码有最小距离为 d 。 ■

定理 3.3.1 异常重要，它是构造任何类型线性分组码的基础。由该定理看出，交换 \mathbf{H} 矩阵的各列，并不会影响码的最小距离。因此，所有列相同但排列位置不同的 \mathbf{H} 所对应的分组码，在纠错能力和其它码参数上完全等价。

推论 3.3.1 (Singleton 限) $[n, k, d]$ 线性分组码的最大可能的最小距离等于 $n-k+1$ ，即 $d \leq n-k+1$ 。

推论的证明读者可自行进行。若系统码的最小距离 $d=n-k+1$ ，则称此码为**极大最小距离可分码**，简称 MDS 码。构造 MDS 码是编码理论中一个重要课题。

二、汉明码与极长码

汉明码是 1950 年由汉明首先构造，用以纠正单个错误的线性分组码。由于它的编译码非常简单，很容易实现，因此用得很普遍，特别是在计算机的存贮和运算系统中更常用到。此外，它与某些码类的关系很密切，因此这是一类特别引人注意的码。

由定理 3.3.1 知，纠正一个错误的 $[n, k, d]$ 分组码，要求其 \mathbf{H} 矩阵中至少两列线性无关，且不能全为 0。若为二进制码，则要求 \mathbf{H} 矩阵中每列互不相同，且不能全为 0。

一个 $[n, k, d]$ 分组码有 $n-k$ 位校验元，在二进制码情况下，这 $n-k$ 个校验元能组成 2^{n-k} 列不同的 $n-k$ 重，其中有 $2^{n-k}-1$ 列不全为 0。所以，如果用这 $2^{n-k}-1$ 列作为 \mathbf{H} 矩阵的每一列，则由此 \mathbf{H} 就产生了一个纠正单个错误的 $[n, k, 3]$ 码，它就是汉明码。

定义 3.3.1 GF(2) 上汉明码的 \mathbf{H} 矩阵的列，是由不全为 0，且互不相同的二进制 m 重组。该码有如下参数： $n=2^m-1$, $k=2^m-1-m$, $R=(2^m-1-m)/(2^m-1)$, $d=3$ 。

例 3.3.3 构造 GF(2) 上的 $[7, 4, 3]$ 汉明码。这时取 $m=3$ ，所有 $2^3=8$ 个三重为：000, 100, 010, 001, 011, 101, 110, 111。挑出其中 7 个非 0 的三重构成

$$\mathbf{H} = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

若码字传输中第一位发生错误，则相应的伴随式 $S=(001)$ ，它就是“1”的二进制表示，若第五位发生错误，伴随式 $S=(101)$ ，是“5”的二进制表示。因此，哪一位发生错误，它的伴随式就是该位号码的二进制表示，所以能很方便地进行译码。 ■

由前知，任意调换 \mathbf{H} 中各列位置，并不会影响码的纠错能力。因此，汉明码的 \mathbf{H} 矩阵形式，除了上述表示外，还可以有其它形式。若把汉明码化成系统码形式，则其

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} = [-\mathbf{p}^T \mathbf{I}_3] \quad (3.3.6)$$

相应地

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} = [\mathbf{I}_4 \mathbf{p}] \quad (3.3.7)$$

可以把 GF(2) 上的汉明推广到 GF(q) 上, 得到多进制汉明码, 此时码有如下参数:

| | |
|---------|-------------------------|
| 码 长 | $n = (q^m - 1)/(q - 1)$ |
| 信 息 位 | $k = n - m$ |
| 码 率 | $R = (n - m)/n$ |
| 最 小 距 离 | $d = 3$ |

显然, 当 $n \rightarrow \infty$ 时, $R \rightarrow 1$, 因此汉明码是纠正单个错误的高效码。

二进制 $[2^m - 1, 2^m - 1 - m, 3]$ 汉明码的对偶码 C_H^\perp 是一个 $[2^m - 1, m, 2^{m-1}]$ 码, 也称为 **单纯码或极长码**。以后将看到, 它可以由 m 级线性移存器产生, 所以也称**最长线性移存器码**。

三、标准阵列

由前面的讨论中可知, $[n, k, d]$ 分组码的译码步骤可归结为以下三步:

- (1) 由接收到的 \mathbf{R} , 计算伴随式 $\mathbf{S} = \mathbf{R} \cdot \mathbf{H}^T$;
- (2) 若 $\mathbf{S} = \mathbf{0}$, 则认为接收无误。若 $\mathbf{S} \neq \mathbf{0}$, 则由 \mathbf{S} 找出错误图样 $\hat{\mathbf{E}}$;
- (3) 由 $\hat{\mathbf{E}}$ 和 \mathbf{R} 找出 $\hat{\mathbf{C}} = \mathbf{R} - \hat{\mathbf{E}}$ 。

汉明码的译码很简单, 它可由 \mathbf{S} 直接得到错误图样 $\hat{\mathbf{E}}$ 。但除了汉明码以外, 对其它分组码而言, 如何由 \mathbf{S} 求得 $\hat{\mathbf{E}}$ 就比较复杂。而一个译码器的复杂性及其译码错误概率也往往由这步决定。下面我们讨论分组码的一般译码方法, 这就是第一章已讲过的标准阵法, 它是由斯勒宾(Slepian)于 1956 年提出的, 是一种在 BSC 中译码错误概率最小的译码方法。

$[n, k, d]$ 码的 2^k 个码字, 组成了 n 维线性空间中的一个 k 维子空间, 显然是一子群。若以此子群为基础, 把整个 n 维空间的 2^n 个元素划分陪集, 则得到如表 3-2 所示的译码表。其中, 2^k 个码字放在表中的第一行, 该子群的恒等元素 \mathbf{C}_1 (全为 0 码字) 放在最左边, 然后在禁用码组中挑出一个 n 重 \mathbf{E}_2 放在恒等元素 \mathbf{C}_1 的下面, 并相应求出 $\mathbf{E}_2 + \mathbf{C}_2, \mathbf{E}_2 + \mathbf{C}_3, \dots, \mathbf{E}_2 + \mathbf{C}_{2^k}$, 分别放在 $\mathbf{C}_2, \mathbf{C}_3, \dots, \mathbf{C}_{2^k}$ 码字的下边构成第二行, 这是码空间的一个陪集。再选一个未写入表中前一行的 n 重 \mathbf{E}_3 , 用以上方法构成另一陪集成为表中的第三行, 依此类推, 一共构成 2^{n-k} 个陪集, 把所有 2^n 个矢量划分完毕, 称 $\mathbf{C}_1, \mathbf{E}_2, \mathbf{E}_3, \dots, \mathbf{E}_{2^{n-k}}$ 为陪集首。按这种方法构成的表称为**标准阵译码表**, 简称**标准阵**。

表 3-2 标准阵译码表

| 码字 | \mathbf{C}_1 (陪集首) | \mathbf{C}_2 | $\cdots \mathbf{C}_i$ | ... | \mathbf{C}_{2^k} |
|------------------|-------------------------|---------------------------------------|--|----------|---|
| 禁 用 码 字 | \mathbf{E}_2 | $\mathbf{C}_2 + \mathbf{E}_2$ | $\cdots \mathbf{C}_i + \mathbf{E}_2$ | ... | $\mathbf{C}_{2^k} + \mathbf{E}_2$ |
| | \mathbf{E}_3 | $\mathbf{C}_2 + \mathbf{E}_3$ | $\cdots \mathbf{C}_i + \mathbf{E}_3$ | ... | $\mathbf{C}_{2^k} + \mathbf{E}_3$ |
| | \vdots | \vdots | \vdots | \vdots | \vdots |
| | $\mathbf{E}_{2^{n-k}}$ | $\mathbf{C}_2 + \mathbf{E}_{2^{n-k}}$ | $\cdots \mathbf{C}_i + \mathbf{E}_{2^{n-k}}$ | ... | $\mathbf{C}_{2^k} + \mathbf{E}_{2^{n-k}}$ |

收到的 n 重 \mathbf{R} 落在某一列中, 则译码器就译成相应于该列最上面的码字。因此, 若发送的码字为 \mathbf{C}_i , 收到的 $\mathbf{R} = \mathbf{C}_i + \mathbf{E}_j$ ($1 \leq j \leq 2^{n-k}$, \mathbf{E}_1 是全 0 矢量), 则能正确译码。如果收到的 $\mathbf{R} = \mathbf{C}_k + \mathbf{E}_j$, 则产生了错误译码。现在的问题是: 如何划分陪集使译码错误概率最小? 这归结到如何挑选陪集首。因为一个陪集的划分主要决定于子群, 而子群就是 2^k 个码字, 这

已决定，因此余下的问题就是如何决定陪集首。

在第一章中已提出，在 $p_e \leq 0.5$ 的 BSC 中，产生 1 个错误的概率比产生 2 个的大，产生 2 个的比 3 个的大…… 总之，错误图样重量越小的产生的可能性越大。因此，译码器必须首先保证能正确纠正这种可能性出现最大的错误图样，也就是重量最轻的错误图样。这相当于在构造译码表时要求挑选重量最轻的 n 重为陪集首，放在标准阵中的第一列，而以全为 0 码字作为子群的陪集首。这样得到的标准阵，能得到最小的译码错误概率。由于这样安排的译码表使得 $C_i + E_j$ 与 C_i 的距离保证最小，因而也称为 **最小距离译码**，在 BSC 下，它们等效于 **最大似然译码**。

在标准阵中，所有陪集首重量之和称为陪集首的**总重量**。应当指出，在给定的 n, k 条件下，如果在 2^n 组中选择不同的 2^k 组作为子群，则相应的标准阵中，陪集首元素的总重量不一定相同。若所选的 V_{n-k} 子空间，能做到使标准阵中陪集首元素的总重量最轻，则此码能得到最大正确译码概率，因而称该 V_{n-k} 子空间构成的 $[n, k, d]$ 码为**最佳码**。如果在 n 维空间中，能找到两个或更多个 V_{n-k} 子空间，都能做到使标准阵中陪集首元素的总重量最轻，则这些子空间构成的不同的 $[n, k, d]$ 码均为最佳码，因此对固定的 n 和 k 来说，最佳码不是唯一的。

例 3.4 $[7, 4, 3]$ 汉明码的缩短码 $[6, 3, 3]$ 码，它的 8 个码组，可由式(3.3.7)的 G 矩阵中去掉第一行和第一列后的 $G_{[6,3]}$ 中的行线性组合而得到。该码的标准阵如表 3-3 所示。

表 3-3 $[6, 3]$ 码标准阵

| 码字 | 000000 (陪集首) | 100110 | 010011 | 001111 | 110101 | 101001 | 011100 | 111010 |
|------|-----------------|--------|---------|--------|--------|--------|--------|--------|
| 禁用码组 | 100000 | 000110 | 110011 | 101111 | 010101 | 001001 | 111100 | 011010 |
| | 010000 | 110110 | 000011 | 011111 | 100101 | 111001 | 001100 | 101010 |
| | 001000 | 101110 | 011011 | 000111 | 111101 | 100001 | 010100 | 110010 |
| | 000100 | 100010 | 0101110 | 001011 | 110001 | 101101 | 011000 | 111110 |
| | 000010 | 100100 | 010001 | 001101 | 110111 | 101011 | 011110 | 111000 |
| | 000001 | 100111 | 010010 | 001110 | 110100 | 101000 | 011101 | 111011 |
| | 110000 | 010110 | 100011 | 111111 | 000101 | 011001 | 101100 | 001010 |

该 $[6, 3, 3]$ 码标准阵陪集首的总重量为 8，而表 1-2 中码标准阵陪集首的总重量也为 8，这说明这两个 $[6, 3]$ 码均是最佳码。

由该表可以看到，用这种标准阵译码，需要把 2^n 个 n 重存储在译码器中。所以，这种译码方法译码器的复杂性随 n 指数增长，很不实用。 ■

根据错误图样与伴随式之间的一一对应关系，可把上述标准阵译码表进行简化，得到一个简化译码表。如上例中的 $[6, 3, 3]$ 码标准阵，可简化为表 3-4 所示的译码表。译码器收到 R 后，与 H 进行运算得到伴随式 S ，由 S 查表得到错误图样 \hat{E} ，从而译出码字 $\hat{C} = R - \hat{E}$ 。因此，这种译码器中不必存贮所有 2^n 个 n 重，而只存贮错误图样 E 与 2^{n-k} 个 $(n-k)$ 重 S 。

表 3-4 [6, 3]码简化译码表

| | | | | | | | | |
|------|--------|--------|--------|--------|--------|--------|--------|--------|
| 错误图样 | 000000 | 100000 | 010000 | 001000 | 000100 | 000010 | 000001 | 110000 |
| 伴随式 | 000 | 110 | 011 | 111 | 100 | 010 | 001 | 101 |

由于 $[n, k, d]$ 分组码的 n, k 通常都比较大，即使用这种简化译码表，译码器的复杂性还是很高的。例如，一个 $[100, 70]$ 分组码，一共有 $2^{30} \approx 10^9$ 个伴随式及错误图样，译码器要存贮如此多的图样和 $(n-k)$ 重是不太可能的。因此，在线性分组码理论中，如何寻找简化译码器是最中心的研究课题之一，这将在以后有关章节讨论。

四、完备译码与限定距离译码

例 3.4 中的 $[6, 3, 3]$ 码利用标准阵译码时，能纠正所有单个错误及一种两个错误，码的所有 $2^3 = 8$ 个伴随式都与某种类型错误图样对应。因此，译码器必然要对收到的 \mathbf{R} 进行判决发送的是何码字，这种译码方法称为**完备译码**。

定义 3.3.2 $[n, k, d]$ 线性分组码的所有 2^{n-k} 个伴随式，在译码过程中若都用来纠正所有小于等于 $t = \lfloor (d-1)/2 \rfloor$ 个随机错误，以及部分大于 t 的错误图样，则这种译码方法称为**完备译码**。否则，称为**非完备译码**。

任一个 $[n, k, d]$ 码，能纠正 $t \leq \lfloor (d-1)/2 \rfloor$ 个随机错误。如果在译码时仅纠正 $t' < t$ 个错误，而当错误个数大于 t' 时，译码器不进行纠错而仅指出发生了错误，称这种译码方法为**限定距离译码**。

如 $[15, 4, 8]$ 极长码，它能纠正 3 个错误及部分 4 个错误。如果设计译码器时，仅使它纠正 1 个错误，而在 ≥ 2 个错误时，只指出接收的 \mathbf{R} 有错，但不进行纠正；则这种译码器就称为**限定距离译码器**。

可知限定距离译码，就是设计译码器时，在 0 至 $t = \lfloor (d-1)/2 \rfloor$ 范围内，事先由设计者指定纠错能力 t' 。当实际产生的错误个数 $\leq t'$ 时，译码器进行纠错译码；当大于 t' 时，译码器进行检错而不纠错。可见，限定距离译码可以是完备译码，也可以是非完备译码，它完全由译码设计者决定。

应当指出，无论是何种译码方法，为了使译码错误概率最小，在设计译码器时都必须遵循最大似然译码准则（码字为等概发送时），在对称无记忆信道中也就是最小汉明距离译码准则。

* § 3.4 线性码的覆盖半径

从几何上讲，码的陪集划分就是把 n 维线性空间 V_n ，按 $[n, k, d]$ 码 C 划分空间。标准阵译码表中的第 j 列，相当于 V_n 中球心为码字 \mathbf{C}_j 、半径为 $\rho = d(\mathbf{C}_j, \mathbf{C}_j + \mathbf{E}) = w(\mathbf{E}_i)$ 的一个球 $B_j(C)$ ，球中的点也就是 \mathbf{C}_j 列中的 n 重：

$$\{\mathbf{V} \in V_n \mid d(\mathbf{v}, \mathbf{C}_j) \leq \rho\} \quad j = 0, 1, 2, \dots, 2^k - 1$$

表中共有 2^k 列，相当于有 2^k 个这种互不相交的球，把整个 V_n 空间覆盖完毕。可知 $B_j(C)$ 球的半径 ρ ，就是码 C 的最大可能的纠错数目。表 3-3 中 $[6, 3]$ 码的 ρ 是 2，也就是它至多可能纠正两个错误，但不能纠正所有两个错误的图样。

如果在 V_n 空间中, 以码字为圆心, $t = \lfloor (d-1)/2 \rfloor$ 为半径作球, 如图 1-15 所示, 则也有 2^k 个互不相交的球, 但这些球一般并不能把整个 V_n 空间全部覆盖完毕。称这些球的半径为码 C 的球半径 $s(C)$ 。可知码 C 的球半径

$$S(C) = \lfloor (d-1)/2 \rfloor \quad (3.4.1)$$

能把整个 V_n 空间覆盖完毕的 $B_t(C)$ 球的半径 ρ 称为码的覆盖半径 $t(C)$ 。可知 $t(C)$ 与 $s(C)$ 均是码的重要的几何参数。

定义 3.4.1 码 C 的覆盖半径

$$t(C) = \max \{ \min \{ d(v, C_j) \mid C_j \in C \}; v \in V_n \} \quad (3.4.2)$$

通常称 $C_j + v$ 为码字 C_j 的平移 ($C_j \in C, v \in V_n$), 称有 $\min w(v)$ 的 v 为平移首, 因此 $t(C)$ 就是有最大重量的平移首的重量。对线性码来说, 就是陪集首的最大重量, 而球半径 $s(C)$ 是码一定能全部纠正的错误数目。显然

$$s(C) \leq t(C) \leq d - 1 \quad (3.4.3)$$

如果码 C 的 $t(C) = s(C)$, 则称 C 码是完备码, 如果

$$t(C) = s(C) + 1 \quad (3.4.4)$$

则称为准完备码。当 n 为奇数时, $[n, 1, n]$ 重复码是完备码; 而当 n 为偶数时是准完备码。重复码又称为平凡码。

由最大似然译码原理可知, 要使译码错误概率最小, 必须使译码表中陪集首的重量最轻。因此, 在同样的码参数下, $t(C)$ 越小的码译码错误概率越小, 因而最好。可知 $t(C)$ 是衡量纠错码性能的又一重要参数。在同样的 n 与码字数目下, 完备码与准完备码都能使 $t(C)$ 最小, 因而是最佳码。

一般情况下, 我们希望在同样的 n, k 下, 构造出具有最大距离的码, 并且具有最小的 $t(C)$ 。但是, 由于构造方法的不同, 这二者之间并不完全一致, 有最大距离的码, 其覆盖半径不一定小。在同样的 n, k 下, 码所能达到的最小覆盖半径用 $t(n, k)$ 表示, 线性码用 $t[n, k]$ 表示。下面几个定理说明了二进制分组码的 $s(C)$ 与 $t(C)$ 的关系。

定理 3.4.1 对任何二进制 (n, k) 码 C , 必满足:

$$\sum_{0 \leq i \leq s(C)} \binom{n}{i} \leq \frac{2^n}{K} \leq \sum_{0 \leq i \leq t(C)} \binom{n}{i} \quad (3.4.5)$$

式中, K 是码字数目。若 C 为 $[n, k]$ 线性码, 且 n 为偶数, 则必须满足:

$$\begin{aligned} \sum_{2i \leq t(C)} \binom{n}{2i} &\geq 2^{n-k-1} \\ \sum_{2i+1 \leq t(C)} \binom{n}{2i+1} &\geq 2^{n-k-1} \end{aligned} \quad (3.4.6)$$

该定理称为球包和球包覆盖限, 证明它比较容易。由该定理不难得得到 $t(C)$ 的下限。

定理 3.4.2 二进制 (n, k, d) 码的覆盖半径:

$$t(n, k) \geq \frac{n}{2} - 2^{-3/2}(kn)^{1/2} \quad (3.4.7)$$

和

$$t(n, k) \geq \frac{n}{2} - (2k)^{1/2} \ln 2k \quad (3.4.8)$$

$$t(n, k) \geq \frac{n}{2} - 8(2k \ln 2k)^{1/2} \quad (3.4.9)$$

下面给出 $t(C)$ 的上限。

定理 3.4.3 若 $2 \leq k \leq 1 + \ln n$, 则 $[n, k]$ 线性码的

$$t[n, k] \geq \lceil \frac{n}{2} \rceil - 2^{k-2} \quad (3.4.10)$$

定理 3.4.4 对某一给定的 k , 存在有整数 n_1, n_2, \dots, n_q 使

$$k > A = \sum_{1 \leq i \leq q} (2^{n_i} - n_i - 1)$$

则当 $n \geq k$ 时有

$$t[n, k] \leq \left\lfloor \frac{1}{2}(n - k + 1 - \sum_{1 \leq i \leq q} n_i) \right\rfloor + q \quad (3.4.11)$$

式中, $\lceil x \rceil$ 表示不小于 x 的最小整数, $\lfloor x \rfloor$ 表示 x 的整数部分。

有关这些限的证明, 和其它各种情况下的上、下限及目前已知的各种码的覆盖半径, 以及 $t(n, k)$ 和 $t[n, k]$, 请参阅文献[6—9]。如同寻求一个码的实际最小距离一样, 当码长和信息位较大时, 如何分析寻求一个码的覆盖半径是一个很困难的问题。覆盖半径不仅与译码错误概率及码的最小距离有关, 而且还涉及到其它问题, 如好码的构造、数据压缩中的失真度等问题, 近来日益引起人们的广泛注意。

§ 3.5 由一个已知码构造新码的简单方法

前面介绍的缩短码和对偶码, 都是在已知 $[n, k]$ 码基础上进行适当修正后得到的。下面再介绍一些对已知码的 G 和 H 矩阵进行适当修正和组合, 以构造新码的方法。这些方法虽然很简单, 但很实用, 而且也是以后构造各种复合码的基础。

一、扩展码

设 C 是一个最小距离为 d 的二进制 $[n, k, d]$ 线性分组码, 它的码字有奇数重量也有偶数重量。若对每一个码字 $(c_{n-1}, c_{n-2}, \dots, c_1, c_0)$ 增加一个校验元 c'_0 , 满足以下校验关系:

$$c_{n-1} + c_{n-2} + \dots + c_1 + c_0 + c'_0 = 0 \quad (3.5.1)$$

称 c'_0 为全校验位。

因此, 若原来码字的重量为奇数, 则应用式(3.5.1), 在加上 c'_0 全校验位以后, 码字重量增加了1, 变为偶数; 当然码长也相应地增加了一位, 由原来的 n 变成 $n+1$ 。若原来码字的重量为偶数, 则加上 c'_0 后, 码字重量仍没有变化(此时 $c'_0 = 0$)。所以, 加了满足式(3.5.1)的全校验位 c'_0 后, $[n, k, d]$ (d 为奇数) 码变成了 $[n+1, k, d+1]$ 线性分组码, 称该码为 $[n, k, d]$ 码 C 的扩展码 \hat{C} 。扩展码的覆盖半径 $t(\hat{C}) = t(C) + 1$ 。

若原码的校验矩阵为 H , 则扩展码 \hat{C} 的校验矩阵 \hat{H} 为

$$\hat{H} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ & & & 0 \\ H & & & \vdots \\ & & & 0 \end{bmatrix} \quad (3.5.2)$$

$[2^m-1, 2^m-1-m, 3]$ 汉明码的扩张码是 $[2^m, 2^m-1-m, 4]$ 码，它的 $\hat{\mathbf{H}}$ 中的 \mathbf{H} 是汉明码的校验矩阵。

例3.5 对例3.3中的 $[7, 4, 3]$ 汉明码，附加一个全校验位，则得到了一个 $[8, 4, 4]$ 扩展汉明码，它的校验矩阵由式(3.3.6)可得：

$$\hat{\mathbf{H}} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \quad (3.5.3)$$

可以检验它是一个双偶自对偶码。 ■

二、删余码

删余码是由扩展码的逆过程而得到的。它在原 $[n, k]$ 码 C 的基础上，删去一个校验元而构成，变为 $[n-1, k]$ 删余码 C^* 。 C^* 码的最小距离可能比原码小1，也可能不变。

例如把上例中的 $[8, 4, 4]$ 码，删去最后一个校验位 c_0 ，便变成了 $[7, 4, 3]$ 汉明码。

三、增广码(增信删余码)

增广码 C^a 是在原码 C 的基础上，增加一个信息元，删去一个校验元得到的。因此，码长与原码同，但信息位增加了一个。

设原码 C 是一个没有全1码字的 $[n, k, d]$ 二进制码，在它的 \mathbf{G} 矩阵上增加一组全为“1”的行，便得到了增广码 C^a 的生成矩阵 \mathbf{G}^a ：

$$\mathbf{G}^a = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \mathbf{G} \end{bmatrix}$$

或

$$C^a = C \cup (\mathbf{1} + C_i) \quad i = 1, 2, \dots, 2^k \quad (3.5.4)$$

可知增广码 C^a 是一个 $[n, k+1, d_a]$ 分组码，其最小距离 d_a 由下式决定：

$$d_a = \min\{d, n - d'\} \quad (3.5.5)$$

式中， d' 是原码 C 中码字的最大重量。

如例 3.1 中的 $[7, 3, 4]$ 码对它进行增广变成为 $[7, 4, 3]$ 汉明码，它的生成矩阵由式(3.2.10)为

$$\mathbf{G}^a = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix} \quad (3.5.6)$$

四、增余删信码

和增广码构造过程相反的是增余删信码。它是在原码的基础上，删去一个信息位增加一个校验位得到的，其实这个过程就是在原来的二进制 $[n, k, d]$ (d 为奇数)码上，挑选所有偶数重量的码字组成一个新码，该码就是增余删信码。由定理 3.3.1 可知，偶数重量码字数是原码字数的一半，因此增余删信码是一个 $[n, k-1, d+1]$ 码。

如在 $[7, 4, 3]$ 汉明码中挑出所有重量为4的码字，便得到一个 $[7, 3, 4]$ 增余删信汉明码。

五、延长码(增信码)与 RM 码

延长码是在原 $[n, k]$ 码 C 的基础上，先进行增广然后再加一个全校验位构成的。因此，延长码是一个 $[n+1, k+1]$ 分组码。延长码的码率

$$R = \frac{k+1}{n+1}$$

比原码的码率 $R=k/n$ 要大一些，其码的最小距离也可能与原码相同。

例如，可把 $[7, 3, 4]$ 增余删信汉明码先进行增广，变成 $[7, 4, 3]$ 汉明码，然后再增加一个全校验位，变成 $[8, 4, 4]$ 扩展汉明码。该码比原来的 $[7, 3, 4]$ 码的 R 要高，而最小距离相同。

现以 $[7, 3, 4]$ 汉明码为例，把上述各类码之间的关系，画于图 3-2 中。

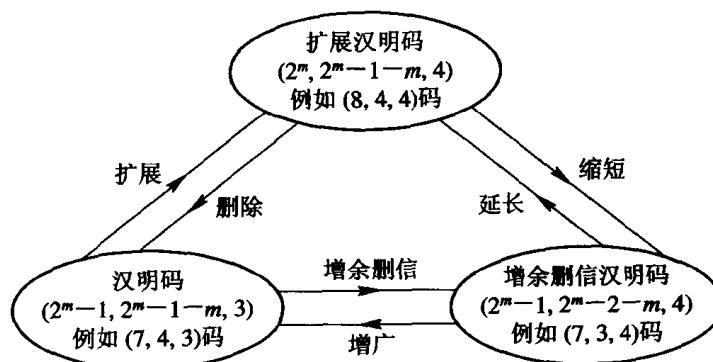


图 3-2 汉明码的各类修正码之间的关系图

如果把 $(2^m-1, 2^m-1-m, 3)$ 汉明码的对偶码，也就是单纯码 $(2^m-1, m, 2^{m-1})$ 进行延长，就得到一个 $(2^m, m+1, 2^{m-1})$ 码，称它为一阶里德—谬勒尔(Reed—Muller 码)用 RM($1, m$)表示。RM 码是 Muller 于 1954 年提出其构造方法，同年 Reed 用大数逻辑译码方法解决了它的译码。RM 码最早是从线性空间的角度出发构造的，以后发现它与循环码、几何码和格等有密切关系，因此这是一类很重要的线性码^[2,3]。

一般而言， r 阶 RM 码 RM(r, m) 是 $[2^m, k, 2^{m-r}]$ 码，其中

$$k = 1 + \binom{m}{1} + \binom{m}{2} + \cdots + \binom{m}{r}$$

$$n - k = 1 + \binom{m}{1} + \cdots + \binom{m}{m-r-1}$$

所以它的对偶码 $[2^m, 2^m-k, 2^r]$ 码是一个 $m-r-1$ 阶的 RM($m-r-1, m$) 码。

现以 $m=3$ 为例说明 RM 码的生成，RM($r, 3$)码从以下矢量中挑选构造 \mathbf{G} 矩阵的行。

$$\mathbf{V}_0 = (11111111)$$

$$\mathbf{V}_3 = (00001111)$$

$$\mathbf{V}_2 = (00110011)$$

$$\begin{aligned}
V_1 &= (01010101) \\
V_3 V_2 &= (00000011) \\
V_3 V_1 &= (00000101) \\
V_2 V_1 &= (00010001) \\
V_3 V_2 V_1 &= (00000001)
\end{aligned}$$

如果码以 V_0, V_3, V_2, V_1 作为 \mathbf{G} 矩阵的行，则得到一个 RM(1, 3) 码的生成矩阵。可以证明，RM(1, 3) 码是一个 E_8 格。如果挑选 V_0 至 $V_2 V_1$ 等 7 个矢量作为 \mathbf{G} 矩阵的行，则得到一个二阶 RM(2, 3) 码。可以验证用这种方法构造的 RM(1, 3) 码，经过适当的线性组合就是 [7, 3, 4] 单纯码进行延长得到的码。有关 RM 码的详细讨论及译码方法请参阅 [2]。

* § 3.6 用多个已知码构造新码的方法

除了上节介绍的用一个已知码构造新码以外，常常还用两个或多个已知码构造新码。下面将介绍一些主要的构造方法。

一、直和

设 C_1 和 C_2 分别是 $[n_1, k_1, d_1]$ 和 $[n_2, k_2, d_2]$ 二进制码，且 $n_1 = n_2$, $C_1 \cap C_2 = 0$ ，则定义 C_1 和 C_2 码的直和码 $C_1 \oplus C_2 = C$ 是

$$C = \{(a \oplus b) | a \in C_1, b \in C_2\} \quad (3.6.1)$$

式中， \oplus 表示 a, b 两个码字按位模 2 加。直和码 C 的生成矩阵

$$\mathbf{G} = \begin{bmatrix} \mathbf{G}_1 \\ \mathbf{G}_2 \end{bmatrix} \quad (3.6.2)$$

式中， \mathbf{G}_1 和 \mathbf{G}_2 分别是 C_1 和 C_2 码的生成矩阵。由此 \mathbf{G} 矩阵就生成一个 $[n, k_1 + k_2, d]$ 码， $d \leq \min\{d_1, d_2\}$ 。

例如， C_1 是一个 [7, 1, 7] 重复码， C_2 是 [7, 3, 4] 单纯码，且 $C_1 \cap C_2 = 0$ 。它们的生成矩阵分别是：

$$\mathbf{G}_1 = [1111111] \quad \mathbf{G}_2 = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix} \quad (3.6.3)$$

则 $C_1 \oplus C_2 = C$ 码的生成矩阵

$$\mathbf{G} = \begin{bmatrix} \mathbf{G}_1 \\ \mathbf{G}_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

可知这是一个 [7, 4, 3] 汉明码。

若用 l 个等码长，且除全 0 码字外均互不相交的线性分组码 C_1, C_2, \dots, C_l 直和，则得到一个 $[n, k_1 + k_2 + \dots + k_l, d]$ 线性分组码， $d \leq \min\{d_1, d_2, \dots, d_l\}$ ， \mathbf{G} 矩阵与式 (3.6.2) 形式相同，为 l 个生成矩阵之和。

如果 C_1 和 C_2 码的覆盖半径分别是 $t(C_1)$ 和 $t(C_2)$ ，则直和码 C 的覆盖半径

$$t(C) \leq \min(t(C_1), t(C_2)) \quad (3.6.4)$$

上例中 $t(C) \leq \min(3, 2)$, 实际上 $t(C)=1$ 。若为多个码直和, 则

$$t(C) \leq \min(t(C_1), t(C_2), \dots, t(C_i))$$

二、笛卡儿积

由 C_1 和 C_2 码的笛卡儿积 $C_1 \times C_2$ 得到的码是

$$C = C_1 \times C_2 = \{(\mathbf{a}, \mathbf{b}) | \mathbf{a} \in C_1, \mathbf{b} \in C_2\} \quad (3.6.5)$$

可知, C 码是一个 $[n_1+n_2, k_1+k_2, \min\{d_1, d_2\}]$ 码。该码的生成矩阵

$$\mathbf{G} = \begin{bmatrix} \mathbf{G}_1 & \mathbf{0}_2 \\ \mathbf{0}_1 & \mathbf{G}_2 \end{bmatrix} \quad (3.6.6)$$

式中, $\mathbf{0}_1$ 和 $\mathbf{0}_2$ 分别是 $k_2 \times n_1$ 阶和 $k_1 \times n_2$ 阶全 0 阵。例如, C_1 是 $[3, 1, 3]$ 重复码, C_2 是 $[7, 3, 4]$ 单纯码, 则它们的笛卡儿积码 C 的生成矩阵

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

由此得到一个 $[10, 4, 3]$ 码。

笛卡儿积 $C_1 \times C_2 = C$ 码的覆盖半径

$$t(C) = t(C_1) + t(C_2) \quad (3.6.7)$$

该例中 $t(C) = t(C_1) + t(C_2) = 1 + 2 = 3$ 。

三、链接

C_1 和 C_2 码的链接码 $C = C_1 + C_2$, 是一个 $[n_1 + n_2, k_2, d]$ 码, 这里要求 $k_2 \geq k_1$, $d \geq \min\{d_1, d_2\}$ 。链接码 C 的生成矩阵

$$\mathbf{G} = \begin{bmatrix} \mathbf{G}_1 \\ \vdots \\ \mathbf{G}_2 \\ \mathbf{0} \end{bmatrix} \quad (3.6.8)$$

式中, $\mathbf{0}$ 是 $(k_2 - k_1) \times n_1$ 阶全 0 矩阵。如 C_1 码是 $[3, 1, 3]$ 重复码, C_2 是 $[7, 3, 4]$ 单纯码, 则 $C = C_1 + C_2$ 码的生成矩阵

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

得到一个 $[10, 3, 4]$ 码。链接码的覆盖半径

$$t(C) \geq t(C_1) + t(C_2) \quad (3.6.9)$$

该例中 $C_1 + C_2 = C$ 码的覆盖半径是 3。

四、 $[C_1, C_1 + C_2]$ 构造

设 $n_1 = n_2$, 且 $C_2 \subseteq C_1$, 则 C 码是

$$C = \{(a, a+b); a \in C_1, b \in C_2\} \quad (3.6.10)$$

可知要求 C_1 与 C_2 码的码长相等, $n_1 = n_2 = n$ 。 C 码的生成矩阵

$$G = \begin{bmatrix} G_1 & G_1 \\ \mathbf{0}_2 & G_2 \end{bmatrix} \quad (3.6.11)$$

式中, $\mathbf{0}_2$ 是 $k_2 \times n_1$ 阶全 0 阵。由此 G 得到一个 $[2n, k_1 + k_2, \min\{2d_1, d_2\}]$ 码。

例如, C_1 是 $[7, 4, 3]$ 汉明码, C_2 是 $[7, 1, 7]$ 重复码, 显然 $C_2 \subseteq C_1$, 因此 C 码的生成矩阵

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

得到一个 $[14, 5, 6]$ 码。

这类码的覆盖多项式

$$t(C) \geq t(C_1) + t(C_2) \quad (3.6.12)$$

此例中 $t(C_1) = 1$, $t(C_2) = 3$, 因此 $t(C) \geq 4$, 实际上为 4。

五、直积(Kronecker 积)

C_1 和 C_2 码的直积 $C_1 \otimes C_2 = C$, 是一个 $[n_1 n_2, k_1 k_2, d_1 d_2]$ 码。 C 码的生成矩阵

$$G = G_1 \otimes G_2 = \begin{bmatrix} g_{11}G_2 & \cdots & g_{1n_1}G_2 \\ \vdots & & \vdots \\ g_{k_1}G_2 & \cdots & g_{k_1n_1}G_2 \end{bmatrix} \quad (3.6.13)$$

式中, g_{ij} 是 C_1 码生成矩阵中第 i 行第 j 列元素。

如 C_1 是 $[7, 3, 4]$ 码, C_2 是 $[3, 1, 3]$ 码, 则由式(3.6.3)中的 G_2 和重复码的 G_1 矩阵, 可得 $C_1 \otimes C_2 = C$ 码的生成矩阵

$$G = \begin{bmatrix} G_2 & 0 & 0 & G_2 & G_2 & G_2 & 0 \\ 0 & G_2 & 0 & 0 & G_2 & G_2 & G_2 \\ 0 & 0 & G_2 & G_2 & G_2 & 0 & G_2 \end{bmatrix} = \begin{bmatrix} 111 & 000 & 000 & 111 & 111 & 111 & 000 \\ 000 & 111 & 000 & 000 & 111 & 111 & 111 \\ 000 & 000 & 111 & 111 & 111 & 000 & 111 \end{bmatrix}$$

得到一个 $[21, 3, 12]$ 直积码。

直积码的覆盖半径

$$t(C) \geq \max\{n_1 t(C_2), n_2 t(C_1)\} \quad (3.6.14)$$

该例中 $t(C) = 7$ 。

除了上面所介绍的这些方法外, 还有其它一些用数个码构造新码的方法, 这将在以后各章介绍。上面介绍的用两个码构造新码的方法也可推广到用多个码构造新码。如可用 l 个码直积以得到新码 $C = C_1 \otimes C_2 \otimes \cdots \otimes C_l$ 。当然也可以把上述几种方法组合而得到新码, 如先直积再直和得到新码 $C = C_1 \otimes C_2 \otimes C_3 \otimes C_4$; 或先直和再直积等等。并且也不局限于线性码, 也可用非线性码组合, 或用线性与非线性码组合等等。总之, 各种方法灵活组合, 就可能由已知的好码构造出新的好码。

§ 3.7 线性码的重量分布与译码错误概率

$[n, k, d]$ 线性分组码的不可检错误概率和译码错误概率的计算，是估计 FEC 和 ARQ 等差错控制系统性能的基础。但译码错误概率和不可检错误概率的计算，又与码的重量分布密切相关。

一、线性码的重量分布

所谓码的重量分布是指一个 $[n, k, d]$ 线性分组码或非线性码的码字重量的分布情况，它不仅是计算各种译码错误概率的主要依据之一，而且也是探索码结构的重要窗口，通过它能透彻地了解码的内部关系。

设 A_i 是 $[n, k, d]$ 分组码中重量为 i 的码字数目，则集合 $\{A_0, A_1, \dots, A_n\}$ 称为该分组码的重量分布。

例如 $[7, 4, 3]$ 汉明码的重量分布为

$$\{A_i\} = \{A_0, A_1, A_2, A_3, A_4, A_5, A_6, A_7\} = \{1, 0, 0, 7, 7, 0, 0, 1\}$$

其中，一个全0码字($A_0=1$)是任何线性码所必须有的。

也可把码的重量分布 $\{A_0, A_1, \dots, A_n\}$ 写成如下形式的多项式：

$$A(x) = A_0 + A_1x + \dots + A_nx^n = \sum_{i=0}^n A_i x^i \quad (3.7.1)$$

称 $A(x)$ 为码的重量估值算子，简称重量算子。

定理3.7.1 设二进制 $[n, k]$ 线性分组码及其 $[n, n-k]$ 对偶码的重量算子分别是：

$$A(x) = \sum_{i=0}^n A_i x^i \quad B(x) = \sum_{i=0}^n B_i x^i$$

则它们之间有如下关系：

$$A(x) = 2^{-(n-k)}(1+x)^n B\left(\frac{1-x}{1+x}\right) \quad (3.7.2)$$

称此式为马克威伦(MacWilliams)恒等式。

证明 设 C 是二进制 n 重，它的汉明重量 $w(C)=w$ ，产生该 n 重的概率定义为

$$P(C) = \epsilon^w (1-\epsilon)^{n-w} \quad 0 \leq \epsilon \leq 1$$

设 H 为码的校验矩阵，则 C 的伴随式 $S(C)=HC^T=S$ ，这里 C^T 是 C 的转置。令 E 是使 $S(C)=0$ 的事件，现计算事件 E 出现的概率 $P(E)$ 。

显然，当且仅当 C 是码的码字时， $S=0$ 。所以 $P(E)$ 是所有码字出现的概率之和。令码

的重量估值算子 $A(x)=\sum_{i=0}^n A_i x^i$ ，则

$$P(E) = \sum_{i=0}^n A_i \epsilon^i (1-\epsilon)^{n-i} = \sum_{i=0}^n A_i \left(\frac{\epsilon}{1-\epsilon}\right)^i (1-\epsilon)^n = (1-\epsilon)^n A\left(\frac{\epsilon}{1-\epsilon}\right) \quad (3.7.3)$$

由 H 行的所有线性组合所生成的全部矢量，组成了 H 的扩张校验矩阵 H^* 。所以， H^* 有 2^{n-k} 行，它们就是 $[n, k]$ 码对偶码的所有码字。对任何二进制 n 重 C ，定义它的扩张伴随式 $S^*(C)=H^*C^T=S^*$ 。

引理3.7.1 当且仅当 $S(C)=\mathbf{0}$ 时, $S^*(C)=\mathbf{0}$ 。若 $S^*(C)\neq\mathbf{0}$, 则它的 2^{n-k} 个分量中有一半是 0 一半是 1。

该引理是显而易见的, 请读者证明。令 E_j 和 \bar{E}_j , $j=1, 2, \dots, 2^{n-k}$ 分别是 $S^*(C)$ 的第 j 个分量是 0 和 1 的事件。所以, $S(C)=\mathbf{0}$ 的概率 $P(E)$, 也就是 $S^*(C)=\mathbf{0}$ 的概率, 可知:

$$P(E) = 1 - P(\bar{E}_1 \cup \bar{E}_2 \cup \dots \cup \bar{E}_{2^{n-k}})$$

由引理3.7.1知, 若 $S^*(C)\neq\mathbf{0}$, 则在 \bar{E}_j 事件中, 正确地有 2^{n-k-1} 个事件出现, 若 $S^*(C)=\mathbf{0}$, 则没有一个 \bar{E}_j 事件出现, 即

$$\sum_{j=1}^{2^{n-k}} P(\bar{E}_j) = 2^{n-k-1} \times (S^*(C) \neq \mathbf{0} \text{ 出现的概率})$$

所以

$$P(E) = 1 - \frac{1}{2^{n-k-1}} \sum_{j=1}^{2^{n-k}} P(\bar{E}_j) \quad (3.7.4)$$

若 H^* 的第 j 行重量为 w_j , 则 $P(\bar{E}_j)$ 是在相应的 w_j 个位置上, C 有奇数个 1 的概率, 因此

$$P(\bar{E}_j) = \sum_{\substack{l=0 \\ l=\text{奇数}}}^{w_j} \binom{w_j}{l} \epsilon^l (1-\epsilon)^{w_j-l}$$

由概率论的基本知识可知:

$$\sum_{\substack{l=0 \\ l=\text{奇数}}}^K \binom{K}{l} a^l b^{K-l} = \frac{1}{2} [(b+a)^K - (b-a)^K]$$

所以

$$P(\bar{E}_j) = \frac{1}{2} (1 - 2\epsilon)^{w_j} \quad (3.7.5)$$

由式(3.7.4)和式(3.7.5)可得:

$$P(E) = 1 - \frac{1}{2^{n-k-1}} \sum_{j=1}^{2^{n-k}} \left(\frac{1}{2} - \frac{1}{2} (1 - 2\epsilon)^{w_j} \right)$$

或

$$P(E) = 2^{-(n-k)} \sum_{i=0}^n B_i (1 - 2\epsilon)^i \quad (3.7.6)$$

式中, B_i 是 H^* 中有重量为 i 的行数。令 $B(x)$ 是对偶码的重量算子, 则上式成为

$$P(E) = 2^{-(n-k)} B(1 - 2\epsilon) \quad (3.7.7)$$

由式(3.7.7)和式(3.7.3), 并代入 $x/(1+x)=\epsilon$, 则得

$$A(x) = 2^{-(n-k)} (1+x)^n B\left(\frac{1-x}{1+x}\right) \quad \blacksquare$$

对于 q 进制 $[n, k]$ 线性分组码, 用类似方法可以证明有

$$A(x) = q^{-(n-k)} (1 + (q-1)x)^n B\left(\frac{1-x}{1+(q-1)x}\right) \quad (3.7.8)$$

一旦对偶码的重量分布已知时, 就可通过上面两个 MacWilliams 恒等式求得码的重量分布。

$[2^m-1, 2^m-1-m, 3]$ 汉明码的对偶码是 $[2^m-1, m, 2^{m-1}]$ 极长码, 这是一个等重码。它除了一个全 0 码字外, 其余码字的重量都等于 2^{m-1} , 所以

$$B(x) = 1 + (2^m - 1)x^{2^m-1} \quad (3.7.9)$$

由 MacWilliams 恒等式(3.7.2)可得汉明码的重量算子为

$$A(x) = \frac{1}{n+1}[(1+x)^n + n(1-x)(1-x^2)^{\frac{n-1}{2}}] \quad (3.7.10)$$

这里, $n=2^m-1$ 。而 $[2^m, 2^m-1-m, 4]$ 扩张汉明码的重量算子为

$$A'(x) = \frac{1}{2n}[(1+x)^n + (1-x)^n + 2(n-1)(1-x^2)^{\frac{n}{2}}] \quad (3.7.11)$$

如果汉明码的重量分布为

$$\{A_i\} = \{1, 0, 0, A_3, A_4, \dots, A_n\}$$

则它的扩展汉明码的重量分布为

$$\{A'_i\} = \{1, 0, 0, 0, A_3 + A_4, 0, A_5 + A_6, 0, \dots, A_{n-1} + A_n\}$$

其中没有奇重量码字, 且 $A'_{j(\text{偶数})} = A_{j-1} + A_j$ 。例如, $[7, 4, 3]$ 汉明码的 $\{A_i\} = \{1, 0, 0, 7, 7, 0, 0, 1\}$, 而它的 $[8, 4, 4]$ 扩展汉明码的 $\{A'_i\} = \{1, 0, 0, 0, 14, 0, 0, 0, 1\}$ 。

任何一个二进制 $[n, k, d]$ 线性分组码的码字集合中, 必有一个全 0 码字, 如果还有一个全 1 码字, 则由于线性码的封闭性, 该码的重量分布必是对称的: $A_i = A_{n-i}$, 如前面列举的 $[7, 4, 3]$ 汉明码。

除了少数几类码的重量分布是已知的外, 还有很多码的重量分布并不知道, 特别当 n 和 k 较大时, 要得到码的重量分布更为困难。事实上已证明, 线性码的重量分布、最小距离、覆盖半径和译码等问题, 除少数几类码是已知以外, 一般要解决它们都是很困难的, 是一个 NP- 完全问题。

二、线性分组码的不可检错误概率

根据不同的译码方法和译码器, 一般译码错误概率分为不可检错误概率、译码失败概率和译码错误概率。

正确计算一个码的不可检错误概率, 在 ARQ 和 HEC 差错控制系统的性能分析中起着重要作用。

码字通过 BSC 传输时, 若由于干扰变成了另一码字, 则检错译码器就不能发现此类型的错误, 产生了不可检错误。所以码长为 n , 有 M 个码字, 最小距离为 d 的 (n, M, d) 二进制分组码的平均不可检错误概率

$$P_{ud} = \sum_{j=1}^M P_j \sum_{i=1}^n A_{j,i} p_e^i (1-p_e)^{n-i} \quad (3.7.12)$$

式中, p_e 是 BSC 的误码率, P_j 是发第 j 个码字的概率, $A_{j,i}$ 是与第 j 个码字的距离为 i 的码字数。设

$$A_j(x) = A_{j,0} + A_{j,1}x + A_{j,2}x^2 + \dots + A_{j,n}x^n \quad j = 1, 2, \dots, M$$

是 (n, M, d) 码中第 j 个码字的**距离分布多项式**。若对码中所有码字恒有

$$A_j(x) = A(x) = A_0 + A_1x + A_2x^2 + \dots + A_nx^n$$

则此码称为**不变距离分布码**或**同距离分布码**。

对线性码而言, 由于码的封闭性, 可知是同距离分布码, 且码的距离分布就等于码的重量分布。可知, 对 $[n, k, d]$ 二进制线性分组码的不可检错误概率, 由式(3.7.12)可得为

$$P_{ud} = \sum_{j=1}^{2^k} P_j \sum_{i=1}^n A_i p_e^i (1 - p_e)^{n-i} \quad (3.7.13)$$

若码字等概发送，则上式成为

$$P_{ud} = \sum_{i=1}^n A_i p_e^i (1 - p_e)^{n-i} \quad (3.7.14)$$

式中， A_i 是 $[n, k, d]$ 码的重量为 i 的码字数，由于码的最小距离等于 d ，所以 $A_1 = A_2 = \dots = A_{d-1} = 0$ 。

根据 MacWilliams 恒等式，可以由对偶码的重量分布计算 $[n, k, d]$ 码的不可检错误概率。为此，首先把式(3.7.14)写成

$$P_{ud} = \sum_{i=1}^n A_i p_e^i (1 - p_e)^{n-i} = (1 - p_e)^n \sum_{i=1}^n A_i \left(\frac{p_e}{1 - p_e} \right)^i \quad (3.7.15)$$

任何线性分组码的 $A_0 = 1$ ，应用式(3.7.1)，并令 $x = p_e / (1 - p_e)$ ，则

$$\begin{aligned} A(p_e / (1 - p_e)) - 1 &= A_1(p_e / (1 - p_e)) + \dots + A_n(p_e / (1 - p_e))^n \\ &= \sum_{i=1}^n A_i (p_e / (1 - p_e))^i \end{aligned}$$

把它代入式(3.7.15)得

$$P_{ud} = (1 - p_e)^n [A(p_e / (1 - p_e)) - 1]$$

把马克威伦恒等式(3.7.2)代入上式，可得

$$\begin{aligned} P_{ud} &= (1 - p_e)^n \left[2^{-(n-k)} \left(1 + \frac{p_e}{1 - p_e} \right)^n B \left(\left(1 - \frac{p_e}{1 - p_e} \right) / \left(1 + \frac{p_e}{1 - p_e} \right) \right) - 1 \right] \\ &= 2^{-(n-k)} B(1 - 2p_e) - (1 - p_e)^n \end{aligned} \quad (3.7.16)$$

式中

$$B(1 - 2p_e) = \sum_{i=0}^n B_i (1 - 2p_e)^i \quad (3.7.17)$$

可知，利用式(3.7.16)可由对偶码的重量估值算子 $B(x)$ ，计算 $[n, k, d]$ 码的不可检错误概率。

由于对很多码的重量分布并不知道，特别当 n, k 或 $n-k$ 较大时，计算码的重量分布非常困难，因此要正确计算 P_{ud} 是很难的。但是我们能够较容易地计算 $[n, k]$ 线性码集合中的平均不可检错误概率，以此作为估计码的不可检错误概率的上限。

定理3.7.2^[4] 二进制 $[n, k]$ 线性分组码集合中，码的平均不可检错误概率

$$\bar{P}_{ud} = 2^{-(n-k)} (1 - (1 - p_e)^k) \quad (3.7.18)$$

式中， p_e 是 BSC 的误码率。

通常情况下， $1 - (1 - p_e)^k \leq 1$ ，所以

$$\bar{P}_{ud} \leq 2^{-(n-k)} \quad (3.7.19)$$

\bar{P}_{ud} 是在所有 $2^{k(n-k)} - 1$ 个 $[n, k]$ 分组码构成的码集合中的平均不可检错误的概率。因此，在 $0 \leq p_e \leq 1/2$ 时，必定在该集合中存在有不可检错误概率 $P_{ud} \leq 2^{-(n-k)}$ 的二进制码，称这类码为**最佳检错码**。

类似地，对于 q 进制 $[n, k]$ 线性码，有

$$\bar{P}_{ud} \leq q^{-(n-k)} \quad (3.7.20)$$

在 q 进制信道的误码率 $0 \leq p_e \leq (q-1)/q$ 范围内，满足该式的码也称为最佳检错码。

那么,如何判断一个 $[n, k]$ 码,在BSC的误码率满足 $0 \leq p_e \leq 1/2$ 范围内,其不可检错误概率是否服从 $P_{ud} \leq 2^{-(n-k)}$ 的上限呢?显然,如果我们能够证明任何码(线性或非线性码)的 P_{ud} 是 p_e 的单调增函数,即

$$\frac{dP_{ud}}{dp_e} \geq 0 \quad (3.7.21)$$

则该 $[n, k]$ 码的 P_{ud} 服从 $2^{-(n-k)}$ 的上限,对非线性码而言,则是最佳的。因为当 $p_e = 1/2$ 时,所有 2^n 个 n 重在接收端出现的机会均等,其概率是 2^{-n} ,在这 2^n 个中仅有 2^k 个是码字,其中之一是发的码字。因此,当 $p_e = 1/2$ 时, $[n, k]$ 码的不可检错误概率

$$P_{ud}(1/2) = (2^k - 1)2^{-n} = 2^{-(n-k)} - 2^{-n} < 2^{-(n-k)} \quad (3.7.22)$$

说明在最大误码率情况下, P_{ud} 不会超过 $2^{-(n-k)}$ 。可知 $2^{-(n-k)}$ 是不可检错误概率的上限。

由上讨论可知,如果我们能够证明在BSC的误码率 $0 \leq p_e \leq 1/2$ 范围内,式(3.7.21)满足,则该码(线性或非线性码)是最佳检错码。但是,除了汉明码、Golay、纠两个错误的本原BCH码等少数几类码是最佳检错码以外,如何寻找或确定最佳检错码,目前尚不完全清楚。有关最佳码的构造与性质,不可检错误概率的上、下限及近似计算请参阅[4]。

至于非线性码,如我国电传通信中所用的5中取3(3个1,2个0)码和国际电传通信中所用的7中取3的(7, 2, 3)码,以及8中取4的(8, 2, 4)码和(6, 2, 3)码已由作者证明是最佳检错码外^[17],还不知道是否存在有其它的最佳非线性检错码。

三、译码错误与译码失败概率

如果 $[n, k, d]$ 码的译码器,用来纠 t 个错误,同时检测 e 个错误,则称为teD译码器,此时要求 $d \geq t+e+1$, $e \geq t$ 。0eD是纯检错译码器,这时 $e \leq d-1$; t0D是纯纠错译码器,此时 $t \leq \lfloor (d-1)/2 \rfloor$ 。显然,teD译码器是一类限定距离译码器。

teD译码器正确译码的码字概率

$$P_{uc} = \sum_{i=0}^t \binom{n}{i} p_e^i (1-p_e)^{n-i} \quad (3.7.23)$$

这里及以后 p_e 均指BSC的误码率,且码字等概发送。

译码中,若teD译码器输出的码字与发送的码字不相同,则产生了译码错误。如果译码器不能译出码字,而仅指出收到的码字有错,则称为译码失败。下面先讨论译码错误概率。

由于线性码的封闭性,不失一般性我们认为发的是全0码字。当接收到的 n 重进入除全0码字以外的其它码字的译码区域内时,则产生了译码错误。这里所指的每个码字的译码区域,是在 n 维线性空间中以该码字为中心、以 t 为半径的球。显然,当收到的 n 重落入该球内时,则该 n 重译成处在球心的码字。

令 B_i 表示重量为 2^i ,落入除全0码字以外的全部(2^k-1 个)码字译码区域内的所有 n 重数目,因此译码错误概率

$$P_{ue} = \sum_{i=t+1}^n B_i p_e^i (1-p_e)^{n-i} \quad (3.7.24)$$

设 u 是 A 码的一个码字, $\varphi(i, j, s)$ 表示满足下列条件的 n 重 v 的数目:

$$w(u) = j \quad w(v) = i$$

$$d(u, v) = s \quad 0 \leq s \leq t$$

设 u 和 v 两个 n 重的组成如下：

$$\begin{aligned} & | \leftarrow j \rightarrow | \leftarrow n-j \rightarrow | \\ u: & (1 \cdots 11 \cdots 10 \cdots 00 \cdots 0) \\ & | \leftarrow x \rightarrow | \leftarrow s-x \rightarrow | \\ v: & (0 \cdots 01 \cdots 11 \cdots 10 \cdots 0) \\ & | \leftarrow i \rightarrow | \end{aligned}$$

由此可以看出， $x = (j-i+s)/2$ ，且要求 $0 \leq j \leq s$ ， $s-x \leq n-j$ 。由此可得：

$$\varphi(i, j, s) = \begin{cases} \binom{j}{x} \binom{n-j}{s-x} & x = (j-i+s)/2, \text{ 是整数, 且} \\ & \min(2n-i-j, i+j) \geq s \geq |j-i| \\ 0 & \text{其它} \end{cases}$$

所以

$$B_i = \sum_{j=0}^n \sum_{s=0}^t \varphi(i, j, s) A_j \quad (3.7.25)$$

式中， A_j 是线性码 A 的重量为 j 的码字数目。可知译码错误概率

$$P_{we} = \sum_{i=t+1}^n \sum_{j=0}^n \sum_{s=0}^t \varphi(i, j, s) A_j p_e^i (1 - p_e)^{n-i} \quad (3.7.26)$$

而译码失败概率

$$P_{wf} = 1 - P_{wc} - P_{we}$$

例如，[15, 11, 3]汉明码的对偶码[15, 4, 8]码，能纠正 $t=3$ 个错误，同时发现 $e=4$ 个错误。设误码率 $p_e=0.1$ 。则由式(3.7.23)可得正确译码概率

$$P_{wc} = \sum_{i=0}^3 \binom{15}{i} p_e^i (1 - p_e)^{15-i} = 0.94444$$

[15, 4, 8]码是一极长码，它的重量分布： $A_0=1$, $A_8=15$ 。由式(3.7.25)及 $\varphi(i, j, s)$ 表示式可得 B_i 为： $B_0=1$, $B_1=15$, $B_2=105$, $B_3=445$, $B_4=0$, $B_5=840$, $B_6=420$, $B_7=3060$, $B_8=855$, $B_9=2625$, $B_{10}=315$, $B_{11}=525$, $B_{12}=B_{13}=B_{14}=B_{15}=0$ 。把这些 B_i 代入式(3.7.24)得译码错误概率

$$P_{we} = 0.00322$$

由此得译码失败概率

$$P_{wf} = 1 - P_{wc} - P_{we} = 0.05234$$

四、误码率计算

误码率是衡量一个数字通信系统质量的重要指标。下面讨论如何计算 teD 译码器的误码率。设发送的仍为全0码字，接收的 n 重落入重量为 j 的码字的译码区域时，译码器输出重量为 j 的码字，因此产生了 j 个码元错误。产生这种事件的概率为

$$P_{ej} = A_j \sum_{i=t+1}^n \sum_{s=0}^i \varphi(i, j, s) p_e^i (1 - p_e)^{n-i} \quad (3.7.27)$$

设 j 个错误在 n 长码字内均匀分布，则译码后的误码率

$$p_b = \frac{1}{n} \sum_{j=d}^n j P_{ej} \quad (3.7.28)$$

而译码失败引起的误码率

$$p_b' = \frac{1}{n} \sum_{i=t+1}^n \left(\binom{n}{i} - B_i \right) i p_e^i (1 - p_e)^{n-i} \quad (3.7.29)$$

$p_b + p_b'$ 称为 etD 译码器的输出误码率。式中 B_i 由式(3.7.25)决定。

仍以[15, 4, 8]码为例。该码的 $A_0=1$, $A_8=15$ 。由式(3.7.27)得 $P_{e8}=0.00322$; 由式(3.7.28)得 $p_b=0.00172$; 由式(3.7.29)得 $p_b'=0.01395$ 。所以, 译码器输出误码率 $p_b + p_b'=0.01567$, 可知主要由译码失败所引起的误码率决定。

§ 3.8 线性码的纠错能力

研究码的纠错能力, 也就是分析码的 n 、 k 、 d 之间的关系, 不仅能从理论上指出哪些码可以构造出, 哪些码不能构造出, 而且也为工程实验提供了对各种码性能估计的理论依据。因此, 研究码的纠错能力始终是编码理论中一个重要的课题。在香农的信道编码定理中指出, 仅当分组码的 n 趋向于 ∞ 时, 译码错误概率才能任意地接近于零。因此, 研究 $n \rightarrow \infty$ 时码的渐近性能, 具有特别重大的理论意义。

本节将介绍某些最基本的分组码距离的上、下限, 其结果比较粗糙和简单。目前已有更为精确的结果, 关于这方面的详细情况, 读者可参阅有关文献^[2,5]。

一、普洛特金(Plotkin)限(P 限)

定理3.8.1 GF(q)上(n, M, d)分组码的最小距离 d 为

$$d \leq \frac{nM(q-1)}{(M-1)q} \quad (3.8.1)$$

证明 GF(q)上(n, M, d)码的码字($c_{n-1}, c_{n-2}, \dots, c_1, c_0$)中, $c_i \in GF(q)$ 。设 GF(q)中的 q 个元素是 $\{\alpha_1, \alpha_2, \dots, \alpha_q\}$ 。 M 个码字中, 第一位(c_{n-1})取值为 α_i 的码字假设有 M_i 个, 则 $M_1 + M_2 + \dots + M_q = M$ 。第一位取值不是 α_i 的共有 $M - M_i$ 个码字。因此, 第一位取值为 α_i 的码字与其它($M - M_i$)个码字, 由于这一位不同而带来的总距离是 $M_i(M - M_i)$ 。 c_{n-1} 这一位可以有 q 种不同取值, 因此由于 c_{n-1} 不同所带来的总距离

$$\begin{aligned} s_1 &= M_1(M - M_1) + M_2(M - M_2) + \dots + M_q(M - M_q) \\ &= M^2 - (M_1^2 + M_2^2 + \dots + M_q^2) \end{aligned}$$

为了使 s_1 最大, 就必须要求 M 最大、 $\sum_{i=1}^q M_i^2$ 最小, 仅当 $M_1 = M_2 = \dots = M_q = M/q$ 时才能满足此要求。因而 $s_1 \leq M^2(q-1)/q$, 该式是仅考虑码中第一位不同的所有码对给出的总距离。由于每个码字有 n 位, 因此码中所有不同码对所给出的总距离是 $nM^2(q-1)/q$ 。 (n, M, d) 码中共有 $M(M-1)$ 对不同的码字对, 因此不同码字对之间的平均距离

$$d_{av} \leq \frac{nM^2(q-1)}{M(M-1)q} = \frac{nM(q-1)}{(M-1)q}$$

码的最小距离 d 不可能大于平均距离 d_{av} , 因此 $d \leq nM(q-1)/[(M-1)q]$ 。 ■

若为 q 进制线性分组码, 则码字数 $M = q^k$, 因而由式(3.8.1)立刻可得

$$d \leq nq^{k-1}(q-1)/(q^k - 1) \quad (3.8.2)$$

由此定理还可推出给定 n 、 d 条件下的 M 的上限，以及给定 k 和 d 条件下码长 n 的最小值。

二、汉明限(球包限、H 限)

定理3.8.2 长为 n 纠 t 个错误的 q 进制分组码的码字数 M 为

$$M \leq \frac{q^n}{\sum_{i=0}^t \binom{n}{i} (q-1)^i} \quad (3.8.3)$$

证明 在 $GF(q)$ 上的 n 维线性空间中，以码字 C_i 为中心 t 为半径作球，当接收到的 n 重落入该球内时，就译成位于该球中心的码字 C_i ，因此，这个球是 C_i 码字的译码区。该球内共含有 $s_t = \sum_{i=0}^t \binom{n}{i} (q-1)^i$ 个 n 重。由于各个码字的译码区不能相交，每个球也不能相交，因此所有 M 个球内含有的 n 重数目是 Ms_t 。 $GF(q)$ 上 n 维线性空间中共有 q^n 个 n 重，因此 $Ms_t \leq q^n$ ，由此可立即得到式(3.8.3)。 ■

由式(3.8.3)可立即得到 q 进制 $[n, k, 2t+1]$ 线性分组码校验位数目的下限为

$$n - k \geq \log_q \left(\sum_{i=0}^t \binom{n}{i} (q-1)^i \right) \quad (3.8.4)$$

在二进制情况下，上式可简化为

$$n - k \geq \log_2 \left(\sum_{i=0}^t \binom{n}{i} \right) \quad (3.8.5)$$

由上可知， s_t 是重量 $\leq t$ 的所有错误图样数目之和。

如果 $(n, M, 2t+1)$ 分组码能使式(3.8.3)的等号成立，即 M 个球能把 q^n 个 n 重全部划分完，则此码的伴随式与 $\leq t$ 个错误的图样完全一一对应，校验元利用得充分，达到了最佳情况，也就是完备码的情况。这也相当于在标准阵译码表中，该码能将重量 $\leq t$ 的所有错误图样作为陪集首，而没有重量 $> t$ 的错误图样作为陪集首，也就是码的球半径等于覆盖半径的情况。汉明码、[23, 12, 7] 戈莱(Golay)码以及三进制的[11, 6]码是目前已知的非凡完备码，奇数码长的二进制重复码是平凡的完备码。除此以外，已证明在 $GF(q)$ 上不再存在有其它任何非凡的线性完备码^[2]。

如果某一 $(n, M, 2t+1)$ 码除了能把重量 $\leq t$ 的所有错误图样都有伴随式与之对应外，还有部分 $t+1$ 的错误也能纠正，这相当于准完备码情况。对线性码来说就是在标准阵中，除了所有重量 $\leq t$ 的错误图样作为陪集首外，还有部分重量为 $t+1$ 的错误图样作为陪集首。表1—2、表3—3所列出的两个[6, 3]码都是准完备码，增余删信汉明码也是准完备码的一个例子。

Plotkin 限和汉明限都是必要条件，也就是说任何线性或非线性码都是必须满足的，否则码就构造不出。下面介绍一下构造码的充分条件。

三、沃尔沙莫夫—吉尔伯特(V—G)限

定理3.8.3 若码的校验元数目 $n-k$ 满足

$$q^{n-k} - 1 > \binom{n-1}{1} (q-1) + \binom{n-1}{2} (q-1)^2 + \cdots + \binom{n-1}{d-2} (q-1)^{d-2} \quad (3.8.6)$$

的最小整数，则一定可以构造出一个长为 n 、最小距离为 d 的 $[n, k]$ 线性分组码。

证明 要构造有最小距离为 d 的分组码，由定理3.3.1可知，要求 \mathbf{H} 矩阵的任意 $d-1$ 列线性无关。 \mathbf{H} 矩阵的列都是从 $q^{n-k}-1$ 列选取的 $n-k$ 重。在 \mathbf{H} 矩阵的 $n-1$ 列中，要求每一列都不相同，且又不是另一列的线性组合。因此要求 $q^{n-k}-1$ 列中能提供 $\binom{n-1}{1} (q-1)$ 列。同理可知：

$$(n-1) \text{列中任二列的线性组合需 } \binom{n-1}{2} (q-1)^2 \text{列；}$$

$$(n-1) \text{列中任三列的线性组合需 } \binom{n-1}{3} (q-1)^3 \text{列；}$$

⋮

$$(n-1) \text{列中任 } d-2 \text{列的线性组合需 } \binom{n-1}{d-2} (q-1)^{d-2} \text{列。}$$

若除了上述这些列之外， $q^{n-k}-1$ 列中还能提供一列作为 \mathbf{H} 矩阵的第 n 列，则此列与前 $n-1$ 列的所有 $d-2$ 列线性无关，因而加上此列后能保证 \mathbf{H} 矩阵中任意 $d-1$ 列线性无关。所以要求

$$q^{n-k}-1 > \binom{n-1}{1} (q-1) + \binom{n-1}{2} (q-1)^2 + \cdots + \binom{n-1}{d-2} (q-1)^{d-2}$$

或

$$q^{n-k} > \sum_{i=0}^{d-2} \binom{n-1}{i} (q-1)^i$$

■

若为二进制码，则上式可简化为

$$n - k > \log_2 \sum_{i=0}^{d-2} \binom{n-1}{i} \quad (3.8.7)$$

与式(3.8.5)比较可知，V-G 限所要求的校验位数比汉明限要高。

四、讨论

Plotkin 限和汉明限都是构成码的必要条件，任何码都必须满足此必要条件，越接近它就越有效，等于它时就达到最佳。而 V-G 限是充分条件，并限定于线性码，满足这一条件必存在一个最小距离为 d 的 $[n, k]$ 线性码。但是，当 $n \rightarrow \infty$ 时，满足 V-G 限的线性码目前仅找到两类：某些代数几何码（包括 Goppa 码）和 Justesen 码。有些码类，如 $[2m, m]$ 二进制准循环码，码字重量为 $4m$ 的线性二进制自对偶码，已证明能接近 V-G 限，但遗憾的是直到目前还未找到具体的构造方法。

当 $n \rightarrow \infty$ 时，比较这3个限所表示的 n, R, d 之间的关系（只限于二进制码）：

(1) 当 $n \rightarrow \infty$ 时由 P 限可推出

$$k/n \leqslant 1 - 2d/n \quad (3.8.8)$$

(2) 由汉明限可得到

$$k/n \leqslant 1 - H_2(d/2n) \quad (3.8.9)$$

(3) 由 V-G 限可导出

$$k/n \geqslant 1 - H_2(d/n) \quad (3.8.10)$$

式中

$$H_2(x) = -x \log_2 x - (1-x) \log_2 (1-x)$$

由图 3-3 可以看出, 当 n, d 给定时, P 限和汉明限给出了传信率 R 的上限, 而 V-G 限提供了 R 的下限。当 n, k 给定时, P 限和汉明限给出了最小距离 d 的上限, 而 V-G 限给出了最小距离的下限。在相同条件下, 最小距离越接近于上限的码越好。

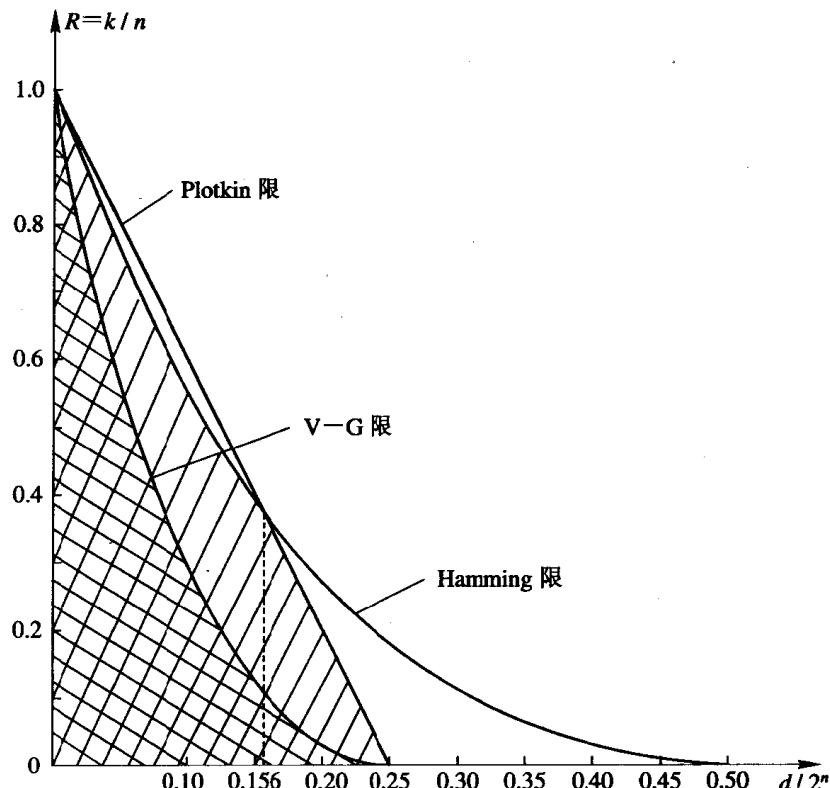


图 3-3 二进制下三个码限比较图

从图可以看到 P 限和汉明限有一个交叉点在 $R=0.4, d/2n=0.156$ 附近。当 $d/2n < 0.156$ 时(高码率, 低纠错力的码)应用汉明限较精确, 而当 $d/2n \geq 0.156$ 时(低码率, 强纠错能力的码)则应用 P 限较精确。图中的所有斜线区是可能实现部分, 而双重斜线部分是必能实现的部分。在相同条件下, 越接近斜线区上边缘的码越好。

某些高码率的里德-缪勒尔(Reed-Muller, RM)码和汉明码, 以及某些 BCH 码与汉明限符合, 故为最佳码。而低码率的 RM 码及 BCH 码等, 与 P 限符合, 也为最佳码。

香农的编码定理指出存在有 $n \rightarrow \infty$, 码率接近信道容量, 译码误码率接近 0 的分组码(称这种码为 Shannon 码或渐近好码), 而 V-G 限也保证存在有这种性能的码, 因此达到或超过 V-G 限的码是渐进好码。但上述的这两类码当 $n \rightarrow \infty$ 时, 高码率码的 $R \rightarrow 1$, 而其纠错能力 $d/2n \rightarrow 0$; 而低码率码在 $n \rightarrow \infty$ 时, $d/2n \rightarrow 1/4$, 但 $R \rightarrow 0$ 。此外, 对于中等速率的码来说, 如某些 BCH 码和中码率 RM 码, 在保证码率 k/n 不为零条件下, 当 $n \rightarrow \infty$ 时, $d/2n \rightarrow 0$, 因此都不符合 Shannon 码的要求。直到 1970 年 Goppa 才找到了一类线性码——

Goppa 码，这类码中的一个子类能达到 $n \rightarrow \infty$ 时性能接近 V-G 限。但当 $n \rightarrow \infty$ 时，如何具体构造出这类码仍很困难，而且达到 V-G 限的译码方法至今仍没有解决。1972 年 Justesen 构造的 Justesen 码也能做到 $n \rightarrow \infty$ ， R 一定时， $d/2n > 0$ ，但遗憾的是当码率 $R < 0.3$ 时离 V-G 限还有相当距离。

80 年代初 Goppa 把分组码看成是射影平面上的曲线，从而把代数几何引入了分组码的构造。1982 年，斯法斯曼(Tsfasman)等人利用模(Modular)曲线构造了一类代数几何码，当 $q \geq 49$ 时，其性能超过了 V-G 限。这具有重大的理论意义，说明从理论上讲，利用代数几何方法可以构造出 Shannon 码。但是，正如同 Goppa 码所遇到的困难一样，当 $n \rightarrow \infty$ 时，如何构造这类码以及如何译码，都没有完全解决。这也说明如何具体地构造 Shannon 码仍是一个没有解决的问题，也是当前编码理论研究的热门课题之一。

* § 3.9 不等保护能力线性分组码

前面所介绍的线性分组码，都是等保护能力的，也就是对码字中每个码元的保护能力均相等。例如，[7, 4, 3] 汉明码，它能纠一个随机错误，不论这一个错误产生在被传码字中的哪一位，译码时均能纠正，可知每个码元平均的抗干扰能力为 $1/7$ 。也就是码对每位码元或信息元均一视同仁的按照 $1/7$ 的能力保护。对一般的 $[n, k, d]$ 分组码来说，每位码元平均的保护能力为 t/n 或 d/n 。

然而，在很多实际场合中，某些信息位的重要性比其余的要高得多。例如，在银行数据传输中，前面的收、支符号及高值位的数据，如亿、万、千等，比低值位的数据角、分等重要得多。又例如，在多用户数据传输中，某些用户的数据比其余的重要得多。因此，就提出了一个要求：对不同的数据(信息元)，要按照其重要程度，分别予以不同的抗干扰保护。能完成这种不等保护能力的纠错码，称为不等保护能力码，简称 UEP 码。若线性码具有不等保护能力，则称为 LUEP 码。

一、UEP 码的基本概念

下面先通过一具体例子，说明 UEP 码的基本概念。若要传输的两个信息元是 m_1, m_2 ，现要求编成长为 4 的码。下表给出了 [4, 2] 码的两种不同编码方法得到的 C_1 和 C_2 码。

| | C_1 | | C_2 | |
|-----------|-----------|-----------|-----------|-----------|
| $m_1 = 0$ | $m_2 = 0$ | $m_2 = 1$ | $m_2 = 0$ | $m_2 = 1$ |
| | 0000 | 0111 | 0000 | 0111 |
| $m_1 = 1$ | 1100 | 1011 | 1011 | 1100 |

可知 C_1 和 C_2 码的

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

但 C_2 码是系统码，而 C_1 码是非系统码。 C_1 与 C_2 码的最小距离均为 2，由定理 1.3.1 知，它们均不能纠正单个错误。但是对 C_1 码来说，却可以纠正 m_2 信息位的错误。

设发送的码字是(0000)，接收的是(0100)，按最小汉明距离译码方法，译码器译成(0000)或(1100)。但无论译成谁，对应的信息位 m_2 均是0，也就是该码能纠正信息位 m_2 的一位错误，但不能纠正 m_1 中的错误。因此， C_1 码是一个不等纠错能力保护码。由此看出，对不等纠错能力保护码来说，即使整个码组可能译错了，但却能保证受高等级保护能力的信息位是正确的。如上例中 C_1 码的 d 虽然只有2，却能纠正信息位 m_2 的错误，但不能纠正 m_1 的错误。

对一个码字来说，我们仅关心其中信息位的保护能力，因此对每一信息位，定义一个保护能力大小的度量。如同用码的最小距离定义码的抗干扰能力一样，我们也可对每一位信息位定义一个最小距离，这 k 个最小距离所组成的矢量，称为码的分离矢量。

定义3.9.1 称 k 重矢量 $s=(s_1, s_2, \dots, s_k)$ 为 $GF(q)$ ($q=p^m$, p 为素数)上 $[n, k]$ 线性系统分组码 C 的分离矢量，其中

$$s_i = \min\{w(MG); M \in GF(q)^k, m_i \neq 0, i = 1, 2, \dots, k\} \quad (3.9.1)$$

$M=(m_1, m_2, \dots, m_k)$ 是信息组， G 是 C 码的生成矩阵， $GF(q)^k$ 是 $GF(q)$ 上的 k 维线性空间。

式(3.9.1)说明对任何 $\alpha, \beta \in GF(q)$ ，当 $\alpha \neq \beta$ 时，集合 $\{MG; M \in GF(q)^k, m_i = \alpha\}$ 和 $\{MG; M \in GF(q)^k, m_i = \beta\}$ 的最小距离为 s_i , $i=1, 2, \dots, k$ 。

由定义3.9.1可知，码的最小距离 $d=\min\{s_i, i=1, 2, \dots, k\}$ ，如果码的分离矢量 s 中的 k 个 s_i 分量不完全相同，则该码称为线性不等保护能力码(LUEP)，否则就是一般的等保护能力码。

由定理1.3.1知，若 $[n, k]$ 码的最小距离为 d ，则码能纠正 $t \leq \lfloor(d-1)/2\rfloor$ 个错误。也就是，当错误图样的重量不超过 $\lfloor(d-1)/2\rfloor$ 时，每位信息位均能正确译码，受到同等保护。显然，若分离矢量中第 i 位分量为 s_i ，则当错误图样的重量不超过 $t_i = \lfloor(s_i-1)/2\rfloor$ ，不论其它信息位如何，第 i 位信息位必能正确译码。称 t_i 为第 i 位信息位 m_i 的保护能力或保护等级。由此得到以下引理。

引理3.9.1 $[n, k]$ 线性码的第 i 位信息位有保护能力为 t_i 的充要条件是：第 i 位信息位为非零的码字之间的最小距离 $d_i \geq 2t_i + 1$ ；或分离矢量的第 i 个分量 $s_i \geq 2t_i + 1$ ，即第 i 位信息元 $m_i \neq 0$ 的码字的重量 $w(C_i) \geq 2t_i + 1$ 。

二、LUEP 码的生成矩阵和校验矩阵

下面定理给出了 LUEP 码的码字集合所必须满足的条件。

定理3.9.1 为了使 $[n, k]$ 码 V 的不少于 k^* 个信息元有保护能力为 δ ，则其充要条件是码字集合 $\{C_i\} = \{C_i \in V, w(C_i) \leq 2\delta\}$ 所组成的线性子空间的维数不大于 $k - k^*$ 。

证明

必要性。设 $M=(m_1, m_2, \dots, m_{k^*}, \dots, m_k)$ 是码字的任一信息组。不失一般性，假设首 k^* 个信息元有保护能力为 δ ，其余的信息元有保护能力为 t 。显然，首 k^* 个信息元均为0的所有码字集合，组成了码 V 中的一个 $(k-k^*)$ 维子空间 V' 。考虑不含在 V' 中的任一码字 \tilde{C} ，它相应于信息组中至少有一个 $m_i \neq 0$, $i=1, 2, \dots, k^*$ 。由引理3.9.1, $w(\tilde{C}) \geq 2\delta + 1$ ，因此 V 中的所有重量 $w(C) \leq 2\delta$ 的所有码字 C 均应在码字集合 V' 中，所以 V' 子空间的维数不大于 $k - k^*$ 。

充分性。令集合 $\{C_M\} = \{C_i \in V, w(C_i) \leq 2\delta\}$ 的维数等于 $l \leq k - k^*$, $\{g_1, g_2, \dots, g_l\}$ 是集合 $\{C_M\}$ 中的最大线性无关矢量组; 用该矢量组张成的矢量空间为 V^* , 它是码 V 中的一个 l 维子空间; 在 V 中另外挑选与 $\{g_1, g_2, \dots, g_l\}$ 线性无关的 $k-l$ 个线性无关矢量 g_{l+1}, \dots, g_k , 则 $\{g_1, \dots, g_l, g_{l+1}, \dots, g_k\}$ 张成了码空间 V , 它包含了 V^* 与 $V' = V - V^*$ 。

对至少有一个信息元 $m_{l+j} \neq 0$ ($j=1, \dots, k-l$) 的所有信息组 $M = (m_1, \dots, m_k)$, 作 $C_a = MG$ 和 $G = [g_1, \dots, g_k]^T$, 则重量 $w(C_a) \geq 2\delta + 1$ 的所有码字 $C_a \in V^*$, 由引理3.9.1知, 这意味着第 $(l+j)$ ($j=1, \dots, k-l$) 个信息元有保护能力为 δ 。 ■

推论3.9.1 $[n, k]$ 码 V 的 k^* 个信息元有保护能力为 δ 的充要条件是: 码 $V = V^* \oplus V'$, V' 码的维数为 k^* , 而所有重量不大于 2δ 的码字在 V^* 中。

由此推论可得到有 k^* 个信息元的保护能力至少为 δ 的 LUEP 码的生成矩阵

$$\begin{aligned} G &= \begin{bmatrix} G \\ G^* \end{bmatrix} \quad (3.9.2) \\ G' &= \begin{bmatrix} g_1 \\ \vdots \\ g_l \end{bmatrix} = \begin{bmatrix} g_{11} & \cdots & g_{1n} \\ \vdots & & \vdots \\ g_{l1} & \cdots & g_{ln} \end{bmatrix} \\ G^* &= \begin{bmatrix} g_{l+1} \\ \vdots \\ g_k \end{bmatrix} = \begin{bmatrix} g_{l+1, 1} & \cdots & g_{l+1, n} \\ \vdots & & \vdots \\ g_{k, 1} & \cdots & g_{k, n} \end{bmatrix} \end{aligned}$$

对式(3.9.2)的 G 矩阵进行初等行变换可得到 LUEP 码的典型 G 矩阵

$$G = \begin{bmatrix} 1 & \mathbf{0} & g_{1, l+1} & \ddots & g_{1, k} & g_{1, k+1} & \cdots & g_{1, n} \\ \ddots & g_{2, l+1} & \cdots & g_{2, k} & g_{2, k+1} & \cdots & g_{2, n} \\ 1 & \vdots & & \vdots & \vdots & & \vdots \\ g_{l, l+1} & \cdots & g_{l, k} & g_{l, k+1} & \cdots & g_{l, n} \\ \mathbf{0} & 1 & \mathbf{0} & & g_{l+1, k+1} & \cdots & g_{l+1, n} \\ & & & \ddots & g_{l+2, k+1} & \cdots & g_{l+2, n} \\ & & & & \vdots & & \vdots \\ 1 & & & & g_{k, k+1} & \cdots & g_{k, n} \end{bmatrix} \quad (3.9.3)$$

由式(3.9.3)所生成的码并不是系统码, 必须对该矩阵再进行初等行变换才能得到系统码形式的生成矩阵。

如 V' 与 V^* 码均是二进制 $[10, 2]$ 码, 它们的生成矩阵分别是:

$$G' = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \end{bmatrix} \quad G^* = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

则二进制 $[10, 4]$ LUEP 码的典型形式的生成矩阵是

$$G = \begin{bmatrix} G' \\ G^* \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

对它进行初等行变换可得到如下的系统码形式的生成矩阵和校验矩阵:

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \quad H = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

可以检验，该码的分离矢量 $s=(5, 5, 4, 4)$ ，是一个 $[10, 4, 4]$ LUEP 码，前两个信息元具有纠 2 个错误的保护能力，后两个信息元只具有纠 1 个错误的保护能力。

可以把定理3.9.1的结果推广到具有多级保护能力的码。

定理3.9.2 为了对 $[n, k]$ 线性码 V 的不少于 l_i 个信息元提供 t_i 保护能力， $i=1, 2, \dots, u$ 。其充要条件是：对任何码字集合 $\{C_i\}=\{C_i \in V; w(C_i) \leq 2t_i\}$ 所组成的子空间具有维数不大于 $k-l_i$ 。这里， $t_1 \leq t_2 \leq \dots \leq t_u$ ， $k=\sum_{j=1}^u k_j$ ， $l_i=\sum_{j=i}^u k_j$ ，显然 $l_i=k$ 。

由该定理可得到 $[n, k]$ LUEP 码典型生成矩阵的一般表示式

$$G = \begin{bmatrix} I_{k_u} & G_u \\ 0_{k_u} & I_{k_{u-1}} & G_{u-1} \\ \vdots & \vdots \\ 0_{k_u} & 0_{k_{u-1}} & \cdots & 0_{k_2} & I_{k_1} & G_1 \end{bmatrix} \quad (3.9.4)$$

式中， I_{k_i} 与 0_{k_i} 分别是一个 $k_i \times k_i$ 阶单位方阵和全 0 矩阵， G_u 是一个 $k_u \times (n-k_u)$ 阶矩阵。

下面讨论 LUEP 码的校验矩阵 H 所必须满足的条件。

定理3.9.3 $[n, k]$ 线性分组码 V 的第 i 个码元有保护能力为 t_i 的充要条件是， V 的校验矩阵 H 的第 i 列与任意的不少于 $2t_i$ 列线性相关，或与任意的 $2t_i-1$ 或更少列线性无关。

该定理与定理3.3.1相似，其证明方法也相同。

如上面例中的 $[10, 4]$ LUEP 码，它的 H 矩阵中前两列中的任一列与任意其它三列的组合线性无关，而矩阵3、4列中任一列与任意其它两列线性无关。

三、LUEP 码的构造

根据定理3.9.1和定理3.9.2或定理3.9.3，利用已知分组码通过上节介绍的各种组合方法，就可构造各种 LUEP 码。

1. 码的直和构造 定理3.9.2和定理3.9.3告诉我们，利用多个不同纠错能力分组码，构造 LUEP 码所必须满足的条件。下面定理以更明确的形式给出用 m 个不同码，利用直和方法构造 LUEP 码所必须满足的条件。

定理3.9.4^[13] 令 C_i 是 $[n, k_i]$ 二进制线性分组码， $i=1, 2, \dots, m$ 。令 $C=C_1 \oplus C_2 \oplus \dots \oplus C_m$ 是 C_1, C_2, \dots, C_m 的直和，若以下的距离条件满足：

- (1) C_m 中的任一非零码字的重量至少是 d_m ；
- (2) 在 $C-C_{i+1} \oplus C_{i+2} \oplus \dots \oplus C_m$ 中，任一码字的重量至少是 d_i ， $i=1, 2, \dots, m$ ，且 $d_1 > d_2 > \dots > d_m$ 。则 C 是一个 m 级 LUEP 码，有分离矢量 $s=(s_1, s_2, \dots, s_m)$ ，这里 $s_i \geq d_i$ ， $i=1, 2, \dots, m$ 。

该定理的证明类似于定理3.9.1^[10,13]。设 C_1, C_2, \dots, C_m 码的生成矩阵分别是 G_1, G_2, \dots, G_m , 则 C 码的生成矩阵

$$G = [G_1^T \ G_2^T \ \cdots \ G_m^T]^T \quad (3.9.5)$$

2. 码的链接构造 设 C_1, C_2 分别是 $\text{GF}(q)$ 上的 $[n_1, k_1, d_1]$ 和 $[n_2, k_2, d_2]$ 码, 它们的生成矩阵分别是 G_1 和 G_2 。若 C_1 码含有一个 $[n_1, \bar{k}_1, \bar{d}_1]$ 子码 \tilde{C}_1 , $\bar{k}_1 = k_1 - k_2$ (设 $k_1 > k_2$), $\bar{d} > d_1$, 且 $\bar{d} < d_1 + d_2$ 。把 C_1 和 C_2 码链接, 得到码 C 的生成矩阵

$$G = \begin{bmatrix} G_2 \\ & G_1 \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} G_2 & \hat{G}_1 \\ \mathbf{0} & \tilde{G}_1 \end{bmatrix} \quad (3.9.6)$$

式中, \tilde{G}_1 是子码 \tilde{C}_1 的生成矩阵, \hat{G}_1 是 G_1 中去掉 \tilde{G}_1 后得到的矩阵, $\mathbf{0}$ 是 $\bar{k}_1 \times n_2$ 阶全 0 矩阵。

定理 3.9.5 由式(3.9.6) G 矩阵生成了一个 $\text{GF}(q)$ 上的 $[n_1+n_2, k_1, \bar{d}_1]$ 码 C , 有 k_2 个信息元具有 $t_2 = \lfloor (d_1+d_2-1)/2 \rfloor$ 级保护能力。

证明 显然, 由链接码的结构性质可知, C 码的最小距离为 \bar{d}_1 。在 C 码中, 如果某一码字 C_i 是由 G_2 和 \hat{G}_1 的行连接而成, 则 $w(C_i) \geq d_2 + d_1$; 否则 $w(C_i) < d_1 + d_2 = \bar{d}_1$ 。在 G 矩阵中, 若取后面的 $k_2 - \bar{k}_1$ 行, 则由它们的线性组合产生一个 $[n_1+n_2, k_1-k_2]$ 子空间 \tilde{C} , 其生成矩阵 $\tilde{G} = [\mathbf{0} \ \tilde{G}_1]$, 它由 C 码中所有重量小于 $d_1 + d_2$ 的码字组成, 因此由定理3.9.1知, C 码的 k_2 个信息元, 具有 $t_2 = \lfloor (d_1+d_2-1)/2 \rfloor$ 级保护能力。 ■

例如, C_1 是二进制的 $[7, 7, 1]$ 码, 它含有 $[7, 4, 3]$ 子码 \tilde{C}_1 , C_2 是二进制的 $[7, 3, 4]$ 码, 按式(3.9.6)的 G 矩阵形式进行链接, 得到 $[14, 7, 3]$ 码 C , 其生成矩阵

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

可知该码的 3 个信息元具有纠两个错误的保护能力, 4 个信息元具有纠一个错误的保护能力, 其分离矢量 $s = (5, 5, 5, 3, 3, 3, 3)$ 。

一般而言, 式(3.9.6)的 G 矩阵是与式(3.9.3)不同的非典型形式, 为了得到等价的典型形式, 可先对 G_1 矩阵进行初等行变换, 成为

$$G_1 = \begin{bmatrix} \mathbf{0}P_1 \\ I_{k_2}\tilde{P}_1 \end{bmatrix}$$

$G_2 = [I_{k_2}P_2]$ 。由此得到 C 码的典型形式生成矩阵

$$G = \begin{bmatrix} I_{k_2}P_2 & \mathbf{0}P_1 \\ \mathbf{0} & I_{k_2}\tilde{P}_1 \end{bmatrix} \quad (3.9.7)$$

然而再对式(3.9.7)的 G 进行初等行变换就可得到系统码的生成矩阵。

除了上面介绍的两种方法外, 还有很多其它的构造方法, 如用多个码的直积和直和,

以及用循环码构造等等^[10-13]。

LUEP 码的译码可以用译码表译码，或用其它方法如大数逻辑译码和循环码的译码方法等。

四、有关码限

在给定码的维数(即信息元的数目)和分离矢量下，使码长最短或使校验元数目最少，从而使码具有最大的码率，是构造 LUEP 码的基本问题之一。

定义3.9.2 定义 n_{qs} 是分离矢量至少为 s ，信息元数目为 k 的 q 进制 LUEP 码的最短的码长。 n_{qs}^e 表示分离矢量恰好为 s ，信息元数目为 k 的 q 进制 LUEP 码的最佳码长。

用 $[n, k, s]$ 表示分离矢量为 s 的 LUEP 码。如果当 $t \geq s, t \neq s$ 时，不存在任何的 q 进制 $[n_{qs}, k, t]$ LUEP 码，则称可以构造出的 $[n_{qs}, k, t]$ 码是最佳码。显然

$$n_{qs} \leq n_{qs}^e \quad (3.9.8)$$

$$s \leq t \Rightarrow n_{qs}^e \leq n_{qt} \quad (3.9.9)$$

$$s \leq t \not\Rightarrow n_{qs}^e \leq n_{qt} \quad (3.9.10)$$

今后规定：若 $s_i \geq t_i, i=1, 2, \dots, k$ ，则分离矢量 $s = (s_1, s_2, \dots, s_k) \geq t = (t_1, t_2, \dots, t_k)$ 。

1. 码长上限

定理3.9.6 对任一 $q=p^m, k, v$ 和分离矢量 $s = (s_1, s_2, \dots, s_v)$ ，这里 $s_i = (s_{i1}, s_{i2}, \dots, s_{im})$ ，

且 $\sum_{i=1}^v i_m = k$ 。下式成立：

$$n_{qs}^e \leq \sum_{u=1}^v n_{qs_u}^e \quad (3.9.11)$$

对 n_{qs} 也有同样的不等式。

证明 对 $u=1, 2, \dots, v$ ，令 G_u 是码长为 $n_{qs_u}^e$ 分离矢量是 s_u 的 LUEP 码的生成矩阵，则由

$$G = \begin{bmatrix} G_1 & \mathbf{0} \\ & G_2 \\ \mathbf{0} & G_v \end{bmatrix} \quad (3.9.12)$$

生成的码有分离矢量 $s = (s_1, s_2, \dots, s_v)$ ，码长为 n_{qs}^e ，而该码显然是最长码长的 LUEP 码。 ■

推论3.9.2 对任何 $k, s = (s_1, s_2, \dots, s_k), q=p^m$ ，有

$$n_{qs}^e \leq \sum_{i=1}^k s_i \quad (3.9.13)$$

证明 应用定理3.9.6，令 $v=k$ ， $G_u = [1 \ 1 \ \dots \ 1]$ ，对所有 $u=1, 2, \dots, k, s_u=1$ ，即可得到。 ■

因此可知，对任何 s ，有可能构造出 k 维的有分离矢量为 s 的 q 进制 LUEP 码，当然所构造出的码不一定是最佳的，如用式(3.9.12)的 G 矩阵所生成的码，就不一定是最佳的。

2. 码长下限

定理3.9.7 对任何 $q=p^m, k$ 和 $s = (s_1, s_2, \dots, s_k), s' = (s_1-1, s_2-1, \dots, s_k-1)$ 有

$$n_{qs} \geq n_{qs'} + 1 \quad (3.9.14)$$

证明 在 $[n_{qs}, k, s]$ LUEP 码的生成矩阵 G 中, 删去一列, 就得到 $[n_{qs}-1, k, s']$ 的 LUEP 码。 ■

定理 3.9.8 对任何 $q=p^n$, k 和 $s=(s_1, s_2, \dots, s_k)$, 且 $s_1 \geq s_2 \geq \dots \geq s_k$, 满足以下不等式:

$$n_{qs} \geq \sum_{i=1}^k \lceil s_i/q^{i-1} \rceil \quad (3.9.15)$$

式中, $\lceil x \rceil$ 表示不小于 x 的最小整数。

该定理的证明比较长, 请参阅[12]。上面这些定理给出了当 k, s 给定时 LUEP 码的码长上、下限。

下面我们讨论当 k 和 s 给定时, 具有二级保护能力的 LUEP 码的校验元数目 r , 所必须满足的下限, 当然此限与上面讨论的码长的限是一致的。

定理 3.9.9 对任何 k 和 $s=(s_{k_1}, s_{k_2})$, $s_{k_1}=(2t_1+1, \dots, 2t_1+1)$ 和 $s_{k_2}=(2t_2+1, \dots, 2t_2+1)$ 分别是长为 k_1 和 k_2 的整数序列, $t_2 \geq t_1$, $k=k_1+k_2$, 则 $[n, k, s]$ 二进制 LUEP 码的校验元数目

$$r \geq \left\lceil \log_2 \left(1 + \sum_{i=1}^{t_1} \binom{n}{i} + \sum_{j=t_1+1}^{t_2} \sum_{i=0}^{t_1} \binom{n-k_2}{i} \binom{k_2}{j-i} \right) \right\rceil \quad (3.9.16)$$

证明 由前一节知, 为了使码能纠正 t 个错误, 则必须使 $\leq t$ 个错误的所有可能的错误图样, 均有不同的伴随式与之一一对应。式(3.9.16)不等式的右边第一项表示没有错误, 需一个全 0 伴随式与之对应; 第二项表示 $\leq t_1$ 个错误时, 所有可能的错误图样; 第三项表示在 $n-k_2$ 个码元中产生 i 个错误, 在 k_2 个信息元中产生 t_2-i 个错误的所有错误图样; 而所有这些错误图样之和应小于 2^r 。 ■

能使式(3.9.16)成立的 LUEP 码, 称为完备 LUEP 码, 完备码当然是最佳的。但到目前为止, 还没有找到一个完备 LUEP 码, 在 $GF(q)$ 上是否存在有完备 LUEP 码, 还是一个没有解决的问题。

* § 3.10 纠非对称、单向错误及 $t-EC/AUED$ 码

以前介绍的纠错码都是针对对称信道而设计的。也就是信道所产生的 1 错成 0 与 0 错成 1 的概率相等, 这类码也称为**纠对称(随机)错误码**。但是, 在许多实际信道中 0 错成 1 (**0 错误**) 与 1 错成 0 (**1 错误**) 的概率并不相等 (非对称信道), 或者仅仅产生一种类型的错误: 0 错误或 1 错误 (单向信道错误)。例如, 在光纤通信中, 一般而言仅发生 1 错误。而在最近发展起来的大规模与超大规模集成电路 (LSI/VLSI) 中, 由于芯片缺陷所引起的错误, 以及 ROM 和 RAM 等器件记忆单元中的缺陷所造成的错误, 虽然有少量的随机错误, 但绝大部分都是单向错误, 也就是在一个字节或一组中仅产生 1 错误或 0 错误。

因此, 必须针对这种**非对称信道或单向信道**设计一类纠错码。从理论上讲, 前几节及以后各章所讨论的适用于对称信道的纠错码, 都可用来纠正非对称错误或单向错误。但是与专门设计用来纠非对称或单向错误的码来说, 在同样的纠错能力下, 纠对称错误码的码率相对而言要低, 因此有必要专门设计纠正这些**非对称错误的码 (ASEC)** 及**纠单向错误的码 (AUED)**。

码(UEC)，或者纠正 t 个随机错误，同时检测所有单向错误的码(t -EC/AUED)。下面我们仅讨论二进制码的情况。

一、基本概念

首先，再次明确一下什么是对称错误、非对称错误和单向错误。

对称错误(随机错误)：在对称信道中(如 BSC 和 DMC)所引起的错误，称为对称错误。

非对称错误：在 Z 信道中所产生的错误称为非对称错误。也就是在该信道中仅产生 1 错误或 0 错误，并且产生什么类型的错误事前是已知的。若无特别说明，以后均指产生 1 错误的**非对称信道**。

单向错误：在接收的一个码字中既可能是 1 错误，也可能是 0 错误。但是，在任何一个接收的码字中，仅产生一种类型的错误(或者是 1 错误，或者是 0 错误)，即 0 错误或 1 错误不能同时在一个接收码字中并存。产生这种错误类型的信道简称**单向信道**。

下面介绍一下适用于这些信道纠错码的有关距离函数。

设 $\mathbf{u}=(u_{n-1}, \dots, u_1, u_0) \in V_n$

$\mathbf{v}=(v_{n-1}, \dots, v_1, v_0) \in V_n$

令 $N(\mathbf{u}, \mathbf{v})=\{i|u_i=1 \wedge (\text{同时})v_i=0\}$

也即 $N(\mathbf{u}, \mathbf{v})$ 是对应位分量中， u_i 为 1， v_i 为 0 的数目。

$N(\mathbf{u}, \mathbf{v})=0$ ，则称为矢量 \mathbf{u} 被矢量 \mathbf{v} 所包含，或 \mathbf{v} 包含 \mathbf{u} ，用 $\mathbf{v} \geq \mathbf{u}$ 表示之。

定义3.10.1 设所传输的矢量 $\mathbf{u} \in V_n$ ，而接收的矢量为 $\mathbf{f} \in V_n$ ，并有

(a) 若 $\{N(\mathbf{u}, \mathbf{f})+N(\mathbf{f}, \mathbf{u})=t\}$ ，则称 \mathbf{u} 有 t 个随机错误；

(b) 若 $\{N(\mathbf{f}, \mathbf{u})=0 \wedge N(\mathbf{u}, \mathbf{f})=t\}$ ，则称 \mathbf{u} 有 t 个非对称错误(1错误)；

(c) 若 $\{N(\mathbf{u}, \mathbf{f})=0 \wedge N(\mathbf{f}, \mathbf{u})=t \vee (\text{或者}) (N(\mathbf{u}, \mathbf{f})=t \wedge N(\mathbf{f}, \mathbf{u})=0)\}$ ，则称 \mathbf{u} 有 t 个单向错误。例如：令 $n=10$ ，且

$$\mathbf{u}=1111100000$$

$$\mathbf{f}_1=1100010001$$

$$\mathbf{f}_2=1110000000$$

$$\mathbf{f}_3=1111100001$$

当在对称信道中传输 \mathbf{u} 时，接收端可能收到有 $\mathbf{f}_1(t=5)$ ， $\mathbf{f}_2(t=2)$ ， $\mathbf{f}_3(t=1)$ 。当 \mathbf{u} 在非对称信道传输时，则仅可能收到 \mathbf{f}_2 ，而决不可能收到 \mathbf{f}_1 或 \mathbf{f}_3 。若在单向信道传输时，则仅可能收到 \mathbf{f}_2 或 \mathbf{f}_3 ，而决不可能收到 \mathbf{f}_1 。

为了研究分组码 C 的纠错能力，必须定义针对对称、非对称与单向信道纠错码的距离度量。显然，§ 1.3 定义的汉明距离

$$d(\mathbf{u}, \mathbf{v}) = N(\mathbf{u}, \mathbf{v}) + N(\mathbf{v}, \mathbf{u}) \quad (3.10.1)$$

它适用于对称信道的纠错码，也就是以前及今后我们所主要讨论的纠错码。

定义3.10.2 令 $\mathbf{u} \in V_n$, $\mathbf{v} \in V_n$ ，则

(1) $d_a(\mathbf{u}, \mathbf{v})=2\max\{N(\mathbf{u}, \mathbf{v}), N(\mathbf{v}, \mathbf{u})\}$

(2) $d_u(\mathbf{u}, \mathbf{v})=\begin{cases} d_a(\mathbf{u}, \mathbf{v}) & \text{若 } N(\mathbf{u}, \mathbf{v})=0 \vee N(\mathbf{v}, \mathbf{u})=0 \\ d_s(\mathbf{u}, \mathbf{v}) & \text{若 } N(\mathbf{u}, \mathbf{v})>0 \wedge N(\mathbf{v}, \mathbf{u})>0 \end{cases}$

称 $d_a(\mathbf{u}, \mathbf{v})$ 为 \mathbf{u} 、 \mathbf{v} 之间的**非对称距离**， $d_u(\mathbf{u}, \mathbf{v})$ 为**单向距离**，而汉明距离 $d(\mathbf{u}, \mathbf{v})$ 称为**对称**

距离。

这三种距离之间的关系显而易见为

$$d_a(u, v) = d(u, v) + [w(u) - w(v)] \quad (3.10.2)$$

若 $w(u)=w(v)$, 则

$$d(u, v) = d_a(u, v) - d_u(u, v) \quad (3.10.3)$$

一个分组码 C 的单向距离与非对称距离定义为:

$$d_u = \min \{d_u(u, v) | u, v \in C, u \neq v\}$$

$$d_a = \min \{d_a(u, v) | u, v \in C, u \neq v\}$$

例如, C 是码长为8, 含有以下4个码字的集合:

$$C_1 = (10000000) \quad C_2 = (01110000)$$

$$C_3 = (00001110) \quad C_4 = (11101101)$$

显然, 该码是非线性分组码, 它的汉明、非对称和单向距离分别是 $d=4$, $d_a=6$, $d_u=5$ 。

与定理1.3.1相似, 下面定理给出了分组码 C 能纠正 t 个非对称错误或单向错误的充要条件。

定理3.10.1

- (1) 当且仅当 $d_a \geq 2t+1$ 时, 码 C 能纠正 $\leq t$ 个非对称错误;
- (2) 当且仅当 $d_u \geq 2t+1$ 时, 码 C 能纠正 $\leq t$ 个单向错误。

证明 先证明(2), 定义

$$S_0(x) = \{a \in V_n | a \geq x \wedge w(a) - w(x) \leq t\}, x \in V_n$$

$$S_1(x) = \{a \in V_n | x \geq a \wedge w(x) - w(a) \leq t\}, x \in V_n$$

$$S(x) = S_0(x) \cup S_1(x)$$

令 $C_i \in C$, 则可知 $S(C_i)$ 是码字 C_i 通过单向信道后, 遭受到 $\leq t$ 个单向错误(1错误与0错误)后所有可能的矢量集合。因此, 当且仅当 $S(C_1) \cap S(C_2) = 0$ 时, 也就是说码字 C_1 与 C_2 遭受到 $\leq t$ 个单向错误后, 它们的集合互不相交, 才能使译码器正确区分 C_1 与 C_2 , 这与证明定理1.3.1的情况相似。因此, 我们只要证明对所有 $C_1, C_2 \in C$, $S(C_1) \cap S(C_2) = 0$ 的充要条件是 $d_u(C_1, C_2) \geq 2t+1$ 。

充分性。令 $C_1, C_2 \in C$, 且 $C_1 \neq C_2$, $S(C_1) \cap S(C_2) = 0$, 我们要证明 $d_u(C_1, C_2) \geq 2t+1$, 可用反证法证明。

设 $d_u(C_1, C_2) \leq 2t$, $w(C_2) > w(C_1)$ 。分两种情况: $N(C_1, C_2) = 0$ 和 $N(C_1, C_2) > 0$ 。

先讨论 $N(C_1, C_2) = 0$ 的情况。不失一般性, 设 C_1 与 C_2 有如下形式:

$$\begin{array}{cccc} C_1: & 11\cdots 1 & 00\cdots & 0 & 00\cdots 0 \\ C_2: & 11\cdots 1 & 11\cdots & 1 & 00\cdots 0 \\ a: & \underbrace{11\cdots 1}_i & \underbrace{11\cdots 10\cdots 0}_j & \underbrace{00\cdots 0}_l \end{array}$$

这里, $i+j+l=n$, 且 $j=d(C_1, C_2)=d_u(C_1, C_2) \leq 2t$ 。矢量 a 有 b 个1, $n-b$ 个0, 如上图所示, 且 $b=i+\lfloor j/2 \rfloor$ 。由于 $a \geq C_1$, 且 $w(a)-w(C_1)=\lfloor j/2 \rfloor \leq t$, ($\lceil x \rceil$ 表示取 $\geq x$ 的最小整数, 下同)。因而 $a \in S_0(C_1)$ 。此外, $C_2 \geq a$, $w(C_2)-w(a)=j-\lfloor j/2 \rfloor=\lceil j/2 \rceil \leq t$, 所以 $S(C_1) \cap S(C_2) \neq 0$, 与 $S(C_1) \cap S(C_2) = 0$ 的假设相矛盾。

下面讨论 $N(\mathbf{C}_1, \mathbf{C}_2) > 0$ 的情况。不失一般性，码字 $\mathbf{C}_1, \mathbf{C}_2$ 有如下形式：

$$\begin{aligned}\mathbf{C}_1: & 11\cdots 1 \quad 00\cdots 0 \quad 11\cdots 1 \quad 00\cdots 0 \\ \mathbf{C}_2: & 11\cdots 1 \quad 11\cdots 1 \quad 00\cdots 0 \quad 00\cdots 0 \\ \mathbf{a} : & \underbrace{11\cdots 1}_i \quad \underbrace{00\cdots 0}_j \quad \underbrace{00\cdots 0}_k \quad \underbrace{00\cdots 0}_l\end{aligned}$$

这里， $i+j+k+l=n$, $1 \leq k = N(\mathbf{C}_1, \mathbf{C}_2) \leq j = N(\mathbf{C}_2, \mathbf{C}_1) = d_u(\mathbf{C}_1, \mathbf{C}_2)/2 = d_u(\mathbf{C}_1, \mathbf{C}_2)/2 \leq t$ 。定义一个矢量 \mathbf{a} 如上图所示，它有 i 个 1, $n-i$ 个 0。 $\mathbf{C}_1 \geq \mathbf{a}$, 且 $w(\mathbf{C}_2) - w(\mathbf{a}) = k \leq t$, 因而 $\mathbf{a} \in S_1(\mathbf{C}_1)$ 。此外, $\mathbf{C}_2 > \mathbf{a}$ 且 $w(\mathbf{C}_2) - w(\mathbf{a}) = j \leq t$, 从而 $\mathbf{a} \in S_1(\mathbf{C}_2)$, 所以 $S(\mathbf{C}_1) \cap S(\mathbf{C}_2) \neq \emptyset$, 与假设相矛盾。

由以上讨论可知，反证法的假设 $d_u(\mathbf{C}_1, \mathbf{C}_2) \leq 2t$ 不能成立，因而 $d_u(\mathbf{C}_1, \mathbf{C}_2) \geq 2t+1$ 。

必要性。设 $\mathbf{C}_1, \mathbf{C}_2 \in C$, 且 $\mathbf{C}_1 \neq \mathbf{C}_2$, $d_u(\mathbf{C}_1, \mathbf{C}_2) \geq 2t+1$, $w(\mathbf{C}_1) > w(\mathbf{C}_2)$, 我们要证明 $S(\mathbf{C}_1) \cap S(\mathbf{C}_2) = \emptyset$, 可用反证法证明。

设 $S(\mathbf{C}_1) \cap S(\mathbf{C}_2) \neq \emptyset$, 则必存在一个矢量 $\mathbf{a} \in V_n$, $\mathbf{a} \in S(\mathbf{C}_1) \cap S(\mathbf{C}_2)$ 。也分两种情况讨论, $N(\mathbf{C}_1, \mathbf{C}_2) = 0$ 和 $N(\mathbf{C}_1, \mathbf{C}_2) > 0$ 。

先讨论 $N(\mathbf{C}_1, \mathbf{C}_2) = 0$ 的情况。由于 $\mathbf{a} \in S(\mathbf{C}_1) \cap S(\mathbf{C}_2)$, 因而必有

$$d_u(\mathbf{C}_1, \mathbf{C}_2) = d(\mathbf{C}_1, \mathbf{C}_2) \leq d(\mathbf{C}_1, \mathbf{a}) + d(\mathbf{a}, \mathbf{C}_2) = t + t \leq 2t$$

这与所给的条件 $d_u(\mathbf{C}_1, \mathbf{C}_2) \geq 2t+1$ 相矛盾。

若 $N(\mathbf{C}_1, \mathbf{C}_2) > 0$, 不失一般性, 假设码字 $\mathbf{C}_1, \mathbf{C}_2$ 有如下形式：

$$\begin{aligned}\mathbf{C}_1: & 11\cdots 1 \quad 00\cdots 0 \quad 11\cdots 1 \quad 00\cdots 0 \\ \mathbf{C}_2: & \underbrace{11\cdots 1}_i \quad \underbrace{11\cdots 1}_j \quad \underbrace{00\cdots 0}_k \quad \underbrace{00\cdots 0}_l\end{aligned}$$

这里, $i+j+k+l=n$, 且 $1 \leq k = N(\mathbf{C}_1, \mathbf{C}_2) \leq j = N(\mathbf{C}_2, \mathbf{C}_1)$ 。

(1) 设矢量 $\mathbf{a} \in S_0(\mathbf{C}_1) \cap S_0(\mathbf{C}_2)$, 则 $\mathbf{a} \geq \mathbf{C}_1$, $\mathbf{a} \geq \mathbf{C}_2$, $k \leq w(\mathbf{a}) - w(\mathbf{C}_2) \leq t$, 且 $j \leq w(\mathbf{a}) - w(\mathbf{C}_1) \leq t$ 。因此, $d_u(\mathbf{C}_1, \mathbf{C}_2) = d_a(\mathbf{C}_1, \mathbf{C}_2) = 2\max\{j, k\} \leq 2t$ 。这与所给条件 $d_u(\mathbf{C}_1, \mathbf{C}_2) \geq 2t+1$ 相矛盾。

(2) 若 $\mathbf{a} \in S_1(\mathbf{C}_1) \cap S_1(\mathbf{C}_2)$, 则 $\mathbf{C}_1 \geq \mathbf{a}$, $\mathbf{C}_2 \geq \mathbf{a}$, $j \leq w(\mathbf{C}_2) - w(\mathbf{a}) \leq t$, 且 $k \leq w(\mathbf{C}_1) - w(\mathbf{a}) \leq t$, 因此, $d_u(\mathbf{C}_1, \mathbf{C}_2) = d_a(\mathbf{C}_1, \mathbf{C}_2) = 2\max\{j, k\} \leq 2t$ 。这与假设条件 $d_u(\mathbf{C}_1, \mathbf{C}_2) \geq 2t+1$ 相矛盾。

(3) 若 $\mathbf{a} \in S_0(\mathbf{C}_1) \cap S_1(\mathbf{C}_2)$, 则 $\mathbf{C}_1 \geq \mathbf{a} \geq \mathbf{C}_2$, 因此 $j=0$, 这与假设 $j \geq 1$ 相矛盾。

(4) 若 $\mathbf{a} \in S_1(\mathbf{C}_1) \cap S_0(\mathbf{C}_2)$, 则 $\mathbf{C}_2 \geq \mathbf{a} \geq \mathbf{C}_1$, 因此 $k=0$, 这与 $k \geq 1$ 相矛盾。

综上所述可知, $S(\mathbf{C}_1) \cap S(\mathbf{C}_2) = \emptyset$, 由此证明了定理3.10.1(2)。定理3.10.1(1)的证明完全类似, 并且更为简单, 这里不再重复。 ■

定理 3.10.2 一个码 C 能纠正 t 个随机错误, 同时检测 e 个 ($e > t$) 单向错误 (t -EC/ e -UED) 的充要条件是:

$$(1) d(\mathbf{C}_1, \mathbf{C}_2) = t+e+1 \quad \mathbf{C}_1, \mathbf{C}_2 \in C, \mathbf{C}_1 \neq \mathbf{C}_2;$$

或

$$(2) N(\mathbf{C}_1, \mathbf{C}_2) \geq t+1, N(\mathbf{C}_2, \mathbf{C}_1) \geq t+1.$$

证明 由定理1.3.1可知, (1) 是纠 t 个随机错误, 同时检测 e 个随机错误分组码所必

须满足的条件。因此，该条件满足，则码必定能纠正 t 个随机错误同时检测 e 个单向错误。下面主要证明(2)。

若(2)满足，则由式(3.10.1)知，码的汉明距离 $d=2t+2$ ，因而该码必能纠正 t 个随机错误，同时检测 $t+1$ 个随机错误。下面只要证明它能检测 e ($e>t$) 个单向错误即可。

先证明充分性。设码字 C_1 受到了 t_1 个单向错误后成为 C'_1 ， $t < t_1 \leq e$ ，只要证明 $d(C'_1, C_2) \geq t+1$ ，对所有 $C_2 \neq C_1$ ，也就是 C'_1 不可能译成 C_2 。不失一般性，设 C_1 受到了 t_1 个 1 错误。若 $N(C_1, C_2) \geq t+1$ 和 $N(C_2, C_1) \geq t+1$ ，则 $d(C'_1, C_2) = N(C'_1, C_2) + N(C_2, C'_1) \geq N(C_2, C_1) \geq N(C_2, C_1) \geq t+1$ 。

下面证明必要性。证明方法与定理3.10.1的必要性证明相似。设码能纠正 t 个随机错误，同时发现 e 个单向错误，要证明 $N(C_1, C_2) \geq t+1$ 和 $N(C_2, C_1) \geq t+1$ 。用反证法证明。设 $N(C_1, C_2) = t_1 \leq t$ ， $N(C_2, C_1) = t_2 \leq t$ 。不失一般性，设 C_1, C_2 是如下形式的码字：

$$\begin{array}{ll} C_1: & 11\cdots 1 \quad 11\cdots 1 \quad 00\cdots 0 \quad 00\cdots 0 \\ C_2: & 11\cdots 1 \quad 00\cdots 0 \quad 11\cdots 1 \quad 00\cdots 0 \\ a: & \underbrace{11\cdots 1}_i \quad \underbrace{00\cdots 0}_j \quad \underbrace{00\cdots 0}_k \quad \underbrace{00\cdots 0}_l \end{array}$$

这里， $i+j+k+l=n$ ， $j=N(C_1, C_2)=t_1 \leq t$ ， $k=N(C_2, C_1)=t_2 \leq t$ 。注意 a 矢量可以从 C_1 受到 t_1 个 1 错误，或 C_2 受到 t_2 个 1 错误得到，从而使得 a 既含在 $S_1(C_1)$ ，又含在 $S_1(C_2)$ 中，即 $S_1(C_1) \cap S_1(C_2) \neq \emptyset$ ，这与码能纠正 t 个随机错误检测 e 个单向错误的假设相矛盾。 ■

由上述证明可知，如果码 C 满足条件(2)，则它不仅能纠正 t 个随机错误和检测 e ($>t$) 个单向错误，而且能检测一个码字中的所有单向错误。

推论3.10.1 码 C 能纠正 t 个随机错误，同时检测所有单向错误(t -EC/AUED)的充分条件是对码中的任何两个码字 C_1, C_2 ， $C_1 \neq C_2$ ，有

$$N(C_1, C_2) \geq t+1 \text{ 及 } N(C_2, C_1) \geq t+1 \quad (3.10.4)$$

由于在计算机存贮系统(ROM 和 RAM)中，所产生的错误大部分是单向错误和少量的随机错误并存，因此下面主要讨论 t -EC/AUED 码的构造。有关纠非对称错误分组码及纠单向错误码的构造及性质请参阅参考文献[14~16]。

二、 t -EC/AUED 码的构造

在一个码字中若仅产生单个错误，则这个错误既可看成随机错误又可看成单向错误。因此，对于纠单个错误码来说，不必区分它是纠随机错误还是纠单向错误的码。所以汉明码既是纠单个随机错误的最佳码，也是纠单个单向错误的最佳码。但是，对于 t -EC/AUED 码来说，由式(3.10.4)可知，该码中一定没有全0和全1码字，因此不可能是线性码，而纠不对称与单向错误的码则可以是线性码。

构造 t -EC/AUED 码的方法通常有两种：一是在已知的纠 t 个随机错误或不对称错误分组码的基础上，加必要的校验元，使码具有检测所有单向错误(AUED)的能力；二是从已知的 AUED 码的基础上，加上必要的校验元，使码具有纠正 t 个随机错误的能力。

(1) 由已知的 AUED 码构造。最常用的一类 AUED 码是等重码。一个长为 n 、重量为 w 、距离为 d 的 (n, d, w) 等重码，就是 n 中取 w 码。我国电传信用的 $(5, 2, 3)$ 码，就是 5 中取 3 码，而 $(7, 2, 3)$ 码就是 7 中取 3 码，它们的距离均为 2。

(n, d, w) 等重码是非线性码，可以证明码字之间的汉明距离必是偶数^[17]，因而等重码一般用 $(n, 2\delta, w)$ 表示。对于 $(n, 2, w)$ 码 C_w 来说，由于码字之间的距离必是偶数，因此对于任何 $C_1, C_2 \in C_w$ ，必有

$$N(C_1, C_2) \geq 1 \quad N(C_2, C_1) \geq 1$$

由式(3.10.4)知，该码必能检测码字中的所有单向错误。

以 $(n, 2, w)$ 码为基础，可以构造纠 t 个错误的 t -EC/AUED 码。把 $(n, 2, w)$ 码的每个码字作为信息组，输入到纠 t 个错误的 $[n', n, 2t+1]$ 线性分组码的编码器，就得到了 $(n', M, 2t+1)$ 分组码的码字。它的码长 $n' = n+r$ ， r 是纠 t 个错误码的校验位，共有 M 个码字， $M = \binom{n}{w}$ 。

例如， $(4, 2, 2)$ 等重码的 $\binom{4}{2} = 6$ 个码字是：(1100), (0110), (0011), (1010), (0101)和(1001)，把它们输入到 $[7, 4, 3]$ 汉明码编码器，就得到6个码字。设汉明码的 G 矩阵如式(3.3.7)所示，则所得的6个码字为(1100011), (0110101), (0011100), (1010110), (0101001)和(1001010)。它们组成了 $(7, 6, 4)$ 1-EC/AUED 码。该码的码率 $R = (\log_2 6)/7 = 0.37$ ，比 $[7, 4, 3]$ 汉明码的码率0.57要低。

定理 3.10.3 纠 t 个错误的 $(n, 2\delta, w)$ 等重码中，任何两个码字 C_1, C_2 之间必满足：

$$N(C_1, C_2) \geq \delta = t + 1 \quad N(C_2, C_1) \geq \delta = t + 1$$

请读者证明该定理。由此可知，任何 $(n, 2\delta, w)$ 等重码，一定是一个 $(\delta-1)$ -EC/AUED 码。例如 $(8, 4, 4)$ 等重码，它的14个码字是：(10110001), (01011001), (00101101), (00010111), (10001011), (11000101), (01100011), (01001110), (10100110), (11010010), (11101000), (01110100), (00111010), (10011100)。可以检验出每个码字之间的距离等于4，且任两码字 C_1, C_2 之间的 $N(C_1, C_2) \geq 2$ 和 $N(C_2, C_1) \geq 2$ ，因而这是一个1-EC/AUED 码。

该码其实就是 $[7, 4, 3]$ 汉明码加一全校验位后，成为 $[8, 4, 4]$ 扩张汉明码，然后去掉全0和全1码字组成。该码的码率为0.475，比上面列举的 $(7, 6, 4)$ 码的要高。一般而言，用最佳的 $(n, 2\delta, w)$ 等重码组成的 $(\delta-1)$ -EC/AUED 码的码率，比用其它方法构造的码平均要高，但如何构造码字数目最多的最佳 $(n, 2\delta, w)$ 码，仍没有完全解决。下面给出一个系统地构造 $(n, 2\delta=4, w)$ 等重码的方法^[18]。

设 F_w^n 是 $\binom{n}{w}$ 个等重码字集合。作映射

$$T: F_w^n \longrightarrow Z_n$$

Z_n 表示模 n 的剩余类。若等重码字 $a = (a_0, a_1, \dots, a_{n-1}) \in F_w^n$ ，则它的映射

$$T(a) \equiv \sum_{a_i=1}^{n-1} i \pmod{n} \equiv \sum_{i=0}^{n-1} ia_i \pmod{n} \quad (3.10.5)$$

令 $\{C_i\} = \{a \in F_w^n; T(a) \equiv i \pmod{n}\}$ ，则等重码字集合 $\{C_i\}$ 是一个 $(n, 4, w)$ 等重码。

定理3.10.4 由上述方法构造的等重码 C_i ，它的各码字之间的距离至少为4，且对任何 $C_1, C_2 \in C_i$ ，有 $N(C_1, C_2) \geq 2$, $N(C_2, C_1) \geq 2$ 。

证明 设 C_1, C_2 是 C_i 中的任二码字。若它们之间的距离不为4，设为2，则由于每个码字的重量均为 w ，除了两个位置如 r 和 s 以外， C_1, C_2 之间的相应分量应相等，如下所示：

$$\begin{aligned} \mathbf{C}_1: & \quad x \cdots x \ 0 \ x \cdots x \ 1 \ x \cdots x \\ \mathbf{C}_2: & \quad x \cdots x \ 1 \ x \cdots x \ 0 \ x \cdots x \\ 0 \cdots & \quad r \cdots \quad s \cdots n-1 \end{aligned}$$

由于 $T(\mathbf{C}_1)=T(\mathbf{C}_2) \equiv i \pmod{n}$, 所以由式(3.10.5)可知:

$$\begin{aligned} T(\mathbf{C}_1) &= b + s \equiv i \pmod{n} \\ T(\mathbf{C}_2) &= b + r \equiv i \pmod{n} \end{aligned}$$

式中, $b \in Z_n$, 是除了 r 和 s 坐标以外的所有其余坐标之和 (\pmod{n}) 。可知 $s \equiv r \pmod{n}$, 但由于 s, r 均小于 n , 因而 $s \equiv r \pmod{n}$, 所以 $\mathbf{C}_1, \mathbf{C}_2$ 之间的距离不能为 2, 至少为 4。

此外, 若 $N(\mathbf{C}_1, \mathbf{C}_2) < 2$, 例如说为 1, 则由上面的 $\mathbf{C}_1, \mathbf{C}_2$ 形式, 意味着 $\mathbf{C}_1, \mathbf{C}_2$ 的距离为 2, 而这与距离为 4 相矛盾, 因此 $N(\mathbf{C}_1, \mathbf{C}_2) \geq 2$ 。 ■

为了使码字数目最多, 在同样码长 n 下, 应使码字的重量 $w = \lfloor n/2 \rfloor$ 。

如 $n=5, w=2$, 则 $\binom{n}{w} = \binom{5}{2} = 10$ 。对这 10 个码字按 $T(\mathbf{a}) \equiv i \pmod{5}$ 进行分类, 可得 5 类, 每类有两个码字。如取 $i=1$, 则两个码字是: (01110) 和 (10101), 因为:

$$\begin{aligned} T(01110) &= 1 + 2 + 3 = 6 \equiv 1 \pmod{5} \\ T(10101) &= 0 + 2 + 4 = 6 \equiv 1 \pmod{5} \end{aligned}$$

用这种方法构造的 $(n, 4, w)$ 码 $\mathbf{C}_i (i=0, 1, \dots, n-1)$, 它的码字数目 $|\mathbf{C}_i| \geq \binom{n}{w}/n$, 并没有达到最大。例如 $(8, 4, 4)$ 码, 它的 $|\mathbf{C}_i| \geq \binom{8}{4}/8 \geq 9$, 比上面例中的码字数目 14 要少很多。如何寻找具有最多码字数目的 $(n, 2\delta, w)$ 等重码, 目前还没有完全解决。有关等重码的其它性质及码字数目的上、下限请参阅 [18]。而用等重码构造 t -EC/AUED 码的其它方法可参阅 [19]。

(2) 由已知的 $[n, k, 2t+1]$ 码构造。这种方法是在纠 t 个随机错误的 $[n, k, 2t+1]$ 线性分组码(非线性码也可以)或纠非对称错误码的基础上, 在其每个码字后面适当加一些多余码元而构成。因此, 这类码可以是系统码, 而用(1)中方法构造的码一般是非系统码。

用纠 t 个随机错误的系统分组码 C , 或纠 t 个非对称错误的系统分组码构造系统 t -EC/AUED 码的关键, 是在 C 码的每个码字后面加上适当的多余度码元, 使式(3.10.4)满足。在每个码字后面所附加的多余(校验)元素应满足什么条件才能使每个码字满足式(3.10.4)呢? 为此, 必须知道 C 码中每对码字 $\mathbf{C}_1, \mathbf{C}_2$ 之间的 $N(\mathbf{C}_1, \mathbf{C}_2)$ 的性质。

定理3.10.5 令 C 是 $[n, k, 2t+1]$ 分组码, $\mathbf{C}_1, \mathbf{C}_2 \in C$, 且 $w(\mathbf{C}_1) \geq w(\mathbf{C}_2)$, 则有:

$$\begin{aligned} N(\mathbf{C}_1, \mathbf{C}_2) &\geq t+1 \\ N(\mathbf{C}_2, \mathbf{C}_1) &\geq \max\{0, t+1 - \lfloor (w(\mathbf{C}_1) - w(\mathbf{C}_2))/2 \rfloor\} \end{aligned} \tag{3.10.6}$$

证明 设 $\mathbf{C}_1, \mathbf{C}_2$ 有如下形式:

$$\begin{aligned} \mathbf{C}_1: & \quad 11 \cdots 1 \quad 11 \cdots 1 \quad 00 \cdots 0 \quad 00 \cdots 0 \\ \mathbf{C}_2: & \quad \underbrace{11 \cdots 1}_i \quad \underbrace{00 \cdots 0}_j \quad \underbrace{11 \cdots 1}_q \quad \underbrace{00 \cdots 0}_p \end{aligned}$$

这里, $i+j+q+p=n$ 。显然, $w(\mathbf{C}_1)=i+j$, $w(\mathbf{C}_2)=i+q$, $d(\mathbf{C}_1, \mathbf{C}_2)=j+q \geq 2t+1$ 。 $j=w(\mathbf{C}_1)-i=w(\mathbf{C}_1)-(w(\mathbf{C}_2)-q)=w(\mathbf{C}_1)-w(\mathbf{C}_2)+q \geq q$ 。 $2N(\mathbf{C}_1, \mathbf{C}_2)=2j \geq j+q \geq 2t+1$, 所以 $N(\mathbf{C}_1, \mathbf{C}_2) \geq t+1$ 。

另一方面, $j = w(\mathbf{C}_1) - i = w(\mathbf{C}_1) - (w(\mathbf{C}_2) - q) = q + w(\mathbf{C}_1) - w(\mathbf{C}_2)$, 由此可知: $2q + w(\mathbf{C}_1) - w(\mathbf{C}_2) = j + q = d(\mathbf{C}_1, \mathbf{C}_2) \geq 2t + 1$ 。所以

$$N(\mathbf{C}_2, \mathbf{C}_1) = q \geq \lceil (2t + 1 - w(\mathbf{C}_1) + w(\mathbf{C}_2)) / 2 \rceil = t + 1 - \lceil (w(\mathbf{C}_1) - w(\mathbf{C}_2)) / 2 \rceil$$

若 $\lceil (w(\mathbf{C}_1) - w(\mathbf{C}_2)) / 2 \rceil > t + 1$, 则 $N(\mathbf{C}_2, \mathbf{C}_1) = 0$ 。由此得

$$N(\mathbf{C}_2, \mathbf{C}_1) \geq \max\{0, t + 1 - \lceil (w(\mathbf{C}_1) - w(\mathbf{C}_2)) / 2 \rceil\}$$

由该定理可知, 若 $w(\mathbf{C}_1) > w(\mathbf{C}_2)$, 则在 \mathbf{C}_2 后面所加的校验元的重量至少应为 $t + 1$ 或 $\lceil (w(\mathbf{C}_1) - w(\mathbf{C}_2)) / 2 \rceil$, 以保证式(3.10.4)满足。根据不同的加校验码元的方法就得到了不同的码。其中最早和最重要的一种方法, 是伯杰(Berger)在构造纠非对称错误码时提出的构造方法(B方法)^[20]。

令 C 表示 $[n, k, 2t + 1]$ 系统分组码, C^* 表示 $(n^*, k) t$ -EC/AUED 系统码。 C^* 的码字有如下形式:

$$\mathbf{x}_0 \mathbf{x}_1 \mathbf{x}_2 \cdots \mathbf{x}_{t+1} \quad \mathbf{x}_0 \in C \quad (3.10.7)$$

也就是 C^* 中每一码字, 是在 C 中码字 \mathbf{x}_0 后面加一些校验元 $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{t+1}$ 构成。这里, \mathbf{x}_i ($i = 1, 2, \dots, t + 1$) 是用 B 方法产生: \mathbf{x}_i 是 $\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_{i-1}$ 中 0 数目的二进制表示序列。可知用这种方法构造的 C^* 码是系统码。

表3-5 用B和THL方法得到的系统1-EC/AUED码

| B码 | | | THL码 | |
|----------------|----------------|----------------|----------------|----------------|
| \mathbf{x}_0 | \mathbf{x}_1 | \mathbf{x}_2 | \mathbf{x}_1 | \mathbf{x}_2 |
| 0000000 | 111 | 111 | 11 | 11 |
| 1011000 | 100 | 110 | 10 | 01 |
| 0101100 | 100 | 110 | 10 | 01 |
| 0010110 | 100 | 110 | 10 | 01 |
| 0001011 | 100 | 110 | 10 | 01 |
| 1000101 | 100 | 110 | 10 | 01 |
| 1100010 | 100 | 110 | 10 | 01 |
| 0110001 | 100 | 110 | 10 | 01 |
| 1110100 | 011 | 100 | 01 | 10 |
| 0111010 | 011 | 100 | 01 | 10 |
| 0011101 | 011 | 100 | 01 | 10 |
| 1001110 | 011 | 100 | 01 | 10 |
| 0100111 | 011 | 100 | 01 | 10 |
| 1010011 | 011 | 100 | 01 | 10 |
| 1101001 | 011 | 100 | 01 | 10 |
| 1111111 | 000 | 011 | 00 | 00 |

以 $[7, 4, 3]$ 系统汉明码 C 为例, 说明系统的 1-EC/AUED 码 C^* 的构造方法。设 $\mathbf{C}_1 = (1011000) \in C$, 由式(3.10.7)知需求出 $\mathbf{x}_1, \mathbf{x}_2$ 。 \mathbf{C}_1 中有 4 个 0, $\mathbf{x}_1 = 100$, \mathbf{C}_1 和 \mathbf{x}_1 中共有 6 个 0, $\mathbf{x}_2 = 110$, 因而相应的码字 $\mathbf{C}_1^* = (1011000 100 110)$ 。表 3-5 中的左边栏列出了用 B 方法, 在 $[7, 4, 3]$ 码基础上构造的系统 1-EC/AUED 码 C^* 。显然, C^* 是一个 $(13, 4)$ 系统

1-EC/AUED 码。

从 B 方法求 x_1, x_2, \dots, x_{t+1} 的原理可知, 码中所有重量相同的码字, 具有相同的 x_1, x_2, \dots, x_{t+1} , 如表 3-5 所示。这就给我们一个减少 x_1, \dots, x_{t+1} 中码元数目的方法。其中一个方法就是下面的 THL^[21]方法。它把 $[n, k, 2t+1]$ 码 C 中的码字按重量分组: $A_0, A_{2t+1}, A_{2t-2}, \dots, A_n$, 这里, A_i 表示重量为 i 的码字集合, 可见, A_i 中每一码字的重量均相等, 故是一等重码。由定理 3.10.3 知, A_i 中任两码字之间必满足式(3.10.4)。因此, A_i 中的 x_1, \dots, x_{t+1} 均可相同; 而 A_i 与 A_j 之间则应取不同的 x_1, \dots, x_{t+1} , 它们的重量应满足定理 3.10.5 的条件, 使式(3.10.4)满足。可知 x_i 中的码元数目 a_i , 必须满足:

$$2^{a_i} \geq |A_w| \quad (3.10.8)$$

式中, $|A_w|$ 是 C 码中重量类型数目。如 $[7, 4, 3]$ 码中的码字重量又有 0, 3, 4, 7 四种, $|A_w|=4$, 因而 $a_i \geq 2$, 也就是 $x_i (i=1, 2)$ 中只要两个码元就够了。由此得到的 x_1 和 x_2 , 如表 3-5 中右边栏所示。用这种 THL 方法得到的系统 1-EC/AUED 码是 $(11, 4)$ 码, 它比 $(13, 4)$ 码的码率要高得多。有关这类码的详细构造方法和译码请参阅 [21, 26]。

除了上述构造系统 t -EC/AUED 码的方法以外, 最近还提出了许多其它构造方法如: NGP^[23]、Andrew^[24]、GKR^[25] 和 BT^[22] 等, 这些方法所构造的码, 其码率均比用 B 方法构造的要高。

定理 3.10.6^[26] 由 $[n, k, 2t+1]$ 线性系统分组码构造的系统 t -EC/AUED 码, 其校验元

$$r \leq (n-k) + (t+1)\log_2 n \quad (3.10.9)$$

定理 3.10.7^[27] k 长信息位的二进制系统 t -EC/AUED 码的校验元

- (1) $r \geq \lceil \log_2(k+1) \rceil$ 当 $t=0$
- (2) $r \geq \lceil (t+1)\log_2 k - \log_2((t+1)!) \rceil$ 当 $1 \leq t \leq 4$
- (3) $r \geq \lceil (t+1)\log_2 k - \log_2((t+1)!) \rceil - 1$ 当 $t \geq s$ 且 $k \gg t$

上述这些码限以及码字数目的限, 请参阅文献 [22, 27, 28]。由于系统 t -EC/AUED 码在计算机纠、检错系统中有较大作用, 因此有关这类码的研究方兴未艾^[23, 24, 27], 值得注意。特别是 Rao 等近来出版了一本计算机纠错码的专著, 详细讨论了上述这些码的构造原理、方法和性能, 有兴趣读者可参阅 [29]。

习题

1. 证明 $[n, k]$ 线性分组码的最大距离为 $n-k+1$ 。
2. 设一个 $[7, 4]$ 码的生成矩阵为

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

- (1) 求出该码的全部码矢;
- (2) 求出该码的一致校验矩阵;
- (3) 作出该码的标准阵译码表。

3. 证明定理3.1.3。

4. 一个[8, 4]系统码，它的一致校验方程为：

$$c_0 = m_1 + m_2 + m_3$$

$$c_1 = m_0 + m_1 + m_2$$

$$c_2 = m_0 + m_1 + m_3$$

$$c_3 = m_0 + m_2 + m_3$$

式中， m_0, m_1, m_2, m_3 是信息位， c_0, c_1, c_2, c_3 是校验位。找出该码的 G 和 H ，并证明该码的最小距离为4。

5. 构造第4题中码的对偶码。

6. 设 H_1 是 $[n, k]$ 线性分组码 C_1 的校验矩阵，且有奇数最小距离为 d 。作一个新的码 C_2 ，它的校验矩阵为

$$H_2 = \left[\begin{array}{cccc|c} & & & & 0 \\ & H_1 & & & 0 \\ & & & & \vdots \\ & & & & 0 \\ \hline 1 & 1 & \cdots & 1 & 1 \end{array} \right]$$

(1) 证明 C_2 是一个 $(n+1, k)$ 分组码；

(2) 证明 C_2 中每一码字的重量为偶数；

(3) 证明 C_2 码的最小重量为 $d+1$ 。

7. 设 C_1 是一个有最小距离为 d_1 的 $[n_1, k]$ 线性系统码，生成矩阵为 $G_1 = [P_1 I_k]$ 。 C_2 是一个有最小距离为 d_2 的 $[n_2, k]$ 线性系统码，它的生成矩阵 $G_2 = [P_2 I_k]$ 。对满足下述一致校验矩阵

$$H = \left[\begin{array}{c|cc} P_1^T & & \\ \hline I_{n-n_2-k} & I_k \\ P_2^T & \end{array} \right]$$

的 $[n_1+n_2, k]$ 线性码，证明它有最小距离至少为 d_1+d_2 。

8. 设一个二进制 $[n, k]$ 码 C 的 G 矩阵不含全零列，将 C 的所有码字排成 $2^k \times n$ 的阵。证明：

(a) 阵中不含有全零列；

(b) 阵中的每一列由 2^{k-1} 个零和 2^{k-1} 个1组成；

(c) 在一特定分量上为0的所有码字构成 C 的一个子空间，问该子空间的维数是多少？

9. 令 Γ 是所有二进制 $[n, k]$ 线性系统码的集合。证明非零二进制 n 重 V 或者恰巧含于 Γ 的 $2^{(k-1)(n-k)}$ 个码中，或者不在 Γ 的任一码中。

10. 证明二进制 $[23, 12, 7]$ Golay 码和三进制的 $[11, 6, 5]$ Golay 码是完备码。

11. 若 d 是码 C 的最小重量，且为偶数， $t = \lfloor (d-1)/2 \rfloor$ 。证明有两个重量均为 $t+1$ 的矢量必在 C 码的同一陪集中。

12. 求出 $d=3$ ，至多只有3个校验元的二进制码的码长 n ；和 $d=5$ ，至多只有8个校验元的二进制码的码长 n 。

13. 计算二进制[24, 12, 8]扩张Golay码的覆盖半径, 及[8, 4]RM码的覆盖半径。
14. 证明定理3.9.3。
15. 构造三个二进制的[10, 3, 5]LUEP码, 其分离矢量分别为(8, 2, 2), (7, 4, 4), (6, 4, 4)。写出它们标准形式的 G 和系统码形式的 G 。
16. 证明定理3.10.3。
17. 构造一个具有最高码率的 $k=10, t=2$ 的2-EC/AUED码。
18. 证明定理3.10.6。

参 考 文 献

- [1] 林舒, D. J. Costello, Jr. :《差错控制编码: 基础和应用》(王育民、王新梅译), 人民邮电出版社, 1986.
- [2] F. J. MacWilliams, N. J. A. Sloane, *The Theory of Error-Correcting Codes*, Amsterdam, North-Holland, 1977.
- [3] V. Pless, *Introduction to the Theory of Error-Correcting Codes*, John Wiley and Sons, 1982.
- [4] 今井秀樹:《符号理論》,電子情報通信学会, 1990.
- [5] 王育民, 梁传甲:《信息论与编码理论》, 西北电讯工程学院出版社, 1986.
- [6] G. D. Cohen et al., “Covering radius—survey and recent results”, *IEEE Trans. on IT*, No. 3, pp. 328—343, 1985.
- [7] R. L. Graham and N. J. A. Sloane, “On the covering radius of codes”, *IEEE Trans. on IT*, No. 3, pp. 385—401, 1985.
- [8] G. D. Cohen, “A Nonconstructive upper bound on covering radius”, *IEEE Trans. on IT*, No. 3, pp. 356—363, 1983.
- [9] A. R. Calderbank and N. J. A. Sloane, “Inequalities for covering radius”, *IEEE Trans. on IT*, No. 5, pp. 1276—1280, 1988.
- [10] L. A. Dunning, “Optimal encodings of linear block codes for unequal error protection”, *Information and Control*, Vol. 37, pp. 150—177, 1978.
- [11] I. M. Boyarinov and G. L. Katsman, “Linear unequal error protection codes”, *IEEE Trans. on IT*, No. 2, pp. 168—175, 1981.
- [12] W. J. V. Gils, “Two topics on linear unequal error protection codes bounds on their length and cyclic code classes”, *IEEE Trans. on IT*, No. 6, pp. 866—876, 1983.
- [13] M. C. Lin, S. Lin, “Cyclic unequal error protection codes constructed from cyclic codes of composite length”, *IEEE Trans. on IT*, No. 4, pp. 867—871, 1988.
- [14] J. H. Weber, C. de Vroedt and D. E. Boekee, “Bounds and constructions for binary codes of length less than 24 and asymmetric distance less than 6”, *IEEE Trans. on IT*, No. 5, pp. 1321—1331, 1988.
- [15] P. Delsarte and P. Piret, “Bounds and constructions for binary asymmetric error-correcting codes”, *IEEE Trans. on IT*, No. 1, pp. 125—128, 1981.
- [16] J. H. Weber, C. de Vroedt and D. E. Boekee, “Bounds and constructions for codes correcting unidirectional errors”, *IEEE Trans. on IT*, No. 4, pp. 797—810, 1989.
- [17] 王新梅: 最佳(n, 2, w)二进制等重检错码的存在性及猜想, 《中国科学》, A辑, No. 11, pp. 1225—1232, 1987.
- [18] R. L. Graham and N. J. A. Sloane, “Lower bounds for constant weight codes”, *IEEE Trans. on IT*, No. 1, pp. 37—40, 1980.
- [19] B. Bose and T. R. N. Rao, “Theory of unidirectional error correcting/detecting codes”, *IEEE Trans. on*

- C, No. 6, pp. 521—530, 1982.
- [20] J. M. Berger, “A note on error detecting codes for asymmetric channels”, Inform. Contr., Vol. 4, pp. 68—73, 1961.
- [21] D. L. Tao, C. R. P. Hartmann and P. K. Lala, “An efficient class of unidirectional error detecting /correcting codes”, IEEE Trans. on C, No. 7, pp. 879—882, 1988.
- [22] M. Blaum and H. V. Tilborg, “On t —error correcting/all unidirectional error detecting codes”, IEEE Trans. on C, No. 11, pp. 1493—1501, 1989.
- [23] D. N. Kolos, N. Gaitanis and Philokyprou, “Systematic t —error correcting/all unidirectional error detecting codes”, IEEE Trans. on C, No. 5, pp. 394—401, 1986.
- [24] R. Andrew, Construction of t —EC/AUED Codes, Electronics Letters, No. 20, 1988.
- [25] 斎藤雄一, 山口和彦, 今井秀樹, 効率の良い t 重誤ソ訂正/全一方向誤ソ検出符号, 電子情報通信学会論文志(信学志)A, No. 7, pp. 1125—1130, 1989.
- [26] B. Bose and D. K. Pradhan, “Optimal unidirectional error detecting/correcting codes”, IEEE Trans. on C, No. 6, pp. 564—568, 1982.
- [27] 陈溧: t —EC—MUED 系统码的一个注记, 《计算机学报》, No. 4, pp. 306—312, 1984.
- [28] D. J. Lin and B. Bose, “Theory and design of t —error correcting and $d(d>t)$ —unidirectional error detecting (t —EC d —UED) codes”, IEEE Trans. on C, No. 4, pp. 433—439, 1988.
- [29] T. R. N. Rao and E. Fujiwara: Error—Control Coding for Computer Systems, Prentice Hall, 1989.

第四章 多项式环与有限域

多项式环与有限域是学习编码理论，特别是学习循环码理论的基础，它在编码理论中起着关键作用。这一章主要介绍多项式环的基本概念和有限域中的一些重要性质，以及有限域的结构。

§ 4.1 子环与理想

一、子环

类似于子群，在环中也可定义子环。

定义 4.1.1 若环中的子集 S ，关于 R 中的代数运算也构成环，则称 S 是 R 的子环， R 为 S 的扩环。

定理 4.1.1 非空子集 S 是 R 的子环的充要条件是：

- (1) 对任何两个元素 $a, b \in S$ ，恒有 $a - b \in S$ ；
- (2) 对任何 $a, b \in S$ ，恒有 $ab \in S$ 。

证明 若 S 是 R 的子环，则由环的定义可知，条件(1)、(2)显然成立。

若条件(1)、(2)成立，则由定理 2.4.2 可知， S 对环 R 中的加法运算构成可换子群，并且 S 关于 R 的乘法封闭。由于 S 是环中的子集，因此 R 中的乘法结合律，加法和乘法间的分配律在 S 中自然成立，由此可知 S 是一个环。■

例 4.1 全体整数集合构成一个可换环。以某一整数 m 的倍数全体构成其中的一个子集，该子集就是整数环中的一个子环。如 $m=3$ ，则所有 3 的倍数的全体集合 $\{0, \pm 3, \pm 6, \pm 9, \dots\}$ 构成一个子环。■

一个环至少有两个子环，一个是由 0 元素组成，一个是环 R 本身，称此两个子环为环 R 的假子环，其余的称为真子环。

二、理想

理想是很重要的一类子环，它在环中的作用相当于正规子群在群中的作用。

定义 4.1.2 设 R 是交换环， I 是 R 的非空子集，若

- (1) 对任意两个元素 $a, b \in I$ ，恒有 $a - b \in I$ ；
- (2) 对任意 $a \in I, r \in R$ ，恒有 $ar = ra \in I$ ，则称 I 是 R 中的一个理想。

由此定义可知，理想其实就是可换环中的一个子环，该子环也称双边子环。由(1)可知 I 是一个阿贝尔加群，由(2)可知 I 的某些元素都由某一元素 a 的倍数组成。因此，若理想中包含了元素 a ，则它就包含了 a 的一切倍元。由于 I 构成一个可换加群，所以可用它作为一个正规子群，把 R 中的元素进行分类划分陪集。

例 4.2 全体整数 \mathbf{Z} 构成一个可换环, 某一整数 m 的倍数构成一个理想 I_m , 以此理想可把全体整数 \mathbf{Z} 划分陪集。如 $m=5$, 则 $I_5: \{0, \pm 5, \pm 10, \pm 15, \dots\}$, 划分陪集如下:

$$\begin{aligned}\bar{0}: & 0, 5, -5, 10, -10, \dots \\ \bar{1}: & 1, 6, -4, 11, -9, \dots \\ \bar{2}: & 2, 7, -3, 12, -8, \dots \\ \bar{3}: & 3, 8, -2, 13, -7, \dots \\ \bar{4}: & 4, 9, -1, 14, -6, \dots\end{aligned}$$

由正规子群的理论可知, 这些陪集全体 $\{\bar{0}, \bar{1}, \bar{2}, \bar{3}, \bar{4}\}$ 构成一个模 5 的阿贝尔加群, 称为模 5 的剩余类群, 用 $Z \pmod{5}$ 表示; 一般情况用 $R \pmod{I}$ 表示之。 ■

若 R 中任二元素 a, b 属于 I 的同一剩余类, 则称 a, b 对模 I 同余, 记为

$$a \equiv b \pmod{I}$$

或

$$\bar{a} - \bar{b} \equiv 0 \pmod{I}$$

由上看到, 每个剩余类 \bar{a} 中的所有元素都是如下形式:

$$\bar{a} = a + I$$

我们定义剩余类 \bar{a}, \bar{b} 之积为

$$\bar{a} \cdot \bar{b} = \overline{ab} = ab + I$$

现在证明此种定义的合理性。为此只要证明定义 \overline{ab} 不因代表元的不同选择而改变。

设 $a_1 \in \bar{a}, b_1 \in \bar{b}$, 由定义可知:

$$a_1 = a + i_1 \quad b_1 = b + i_2 \quad i_1, i_2 \in I$$

于是

$$\overline{a_1 b_1} = (a + i_1)(b + i_2) = ab + (ai_2 + bi_1 + i_1 i_2)$$

由于 I 是一个理想, 所以 $(ai_2 + bi_1 + i_1 i_2) \in I$, 故

$$\overline{a_1 b_1} = ab + I$$

所以

$$\overline{a_1 b_1} = \overline{ab} \pmod{I}$$

可知不因代表元 a_1, b_1 的不同而变化。

定理 4.1.2 设 R 是可换环, I 为 R 的一个理想, 于是 R 模 I 构成一个可换环, 称它为环 R 以理想 I 为模的剩余类环 M 。

证明

(1) 对任意 $\bar{a}, \bar{b} \in M$, 一定存在有元素 $a, b \in R$, 且 $\bar{a} = a + I, \bar{b} = b + I$ 。所以

$$\bar{a} - \bar{b} = a + I - (b + I) = a - b + (I - I) = c + I = \bar{c} \in M$$

由定理 2.4.2 可知 M 构成阿贝尔加群。

(2) 乘法封闭性成立, 交换律成立。对一切元素 $\bar{a}, \bar{b} \in M$, 恒有

$$\bar{a} \cdot \bar{b} = \overline{ab} = ab + I \in M$$

$$\bar{a} \cdot \bar{b} = \overline{ab} = \overline{ba} = \bar{b} \cdot \bar{a}$$

(3) 结合律成立

$$(\bar{a} \cdot \bar{b}) \cdot \bar{c} = (\bar{a}\bar{b}) \cdot \bar{c} = \overline{(ab)c} = \overline{a(bc)} = \bar{a}(\bar{b}\bar{c}) = \bar{a} \cdot (\bar{b}\bar{c})$$

由此可知 M 组成一可换环。 ■

由理想的(2)条件可知, 理想中的一切元素都是某些元素的倍数组成。如果理想中的元素由一个元素的所有倍数及其线性组合生成, 则称这个理想为**主理想**。

定义 4.1.3 在可换环 R 中, 由一个元素 $a \in R$ 所生成的理想 $I(a) = \{ra + na \mid r \in R, n \in \mathbb{Z}\}$ 称为环 R 的一个**主理想**, 称元素 a 为该**主理想的生成元**。

在例 4.2 中, 以某一整数 m 的倍数所构成的理想 I_m 就是一个**主理想**, m 是**生成元**。

如果 R 是一个有单位元素的无零因子可换环, 且 R 中的每一理想都是**主理想**, 则称该 R 为**主理想环**。例如, 整数环 \mathbb{Z} 就是一个**主理想环**。

§ 4.2 多项式剩余类环

一、多项式的基本性质

定义 4.2.1 含有一个未定元数 x 的域 $F_p(\text{GF}(p))$ 上的多项式定义为

$$f(x) = f_n x^n + f_{n-1} x^{n-1} + \cdots + f_1 x + f_0$$

$$f_i \in F_p \quad i = 0, 1, 2, \dots, n$$

且用 $F_p[x]$ (或 $\text{GF}(p)[x]$) 表示系数取自域 F_p 上的一切多项式集合。

下面介绍有关多项式的几个名词。

多项式次数: 称系数不为零的 x 的最高次数为多项式 $f(x)$ 的次数, 记为 $\deg f(x)$ ($\partial f(x)$) 或 $\deg f(x)$ ($\deg f$)。

由此可知, 若 $f_n \neq 0$, 则 $\deg f(x) = n$, 称 $f(x)$ 为 n 次多项式。若 $f_n = f_{n-1} = \cdots = f_{k-1} = 0$, $f_k \neq 0$, 则 $\deg f(x) = k$, 称 $f(x)$ 为 k 次多项式。

首一多项式: 最高次数的系数为 1 的多项式称为**首一多项式**。

下面定义多项式之间的相等、相乘和相加规则。设

$$f(x) = f_n x^n + f_{n-1} x^{n-1} + \cdots + f_1 x + f_0 \quad f_i \in F_p$$

$$g(x) = g_n x^n + g_{n-1} x^{n-1} + \cdots + g_1 x + g_0 \quad g_i \in F_p$$

若对所有

$$f_i = g_i \quad i = 0, 1, 2, \dots, n$$

则称 $f(x) = g(x)$ 。

多项式的相加运算为

$$f(x) + g(x) = (f_n + g_n)x^n + (f_{n-1} + g_{n-1})x^{n-1} + \cdots + (f_1 + g_1)x + (f_0 + g_0)$$

多项式 $f(x) = f_n x^n + \cdots + f_1 x + f_0$ 与 $g(x) = g_m x^m + \cdots + g_1 x + g_0$ 的相乘运算为

$$f(x)g(x) = h_{n+m} x^{n+m} + h_{n+m-1} x^{n+m-1} + \cdots + h_1 x + h_0$$

式中

$$h_i = \begin{cases} \sum_{j=0}^i f_j g_{i-j} & i = 0, 1, 2, \dots, m, \quad n \geq m \\ \sum_{j=i-m}^n f_j g_{i-j} & i = m+1, m+2, \dots, m+n \end{cases}$$

定理 4.2.1 按以上定义的多项式运算, $F_p[x]$ 构成一个具有单位元素、无零因子的可换环。

证明

(1) $F_p[x]$ 关于加法构成可换群。

1° 封闭性显然成立。

2° 零元为 $f(x) = 0$ 。

3° $f(x) = f_n x^n + f_{n-1} x^{n-1} + \dots + f_1 x + f_0$ 的逆元为 $-f(x) = -f_n x^n - f_{n-1} x^{n-1} - \dots - f_1 x - f_0$ 。

4° 交换律和结合律显然成立。

(2) 乘法封闭性成立。乘法单位元是 $f(x) = 1$ 。

(3) 无零因子。

设 $f(x) \neq 0, g(x) \neq 0$, 令 f_α, g_β 分别是 $f(x)$ 和 $g(x)$ 的最高次数的系数, 即 $f_\alpha \neq 0, g_\beta \neq 0$ 。因此

$$f_\alpha g_\beta \neq 0 \quad f_\alpha g_\beta \in F_p$$

因为域 F_p 无零因子, 而 $f_\alpha g_\beta$ 是 $f(x)g(x)$ 的最高次数项, 且不为 0, 故 $f(x)g(x) \neq 0$ ■

$F_p[x]$ 中的任何多项式不一定有乘法逆元, 所以 $F_p[x]$ 只能组成一个有单位元素的无零因子环, 这与整数环 \mathbf{Z} 完全相似。因此, 整数环 \mathbf{Z} 中的很多性质和算法都适用于 $F_p[x]$ 环。例如, 在整数环 \mathbf{Z} 上有欧几里德除法, 同样在 $F_p[x]$ 环上也有这种算法。

定理 4.2.2 给定任意两个多项式 $f(x)g(x) \in F_p[x]$, 一定存在有唯一的多项式 $q(x)$ 和 $r(x)$ 使

$$f(x) = q(x)g(x) + r(x) \quad 0 \leq \deg r(x) < \deg g(x) \quad \text{或} \quad r(x) = 0$$

该定理的证明与定理 2.1.2 的证明完全类似, 这里不再重复。

二、既约多项式

与 \mathbf{Z} 环上的素数相对应, 在 $F_p[x]$ 上有既约多项式。

定义 4.2.2 设 $f(x)$ 是次数大于零的多项式, 若除了常数和常数与本身的乘积以外, 再不能被域 F_p 上的其它多项式除尽, 则称 $f(x)$ 为域 F_p 上的既约多项式。

由上述定义可得出:

1° 一个常数总是多项式的因子。例如, 实数域上的多项式 $3x^2 + 1 = 3(x^2 + 1/3)$, 由此可知 3 是 $3x^2 + 1$ 多项式的一个因子。

2° $f(x)$ 是否既约与讨论的域有很大关系。例如, $f(x) = x^2 + 1$, 在实数域上是既约的; 但在复数域上就可以分解为 $f(x) = (x+i)(x-i)$ 。

判别 $f(x)$ ($\deg f > 0$) 是否既约的方法是: $f(x)$ 既约的充要条件是 $f(x)$ 不能再分解成两个次数低于 $\deg f$ 的多项式的乘积。所以, 不论在哪一域上, 凡是一次首一多项式都

是既约多项式。

定理 4.2.3 每一个首一多项式 $f(x)$ 必可分解为首一既约多项式之积，并且当不考虑因式的顺序时，这种分解是唯一的，即

$$f(x) = p_1(x)^{\alpha_1} p_2(x)^{\alpha_2} \cdots p_s(x)^{\alpha_s}$$

式中， $p_i(x)$ 为首要一既约多项式， α_i 是某一正整数， $i=1, 2, \dots, s$ 。

推论 4.2.1 d 次多项式 $f(x)$ 不可能有多于 d 个的一次因式。

定理 4.2.4 α 为多项式 $f(x)$ 之根的充要条件是：

$$(x - \alpha) | f(x)$$

证明 由多项式的欧几里德除法可知

$$f(x) = q(x)(x - \alpha) + r(x) = q(x)(x - \alpha) + r \quad \partial \cdot r(x) < \partial \cdot (x - \alpha)$$

若 α 是 $f(x)$ 的根，则

$$f(\alpha) = q(\alpha)(\alpha - \alpha) + r = 0$$

可知 $r=0$ ，故

$$f(x) = q(x)(x - \alpha) \quad (x - \alpha) | f(x)$$

反之，若 $(x - \alpha) | f(x)$ ，则

$$f(x) = q(x)(x - \alpha)$$

因为 $\alpha - \alpha = 0$ ，故 α 是 $x - \alpha$ 的根，由此

$$f(\alpha) = q(\alpha)(\alpha - \alpha) = 0$$

α 也是 $f(x)$ 的根。 ■

定理 4.2.5 d 次多项式至多有 d 个根。

证明 假定 d 次多项式 $f(x)$ 有多于 d 个根，则由定理 4.2.3 和定理 4.2.4 可知， $f(x)$ 应当有多于 d 个的一次因式，这与推论 4.2.1 相矛盾。

定理 4.2.6 若 $p(x)$ 是 $f(x)$ 的 k 重既约因式，则 $p(x)$ 必是 $f(x)$ 的导数 $f'(x)$ 的 $k-1$ 重既约因式。 ■

证明 由假设可知：

$$f(x) = p^k(x)q(x)$$

$q(x)$ 不可能再含 $p(x)$ 的因子，否则与假设相矛盾，所以 $p(x) \nmid q(x)$ 。

$$f'(x) = kp^{k-1}(x)p'(x)q(x) + q'(x)p^k(x) = p^{k-1}(x)[kp'(x)q(x) + q'(x)p(x)]$$

可知 $p^{k-1}(x) | f'(x)$ ，故 $f'(x)$ 是 $p(x)$ 的 $k-1$ 重既约因式。现在只要证明 $[kp'(x)q(x) + q'(x)p(x)]$ 中没有 $p(x)$ 因子，即 $p(x) \nmid [kp'(x)q(x) + q'(x)p(x)]$ 。若

$$p(x) | [kp'(x)q(x) + q'(x)p(x)]$$

则

$$p(x) | kp'(x)q(x)$$

因为 $\partial \cdot p'(x) < \partial \cdot p(x)$ ，故 $p(x) \nmid kp'(x)$ ，因此 $p(x) \nmid q(x)$ ，这与 $q(x)$ 无 $p(x)$ 的因子相矛盾，所以

$$p(x) \nmid [kp'(x)q(x) + q'(x)p(x)]$$
 ■

与整数的最大公约数和最小公倍数相对应，在多项式中也有最大公因式和最小公倍式。

三、最大(高)公因式

定义 4.2.3 若 $(f(x), g(x), \dots, k(x))$ 是同时除尽多项式 $f(x), g(x), \dots, k(x)$ 的次数最高的首一多项式，则称 $(f(x), g(x), \dots, k(x))$ 是 $f(x), g(x), \dots, k(x)$ 的**最大公因式**，用 $(f(x), g(x), \dots, k(x))$ 或 $\text{GCD}\{f(x), g(x), \dots, k(x)\}$ 表示之。

与整数的最大公约数相似，可用同样的方法证明多项式的最大公因式有如下性质：

1° 由欧几里德除法可知，若 $\partial \cdot f(x) \geq \partial \cdot g(x)$ ，则

$$f(x) = g(x)q(x) + r(x) \quad 0 \leq \partial \cdot r(x) < \partial \cdot g(x) \quad \text{或} \quad r(x) = 0$$

可得 $(f(x), g(x)) = (g(x), r(x))$ 。

2° 由整数的欧几里德算法，同样有多项式的欧几里德算法：

$$(f(x), g(x)) = A(x)f(x) + B(x)g(x)$$

式中

$$0 \leq \partial \cdot A(x) < \partial \cdot g(x) - \partial \cdot (f(x), g(x))$$

$$0 \leq \partial \cdot B(x) < \partial \cdot f(x) - \partial \cdot (f(x), g(x))$$

该算法将在以后详细证明。

例 4.3 求 GF(2) 上两个多项式：

$$f(x) = x^9 + x^8 + x^7 + x^2 + x + 1 \quad g(x) = x^4 + x^3 + x + 1$$

的最大公因式，并把它们的最大因式用该两个多项式的组合表示。

由欧几里德除法：

$$f(x) = x^9 + x^8 + x^7 + x^2 + x + 1 = (x^5 + x^3 + x)(x^4 + x^3 + x + 1) + (x^3 + 1)$$

$$g(x) = x^4 + x^3 + x + 1 = (x + 1)(x^3 + 1)$$

所以

$$(f(x), g(x)) = x^3 + 1$$

$$\begin{aligned} (f(x), g(x)) &= x^3 + 1 = 1 \cdot f(x) + (x^5 + x^3 + x)(x^4 + x^3 + x + 1) \\ &= A(x)f(x) + B(x)g(x) \end{aligned}$$

式中， $A(x) = 1$ ， $B(x) = x^5 + x^3 + x$ 。 ■

若两个多项式的 $(f(x), g(x)) = 1$ ，则说明此两多项式互素。若 $f(x), g(x), \dots, k(x)$ 之中任两多项式的最大公因式均为 1，则称此组集合中的多项式两两互素。

四、最小(低)公倍式

定义 4.2.4 若 $[f(x), g(x)]$ 是同时能被 $g(x), f(x)$ 除尽的次数最低的首一多项式，则称 $[f(x), g(x)]$ 是 $f(x)$ 和 $g(x)$ 的**最小公倍式**，用 $[f(x), g(x)]$ 或 $\text{LCM}\{f(x), g(x)\}$ 表示之。

与整数的最小公倍数完全相似，多项式的最小公倍式也有如下性质：

1° $f(x), g(x), k(x)$ 多项式集合中，若两两互素，则 $[f(x), g(x), k(x)] = f(x)g(x)k(x)$ ；

2° $f(x), g(x)$ 的最小公倍式 $[f(x), g(x)]$ 能除尽 $f(x), g(x)$ 的一切公倍式；

3° $f(x)g(x) = f(x), g(x)$ 。

或

$$[f(x), g(x)] = \frac{f(x)g(x)}{(f(x), g(x))}$$

由上面的讨论可知, 由于 F_p 上多项式全体与整数全体均构成一可换环, 因此多项式的理论与整数的理论是完全平行的。多项式相当于一般的整数, 既约多项式相当于素数, 而常数(零次多项式)相当于整数中的 1, 它既不是既约多项式也不是可约多项式。多项式与整数的这种平行关系, 在下面讨论中还将看到。

五、多项式剩余类环与有限域的构造

如同整数的剩余类环一样, 以一个 F_p 上的多项式:

$$f(x) = f_0 + f_1x + \cdots + f_nx^n \quad \partial \circ f(x) > 0$$

为模的剩余类全体也构成一个环, 称为多项式剩余类环。

如同整数剩余类环的构造, 我们首先在 $F_p[X]$ 上寻找一个理想, 然后以此理想为模对全体多项式分类, 就得到一个剩余类环。

引理 4.2.1 $F_p[x]$ 上任一多项式 $f(x)$ 的一切倍式集合 $I_f(x)$ 组成一个理想。

证明 设

$$f(x) = f_0 + f_1x + \cdots + f_nx^n \quad \partial \circ f(x) > 0$$

要证明 $f(x)$ 的一切倍式集合 $I_f(x)$ 是一理想, 只要证明:

- (1) 对一切 $a(x), b(x) \in I_f(x)$, 有 $a(x) - b(x) \in I_f(x)$ 。
- (2) 对任意 $c(x) \in F_p[x]$, $a(x) \in I_f(x)$ 有 $c(x)a(x) \in I_f(x)$ 。

现证明如下:

- (1) 因为 $a(x), b(x) \in I_f(x)$, 所以

$$a(x) = a_1(x)f(x)$$

$$b(x) = b_1(x)f(x)$$

因而

$$a(x) - b(x) = f(x)(a_1(x) - b_1(x)) \in I_f(x)$$

$$(2) c(x)a(x) = c(x)a_1(x)f(x) = d(x)f(x) \in I_f(x)$$

由此可知 $I_f(x)$ 是一个理想。■

在 $I_f(x)$ 理想中, 称 $f(x)$ 为理想的生成元。以此理想为基础把 $F_p[x]$ 划分陪集, 下面将证明这些陪集全体构成了模 $f(x)$ 的剩余类环。

例 4.4 以 $F_2(x)$ 上的多项式 $f(x) = x^2 + 1$ 所构成的理想 $I_f(x) = \{0, x^2 + 1, x(x^2 + 1), \dots\}$ 为基础, 对 $F_2[x]$ 全体划分陪集:

| | | | | | |
|---------------|-------|---------------|---------------|--------------------|----------|
| $\bar{0}$: | 0 | $x^2 + 1$ | $x(x^2 + 1)$ | $(x + 1)(x^2 + 1)$ | \cdots |
| $\bar{1}$: | 1 | x^2 | $x^3 + x + 1$ | $x^3 + x^2 + x$ | \cdots |
| \bar{x} : | x | $x^2 + x + 1$ | x^3 | $x^3 + x^2 + x$ | \cdots |
| $\bar{1+x}$: | $1+x$ | $x^2 + x$ | $x^3 + 1$ | $x^3 + x^2$ | \cdots |

上面从理想的角度讨论了多项式剩余环的构造, 下面从模 $f(x)$ 的角度讨论多项式剩余类环的构造。

定义 4.2.5 若 $F_p[x]$ 中的两多项式 $a(x), b(x)$ 被同一多项式 $f(x)$ 除时有相同的余式:

$$\begin{aligned} a(x) &= q_1(x)f(x) + r(x) \\ b(x) &= q_2(x)f(x) + r(x) \quad 0 \leqslant \deg r(x) < \deg f(x) \end{aligned}$$

则称 $a(x)$ 、 $b(x)$ 关于模 $f(x)$ 同余, 记为

$$a(x) \equiv b(x) \pmod{f(x)}$$

可知, $a(x)$ 和 $b(x)$ 同余就是在以 $I_f(x)$ 为理想的陪集划分中处于同一陪集之内。由于余式 $r(x)$ 的次数小于 $\deg f(x)$, 因此所有次数小于 $\deg f(x) = n$ 次多项式全体: $\{r_i(x)\}$, $i = 0, 1, \dots, n-1$, 必处于不同的剩余类或陪集中, 且在陪集中它们的次数为最低。若以次数最低的多项式 $r_i(x) \in \{r_i(x)\}$ 作为该陪集的代表元, 则此陪集可记为 $\overline{r_i(x)}$, 即

$$\overline{r_i(x)} = r_i(x) + I_f(x)$$

定理 4.2.7 $a(x) \equiv b(x) \pmod{f(x)}$ 的充要条件是:

$$a(x) - b(x) = q(x)f(x) \quad \text{或} \quad f(x) | a(x) - b(x)$$

证明 若 $a(x) \equiv b(x) \pmod{f(x)}$, 则 $a(x)$ 与 $b(x)$ 属于同一剩余类中, 设属于 $b(x)$ 的剩余类中:

$$a(x) \in \overline{b(x)} = b(x) + I_f(x)$$

或

$$a(x) = b(x) + f(x)q(x) \quad q(x) \in f_p(x)$$

所以

$$a(x) - b(x) = f(x)q(x) \quad \text{或} \quad f(x) | a(x) - b(x)$$

反之, 若

$$a(x) - b(x) = f(x)q(x)$$

则

$$a(x) = b(x) + f(x)q(x) \in b(x) + I_f(x) = \overline{b(x)}$$

所以

$$a(x) \equiv b(x) \pmod{f(x)}$$

定理 4.2.8 以 $f(x)$ 为模的多项式剩余类全体, 组成一个有单位元的可换环 $F_p[x]/f(x)$ 。

为了证明此定理, 我们首先规定两个陪集或剩余类之间的运算如下。

(1) 加法

$$\overline{a(x)} + \overline{b(x)} = \overline{a(x) + b(x)}$$

因为

$$\overline{a(x)} = a(x) + I_f(x) \quad \overline{b(x)} = b(x) + I_f(x)$$

所以

$$\begin{aligned} \overline{a(x)} + \overline{b(x)} &= a(x) + I_f(x) + b(x) + I_f(x) \\ &= a(x) + b(x) + I_f(x) = \overline{a(x) + b(x)} \end{aligned}$$

(2) 乘法

$$\begin{aligned} \overline{a(x)} \overline{b(x)} &= (a(x) + I_f(x))(b(x) + I_f(x)) \\ &= a(x)b(x) + I_f(x) = \overline{a(x)b(x)} \end{aligned}$$

证明

(1) 加法组成阿贝尔群。设 $\overline{a(x)}, \overline{b(x)}$ 是 $F_p[x]/f(x)$ 中的任两陪集，由上述规定的加法运算可知：

$$\overline{a(x)} + \overline{b(x)} = \overline{a(x) + b(x)} \in F_p[x]/f(x)$$

封闭性成立。

$$\overline{a(x)} + \overline{0} = \overline{0 + a(x)} = \overline{a(x)}$$

$$\overline{a(x)} + \overline{(-a(x))} = \overline{a(x) - a(x)} = \overline{0}$$

加法恒等元与逆元存在。结合律与交换律显然成立，故组成一个阿贝尔加群。

(2) 乘法封闭、可换。

$$\overline{a(x)} \cdot \overline{b(x)} = \overline{a(x)b(x)} = \overline{b(x)a(x)} = \overline{b(x)} \cdot \overline{a(x)} \in F_p[x]/f(x)$$

(3) 单位元为 $\overline{1}$ 。

(4) 乘法与加法间有分配律。

由此可知 $F_p[x]/f(x)$ 是一个有单位元的交换环，称为多项式剩余类环，它与整数剩余类环在结构和性质上完全相似。

例 4.4 中的四个陪集 $\overline{0}, \overline{1}, \overline{x}, \overline{x+1}$ 就构成了一个 $F_2[x]/(x^2 + 1)$ 剩余类环。该环的加法和乘法表如下：

| $+$ | $\overline{0}$ | $\overline{1}$ | \overline{x} | $\overline{x+1}$ |
|------------------|------------------|------------------|------------------|------------------|
| $\overline{0}$ | $\overline{0}$ | $\overline{1}$ | \overline{x} | $\overline{x+1}$ |
| $\overline{1}$ | $\overline{1}$ | $\overline{0}$ | $\overline{x+1}$ | \overline{x} |
| \overline{x} | \overline{x} | $\overline{x+1}$ | $\overline{0}$ | $\overline{1}$ |
| $\overline{x+1}$ | $\overline{x+1}$ | \overline{x} | $\overline{1}$ | $\overline{0}$ |

| $+$ | $\overline{0}$ | $\overline{1}$ | \overline{x} | $\overline{x+1}$ |
|------------------|----------------|------------------|------------------|------------------|
| \times | $\overline{0}$ | $\overline{1}$ | \overline{x} | $\overline{x+1}$ |
| $\overline{0}$ | $\overline{0}$ | $\overline{0}$ | $\overline{0}$ | $\overline{0}$ |
| $\overline{1}$ | $\overline{0}$ | $\overline{1}$ | \overline{x} | $\overline{x+1}$ |
| \overline{x} | $\overline{0}$ | \overline{x} | $\overline{1}$ | $\overline{x+1}$ |
| $\overline{x+1}$ | $\overline{0}$ | $\overline{1+x}$ | $\overline{x+1}$ | $\overline{0}$ |

从该表看到加法有逆元、恒等元，但对乘法而言 $\overline{x+1}$ 元素就没有逆元，因此 $F_p[x]/(x^2 + 1)$ 只是一个有单位元的可换环，而不能组成域。下面定理说明组成域的条件。

定理 4.2.9 若 $n (> 0)$ 次首一多项式 $f(x)$ 在域 $GF(p)$ 上既约，则由 $f(x)$ 为模所组成的多项式剩余类环是一个有 p^n 个元素的有限域 $GF(p^n)$ 。

证明 用 $f(x)$ 去除任一多项式，所得余式 $r(x)$ 的次数低于 $\deg f(x) = n$ ，因此所有次数低于 n 次多项式必处在不同的剩余类中。所以，下面只要证明每一个次数小于 n 次的多项式 $g(x)$ 都有逆元即可。

$f(x)$ 为既约, 因此对任一 $g(x) \in F_p[x]$ 必有 $g(x) \nmid f(x)$, 或 $(f(x), g(x)) = 1$ 。由欧几里德算法可知, 一定存在有 $a(x), b(x) \in F_p[x]$, 使

$$(f(x), g(x)) = 1 = a(x)f(x) + b(x)g(x)$$

所以

$$1 \equiv b(x)g(x) \pmod{f(x)}$$

或

$$\bar{1} = \overline{b(x)g(x)} = \overline{b(x)} \cdot \overline{g(x)}$$

因此 $\overline{b(x)}$ 是 $\overline{g(x)}$ 的逆元。

由于以 n 次多项式 $f(x)$ 为模的剩余类, 均可由次数小于 n 的多项式作代表。此代表元的一般表示式可写为

$$a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1x + a_0$$

而每一个 $a_i \in F_p$, 有 p 种不同的取值, 因此随着 a_i 的不同取值 ($i=0, 1, 2, \dots, n-1$), 共可得到 p^n 个次数低于 n 次的多项式, 它们均处在不同的陪集或剩余类中, 也就是说共有 p^n 个剩余类, 它们就构成一个有 p^n 个元素的有限域 $GF(p^n)$ 。 ■

例 4.5 $GF(2)$ 上的既约多项式 $f(x) = x^2 + x + 1$, 以它为模可产生一个有 2^2 个元素的 $GF(2^2)$ 有限域, 该域的 4 个元素为: $\bar{0}, \bar{1}, \bar{x}, \overline{x+1}$, 它们之间的加法乘法运算如下表:

| + | $\bar{0}$ | $\bar{1}$ | \bar{x} | $\overline{x+1}$ |
|------------------|------------------|------------------|------------------|------------------|
| $\bar{0}$ | $\bar{0}$ | $\bar{1}$ | \bar{x} | $\overline{x+1}$ |
| $\bar{1}$ | $\bar{1}$ | $\bar{0}$ | $\overline{x+1}$ | \bar{x} |
| \bar{x} | \bar{x} | $\overline{x+1}$ | $\bar{0}$ | $\bar{1}$ |
| $\overline{x+1}$ | $\overline{x+1}$ | \bar{x} | $\bar{1}$ | $\bar{0}$ |

| \times | $\bar{0}$ | $\bar{1}$ | \bar{x} | $\overline{x+1}$ |
|------------------|-----------|------------------|------------------|------------------|
| $\bar{0}$ | $\bar{0}$ | $\bar{0}$ | $\bar{0}$ | $\bar{0}$ |
| $\bar{1}$ | $\bar{0}$ | $\bar{1}$ | \bar{x} | $\overline{x+1}$ |
| \bar{x} | $\bar{0}$ | \bar{x} | $\overline{x+1}$ | $\bar{1}$ |
| $\overline{x+1}$ | $\bar{0}$ | $\overline{1+x}$ | $\bar{1}$ | \bar{x} |

由此表看出每个非 0 元素均有逆元, 故 $\bar{0}, \bar{1}, \bar{x}, \overline{x+1}$ 组成了 $GF(2^2)$ 有限域。以既约多项式为模所构成的剩余类, 在编码理论中有特别重要的意义。 ■

六、多项式主理想环

定理 4.2.1 证明了域 F_p 上的所有多项式全体 $F_p[x]$, 是一个有单位元的可换环, 引理 4.2.1 说明了该多项式环上理想的构造。下面讨论主理想的构造。

引理 4.2.2 多项式环 $F_p[x]$ 中的一切理想均是主理想。

证明 设 $I_f[x]$ 是 $F_p[x]$ 的任一理想。若 $I_f[x]$ 仅由零多项式构成，它显然是主理想。若 $I_f[x]$ 中含有非零多项式，则其中必含有一个次数最低的多项式 $f(x)$ 。设 $a(x)$ 是 $I_f[x]$ 中任一多项式，由欧几里德除法可知：

$$a(x) = q(x)f(x) + r(x) \quad 0 \leq \deg r(x) < \deg f(x) \quad \text{或} \quad r(x) = 0$$

由理想定义可知：

$$r(x) = a(x) - q(x)f(x) \in I_f[x]$$

由于 $f(x)$ 是理想中次数最低的多项式，且由 $\deg r(x)$ 的次数或者等于 0 或者小于 $f(x)$ 。可知， $r(x)$ 的次数只能等于 0，且由无 0 多项式的假设可知 $r(x) = 0$ 。因此， $a(x) = q(x)f(x)$

$$\in I_f[x].$$

引理 4.2.2 中的 $f(x)$ 为主理想的生成元，或生成多项式。当假定生成多项式为首一多项式时，多项式理想中的生成多项式显然是唯一的。

由引理 4.2.1 和引理 4.2.2 可知， $F_p[x]$ 不仅是一个有单位元可换环，而且也是一个**主理想环**。

现在我们讨论多项式剩余类环 $F_p[x]/f(x)$ 。由例 4.4 可看到，这种环可能有零因子，事实上，当 $f(x)$ 为可约时， $f(x) = g_1(x)g_2(x)$, $\deg g_1(x) > 0$, $\deg g_2(x) > 0$, 则 $\overline{f(x)} = \overline{0} = \overline{g_1(x)}\overline{g_2(x)}$, 即 $\overline{g_1(x)}$ 与 $\overline{g_2(x)}$ 为零因子。所以，当 $f(x)$ 为可约时，剩余类环 $F_p[x]/f(x)$ 中必有零因子，因此不满足主理想环的条件。如果我们放弃这一条件，则有以下定理。

定理 4.2.10 多项式剩余类环 $F_p[x]/f(x)$ 中每一理想皆为主理想，且该主理想的生成多项式必除尽 $f(x)$ 。

证明 该定理前半部分证明完全类似于引理 4.2.2 的证明，这里不再重复。设 I 是 $F_p[x]/f(x)$ 中任一理想， $g(x)$ 是它的生成多项式。

下面证明 $g(x)$ 必能除尽 $f(x)$ 。事实上，由欧几里德除法 $f(x) = q_1(x)g(x) + r_1(x)$, $0 \leq \deg r_1(x) < \deg g(x)$ 或 $r_1(x) = 0$, 可得：

$$\overline{0} = \overline{f(x)} = \overline{q_1(x)g(x)} + \overline{r_1(x)}$$

因 $\overline{0} = \overline{f(x)} \in I$, 故 $\overline{f(x)} - \overline{q_1(x)g(x)} = \overline{r_1(x)} \in I$, 由于 $g(x)$ 是理想中次数最低的多项式，必有 $r_1(x) = 0$ ，因此 $f(x) = q_1(x)g(x)$, 即 $g(x) | f(x)$ 。

由此可知，多项式剩余类环 $F_p[x]/f(x)$ 也是一个**多项式主理想环**，若不考虑无零因子这一条件的话。

七、同构与自同构

由上面讨论看到，以某一素数 p 为模的剩余类，和以某一 F_p 上的既约多项式 $p(x)$ 为模的多项式剩余类之间，有几乎完全相同的结构和性质。一般，有时两个集合之间的元素不但有一一对应的关系，而且有完全相同的性质和结构。例如， F_p 上的 $n-1$ 次多项式与 F_p 上的 n 重之间的每个元素，均存在一一对应关系。例如， $GF(2)$ 上的二次多项式与 $GF(2)$ 上的三重，它们元素之间有如下一一对应关系：

| 二次多项式 | 三重 |
|-----------------|-----|
| $0 + 0x + 0x^2$ | 000 |
| $1 + 0x + 0x^2$ | 100 |
| $0 + x + 0x^2$ | 010 |
| $1 + x + 0x^2$ | 110 |
| $0 + 0x + x^2$ | 001 |
| $1 + 0x + x^2$ | 101 |
| $0 + x + x^2$ | 011 |
| $1 + x + x^2$ | 111 |

这两个集合不仅元素之间有一一对应关系，而且在适当定义运算之后具有同样的性质与结构，称具有这种对应关系的这两个集合为同构。

定义 4.2.6 设 $G = \{1, a, b, \dots\}$ 为乘(加)群，而 G_a 是一个定义了乘(加)法运算的集合(不必封闭)， G 中乘(加)法用 \circ 表示， G_a 中乘(加)法用 $*$ 表示。若在 G 与 G_a 间建立了一一对应关系 α ，把 G 中元素 $\{1, a, b, \dots\}$ 变成 G_a 中元素 $\{1_a, a_a, b_a, \dots\}$ ，且对任意 $a, b \in G$ 有

$$\alpha(a \circ b) = \alpha(a) * \alpha(b)$$

称群 G 与集合 G_a 为群同构。

由此可知，群同构是 G 与 G_a 之间的一种一一对应关系，对任何 G 中的元素 a ，恒有 $a \xrightarrow{\alpha} a_a$ ，或 $\alpha(a) = a_a$ ， $a \circ b \xrightarrow{\alpha} a_a * b_a$ ，即不仅元素之间有一一对应关系，而且运算结果也一一对应。

例如，上表中，若设 $a = 1 + x^2$ ，则 $\alpha(a) = 101$ ， $b = 1 + x$ ，则 $\alpha(b) = 110$ 。

$$\begin{aligned} a \circ b &= a + b = 1 + x^2 + 1 + x = x + x^2 \\ \alpha(a) + \alpha(b) &= (101) + (110) = 011 \end{aligned}$$

而

$$\alpha(a \circ b) = \alpha(a + b) = \alpha(x + x^2) = 011$$

所以

$$\alpha(a) + \alpha(b) = \alpha(a \circ b) = \alpha(a + b)$$

因此两个同构的集合认为是一样的，研究其中一个也就代替了对另一个的研究。下面的定理说明了这点。

定理 4.2.11 假定群 G 与 G_a 同构，则 G_a 亦为群。

证明：

(1) 封闭性成立。设 $a, b \in G$ ， $\alpha(a) = a_a, \alpha(b) = b_a, a_a, b_a \in G_a$ 。由定义知， $\alpha(a \circ b) = \alpha(a) * \alpha(b) = a_a * b_a$ ，因为 $a \circ b \in G$ ，所以 $a \circ b$ 的像 $\alpha(a \circ b) = a_a * b_a$ 也必在 G_a 中，故 $a_a * b_a \in G_a$ 。

(2) 结合律成立。设 $a_a, b_a, c_a \in G_a$ ，则

$$\begin{aligned} (a_a * b_a) * c_a &= \alpha(a \circ b) * \alpha(c) = \alpha((a \circ b) \circ c) = \alpha(a \circ (b \circ c)) \\ &= \alpha(a) * \alpha(b \circ c) = a_a * (b_a * c_a) \end{aligned}$$

(3) 有单位元。 $\alpha(1) = 1_a$ 即为单位元。因为对任一 $a_a \in G_a$ ，有 $a_a * 1_a = \alpha(a \circ 1) = \alpha(a)$

$= a_a$ 。

(4) 有逆元。对任一个 $a_a \in G_a$, 令 $a_a^{-1} = a(a^{-1})$ 。所以 $a_a * a_a^{-1} = a(a) * a(a^{-1}) = a(a \circ a^{-1}) = a(1) = 1_a$ 。

同群一样，在环和域等代数系统中也可建立同构。且也可证明若 G 是环或是域，则 G_a 也是环或域；当然在这种情况下，必须建立两种运算(加和乘)的对应关系。

若 $G=G_a$, 则称为自同构，称为群 G 到自身的映射。因此，自同构是集合自身到自身的映射。

在数学上我们把同构的系统看成是本质上完全相同的，它可以把表面上看来似乎是不相干的事情，从本质上找到它们的内在联系。例如， $GF(q)$ 上的 n 重与 $n-1$ 次多项式，是两个完全不同的集合，但本质上它们完全一样。

§ 4.3 循 环 群

有限域在编码理论中起着非常重要的作用。由第二章可知，所谓有限域就是域中元素的个数是有限的，元素的个数称为域的阶，若为 p 阶有限域，用 $GF(p)$ 或 F_p 表示。而循环群的概念则是了解有限域的基础，它在编码理论中，特别在循环码理论中起关键作用。

一、循环群的基本概念

在定义循环群以前，先看以下例子。

例 4.6 考虑以下复数域上的三次方程。

$$x^3 - 1 = 0$$

它在复数域上有 3 个根：

$$\begin{aligned}x_0 &= 1 \\x_1 &= e^{(2\pi/3)i} = \cos(2/3)\pi + i \sin(2/3)\pi \\x_2 &= e^{(4\pi/3)i} = \cos(4/3)\pi + i \sin(4/3)\pi\end{aligned}$$

显然，若取其中某一根 $x_1 = e^{(2\pi/3)i}$ ，则这 3 个根可通过这一个根的各次幂表示。

$$\begin{aligned}x_0 &= x_1^0 = 1 \\x_1 &= x_1^1 = e^{(2\pi/3)i} \\x_2 &= x_1^2 = e^{(4\pi/3)i}\end{aligned}$$

事实上，上述 3 个根 x_0, x_1, x_2 构成阿贝尔乘群：单位元为 x_0 ，封闭性成立，且 x_1, x_2 互为逆元，交换律显然满足。像这种群中的所有元素都可由一个元素的连续运算(在乘群为幂次表示)而产生的群，称为循环群，由此引入循环群的定义。

定义 4.3.1 由一个单独元素的一切幂次所构成的群称为循环群，该元素称为循环群的生成元。

例 4.7 整数全体关于加法构成循环群，它的生成元是 1。例如： $5 = 1 + 1 + 1 + 1 + 1$, $6 = 1 + 1 + 1 + 1 + 1 + 1$, …，这里所指的元素 1 的幂次是指加法运算而言。

$$\begin{aligned}1^5 &= 1 + 1 + 1 + 1 + 1 = 5 \\1^6 &= 1 + 1 + 1 + 1 + 1 + 1 = 6\end{aligned}$$

⋮

由于整数群是一个无限群，故这是一个**无限循环群**。 ■

例 4.8 模整数 m 的剩余类全体关于加法所构成的群，是一个有限循环群，其生成元为 1。如 $m=4$ ，则 $\{\bar{0}, \bar{1}, \bar{2}, \bar{3}\}$ 中的 4 个元素可由 $\bar{1}$ 生成：

$$\bar{1} = \bar{1}$$

$$\bar{2} = \bar{1} + \bar{1} = \overline{1+1} = \bar{2} = \bar{1}^2$$

$$\bar{3} = \bar{1} + \bar{1} + \bar{1} = \overline{1+1+1} = \bar{3} = \bar{1}^3$$

$$\bar{0} = \bar{1} + \bar{1} + \bar{1} + \bar{1} = \overline{1+1+1+1} = \bar{0} = \bar{1}^4$$

该群中的 4 个元素不仅可由 $\bar{1}$ 生成，而且也可由 $\bar{3}$ 元素生成：

$$\bar{3}^3 = \bar{1} = \bar{3} + \bar{3} + \bar{3} = \overline{3+3+3} = \bar{9} = \bar{1}$$

$$\bar{3}^2 = \bar{2} = \bar{3} + \bar{3} = \overline{3+3} = \bar{6} = \bar{2}$$

$$\bar{3}^1 = \bar{3} = \bar{3}$$

$$\bar{3}^4 = \bar{0} = \bar{3} + \bar{3} + \bar{3} + \bar{3} = \overline{3+3+3+3} = \bar{12} = \bar{0}$$
 ■

由此可知，循环群的生成元不止一个。在例 4.6 中 x_1, x_2 都是生成元，但 x_0 不是。在例 4.7 中，整数循环群的生成元只有一个。

由于

$$a^{m1}a^{m2} = a^{m1+m2} = a^{m2+m1} = a^{m2}a^{m1} \quad m1, m2 \text{ 为整数}$$

因此，凡是循环群都是可换群。

二、循环群的构造及其性质

设 a 是循环群中的任一个元素，考虑 a 的一切幂 a^n ，这时可能有两种情况：

(1) a 的所有幂次 a^h ($h=0, \pm 1, \pm 2, \pm 3, \dots$) 均不相同，这时由 a 生成的群 $G(a)=\{\dots a^{-2}, a^{-1}, a^0, a^1, a^2, \dots\}$ 中，元素的个数无限，称为**无限循环群**。

(2) a 的某二次幂相同，也就是存在有整数 h, k (且 $h>k$) 使 $a^h=a^k$ ，于是有

$$a^h a^{-k} = a^k a^{-k} = e$$

$$a^{h-k} = e \quad h - k \text{ 为正整数}$$

于是，群 $G(a)$ 的元素为： $\{e, a^1, a^2, \dots, a^{n-h-k}=e, a^1, \dots\}$ 。群中元素个数有限，所以是**有限循环群**。称 $a^n=e$ 的最小正整数 n 为有限循环群元素 a 的级。若为无限循环群，则 a 是无限级的。在有限循环群中，有以下特点：

1° $a^0=e, a, a^2, \dots, a^{n-1}$ 均互不相同。若相反， $a^h=a^k, 0 \leqslant k < h < n$ ，则

$$a^{h-k} = e \quad 0 \leqslant h - k = n_1 < n$$

于是有

$$a^{n_1} = a^{h-k} = e \quad 0 \leqslant n_1 < n$$

这与假设 n 是使 $a^n=e$ 的最小正整数相矛盾。

2° a 为 n 级元素，即 $a^n=e$ ，则 a 的一切幂次生成的元素都在 $G(a)=\{a^0=e, a^1, a^2, \dots, a^{n-1}\}$ 中。

设 m 为任意整数，由欧几里德除法有

$$m = qn + r \quad 0 \leqslant r < n$$

$$a^m = a^{m+r} = (a^n)^q \cdot a^r = e^q \cdot a^r = e \cdot a^r = a^r$$

而 $0 \leq r < n$, 所以 a 的一切幂均在 $G(a)$ 中。

3° 凡是循环群必是可换群。

定理 4.3.1 可换群 G 中的每一元素 a 皆能生成一个循环群, 它是 G 的子群。若 a 是无限级元素, 则生成的是无限循环群。如果 a 是有限级元素, 则生成的是有限循环群, 元素 a 的级就是有限循环群的阶数。

三、有限循环群中元素级的性质

1° 若 $a \in G$ 是 n 级元素, 则 $a^m = e$ 的充要条件是 $n | m$ (m 为整数)。

证明 若 $a^m = e$, 由欧几里德除法有

$$m = qn + r \quad 0 \leq r < n$$

则

$$a^m = e = a^{qn+r} = a^{qn} \cdot a^r = (a^n)^q \cdot a^r = e^q \cdot a^r = a^r$$

所以 $a^m = a^r = e$, 而 $0 \leq r < n$, 这与 n 是 a 的级相矛盾, 故 $r=0$, 由此 $m=qn$, $n | m$ 。

反之, 若 $n | m$, 则

$$a^m = a^{qn} = (a^n)^q = e^q = e$$

2° 设 $a, b \in G$, a 为 n 级元素, b 为 m 级元素, 且 $(n, m)=1$, 则 (ab) 之级等于 nm 。

证明

(1) 由假设可知:

$$(ab)^{mn} = (a^n)^m (b^m)^n = e^m e^n = e$$

(2) 由性质 1° 可知, 若 (ab) 之级为 k , 则 $k | mn$ 。下面证明 $k=mn$, 为此只要证明 $mn | k$ 。由假设: $(ab)^k = e$, 两边乘以 b^{-k} 可得:

$$a^k = b^{-k}$$

两边乘以 m 次方得

$$a^{mk} = b^{-mk} = (b^m)^{-k} = e^{-k} = e$$

又由 $(ab)^k = e$, 两边乘以 a^{-k} 可得 $b^k = a^{-k}$, 由此可知:

$$b^{nk} = a^{-nk} = (a^n)^{-k} = e^{-k} = e$$

所以, 由 $a^{mk} = e$, 可得 $n | mk$, 由于 $(n, m)=1$, 所以 $n | k$ 。由 $b^{nk} = e$ 可得 $m | nk$, 故 $m | k$ 。由于 m, n 均整除 k , 所以 k 必是 n 和 m 的公倍数。公倍数一定是最小公倍数的倍数:

$$k = [n, m]t = nm t$$

所以 $nm | k$ 。

3° 若 a 为 n 级元素, 则 a^k 元素之级为 $n/(k, n)$ 。

证明

(1) 由 $(a^k)^n = (a^n)^k = e^k = e$ 可知 a^k 的级为有限, 设它的级为 m 。由

$$(a^k)^{n/(k,n)} = (a^n)^{k/(k,n)} = e^{k/(k,n)} = e$$

并由 1° 可知 $m | (n/(k,n))$ 。

(2) 由 $a^{km} = (a^k)^m = e$, 且 a 为 n 级元素和 1° 可知, $n | km$ 。

显然, $k | km$, 所以 km 是 k 和 n 的公倍数, 公倍数一定是最小公倍数的倍数, 所以

$$km = [k, n]t$$

由最小公倍数的性质 3。可知 $kn = k, n$, 代入上式得

$$km = t(kn/(k, n))$$

两边消去 k 得

$$m = t(n/(k, n))$$

故 $(n/(k, n)) | m$, 由(1)知, $m = n/(k, n)$ 。

推论 4.3.1 若 a 为 dk 级元素, 则 a^k 为 d 级元素。

证明 $a^{dk} = (a^k)^d = e$

推论 4.3.2 n 阶循环群中, 每个元素的级是群阶数 n 的因子。

此推论直接由性质 3° 得到。

由性质 3° 可知, 群中每一元素 a^i 的级为 $n/(i, n)$, 若 $(n, i) = 1$, 则 a^i 元素的级亦为 n , 也就是说由 a^i 的幂次亦可生成循环群 $G(a)$ 中的全部元素:

$$\{(a^i)^0 = e, a^i, a^{2i}, \dots, a^{(n-1)i}\}$$

定义 4.3.2 n 阶循环群中, 每一个 n 级元素称为 n 次单位原根。

由上可知, 在 $G(a)$ 循环群中单位原根可以不止一个, 凡是 $(i, n) = 1$ 的 a^i 元素都是 $G(a)$ 的单位原根。下面讨论单位原根的个数。为此首先定义欧拉函数。

定义 4.3.3 $0, 1, 2, \dots, n-1$ 中与 n 互素的个数称为欧拉函数, 记为 $\varphi(n)$ 。

由数论可证明, 若 n 的标准分解式为

$$n = p_1^{a_1} p_2^{a_2} \cdots p_s^{a_s}$$

则

$$\varphi(n) = \prod_{i=1}^s \varphi(p_i^{a_i})$$

而

$$\varphi(p_i^{a_i}) = p_i^{a_i-1}(p_i - 1)$$

代入后得

$$\varphi(n) = \prod_{i=1}^s p_i^{a_i-1}(p_i - 1)$$

例如: $n = 12 = 3 \times 4 = 3 \times 2^2$, 则

$$\varphi(12) = 2^{2-1}(2 - 1)(3 - 1) \cdot 3^0 = 4$$

所以与 12 互素的个数有四个: 1, 5, 7, 11。

显然, 若 p 为素数, 则

$$\varphi(p) = p - 1$$

推论 4.3.3 n 阶循环群中有 $\varphi(n)$ 个单位原根。

四、由已知循环群寻找它的全部子群

定理 4.3.2 设 $G(a)$ 是由元素 a 生成的 n 阶有限循环群, 而 H 是它的子群。于是:

(1) H 仍为有限阶循环群。它或者是仅由单位元构成的群, 或者是由仅有最小正整数 m 的元素 a^m 所生成;

(2) 这种最小正整数 m 必是 n 的因子, 且子群的阶为 n/m ;

(3) 对于 n 的一个因子 $m (> 0)$, G 中必有且仅有一个阶数为 n/m 的循环子群。

证明

(1) 若 H 仅由单位元素所组成，则定理已证完，因为由单位元素所组成的子群仅有一个元素，故必是有限子群。

另一方面由子群的拉格朗日定理可知，有限群子群的阶数必是群阶数的因子，所以子群的元素为有限，现令 $m=n/q$ ，则由于 $(a^n)^q=a^{mq}=a^n=e$ ，因此 a^n 生成的子群必是 q 阶有限循环群，显然这个 m 是生成 q 阶循环群的最小 m ， $q=n/m$ 。

(2) 现证 $H=G(a^n)$ 是 $G(a)$ 中唯一的 $q=n/m$ 阶子群。若还有一个 n/m 阶子群 H' ，它由 $a^{m'}$ 生成，则由前面证明可知 m' 是生成 $q=n/m$ 阶循环群的最小正数，即 $m'=n/q$ ，因此 $m=m'$ ， $a^n=a^{m'}$ ， $H=H'$ 。 ■

若用 $d(n)$ 表示正整数 n 的因子个数，则由该定理可知， n 阶循环群共有 $d(n)$ 个有限循环子群。对于无限循环群，可以证明也有类似的结果。

§ 4.4 有限域 (Galois 域) 的乘法结构

由前面讨论可知，以素数 p 为模的整数剩余类环构成 p 阶有限域 $GF(p)$ 。以 F_p 上 m 次首一既约多项式 $f(x)$ 为模的多项式剩余类环构成 p^m 阶有限域，通常以 $GF(p^m)$ 表示。

由于有限域理论在编码理论研究中起着特别重要的作用，故下面将较详细地讨论它的乘法结构、加法结构和代数结构。

一、基本概念

域中的所有非 0 元素构成阿贝尔乘群，所有元素构成阿贝尔加群。若为有限域，则元素的个数为有限。因此，所有非 0 元素全体一定构成一个有限乘群，群中每个元素的级为有限，并由定理 4.3.1 可知它必由一个元素生成且是循环群。不仅如此，下面我们将证明在任何 q 阶有限域中能找到一个生成元素 α ，它的级为 $q-1$ ，能生成域中所有 $q-1$ 个非 0 元素，从而组成一个循环乘群 $G(\alpha)$ ： $1, \alpha, \alpha^2, \dots, \alpha^{q-2}, \alpha^{q-1} = 1$ 。

定义 4.4.1 域中非 0 元素所构成的乘法群之阶定义为域中该元素的级。

定义 4.4.2 若 α 为域 $GF(q)$ 中的 n 级元素，则称 α 为 n 次单位原根。若在 $GF(q)$ 中，某一元素 α 的级为 $q-1$ ，则称 α 为本原域元素。

由于 $GF(q)$ 中所有 $q-1$ 个非 0 元素组成一个乘群，因此本原域元素 α 能生成这个乘群，与循环群中的定义类似，显然有 $\alpha^{q-1}=e$ 。

定理 4.4.1 在 $GF(q)$ 中，每一个非 0 元素均满足 $x^{q-1}=1$ ，即都是 $x^{q-1}-1=0$ 方程的根。反之， $x^{q-1}-1=0$ 的根必在 $GF(q)$ 中。

证明 由定理 4.2.5 知， $x^{q-1}-1=0$ 方程至多有 $q-1$ 个根。下面证明 $GF(q)$ 中的所有 $q-1$ 个元素就是该方程的全部根。

$$x^q - x = x(x^{q-1} - 1) = (x - 0)(x^{q-1} - 1)$$

因此 $GF(q)$ 中的 0 元素是 $x^q - x = 0$ 方程的一个根。因为 $GF(q)$ 中的 $q-1$ 个非 0 元素组成一个循环乘群，它由级为 $q-1$ 的 α 的所有幕次组成：

$$\alpha^0 = 1, \alpha, \alpha^2, \dots, \alpha^{q-2}$$

因为 $1, \alpha, \alpha^2, \dots, \alpha^{q-2}$ 中的每一个均满足：

$$(\alpha^i)^{q-1} = (\alpha^{q-1})^i = 1 \quad i = 0, 1, 2, \dots, q-2$$

或

$$(\alpha^i)^{q-1} - 1 = 0 \quad i = 0, 1, 2, \dots, q-2$$

所以 $\text{GF}(q)$ 中的 $q-1$ 个元素都是 $x^{q-1}=0$ 方程的根。

反之，若 α 是方程 $x^{q-1}-1=0$ 的根，则 $\alpha^{q-1}=1$ ，因此 α 必是由 $\text{GF}(q)$ 中的本原域元素 β 的幂次生成，也就是 α 是 $\text{GF}(q)$ 中的元素。 ■

推论 4.4.1 由 $\text{GF}(q)$ 中 n 级元素 α 生成的循环群 $G(\alpha)$ ，一定是 $x^n-1=0$ 方程的根。

在上面曾谈到， $\text{GF}(q)$ 中的所有 $q-1$ 个非 0 元素组成一个循环群。换言之， $q-1$ 个非 0 元素一定可以由一个级为 $q-1$ 的本原域元素生成，下面定理说明了这点。

定理 4.4.2 $\text{GF}(q)$ 中必有本原域元素存在。

该定理作为习题，请读者证明。

下面对该定理作些说明。对于有限域而言，它的非 0 元素全体，不仅构成一个阿贝尔乘群，而且构成一个循环群，所以由推论 4.3.3 知， $\text{GF}(q)$ 中有 $\varphi(q-1)$ 个 $q-1$ 级的本原域元素，且 $q-1$ 个非 0 元素中的每一个元素的级，都是 $q-1$ 的因子。

由上讨论可知，若 α 是 $\text{GF}(q)$ 中的本原域元素，则可把 x^q-x 在域中完全分解成一次因式：

$$x^q - x = x \prod_{i=1}^{q-1} (x - \alpha^i)$$

称 $\text{GF}(q)$ 为 x^q-x 的最小完全分离(解)域。

二、分圆多项式

设 α 是 $\text{GF}(q)$ 的 n 级元素，则由循环群的性质可知， $G(\alpha) = \{1, \alpha, \alpha^2, \dots, \alpha^{n-1}\}$ 是一个由 α 生成的循环子群，由定理 4.4.1 及推论 4.4.1 可知， $x^n-1=0$ 方程的全部根，就是 $\text{GF}(q)$ 内的 $G(\alpha)$ 子群的元素。因而

$$x^n - 1 = (x - \alpha^0)(x - \alpha)(x - \alpha^2) \cdots (x - \alpha^{n-1}) = \prod_{i=0}^{n-1} (x - \alpha^i)$$

定理 4.4.3 在含有 n 次单位原根的任意域上，有下述因式分解：

$$x^n - 1 = \prod_{i=0}^{n-1} (x - \alpha^i)$$

证明 由于 $G(\alpha)$ 中的每一元素的级都是 n 的因子，且都是 $x^n-1=0$ 的根。反之，在 $\text{GF}(q)$ 中级为 n 和是 n 因子的元素也一定是 $x^n-1=0$ 的根。事实上我们作如下集合：

$$A = \{\text{使 } x^n - 1 = 0 \text{ 的 } \text{GF}(q) \text{ 中 } n \text{ 个元素的集合: } 1, \alpha, \alpha^2, \dots, \alpha^{n-1}\}$$

把 A 中的元素按级进行分类：

$$\{\beta_1 = \alpha^{i_1}\} \subset A, \text{ 该集合元素的级均为 } n/(i_1, n) = n_{i_1}$$

$$\{\beta_2 = \alpha^{i_2}\} \subset A, \text{ 该集合元素的级均为 } n/(i_2, n) = n_{i_2}$$

⋮

$$\{\beta_d = \alpha^{i_d}\} \subset A, \text{ 该集合元素的级均为 } n/(i_d, n) = n_{i_d}$$

$$\{\beta_1\} \cup \{\beta_2\} \cup \cdots \cup \{\beta_d\} = A$$

再作一个集合：

$$B = \{GF(q) \text{ 中所有级能整除 } n \text{ 的域元素}\}$$

A 集合中的元素其实就组成了 $G(\alpha)$ ，由于其中每一元素的级都整除 n ，所以 $A \subset B$ 。另一方面，在 $GF(q)$ 中所有能整除 n 元素也必在 A 中。因为对每一个 $\beta \in B$ ，则它的级 $n_\beta | n$ ，或 $n = k \cdot n_\beta$ 。所以， $\beta^n = \beta^{kn_\beta} = (\beta^{n_\beta})^k = e^k = e = 1$ 。因此， β 必是 $x^n - 1 = 0$ 方程的根，故 $\beta \in A$ 中，可知 $A = B$ 。 ■

所以，根据根的级数的不同，可以把 $x^n - 1$ 写成如下形式：

$$x^n - 1 = \prod_{i=0}^{n-1} (x - \alpha^i) = \prod_{d|n} \prod_{\substack{\beta \text{ 是 } GF(q) \text{ 中} \\ \text{的 } n_d \text{ 级元素}}} (x - \beta) = \prod_{d|n} \prod_{\substack{\beta \text{ 是 } GF(q) \text{ 中} \\ \text{的 } d \text{ 级元素}}} (x - \beta) = \prod_{d|n} Q^{(d)}(x) \quad (4.4.1)$$

式中， $Q^{(d)}(x)$ 是以 d 级元素为根的所有一次因式的乘积，称它为 d 级分圆多项式。

定义 4.4.3 以 $GF(q)$ 中彼此不同的 d 级元素为全部根的首一多项式，称为 **d 级分圆多项式**，记为 $Q^{(d)}(x)$ 。

定理 4.4.4 d 级分圆多项式 $Q^{(d)}(x)$ 的次数为 $\varphi(d)$ 。

证明 由元素级的性质可知，若 β 为 d 级元素，则 β^i 为 d 级元素的充要条件是 $(d, i) = 1$ 。因为 $Q^{(d)}(x)$ 是由级为 d 的元素作根的一次因式的乘积作成，而在 $G(\beta) = \{\beta, \beta^2, \beta^3, \dots, \beta^d\}$ ($G(\beta) \subset GF(q)$) 集合中，仅含有 $\varphi(d)$ 个与 d 互素的元素，也即仅含有 $\varphi(d)$ 个 d 级元素，所以 $Q^{(d)}(x)$ 是 $\varphi(d)$ 次多项式。 ■

例 4.9 分解 $GF(2)$ 上的 $x^{15} - 1$ 多项式。

$x^{15} - 1$ 共有 15 个根，由定理 4.4.1 知， $GF(16) = GF(2^4)$ 上的所有 15 个非 0 元素都是它的根。15 的因子有 1, 3, 5, 15，故在 $GF(2^4)$ 中的非 0 元素的级分别为 1, 3, 5, 15 级。由式(4.4.1)知：

$$x^{15} - 1 = \prod_{d|15} Q^{(d)}(x) = Q^{(1)}(x)Q^{(3)}(x)Q^{(5)}(x)Q^{(15)}(x)$$

$Q^{(1)}(x)$ 是以 1 级元素 $\alpha^0 = 1$ 为根

$Q^{(3)}(x)$ 是以 3 级元素 α^5, α^{10} 为根

$Q^{(5)}(x)$ 是以 5 级元素 $\alpha^3, \alpha^6, \alpha^9, \alpha^{12}$ 为根

$Q^{(15)}(x)$ 是以 15 级元素 $\alpha, \alpha^2, \alpha^4, \alpha^8, \alpha^7, \alpha^{14}, \alpha^{13}, \alpha^{11}$ 为根

所以

$$Q^{(1)}(x) = (x - 1)$$

$$Q^{(3)}(x) = (x - \alpha^5)(x - \alpha^{10})$$

$$Q^{(5)}(x) = (x - \alpha^3)(x - \alpha^6)(x - \alpha^9)(x - \alpha^{12})$$

$$Q^{(15)}(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^4)(x - \alpha^8)(x - \alpha^7)(x - \alpha^{14})(x - \alpha^{13})(x - \alpha^{11})$$

故

$$x^{15} - 1 = \prod_{d=1,3,5,15} Q^{(d)}(x) = \prod_{i=1}^{14} (x - \alpha^i) \quad \blacksquare$$

但到此为止并没有具体地找出 $Q^{(d)}(x)$ 是什么，即在 $GF(2)$ 上 $Q^{(d)}(x)$ 的表示式是什么。为了解决这个问题，下面引入 Möbius 反演公式，为此先介绍 Möbius 函数。

定义 4.4.4 定义

$$\mu(a) = \begin{cases} 0 & \text{若 } a \text{ 含有平方因子} \\ (-1)^k & \text{a 不含有平方因子 } a = p_1 \cdot p_2 \cdot \cdots \cdot p_k \\ 1 & a = 1 \end{cases}$$

为 Möbius 函数，上式中 p_1, p_2, \dots, p_k 为素数。

例如， $\mu(15) = (-1)^2 = 1$ ，因为 $15 = 3 \cdot 5$, $k = 2$ 。又如 $\mu(2) = (-1)$, $\mu(4) = 0$ 。
若 p 为素数，按定义 $\mu(p) = -1$ 。

定理 4.4.5

$$Q^{(n)}(x) = \prod_{\substack{d \\ d|n}} (x^d - 1)^{\mu(n/d)} = \prod_{\substack{d \\ d|n}} (x^{n/d} - 1)^{\mu(d)}$$

该定理的证明较复杂，这里暂且略去，读者若有兴趣可参阅现代代数书籍。由此定理可对 $Q^{(d)}(x)$ 再进行分解。

例如上例中我们对 $Q^{(3)}(x)$ 、 $Q^{(5)}(x)$ 、 $Q^{(15)}(x)$ 应用定理 4.4.5，可得到它们在 GF(2) 上的表示式：

$$\begin{aligned} Q^{(3)}(x) &= (x - 1)^{\mu(3)}(x^3 - 1)^{\mu(1)} = (x - 1)^{-1}(x^3 - 1) = x^2 + x + 1 \\ Q^{(5)}(x) &= (x - 1)^{\mu(5)}(x^5 - 1)^{\mu(1)} = (x - 1)^{-1}(x^5 - 1) = x^4 + x^3 + x^2 + x + 1 \\ Q^{(15)}(x) &= (x - 1)^{\mu(15)}(x^3 - 1)^{\mu(5)}(x^5 - 1)^{\mu(3)}(x^{15} - 1)^{\mu(1)} \\ &= (x - 1)(x^3 - 1)^{-1}(x^5 - 1)^{-1}(x^{15} - 1) \\ &= x^8 + x^7 + x^5 + x^4 + x^3 + x + 1 \\ &= (x^4 + x + 1)(x^4 + x^3 + 1) \end{aligned}$$

所以

$$\begin{aligned} x^{15} - 1 &= Q^{(1)}(x)Q^{(3)}(x)Q^{(5)}(x)Q^{(15)}(x) \\ &= (x - 1)(x^2 + x + 1)(x^4 + x^3 + x^2 + x + 1) \\ &\quad \cdot (x^8 + x^7 + x^5 + x^4 + x^3 + x + 1) \end{aligned}$$

应用定理 4.4.4 可以检验 $\partial \circ Q^{(3)}(x) = \varphi(3) = 2$, $\partial \circ Q^{(5)}(x) = \varphi(5) = 4$, $\partial \circ Q^{(15)}(x) = \varphi(15) = \varphi(3)\varphi(5) = 8$ ，这与上面求得的结果完全一致。

由此可知分圆多项式的概念，在分解 $x^n - 1$ 型的多项式中起着很重要的作用。

§ 4.5 有限域的加法结构

这一节主要讨论 GF(q) 有限域中，所有元素在相加运算下所具有的性质。

一、基本概念

在域中必有一个单位元 e ，若作 $e+e+e+\cdots$ 运算，对无限域来说，则有可能 $ne \neq 0$ ，但在有限域中， $e+e+\cdots+e=0$ ，否则必成为无限域了。例如，在 GF(2) 中， $1+1=0$ 。

定义 4.5.1 满足 $ne=0$ 的最小正整数 n ，称为域的特征。如果对于每一个 n ，恒有 $ne \neq 0$ ，则称该域的特征为 ∞ 。

由上面定义可知，GF(2) 的特征为 2, GF(p) 的特征为 p 。若域的特征为 n , $ne=n \cdot 1=0$ ，则对域中每一个非 0 元素 a ，均有 $n \cdot a=0$ 。因为

$$\begin{aligned}
na &= \underbrace{a+a+\cdots+a}_{n} = 1 \cdot a + 1 \cdot a + \cdots + 1 \cdot a \\
&= \underbrace{(1+1+\cdots+1)}_{n} \cdot a = n \cdot 1 \cdot a = 0 \cdot a = 0
\end{aligned}$$

从这可知，特征为 n 的域中任一元素连续相加 n 次后所得的结果均为 0。因此，如同有限域乘法运算中，每个元素 a 有级一样，在加法运算下，对每个元素可定义周期。

定义 4.5.2 对每一个 $a \neq 0, a \in GF(q)$ ，满足 $na=0$ 的最小正整数 n ，称为域元素 a 的周期。所以，域中的元素 a ，在加法和乘法运算下有如下形式：

乘法： $a, a^2, a^3, \dots, a^m=1, a, \dots, m$ 为 a 之级

加法： $a, 2a, 3a, \dots, na=0, a, \dots, n$ 为 a 之周期

因此对加法而言，域中的所有元素也构成循环阿贝尔加群。

定理 4.5.1 域中一切非 0 元素的周期都是相同的，等于域的特征。

证明

(1) 若域的特征为 ∞ ，则对任何 $a \in GF(p), a \neq 0$ ，恒有 $na \neq 0$ 。因为 $na=ne \cdot a, a \neq 0, e \neq 0$ 而 $ne \neq 0$ ，所以 $na \neq 0$ 。

(2) 若域的特征为 p ，则 $pe=0$ 。设 $a \neq 0$ ，且 $a \in GF(p)$ 中，又设 a 的周期为 n ，则由欧几里德除法可知：

$$n = pq + r \quad 0 \leq r < p$$

$$0 = na = apq + ar = a \cdot (pe) \cdot q + ar = a \cdot 0 \cdot q + ar = ar = a \cdot (re)$$

因为已知 $a \neq 0$ ，所以只有 $re=0$ ，而 $r < p$ ，这与 p 是域的特征矛盾，故 $r=0$ 。由此可知：

$$n = pq \quad p | n$$

另一方面，由欧几里德除法

$$p = nj + r \quad 0 \leq r < n$$

$$0 = pea = njea + rea = naje + rea = 0ej + rea = rea$$

而 $e \neq 0$ ，故 $ra=0, r < n$ ，这与 a 的周期为 n 的假设相矛盾，故 $r=0$ 。所以

$$p = nj \quad n | p$$

由此可知 $p=n$ 。■

从上定理得出如下结论：域中一切非 0 元素的特征都相同，且都等于域的特征。因此以后均称为域的特征，而不称为某一元素的特征。

由于对每一个 $a \in GF(p)$ 中，恒有

$$a, 2a, 3a, \dots, (p-1)a, pa = 0, a, \dots$$

因此，由 a 的连续相加就构成此域中的一个阿贝尔循环加群，共有 $p-1$ 个非 0 元素，一个 0 元素，恰好是 $GF(p)$ 中全部元素。由此可见，域的特征(或元素的周期)说明了域中加法运算的循环性，而域中元素的级说明了域中乘法运算的循环性。

定理 4.5.2 每一个域的特征或为素数，或为 ∞ 。

证明

(1) 设域的特征为 ∞ ，这是一个无限域，定理证完。

(2) 域的特征不为 ∞ ，设为 h 。若

$$h = m \cdot n, \text{ 且 } m, n < h$$

$$h \cdot e = (m \cdot n)e = (m \cdot e)(n \cdot e) = 0$$

域中无 0 因子，所以 $(m \cdot e)$ 与 $(n \cdot e)$ 中至少一个为 0，而 $m, n < h$ ，这与 h 是域的特征假设相矛盾，故 h 必为素数。 ■

应当注意，有限域元特征必为有限，但域的特征为有限时，可以是有限域也可以是无限域。

例如， $R[x] = \{f(x)/g(x)\}$, $f(x), g(x) \in F_p[x]$ 。 $R[x]$ 是一个域，它的单位元为 1，且 $p \cdot 1 = 0$ ，所以域的特征为 p 。但多项式 $f(x)$ 或 $g(x)$ 的次数可以是无限的，故 $R[x]$ 中的元素个数是无限的，因此这是一个有限特征的无限域。

定理 4.5.3 在 p 特征域中，域整数全体（即形如 ne 全体的域元素： $n=0, \pm 1, \pm 2, \dots$ ）构成 p 阶素子域，并且这一素子域同构于模 p 的整数域 $GF(p)$ 。

证明

(1) 所谓素子域就是本身不再含有真子域的域，是最小的子域。

(2) 由于域的特征为 p ，故域整数只有下述元素：

$$R_p : \{0, 1, 2 \cdot 1, 3 \cdot 1, \dots, (p-1) \cdot 1\}$$

令 R_p 代表域整数的全体集合，并在 R_p 与素数剩余类域 $GF(p)$ 之间建立一一对应关系：

$$0 \longleftrightarrow \bar{0}, 1 \longleftrightarrow \bar{1}, \dots, (p-1) \cdot 1 \longleftrightarrow \overline{p-1}$$

显然，在这种对应下， R_p 与 $GF(p)$ 同构。因此， R_p 作为 $GF(p)$ 的同构的像，当然也是 p 阶有限域。至于 R_p 为素子域是显然的，因为不可能在 R_p 中找出更小的真子域。 ■

以 p 为特征的域是 $GF(p^m)$, $m = 1, 2, \dots$ ，称 $GF(p)$ 是 $GF(p^m)$ 的基域， $GF(p^m)$ 为 $GF(p)$ 的扩域。例如 $GF(2^2)$ 域是 $GF(2)$ 的扩域，因为 $GF(2^2)$ 中的 4 个元素都可用 $GF(2)$ 中的两个元素 0, 1 组合表示。

$$GF(2^2) : 00, 01, 10, 11$$

$$GF(2) : 0, 1$$

可以检验 $GF(2^2)$ 中每一个元素的特征均为 2：

$$11 + 11 = 00 \quad 01 + 01 = 00 \quad 10 + 10 = 00$$

所以 $GF(2^2)$ 是一个特征为 2 的域，它的二阶素子域由 00 和 10 两个元素组成。今后我们把 p 特征域中的 p 阶素子域记为 $GF(p)$ 。

定理 4.5.4 在特征为 p 的域中，恒有

$$(x - a)^p = x^p - a^p$$

式中， a 是域中的任一元素。

证明 由二项式定理可知：

$$\begin{aligned} (x - a)^p &= \sum_{k=0}^p \binom{p}{k} (-a)^k x^{p-k} \\ \binom{p}{k} &= \frac{p!}{k!(p-k)!} = \frac{p(p-1)(p-2)\cdots(p-k+1)(p-k)\cdots2 \cdot 1}{k!(p-k)(p-k-1)\cdots2 \cdot 1} \\ &= \frac{p(p-1)\cdots(p-k+1)}{k!} \end{aligned}$$

因为 p 为素数，且 $k < p$ ($k!, p = 1$, $k! \nmid p$)。令

$$\frac{(p-1)\cdots(p-k+1)}{k!} = n_1$$

则 $\binom{p}{k} = pn_1$ ($0 < k < p$)。因域的特征为 p , 故

$$\binom{p}{k}(-a)^k = pn_1(-a)^k = n_1(p(-a)^k) = 0$$

而

$$\binom{p}{0} = 1 \quad \binom{p}{p} = 1$$

所以

$$(x - a)^p = x^p + (-a)^p = x^p - a^p$$

推论 4.5.1 在 p 特征域中, 对任何域元素 a, b , 恒有

$$(a \pm b)^p = a^p \pm b^p$$

推论 4.5.2 在 p 特征域中, 任一元素的级均不是 p 的倍数。

由此及定理 4.4.2 知, 在 $GF(p^n)$ 中, 本原域元素的级是 $p^n - 1$, 而其它所有元素的级必是 $p^n - 1$ 的因子。在 $GF(2^n)$ 中, 本原域元素和其它一切元素的级均为奇数。

推论 4.5.3 若 w_1, w_2, \dots, w_k 是 p 特征域的元素, 则对于一切自然数 n , 恒有

$$(\sum_{i=1}^k w_i)^{p^n} = \sum_{i=1}^k w_i^{p^n}$$

推论 4.5.4 若 k 是 p 特征域的域整数, 则对一切自然数 n , 必满足以下方程:

$$x^{p^n} = x$$

即

$$k^{p^n} = k$$

证明 设 k 为域整数, 所以

$$k^{p^n} = (k \cdot e)^{p^n} = \underbrace{(e + e + \dots + e)}_k^{p^n} = \sum_{i=1}^k e^{p^n} = \sum_{i=1}^k e = ke = k$$

推论 4.5.5 对 $GF(p^n)$ 中的任何元素 w , 恒有 $w^{p^n} = w$ 。

证明 由定理 4.4.1 知, $GF(p^n)$ 中的所有元素都是 $x^{p^n} - x = 0$ 方程的根, 所以对 $w \in GF(p^n)$ 有

$$w^{p^n} - w = 0 \quad w^{p^n} = w$$

该推论就是著名的费尔马(Fermat)定理。

推论 4.5.6 由 Fermat 定理可知, 任何小于素数 p 的整数 b 满足 $b^p \equiv b \pmod{p}$ 。

例 4.10 在模 $p=7$ 的剩余类中, 有

$$2^6 \equiv 1 \pmod{7}$$

在模 $p=5$ 的剩余类中, 有

$$2^4 \equiv 1 \pmod{5}$$

定理 4.5.5 p 特征域中, 元素为域整数的充要条件是, 该元素是 $x^p - x = 0$ 方程的根。

证明 若 k 为 p 特征域中的域整数, 则由推论 4.5.4 可知, $k^p = k$, $k^p - k = 0$, 故 k 是 $x^p - x = 0$ 方程的根。

反之，若 k 是 $x^p - x = 0$ 方程的根，则 $k^p = k$ ，由定理 4.4.1 可知，它必在 $\text{GF}(p)$ 域中，所以 k 是域整数。 ■

二、最小多项式与本原多项式

下面讨论 $\text{GF}(p)$ 上多项式 $f(x) = \sum_{i=0}^k f_i x^i$ ($f_i \in \text{GF}(p)$) 的根在什么地方及其根的性质。

定理 4.5.6 设 $f(x) = \sum_{i=0}^k f_i x^i$, $f_i \in \text{GF}(p)$ 。若 p 特征域的元素 w 是方程 $f(x)$ 的根， $f(w) = 0$ ，则对于一切自然数 n , w^{p^n} 也必是 $f(x)$ 的根。

证明 由假设

$$f(w) = \sum_{i=0}^k f_i w^i = 0$$

$$0 = (f(w))^{p^n} = \left[\sum_{i=0}^k f_i w^i \right]^{p^n} = \sum_{i=0}^k f_i^{p^n} w^{ip^n} = \sum_{i=0}^k f_i^{p^n} (w^{p^n})^i$$

因 $f_i \in \text{GF}(p)$ ，由推论 4.5.4 可知 $f_i^{p^n} = f_i$ ，所以

$$0 = \sum_{i=0}^k f_i (w^{p^n})^i$$

故对一切自然数 n , w^{p^n} 是 $f(x)$ 的根。 ■

多项式 $f(x)$ 的次数 k 是有限的，因而方程在某一有限域上的根必有限，这表明在下述根序列： $w, w^{p^1}, w^{p^2}, w^{p^3}, \dots, w^{p^n}, \dots$ 中必有重复，也就是说在这一序列中各不相同的元素个数是有限的。由推论 4.5.5 可知，若 $w \in \text{GF}(p^m)$ ，则 $w^{p^m} = w$ ，因此

$$w, w^p, w^{p^2}, \dots, w^{p^{m-1}}, w^{p^m} = w$$

共 m 个，必是 $\text{GF}(p^m)$ 中互不相同的元素，它们都是 $f(x)$ 的 m 个不同的根。这 m 个根称为方程 $f(x)$ 的共轭根系。

由于 $w^{p^m} = w$, $w^{p^{m-1}} = 1$ 。因此，若 w 的级为 n ，则 $n | p^m - 1$ 。所以

$$p^m - 1 = nq \quad \text{或} \quad p^m = 1 + nq$$

可得

$$p^m \equiv 1 \pmod{n}$$

定义 4.5.3 能满足 $p^m \equiv 1 \pmod{n}$ 的最小整数 m ，称为 p 对模 n 的方次数。

定理 4.5.7 $\text{GF}(p)$ 上多项式 $f(x)$ 的共轭根系： $w, w^p, w^{p^2}, \dots, w^{p^{m-1}}$ 中的每一个均互不相同。

证明 用反证法。若 $w^{p^k} = w^{p^i}$, $k > i$, $k, i = 0, 1, 2, \dots, m-1$ ，则 $w^{p^k-p^i} = 1$ ，由此可得 w 的级 n 整除 $p^k - p^i$ ，即 $n | p^k - p^i$ 。因而

$$p^k - p^i \equiv 0 \pmod{n} \quad p^k \equiv p^i \pmod{n}$$

p 为素数， $(p, n) = 1$ ，故 $p^{k-i} \equiv 1 \pmod{n}$ 。因为 m 是 p 对模 n 的方次数，它是使 $p^m \equiv 1 \pmod{n}$ 成立的最小 m ，因而 $m | (k-i)$ 。但 i, k 均小于 m ，因此 $m \nmid (k-i)$ ，所以 $w^{p^k} \neq w^{p^i}$ 。 ■

如果 w 是 p 特征域上的 n 级元素， $w^n = 1$ ，则 w 是系数取自 $\text{GF}(p)$ 上的 $x^n - 1$ 的根。

系数取自 $\text{GF}(p)$ 上，且以 w 为根的多项式有许多个，其中必有一个次数最低的。

定义 4.5.4 系数取自 $\text{GF}(p)$ 上，且以 w 为根的所有首一多项式中，必有一个次数最低的，我们称它是 w 的**最小多项式**，记为 $m(x)$ 。

下面我们讨论最小多项式的性质。

定理 4.5.8 在 p 特征有限域中的每一个元素 w ，皆存在有唯一的最小多项式 $m(x)$ ，它具有以下性质：

1° $m(x)$ 在 $\text{GF}(p)$ 域上既约；

2° 若 $f(x)$ 也是 $\text{GF}(p)$ 域上的多项式，且 $f(w) = 0$ ，则 $m(x) | f(x)$ ，显然 $m(x) | (x^{p^m} - x)$ 。

证明

(1) 若 $m(x)$ 不是既约，则

$$m(x) = m_1(x)m_2(x) \quad \partial \circ m_1(x), \partial \circ m_2(x) < \partial \circ m(x)$$

$$m(w) = m_1(w)m_2(w) = 0$$

由定理 4.2.1 知， $\text{GF}(p)[x]$ 多项式环为无零因子环，所以必有 $m_1(w) = 0$ 或 $m_2(w) = 0$ ，而这与 $m(x)$ 是 w 的最小多项式的假设相矛盾，所以 $m(x)$ 是既约的。

(2) $f(x) \in \text{GF}(p)[x]$ ，则由欧几里德除法

$$f(x) = m(x)q(x) + r(x) \quad 0 \leqslant \partial \circ r(x) < \partial \circ m(x) \quad \text{或} \quad r(x) = 0$$

$$f(w) = m(w)q(w) + r(w) = 0$$

$m(w) = 0$, $m(w)q(w) = 0$ ，因此 $r(w) = 0$ ，这也与 $m(x)$ 是 w 的最小多项式相矛盾，故 $r(x) = 0$ ，所以

$$m(x) | f(x)$$

因为 $w \in \text{GF}(p^m)$ ，由推论 4.5.5 知， $w^{p^m} - w = 0$ ，故 w 也是 $x^{p^m} - x = 0$ 方程的根，所以 $m(x) | (x^{p^m} - x)$ 。

(3) 证明唯一性。若 $m_1(x)$ 也是 w 的另一最小多项式，则由性质(2)可知 $m_1(x) | m(x)$ ， $m(x)$ 是 w 的最小多项式，所以 $m(x) | m_1(x)$ ，因而 $m_1(x) = m(x)$ 。 ■

定理 4.5.9 设 w 是 p 特征有限域 $\text{GF}(p^m)$ 中的 n 级元素，而 m 是 p 关于模 n 的方次数，则 w 的最小多项式 $m(x)$ 是 m 次多项式，且

$$m(x) = \prod_{i=0}^{m-1} (x - w^{p^i})$$

证明 首先证明 $\prod_{i=0}^{m-1} (x - w^{p^i})$ 多项式在 $\text{GF}(p)[x]$ 中。为此令

$$f(x) = \prod_{i=0}^{m-1} (x - w^{p^i}) = \sum_{i=0}^{m-1} f_i x^i$$

由定理 4.5.4 及推论 4.5.5，有

$$\begin{aligned} [f(x)]^p &= \prod_{i=0}^{m-1} (x - w^{p^i})^p = \prod_{i=0}^{m-1} (x^p - w^{p^{i+1}}) \\ &= \prod_{i=1}^{m-1} (x^p - w^{p^i}) = \prod_{i=0}^{m-1} (x^p - w^{p^i}) = f(x^p) \end{aligned}$$

另一方面

$$[f(x)]^p = \left(\sum_{i=0}^m f_i x^i \right)^p = \sum_{i=0}^m f_i^p x^{pi}$$

$$f(x^p) = \sum_{i=0}^m f_i x^{pi}$$

因为以上两式左边相等，所以对每一个 i , $f_i^p = f_i \in GF(p)$, 所以 $f(x)$ 在 $GF(p)[x]$ 中。

由定理 4.5.6 知，对每一个系数取自 $GF(p)$ 的多项式，若以 w 为根，则必以 $w, w^p, w^{p^2}, \dots, w^{p^{m-1}}$ 为根，所以 m 次多项式

$$f(x) = \prod_{i=0}^{m-1} (x - w^{p^i})$$

是 $GF(p)$ 上以 w 为根，且次数为最低的首一多项式，于是 $m(x) = f(x)$ 。 ■

以后我们定义元素 w 的最小多项式的次数为 w 元素的次数，称 w 为 m 次域元素。

由此可见域元素 w 的次数，就等于 p 对模 n 的方次数， p 是域的特征， n 为 w 之级。

该定理说明 m 次域元素 w 的最小多项式 $m(x)$ ，在 $GF(p^m)$ 域中完全分解成一次因式的乘积，所以称 $GF(p^m)$ 为 $m(x)$ 的分离域或分解域。

由于 $w, w^p, w^{p^2}, \dots, w^{p^{m-1}}$ 是 $m(x)$ 的共轭根系，现在证明它们具有相同的级。

设 w 为 n 级，则 w^{p^i} 之级为 $n/(p^i, n)$ ，由于 p 是素数，且是域的特征，由推论 4.5.2 知， w 的级 n 不可能是 p 的倍数，故 $(p^i, n) = 1$ ，所以 w^{p^i} 也必为 n 级。由此得到如下推论。

推论 4.5.7 共轭根系内的每个元素 $w, w^p, w^{p^2}, \dots, w^{p^{m-1}}$ 的最小多项式均相同。

下面讨论在编码理论中起着非常重要作用的本原多项式。

定义 4.5.5 系数取自 $GF(p)$ 上的以 $GF(p^m)$ 中本原域元素为根的最小多项式，称为本原多项式。

所以本原多项式一定以 $n=p^m-1$ 级元素为根，设 w 为本原域元素，则以它为根的本原多项式的共轭根系是： $w, w^p, w^{p^2}, \dots, w^{p^{m-1}}$ ，共有 m 个根。因此，以 $GF(p^m)$ 上本原域元素为根的 $GF(p)$ 上的本原多项式，必是 m 次的。

下面定理说明有限域与最小多项式之间的关系。

定理 4.5.10 若 w 是 p 特征有限域 F 上的 m 次域元素，则 $GF(p)$ 上的次数小于 m 的、 w 的多项式全体 F_w ，构成域 F 上的 p^m 阶子域。

证明

(1) F_w 中 w 多项式全体两两互不相同。事实上，若 F_w 中有两个多项式相同：

$$\sum_{i=0}^{m-1} a_i w^i = \sum_{i=0}^{m-1} a'_i w^i \quad a_i, a'_i \in GF(p)$$

式中， a_i 与 a'_i 并不完全相同，则

$$\sum_{i=0}^{m-1} (a_i - a'_i) w^i = 0$$

这表明 $GF(p)$ 上的非 0 多项式

$$f(x) = \sum_{i=0}^{m-1} (a_i - a'_i) x^i$$

以 w 为根，但它的次数低于 m 次，而这与 w 为 m 次域元素的假设相矛盾。因此， F_w 中共含有 p^m 个多项式。

(2) 先证 F_w 是 F 上的一个交换子环。设 $f(w), g(w) \in F_w$, 显然 $f(w) - g(w) \in F_w$ 。其次, 设 $h(x) = f(x)g(x)$, 并设 $m(x)$ 是 w 的最小多项式, 则

$$h(x) = m(x)g_1(x) + r(x) \quad 0 \leqslant \deg r(x) < \deg m(x) \quad \text{或} \quad r(x) = 0$$

从而 $h(w) = m(w)g_1(w) + r(w) = r(w)$, 这说明 $r(w) \in F_w$ 。因此, $h(w) = f(w)g(w) \in F_w$, 故 F_w 构成交换环。

(3) F_w 含有单位元。显然, 单位元为 1, 就是原来域 F 中的单位元素。

(4) F_w 中的每一个非 0 元素皆有逆元。设 $g(w) \in F_w$, 且 $g(w) \neq 0$, 因 w 的最小多项式 $m(x)$ 是既约的, $(m(x), g(x)) = 1$, 故由欧几里德算法一定有 $a(x), b(x) \in GF(p)[x]$, 使下式成立:

$$1 = a(x)g(x) + b(x)m(x)$$

从而 $a(w)g(w) + b(w)m(w) = a(w)g(w) = 1$, 由欧几里德除法:

$$a(x) = m(x)g_2(x) + r(x) \quad 0 \leqslant \deg r(x) < \deg m(x) \quad \text{或} \quad r(x) = 0$$

因而 $a(w) = r(w)$, 由于 $a(w)$ 的次数小于 m , 故 $a(w) \in F_w$, 且 $a(w) = r(w)$, 即为 $g(w)$ 的逆元。

综上所述, F_w 构成一个 p^m 阶子域。 ■

例 4.11 $GF(2)$ 上多项式 $f(x) = x^3 + x + 1$, 它以特征为 2 有限域 $GF(2^m)$ 上的 3 次域元素 w 为根, 则 $GF(2)$ 上小于三次 w 多项式全体, 构成 2^3 阶子域: 0 (000), $w^2(001)$, $1+w+w^2(111)$, 1 (100), $1+w$ (110), $w+w^2(011)$, $w(010)$, $1+w^2(101)$ 。 ■

该定理指出, $GF(P^m)$ 中的每个元素都可用次数低于 m 的 w 多项式表示, w 是 m 次域元素。

显然, m 个元素集合 $\{1, w, w^2, \dots, w^{m-1}\}$ 是线性无关的。如果线性相关, 则定可找到一组系数 $(a_0, a_1, \dots, a_{m-1})$, 使 $\sum_{i=0}^{m-1} a_i \cdot w^i = 0$ 成立, 这里 $a_i \in GF(p)$, 可知 $a(x) = a_0 + a_1x + \dots + a_{m-1}x^{m-1}$ 必是 $GF(p)$ 上的以 α 为根的多项式。设以 α 为根的既约多项式 $f(x)$ 与 $a(x)$ 的最大公因式 $g(x) = (f(x), a(x))$, 可知, $g(\alpha) = 0$, 但由于 $f(x)$ 在 $GF(p)$ 上既约, 因而 $g(x) = 1$, 也就是 $f(x)$ 与 $a(x)$ 互素, 而这与 $a(x)$ 以 α 为根的假设相矛盾, 故这组元素集合线性无关。因而, 集合 $\{1, w, w^2, \dots, w^{m-1}\}$ 是 $GF(p^m)$ 中的一组基底, 通常称它为**自然(或本原)基底或基底元素**。由此可知, 可以由 $GF(p)$ 上的一个 m 次本原多项式或既约多项式, 用它的根 w 和这组基底的线性组合, 构造一个 $GF(p^m)$ 有限域。

$GF(p^m)$ 中的基底不是唯一的, 除了用自然基底表示域中的元素, 还可以用其它基底及其线性组合来表示, 这将在后面讨论。

三、互反多项式

定义 4.5.6 设 $GF(p)$ 上的 m 次多项式

$$f(x) = \sum_{k=0}^m a_k x^k = a_0 + a_1 x + \dots + a_m x^m \quad a_m \neq 0, a_0 \neq 0$$

则

$$f^*(x) = x^m \sum_{k=0}^m a_k x^{-k} = \sum_{k=0}^m a_k x^{m-k} = a_0 x^m + a_1 x^{m-1} + \dots + a_m$$

称为 $f(x)$ 的**互反多项式**。

如 GF(2) 上的三次多项式 $f(x) = x^3 + x + 1$ ，它的互反多项式。

$$f^*(x) = x^3(x^{-3} + x^{-1} + 1) = 1 + x^2 + x^3$$

互反多项式有如下性质：

性质 1 若 α 为 $f(x)$ 之根，则 α^{-1} 必为 $f^*(x)$ 之根。

性质 2 若 $f(x)$ 为既约，则 $f^*(x)$ 也为既约；反之亦然。

性质 3 $f(x)$ 为本原多项式，则 $f^*(x)$ 亦为本原多项式；反之亦然。

上述 3 个性质的证明较容易，请读者证明。

四、多项式的周期

定义 4.5.7 设 $f(x) \in F_p[x]$, $f(0) \neq 0$ (即 $x \nmid f(x)$)，则 $f(x) | (x^l - 1)$ 的最小正整数 l ，称为 $f(x)$ 的 **周期**(或**指数**)，记为 $p(f)$ 。

多项式的周期有以下几个性质：

性质 1 $f(x)$ 的周期 l 是以 $f(x)$ 为模所构成多项式剩余类环中乘法群内元素 $\bar{x} \pmod{f(x)}$ 之级。

证明 首先要证明在以 $f(x)$ 为模的多项式剩余类环 $F_p[x]/f(x)$ 中，与 $f(x)$ 互素的剩余类集合全体构成可换乘群。

设 $g(x)$ 与 $f(x)$ 互素，则

$$(f(x), g(x)) = 1 = g(x)A(x) + f(x)B(x)$$

所以， $g(x)A(x) \equiv 1 \pmod{f(x)}$ ，故 $g(x)$ 有逆元 $A(x)$ ，因此构成阿贝尔乘群，用 $F_p^*[x]/f(x)$ 表示。

由于 $f(0) \neq 0$ ，所以 x 与 $f(x)$ 互素，因而 $x \in F_p^*[x]/f(x)$ 。

由周期定义： $f(x) | (x^l - 1)$ 可得

$$x^l - 1 = f(x)q(x) \equiv 0 \pmod{f(x)}$$

$$\bar{x}^l \equiv 1 \pmod{f(x)}$$

所以， l 是 \bar{x} 之级。 ■

性质 2 $f(x) \in F_p[x]$, $f(0) \neq 0$ ，则

$$f(x) | x^l - 1 \text{ 的充要条件是 } p(f) | l$$

为证明此性质，先证明引理 4.5.1。

引理 4.5.1 多项式 $(x^m - 1) | (x^n - 1)$ 的充要条件是 $m | n$ 。

证明 若 $(x^m - 1) | (x^n - 1)$ 。由欧几里德除法

$$n = md + r \quad 0 \leq r < m$$

$$x^n - 1 = x^{md+r} - 1 = x^r \cdot x^{md} - 1 = x^r x^{md} + x^r - x^r - 1$$

$$= x^r(x^{md} - 1) + x^r - 1$$

$$\frac{x^n - 1}{x^m - 1} = \frac{x^r(x^{md} - 1)}{x^m - 1} + \frac{x^r - 1}{x^m - 1}$$

$$x^n - 1 = \frac{x^r(x^{md} - 1)}{x^m - 1}(x^m - 1) + x^r - 1$$

由假设 $(x^m - 1) | (x^n - 1)$ ，所以

$$x^n - 1 = q(x)(x^m - 1)$$

比较上两式可知 $x^r - 1 = 0$, $x^r = 1$, 故 $r=0$, 所以 $n=md$, $m \mid n$ 。

反之, 已知 $m \mid n$, 设 $n=md$, $x^m=y$, 则

$$x^n - 1 = x^{md} - 1 = y^d - 1$$

$y=1$ 是方程 $y^d - 1 = 1$ 的根, 所以, $(y-1) \mid (y^d - 1)$, 即 $(x^m - 1) \mid (x^n - 1)$ 。 ■

性质 2 证明: 若 $f(x) \mid (x^l - 1)$, 由周期定义可知 $f(x) \mid (x^{p(f)} - 1)$ 。所以

$$x^l - 1 = f(x)g(x) \equiv 0 \pmod{f(x)}$$

$$\bar{x}^l \equiv 1 \pmod{f(x)}$$

同理, $\bar{x}^{p(f)} \equiv 1 \pmod{f(x)}$ 。

由性质 1 可知, x 元素的级是 $p(f)$, 由级的性质可知 $p(f) \mid l$ 。

反之, 若 $p(f) \mid l$, 则由引理 4.5.1 可得 $(x^{p(f)} - 1) \mid (x^l - 1)$ 。因而 $f(x) \mid (x^l - 1)$ 。 ■

性质 3 若 $f(x)$ 是 $F_p[x]$ 中之 m 次既约多项式, 则 $f(x)$ 之周期 $p(f)$ 等于 $f(x)$ 在 $GF(p^m)$ 中的根的级。

证明 $f(x)$ 为 m 次既约多项式, 则 $F_p[x]/f(x)$ 多项式剩余类环变成为域。由定理 4.5.10 可知, 该域就是 $GF(p^m)$ 有限域, $f(x)$ 的根在 $GF(p^m)$ 中, 设为 $w, w^p, w^{p^2}, \dots, w^{p^{m-1}}$ 。所以

$$f(x) = (x - w)(x - w^p) \cdots (x - w^{p^{m-1}})$$

而 \bar{x} 显然是 $f(x)$ 的一个根, 设 $\bar{x}=w$, 故由性质 1 可知, $f(x)$ 的周期就是 $x=w$ 的级。 ■

性质 4 $GF(p)$ 上多项式 $f(x)$ 的标准分解式若为

$$f(x) = \prod_{i=1}^k f_i^{n_i}(x)$$

则 $f(x)$ 的周期

$$p(f) = n \cdot p^R$$

式中

$$n = \text{LCM}(n_1, n_2, \dots, n_k)$$

而 $n_i = p(f_i)$, $i=1, 2, \dots, k$ 。

$$R = \lceil \log_p(\max(m_1, m_2, \dots, m_k)) \rceil$$

$\lceil x \rceil$ 是 $\geq x$ 的最小整数。

例 4.12 $GF(2)$ 上的多项式

$$f(x) = (x^4 + x^3 + x^2 + x + 1)(x^4 + x + 1)$$

由例 4.9 可知 $x^4 + x^3 + x^2 + x + 1$ 以 5 级元素为根, $x^4 + x + 1$ 以 15 级元素为根。

$$R = \lceil \log_2 \max(1, 1) \rceil = 0$$

$$n = \text{LCM}(15, 5) = 15$$

所以

$$p(f) = 15 \cdot 2^0 = 15$$

由上述这些周期的性质, 可以得到有关本原多项式的一个重要定理。

定理 4.5.11 下述 4 个命题等价。

(1) $f(x)$ 为 $GF(p)$ 上 m 次本原多项式;

(2) $f(x)$ 为 $\text{GF}(p)$ 上 m 次既约多项式且 $p(f) = p^m - 1$;

(3) $f(x)$ 为 $\text{GF}(p)$ 上 m 次既约多项式, 且 $f(x) \nmid (x^{p^m-1} - 1)$, 而当 $r < p^m - 1$ 时, 则 $f(x) \mid (x^r - 1)$;

(4) $f(x)$ 为 $\text{GF}(p)$ 上 m 次既约多项式, 且 $f(x) \mid Q^{(p^m-1)}(x)$ 。

该定理的证明比较容易, 可以自行证明。

§ 4.6 有限域的代数结构与多项式的因式分解

在前面两节里, 我们分别讨论了有限域在乘法和加法运算下的结构及性质, 这一节将讨论有限域的代数结构及与此有关的 $x^n - 1$ 的因式分解。

定理 4.6.1 有限域的阶必为其特征(素数)之幂。

证明 设有限域的阶为 q , 特征为 p 。若 α 是域中的一个本原域元素, 则 α 的级为 $q-1$, 它的幂次生成了所有 $q-1$ 个非 0 元素。又设 m 是 p 关于模 $q-1$ 的方次数, 则

$$p^m \equiv 1 \pmod{q-1}$$

$$\alpha^{p^m} = \alpha \quad \text{或} \quad \alpha^{p^m-1} = 1$$

因 α 的级为 $q-1$, 所以 $(q-1) \mid (p^m - 1)$, 可知 $q \leq p^m$ 。

另一方面, 由 $\alpha^{p^m-1} = 1$ 可知, α 是 m 次域元素, 则由定理 4.5.10 可知, 系数取自 $\text{GF}(p)$ 上, 且次数低于 m 次的 α 多项式全体构成这个 q 阶有限域的 p^m 阶子域, 所以 $q \geq p^m$ 。比较上式得 $q = p^m$ 。■

定理 4.6.2 设 $f(x)$ 为 p 阶有限域 $\text{GF}(p)$ 上的一个 d 次既约多项式, 则多项式剩余集合 $F_p[x]/f(x)$ 构成 p^d 阶有限域 $\text{GF}(p^d)$ 。即 $\text{GF}(p^d)$ 是 $\text{GF}(p)$ 的扩域, 且 $f(x)$ 在 $\text{GF}(p^d)$ 内有根。

证明

(1) 由定理 4.2.9 可知, $F_p[x]/f(x)$ 是一个 p^d 阶有限域 $\text{GF}(p^d)$ 。

(2) 证明 $\text{GF}(p^d)$ 是 $\text{GF}(p)$ 的扩域, 设 $a \in \text{GF}(p)$, 让它对应 $\text{GF}(p^d)$ 中的 $a + 0 \cdot x + \dots + 0 \cdot x^{d-1} = \bar{a}$, 即

$$a \longleftrightarrow \bar{a}$$

这是 $\text{GF}(p)$ 与 $\text{GF}(p^d)$ 中的一个子集 $F = \{\bar{a} \in \text{GF}(p^d)\}$ 之间的一一对应。并且当 $a \longleftrightarrow \bar{a}$, $b \longleftrightarrow \bar{b}$ 时, 有:

$$a + b \longleftrightarrow \bar{a} + \bar{b} \quad a \cdot b \longleftrightarrow \bar{a} \cdot \bar{b}$$

因此, 这种对应是 $\text{GF}(p)$ 与 F 之间的同构对应。由于 $\text{GF}(p)$ 是域, 所以 F 也是域。如果 $\text{GF}(p)$ 与 F 视为同一, 则由于 $\text{GF}(p^d)$ 是 F 的扩域, 故 $\text{GF}(p^d)$ 也是 $\text{GF}(p)$ 的扩域。

(3) 设 $\alpha = \bar{x} \in F_p[x]/f(x)$,

$$f(x) = f_0 + f_1x + \dots + f_dx^d$$

则

$$f(\alpha) = f(\bar{x}) = f_0 + f_1\bar{x} + \dots + f_d\bar{x}^d = \overline{f_0 + f_1x + \dots + f_dx^d} = \overline{f(x)} = 0$$

这说明 $\alpha = \bar{x}$ 为 $f(x)$ 的根。■

定理 4.6.3

- (1) $\text{GF}(p^r)$ 含有子域 $\text{GF}(p^s)$ ($\text{GF}(p^r) \supset \text{GF}(p^s)$) 的充要条件是 $s|r$;
- (2) 若 $\beta \in \text{GF}(p^r)$, 则 $\beta \in \text{GF}(p^s)$ 中的充要条件是 $\beta^{p^s} = \beta$ 。特别是在任何域中若 $\beta^2 = \beta$, 则 β 是 0 或 1。

证明

(1) 由定理 4.4.1 知, $\text{GF}(p^r)$ 中所有非 0 元素都是 $x^{p^r-1} - 1 = 0$ 的根。 $\text{GF}(p^s)$ 中的所有非 0 元素是 $x^{p^s-1} - 1 = 0$ 方程的根。若 $\text{GF}(p^r) \supset \text{GF}(p^s)$, 这意味着 $(x^{p^s-1} - 1) | (x^{p^r-1} - 1)$ 。由引理 4.5.1 可知, 这意味着 $p^s - 1 | p^r - 1$, 再由引理 4.5.1 可得 $s|r$ 。

(2) 若 $\beta \in \text{GF}(p^r)$, $\beta \in \text{GF}(p^s)$ 。 β 在 $\text{GF}(p^r)$ 域中, 它的级 n 必是 $p^r - 1$ 的因子, $n|(p^r - 1)$ 。又因 β 在 $\text{GF}(p^s)$ 中, $n|(p^s - 1)$ 。所以

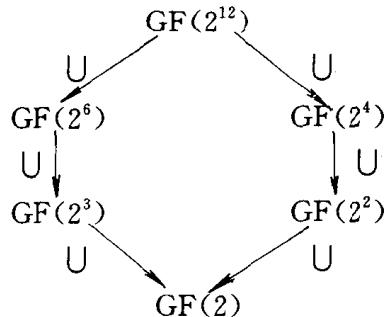
$$n = (p^s - 1, p^r - 1)$$

又由 (1) $(p^s - 1)|(p^r - 1)$, 所以 $n = p^s - 1$, 由此可知:

$$\beta^{p^s-1} = 1 \quad \beta^{p^s} = \beta$$

(3) 在任何域中只有单位元 $1^2 = 1$ 或 $0^2 = 0$ 。 ■

例 4.13 $\text{GF}(2^{12})$ 的子域如下图所示。



定理 4.6.4 q 阶有限域中的每一元素均满足方程

$$x^{q^n} - x = 0$$

n 为任意自然数。

定理 4.6.5 p 阶有限域上的每一个 d 次首一既约多项式, 皆能整除 $x^{p^m} - x$, 只要 $d|m$ 。

证明 令 $m=dn$, 设 $f(x)$ 是系数在 $\text{GF}(p)$ 上的 d 次首一既约的多项式, 由定理 4.2.9 可知, 模 $f(x)$ 的剩余类全体构成一个 p^d 阶有限域。由定理 4.6.2 可知 $f(x)$ 在该域中必有根, 设为 α , 则由定理 4.6.4 可知 $\text{GF}(p^d)$ 域中的每一元素均满足

$$x^{(p^d)^n} - x = 0$$

$$x^{p^{dn}} - x = x^{p^m} - x = 0$$

所以 $\alpha^{p^m} - \alpha = 0$ 。这说明 α 既是 $f(x)$ 的根, 又是 $x^{p^m} - x$ 的根。因 $\alpha \in \text{GF}(p^d)$, 所以 α 必是 $x^{p^d} - x$ 的根。故 $f(x) | (x^{p^d} - x)$, 而 $(x^{p^d} - x) | (x^{p^{dn}} - x = x^{p^m} - x)$, 所以 $f(x) | (x^{p^m} - x)$ 。 ■

该定理给我们提供了一个寻找既约多项式的方法。例如, 我们要求把 $\text{GF}(2)$ 上的 $f(x) = x^8 - x$ 进行既约分解, 则由该定理可知, $\text{GF}(2)$ 上的每一个一次及三次既约多项

式均整除 $x^8 - x = x^8 - x$ 。而 GF(2) 上的一次既约多项式为 $1 + x, x$, 而 GF(2) 上的三次既约多项式为 $x^3 + x + 1, x^3 + x^2 + 1$, 所以 $x, (1 + x), (x^3 + x + 1), (x^3 + x^2 + 1)$ 均除得尽 $(x^8 - x)$, 由此可知:

$$x^{2^3} - x = x^8 - x = x(1 + x)(x^3 + x + 1)(x^3 + x^2 + 1)$$

由上面讨论可看出, 构造一个 p^d 级有限域, 就是找一个 GF(p) 上的 d 次既约多项式 $f(x)$, 然后, 以此 $f(x)$ 为模构成一个剩余类, 就得到一个 p^d 阶有限域。因此, 寻找既约多项式, 在有限域的构造上有重大意义, 下面定理再给出寻找既约多项式的方法。

定理 4.6.6 系数取自 GF(p) 上的多项式

$$x^{p^m} - x = \text{所有次数除尽 } m \text{ 的 GF}(p) \text{ 上的首一既约多项式之积}$$

证明

(1) 由定理 4.6.5 知 GF(p) 上的次数除尽 m 的任一首一既约多项式 $f(x) | (x^{p^m} - x)$ 。

(2) 下面我们证明凡是 $f(x) | (x^{p^m} - x)$ 的 d 次首一既约多项式均有 $d | m$ 。

若 $f(x) \neq x$, 则 $f(x) | (x^{p^{m-1}} - 1)$, 由定理 4.5.10 知, $f(x)$ 构造出了一个 p^d 阶有限域 GF(p^d)。令 $\alpha \in \text{GF}(p^d)$, 它是 $f(x)$ 的根, 所以 $\alpha \in \text{GF}(p^m)$, 故 α 也是 $x^{p^m} - x$ 方程的一个根。因此 $\alpha^{p^m} - \alpha = 0, \alpha^{p^m} = \alpha$ 。令 $\beta \in \text{GF}(p^d)$ 且为本原域元素, 则 $\alpha = \beta^i$, 由上可知 $(\beta^i)^{p^m} = \beta^i, \beta^{i(p^{m-1})} = 1$, 而 β 的级为 $p^d - 1$, 所以 $p^d - 1 | i(p^m - 1)$, i 显然小于 p^d , 因此 $(p^d - 1) | i$ 。由此可知 $(p^d - 1) | (p^m - 1)$, 由引理 4.5.1 可得 $d | m$ 。 ■

定理 4.6.7 若 $f(x)$ 为 GF(p) 上的 m 次既约多项式, 且 $m | d$, 则任何 p^d 阶有限域必含有 $f(x)$ 的全部根。

证明 由定理 4.6.6 可知:

$$x^{p^d} - x = \prod_{d | g(x)} g(x)$$

$g(x)$ 是 GF(p) 上的首一既约多项式。

另一方面在 p^d 阶有限域 F 中, $x^{p^d} - x$ 可分解成 p^d 个一次因式的乘积

$$x^{p^d} - x = \prod_{a^i \in F} (x - a^i)$$

所以

$$\prod_{d | g(x)} g(x) = \prod_{a^i \in F} (x - a^i)$$

由于 GF(p) 上的既约多项式 $f(x)$ 的次数 m 必整除 d , 因此 $f(x)$ 必是上式方程左边的一个因式, 故可在等式右边挑出属于 $f(x)$ 的全部 m 个一次因式

$$f(x) = \prod_{j=i_1}^{i_m} (x - a^j) \quad ■$$

由此定理知, 若 $d = m$, 则 m 次首一多项式 $f(x)$ 的全部根在 GF(p^m) 中, 称 GF(p^m) 为 $f(x)$ 的包含所有根的 **最小扩域**, 称为 $f(x)$ 的分裂域或分解域。

如图 4-1 所示, $\text{GF}(p^m) \supset \text{GF}(p^d) \supset \text{GF}(p)$ 。若 $f(x)$ 是 GF(p) 上的 m 次既约多项式, 则 $f(x)$ 在

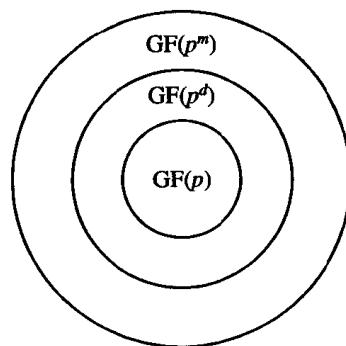


图 4-1 分裂域之间关系图

$GF(p^m)$ 上能完全分裂成一次因式。但它不能在任何中间域 $GF(p^d)$ 上完全分裂，所以 $GF(p^m)$ 上包含了 $f(x)$ 的全部根，是包含了 $f(x)$ 全部根的 $GF(p)$ 的最小扩域。

例 4.14 $GF(2)$ 上的既约多项式 $f(x) = x^4 + x + 1$ ，它必在 $GF(2^4)$ 域上完全分裂，但不能在任何中间域 $GF(2^2)$ 完全分裂。在 $GF(2^4)$ 上 $f(x)$ 完全分解为

$$f(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^4)(x - \alpha^8)$$

式中， $x = \alpha \in GF(2^4)$ ，且是本原域元素，因此 $f(x) = x^4 + x + 1$ 是一个本原多项式。■

上面我们讨论了如何寻找既约多项式的问题，下面讨论 $GF(p)$ 上既约多项式的数目。

定理 4.6.8 $GF(p)$ 上 m 次既约多项式的数目是

$$I_m = \frac{1}{m} \sum_{\substack{d \\ d|m}} \mu(d) p^{m/d} \quad (4.6.1)$$

式中， $\mu(d)$ 为 Möbius 函数。

该定理的证明比较复杂，请参阅 [4]。

例 4.15 求 $GF(2)$ 上既约多项式数目：

$$m = 1, I_1 = \frac{1}{1} \mu(1) \cdot 2 = 2, \text{ 为 } x, x + 1$$

$$m = 2, I_2 = \frac{1}{2} [\mu(1)2^2 + \mu(2)2^1] = \frac{1}{2}(4 - 2) = 1, \text{ 为 } x^2 + x + 1$$

$$m = 3, I_3 = \frac{1}{3} [\mu(1)2^3 + \mu(3)2^1] = \frac{1}{3}(8 - 2) = 2, \text{ 为 } x^3 + x + 1 \text{ 和 } x^3 + x^2 + 1$$

⋮

$$m = 10, I_{10} = \frac{1}{10} [\mu(1)2^{10} + \mu(2)2^5 + \mu(5)2^2 + \mu(10)2^1] = \frac{1}{10}(2^{10} - 2^5 - 2^2 + 2^1) = 99$$

$$m = 60, I_{60} = 19\ 215\ 358\ 392\ 200\ 893$$

由此例看出 m 次既约多项式的个数随 m 指数增长。可以证明在所有 $GF(2)$ 上 m 次多项式集合中，任意取出一个为既约多项式的概率大约是 $1/m$ 。■

下面讨论有限域的同构与唯一性。

由定理 4.6.1 知，有限域的阶必是其特征的幂，而有限域的特征由定理 4.5.2 知是素数。因此有限域的阶必是素数 p 的幂 p^m 。反之，若有一个素数 p ，则一定可以从 $GF(p)[x]$ 上选出一个 m 次既约多项式，以它为模构造出一个 p^m 阶有限域 $GF(p^m)$ 。并且该域包含了 $GF(p)$ 上所有 m 次既约多项式的全部根。如果把 $f(x)$ 的一个根称为 α ，则由定理 4.5.10，可把 p^m 阶有限域中的每一元素，表示成系数在 $GF(p)$ 上且次数低于 m 的 α 多项式。而 $GF(p)$ 上小于 m 次多项式与 $GF(p)$ 上的 m 重同构。因此， m 重、多项式剩余类以及 α 多项式之间均同构，都可用来表示 p^m 阶有限域。

例 4.16 $GF(2)$ 上的三次既约多项式 $f(x) = x^3 + x + 1$ ，以它为模所构成的剩余类构成了 $(2^3) = 8$ 阶有限域 $F_2[x]/(x^3 + x + 1)(GF(2^3))$ 。若 $f(x)$ 的一个根为 α ，则 $f(\alpha) = \alpha^3 + \alpha + 1 = 0$ ， $\alpha^3 = \alpha + 1$ ，因而 2^3 阶有限域也可用低于三次的 α 多项式表示，这时 $GF(2^3)$ 中的自然基底就是 $\{1, \alpha, \alpha^2\}$ 。与此对应的也可用模 $f(x)$ 剩余类及 $GF(2)$ 上的三维矢量表示，当然也可用本原域元素 α 的各次幂表示，如表 4-1 所示。

表 4-1

| 本原域元素表示 | 剩余类表示 | α 多项式表示 | 三维矢量表示 |
|----------------|-----------------|---------------------|--------|
| 0 | $\bar{0}$ | 0 | 000 |
| $\alpha^0 = 1$ | $\bar{1}$ | α^0 | 001 |
| α | \bar{x} | α | 010 |
| α^2 | $\bar{x^2}$ | α^2 | 100 |
| α^3 | $\bar{1+x}$ | $1+\alpha$ | 011 |
| α^4 | $\bar{x+x^2}$ | $\alpha+\alpha^2$ | 110 |
| α^5 | $\bar{1+x+x^2}$ | $1+\alpha+\alpha^2$ | 111 |
| α^6 | $\bar{1+x^2}$ | $1+\alpha^2$ | 101 |

显然，用上述几种方法表示均可以，它们都是一样的。因此，今后不论用何种方法产生的 p^n 阶有限域，也不论是几个 p^n 阶有限域，它们都是同构的。从这个意义上讲， p^n 阶有限域是唯一的，也就是所有 p^n 阶有限域都同构于用 $GF(p)[x]$ 上模 m 次既约多项式 $f(x)$ 构造的剩余类域，均可用 $GF(p^n)$ 表示。

到此为止，我们较详细地讨论了有限域的结构，其目的主要是为了像 $x^{p^n} - x$ 型的多项式的因式分解，以便为以后学习循环码打下基础。下面我们举一个例子，说明这种类型多项式分解成既约因式的方法。

例 4.17 分解 $x^{16} - x$ 。

由例 4.9 已知 $x^{16} - x = x(x^{15} - 1) = xQ^{(1)}(x)Q^{(3)}(x)Q^{(5)}(x)Q^{(15)}(x)$ ，且已求出：

$$\begin{aligned} Q^{(1)}(x) &= x + 1 \\ Q^{(3)}(x) &= x^2 + x + 1 \\ Q^{(5)}(x) &= x^4 + x^3 + x^2 + x + 1 \\ Q^{(15)}(x) &= x^8 + x^7 + x^5 + x^4 + x^3 + x + 1 \end{aligned}$$

$Q^{(5)}(x) = x^4 + x^3 + x^2 + x + 1$ 是以 5 级元素为根，由方指数定义可知， $2^m \equiv 1 \pmod{5}$ ， $m = 4$ ，可知该 5 级元素的最小多项式次数为四次，因此 $x^4 + x^3 + x^2 + x + 1$ 已为既约，在 $GF(2)$ 上不能再分解。

$Q^{(15)}(x) = x^8 + x^7 + x^5 + x^4 + x^3 + x + 1$ 是以 15 级元素为根，由于 $2^m \equiv 1 \pmod{15}$ ， $m = 4$ 。可知 15 级元素在 $GF(2)$ 上的最小多项式是四次，而现在的 $Q^{(15)}(x)$ 为八次，故可以再分解成两个四次既约多项式。下面介绍分解方法：

用待定系数法求解既约因式： $Q^{(15)}(x)$ 的既约因式必为 $x^4 + Ax^3 + Bx^2 + Cx + 1$ 形式，常数项必为 1，否则， $Q^{(15)}(x)$ 必有 x 因子，而这是不可能的。因为 1 元素不能是方程的根（1 不是 15 级元素），故

$$A + B + C = 1 \quad A, B, C \in GF(2)$$

所以，或者 A, B, C 三者均为 1，或者只有其中一个为 1。若三者均为 1，则为 $x^4 + x^3 + x^2 + x + 1$ ，而这是 $Q^{(5)}(x)$ ，所以不可能。因此，只有 A, B, C 中有一个为 1。当 $B=1$ ， $A=0, C=0$ 时，则

$$x^4 + Ax^3 + Bx^2 + Cx + 1 = x^4 + x^2 + 1 = (x^2 + x + 1)^2$$

这也是不可能的。因此，仅只有 $A=1$ 或 $C=1$ ，若 $A=1$ ，其它为 0，则

$$x^4 + Ax^3 + Bx^2 + Cx + 1 = x^4 + x^3 + 1$$

若 $C=1$ ，其它为 0，则

$$x^4 + Ax^3 + Bx^2 + Cx + 1 = x^4 + x + 1$$

显然，这两个多项式均是既约的，且为互反多项式，所以

$$Q^{(15)}(x) = x^8 + x^7 + x^5 + x^4 + x^3 + x + 1 = (x^4 + x^3 + 1)(x^4 + x + 1)$$

由定理 4.6.6 知 $x^4 - x$ 是等于所有次数除尽 4 的 GF(2) 上首一既约多项式之积，可以检查上边等式右边，就是所有次数除尽 4，即 1, 2, 4 次的 GF(2) 上既约多项式之积。

由上可知， $x^4 + x + 1$ 或 $x^4 + x^3 + 1$ 均以 15 级元素为根。因此，若以此多项式作剩余类，就能得到 2^4 阶有限域 GF(2⁴)；若以它的根 α 表示，则 GF(2⁴) 上的 15 个非 0 元素如表 4-2 所示（用 $x^4 + x + 1$ 为模作剩余类）。

表 4-2 以 $x^4 + x + 1$ 为模的 GF(2⁴) 的元素

| | | | |
|------------------------------------|------|--|------|
| $\alpha^0 = 1$ | 0001 | $\alpha^8 = 1 + \alpha^2$ | 0101 |
| α | 0010 | $\alpha^9 = \alpha + \alpha^3$ | 1010 |
| α^2 | 0100 | $\alpha^{10} = \alpha^2 + \alpha + 1$ | 0111 |
| α^3 | 1000 | $\alpha^{11} = \alpha^3 + \alpha^2 + \alpha$ | 1110 |
| $\alpha^4 = \alpha + 1$ | 0011 | $\alpha^{12} = 1 + \alpha + \alpha^2 + \alpha^3$ | 1111 |
| $\alpha^5 = \alpha^2 + \alpha$ | 0110 | $\alpha^{13} = 1 + \alpha^2 + \alpha^3$ | 1101 |
| $\alpha^6 = \alpha^3 + \alpha^2$ | 1100 | $\alpha^{14} = 1 + \alpha^3$ | 1001 |
| $\alpha^7 = 1 + \alpha + \alpha^3$ | 1011 | $\alpha^{15} = 1$ | 0001 |

下面我们进行检验，看看各次多项式是否以相应的元素为根：

$$Q^{(15)}(x) = (x^4 + x + 1)(x^4 + x^3 + 1)$$

它们均以 15 级元素为根。 $x^4 + x + 1$ 的四个根为： $\alpha, \alpha^2, \alpha^4, \alpha^8$ ，例如， α 代入方程的 $x^4 + x + 1 = (1 + \alpha) + \alpha + 1 = 0$ ；而 $x^4 + x^3 + 1$ 的根为： $\alpha^7, \alpha^{14}, \alpha^{13}, \alpha^{11}$ ，代入 α^7 得 $(\alpha^7)^4 + (\alpha^7)^3 + 1 = \alpha^{28} + \alpha^{21} + 1 = \alpha^{13} + \alpha^6 + 1 = (1 + \alpha^2 + \alpha^3) + (\alpha^3 + \alpha^2) + 1 = 0$ 。

$Q^{(5)}(x) = x^4 + x^3 + x^2 + x + 1$ 。它以 5 级元素 $\alpha^3, \alpha^6, \alpha^{12}, \alpha^9$ 为根。如 α^3 代入得 $(\alpha^3)^4 + (\alpha^3)^3 + (\alpha^3)^2 + \alpha^3 + 1 = \alpha^{12} + \alpha^9 + \alpha^6 + \alpha^3 + 1 = (1 + \alpha + \alpha^2 + \alpha^3) + (\alpha + \alpha^3) + (\alpha^3 + \alpha^2) + \alpha^3 + 1 = 0$ 。

$Q^{(3)}(x) = x^2 + x + 1$ 。它以 3 级元素 α^5, α^{10} 为根。如以 α^5 代入则得 $(\alpha^5)^2 + \alpha^5 + 1 = \alpha^{10} + \alpha^5 + 1 = (\alpha^2 + \alpha + 1) + (\alpha^2 + \alpha) + 1 = 0$ 。

$Q^{(3)}(x) = x + 1$ 以 $\alpha^0 = 1$ 为根。

由定理 4.3.2 我们可以在 GF(2⁴) 中找到所有循环子群。子群的阶数必是 $n = 2^4 - 1 = 15$ 的因子，所以有以下子群：

(1) $1, \alpha, \alpha^2, \alpha^3, \alpha^4, \dots, \alpha^{14}$ ，这是一个 15 阶的循环子群，生成元为 α ，是假子群。

(2) $1, \alpha^7, \alpha^{14}, (\alpha^7)^3 = \alpha^{21} = \alpha^6, (\alpha^7)^4 = \alpha^{13}, \dots, (\alpha^7)^{14} = \alpha^8$ ，这也是一个 15 阶循环群，生成元是 α^7 ，也是一个假子群。

(3) 以 $1, \alpha^3, \alpha^6, \alpha^{12}, \alpha^9$ 元素组成一个五阶子群，生成元是 5 级元素 α^3 。

(4) 以 $1, \alpha^5, \alpha^{10}$ 三个元素组成一个三阶子群，生成元是 3 级元素 α^5 。并且加上 0 元素后，组成一个 4 个元素的域，它就是 $GF(2^4)$ 中的一个 $GF(2^2)$ 子域。

(5) 以一个单位元“1”组成一个子群，且加上一个 0 元素后，也组成 $GF(2^4)$ 中的一个子域，它就是基域 $GF(2)$ 。

我们知道，一个多项式是否既约与它所在域很有关系。在该例中：

$$\begin{aligned} x(x^{15}-1) &= x(x+1)(x^2+x+1)(x^4+x^3+x^2+x+1) \\ &\quad \cdot (x^4+x+1)(x^4+x^3+1) \end{aligned}$$

等式右边的多项式在 $GF(2)$ 全部都为既约，但是如果在 $GF(2^2) = GF(4)$ 上，则 3 个四次方程不是既约的，还可以再分解。

设 $GF(2^2) = GF(4)$ 中的 4 个元素为 $0, 1, \beta, \gamma$ ，若 β 为生成元，则 $\beta^2=\gamma, \beta^3=1$ 。由定理 4.6.6 知：

$$x^{16}-x=x^{4^2}-x$$

是等于所有次数除尽 m 的 $GF(4)$ 上的首一既约多项式之积。这里的 $m=2$ ，所以上式可分解为一次和二次既约多项式之积，应用如上所述的特定系数法：对 3 个四次方程分解如下：

$$\begin{aligned} x^4+x^3+x^2+x+1 &= (x^2+Ax+B)(x^2+Cx+D) \\ &= x^4+(A+C)x^3+(AC+D+B)x^2+(DA+BC)x+BD \end{aligned}$$

进行分析后可得

$$x^4+x^3+x^2+x+1=(x^2+\beta x+1)(x^2+\gamma x+1)$$

用类似的方法可得到：

$$\begin{aligned} x^4+x^3+1 &= (x^2+\beta x+\beta)(x^2+\gamma x+\gamma) \\ x^4+x+1 &= (x^2+x+\beta)(x^2+x+\gamma) \end{aligned}$$

所以在 $GF(2^2) = GF(4)$ 上。

$$\begin{aligned} x^{16}-x &= x(x+1)(x^2+x+1)(x^2+\beta x+1)(x^2+\gamma x+1)(x^2+x+\beta) \\ &\quad \cdot (x^2+x+\gamma)(x^2+\beta x+\beta)(x^2+\gamma x+\gamma) \end{aligned}$$

正如前面提到过， $GF(p)$ 上的 m 次既约多项式 $f(x)$ ，可以在 $GF(p^m)$ 上完全分裂成一次因式，但不能在任何中间域 $GF(p^d) \subset GF(p^m)$ 上完全分解。故所有前面 3 个四次多项式，只能在 $GF(2^2)$ 域上分解成二次因式，但在 $GF(2^4)$ 上能完全分解成一次因式。所以，再应用定理 4.6.6，我们还可对 $x^{16}-x$ 再进行分解，在 $GF(2^4) = GF(16)$ 域上完全分解成一次因式：

$$\begin{aligned} x^{16}-x &= (x+1)(x+\alpha^5)(x+\alpha^{10})(x+\alpha^3)(x+\alpha^6)(x+\alpha^{12})(x+\alpha^9)(x+\alpha) \\ &\quad \cdot (x+\alpha^2)(x+\alpha^4)(x+\alpha^8)(x+\alpha^7)(x+\alpha^{14})(x+\alpha^{13})(x+\alpha^{11}) \end{aligned}$$

综上所述，随着域由 $GF(2)$ 到 $GF(2^2)$ 再扩到 $GF(2^4)$ 上， $x^{16}-x$ 的因式分解逐步深入，最后完全分裂成一次因式，所以可把上面结果归结如下：

$$\begin{aligned} GF(2): x^{16}-x &= x(x+1)(x^2+x+1)(x^4+x^3+x^2+x+1) \\ &\quad \cdot (x^4+x+1)(x^4+x^3+1) \end{aligned}$$

$$\begin{aligned} GF(2^2): x^{16}-x &= x(x+1)(x^2+x+1)(x^2+\beta x+1)(x^2+\gamma x+1)(x^2+x+\beta) \\ &\quad \cdot (x^2+x+\gamma)(x^2+\beta x+\beta)(x^2+\gamma x+\gamma) \end{aligned}$$

$$\begin{aligned} \text{GF}(2^4): x^{16} - x = & x(x+1)(x+\alpha^5)(x+\alpha^{10})(x+\alpha^3)(x+\alpha^6)(x+\alpha^{12})(x+\alpha^9) \\ & \cdot (x+\alpha)(x+\alpha^2)(x+\alpha^4)(x+\alpha^8)(x+\alpha^7)(x+\alpha^{14})(x+\alpha^{13})(x+\alpha^{11}) \quad \blacksquare \end{aligned}$$

例 4.18 把 $x^9 - 1$ 分解成 GF(2) 上的既约因式乘积。先对 $x^9 - 1$ 进行分圆多项式分解：

$$x^9 - 1 = Q^{(1)}(x)Q^{(3)}(x)Q^{(9)}(x)$$

所以, $x^9 - 1$ 的根的级数为 1、3、9 级。现在主要找 9 级元素的最小多项式, 由方次数可知:

$$2^m \equiv 1 \pmod{9}$$

得到 $m=6$, 所以含 9 级元素的最小多项式的次数为 6 次, 因此 9 级元素必在 GF(2⁶) 域上。设 α 为 GF(2⁶) 上的本原域元素 $\alpha^{63}=1$, 则 $(\alpha^7)^9=1$, α^7 为 9 级元素, 因此含 9 级元素的多项式为

$$f(x) = (x - \alpha^7)(x - \alpha^{14})(x - \alpha^{28})(x - \alpha^{56})(x - \alpha^{49})(x - \alpha^{35})$$

再由此式化成 GF(2) 上的形式, 这较麻烦, 但可以查文献[2]中的附录得到

$$f(x) = x^6 + x^3 + 1$$

而以 1 级和 3 级元素为根的最小多项式前面已求出, 所以

$$x^9 - 1 = (x+1)(x^2+x+1)(x^6+x^3+1) \quad \blacksquare$$

由上讨论可知, 如果要把 $x^n - 1$ 多项式分解成 GF(p) 上既约多项式的乘积, 则首先由

$$p^m \equiv 1 \pmod{n}$$

找到含有 n 作为因式的最小 p^m , 可知 $x^n - 1$ 的全部根必含在 GF(p^m) 中, 而 GF(p^m) 是含有这 n 个根的最小扩域。在 GF(p^m) 中, 找到一个 n 级元素 β , $\beta^n = 1$, 若 α 是 GF(p^m) 的本原域元素, 则 $\beta = \alpha^j$, 由 $\beta^n = (\alpha^j)^n = 1$ 可知, $j \mid n = p^m - 1$ 。由此可知:

$$x^n - 1 = \prod_{i=0}^{n-1} (x - \beta^i) = \prod_i f_i(x)$$

这里, $f_i(x)$ 就是以 β^i 为根的 GF(p) 上的最小多项式。对于 GF(2) 域来说, 以 β^i 为根的 GF(2) 上的最小多项式可以查文献[2] 中的附录 C 得到。

* § 4.7 迹与对偶基

正如例 4.16 中所看到的, GF(p^m) 中的 $p^m - 1$ 个非 0 元素, 均可以用本原域元素 α 的各次幂表示: $\alpha^0 = 1, \alpha, \alpha^2, \dots, \alpha^{p^m-1}$, 也可用本原多项式的根 α 多项式表示, 此时 $\{1, \alpha, \alpha^2, \dots, \alpha^{m-1}\}$ 就是域的一组自然基底或本原基底, 如 GF(2³) 中的自然基底是 $\{1, \alpha, \alpha^2\}$, 由于域中的基底不是唯一的, 因此也可以用另一组基底的线性组合来表示域中的元素。

一、域的正规基

定义 4.7.1 定义 $\{\alpha, \alpha^p, \alpha^{p^2}, \dots, \alpha^{p^{m-1}}\}$ 为 GF(p^m) 域的正规基底, 这里, α 是 GF(p^m) 中的本原域元素。

由定理 4.5.6 知, 正规基底其实是以 α 为根的本原多项式 $f(x)$ 的共轭根系。但是, 并不是每个本原多项式的共轭根系都能作为正规基。这由于作为一组基底元素, 它们之间必须线性无关。如例 4.16 中所指出, α 是 GF(2³) 中的本原元, 它是本原多项式 $f(x) = x^3 + x + 1$ 的根, 则它的共轭根系是 $\{\alpha, \alpha^2, \alpha^4\}$, 但它们不能作为 GF(2³) 的正规基, 因为

$$\alpha + \alpha^2 + \alpha^4 = \alpha + \alpha^2 + (\alpha + \alpha^2) = 0$$

这 3 个元素线性相关。

如果 α 是本原多项式 $p(x) = x^3 + x^2 + 1$ 的根，则 $p(\alpha) = \alpha^3 + \alpha^2 + 1 = 0$, $\alpha^3 = 1 + \alpha^2$, 此时 $\{\alpha, \alpha^2, \alpha^4\}$ 可以作为 $GF(2^3)$ 的正规基，因为此时不可能找到一组非全 0 的系数 (C_1, C_2, C_3) 能使

$$C_1\alpha + C_2\alpha^2 + C_3\alpha^4 = 0 \quad C_1, C_2, C_3 \in GF(2)$$

成立。也就是 $\{\alpha, \alpha^2, \alpha^4\}$ 中元素之间线性无关，由此可知用这组正规基表示 $GF(2^3)$ 的元素，如表 4-3 所示。

可以证明，对任意的 $GF(p)$ ，除少数次数之外必存在有任意次数的 $GF(p)$ 上的本原多项式，它们的共轭根系是一组正规基。表 4-4 列出了共轭根系是正规基的 $GF(2)$ 上的本原多项式。

利用自然基表示的最大优点是 $GF(p^n)$ 域中的任一元素 α^i 乘 α 的运算非常简单，如 $GF(2^3)$ 中的 α^3 乘 α ，即为 α^4 ，这就是 α^3 的自然基表示系数 (011) 的向左循环移位，并按 $\alpha^3=1+\alpha$ 运算的结果 (110)，而这种运算可用很简单的电路实现。

表 4-3 $GF(2^3)$ 中元素的各种基的表示及其迹

| α 幂次 | 自然基表示 | 自然基系数 $\alpha^2 \alpha^1$ | 自然基 的对偶基 | 对偶基系数 $\alpha^5 \alpha^3 \alpha^4$ | 正规基表示 | 正规基系数 $\alpha^4 \alpha^2 \alpha$ | 迹 |
|-------------|-------------------------|------------------------------|----------------------------------|---------------------------------------|--------------------------------|-------------------------------------|---|
| 0 | 0 | 000 | | 000 | | 000 | 0 |
| 1 | 1 | 001 | $\alpha^5 + \alpha^3 + \alpha^4$ | 111 | $\alpha^4 + \alpha^2 + \alpha$ | 111 | 1 |
| α | α | 010 | $\alpha^3 + \alpha^4$ | 011 | α | 001 | 1 |
| α^2 | α^2 | 100 | $\alpha^5 + \alpha^4$ | 101 | α^2 | 010 | 1 |
| α^3 | $\alpha^2 + 1$ | 101 | α^3 | 010 | $\alpha^4 + \alpha$ | 101 | 0 |
| α^4 | $\alpha^2 + \alpha + 1$ | 111 | α^4 | 001 | α^4 | 100 | 1 |
| α^5 | $\alpha + 1$ | 011 | α^5 | 100 | $\alpha^4 + \alpha^2$ | 110 | 0 |
| α^6 | $\alpha^2 + \alpha$ | 110 | $\alpha^5 + \alpha^3$ | 110 | $\alpha^2 + \alpha$ | 011 | 0 |

表 4-4

| 次数 | 本原多项式 | 次数 | 本原多项式 |
|----|-----------------------------|----|--------------------------------------|
| 1 | $x+1$ | 9 | $x^9 + x^8 + x^5 + x^4 + 1$ |
| 2 | $x^2 + x + 1$ | 10 | $x^{10} + x^9 + x^4 + x + 1$ |
| 3 | $x^3 + x^2 + 1$ | 11 | $x^{11} + x^{10} + x^3 + x^2 + 1$ |
| 4 | $x^4 + x^3 + 1$ | 12 | $x^{12} + x^{11} + x^8 + x^6 + 1$ |
| 5 | $x^5 + x^4 + x^2 + x + 1$ | 13 | $x^{13} + x^{12} + x^{10} + x^9 + 1$ |
| 6 | $x^6 + x^5 + 1$ | 14 | $x^{14} + x^{13} + x^8 + x^4 + 1$ |
| 7 | $x^7 + x^6 + 1$ | 15 | $x^{15} + x^{14} + 1$ |
| 8 | $x^8 + x^7 + x^5 + x^3 + 1$ | 16 | $x^{16} + x^{15} + x^{13} + x^4 + 1$ |

利用正规基表示域元素的最大优点是 $GF(p^n)$ 上任意元素 β 的 p 次幂运算非常简单。

设 $\beta \in GF(p^m)$, 它的正规基表示是

$$\beta = a_{m-1}\alpha^{p^{m-1}} + a_{m-2}\alpha^{p^{m-2}} + \cdots + a_1\alpha^p + a_0\alpha$$

相应的正规基表示的系数 m 重是 $(a_{m-1}, a_{m-2}, \dots, a_1, a_0)$

$$\begin{aligned}\beta^p &= (a_{m-1}\alpha^{p^{m-1}} + a_{m-2}\alpha^{p^{m-2}} + \cdots + a_1\alpha^p + a_0\alpha)^p \\ &= (a_{m-1}\alpha^{p^m} + a_{m-2}\alpha^{p^{m-1}} + \cdots + a_1\alpha^{p^2} + a_0\alpha^p) \\ &= (a_{m-2}\alpha^{p^{m-1}} + a_{m-3}\alpha^{p^{m-2}} + \cdots + a_0\alpha^p + a_{m-1}\alpha)\end{aligned}$$

相应的正规基表示的系数 m 重是 $(a_{m-2}, a_{m-3}, \dots, a_0, a_{m-1})$, 它是 β 正规基系数 $(a_{m-1}, a_{m-2}, \dots, a_1, a_0)$ 向左循环移位的结果。这种优点使得在实现元素的乘 p 次幂运算时, 能用很简单的电路实现^[5,6]。

二、迹(trace)与对偶基

在域的运算和研究中, 迹是非常有用分析工具。

定义 4.7.2 $\alpha \in GF(q^m)$, 则它在 $GF(q)$ 上的迹定义为

$$T_r(\alpha) = \alpha + \alpha^q + \alpha^{q^2} + \cdots + \alpha^{q^{m-1}} \quad q \text{ 为素数或素数幂} \quad (4.7.1)$$

迹有以下性质:

定理 4.7.1 对所有 $\alpha, \beta \in GF(q^m)$, 有

- (1) $T_r(\alpha) \in GF(q)$;
- (2) $T_r(\alpha + \beta) = T_r(\alpha) + T_r(\beta)$;
- (3) $T_r(\lambda\alpha) = \lambda T_r(\alpha) \quad \lambda \in GF(q)$;
- (4) $T_r(\alpha^q) = T_r(\alpha)$;

(5) 迹是把 $GF(q^m)$ 中的元素映射到 $GF(q)$ 中。

该定理的证明留作习题, 请读者证明。

设 β 是 $GF(q^m)$ 中的一个元素, 则它可以用 $GF(q^m)$ 中的自然基底表示为

$$\beta = a_0 + a_1\alpha + a_2\alpha^2 + \cdots + a_{m-1}\alpha^{m-1} \quad a_i \in GF(q)$$

因而由定理 4.7.1 中的(2)、(3)可得:

$$T_r(\beta) = a_0T_r(1) + a_1T_r(\alpha) + \cdots + a_{m-1}T_r(\alpha^{m-1}) \quad (4.7.2)$$

因此为了计算域中任何元素的迹, 只要计算自然基底元素的迹就足够了, 如在 $GF(2^3)$ 中, 若 α 是 $f(x) = x^3 + x^2 + 1$ 的根, 则

$$T_r(1) = 1 + 1 + 1 = 1$$

$$T_r(\alpha) = \alpha + \alpha^2 + \alpha^4 = 1$$

$$T_r(\alpha^2) = \alpha^2 + \alpha^4 + \alpha^8 = 1$$

而其它元素的迹, 则可用式(4.7.2)计算得到。如 $\alpha^3 = 1 + \alpha^2$, 它的迹

$$T_r(\alpha^3) = a_0T_r(1) + a_1T_r(\alpha) + a_2T_r(\alpha^2) = T_r(1) + T_r(\alpha^2) = 0$$

$GF(2^3)$ 中各元素的迹如表 4-3 所示。 $\{\alpha, \alpha^2, \alpha^4\}$ 是 $f(x) = x^3 + x^2 + 1$ 的一组共轭根系, 由定理 4.7.1 的(4)可知, 它们有相同的迹。同样, $\{\alpha^3, \alpha^5, \alpha^6\}$ 也是一组共轭根, 它们也有相同的迹。

定理 4.7.2 当且仅当有一个元素 $\beta \in GF(q^m)$, 使 $\alpha = \beta - \beta^q$, 则 $T_r(\alpha) = 0$ 。

证明 如果 $\alpha = \beta - \beta^q$, 则由定理 4.7.1 可知:

$$T_r(\alpha) = T_r(\beta - \beta^q) = T_r(\beta) - T_r(\beta^q) = T_r(\beta) - T_r(\beta) = 0$$

在 $GF(q^m)$ 中必存在有一个元素 Q , 使 $T_r(Q) = 1$, 取

$$\beta = \alpha Q^q + (\alpha + \alpha^q)Q^{q^2} + \cdots + (\alpha + \alpha^q + \cdots + \alpha^{q^{m-2}})Q^{q^{m-1}}$$

则

$$\beta^q = (\alpha Q^q)^q + (\alpha + \alpha^q)^q Q^{q^3} + \cdots + (\alpha + \alpha^q + \cdots + \alpha^{q^{m-2}})^q Q^{q^m}$$

可得

$$\begin{aligned} \beta - \beta^q &= \alpha(T_r(Q) - Q) - Q(T_r(\alpha) - \alpha) = \alpha T_r(Q) - \alpha Q - QT_r(\alpha) + \alpha Q \\ &= \alpha T_r(Q) - QT_r(\alpha) \end{aligned}$$

若 $T_r(\alpha) = 0$, 并由假设知 $T_r(Q) = 1$, 因而

$$\beta - \beta^q = \alpha$$

由该定理可知, 在 $GF(2^3)$ 中, $\alpha^5 = \alpha^4 + \alpha^2 = (\alpha^2)^2 + (\alpha^2) = \beta^2 + \beta = \beta + \beta^2 = \beta - \beta^2$, 因而 $T_r(\alpha^5) = 0$ 。

定理 4.7.3 与 $GF(p^m)$ 的基底 $B = (\beta_0, \beta_1, \dots, \beta_{m-1})$ 相对应, 若

$$T_r(\beta_i \gamma_j) = \begin{cases} 1 & i = j \\ 0 & \text{其它} \end{cases}$$

满足, 则称 $GF(p^m)$ 的基底 $B' = (\gamma_0, \gamma_1, \dots, \gamma_{m-1})$ 是 B 的**对偶基**。如果 B 的对偶基是自身, 即 $B' = B$, 则称 B 为**自对偶基**。

可以证明 $GF(p^m)$ 中必存在有唯一的对偶基。如 $GF(2^3)$ 中与自然基 $\{1, \alpha, \alpha^2\}$ 相对应的对偶基是 $\{\alpha^4, \alpha^3, \alpha^5\}$, 与正规基 $\{\alpha, \alpha^2, \alpha^4\}$ 相对应的对偶基是 $\{\alpha, \alpha^2, \alpha^4\}$, 这里, α 是 $f(x) = x^3 + x^2 + 1$ 的根。

下面讨论如何找对偶基。定义 $GF(p)$ 上的 $m \times m$ 阶矩阵 $[A]$, 它的第 i 行第 j 列元素

$$a_{ij} = T_r(\beta_i \beta_j), \quad i, j = 0, 1, \dots, m-1, \text{ 且令 } \gamma_i = \gamma^i, \beta_i = \beta^i.$$

令 $[B] = [A^{-1}]$, 若 $[B]$ 的第 k 行第 j 列元素为 b_{kj} , 则基底 $(\beta_0, \beta_1, \dots, \beta_{m-1})$ 的对偶基

$$\begin{aligned} (\gamma_0, \gamma_1, \dots, \gamma_{m-1}) &= (\beta_0, \beta_1, \dots, \beta_{m-1}) \begin{bmatrix} b_{0,0} & \cdots & b_{0,m-1} \\ \vdots & & \vdots \\ b_{m-1,0} & \cdots & b_{m-1,m-1} \end{bmatrix} \\ &= (\beta_0, \beta_1, \dots, \beta_{m-1}) [B] \end{aligned} \tag{4.7.3}$$

或

$$\gamma_j = \sum_{k=0}^{m-1} b_{kj} \beta_k, \quad j = 0, 1, \dots, m-1 \tag{4.7.4}$$

如果 $x \in GF(p^m)$, 且令

$$x = \sum_{i=0}^{m-1} x_i \beta_i, \quad x_i \in GF(p)$$

是 x 的 $(\beta_0, \beta_1, \dots, \beta_{m-1})$ 基底表示, 则它的对偶基表示为

$$x = \sum_{j=0}^{m-1} x'_j \gamma_j, \quad x'_j \in GF(p)$$

设 $x = (x_0, x_1, \dots, x_{m-1})$ 和 $x' = (x'_0, x'_1, \dots, x'_{m-1})$ 的列矢量分别为 x^T 和 x'^T , 则可以验证:

$$\mathbf{x}'^T = [\mathbf{A}]\mathbf{x} \quad x'_j = T_r(\mathbf{x}\beta_j) \quad (4.7.5)$$

$$\mathbf{x}^T = [\mathbf{B}]\mathbf{x}' \quad x_i = T_r(\mathbf{x}\gamma_i) \quad (4.7.6)$$

例 4.19 $\text{GF}(2^3)$ 上的自然基底是 $B = (\beta_0, \beta_1, \beta_2) = \{1, \alpha, \alpha^2\}$, α 是本原元, 且 $\alpha^3 = \alpha^2 + 1$ 。由表 4-3 可知, $T_r(1) = T_r(\alpha) = T_r(\alpha^2) = T_r(\alpha^4) = 1$, $T_r(\alpha^3) = T_r(\alpha^5) = T_r(\alpha^6) = 0$, 因而矩阵

$$[\mathbf{A}] = T_r \begin{bmatrix} \alpha^0 \alpha^0 & \alpha^0 \alpha^1 & \alpha^0 \alpha^2 \\ \alpha^1 \alpha^0 & \alpha^1 \alpha^1 & \alpha^1 \alpha^2 \\ \alpha^2 \alpha^0 & \alpha^2 \alpha^1 & \alpha^2 \alpha^2 \end{bmatrix} = T_r \begin{bmatrix} 1 & \alpha & \alpha^2 \\ \alpha & \alpha^2 & \alpha^3 \\ \alpha^2 & \alpha^3 & \alpha^4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

$$[\mathbf{B}] = [\mathbf{A}^{-1}] = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

所以, 由式(4.7.3)可得与自然基 $(\beta_0, \beta_1, \beta_2) = (\alpha^0, \alpha^1, \alpha^2)$ 对应的对偶基是 $\mathbf{B}' = (\gamma_0, \gamma_1, \gamma_2) = (\alpha^4, \alpha^3, \alpha^5)$, 用它表示的域中元素如表 4-3 所示。同理, 可以计算得到与正规基 $\{\alpha, \alpha^2, \alpha^4\}$ 相对应的对偶基是 $\{\alpha, \alpha^2, \alpha^4\}$, 可知该例中的正规基是自对偶基。

利用对偶基的一个优点是用 α 乘某一元素 $a \in \text{GF}(p^n)$ 时, 非常容易实现, 并且能简化某些码的译码电路^[5]。

* § 4.8 孙子定理(中国剩余定理)

孙子定理又称**中国剩余定理**, 它是我国古代数学的重要成就之一, 它在代数编码理论中, 特别是循环码的译码、Goppa 码的译码和纠突发错误码的快速译码中, 以及广义码的构造中起非常重要的作用。

整数环上的孙子定理: 给定一组两两互素的整数 m_1, m_2, \dots, m_k , 和整数 c_1, c_2, \dots, c_k , 则同余组

$$x \equiv c_1 \pmod{m_1}$$

$$x \equiv c_2 \pmod{m_2}$$

⋮

$$x \equiv c_k \pmod{m_k}$$

具有唯一解:

$$x \equiv x_0 \pmod{(m_1 m_2 \cdots m_k)}$$

其中

$$(a) x_0 = M_1 M'_1 c_1 + M_2 M'_2 c_2 + \cdots + M_k M'_k c_k$$

$$(b) m_1 m_2 \cdots m_k = m_s M_s \quad M_s = m_1 m_2 \cdots m_{s-1} m_{s+1} \cdots m_k$$

$$(c) M_s M'_s \equiv 1 \pmod{m_s} \quad s = 1, 2, \dots, k$$

证明

(1) 由假设(b)和(c)可知, 除了 m_s 以外, 在所有 m_λ , $\lambda = 1, 2, \dots, k$ ($\lambda \neq s$) 中, $m_\lambda | M_s$, 即

$$M_s \equiv 0 \pmod{m_\lambda} \quad \lambda \neq s, \lambda = 1, 2, \dots, k$$

由此可知：

$$M_s M'_s \equiv \begin{cases} 1 \pmod{m_\lambda} & \lambda = s \\ 0 \pmod{m_\lambda} & \lambda \neq s \end{cases}$$

该式表示正交性，由此正交性可得：

$$\begin{aligned} x_0 &= M_1 M'_1 c_1 + M_2 M'_2 c_2 + \cdots + M_s M'_s c_s + \cdots + M_k M'_k c_k \\ &\equiv c_s \pmod{m_s} \quad s = 1, 2, \dots, k \end{aligned}$$

这就告诉我们 x_0 确实是同余组的解。

(2) 证明解的唯一性。若另有一个 y_0 ，也是同余组的解，则由 $y_0 \equiv c_s, x_0 \equiv c_s \pmod{m_s}, s = 1, 2, \dots, k$ 可得：

$$x_0 \equiv y_0 \pmod{m_s} \quad s = 1, 2, \dots, k$$

所以

$$m_s | (x_0 - y_0) \quad s = 1, 2, \dots, k$$

也就是说：

$$m_1 | (x_0 - y_0), m_2 | (x_0 - y_0), \dots, m_k | (x_0 - y_0)$$

所以， $(x_0 - y_0)$ 是 m_1, m_2, \dots, m_k 的一个公倍数。而公倍数一定是最小公倍数的倍数，又由假设可知 m_1, m_2, \dots, m_k 两两互素，所以

$$\text{LCM}(m_1, m_2, \dots, m_k) = m_1 m_2 \cdots m_k$$

故

$$x_0 - y_0 = t(m_1 m_2 \cdots m_k) \equiv 0 \pmod{m_1 m_2 \cdots m_k}$$

所以

$$x_0 \equiv y_0 \pmod{m_1 m_2 \cdots m_k}$$

因此解具有唯一性。 ■

例 4.20 已知： $x \equiv 2 \pmod{3}$, $x \equiv 3 \pmod{5}$, $x \equiv 2 \pmod{7}$, 求解 x 。

由此可知： $c_1 = 2, c_2 = 3, c_3 = 2$, $m_1 = 3, m_2 = 5, m_3 = 7$ 。可得： $M_1 = m_2 m_3 = 35$, $M_2 = m_1 m_3 = 21$, $M_3 = m_1 m_2 = 15$ 。下面求 M'_1, M'_2 和 M'_3 。

$$M_1 M'_1 \equiv 1 \pmod{m_1 = 3}$$

$$35 M'_1 \equiv 1 \pmod{3}$$

由欧几里德算法可知：

$$(3, 35) = 1 = 12 \cdot 3 + (-1) \cdot 35$$

由 $(-1) \cdot 35 \equiv 1 \pmod{3}$ ，得 $M'_1 = (-1) = 2$ 。

同理可得： $21 \cdot 1 \equiv 1 \pmod{5}$, $M'_2 = 1$; $15 \cdot 1 \equiv 1 \pmod{7}$, $M'_3 = 1$; 所以

$$x = 2 \times 35 \times 2 + 3 \times 1 \times 21 + 2 \times 15 \times 1 = 233 \equiv 23 \pmod{105}$$

把 $x = 23$ 代入同余组进行检验，说明是对的。 ■

类似于整数环上的孙子定理，在多项式环上也有类似的结果。

多项式环上的孙子定理：给定两两互素的多项式 $m_1(x), m_2(x), \dots, m_k(x)$ ，及一组多项式 $c_1(x), c_2(x), \dots, c_k(x)$ ，则同余组

$$h(x) \equiv c_1(x) \pmod{m_1(x)}$$

$$h(x) \equiv c_2(x) \pmod{m_2(x)}$$

⋮

$$h(x) \equiv c_k(x) \pmod{m_k(x)}$$

具有唯一解, $h(x) \equiv h_0(x) \pmod{(m_1(x)m_2(x)\cdots m_k(x))}$ 。

其中

$$h_0(x) = M_1(x)M'_1(x)c_1(x) + M_2(x)M'_2(x)c_2(x) + \cdots + M_k(x)M'_k(x)c_k(x)$$

$$m_1(x)m_2(x)\cdots m_k(x) = M_s(x)m_s(x)$$

$$M_s(x)M'_s(x) \equiv 1 \pmod{m_s(x)} \quad s = 1, 2, \dots, k$$

该定理的证明与整数环上孙子定理的证明完全相似, 这里不再重复。

例 4.21 已知 $m_1(x) = x^4 + x + 1$, $m_2(x) = x^2 + x + 1$, $c_1(x) = x^3 + x + 1$, $c_2(x) = x + 1$. 解同余组:

$$h(x) \equiv x^3 + x + 1 \pmod{m_1(x) = x^4 + x + 1}$$

$$h(x) \equiv x + 1 \pmod{m_2(x) = x^2 + x + 1}$$

$$h(x) \equiv h_0(x) \pmod{m_1(x)m_2(x) = x^6 + x^5 + x^4 + x^3 + 1}$$

$$h_0(x) = M_1(x)M'_1(x)c_1(x) + M_2(x)M'_2(x)c_2(x)$$

$$M_1(x) = x^2 + x + 1$$

$$M_2(x) = x^4 + x + 1$$

$$(M_1(x), x^4 + x + 1) = (x^2 + x + 1, x^4 + x + 1) = 1$$

由欧几里德算法可得:

$$1 = 1(x^4 + x + 1) + (x^2 + x)(x^2 + x + 1)$$

$$M_1(x)M'_1(x) \equiv 1 \pmod{m_1(x) = x^4 + x + 1}$$

所以

$$M_1(x)M'_1(x) = (x^2 + x + 1)(x^2 + x) \equiv 1 \pmod{x^4 + x + 1}$$

$$M_2(x)M'_2(x) = (x^4 + x + 1) \cdot 1 \equiv 1 \pmod{x^2 + x + 1}$$

$$\begin{aligned} h_0(x) &= (x^2 + x + 1)(x^2 + x)(x^3 + x + 1) + (x^4 + x + 1) \cdot 1(x + 1) \\ &= x^7 + x^5 + x^2 + x + x^5 + x^4 + x^2 + 1 \\ &= x^7 + x^4 + x + 1 \end{aligned}$$

$$h(x) \equiv h_0(x) \equiv x^4 + x^3 \pmod{(x^4 + x + 1)(x^2 + x + 1)}$$

把 $h(x) = x^4 + x^3$ 代入同余组验证, 可知解是正确的。 ■

习 题

1. 在模 8 的剩余类环中找出所有子环、理想和主理想。

2. 求 GF(2) 上多项式的最大公因子

$$(x^2 + x + 1, x^4 + x^3 + x^2 + 1) = C(x)$$

并将它表示成 $A(x)(x^2 + x + 1) + B(x)(x^4 + x^3 + x^2 + 1) = C(x)$ 的形式。

3. 构造 GF(2) 上模 $f(x) = x^3 + x + 1$ 多项式剩余类环, 并列出加法表和乘法表。

4. 在模 $x^6 - 1 \in \text{GF}(2)[x]$ 多项式剩余类环中有几个真理想? 并找出它们的生成元。

5. 构造 GF(7) 的加法表和乘法表? 找出每一个元素的级, 找出哪些元素是生成元。

6. 模 9 剩余加群是否是循环群? 若是, 则确定每个元素的级, 找出所有生成元。

7. 若 $G(\alpha)$ 群的阶数为素数, 问有几个有限循环子群?
8. 基于 $GF(2)$ 上的多项式 $p(x) = x^5 + x^2 + 1$, 构造 $GF(2^5)$ 的加法表和乘法表。令 α 是 $GF(2^5)$ 的本原域元素, 求 α^3 和 α^7 的最小多项式。
9. 分解 $GF(2)$ 上的多项式: $x^{63}-1$ 和 $x^{21}-1$, 为 $GF(2)$ 上的既约多项式之积。求出它们的分圆多项式。
10. 能完全分解 $x^{17}-1$ 为 $GF(2)$ 上既约多项式的次数各为多少? 能完全分解 $x^{17}-1$ 为一次因式的最小分离域是什么?
11. 求出 $GF(2)$ 上次数 ≤ 5 次的全部既约多项式。
12. 设 α 是 $GF(2)$ 上四次既约多项式 $p(x) = x^4 + x^3 + x^2 + x + 1$ 在扩域 $GF(2^4)$ 上之根。试求 $\alpha+1$ 的最小多项式, 并判断这一最小多项式是否为本原多项式?
13. 试证明互反多项式的 3 个性质。
14. 证明次数 > 2 的任何多项式, 如果互反多项式也是自身, 则它不可能是本原多项式。
15. 证明定理 4.4.2(提示: 在 $GF(q)$ 中找一个有最大级为 r 的元素, 证明域中的一切元素的级都是 r 的因子)。
16. 证明推论 4.5.2。
17. 证明推论 4.5.6。
18. 找出 $GF(2^{16})$ 及 $GF(2^{20})$ 中的所有真子域, 并用图表示之。
19. 找出 $GF(2^4)$ 中元素的正规基表示, α 是 x^4+x^3+1 的根, 并找出它的对偶基。
20. 找出 $GF(2^4)$ 中自然基的对偶基, 其中, α 是本原元, 它是 x^4+x+1 的根。
21. 证明定理 4.7.1 中迹的前四个性质。
22. $\alpha \in GF(q^m)$ 是任一元素, 定义 $N(\alpha) = \alpha\alpha^q \cdots \alpha^{q^{m-1}}$ 为 α 的范数, 证明范数有以下性质:
 - (1) $N(\alpha) \in GF(q)$;
 - (2) $N(\alpha, \beta) = N(\alpha)N(\beta)$;
 - (3) $N(\lambda\alpha) = \lambda N(\alpha) \quad \lambda \in GF(q)$;
 - (4) $N(\alpha^q) = N(\alpha)$ 。

参 考 文 献

- [1] 肖国镇:《代数编码引论》,西北电讯工程学院,1975.
- [2] W. W. Peterson, Error-Correcting Codes, MIT Press, 1961.
- [3] 万哲光:《代数与编码》,科学出版社,1976.
- [4] E. R. Berlekamp, Algebraic Coding Theory, Park Press, 1984.
- [5] R. J. McEliece, Finite Fields for Computer Scientists and Engineers, Kluwer Academic, 1987.
- [6] 今井秀樹, 符号理論, 電子情報通信学会, 1990.

第五章 循 环 码

循环码是一类最重要的线性码。由于它具有以下性质：

(1) 循环码具有严谨的代数结构，其性能易于分析。特别是目前已发现的大部分线性码与循环码有密切关系，它们之中的大部分码都可归结于循环码。

(2) 循环码具有循环特性，编译码电路，特别是编码电路简单易于实现。因此循环码特别引人注目，对它的研究也比较深入和系统。

本章先讨论循环码的基本概念、谱特性、平方剩余码以及循环码的一般编码原理和电路。

§ 5.1 循环码与理想

一、基本概念

[7, 4]汉明码 C_H 的 H 矩阵为

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

由第二章可知，交换 H 矩阵各列，并不会影响码的纠错能力。把上述 H 矩阵的列进行交换后变为

$$H = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

由此矩阵可明显看出，第二行是第一行循环右移一位得到，第三行是第二行循环右移一位。由此矩阵编出的 16 个码字为：1000110, 0100011, 1010001, 1101000, 0110100, 0011010, 0001101; 1001011, 1100101, 1110010, 0111001, 1011100, 0101110, 0010111; 1111111; 0000000。由这些码字看出，若 $\mathbf{c}_1 \in C_H$ ，则它的右(左)移循环移位所得到的 n 重也是一个码字，具有这种特性的 $[n, k]$ 分组码称为循环码。由于 $[n, k]$ 线性分组码是 n 维线性空间 V_n 中的一个 k 维子空间，因此 $[n, k]$ 循环码是 n 维线性空间中的一个 k 维循环子空间。

定义 5.1.1 一个 n 重子空间 $V_{n,k} \in V_n$ ，若对任何一个 $\mathbf{v} = (a_{n-1}, a_{n-2}, \dots, a_0) \in V_{n,k}$ ，恒有 $\mathbf{v}_1 = (a_{n-2}, a_{n-3}, \dots, a_0, a_{n-1}) \in V_{n,k}$ ，则称 $V_{n,k}$ 为循环子空间或循环码。

二、码的多项式描述

从第二章可知， $GF(p)$ 上的所有 n 重构成一个线性空间 V_n ，其中每个矢量是分量取自

$\text{GF}(p)$ 上 n 重, 若将每个 n 重和系数取自 $\text{GF}(p)$ 上的多项式相对应:

$$n \text{ 重: } (a_{n-1}, a_{n-2}, \dots, a_1, a_0) \quad a_i \in \text{GF}(p)$$

$$\text{多项式: } (a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \dots + a_1x + a_0) = f(x)$$

则它们之间建立了一一对应关系。在第四章中已指出, 所有次数小于 n 次的多项式一定在模 n 次多项式 $F(x) \in F_p[x]$ 的不同剩余类中, 即

$$f(x) \in \{a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \dots + a_1x + a_0\} (\bmod F(x))$$

因此, V_n 中每一个 n 重都与 $\text{GF}(p)$ 上的次数低于 n 次的一个多项式相对应, 并必在模 $F(x)$ 的某一剩余类中。第四章中已证明, 在模 $F(x)$ 运算下, 模 $F(x)$ 的剩余类构成一多项式剩余类环 $F_p[x]/F(x)$, 若在该环中再定义一个数乘, 即

$$ca(x) = \{ca_{n-1}x^{n-1} + ca_{n-2}x^{n-2} + \dots + ca_0\} \quad c \in \text{GF}(p)$$

则可以证明模 $F(x)$ 的剩余类构成一个 n 维线性空间, 称为剩余类线性结合代数。

在 $[n, k]$ 循环码中, 码字 $(a_{n-1}, a_{n-2}, \dots, a_1, a_0)$ 的多项式表示为 $a(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \dots + a_1x + a_0$ 。它的循环移位一次后所得码字为 $(a_{n-2}, \dots, a_0, a_{n-1})$, 相应的码多项式表示为

$$a_1(x) = a_{n-2}x^{n-1} + \dots + a_0x + a_{n-1}$$

相当于 $a(x)$ 乘以 x 后, 用 $F(x) = x^n - 1$ 取模:

$$\begin{aligned} xa(x) &= a_{n-1}x^n + a_{n-2}x^{n-1} + \dots + a_0x \\ &\equiv a_{n-2}x^{n-1} + \dots + a_0x + a_{n-1} (\bmod x^n - 1) \end{aligned}$$

因为在模 $x^n - 1$ 情况下, $x^n \equiv 1$ 。所以

$$\overline{xa(x)} = \overline{a_{n-2}x^{n-1} + \dots + a_0x + a_{n-1}}$$

若在 $\{a_{n-2}x^{n-1} + \dots + a_0x + a_{n-1}\}$ 剩余类中, 以最低次数 $a_{n-2}x^{n-1} + \dots + a_0x + a_{n-1}$ 多项式作为该类的代表元, 则循环码就可用 $\bmod x^n - 1$ 的多项式表示。

定理 5.1.1 以多项式 $x^n - 1$ 为模的剩余类线性结合代数中, 其一个子空间 $V_{n, k}$ 是一个循环子空间(循环码)的充要条件是: $V_{n, k}$ 是一个理想。

证明 已知 $V_{n, k}$ 是一个循环子空间, 所以对任何一个 $v = (a_{n-1}, a_{n-2}, \dots, a_1, a_0) \in V_{n, k}$, 恒有 $v_1 = (a_{n-2}, a_{n-3}, \dots, a_0, a_{n-1}) \in V_{n, k}$ 。相应地模 $x^n - 1$ 多项式剩余类表示为:

$$\begin{aligned} \overline{v(x)} &: \quad \{a_{n-1}x^{n-1} + \dots + a_1x + a_0\} = v(x) \in V_{n, k} \\ \overline{xv(x)} &: \quad \{a_{n-2}x^{n-1} + \dots + a_0x + a_{n-1}\} = v_1(x) \in V_{n, k} \\ \overline{x^2v(x)} &: \quad \{a_{n-3}x^{n-1} + \dots + a_{n-1}x + a_{n-2}\} = v_2(x) \in V_{n, k} \\ &\vdots \\ \overline{x^{n-1}v(x)} &: \quad \{a_0x^{n-1} + \dots + a_2x + a_1\} = v_{n-1}(x) \in V_{n, k} \end{aligned}$$

而这些元素的线性组合

$$c_0v + c_1xv + \dots + c_{n-1}x^{n-1}v = \{c_0 + c_1x + \dots + c_{n-1}x^{n-1}\}v \in V_{n, k}$$

式中, $c_i \in \text{GF}(p)$ 。因此, $V_{n, k}$ 是一个理想。

反之, 若 $V_{n, k}$ 是一个理想, 则对任何一个 $v = (a_{n-1}, \dots, a_1, a_0) \in V_{n, k}$, 或模 $x^n - 1$ 的剩余类:

$$\overline{v(x)}: \{a_{n-1}x^{n-1} + \cdots + a_1x + a_0\} = \overline{v(x)} \in V_{n,k}$$

恒有

$$\langle x \rangle \{a_{n-1}x^{n-1} + \cdots + a_1x + a_0\} = \{a_{n-2}x^{n-1} + \cdots + a_0x + a_{n-1}\} \in V_{n,k}$$

所以, $(a_{n-2}, \dots, a_0, a_{n-1}) \in V_{n,k}$ 中, 故 $V_{n,k}$ 是一个循环子空间。 ■

由上面定理可得如下重要结论: 一个循环码是模 $x^n - 1$ 多项式剩余类线性结合代数中的一个理想。反之, 其中的一个理想必是一个循环码。

从理想的定义可知, 理想必是由某一元素的倍数所构成。因此, 至少可在理想中找到一个次数最低的非零首一多项式 $g(x)$, 由它的一切倍式生成一个理想(循环码)。我们称此多项式为码的生成多项式, 所有码多项式必是它的倍式。

定义 5.1.2 生成多项式 $g(x)$ 是模 $x^n - 1$ 剩余类代数中, 一个理想的次数最低的非零首一多项式, 它是理想或循环码的生成元。

定理 5.1.2 $\text{GF}(q)$ (q 为素数或素数的幂) 上的 $[n, k]$ 循环码中, 存在有唯一的 $n-k$ 次首一多项式 $g(x) = x^{n-k} + g_{n-k-1}x^{n-k-1} + \cdots + g_1x + g_0$, 每一码多项式 $C(x)$ 都是 $g(x)$ 的倍式, 且每一个 $\leq (n-1)$ 次的 $g(x)$ 倍式一定是码多项式。

证明 令

$$g(x) = x^r + g_{r-1}x^{r-1} + \cdots + g_1x + g_0$$

是 $[n, k]$ 循环码中次数最低的非零首一码多项式, 由于码的循环特性, $xg(x), x^2g(x), \dots, x^{n-1-r}g(x)$ 也必是码多项式。因为循环码是线性码, 所以 $g(x), xg(x), \dots, x^{n-1-r}g(x)$ 的线性组合

$$\begin{aligned} & m_{n-1-r}g(x)x^{n-1-r} + m_{n-1-r-1}g(x)x^{n-1-r-1} + \cdots + m_1xg(x) + m_0g(x) \\ &= g(x)(m_{n-1-r}x^{n-1-r} + m_{n-1-(r+1)}x^{n-1-(r+1)} + \cdots + m_1x + m_0) \\ &= g(x)M(x) \end{aligned}$$

$$m_i \in \text{GF}(q), \quad i = 0, 1, \dots, n-1-r$$

也必在循环码中, 是一个码多项式。所以, 每一次数 $\leq (n-1)$ 次的 $g(x)$ 倍式必是码多项式。

反之, 若 $C(x)$ 是一码多项式, 则由欧几里德除法可知:

$$C(x) = q(x)g(x) + r(x) \quad 0 \leq \deg r(x) < \deg g(x) \quad \text{或} \quad r(x) = 0$$

$$r(x) = C(x) - q(x)g(x)$$

由于是线性码, 所以 $C(x) - q(x)g(x) = r(x)$ 也必是码多项式。但 $\deg r(x) < \deg g(x)$, 这与 $g(x)$ 是码多项式集合中的次数最低的假设相矛盾, 故 $r(x) = 0$, 所以

$$C(x) = q(x)g(x)$$

设 $g(x)$ 不是唯一的, 还有一个同次数的非零首一码多项式 $g'(x) = x^r + g'_{r-1}x^{r-1} + \cdots + g'_1x + g'_0$, 则它与 $g(x)$ 的线性组合

$$g(x) - g'(x) = (g_{r-1} - g'_{r-1})x^{r-1} + \cdots + (g_1 - g'_1)x + (g_0 - g'_0)$$

也必是一个码多项式, 但 $\deg(g(x) - g'(x)) < \deg g(x)$, 这与 $g(x)$ 是最低次数的假设相矛盾, 所以只有

$$g(x) - g'(x) = 0 \quad g(x) = g'(x)$$

最后, 我们证明 $\partial \circ g(x) = n - k$ 。我们知道

$$\{g(x)\}, \{xg(x)\}, \dots, \{x^{n-1}g(x)\}$$

是次数小于 n 的线性无关的剩余类, 它们的任意线性组合

$$\{(m_{n-1-r}x^{n-1-r} + \dots + m_1x + m_0)g(x)\} \quad m_i \in GF(q)$$

也必是次数小于 n 的多项式剩余类, 故 $(m_{n-1-r}, \dots, m_1, m_0)$ 中的数字不能全为 0, 可知由 $g(x)$ 生成的子空间(即理想)有 q^{n-r} 个元素, 其维数为 $n-r$ 。而在 $[n, k]$ 循环码中共有 q^k 个码字, 该子空间的维数是 k 维, 所以 $k=n-r$, $r=n-k$ 。因此, $g(x)$ 的次数等于 $n-k$ 。由 $g(x)$ 生成的理想即为 $k=n-r$ 维。 ■

由此定理可知, $[n, k]$ 线性码是循环码的充要条件是: 它是模 $x^n - 1$ 多项式剩余类线性结合代数中的一个理想。且理想中只有一个次数为 $n-k$ 的生成元 $g(x)$, 所以这是一个主理想, 理想的维数即循环码的信息元个数等于 k 。

下面讨论主理想生成元 $g(x)$ 应满足的条件。

定理 5.1.3 $GF(q)$ 上 $[n, k]$ 循环码的生成多项式 $g(x)$ 一定是 $x^n - 1$ 的因式: $x^n - 1 = g(x)h(x)$ 。反之, 若 $g(x)$ 为 $n-k$ 次, 且 $g(x) | (x^n - 1)$, 则该 $g(x)$ 一定生成一个 $[n, k]$ 循环码。

证明 $g(x)$ 所生成的主理想是在多项式剩余类环 $F_q[x]/(x^n - 1)$ 中, 由定理 4.2.10 可知, $g(x) | (x^n - 1)$, 所以 $g(x)h(x) = x^n - 1$ 。

下面证明定理的第二部分。考虑 k 个多项式: $g(x), xg(x), \dots, x^{k-1}g(x)$, 它们的次数均 $\leq n-1$ 次, 且线性独立, 作这些多项式的线性组合

$$\begin{aligned} & m_0g(x) + m_1xg(x) + \dots + m_{k-1}x^{k-1}g(x) \\ &= (m_0 + m_1x + \dots + m_{k-1}x^{k-1})g(x) \quad m_i \in GF(q) \end{aligned}$$

因为 $m_{k-1}x^{k-1}g(x), \dots, m_0g(x)$ 都是 $g(x)$ 的倍式, 所以都在 $g(x)$ 生成的理想中, 而它们的线性组合也显然在该理想中。 $(m_0 + m_1x + \dots + m_{k-1}x^{k-1}) = m(x)$ 的次数等于 $k-1$, 且 (m_{k-1}, \dots, m_0) 中不能全为 0, 所以由 $g(x)$ 生成的理想必是 k 维的, 故 $g(x)$ 生成一个 $[n, k]$ 线性码。由于 $g(x) | (x^n - 1)$, 它是 $x^n - 1$ 的一个因子, 因此由该 $g(x)$ 生成之主理想, 由定理 5.1.1 可知必是一个循环码。 ■

由上面这些定理可得出如下一些结论:

(1) 找一个 k 维循环码, 就是找一个能除尽 $x^n - 1$ 的 $n-k$ 次首一多项式 $g(x)$, 由此 $g(x)$ 作生成元, 生成一个主理想;

(2) 若 $C(x)$ 是一个码组, 则 $g(x) | C(x)$; 反之亦然。

例 5.1 在 $GF(2)$ 上, $x^7 - 1 = (x+1)(x^3+x+1)(x^3+x^2+1)$, 求 $[7, 4]$ 循环码。

找一个能除尽 $x^7 - 1$ 的 $n-k=3$ 次首一多项式 $g(x)$, 可在 x^3+x+1 与 x^3+x^2+1 中任选一个, 现在选 $g(x) = x^3+x^2+1$, 则

$$\begin{aligned} \{xg(x)\}: \quad & xg(x) = x^4 + x^3 + x \\ \{x^2g(x)\}: \quad & x^2g(x) = x^5 + x^4 + x^2 \\ \{x^3g(x)\}: \quad & x^3g(x) = x^6 + x^5 + x^3 \end{aligned}$$

与它们相应的 n 重为: (0001101), (0011010), (0110100), (1101000), 把它们作为生成矩阵的行, 就得到了 $[7, 4]$ 码的生成矩阵。

$$G = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

■

三、循环码的生成矩阵、校验矩阵

由前知, $x^n - 1 = g(x)h(x)$ 。若 $g(x)$ 为 $n-k$ 次, 则 $h(x)$ 为 k 次多项式。以 $g(x)$ 作为生成多项式所组成的 $[n, k]$ 循环码中 $g(x), xg(x), \dots, x^{k-1}g(x)$ 等 k 个码多项式必是线性无关的, 设可以由这些码多项式所对应的码字, 构成循环码的生成矩阵 G , 则

$$\begin{aligned} g(x) &= g_{n-k}x^{n-k} + g_{n-k-1}x^{n-k-1} + \dots + g_1x + g_0 \\ xg(x) &= g_{n-k}x^{n-k+1} + g_{n-k-1}x^{n-k} + \dots + g_1x^2 + g_0x \\ &\vdots \\ x^{k-1}g(x) &= g_{n-k}x^{n-1} + g_{n-k-1}x^{n-2} + \dots + g_0x^{k-1} \end{aligned}$$

所以

$$G = \left[\begin{array}{ccccccccc} g_{n-k} & g_{n-k-1} & \cdots & g_1 & g_0 & 0 & \cdots & \underbrace{0}_{(k-1) \uparrow 0} \\ 0 & g_{n-k} & \cdots & g_2 & g_1 & g_0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & g_{n-k} & g_{n-k-1} & \cdots & g_1 & g_0 & k \times n \end{array} \right] \quad (5.1.1)$$

$$\begin{aligned} x^n - 1 &= g(x)h(x) \\ &= (g_{n-k}x^{n-k} + \dots + g_1x + g_0)(h_kx^k + \dots + h_1x + h_0) \end{aligned}$$

由此可知等式右边的 $x^{n-1}, x^{n-2}, \dots, x$ 的系数均为 0, 即

$$\left. \begin{array}{l} g_0h_0 = -1 \\ g_0h_1 + g_1h_0 = 0 \\ \vdots \\ g_0h_i + g_1h_{i-1} + \dots + g_{n-k}h_{i-(n-k)} = 0 \\ \vdots \\ g_0h_{n-1} + g_1h_{n-2} + \dots + g_{n-k}h_{k-1} = 0 \\ g_{n-k}h_k = 1 \end{array} \right\} \quad (5.1.2)$$

上式可简写成

$$\left\{ \begin{array}{l} g_0h_i + g_1h_{i-1} + \dots + g_{n-k}h_{i-(n-k)} = 0 \quad i = 1, 2, \dots, n-1 \\ g_0h_0 + g_{n-k}h_k = 0 \end{array} \right.$$

因此 $[n, k]$ 循环码的一致校验矩阵

$$H = \begin{bmatrix} h_0 & h_1 & \cdots & h_k & 0 & \cdots & 0 \\ 0 & h_0 & h_1 & \cdots & h_k & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & h_0 & h_1 & \cdots & h_k \end{bmatrix} \underbrace{(n-k-1) \uparrow 0}_{(n-k-1) \uparrow 0} \quad (n-k) \times n \quad (5.1.3)$$

容易验证

$$G \cdot H^T = 0 \quad (5.1.4)$$

所以，我们称 $h(x) = (x^n - 1)/g(x)$ 为码的校验多项式，由式(5.1.3)可以看出， H 矩阵的行完全由 $h(x)$ 的系数决定。

式(5.1.1) G 矩阵的每行数据，是按照 $x^{k-1}g(x), \dots, g(x)$ 多项式的最高次至最低次系数的次序排列的。如果为了使 H 矩阵的每行数据，也按此规则排列，可作 $h(x)$ 的互反多项式

$$h^*(x) = h_0x^k + h_1x^{k-1} + \cdots + h_{k-1}x + h_k$$

以 $x^{n-k-1}h^*(x), x^{n-k-2}h^*(x), \dots, xh^*(x), h^*(x)$ 多项式的自高次至低次的系数次序，可得到 H 矩阵的每一行。

如例 5.1 中 [7, 4] 码的校验多项式

$$h(x) = \frac{x^7 - 1}{g(x)} = \frac{x^7 - 1}{x^3 + x^2 + 1} = x^4 + x^3 + x^2 + 1$$

$$h^*(x) = x^4 + x^2 + x + 1$$

相应的 H 矩阵为

$$H = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

在模 $x^n - 1$ 剩余类代数中， $x^n - 1 = g(x)h(x) \equiv 0$, $\partial \circ g(x) = n - k$, $\partial \circ h(x) = k$ 。若以 $h^*(x)$ 作为主理想的生成元，则 $\{h^*(x)\}, \dots, \{x^{n-k-1}h^*(x)\}$ 就是该主理想的一组基底，所以主理想的维数是 $n - k$ ，由此主理想生成了一个 $[n, n - k]$ 循环码。从式(5.1.2)、式(5.1.3)和式(5.1.4)可知，此主理想与 $g(x)$ 生成之主理想互为零空间。因而由校验多项式 $h(x)$ 的互反多项式 $h^*(x)$ 生成的 $[n, n - k]$ 码，与由生成多项式 $g(x)$ 生成的 $[n, k]$ 码互为对偶码。

定理 5.1.4 令 C_1 和 C_2 分别由 $g_1(x)$ 和 $g_2(x)$ 生成的两个不同的循环码，则当且仅当 $g_2(x) | g_1(x)$ 时， $C_1 \subseteq C_2$ 。

证明 若 $g_2(x) | g_1(x)$ ，则 $g_1(x) = a(x)g_2(x)$ ，也就是由 $g_1(x)$ 的所有倍式生成的循环码 C_1 必在由 $g_2(x)$ 生成的码 C_2 中，所以 $C_2 \supseteq C_1$ 。反之，若 $C_2 \supseteq C_1$ ，则 C_1 的每一个码字不但是 $g_1(x)$ 的倍式，也必是 $g_2(x)$ 的倍式， $g_2(x) | g_1(x)m_1(x)$ 。若 $g_2(x) | m_1(x)$ ，则 $m_1(x)$ 是 C_2 码的一个码字，因而 C_1 码的每一码字 $g_1(x)m_1(x)$ 必包含有 C_2 码的码字， $C_1 \supseteq C_2$ ，这与 $C_1 \subseteq C_2$ 的假设相矛盾，所以 $g_2(x) \nmid m_1(x)$ ，可知 $g_2(x) | g_1(x)$ 。

该定理给出了 C_1 是 C_2 码子码的充要条件。 ■

四、系统码的构成

用式(5.1.1)矩阵生成的循环码，并不是系统码。系统码的 G 矩阵为

$$G = [\mathbf{I}_k \mathbf{p}]$$

左边是 $k \times k$ 阶单位方阵。这相当于码字多项式的第 $n-1$ 次至 $n-k$ 次的系数是信息位，而其余的为校验位，这相当于

$$C(x) = m(x)x^{n-k} + r(x) \equiv 0 \pmod{g(x)} \quad (5.1.5)$$

式中

$$m(x) = m_{k-1}x^{k-1} + m_{k-2}x^{k-2} + \cdots + m_1x + m_0$$

是信息多项式， $(m_{k-1}, \dots, m_1, m_0)$ 是信息位，而

$$r(x) = r_{n-k-1}x^{n-k-1} + r_{n-k-2}x^{n-k-2} + \cdots + r_1x + r_0$$

是校验位多项式，相应的系数是码元的校验位。由上式可得

$$-r(x) = C(x) + m(x)x^{n-k} \equiv m(x)x^{n-k} \pmod{g(x)} \quad (5.1.6)$$

所以要构造用 $g(x)$ 生成的系统码，首先必须将信息组乘以 x^{n-k} 变成 $x^{n-k}m(x)$ ；然后，用 $g(x)$ 除，得到余式 $r(x)$ ；再将其各项系数取加法逆元，就得到了所要求的校验位。因此，循环码系统码的编码问题就是以 $g(x)$ 为模的除法问题。

由 $G = [\mathbf{I}_k \mathbf{p}]$ 可知，信息组的基底矢量是： $(100\dots0), (010\dots0), \dots, (00\dots01)$ ，相应的信息多项式分别为： $m_1(x) = x^{k-1}, m_2(x) = x^{k-2}, \dots, m_k(x) = 1$ 。与这些信息多项式相应的校验多项式分别为： $r_1(x) \equiv x^{n-k}x^{k-1} \pmod{g(x)}, r_2(x) \equiv x^{n-k}x^{k-2} \pmod{g(x)}, \dots, r_k(x) \equiv x^{n-k} \pmod{g(x)}$ 。一般写成

$$r_i \equiv x^{n-k}x^{k-i} \equiv x^{n-i} \pmod{g(x)} \quad i = 1, 2, \dots, k$$

与此相应的码多项式为

$$g(x)q_i(x) = C_i(x) = x^{n-i} - r_i(x) \quad i = 1, 2, \dots, k$$

共 k 个，它们的系数就组成了系统码 G 矩阵的行。

$$G = \begin{bmatrix} 1 & 0 & \cdots & 0 & -\tilde{r}_1(x) \\ 0 & 1 & 0 & \cdots & 0 & -\tilde{r}_2(x) \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -\tilde{r}_k(x) \end{bmatrix} = [\mathbf{I}_k : \mathbf{R}] \quad (5.1.7)$$

式中， $\tilde{r}_i(x)$ 表示 $r_i(x)$ 的系数。因而，校验矩阵 H 可由上式立即得到

$$H = [-\mathbf{p}^T \mathbf{I}_{n-k}] = [\tilde{r}_1(x)^T \quad \tilde{r}_2(x)^T \quad \cdots \quad \tilde{r}_k(x)^T \quad \mathbf{I}_{n-k}] \quad (5.1.8)$$

这说明 H 矩阵的前 k 列是 $r_i(x) \equiv x^{n-i} \pmod{g(x)} (i = 1, 2, \dots, k)$ 的系数。而后 $n-k$ 列是 $x^{n-k-1}, x^{n-k-2}, \dots, x^0$ 的系数，由于它们的次数均小于 $\deg(g(x))$ ，所以它们被 $g(x)$ 取模以后仍不变。因此可把式(5.1.8)写成

$$H = [\tilde{x}^{T_{n-1}}, \tilde{x}^{T_{n-2}}, \dots, \tilde{x}^{T_{n-k}}, \tilde{x}^{T_{n-k-1}}, \dots, \tilde{x}^T, 1^T] \pmod{g(x)} \quad (5.1.9)$$

式(5.1.7)和式(5.1.9)就是循环码的系统码形式的 G 矩阵和 H 矩阵的一般表示式。

例 5.2 二进制 $[7, 4]$ 码的 $g(x) = x^3 + x^2 + 1$ ，求系统码的 G 和 H 矩阵。

$$r_1(x) \equiv x^6 \equiv x^2 + x \pmod{g(x)}$$

$$r_2(x) \equiv x^5 \equiv x + 1 \pmod{g(x)}$$

$$r_3(x) \equiv x^4 \equiv x^2 + x + 1 \pmod{g(x)}$$

$$r_4(x) \equiv x^3 \equiv x^2 + 1 \pmod{g(x)}$$

在 $\text{GF}(2)$ 上, 元素的逆元就是它自己, 所以

$$\begin{aligned} G &= \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} = [\mathbf{I}_k \mathbf{p}] \\ H &= \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} = [-\mathbf{p}^T \mathbf{I}_{n-k}] \\ &= [\tilde{x}^6 \ \tilde{x}^5 \ \tilde{x}^4 \ \tilde{x}^3 \ \tilde{x}^2 \ \tilde{x}^1 \ \tilde{x}^0] \pmod{g(x)} \quad \blacksquare \end{aligned}$$

§ 5.2 由生成多项式的根定义循环码

循环码是模 $x^n - 1$ 剩余代数中的一个以 $g(x)$ 作生成元的理想, 每一个码多项式都是 $g(x)$ 的倍式。因此 $g(x)$ 的根亦必是所有码字多项式的根, 基于这点, 我们可以从根来定义循环码。

设码的生成多项式

$$g(x) = x^r + g_{r-1}x^{r-1} + \cdots + g_1(x) + g_0 \quad g_i \in \text{GF}(q)$$

它必在某一个 $\text{GF}(q)$ 的扩域上完全分解, 即它的全部根必在此扩域上。

关于 $g(x)$ 的根有两种情况: 一是 $g(x)$ 无重根, 一是有重根。由于有重根情况下用 $g(x)$ 生成码, 比无重根时生成的码除个别码外通常要差^[7], 故下面我们仅讨论 $g(x)$ 无重根的情况。为此, 首先要找出 $g(x)$ 无重根的条件。

$g(x) | (x^n - 1)$, 或 $g(x)h(x) = x^n - 1$ 。若 $g(x)$ 有 r_i 个重根, 则 $x^n - 1$ 也必含有 r_i 个重根。因此要保证 $g(x)$ 无重根, 首先必须要求 $x^n - 1$ 无重根。

定理 5.2.1 在 $\text{GF}(q)$ 上多项式 $x^n - 1$ 无重根的充要条件是 $(n, q) = 1$ 。

证明 若 n, q 不互素, 则 $n = n_1 q^s$, $x^n - 1 = x^{n_1 q^s} - 1$, 由定理 4.5.4 可知 $x^{n_1 q^s} - 1 = (x^{n_1} - 1)^{q^s} = (g_1(x)h_1(x))^{q^s} = g_1(x)^{q^s}h_1(x)^{q^s} = g(x)h(x)$ 。由此可知, 若 $(n, q) \neq 1$, 则 $x^n - 1$ 必有重根, $g(x)$ 和 $h(x)$ 也必有重根。反之, 若 $(n, q) = 1$, 则 $x^n - 1 \neq x^{n_1 q^s} - 1$, 因而 $g(x)h(x)$ 无重根。■

可见在 $\text{GF}(2)$ 上要保证 $g(x)$ 无重根的条件是 $x^n - 1 = g(x)h(x)$ 中的 n 是奇数, 因此在二进制循环码中, 码长 n 是奇数。

下面讨论无重根时, 用 $g(x)$ 的根定义循环码。 $g(x)$ 必在 $\text{GF}(q)$ 的扩域 $\text{GF}(q^m)$ 上完全分解

$$g(x) = (x - \alpha_1)(x - \alpha_2) \cdots (x - \alpha_r) \quad \alpha_i \neq \alpha_j, \quad i, j = 1, 2, \dots, r$$

$\alpha_i \in \text{GF}(q^m)$ 。可知, $g(x)$ 有根 $\alpha_1, \alpha_2, \dots, \alpha_r$ 。 $g(x)$ 产生的循环码的每一码多项式必是 $g(x)$ 的倍式, 因此每一码多项式 $C(x)$ 也必以 $\alpha_1, \alpha_2, \dots, \alpha_r$ 为根。设码多项式

$$C(x) = c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \cdots + c_1x + c_0$$

则

$$C(\alpha_i) = c_{n-1}\alpha_i^{n-1} + c_{n-2}\alpha_i^{n-2} + \cdots + c_1\alpha_i + c_0 = 0 \quad i = 1, 2, \dots, r$$

写成矩阵形式

$$\begin{bmatrix} \alpha_1^{n-1} & \alpha_1^{n-2} & \cdots & \alpha_1 & 1 \\ \alpha_2^{n-1} & \alpha_2^{n-2} & \cdots & \alpha_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \alpha_r^{n-1} & \alpha_r^{n-2} & \cdots & \alpha_r & 1 \end{bmatrix} \begin{bmatrix} c_{n-1} \\ c_{n-2} \\ \vdots \\ c_1 \\ c_0 \end{bmatrix} = H \cdot C^T = 0 \quad (5.2.1)$$

所以循环码的 H 矩阵又可写成

$$H = \begin{bmatrix} \alpha_1^{n-1} & \alpha_1^{n-2} & \cdots & \alpha_1 & 1 \\ \alpha_2^{n-1} & \alpha_2^{n-2} & \cdots & \alpha_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \alpha_r^{n-1} & \alpha_r^{n-2} & \cdots & \alpha_r & 1 \end{bmatrix} \quad (5.2.2)$$

这说明若码多项式 $C(x)$ 有 $\alpha_1, \alpha_2, \dots, \alpha_r$ 作为根，则它必在式(5.2.2) H 矩阵的零空间中。

若 α_i 的最小多项式为 $m_i(x), i = 1, 2, \dots, r$ ，则仅只有 $g(x)$ 和 $C(x)$ 同时能被 $m_i(x)$, $m_2(x), \dots, m_r(x)$ 除尽时， $g(x)$ 和 $C(x)$ 才能以 $\alpha_1, \alpha_2, \dots, \alpha_r$ 为根。由于 $g(x)$ 是码多项式中唯一的次数最低的首一多项式，所以

$$g(x) = \text{LCM}(m_1(x), m_2(x), \dots, m_r(x)) \quad (5.2.3)$$

$g(x) | (x^n - 1)$ ，因此 $\alpha_1, \alpha_2, \dots, \alpha_r$ 也必是 $x^n - 1$ 的根。所以，每个 $\alpha_i (i = 1, 2, \dots, r)$ 的级必除尽 n ，由此可知循环码的码长

$$n = \text{LCM}(e_1, e_2, \dots, e_r) \quad (5.2.4)$$

式中， e_i 是 α_i 元素的级。

$\alpha_i \in \text{GF}(q^n)$ ，由有限域的理论可知，该域中的每个元素必定是由 $q^n - 1$ 级的元素 α 的幂次生成：若令 $\alpha_i = \alpha^i$ ，且它的最小多项式为 $m_i(x)$ ，则由定理 4.5.6 可知， $\alpha_i^q = \alpha^{iq}$, $\alpha_i^{q^2} = \alpha^{iq^2}, \dots, \alpha_i^{q^{m-1}} = \alpha^{i(q^{m-1})}$ 也是 $m_i(x)$ 的根。因此，在 $g(x)$ 中 $g(x)$ 仅含有共轭根系的最小多项式作为其中的一个因子。

求 $g(x)$ 的关键是找出每个根的最小多项式，4.6 节已介绍了求最小多项式的一般方法，例 4.17 给出了用待定系数法求既约因式的方法。已知根 α_i ，求它的最小多项式 $m_i(x)$ 也可用这种待定系数法以及直接法求得。但是，当 $\text{GF}(q^n)$ 的域很大时，计算就很复杂，因此一般用查表的方法直接得到 $m_i(x)$ 。例如，可查彼得逊(Peterson)所写的《纠错码》一书的附录 C，该附录中已列出 $\text{GF}(2)$ 上小于 35 次的最小多项式表。

例 5.3 求 $\text{GF}(2^4)$ 上以 $\alpha, \alpha^2, \alpha^4$ 为根的循环码。

设 $\alpha \in \text{GF}(2^4)$ 是本原域元素，则它的最小多项式就是本原多项式 $m_1(x) = x^4 + x + 1$ ， $\alpha, \alpha^2, \alpha^4, \alpha^8$ 是它的共轭根系。因此以 $\alpha, \alpha^2, \alpha^4, \alpha^8$ 为根的循环码的生成多项式 $g(x) = x^4 + x + 1$ ，码长 n 就是 α 的级数，等于 $2^4 - 1 = 15$ ，故得到一个 [15, 11] 码，这是一个循环汉明码。

设[15, 11]码的码多项式 $C(x) = c_{14}x^{14} + c_{13}x^{13} + \dots + c_0$, $c_i \in GF(2)$, $i = 0, 1, 2, \dots, 14$ 。则

$$\begin{aligned} C(\alpha) &= c_{14}\alpha^{14} + c_{13}\alpha^{13} + \dots + c_1\alpha + c_0 = 0 \\ C(\alpha^2) &= c_{14}(\alpha^2)^{14} + c_{13}(\alpha^2)^{13} + \dots + c_1\alpha^2 + c_0 = 0 \\ C(\alpha^4) &= c_{14}(\alpha^4)^{14} + c_{13}(\alpha^4)^{13} + \dots + c_1\alpha^4 + c_0 = 0 \\ C(\alpha^8) &= c_{14}(\alpha^8)^{14} + c_{13}(\alpha^8)^{13} + \dots + c_1\alpha^8 + c_0 = 0 \end{aligned}$$

或

$$\begin{bmatrix} \alpha^{14} & \alpha^{13} & \cdots & \alpha & 1 \\ (\alpha^2)^{14} & (\alpha^2)^{13} & \cdots & \alpha^2 & 1 \\ (\alpha^4)^{14} & (\alpha^4)^{13} & \cdots & \alpha^4 & 1 \\ (\alpha^8)^{14} & (\alpha^8)^{13} & \cdots & \alpha^8 & 1 \end{bmatrix} \begin{bmatrix} c_{14} \\ c_{13} \\ \vdots \\ c_1 \\ c_0 \end{bmatrix} = H \cdot C^T = 0 \quad (5.2.5)$$

码多项式 $C(x)$ 的系数在基域 $GF(2)$ 上, 它的根 $\alpha, \alpha^2, \alpha^4, \alpha^8$ 在扩域 $GF(2^4)$ 上, 而 $GF(2^4)$ 上的元素都可用 $GF(2)$ 上的四重表示, 因此, 式(5.2.5)中的 H 矩阵, 若化成 $GF(2)$ 上的元素则共有 16 行。由于 $g(x) = m_1(x)$, $\partial \circ m_1(x) = 4$, 所以由它生成的理想是十一维($15-4=11$)的, 由定理 2.6.10 可知, 它的零空间必是四维的, 这说明 H 矩阵只有 4 行线性无关。下面的定理证明了这一点。

定理 5.2.1 若 H 矩阵的元素均在 $GF(q^n)$ 中, 且第 j 行元素等于第 i 行相应元素的 q 次幂; 则在 $GF(q)$ 中, 由第 j 行元素所组成的 m 行, 与第 i 行元素所组成的 m 行之间, 每行均线性相关。

证明 设 H 的第 i 行元素为 $\alpha^0, \alpha^1, \alpha^2, \dots, \alpha^{n-1}$, 第 j 行元素为 $\alpha^{q^0}, \alpha^{q^1}, \alpha^{q^2}, \dots, \alpha^{q^{n-1}}$ 。令 x 是 $GF(q^n)$ 中的任一元素, α 是本原域元素, 则 x 与 x^q 可用域的自然基表示成

$$x = a_0 + a_1\alpha + a_2\alpha^2 + \dots + a_{m-1}\alpha^{(m-1)} \quad a_i \in GF(q) \quad (5.2.6)$$

$$x^q = b_0 + b_1\alpha + b_2\alpha^2 + \dots + b_{m-1}\alpha^{(m-1)} \quad b_i \in GF(q) \quad (5.2.7)$$

由式(5.2.6), 并根据推论 4.5.3 可得

$$\begin{aligned} x^q &= (a_0 + a_1\alpha + a_2\alpha^2 + \dots + a_{m-1}\alpha^{(m-1)})^q \\ &= a_0^q + a_1^q\alpha^q + a_2^q\alpha^{2q} + \dots + a_{m-1}^q\alpha^{q(m-1)} \\ &= a_0 + a_1\alpha^q + a_2\alpha^{2q} + \dots + a_{m-1}\alpha^{q(m-1)} \end{aligned} \quad (5.2.8)$$

定义

$$\alpha^{iq} = c_{i0} + c_{i1}\alpha + \dots + c_{i,m-1}\alpha^{m-1} \quad c_{ij} \in GF(q), \quad i = 0, 1, \dots, m-1$$

比较式(5.2.7)与式(5.2.8)并利用上式可知, x^q 的系数 $(b_0, b_1, \dots, b_{m-1})$

$$\begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{m-1} \end{bmatrix} = \begin{bmatrix} c_{00} & c_{10} & \cdots & c_{m-1,0} \\ c_{01} & c_{11} & \cdots & c_{m-1,1} \\ \vdots & \vdots & & \vdots \\ c_{0,m-1} & c_{1,m-1} & \cdots & c_{m-1,m-1} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{m-1} \end{bmatrix}$$

完全可由 x 的系数 $(a_0, a_1, \dots, a_{m-1})$ 的线性组合得到。 ■

$\alpha^2, \alpha^4, \alpha^8$ 与 α 有完全相同的最小多项式 $m_1(x)$, 因而它们有完全相同的零空间。而该定理也说明了这一点。所以在式(5.2.5)的 H 矩阵中, 仅只考虑 $\alpha^{14}, \alpha^{13}, \dots, \alpha, 1$ 这一行的 GF(2) 上的四重表示即可。因而 [15, 11] 循环汉明码的 H 为

$$H = [\alpha^{14} \ \alpha^{13} \ \dots \ \alpha \ 1] = \begin{bmatrix} 1 & 1 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ 1 & 1 & \cdots & 0 & 0 \end{bmatrix}$$

式中, $\alpha^{14}, \alpha^{13}, \dots, \alpha$ 和 1 的四重表示见表 4-2(下同)。

一般情况下, 由于共轭根系有相同的最小多项式, 因此由定理 5.2.1 及式(5.2.2)的 H 矩阵在 GF(2^m) 域中可以简化为

$$H = \begin{bmatrix} \alpha_1^{n-1} & \alpha_1^{n-2} & \cdots & \alpha_1 & 1 \\ \alpha_3^{n-1} & \alpha_3^{n-2} & \cdots & \alpha_3 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \alpha_r^{n-1} & \alpha_r^{n-2} & \cdots & \alpha_r & 1 \end{bmatrix} \quad (5.2.9)$$

例 5.4 求以 GF(2^4) 中的 1, $\alpha, \alpha^2, \alpha^4$ 元素为根的二进制循环码。

$1=\alpha^0$ 的最小多项式是 $1+x$, 所以

$$g(x) = (1+x)(x^4 + x + 1) = x^5 + x^4 + x^2 + 1$$

$$n = \text{LCM}(1, 15) = 15$$

得到一个 [15, 10] 码, 该码的 H 矩阵是

$$H = \begin{bmatrix} \alpha^{14} & \alpha^{13} & \alpha^{12} & \cdots & \alpha & 1 \\ 1 & 1 & 1 & \cdots & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ 1 & 1 & \cdots & 0 & 0 \\ 1 & 1 & \cdots & 1 & 1 \end{bmatrix}$$

显然, 这是一个增余删信汉明循环码。 ■

一般情况下, GF(2) 上的循环汉明码是一个 $[2^m-1, 2^m-1-m]$ 码, 它的校验矩阵

$$H = [\alpha^{2^m-2} \ \alpha^{2^m-3} \ \cdots \ \alpha \ 1] \quad \alpha^i \in \text{GF}(2^m) \quad (5.2.10)$$

而增余删信汉明循环码是 $[2^m-1, 2^m-2-m]$ 码, 它的校验矩阵

$$H = \begin{bmatrix} \alpha^{2^m-2} & \alpha^{2^m-3} & \cdots & \alpha & 1 \\ 1 & 1 & \cdots & 1 & 1 \end{bmatrix} \quad (5.2.11)$$

例 5.5 求以 GF(2^4) 中的 $\alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5$ 为根的二进制循环码。

在 GF(2^4) 上, $\alpha, \alpha^2, \alpha^4, \alpha^8$ 是 15 级元素, 它们组成一共轭根系, 其最小多项式是 $m_1(x) = x^4 + x + 1$ 。 $\alpha^3, \alpha^6, \alpha^9, \alpha^{12}$ 都是 5 级元素也是一共轭根系, 它们的最小多项式是 $m_3(x) = x^4 + x^3 + x^2 + x + 1$, α^5, α^{10} 是 3 级元素, 它们组成一共轭根系, 其最小多项式是 $m_5(x) = x^2 + x + 1$ 。所以, 码的生成多项式

$$\begin{aligned} g(x) &= \text{LCM}(m_1(x)m_3(x)m_5(x)) \\ &= (x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1)(x^2 + x + 1) \\ &= x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1 \end{aligned}$$

$$n = \text{LCM}(15, 5, 13) = 15$$

由此得到一个[15, 5]循环码。它的校验矩阵

$$\begin{aligned} \mathbf{H} &= \begin{bmatrix} \alpha^{14} & \alpha^{13} & \cdots & \alpha & 1 \\ (\alpha^3)^{14} & (\alpha^3)^{13} & \cdots & \alpha^3 & 1 \\ (\alpha^5)^{14} & (\alpha^5)^{13} & \cdots & \alpha^5 & 1 \end{bmatrix} \begin{bmatrix} \alpha^{14} & \alpha^{13} & \alpha^{12} & \cdots & \alpha & 1 \\ \alpha^{12} & \alpha^9 & \alpha^6 & \cdots & \alpha^3 & 1 \\ \alpha^{10} & \alpha^5 & 1 & \cdots & \alpha^5 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \\ &\quad \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} \\ &\quad \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{aligned}$$

上式中最后一行全为 0 可以去掉，最后第二行与第三行完全相同，可以去掉一行，因此最后得到的 \mathbf{H} 矩阵有 10 行 15 列。

由于 α, α^3 的最小多项式都是四次的，它们的零空间都是四维的，而 α^5 的最小多项式是二次的，故它的零空间是二维的，因而由 $m_1(x)m_3(x)m_5(x) = g(x)$ 生成的理想零空间是 10 维的，这说明 \mathbf{H} 矩阵有 10 行独立的行。 ■

通常，若 $g(x) = m_1(x)m_2(x)\cdots m_i(x)$ ，如果 $\partial \cdot m_1(x) = r_1, \partial \cdot m_2(x) = r_2, \dots, \partial \cdot m_i(x) = r_i$ ，则由 $g(x)$ 生成的循环码的零空间，即 \mathbf{H} 矩阵的秩是 $r_1 + r_2 + \cdots + r_i$ ，它等于 $g(x)$ 的次数，也就是校验元的个数。

例 5.6 设 α 是 $\text{GF}(2^{11})$ 的生成元， $\beta = \alpha^{89}$ ，求以 $\beta, \beta^2, \beta^3, \beta^4$ 为根的二进制循环码。

α 的级 $e_1 = 2^{11} - 1 = 2047 = 89 \times 23$ ， $\beta = \alpha^{89}$ ， $(\beta^{23}) = (\alpha^{89})^{23} = 1$ ，故 β 是 23 级元素。若码以 β 为根，则也必以它的共轭根系： $\beta, \beta^2, \beta^4, \beta^8, \beta^{16}, \beta^{32} = \beta^9, \beta^{18}, \beta^{36} = \beta^{13}, \beta^{26} = \beta^3, \beta^6, \beta^{12}$ 为根。而 $\beta^2, \beta^3, \beta^4$ 也包括在这组共轭根系内，所以它们的最小多项式相同。

$$\begin{aligned} g(x) &= \beta_m(x) \\ &= (x - \beta)(x - \beta^2)(x - \beta^3)(x - \beta^4)(x - \beta^6)(x - \beta^8) \\ &\quad \cdot (x - \beta^9)(x - \beta^{12})(x - \beta^{13})(x - \beta^{16})(x - \beta^{18}) \\ &= x^{11} + x^9 + x^7 + x^6 + x^5 + x + 1 \end{aligned}$$

或

$$= x^{11} + x^{10} + x^6 + x^5 + x^4 + x^2 + 1$$

得到一个[23, 12]码。这就是有名的二进制戈莱(Golay)码，能纠正 3 个随机错误。由于

$$\binom{23}{0} + \binom{23}{1} + \binom{23}{2} + \binom{23}{3} = 2048 = 2^{11} = 2^{n-k}$$

所以, [23, 12]码是一个完备码, 而且是唯一已知的 GF(2) 上的纠多个错误的完备码。 ■

*§ 5.3 幂等多项式和最小循环码

一、幂等多项式

由前面讨论可知, 一旦循环码的生成多项式 $g(x)$ 确定, 则码的校验多项式和码的参数 n, k 均确定。但是, 要找 $g(x)$, 必须对 $x^n - 1$ 因式分解, 当 n 较大时, 这就变得很困难。为了避免这种情况, 可在循环码中找到另一重要多项式——**幂等多项式**, 它不需要对 $x^n - 1$ 因式分解就能得到, 并且能唯一地确定循环码。

定义 5.3.1 设 R_n^q 代表 GF(q) 上以 $x^n - 1$ 为模所构成的剩余类环。 $E(x) \in R_n^q$, 若

$$E(x) = E^2(x) = E(x^2)$$

则称 $E(x)$ 为幂等多项式。

例如 $E(x) = x + x^2 + x^4 \in R_7^2$, 就是一个幂等多项式, 因为

$$(x + x^2 + x^4)^2 = x^2 + x^4 + x = x + x^2 + x^4$$

显然, 1 是任何 R_n^q 环上的幂等多项式。

通常在 R_n^2 中

$$E(x) = \sum_{i=0}^{n-1} E_i x^i$$

是一个幂等多项式的充要条件是 $E_i = E_{2i}$ (下标按模 n 取值)。因此, $E(x)$ 的非零系数的次数在同一分圆陪集中。

若 $E(x) \in R_n^2$ 是一个幂等多项式, 则 $1 + E(x) = (1 + E(x))^2 = 1^2 + E^2(x) = 1 + E(x)$ 也是一个幂等多项式。

定理 5.3.1

(1) 一个由 $g(x)$ 生成的循环码或理想 $C_{g(x)}$, 含有一个唯一的幂等多项式 $E(x)$, 且 $C_{g(x)} = C_{E(x)} = C$, $E(x) = p(x)g(x)$ 。当且仅当 $g(\alpha^i) = 0$ 时, $E(\alpha^i) = 0$ 。

(2) 当且仅当 $C(x)E(x) = C(x)$ 时, $C(x) \in C_{g(x)}$ 。

证明

(1) 令 $x^n - 1 = g(x)h(x)$, $(g(x), h(x)) = 1$ 。由欧几里德算法可知, 必存在有 $p(x), q(x)$ 使下式成立:

$$(g(x), h(x)) = 1 = p(x)g(x) + q(x)h(x) \quad (5.3.1)$$

令 $E(x) = p(x)g(x)$, 则

$$\begin{aligned} p(x)g(x) [p(x)g(x) + q(x)h(x)] \\ &= (p(x)g(x))^2 + p(x)g(x)q(x)h(x) \\ &= E^2(x) + p(x)q(x)(x^n - 1) \\ E^2(x) &\equiv E(x) \pmod{x^n - 1} \end{aligned}$$

所以, $E(x) = p(x)g(x) \in C$, 它是码 C 中的一个幂等多项式。

由 $x^n - 1 = g(x)h(x)$ 可知, 一个 n 级元素或者是 $g(x)$ 的根或者是 $h(x)$ 的根。由式(5.3.1)可知, $p(x)$ 与 $h(x)$ 互素。所以, 若有一个 n 级元素是 $p(x)$ 的根, 则它也必是 $g(x)$ 的根, 因而由 § 5.2 可知, $E(x)$ 与 $g(x)$ 生成同一循环码, $C_{E(x)} = C_{g(x)} = C$ 。

(2) 若 $C(x) = C'(x)E(x) = C'(x)p(x)g(x)$, 则 $C(x) \in C$ 。反之, 若 $C(x) \in C$, 则 $C(x)$ 必是 $g(x)$ 或 $E(x)$ 的倍式, 因而 $C(x) = C'(x)E(x)$, $C(x)E(x) = C'(x)E^2(x) = C'(x)E(x) = C(x)$ 。

现证明幂等多项式 $E(x)$ 的唯一性。若还有另一个幂等多项式 $F(x) \in C$, 则由(2)可知, $F(x)E(x) = E(x) = F(x)$ 。 ■

推论 5.3.1 若 $E(x)$ 是码 C 的幂等多项式, 则 C 的生成多项式 $g(x) = \text{GCD}(E(x), x^n - 1)$ 。

例 5.7 GF(2) 上的[7, 4]码。 $g(x) = x^3 + x + 1$, $h(x) = (x + 1)(x^3 + x^2 + 1)$, 且由欧几里德算法

$$xg(x) + h(x) = 1$$

所以, [7, 4]码的幂等多项式 $E(x) = xg(x) = x^4 + x^2 + x$ 。 ■

可以证明幂等多项式有如下性质:

(1) $E(\alpha^i) = 1$, 或 $0 \quad i=0, 1, \dots, n-1$;

(2) 若 $E(x)$ 是一个幂等多项式, 那么它的互反多项式 $E^*(x)$ 也是一个幂等多项式。

码的幂等多项式 $E(x)$ 与码的生成多项式 $g(x)$ 之间有如下关系:

(1) 码 C 信息位的个数, 等于使 $g(\alpha^i) \neq 0$ 的 α^i 的个数, 也等于使 $E(\alpha^i) = 1$ 的 α^i 个数;

(2) $g(x) = \text{GCD}(E(x), x^n - 1)$;

(3) 若码 C 的幂等多项式为 $E(x)$, 则码 C 对偶码 C^\perp 的幂等多项式为 $(1 + E(x))^*$ 。

二、最小循环码

定义 5.3.2 一个理想中若不再含有任何非零的理想, 则称此理想为**最小(素)理想**, 相应的循环码称为**最小循环码或既约循环码**, 而理想的幂等多项式称为**本原幂等多项式**。

可以证明, 每一个幂等多项式必是本原幂等多项式之和, 而多项式环 R_n^q 中的每一个循环码, 必是最小理想之和。

定理 5.3.2 若 $x^n - 1 = p_0(x)p_1(x)p_2(x)\cdots p_k(x)$, 这里 $p_0(x) = x - 1$, $p_i(x)$ 是 GF(q) 上的既约多项式, 则当且仅当以 $g(x) = P_i(x)$ 或 $P_0(x)$ 作为码 C 的生成多项式时, 码 C 才是一个最小循环码。这里, $P_i(x)$ 是 $x^n - 1$ 中除 $p_i(x)$ 之外的所有因子乘积。

可由定理 5.1.4 直接证明该定理。

例 5.8 [7, 4]码, $g(x) = x^3 + x + 1$, $E(x) = x^4 + x^2 + x$ 。在该码中还有一个更小的理想, 它由 $(1 + x)g(x)$ 产生: $(1 + x)g(x)$, $x(1 + x)g(x)$, $x^2(1 + x)g(x)$, $(x^2 + 1)g(x)$, $(x^2 + x + 1)(1 + x)g(x)$, $(x + 1)^3g(x)$, $x(x + 1)^2g(x)$, 0。这 8 个多项式就是[7, 4]码中最小重量为 4 的码字, 它们组成一个理想, 该理想就生成一个[7, 3, 4]循环码, 它的生成多项式 $g'(x) = P_1(x) = (1 + x)(x^3 + x + 1)$, 而幂等多项式是 $E'(x) = (1 + x^2)g'(x)$, 这里, $x^7 - 1 = (x + 1)(x^3 + x^2 + 1)(x^3 + x + 1) = (x + 1)p_1(x)p_2(x)$ 。因而称 $E'(x)$ 为**本原幂等多项式**, 称[7, 3, 4]码是一个既约循环码。 ■

幂等多项式与最小循环码在码的构造中, 特别在分析码的重量分布中起着重要作用。

§ 5.4 缩短循环码与准循环码

一、缩短循环码

循环码生成多项式必是 $x^n - 1$ 的因子，但在绝大部分 n, k 中 $x^n - 1$ 的因子相对而言较少，限制了码的个数。为了增加 n, k 取值的数目增加码的个数，循环码经常用缩短形式，得到**缩短循环码**。

缩短循环码是取 $[n, k]$ 循环码中前 i 位信息位为 0 的码字作为码字，构成一个 $(k-i)$ 维线性子空间，得到一个 $[n-i, k-i]$ ($1 \leq i \leq k$) 缩短循环码。缩短循环码的校验位仍为 $n-k$ ，故该码的纠错能力至少不低于原来的 $[n, k]$ 码，并且它的实现也并不比循环码复杂。

缩短循环码的生成多项式与原码相同，均为 $g(x)$ ，故编码电路与原码同。缩短循环码的 G 矩阵可以从原码的标准 G 阵中去掉前 i 行和前 i 列得到， H 矩阵可以从原典型 H 阵中去掉前 i 列得到。

如由 $[7, 4]$ 循环汉明码缩短一位，便得到了 $[6, 3]$ 缩短码，它的 G 和 H 矩阵可直接由 $[7, 4]$ 码得到：

$$\begin{aligned} G_{[7, 4]} &= \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} \Rightarrow G_{[6, 3]} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} \\ H_{[7, 4]} &= \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \Rightarrow H_{[6, 3]} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

由上述 $G_{[6, 3]}$ 矩阵可以明显看到，缩短循环码的码字之间，不一定存在有循环关系。但是这并不会影响译码器的复杂性，以后将看到缩短循环码的译码器仅在原码译码器基础上，稍加修正即成。

二、准循环码与双环循环码

循环码 C_x 的每一码字，左移或右移循环一位仍是 C_x 的一个码字。也就是说，若 $C(x) \in C_x$ ，则 $xC(x) \in C_x (\bmod x^n - 1)$ ，即码在循环移位下具有**不变性**。但是，对某些码而言，并不具有这种性质，例如，上面例举的 $[7, 4]$ 循环汉明码变成 $[6, 3]$ 缩短码， $[6, 3]$ 码的每一码字循环移位一次不一定是该码的另一码字。对某些码而言，虽然码字循环移位一次得到的不是该码的码字，但若循环移位 $n_0 (>1)$ 次，得到的仍是该码的一个码字。如某一 $[6, 3]$ 码，它的 G 矩阵为

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix} = [I_3 p] \quad (5.4.1)$$

可以检验，若 $C(x) \in C_{[6, 3]}$ ，则 $xC(x) \notin C_{[6, 3]}$ ，但是 $x^2C(x) \in C_{[6, 3]} (\bmod x^6 - 1)$ ，也即循环移位两次所得到的仍是该码的一个码字。

定义 5.4.1 一个 $[mn_0, mk_0]$ 线性分组码，若它的任一码字左移或右移循环移位 n_0 次后，得到的仍是该码的一个码字，则称这类码为**准循环码**。

由此可知，准循环码每个码字的码元位置号在

$$i \rightarrow i + n_0 \pmod{mn_0} \quad i = 1, 2, \dots, n \quad (5.4.2)$$

置换下具有不变性。显然，循环码的 $n_0=1$ ，它是准循环码的一个特例。

由定理 5.1.1 知，循环码是模 $x^n - 1$ 多项式剩余类环中的一个理想，由于准循环码不具有像循环码那样的循环移位一次下的不变性，因此很显然准循环码是模其它多项式 $F(x)$ 剩余类环中的一个理想，当 $F(x) = x^n - 1$ 时，准循环码就是循环码。可以证明^[1] 缩短循环码就是准循环码。

下列矩阵

$$\mathbf{G} = \begin{bmatrix} \mathbf{I}\mathbf{p}_0 & \mathbf{0}\mathbf{p}_1 & \cdots & \mathbf{0}\mathbf{p}_{m-1} \\ \mathbf{0}\mathbf{p}_{m-1} & \mathbf{I}\mathbf{p}_0 & \cdots & \mathbf{0}\mathbf{p}_{m-2} \\ \vdots & \vdots & & \vdots \\ \mathbf{0}\mathbf{p}_1 & \mathbf{0}\mathbf{p}_2 & \cdots & \mathbf{I}\mathbf{p}_0 \end{bmatrix} \quad (5.4.3)$$

在式(5.4.2)的置换下具有不变性，所以由此矩阵 \mathbf{G} 生成的码是一个 $[mn_0, mk_0]$ 准循环码。式中 \mathbf{I} 是 $k_0 \times k_0$ 阶单位方阵， $\mathbf{0}$ 是 $k_0 \times k_0$ 阶 0 阵， \mathbf{p}_i 是 $k_0 \times (n_0 - k_0)$ 矩阵。所以，准循环码的每一码字由 m 段组成，每段前面为 k_0 个信息元，后 $n_0 - k_0$ 个为校验元。与式(5.4.3)对应的准循环码的校验矩阵

$$\mathbf{H} = \begin{bmatrix} \mathbf{p}_0^T \mathbf{I} & \mathbf{p}_{m-1}^T \mathbf{0} & \cdots & \mathbf{p}_1^T \mathbf{0} \\ \mathbf{p}_1^T \mathbf{0} & \mathbf{p}_0^T \mathbf{I} & \cdots & \mathbf{p}_2^T \mathbf{0} \\ \vdots & \vdots & & \vdots \\ \mathbf{p}_{m-1}^T \mathbf{0} & \cdots & \cdots & \mathbf{p}_0^T \mathbf{I} \end{bmatrix} \quad (5.4.4)$$

式中 \mathbf{I} 和 $\mathbf{0}$ 分别表示 $(n_0 - k_0) \times (n_0 - k_0)$ 阶单位方阵和全为 0 矩阵。

如果在式(5.4.3)和(5.4.4)中的 \mathbf{I} 和 $\mathbf{0}$ 矩阵分别用 $k_0 \times k_0$ 阶和 $(n_0 - k_0) \times (n_0 - k_0)$ 阶任意方阵代替，则由此 \mathbf{G} 和 \mathbf{H} 矩阵产生的码称为**广义准循环码**。

我们可把式(5.4.1) \mathbf{G} 矩阵通过列交换变成准循环码系统码形式的 \mathbf{G} 矩阵和 \mathbf{H} 矩阵：

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} = [\mathbf{I}_3 \mathbf{p}] \quad (5.4.5)$$

$$\mathbf{H} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} = [\mathbf{p}^T \mathbf{I}_3] \quad (5.4.6)$$

由该 \mathbf{G} 和 \mathbf{H} 矩阵看到，系统准循环码的生成矩阵和校验矩阵分别由两个循环矩阵 \mathbf{I}_3 和 \mathbf{p} 阵组成， \mathbf{I}_3 和 \mathbf{p} 阵的每行之间均具有循环性。

定义 5.4.2 由两个循环矩阵 \mathbf{I}_k (单位方阵)和 \mathbf{p} 阵组成的 $\mathbf{G} = [\mathbf{I}_k \mathbf{p}]$ 生成的码称为**双环循环码**。

显然，双环循环码等价于 $n_0 = 2$ 的准循环码。很多作者对这类双环循环码进行了研究，发现了一类性能很好的码。表5-1列出了具有最大最小距离的二进制 $[2k, k]$ 双环循环

表 5-1 某些二进制 $[2k, k]$ 双环循环码

| 码长(n) | p 阵生成元 | 最小距离(α) | 码长(n) | p 阵生成元 | 最小距离(α) |
|-----------|----------|------------------|-----------|----------|------------------|
| 6 | 3 | 3 | 26 | 653 | 7 |
| 8 | 7 | 4 | 28 | 727 | 8 |
| 10 | 7 | 4 | 30 | 2 167 | 8 |
| 12 | 7 | 4 | 32 | 557 | 8 |
| 14 | 7 | 4 | 34 | 557 | 8 |
| 16 | 27 | 5 | 36 | 573 | 8 |
| 18 | 117 | 6 | 38 | 557 | 8 |
| 20 | 57 | 6 | 40 | 5 723 | 9 |
| 22 | 267 | 7 | 42 | 14 573 | 10 |
| 24 | 573 | 8 | | | |

码。表中, p 阵生成元栏下的数字, 表示码的生成矩阵 $G = [I_k p]$ 中, p 矩阵第一行的八进制数表示。例如, $[8, 4]$ 码的生成元为 7, 则 p 矩阵的第一行就是 1110, 因而码的 G 矩阵为

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

若化成式(5.4.3)准循环码形式则

$$G = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$

虽然, 这类 $[2k, k]$ 双环循环码等价于 $n_0 = 2$ 的准循环码, 但每个码字之间不一定存在准循环码码字之间的循环关系。

三、双环循环码和准循环码的多项式表示

考虑一个 $[2k, k]$ 双环循环码或系统准循环码。 p 表示 G 矩阵中的循环矩阵, 若 i 表示信息元, 则码字

$$C = mG = m[I_k p] = [m \ mp]$$

令 $m(x)$ 表示信息元的信息多项式, $p(x)$ 表示 p 循环矩阵最上一行的多项式表示, 则 p 矩阵的第二、三、…行的多项式表示分别为:

$$\begin{aligned} &xp(x) \pmod{x^k - 1} \\ &x^2p(x) \pmod{x^k - 1} \\ &\vdots \\ &x^{k-1}p(x) \pmod{x^k - 1} \end{aligned}$$

很容易证明 $k \times k$ 阶循环码矩阵与模 $x^k - 1$ 的多项式代数完全同构。因此, 信息矢量 m 与循环矩阵 p 相乘 mp 的多项式表示为 $m(x)p(x) \pmod{x^k - 1}$ 。所以码字 v 的多项式表示为: $(m(x), m(x)p(x) \pmod{x^k - 1})$ 。由此可知, 系统准循环码或双环循环码的码字, 可以

看成是一个信息多项式 $m(x)$ ，后面跟随一个长为 k 的用 $p(x)$ 产生的、模 $x^k - 1$ 代数中的一个循环码。

例如，(6, 3) 双环循环码中的 $p(x) = x^2 + x$ ，若 $m(x) = x^2$ ，则

$$m(x)p(x) = x^2(x^2 + x) = x^4 + x^3 \equiv x + 1 \pmod{x^3 - 1}$$

因而得到码字的多项式表示是 $(x^2, x + 1)$ ，相应地码字为(001110)。

*§ 5.5 平方剩余码

平方剩余码也称 QR 码，是一类重要的循环码。在较短的码长和 $R \approx 1/2$ 的情况下，它比一般的循环码有更大的最小距离，且译码比较简单，在实际中用得较普遍。

一、平方剩余的基本概念

定义 5.5.1 若对某一素数 p ，存在有一个整数 x ，使下式成立

$$x^2 \equiv i \pmod{p}$$

则称 i 是模 p 的**平方剩余**；否则称为模 p 的**非平方剩余**。

例如，1 是模所有素数的平方剩余，2 是模 7 的平方剩余，但 3 是模 7 的非平方剩余。

要找模 p 的平方剩余，仅只要讨论 1 至 $(p-1)/2$ 的平方即可。事实上

$$(p-a)^2 = p^2 - 2ap + a^2 \equiv a^2 \pmod{p}$$

这说明满足 $(p-a)^2$ 的模 p 平方剩余与满足 a^2 的模 p 平方剩余是相同的。因此，在 1 至 $p-1$ 的整数中，仅只要讨论满足 $1^2, 2^2, \dots, ((p-1)/2)^2$ 的平方剩余即可。

例如， $p=7$ ，则只要考虑 $1^2 \equiv 1, 2^2 \equiv 4, 3^2 \equiv 2, \pmod{7}$ 即可，而 $4^2 = (7-3)^2 \equiv 3^2 \equiv 2, 5^2 = (7-2)^2 \equiv 2^2 \equiv 4, 6^2 = (7-1)^2 \equiv 1^2 \equiv 1 \pmod{7}$ 。

定理 5.5.1 若 p 是奇素数，则在 1 至 $p-1$ 的所有整数中，有 $(p-1)/2$ 个是模 p 的平方剩余，有 $(p-1)/2$ 个为模 p 的非平方剩余。

证明 这只要证明满足 $1^2, 2^2, \dots, ((p-1)/2)^2$ 的模 p 平方剩余的每一个均不相同即可。用反证法。若对一个 $i \neq j, i \geq 1, i, j \leq (p-1)/2$ 有相同的平方剩余：

$$i^2 \equiv j^2 \pmod{p}$$

$$i^2 - j^2 = (i+j)(i-j) \equiv 0 \pmod{p}$$

由于 p 是素数，由第二章可知模 p 的剩余类是一个域，域中无零因子，所以，由上式可得

$$i+j \equiv 0 \quad \text{或} \quad i-j \not\equiv 0 \pmod{p}$$

由于 $i \geq 1, j \leq (p-1)/2$ ，故 $i+j \neq 0$ ，所以只有

$$i-j \equiv 0 \pmod{p}$$

或 $i \equiv j \pmod{p}$ ，又因为 $i, j \leq (p-1)/2$ ，所以这意味着 $i=j$ ，与假设 $i \neq j$ 相矛盾。因此，满足 $1^2, 2^2, \dots, ((p-1)/2)^2$ 的模 p 平方剩余中的每一个均不同，这说明在 $1, 2, \dots, p$ 中只有 $(p-1)/2$ 个不同的平方剩余，其余的 $(p-1)/2$ 个为非平方剩余。 ■

例如， $p=7$ ，则满足 $1^2, 2^2, 3^2$ 的平方剩余为： $1^2 \equiv 1, 2^2 \equiv 4, 3^2 \equiv 2$ ，它们每一个均不同，而其余的三个整数 3, 5, 6 为非平方剩余。这说明可以按照模 p 的平方剩余与非平方剩余对整数或模 p 的剩余类进行分类。

模 p 的平方剩余还有以下性质（请读者证明之）：

(1) 两个平方剩余的积，或两个非平方剩余的积仍是平方剩余。一个平方剩余与另一个非平方剩余之积为非平方剩余。

(2) 若 $p=8m\pm 1$ ，则 2 是模 p 的平方剩余。若 $p=4m+1$ ，则 $-1 \equiv (p-1) \pmod{p}$ 是模 p 的平方剩余；若 $p=4m-1$ ，则 -1 是模 p 的非平方剩余。

二、平方剩余码

下面我们讨论 $\text{GF}(q)$ 上的码长为素数 p 的平方剩余(QR)码。这里， q 是模 p 的平方剩余，且为素数，对二进制 QR 码来说， $q=2$ ，因而由性质(2)可知，码长为 $p=8m\pm 1$ 的形式。

令 Q 表示模 p 的平方剩余集合， N 表示模 p 的非平方剩余集合。若 α 是 $\text{GF}(q^m)$ 域的一个本原域元素，则由性质(1)知：当 i 是偶数时， $\alpha^i \in Q$ ；而 i 为奇数时， $\alpha^i \in N$ ，且 $\alpha^0 = 1 \in Q$ ，因而 Q 是由 α^2 生成的一个循环子群。

由定理4.4.1可知， $\text{GF}(q^m)$ 域中的所有非零元素都是 $x^{q^m}-1=0$ 方程的根。因此若在 $\text{GF}(q^m)$ 中的某一子域内有一个 p 级根 α ，则 $\alpha^p=1$ ， $\alpha, \alpha^2, \alpha^3, \dots, \alpha^{p-1}$ 都是 $x^p-1=0$ 方程的根。所以

$$x^p - 1 = (x - 1)(x - \alpha)(x - \alpha^2) \cdots (x - \alpha^{p-1}) = (x - 1)q(x)n(x)$$

式中

$$\begin{aligned} q(x) &= \prod_{i \in Q} (x - \alpha^i) \\ n(x) &= \prod_{j \in N} (x - \alpha^j) \end{aligned} \quad (5.5.1)$$

是系数在 $\text{GF}(q)$ 上的多项式。

定义 5.5.2 用下列多项式

$$q(x), (x - 1)q(x), n(x), (x - 1)n(x)$$

生成的循环码，称为 $\text{GF}(q)$ 上的平方剩余码，分别用 C_Q, C'_Q, C_N, C'_N 表示。

显然， C_Q 和 C'_N 码分别是 C_Q 和 C_N 码的增余删信码。因而 $C_Q \supseteq C'_Q, C_N \supseteq C'_N$ 。在二进制情况下， C_Q 和 C'_N 均为 C_Q 和 C_N 的偶重量子码。且 C_Q 和 C_N 码的信息位 $k=(p+1)/2$ ，而 C_Q 和 C'_N 码信息位的个数 $k=(p-1)/2$ 。

例 5.9 $q=2, p=8 \cdot 1 - 1 = 7$ 。

$$x^7 - 1 = (x - 1)(x - \alpha)(x - \alpha^2)(x - \alpha^3)(x - \alpha^4)(x - \alpha^5)(x - \alpha^6)$$

$$q(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^4) = x^3 + x + 1$$

$$n(x) = (x - \alpha^3)(x - \alpha^5)(x - \alpha^6) = x^3 + x^2 + 1$$

由 $q(x)$ 与由 $n(x)$ 生成的循环码都是 $[7, 4, 3]$ 循环汉明码。而由 $(x - 1)q(x)$ 和 $(x - 1)n(x)$ 生成的码都是增余删信汉明码。■

例 5.10 例5.6中的 $[23, 12]$ Golay 码也是一个 QR 码。此时 $q=2, p=8 \cdot 3 - 1 = 23$ ， $\beta = \alpha^{89}, \alpha \in \text{GF}(2^{11})$ 是本原域元素。

$$\begin{aligned} x^{23} - 1 &= (x - 1)(x - \beta)(x - \beta^2)(x - \beta^4)(x - \beta^8)(x - \beta^{16})(x - \beta^9)(x - \beta^{18}) \\ &\quad \cdot (x - \beta^{13})(x - \beta^3)(x - \beta^6)(x - \beta^{12})(x - \beta^5)(x - \beta^{10})(x - \beta^{20})(x - \beta^{17}) \\ &\quad \cdot (x - \beta^{11})(x - \beta^{22})(x - \beta^{21})(x - \beta^{19})(x - \beta^{15})(x - \beta^7)(x - \beta^{14}) \\ &= (x - 1)q(x)n(x) \end{aligned}$$

$$q(x) = (x - \beta)(x - \beta^2)\cdots(x - \beta^{12}) = x^{11} + x^9 + x^7 + x^6 + x^5 + x + 1$$

$$n(x) = (x - \beta^5)(x - \beta^{10})\cdots(x - \beta^{14}) = x^{11} + x^{10} + x^6 + x^5 + x^4 + x^2 + 1$$

所以二进制[23, 12]Golay 码可以由 $q(x)$ 或 $n(x)$ 生成。 ■

定理 5.5.2 C_Q 和 C_N 码的最小距离 d 满足下列不等式：

$$d \geq \sqrt{p} \quad (5.5.2)$$

证明 设 $a(x)$ 是 C_Q 中的一个重量最轻且等于 d 的码字, n 是模 p 的一个非平方剩余, 则 $a(x^n)$ 是 C_N 中的一个重量最轻且为 d 的码字。那么 $a(x)a(x^n)$ 既在 C_Q 又在 C_N 中, 因为:

$$\begin{aligned} a(x) &= q(x)f_1(x) = f_1(x) \prod_{i \in Q} (x - \alpha^i) \\ a(x^n) &= n(x)f_2(x) = f_2(x) \prod_{j \in N} (x - \alpha^j) \\ a(x)a(x^n) &= f_1(x)f_2(x) \prod_{k=1}^{p-1} (x - \alpha^k) \end{aligned}$$

又由于:

$$\begin{aligned} x^p - 1 &= (x - 1) \prod_{k=1}^{p-1} (x - \alpha^k) = (x - 1)(x^{p-1} + x^{p-2} + \cdots + x + 1) \\ &= (x - 1) \sum_{j=0}^{p-1} x^j \\ a(x)a(x^n) &= f_1(x)f_2(x) \sum_{j=0}^{p-1} x^j \end{aligned}$$

可知, 非零 $a(x)a(x^n)$ 有最小重量为 p 。因为 $a(x)$ 有最小重量为 d , 则在 $a(x)a(x^n)$ 中的最大非0项数为 d^2 , 所以 $d^2 \geq p$, $d \geq \sqrt{p}$ 。 ■

对 C_Q 和 C_N 码再加一个全校验位后, 变成扩展 QR 码 \hat{C}_Q 和 \hat{C}_N , 最小距离比 C_Q 和 C_N 码增加 1, 码长也增加 1。例如, [7, 4, 3] 码可扩展成 [8, 4, 4] 码, [23, 12, 7] 码可扩展成 [24, 12, 8] 码等。定理 5.5.2 所确定的 QR 码的最小距离仅只是下限, 事实上, 绝大部分 QR 码的最小距离都超过 \sqrt{p} 。

若 QR 码 C_{QR} 是一个自对偶码, 且最小距离是 4 的倍数, 则可以证明 C_{QR} 码的最小距离

$$d \leq 4[n/20] + 4 \quad (5.5.3)$$

如二进制的 [8, 4, 4], [24, 12, 8], [32, 16, 8], [48, 24, 12], [80, 40, 16], [100, 50, 20] 等码均满足式(5.5.3)给出的限。

某些 QR 码与准循环码有密切关系。例如 [17, 9] QR 码, $g(x) = x^8 + x^7 + x^6 + x + 1$, $d=5$, 删去一个信息位后, 就得到一个 [16, 8] 准循环码。可以证明二进制 C_Q QR 码删去一个信息位, 以及扩展 QR 码 \hat{C}_Q 都等效于一个 $n_0=2$ 的准循环码。

QR 剩余码在 $x \rightarrow x'$ (r 为平方剩余) 置换下, 可以证明是不变的, 因而可以应用比较简单的置换译码方法进行译码, 这将在以后谈到。

表 5-2 给出了某些平方剩余码的生成多项式和最小距离。表中生成多项式栏下的数字, 代表生成多项式各项的次数。例如, (3, 1, 0) 代表 $g(x) = x^3 + x + 1$ 。

表 5-2 某些二进制 QR 码

| QR 码 | 生成多项式各项次数 | d |
|----------|--|-----|
| [7, 4] | (3, 1, 0) | 3 |
| [17, 9] | (8, 7, 6, 4, 2, 1, 0) | 5 |
| [23, 12] | (11, 9, 7, 6, 5, 1, 0) | 7 |
| [31, 16] | (15, 12, 7, 6, 2, 1, 0) | 7 |
| [41, 21] | (20, 19, 17, 16, 14, 11, 10, 9, 6, 4, 3, 1, 0) | 9 |
| [47, 24] | (23, 19, 18, 14, 13, 12, 10, 9, 7, 6, 5, 3, 2, 1, 0) | 11 |
| [73, 46] | (9, 7, 4, 3, 0) (9, 4, 2, 1, 0) (9, 8, 0) | 9 |

§ 5.6 多项式及域元素运算电路

从前面讨论可以看到，在循环码的编码中需要用到 $GF(q^n)$ 上的多项式相加、相乘和相除。本节将讨论如何用数字电路实现这些运算，并介绍循环码的编码电路。下面的讨论主要是针对 $GF(2^n)$ 域上的运算，当然也可推广到 $GF(q^n)$ 上。

一、多项式相加电路

已知 $GF(q)$ 上的两多项式

$$A(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \dots + a_1x + a_0 \quad a_i \in GF(q)$$

$$B(x) = b_{n-1}x^{n-1} + b_{n-2}x^{n-2} + \dots + b_1x + b_0 \quad b_i \in GF(q)$$

由第四章所定义的多项式相加

$$C(x) = A(x) + B(x) = (a_{n-1} + b_{n-1})x^{n-1} + \dots + (a_1 + b_1)x + (a_0 + b_0)$$

$$= c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \dots + c_1x + c_0$$

$$c_i \in GF(q), c_i = a_i + b_i, i = 0, 1, 2, \dots, n - 1$$

要完成上述多项式相加，可用图 5-1 所示的电路完成。

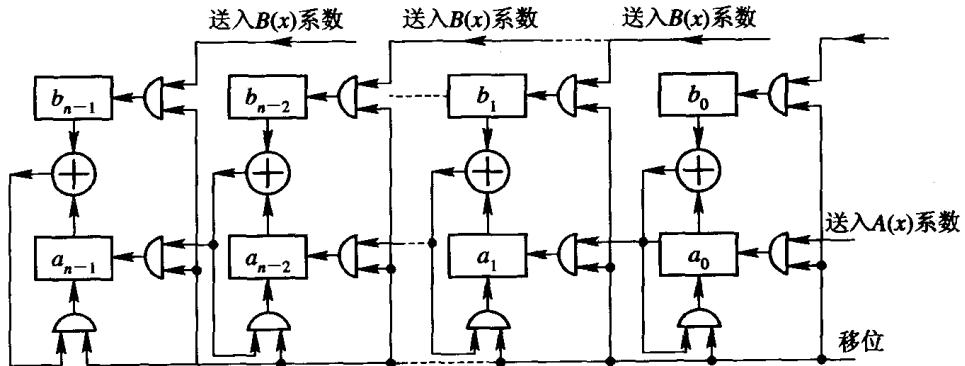


图 5-1 多项式 $A(x) + B(x)$ 并行运算电路

图 5-1 及以后各图中符号的意义如下：

- : 代表一个寄存 GF(q) 上元素的单元，可用触发器、磁芯或其它存贮单元组成；
- : 代表模 q 相加器，无进位运算；
- : 代表两个输入端与门；
- : 代表乘 a 的模 q 常数乘法器，无进位运算；
- : 代表两个输入端的或门。

两个多项式相加之结果 $C(x)$ 的系数存贮在 $A(x)$ 寄存器中。图 5-2 为串行相加电路，相加结果送入 $A(x)$ 系数寄存器中。

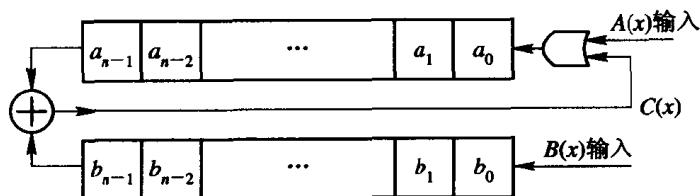


图 5-2 多项式 $A(x) + B(x)$ 串行运算电路

若为 $A(x) - B(x)$ ，则只要把 $B(x)$ 系数以 $GF(q)$ 中的加法逆元代替得到 $-B(x)$ ，再作 $-B(x) + A(x)$ 运算即可。在二进制情况下“-”与“+”运算相同，因此相加电路与相减电路一样。

二、多项式相乘电路

设两多项式为：

$$\begin{aligned} A(x) &= a_k x^k + a_{k-1} x^{k-1} + \cdots + a_1 x + a_0 & a_i \in GF(q) \\ B(x) &= b_r x^r + b_{r-1} x^{r-1} + \cdots + b_1 x + b_0 & b_i \in GF(q) \end{aligned}$$

已知两多项式相乘为

$$\begin{aligned} C(x) &= A(x)B(x) \\ &= a_k b_r x^{k+r} + (a_k b_{r-1} + a_{k-1} b_r) x^{k+r-1} \\ &\quad + (a_k b_{r-2} + a_{k-1} b_{r-1} + a_{k-2} b_r) x^{k+r-2} + \cdots \\ &\quad + (a_k b_{r-i} + a_{k-1} b_{r-(i-1)} + \cdots + a_{k-i} b_r) x^{k+r-i} + \cdots \\ &\quad + (a_1 b_0 + a_0 b_1) x + a_0 b_0 \end{aligned}$$

可用图 5-3 所示的电路完成上述运算过程。该电路由 r 个存贮单元组成的 r 级移位寄存器，至多 r 个模 q 相加器和至多 $(r+1)$ 个模 q 常乘器所组成。

工作开始时， r 级移位存贮器中的存数全清洗为 0，且规定被乘多项式 $A(x)$ 的高次项系数 a_k 首先送入电路，电路的工作过程如下：

(1) 当 $A(x)$ 的最高次系数 a_k 首先送入时，乘积 $C(x)$ 的最高次项 x^{k+r} 的系数 $a_k b_r$ 就出现在输出端，同时 a_k 存入移存器的第一级(最左一级)。

(2) $A(x)$ 的第二个系数 a_{k-1} 送入电路时， a_k 由第一级输出送入第二级，同时与 b_{r-1} 相乘和 $a_{k-1} b_r$ 相加后送到输出端，这就是 $C(x)$ 的 x^{k+r-1} 项系数 $a_k b_{r-1} + a_{k-1} b_r$ 。此时，移存器内的存贮数据为 a_{k-1} , a_k , 0, 0, ..., 0(自左至右)。

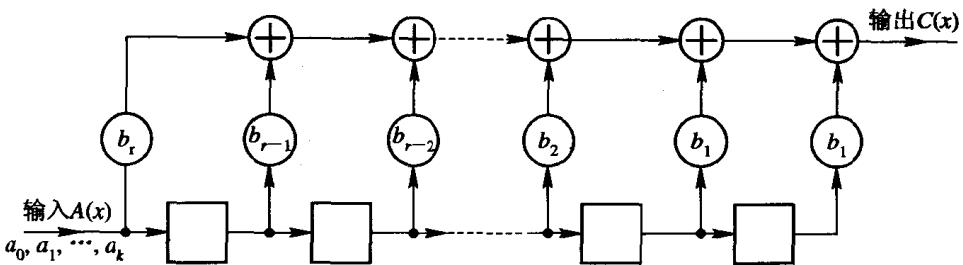


图 5-3 乘 $B(x)$ 运算电路

(3) 上述过程重复进行, 直至 k 次移位后, $A(x)$ 的系数全部送入移存器。 $k+r+1$ 次移位后, 移存器输出 $C(x)$ 的常数项 a_0+b_0 , 移存器中的内容全部恢复到全为 0 初态, 乘法完成。

由上面乘法过程可以看出, 这种乘法电路完成一次乘法运算, 共需移位 $\partial \cdot A(x) + \partial \cdot B(x) + 1$ 次。

除了像图 5-3 所示的乘法电路以外, 还有如图 5-4 所示的一种, 它所用的部件与前一种相同。开始时移存器的内容全清为 0, 第一次移位后, $A(x)$ 的最高次项的系数 a_k 与 b_0 相乘, 乘后所得的结果 $a_k b_0$ 存入移存器第一级(自左至右)中, $a_k b_1$ 存入第二级中, ..., $a_k b_{r-1}$ 存入最后一级中。同时, $a_k b_r$ 输出, 它就是积 $C(x)$ 的最高次 x^{k+r} 的系数。此时, 自左至右移存器中的存数是 $a_k b_0, a_k b_1, \dots, a_k b_{r-2}, a_k b_{r-1}$ 。第二次移位后, $A(x)$ 的 x^{k-1} 项系数 a_{k-1} 送入, 此时电路输出 $a_{k-1}b_r + a_k b_{r-1}$, 这就是积 $C(x)$ 的 x^{k+r-1} 项系数, 移存器的内容变为: $a_{k-1}b_0, a_{k-1}b_1 + a_k b_0, \dots, a_{k-1}b_{r-1} + a_k b_{r-2}$ 。依次类推, 经过 $k+r-1$ 次移位后, 就完成了 $A(x)$ 与 $B(x)$ 相乘的运算。

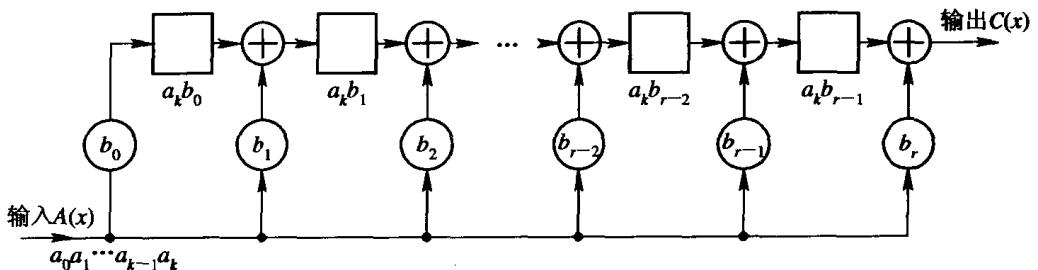


图 5-4 另一种乘 $B(x)$ 电路

从该电路的工作过程看出, 每输入 $A(x)$ 的一个系数 a_i , 相当于 $a_i B(x)$, 而右移一次相当于乘以 x 。因此开始时移存器存数全为 0, 输入 a_k 后便为 $a_k B(x)$ 。第二次移位后输入 a_{k-1} , 得到 $a_k x B(x) + a_{k-1} B(x)$ 。依次类推, 经过 $k+r+1$ 次移位后, 就得到了所要求的积。

$$C(x) = a_k x^k B(x) + a_{k-1} x^{k-1} B(x) + \dots + a_0 B(x) = A(x)B(x)$$

例 5.11 $A(x) = x^3 + x^2 + 1$, $B(x) = x^2 + 1$, 它们都是 GF(2) 上的多项式, 求 $C(x) = A(x)B(x)$ 之电路。

$A(x)$ 乘 $B(x)$ 的电路如图 5-5 所示。因为进行模 2 运算, 故常乘器或者乘 0 或者乘 1。若乘 0, 常乘器是一开路线; 若乘 1, 则是一闭合线。该电路的运算工作过程如表 5-3 所

示。由表看到，6 次移位后，移存器输出了 $\partial \circ A(x) = (x^3+x^2+1)(x^2+1) = x^5+x^4+x^3+1$ 的全部系数：111001，移存器内容全部恢复到全为 0 初态。 ■

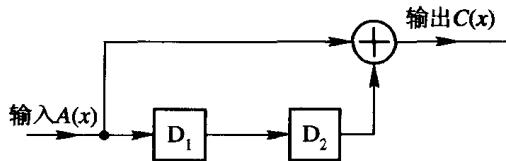


图 5-5 乘 $B(x) = x^2 + 1$ 的电路

表 5-3 $x^3 + x^2 + 1$ 乘 $x^2 + 1$ 过程表

| 节拍 | 输入 | 移存器内容 D ₁ D ₂ | 输出 C(x) 的系数 | C(x) 的次数 |
|----|----|--|-------------|----------|
| 0 | 0 | 0 0 | 0 | |
| 1 | 1 | 1 0 | 1 | x^5 |
| 2 | 1 | 1 1 | 1 | x^4 |
| 3 | 0 | 0 1 | 1 | x^3 |
| 4 | 1 | 1 0 | 0 | x^2 |
| 5 | | 0 1 | 0 | x^1 |
| 6 | | 0 0 | 1 | x^0 |

三、多项式除法电路

GF(q) 上的两多项式

$$\left. \begin{array}{l} A(x) = a_k x^k + a_{k-1} x^{k-1} + \cdots + a_1 x + a_0 \\ B(x) = b_r x^r + b_{r-1} x^{r-1} + \cdots + b_1 x + b_0 \end{array} \right\} \quad k \geq r$$

由欧几里德除法可知：

$$A(x) = q(x)B(x) + r(x) \quad 0 \leq \partial \circ r(x) < \partial \circ B(x) \quad \text{或} \quad r(x) = 0$$

今后均假设 $k \geq r$ ，否则 $q(x) = 0$, $r(x) = A(x)$ 。多项式 $A(x)$ 被 $B(x)$ 除的电路如图 5-6 所示，它由 r 级移存器、至多 r 个 GF(q) 加法器和至多 $r+1$ 个常乘器组成。

为了理解除法电路的工作过程，下面我们列出 $B(x)$ 除 $A(x)$ 的竖式运算式子：

$$\begin{array}{r} b_r x^r + b_{r-1} x^{r-1} + \cdots + b_1 x + b_0 \quad (\text{除式}) \\ \hline b_r^{-1} a_k x^{k-r} + b_r^{-1} (a_{k-1} - b_r^{-1} b_{r-1} a_k) x^{k-r-1} + \cdots & (\text{商式}) \\ a_k x^k + a_{k-1} x^{k-1} + \cdots + a_1 x + a_0 \quad (\text{被除式}) \\ \hline - (a_k x^k + b_r^{-1} b_{r-1} a_k x^{k-1} + \cdots + b_0 b_r^{-1} a_k x^{k-r}) \\ \hline (a_{k-1} - b_r^{-1} b_{r-1} a_k) x^{k-1} + \cdots + (a_{k-r} - b_0 b_r^{-1} a_k) x^{k-r} + \cdots (A) \\ \hline - ((a_{k-1} - b_r^{-1} b_{r-1} a_k) x^{k-1} + \cdots) \\ \hline \cdots \quad \cdots \quad (\text{余式}) \end{array}$$

由上面式子，我们讨论图 5-6 除法电路的工作过程。

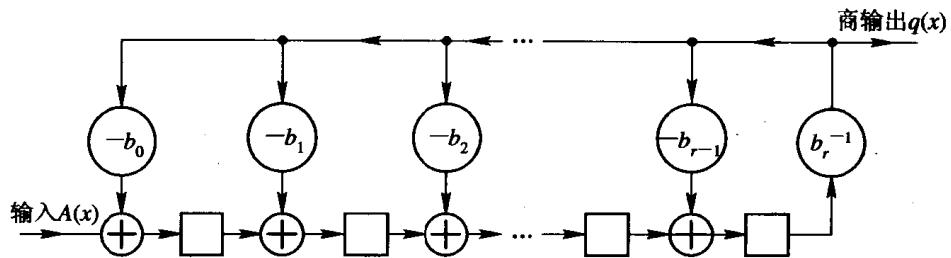


图 5-6 除 $B(x) = b_r x^r + \dots + b_1 x + b_0$ 电路

(1) 开始运算时 r 级移存器中的存数全部清为 0。第一个移位节拍后, 被除多项式 $A(x)$ 的最高次项 x^k 的系数 a_k 首先进入电路的最左一级。 r 次移位后 a_k 进入到移存器的最右一级中, 此时自左至右移存器中的内容为 $a_{k-r+1}, a_{k-r+2}, \dots, a_{k-1}, a_k$ 。

(2) 第 $r+1$ 次移位后, a_k 输出与 b_r^{-1} 相乘得到 $a_k b_r^{-1}$, 这就是商式 $q(x)$ 的第一项 x^{k-r} 的系数。 $a_k b_r^{-1}$ 同时反馈到后面各级寄存器中(所以称这种除法电路为线性反馈移存器)减去 $a_k b_r^{-1} B(x)$, 所以, 此时移存器中自左至右的内容为 $(a_{k-r} - b_0 b_r^{-1} a_k), (a_{k-r+1} - b_1 b_r^{-1} a_k), \dots, (a_{k-1} - b_{r-1} b_r^{-1} a_k)$, 这相应于竖式运算中的第 A 项所示的结果。

(3) 依次类推, 经 $k-1$ 次移位后, 完成了整个除法运算过程。它的输出为商式 $q(x)$, 而移存器中的内容就是余式 $r(x)$ 的系数。

例 5.12 设除式 $B(x) = x^3 + x + 1$, 被除式 $A(x) = x^4 + x^3 + 1$ 都是 GF(2) 上的多项式, 求 $B(x)$ 除 $A(x)$ 的电路。

除 $B(x)$ 的除法电路如图 5-7 所示, 它由 3 级移存器和 2 个模 2 相加器组成。因为在 GF(2) 中, 1 的逆元仍为 1, 相加和相减相同, 所以 b_i^{-1} 与 $-b_i$ 常乘器均为一条闭合线。

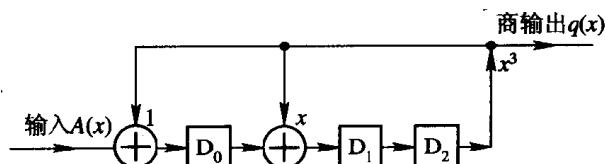


图 5-7 除 $B(x) = x^3 + x + 1$ 电路

完成上述两个多项式相除的长除法运算式如下：

$$\begin{array}{r}
 & & x+1 & (\text{商式}) \\
 & \diagup & \overline{x^4 + x^3} & +1 \\
 x^3 + x + 1 & \diagup & x^4 & + x^2 + x \\
 & & \overline{x^3 + x^2 + x + 1} \\
 & & x^3 & + x + 1 \\
 & & \overline{x^2} & \\
 \end{array}
 \quad (\text{被除式})$$

(除式)

这里

$$x^4 + x^3 + 1 = (x + 1)(x^3 + x + 1) + x^2$$

商为 $x+1$, 余式为 x^2 。

表 5-4 中列出了电路的工作过程。显然, $r+1=4$ 次移位后得到商的第一个系数, $k+1=5$ 次移位后, 就完成了整个除法运算, 并在 D_1 、 D_2 、 D_3 组成的移存器中保留了余式 001, 即 x^2 。

表 5-4 $B(x)$ 除 $x^4 + x^3 + 1$ 的运算过程表

| 节拍 | 输入 | 移存器内容 | | | 输出 |
|----|-------------|------------|----------|------------|----------|
| | | $D_0(x^0)$ | $D_1(x)$ | $D_2(x^2)$ | |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 (x^4) | 1 | 0 | 0 | 0 |
| 2 | 1 (x^3) | 1 | 1 | 0 | 0 |
| 3 | 0 (x^2) | 0 | 1 | 1 | 0 |
| 4 | 0 (x^1) | 1 | 1 | 1 | $1(x^1)$ |
| 5 | 1 (x^0) | 0 | 0 | 1 | $1(x^0)$ |
| | | 余 式 | | | 商 式 |

四、多项式相乘相除电路

$GF(q)$ 上的多项式 $A(x)$ 、 $H(x)$ 、 $G(x)$ 分别为:

$$A(x) = a_k x^k + a_{k-1} x^{k-1} + \cdots + a_1 x + a_0$$

$$H(x) = h_r x^r + h_{r-1} x^{r-1} + \cdots + h_1 x + h_0$$

$$G(x) = g_r x^r + g_{r-1} x^{r-1} + \cdots + g_1 x + g_0$$

若 $A(x)$ 与 $H(x)$ 相乘后再用 $G(x)$ 除, 则

$$A(x)H(x) = q(x)G(x) + r(x) \quad 0 \leqslant \deg r(x) < \deg G(x), \text{ 或 } r(x) = 0$$

该运算可用图 5-8 所示的电路实现, 它由 r 级移存器、至多有 $2(r+1)$ 个 $GF(q)$ 的常乘器和 $r+1$ 个 $GF(q)$ 的相加器组成。显然, 该电路就是图 5-4 与图 5-6 所示的两种电路的结合。如果 $H(x)$ 与 $G(x)$ 次数不等, 则只要按 $G(x)$ 与 $H(x)$ 中最高次数设计移存器级数即可。

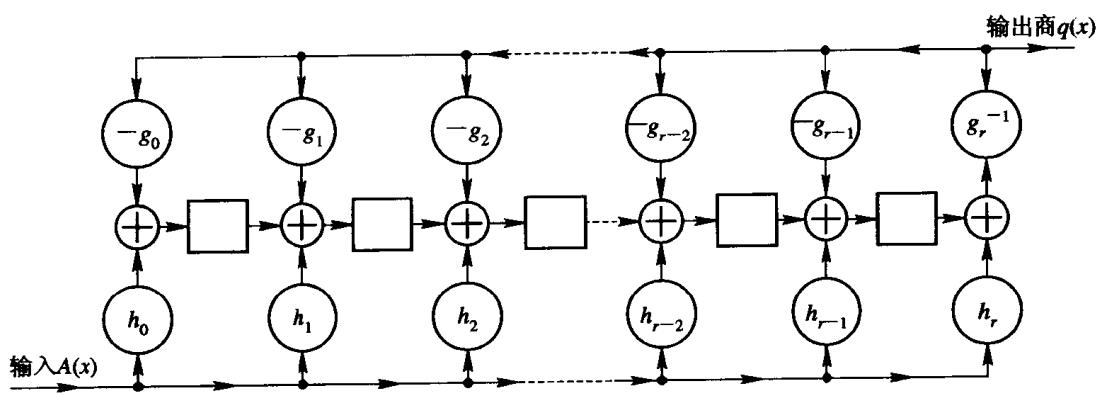


图 5-8 乘 $H(x)$ 除 $G(x)$ 电路

例 5.13 设 GF(2) 上的 3 个多项式为:

$$A(x) = x^4 + x + 1, \quad H(x) = x^2 + 1, \quad G(x) = x^3 + x + 1$$

则

$$\begin{aligned} A(x)H(x) &= (x^4 + x + 1)(x^2 + 1) = x^6 + x^4 + x^3 + x^2 + x + 1 \\ &= x^3(x^3 + x + 1) + (x^2 + x + 1) \\ &= q(x)G(x) + r(x) \end{aligned}$$

可用图 5-9 的电路实现, 该电路的工作过程如表 5-5 所示, 移位 $\partial \cdot A(x) + 1 = 5$ 次后, 即得到了商式 x^3 和余式 $x^2 + x + 1$ 。

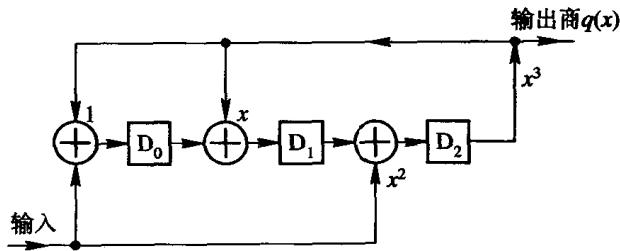


图 5-9 乘 $(x^2 + 1)$ 除 $(x^3 + x + 1)$ 电路

若 GF(2) 上的多项式

$$A(x) = x^4 + x + 1, \quad H(x) = x^3 + x + 1, \quad G(x) = x^2 + 1$$

则

$$\begin{aligned} A(x)H(x) &= (x^4 + x + 1)(x^3 + x + 1) = x^7 + x^5 + x^3 + x^2 + 1 \\ &= (x^5 + x + 1)(x^2 + 1) + x = q(x)G(x) + r(x) \end{aligned}$$

该运算可用图 5-10 所示的电路实现, 它的工作过程如表 5-5 所示。显然, 移位 $\partial \cdot A(x) + \partial \cdot H(x) + 1 - \partial \cdot G(x) = 6$ 次后, 电路即可完成运算, 它的输出是商 $q(x) = (x^5 + x + 1)$, 在移存器中保留了余式 x 。

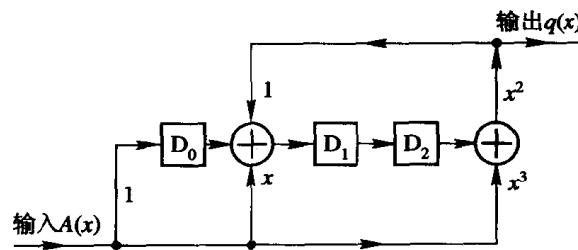


图 5-10 乘 $(x^3 + x + 1)$ 除 $(x^2 + 1)$ 电路

表 5-5 图 5-9 和图 5-10 电路运算过程表

| 节拍 | 图 5-9 电路 | | | | | 图 5-10 电路 | | | | |
|----|--------------------|----------------------------------|----------------------------------|----------------------------------|--------------------|----------------------------------|----------------------------------|----------------------------------|--------------------|--|
| | 输入 | 移存器内容 | | | 输出 | 移存器内容 | | | 输出 | |
| | | D ₀ (x ⁰) | D ₁ (x ¹) | D ₂ (x ²) | | D ₀ (x ⁰) | D ₁ (x ¹) | D ₂ (x ²) | | |
| 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 1 | 1(x ⁴) | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1(x ⁵) | |
| 2 | 0(x ³) | 1 | 0 | 0 | 1(x ³) | 0 | 1 | 0 | 0(x ⁴) | |
| 3 | 0(x ²) | 0 | 1 | 0 | 0(x ²) | 0 | 0 | 1 | 0(x ³) | |
| 4 | 1(x ¹) | 1 | 0 | 0 | 0(x ¹) | 1 | 1 | 0 | 1(x ²) | |
| 5 | 1(x ⁰) | 1 | 1 | 1 | 0(x ⁰) | 1 | 1 | 1 | 1(x ¹) | |
| 6 | | 余式 | | | 商式 | | | 余式 | | |

五、多项式代数和 Galois 域计算电路

1. 计算以 $g(x)$ 为模的多项式剩余类 要确定 $GF(q)$ 上的多项式

$$f(x) = f_k x^k + f_{k-1} x^{k-1} + \dots + f_1 x + f_0$$

在模 $g(x)$ 中属于哪一剩余类，这相当于用 $g(x)$ 除 $f(x)$ 求余式

$$f(x) = q(x)g(x) + r(x) \quad 0 \leqslant \deg r(x) < \deg g(x) \quad \text{或} \quad r(x) = 0$$

所以， $f(x) \in \{r(x)\}$ 或 $f(x) \in \{0\}$ 。

因此，计算 $f(x)$ 在模 $g(x)$ 的哪一剩余类中的电路，就是一个 $g(x)$ 除法电路。

2. 求 $GF(q)$ 扩域 $GF(q^m)$ 上的元素及其相乘电路 由第四章知，如果有一个 $GF(q)$ 上的 m 次本原多项式 $p(x)$ ，则以它的根 α 的所有幂次，就得到了 $GF(q^m)$ 上的所有 $q^m - 1$ 个非 0 元素； $1, \alpha, \alpha^2, \dots, \alpha^{q^m-2}$ ，这 $q^m - 1$ 个元素构成一个循环乘法群。要得到这些元素，可以用 $p(x)$ 除法电路自发运算(无输入运算) $q^m - 1$ 次得到，且电路的初始状态为 α 或其共轭根系元素的 $GF(q)$ 上的 m 重。

例 5.14 求 $GF(2^4)$ 上的所有 15 个非 0 元素。

用 $GF(2)$ 上的四次本原多项式 $p(x) = x^4 + x + 1$ 作除法电路，进行自发运算即可得 15 个非 0 元素，如图 5-11 所示。

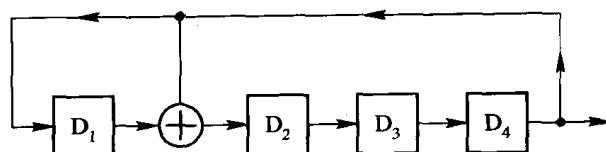


图 5-11 求 $GF(2^4)$ 元素电路

开始计算时电路的初始状态置 $\alpha = (1000)$ ，每移位一次就在四级移存器中得到 $\alpha^2, \alpha^3,$

$\alpha^4, \dots, \alpha^{15}, \alpha, \alpha^2 \dots$ 。相应地，输出序列就是 $\alpha, \alpha^2, \alpha^3, \dots$ ，是一个周期为 15 的二进制序列。该电路的运算过程如表 5-6 所示。

表 5-6 图 5-11 电路运算表

| 节拍 | 移存器内容 | | | | GF(2 ⁴)元素 | 输出 |
|----|----------------|----------------|----------------|----------------|------------------------------|----|
| | D ₁ | D ₂ | D ₃ | D ₄ | | |
| 0 | 1 | 0 | 0 | 0 | $\alpha^0 = 1$ | |
| 1 | 0 | 1 | 0 | 0 | α | 0 |
| 2 | 0 | 0 | 1 | 0 | α^2 | 0 |
| 3 | 0 | 0 | 0 | 1 | α^3 | 0 |
| 4 | 1 | 1 | 0 | 0 | α^4 | 1 |
| 5 | 0 | 1 | 1 | 0 | α^5 | 0 |
| 6 | 0 | 0 | 1 | 1 | α^6 | 0 |
| 7 | 1 | 1 | 0 | 1 | α^7 | 1 |
| 8 | 1 | 0 | 1 | 0 | α^8 | 1 |
| 9 | 0 | 1 | 0 | 1 | α^9 | 0 |
| 10 | 1 | 1 | 1 | 0 | α^{10} | 1 |
| 11 | 0 | 1 | 1 | 1 | α^{11} | 0 |
| 12 | 1 | 1 | 1 | 1 | α^{12} | 1 |
| 13 | 1 | 0 | 1 | 1 | α^{13} | 1 |
| 14 | 1 | 0 | 0 | 1 | α^{14} | 1 |
| 15 | 1 | 0 | 0 | 0 | $\alpha^{15} = \alpha^0 = 1$ | 1 |

由此看出，移位寄存器每右移一次相当于乘 α （左移一次相当于乘 α^{-1} ），15 次移位后电路回到初始状态，它的输出序列显然也以 15 个码元为周期。由于用 4 级线性移存器所能产生的最长序列的长度为 $2^4 - 1 = 15$ ，所以，这种用 GF(2) 上的 m 次本原多项式组成的除法电路（无输入情况），称为**最长序列线性移存器电路**，所产生的序列称为 **m 序列**，它的周期是 $2^m - 1$ 。

在这种电路中，由于每右移一次相当于乘 α ，因此可以应用这种电路进行 GF(q^m) 上域元素之间的相乘和相除运算。如要计算 α^5 乘 α^6 ，则只要在开始时置移存器的初态为 α^5 ，然后往右移位 6 次，即得到了 $\alpha^5 \cdot \alpha^6 = \alpha^{11}$ 的结果。同理，若要计算 $\alpha^5 / \alpha^4 = \alpha^5 \cdot \alpha^{-4}$ ，则只要在移存器中置初态 α^5 ，然后往左移位 4 次，就在移存器中得到了 $\alpha^5 \cdot \alpha^{-4} = \alpha$ 的结果。 ■

3. 计算 $R(x^j)$ 电路 循环码的译码电路中经常要计算 $R(x) = r_{n-1}x^{n-1} + r_{n-2}x^{n-2} + \dots + r_1x + r_0$ 的 $R(\alpha)$ 或 $R(\alpha^j)$ 的值，式中， α 是 $g(x)$ 的根， $g(\alpha) = 0$ 。因为

$$R(x) = q(x)g(x) + r(x) \quad 0 \leqslant \deg r(x) < \deg g(x) \text{ 或 } r(x) = 0$$

所以

$$R(\alpha) = q(\alpha)g(\alpha) + r(\alpha) = r(\alpha)$$

因此，把 $R(x)$ 的系数 ($r_{n-1}, r_{n-2}, \dots, r_1, r_0$) 送入 $g(x)$ 的除法电路中，最后在移存器中得到的余式就是 $R(\alpha)$ 。

计算 $R(\alpha^j)$ 比较麻烦。因为，在 GF(q^m) 上，每个元素 α^j ($j = 0, 1, \dots, q^m - 2$) 用自然基表示时，都可表示成次数小于 m 次的 α 多项式， $\alpha \in GF(q^m)$ 是本原域元素，它是 m 次本

原多项式的根。因此要计算 $R(\alpha^5)$ ，首先必须把 α^5 表示成次数小于 m 的 α 多项式表示。现举例说明。

例 5.15 计算 $GF(2^4)$ 上的 $R(\alpha^5)$ ， α 是 $g(x) = x^4 + x + 1$ 的根， $g(\alpha) = \alpha^4 + \alpha + 1 = 0$ 。由于：

$$1 \cdot \alpha^5 = \alpha^2 + \alpha \quad \alpha \cdot \alpha^5 = \alpha^6 = \alpha^3 + \alpha^2$$

$$\alpha^2 \cdot \alpha^5 = \alpha^7 = \alpha^3 + \alpha + 1 \quad \alpha^3 \cdot \alpha^5 = \alpha^8 = \alpha^2 + 1$$

现在要求计算 $R(\alpha^5)$ ，也就是

$$R(\alpha^5) = q(\alpha^5)g(\alpha^5) + r(\alpha^5)$$

因为要求在 $g(x)$ 除法电路中，每右移一次相当于乘 α^5 ，为此必须先介绍乘 α^5 电路。设原移存器的初始值为 $a_0 + a_1\alpha + a_2\alpha^2 + a_3\alpha^3$ ，乘以 α^5 则变成

$$\begin{aligned} & \alpha^5(a_0 + a_1\alpha + a_2\alpha^2 + a_3\alpha^3) \\ &= a_3\alpha^8 + a_2\alpha^7 + a_1\alpha^6 + a_0\alpha^5 \\ &= a_3(\alpha^2 + 1) + a_2(\alpha^3 + \alpha + 1) + a_1(\alpha^3 + \alpha^2) + a_0(\alpha^2 + \alpha) \\ &= (a_1 + a_2)\alpha^3 + (a_0 + a_1 + a_3)\alpha^2 + (a_0 + a_2)\alpha + (a_2 + a_3) \end{aligned}$$

所以，新的 a_0 是原 $a_2 + a_3$ ，新的 a_1 是原 $a_0 + a_2$ ，新的 a_2 是原 $a_0 + a_1 + a_3$ ，新的 a_3 是原 $a_1 + a_2$ 等。因此，在 $GF(2^4)$ 上乘 α^5 的电路如图 5-12 所示。送入 $R(x)$ ，15 次移位后就在该电路中的存贮器内得到 $R(\alpha^5)$ 。

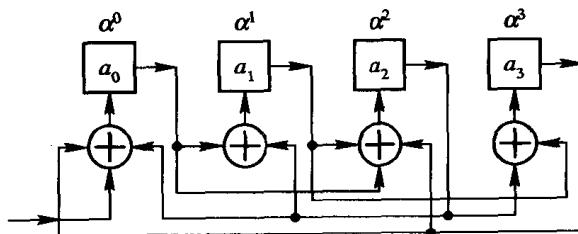


图 5-12 $GF(2^4)$ 域上乘 α^5 电路

§ 5.7 循环码的编码电路

循环码的每个码字必是生成多项式 $g(x)$ 的倍式，由此可知其编码器基本上分为两类： k 级和 $n-k$ 级编码器，下面分别介绍。

一、 $n-k$ 级编码器

$n-k$ 级编码器有两种：一是 $g(x)$ 的乘法电路，另一是 $g(x)$ 的除法电路。前者所编出的码是非系统码，后者是系统码。

1. $g(x)$ 乘法电路编码器 由 § 5.1 可知， $[n, k]$ 循环码的编码就是将 k 位长的信息组 $m(x) = m_{k-1}x^{k-1} + m_{k-2}x^{k-2} + \dots + m_1x + m_0$ ，编成长为 n 的码字 $C(x)$ 。由循环码理论可知， $C(x) = m(x)g(x)$ ， $\partial \cdot g(x) = n - k$ 。因此可用 $g(x)$ 乘法电路实现非系统码编码。

例 5.16 求 $GF(2)$ 上 $[7, 4]$ 循环汉明码编码器。该码的生成多项式 $g(x) = x^3 + x + 1$ ，

由此可得图 5-13 的 $n-k=3$ 级乘法编码器。

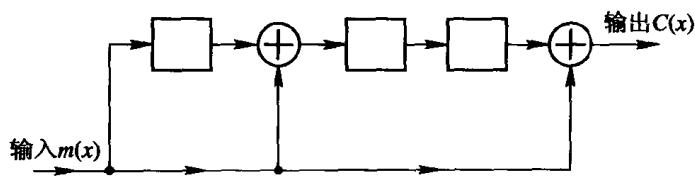


图 5-13 [7, 4] 循环汉明码三级乘法编码器

设送入的信息组为 1001，则编出的码组是 1010011，相应的码多项式

$$C(x) = (x^3 + 1)(x^3 + x + 1) = x^6 + x^4 + x + 1$$

2. $g(x)$ 除法电路编码器 由式(5.1.6)和式(5.1.5)知，要得到系统码必须首先将信息组 $m(x)$ 乘以 x^{n-k} ，变成 $x^{n-k}m(x)$ ，然后再用 $g(x)$ 除，求得相应的余式 $r(x)$ ，把其系数取“-”号就得到了相应的校验位，加上原来的信息组就组成了码字 C 。即

$$m(x)x^{n-k} = q(x)g(x) + r(x) \quad 0 \leqslant \text{deg } r(x) < \text{deg } g(x)$$

$$C(x) = m(x)x^{n-k} - r(x)$$

所以， $[n, k]$ 系统码编码电路就是乘 x^{n-k} 除 $g(x)$ 的电路。

[7, 4] 循环汉明码系统码编码电路如图 5-14 所示。这种电路的工作过程如下：

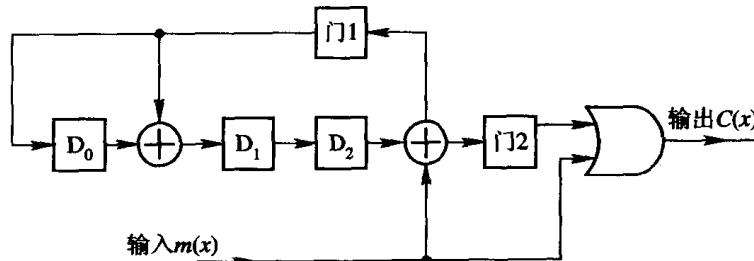


图 5-14 [7, 4] 循环汉明码系统码编码电路

(1) 3 级移存器的初始状态全清为 0，门 1 开、门 2 关，然后进行移位，送入信息组 $m(x)$ 的系数，高次位系数首先进入电路，它一方面经或门输出，一方面自动乘以 $x^{n-k}=x^3$ 次后进入 $g(x)$ 除法电路，从而完成了 $x^{n-k}m(x)=x^3m(x)$ 的作用。

(2) 4 次移位后 $m(x)$ 全部送入电路，完成了除法作用，此时在移存器内保留了余式 $r(x)$ 的系数，在二进制情况下就是校验元。

(3) 此时门 1 关、门 2 开，再经过 3 次移位后，把移存器的校验元全部输出，与原先的 4 位信息元组成了一个长为 7 的码字 $C(x)$ 。

(4) 门 1 开、门 2 关，送入第二组信息组重复上述过程。

表 5-7 列出了图 5-14 的编码工作过程。输入的信息组为 1001，7 次移位后输出端得到已编好的码组 1001110。

表 5-7 图 5-14 电路的编码过程

| 节拍 | 信息组输入 | 移存器内容 | | | 输出码字 |
|----|-------|----------------------------------|----------------------------------|----------------------------------|------|
| | | D ₀ (x ⁰) | D ₁ (x ¹) | D ₂ (x ²) | |
| 0 | | 0 | 0 | 0 | |
| 1 | 1 | 1 | 1 | 0 | 1 |
| 2 | 0 | 0 | 1 | 1 | 0 |
| 3 | 0 | 1 | 1 | 1 | 0 |
| 4 | 1 | 0 | 1 | 1 | 1 |
| 5 | | 0 | 0 | 1 | 1 |
| 6 | | 0 | 0 | 0 | 1 |
| 7 | | 0 | 0 | 0 | 0 |

二、 k 级编码器

[n, k] 循环码系统码的编码器，也可根据校验多项式 $h(x) = h_k x^k + h_{k-1} x^{k-1} + \dots + h_1 x + h_0$ 构造。设系统码的码多项式为

$$C(x) = c_{n-1} x^{n-1} + c_{n-2} x^{n-2} + \dots + c_{n-k} x^{n-k} + c_{n-k-1} x^{n-k-1} + \dots + c_1 x + c_0$$

它的前 k 位系数: $c_{n-1}, c_{n-2}, \dots, c_{n-k}$ 是已知的信息位，而后 $n-k$ 位系数: $c_{n-k-1}, c_{n-k-2}, \dots, c_1, c_0$ 是需求的校验位。码多项式必是生成多项式 $g(x)$ 的倍式，所以

$$C(x) = q(x)g(x) \quad \partial \circ C(x) \leq n-1, \quad \partial \circ g(x) = n-k, \quad \partial \circ q(x) \leq k-1$$

而

$$h(x)C(x) = q(x)g(x)h(x) = q(x)(x^n - 1) = q(x)x^n - q(x)$$

由于 $\partial \circ C(x) \leq n-1, \partial \circ g(x) = n-k, \partial \circ q(x) \leq k-1$ ，所以 $q(x)x^n$ 的最低次数至少为 n 次，因此在 $h(x)C(x)$ 的乘积中， $x^{n-1}, x^{n-2}, \dots, x^k$ 次的系数应为 0，而 x^{n-1} 的系数由

$$c_{n-1-0}h_0 + c_{n-1-1}h_1 + \dots + c_{n-1-k}h_k$$

组成， x^{n-2} 的系数由

$$c_{n-2-0}h_0 + c_{n-2-1}h_1 + \dots + c_{n-2-k}h_k$$

组成。因此

$$\sum_{j=0}^k c_{n-i-j}h_j = 0 \quad i = 1, 2, \dots, n-k \quad (5.7.1)$$

由于 $h(x)$ 为首一多项式， $h_k=1$ ，故上式可写成

$$c_{n-i-k} = - \sum_{j=0}^{k-1} c_{n-i-j}h_j \quad i = 1, 2, \dots, n-k$$

或

$$c_{n-k-i} = - \sum_{j=0}^{k-1} c_{n-j-i}h_j \quad i = 1, 2, \dots, n-k \quad (5.7.2)$$

展开为

$$\begin{aligned} c_{n-k-1} &= - (c_{n-1}h_0 + c_{n-2}h_1 + \dots + c_{n-k}h_{k-1}) \\ c_{n-k-2} &= - (c_{n-2}h_0 + c_{n-3}h_1 + \dots + c_{n-k-1}h_{k-1}) \\ &\vdots \end{aligned} \quad (5.7.3)$$

$$c_{n-k-(n-k)} = c_0 = -(c_k h_0 + c_{k-1} h_1 + \dots + c_1 h_{k-1})$$

这表明码字 C 的第一个校验元 c_{n-k-1} 可由 k 个信息元 $c_{n-1}, c_{n-2}, \dots, c_{n-k}$ 与 $h(x)$ 的系数相乘得到，而由 $c_{n-2}, c_{n-3}, \dots, c_{n-k}, c_{n-k-1}$ 可得到第二个校验元 c_{n-k-2} ，再由 c_{n-3}, \dots, c_{n-k} 信息元和第一、第二校验元 c_{n-k-1}, c_{n-k-2} 可得到第三校验元 c_{n-k-3} 。按这样的线性递推关系，一直可求得所有的 $n-k$ 个校验元 $c_{n-k-1}, c_{n-k-2}, \dots, c_1, c_0$ 。

根据式(5.7.2)的线性递推关系，就不难作出这种形式的循环码系统码的 k 级编码器，如图 5-15 所示。而决定此编码器联接关系的 $h(x)$ 也称为该电路的联接多项式。

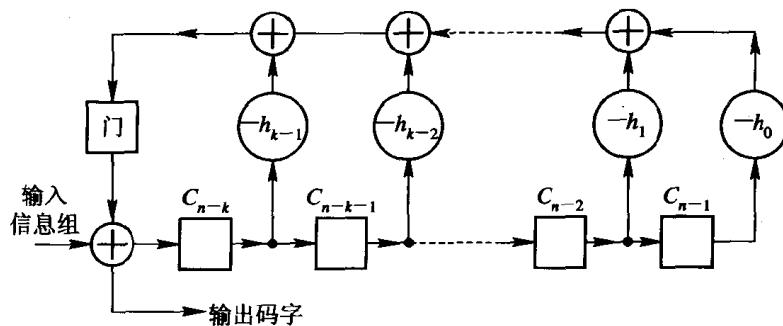


图 5-15 $[n, k]$ 循环码 k 级编码器

例 5.17 二进制 $[7, 4]$ 循环汉明码。 $g(x) = x^3 + x + 1$, $h(x) = x^4 + x^2 + x + 1$ 。它的 $k=4$ 级系统码编码电路如图 5-16 所示。该电路的编码工作过程如下：

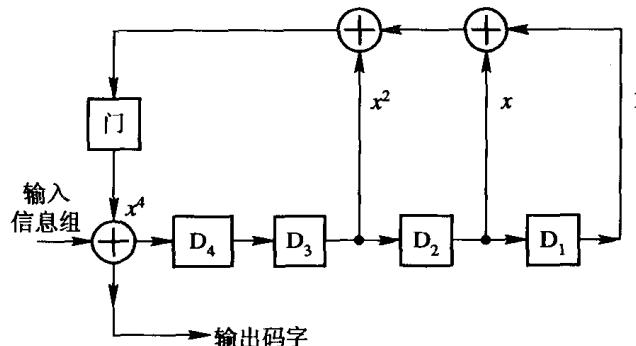


图 5-16 $[7, 4]$ 循环汉明码四级编码器

(1) 开始工作时，4 级移存器全清为 0，门关闭。接着移动 $k=4$ 次， k 个信息元一方面移入 4 级移存器中，另一方面输出。

(2) $k=4$ 次移位后，门打开，此时一组信息元已全部送入移存器中，停止送入信息元。第 $k+1=5$ 次移位后得到第一个校验元 $c_{n-k-1} = c_2$ ，它一方面跟随在第 $c_{n-k} = c_3$ 信息元后输出，另一方面存入移存器的输入级 D_4 中。依此类推，可得到 $c_{n-k-2}, c_{n-k-1}, \dots, c_1, c_0$ 校验元， $k+(n-k)=n$ 次移位后就得到了所有的 $n-k=3$ 个校验元，并与前 $k=4$ 个信息元一起组成了长为 $n=7$ 的码字输出。

表 5-8 图 5-16 编码器工作过程表

| 节拍 | 输入信息组 | 移存器状态 | | | | 输出码字 |
|----|-------|----------------|----------------|----------------|----------------|------|
| | | D ₄ | D ₃ | D ₂ | D ₁ | |
| 0 | | 0 | 0 | 0 | 0 | |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| 2 | 0 | 0 | 1 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 1 | 0 | 0 |
| 4 | 1 | 1 | 0 | 0 | 1 | 1 |
| 5 | 校 | 1 | 0 | 0 | 1 | |
| 6 | 验 | 1 | 1 | 0 | 1 | |
| 7 | 元 | 1 | 1 | 1 | 0 | |

(3) $n=7$ 次移位后, 门重新关闭, 开始接收第二组的 k 个信息元, 重复(1)、(2)过程。表 5-8列出了图 5-16 编码器中各级移存器的状态变化和工作过程。 ■

*§ 5.8 循环码的谱表示与 MS 多项式

如同数字信号处理中, 既可用时域表示信号 $f(t)$, 也可用它的频域 $F(\omega)$ 来表示, 在纠错码理论中, 如果码多项式

$$C(x) = c_{n-1}x^{n-1} + \cdots + c_1x + c_0$$

是码字的时域表示, 则也可用频域来表示码字。

为此我们首先定义有限域 $GF(q^n)$ 上的离散傅里叶变换(DFT)。

定义 5.8.1 设 $GF(q)$ 上的多项式

$$a(x) = a_{n-1}x^{n-1} + \cdots + a_1x + a_0 \quad a_i \in GF(q) \quad (5.8.1)$$

则它在 $GF(q^n)$ 上的谱多项式

$$A(z) = A_{n-1}z^{n-1} + \cdots + A_1z + A_0 = \sum_{j=0}^{n-1} A_j z^j \quad A_i \in GF(q^n) \quad (5.8.2)$$

这里

$$A_j = \sum_{i=1}^{n-1} a_i \alpha^{ji} \quad j = n-1, \dots, 1, 0 \quad (5.8.3)$$

α 是 $GF(q^n)$ 中的 n 级单位本原根, $\alpha^n = 1$ 。称 $A(z)$ 为 $a(x)$ 的谱表示, A_i 为 $A(x)$ 的第 i 个频率分量的值。 $A = (A_{n-1}, \dots, A_1, A_0)$ 是 $a = (a_{n-1}, \dots, a_1, a_0)$ 的 $GF(q^n)$ 上的离散傅里叶变换或谱表示。

由式(5.8.1)和式(5.8.3)可知:

$$a(\alpha^j) = a_{n-1}(\alpha^j)^{n-1} + \cdots + a_1(\alpha^j) + a_0 = \sum_{i=1}^{n-1} a_i (\alpha^j)^i = A_j \quad (5.8.4)$$

因此, $A = (A_{n-1}, \dots, A_1, A_0) = (a(\alpha^{n-1}), \dots, a(\alpha), a(1))$ 。 $a(x)$ 的谱多项式 $A(z)$ 也称为 Mattson-Solomon(MS)多项式。它是1961年由麦迪逊(Mattson)和索罗门(Solomon)在构造非系统 RS 码时提出的。MS 多项式是表征循环码的另一个重要多项式, 它在码的构造和译码中扮演着重要角色。

若用矩阵表示式(5.8.3), 则

$$\begin{bmatrix} A_0 \\ A_1 \\ \vdots \\ A_{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ 1 & \alpha^2 & (\alpha^2)^2 & \cdots & (\alpha^2)^{n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \alpha^{n-1} & (\alpha^{n-1})^2 & \cdots & (\alpha^{n-1})^{n-1} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{bmatrix} \quad (5.8.5)$$

这与数字信号处理中的离散傅里叶变换的形式完全相同。(除了 α 代替频率 ω 以外。)

例如, [7, 3, 4] 码的 $a_1(x) = x(x^4 + x^2 + x + 1)$ 和 $a_2(x) = x^4 + x^2 + x + 1$ 的 MS 多项式或 DFT 分别是:

$$A_1(z) = \alpha^4 z^4 + \alpha^2 z^2 + \alpha z$$

$$A_2(z) = z^4 + z^2 + z$$

图 5-17 给出了 $a_1(x)$ 与 $a_2(x)$ 的谱表示。

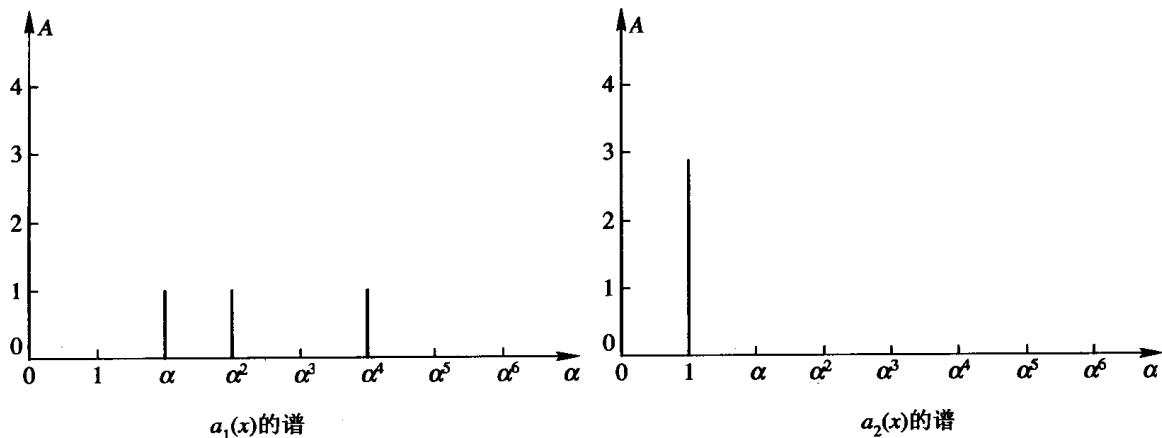


图 5-17 $a_1(x)$ 和 $a_2(x)$ 的谱图

定理 5.8.1(反演公式) 在 $GF(q)$ 上, $a(x)$ 的系数 a_i 与它的谱多项式 $A(z)$ 的系数 A_i 有以下关系:

$$\begin{cases} A_j = \sum_{i=0}^{n-1} a_i \alpha^{ji} & j = 0, 1, \dots, n-1 \\ a_i = \frac{1}{n} \sum_{j=0}^{n-1} A_j \alpha^{-ij} & i = 0, 1, \dots, n-1 \end{cases} \quad (5.8.6)$$

证明 在任何域中均有 $x^n - 1 = (x - 1)(x^{n-1} + x^{n-2} + \dots + x + 1)$ 。由定义 5.8.1 知, α 是 $GF(q^n)$ 中的 n 级元素, 因此, $\alpha^i (i = 0, 1, \dots, n-1)$ 是 $x^n - 1 = 0$ 方程的所有根, 而除了 $\alpha^0 = 1$ 以外, 其余的 α 幂次是 $x^{n-1} + \dots + x + 1 = 0$ 方程的根, 即

$$\sum_{j=0}^{n-1} \alpha^{ij} = 0 \quad i = 1, 2, \dots, n-1, ij \neq 0 \pmod{n}$$

如果 $ij = 0 \pmod{n}$, 则 $\alpha^{ij} = \alpha^0 = 1$ 。代入上式后得

$$\sum_{j=0}^{n-1} \alpha^0 = n$$

当 n 不是域的特征 p 的倍数时, 上式不为0, 所以

$$\sum_{j=0}^{n-1} A_j \alpha^{-ij} = \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} a_i \alpha^{ik} \alpha^{-kj} = \sum_{k=0}^{n-1} a_i \sum_{j=0}^{n-1} \alpha^{(i-k)j} = n a_i$$

因为, $\text{GF}(q^n) = \text{GF}(p^M)$, 而 $\alpha \in \text{GF}(q^n)$, 由推论4.5.2知, 它的级 n 不是域的特征 p 的倍数, 因此, 当 $i=k$ 时, $n \not\equiv 0 \pmod p$ 。 ■

式(5.8.6)称为 A_i 的反离散傅里叶变换(DFT⁻¹)。

把式(5.8.5)写成以下形式:

$$[A^n] = F[\alpha^n]$$

式中, $[A^n] = [A_0 \ A_1 \cdots \ A_{n-1}]^T$, $[\alpha^n] = [\alpha^0 \ \alpha^1 \ \cdots \ \alpha^{n-1}]^T$, 而

$$F = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \alpha^{n-1} & \alpha^{2(n-1)} & \cdots & \alpha^{(n-1)(n-1)} \end{bmatrix} \quad (5.8.7)$$

称为 DFT 矩阵。把 $[A^n]$ 列矩阵进行自上而下的循环移位 i 位, 并由式(5.8.3)不难得到

$$\begin{bmatrix} A_i \\ A_{i+1} \\ \vdots \\ A_{i-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \alpha^{n-1} & \alpha^{2(n-1)} & \cdots & \alpha^{(n-1)(n-1)} \end{bmatrix} \begin{bmatrix} a_0 \\ \alpha a_1 \\ \vdots \\ \alpha^{(n-1)i} a_{n-1} \end{bmatrix} \quad (5.8.8)$$

下面讨论矩阵 $M(A^n)$, 它的行是 $[A^n]$ 的循环移位组成, 即

$$M(A^n) = \begin{bmatrix} A_0 & A_1 & \cdots & A_{n-1} \\ A_1 & A_2 & \cdots & A_0 \\ \vdots & \vdots & & \vdots \\ A_{n-1} & A_0 & \cdots & A_{n-2} \end{bmatrix} \quad (5.8.9)$$

则由式(5.8.8)和式(5.8.9)可得

$$M(A^n) = F \begin{bmatrix} a_0 & a_0 & \cdots & a_0 \\ a_1 & \alpha a_1 & \cdots & \alpha^{n-1} a_1 \\ \vdots & \vdots & & \vdots \\ a_{n-1} & \alpha^{(n-1)} a_{n-1} & \cdots & \alpha^{(n-1)(n-1)} a_{n-1} \end{bmatrix} \quad (5.8.10)$$

该式右边的最后一个矩阵

$$\begin{bmatrix} a_0 & a_0 & \cdots & a_0 \\ a_1 & \alpha a_1 & \cdots & \alpha^{n-1} a_1 \\ \vdots & \vdots & & \vdots \\ a_{n-1} & \alpha^{(n-1)} a_{n-1} & \cdots & \alpha^{(n-1)(n-1)} a_{n-1} \end{bmatrix} = \begin{bmatrix} a_0 & 0 & \cdots & 0 \\ 0 & a_1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & \cdots & \cdots & a_{n-1} \end{bmatrix} F = D_a F \quad (5.8.11)$$

由式(5.8.9)、式(5.8.10)和式(5.8.11)可得

$$M(A^n) = FD_a F$$

类似地, 由式(5.8.2)

$$A(z) = A_{n-1} z^{n-1} + \cdots + A_1 z + A_0$$

可得

$$\begin{aligned}\frac{1}{n}A(\alpha^{-i}) &= \frac{1}{n}(A_{n-1}\alpha^{-i(n-1)} + \cdots + A_1\alpha^{-i} + A_0) \\ &= \frac{1}{n}\sum_{j=0}^{n-1}A_j\alpha^{-ij} = a_i \quad i = 0, 1, \dots, n-1\end{aligned}$$

用矩阵表示上式，并用类似于式(5.8.5)和式(5.8.7)至式(5.8.11)的处理方法，可得

$$M(\mathbf{a}^n) = F^* D_A F^* \quad (5.8.12)$$

式中， F^* 表示 DFT^{-1} 矩阵， D_A 是对角线矩阵， $M(\mathbf{a}^n)$ 的行是 (\mathbf{a}^n) 的循环移位组成。它们分别是：

$$F^* = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha^{-1} & \alpha^{-2} & \cdots & \alpha^{-(n-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \alpha^{-(n-1)} & \alpha^{-2(n-1)} & \cdots & \alpha^{-(n-1)(n-1)} \end{bmatrix} \quad (5.8.13a)$$

$$D_A = \begin{bmatrix} A_0 & 0 & \cdots & 0 \\ 0 & A_1 & & 0 \\ \vdots & \ddots & & \vdots \\ 0 & \cdots & & A_{n-1} \end{bmatrix} \quad (5.8.13b)$$

$$M(\mathbf{a}^n) = \begin{bmatrix} a_0 & a_1 & \cdots & a_{n-1} \\ a_1 & a_2 & \cdots & a_0 \\ \vdots & \vdots & & \vdots \\ a_{n-1} & a_0 & \cdots & a_{n-2} \end{bmatrix} \quad (5.8.13c)$$

由 $M(\mathbf{a}^n) = F^* D_A F^*$ 和 $M(A^n) = F D_A F$ 可知， F 和 F^* 均是范得蒙矩阵。由于 $\alpha^n = 1$ ，所以 $\alpha^i \neq \alpha^j$ ，当 $i \neq j$ 时， F 和 F^* 均是满秩矩阵。因此， $M(A^n)$ 和 $M(\mathbf{a}^n)$ 的秩分别由 D_a 和 D_A 的秩决定，即

$$\begin{aligned}\text{rank}[M(A^n)] &= \text{rank}[D_a] \\ \text{rank}[M(\mathbf{a}^n)] &= \text{rank}[D_A]\end{aligned} \quad (5.8.14)$$

但是，对角线矩阵 D_A 和 D_a 的秩等于对角线分量中非 0 元素的数目，也就是对角线元素的汉明重量。由此得到以下一个非常有用的定理。

定理 5.8.2 序列 $(\mathbf{a}^n) = (a_0, a_1, \dots, a_{n-1})$ 与它的 DFT 序列 $(A^n) = (A_0, A_1, \dots, A_{n-1})$ 以及它们的 DFT^{-1} 有以下关系：

$$\begin{aligned}\text{rank}[M(A^n)] &= \text{rank}[D_a] = w((\mathbf{a}^n)) \\ \text{rank}[M(\mathbf{a}^n)] &= \text{rank}[D_A] = w((A^n))\end{aligned} \quad (5.8.15)$$

可以证明，MS 多项式还有如下性质：

(1) 若 \mathbf{a} 是一个二进制矢量，则 $A(z)$ 是 $\text{GF}(q^n)[z]/z^n - 1$ 多项式环上的幂等多项式，即

$$[A(z)^2]_n = A(z) \quad (5.8.16)$$

这里， $[A(z)^2]_n$ 表示括号中的多项式 $A(z)$ 模 $z^n - 1$ 运算。

(2) 设 $a(x), b(x), c(x) \in \text{GF}(q^n)[x]/x^n - 1$ ，若 $c(x) = a(x) + b(x)$ ，则 $C(z) = A(z) + B(z)$ 。

(3) $c(x) = [a(x)b(x)]_n$ 成立的充要条件是

$$C(z) = A(z) * B(z) = \sum_{i=0}^{n-1} A(\alpha^i)B(\alpha^i)z^i \quad (5.8.17)$$

式中, 符号 * 表示两个多项式之间的卷积运算。

$$(4) c(x) = a(x) * b(x) = \sum_{i=0}^{n-1} a_i b_i x^i \text{ 成立的充要条件是} \\ C(z) = \frac{1}{n} [A(z)B(z)]_n \quad (5.8.18)$$

(5) a 的循环移位($a_{n-2}, a_{n-3}, \dots, a_0, a_{n-1}$)的 MS 多项式是 $A(\alpha z)$ 。

(6) 全 0 矢量和全 1 矢量的 MS 多项式分别是 0 和 1。

(7) 对矢量 a 进行全校验, 则

$$\sum_{i=0}^{n-1} a_i = A(0) = n \quad (5.8.19)$$

$$(8) (A_j)^q = A_j^q \quad (5.8.20)$$

定理 5.8.3

(1) 多项式 $a(x)$ 以 α^j 为根的充要条件是第 j 个频率分量 A_j 等于零。

(2) 谱多项式 $A(z)$ 以 α^{-i} 为根的充要条件是第 i 个时间分量 a_i 等于零。

证明 由式(5.8.4)

$$a(\alpha^j) = \sum_{i=1}^{n-1} a_i (\alpha^j)^i = A_j$$

很快得到(1)。其它由式(5.8.6)可以很快得到第(2)部分的证明。 ■

由图 5-17 可知, $a_1(x)$ 与 $a_2(x)$ 的 $A_1(z)$ 与 $A_2(z)$ 中第 $j=0, 3, 5, 6$ 的频率分量均为 0, 所以, 该[7, 3, 4]码以 $\alpha^0=1, \alpha^3, \alpha^5, \alpha^6$ 为根。

下面讨论如何由 MS 多项式构造 $GF(q)$ 上的循环码。设 $\{a(x)\}$ 是满足以下条件的多项式集合:

(1) 对每一个 i , $a(\alpha^i) \in GF(q)$, α 是 $GF(q)$ 上的 n 级元素, 这意味着 $A(z)$ 的系数在 $GF(q)$ 上。

(2) 对每一个 $a(x) \in \{a(x)\}$, 系数 $a_{i1}, a_{i2}, \dots, a_{ir}$ 全为 0。

由于多项式的所有线性组合也满足上两个条件, 因而 $\{a(x)\}$ 组成一个线性空间。而与 $\{a(x)\}$ 相应的 MS 多项式集合 $\{A(z)\}$ 必然满足以下条件:

(1) $A(z)$ 的系数 $A_i = a(\alpha^i)$ 在 $GF(q)$ 上;

(2) 由定理 5.8.3 可知, $\alpha^{-i_1}, \alpha^{-i_2}, \dots, \alpha^{-i_r}$ 必是 $A(z)$ 的根。

所以, 多项式集合 $\{A(z)\}$ 组成了一个 $GF(q)$ 上的循环码, 它的生成多项式以 $\alpha^{-i_1}, \alpha^{-i_2}, \dots, \alpha^{-i_r}$ 为根。

在上面讨论中, 我们把 $A(z)$ 多项式的系数取值由 $GF(q^n)$ 限制到 $GF(q)$ 上。由于 $A(z)$ 的系数是 $a(\alpha^i)$, 这意味着 $x=1, \alpha, \alpha^2, \dots, \alpha^{n-1}, a(x)$ 的取值 $a(\alpha^i)$ ($i = 0, 1, \dots, n-1$) 必在 $GF(q)$ 上。因而这要求

$$\begin{aligned} [a(x)]^q &= (a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \dots + a_1x + a_0)^q \\ &= a_{n-1}^qx^{q(n-1)} + a_{n-2}^qx^{q(n-2)} + \dots + a_1^qx^q + a_0^q \\ &= a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \dots + a_1x + a_0 \\ &= a(x) \end{aligned} \quad (5.8.21)$$

若令 $a_i^q = a_{iq}$, 则由上式可得到 $a_i = a_i^q = (a_i)^q$, 所以

$$a_{iq} = (a_i)^q \quad (5.8.22)$$

式中, 下标按模 n 取值。式(5.8.22)不仅是使式(5.8.21)成立的必要条件, 而且也是充分条件。由此可得如下定理。

定理 5.8.4 $\text{GF}(q^n)$ 上的多项式 $a(x)$, 当 $x=1, \alpha, \alpha^2, \dots, \alpha^{n-1}, \alpha \in \text{GF}(q^n)$ 是 n 级元素, 且 $(n, q) = 1$, 则 $a(x)$ 取值在 $\text{GF}(q)$ 上的充要条件是对每一个 i , 有 $a_{qi} = (a_i)^q$ 。

例 5.18 $\text{GF}(2^4)$ 上的某一多项式 $a(x)$, $\partial \circ a(x) < 9$, 它的值在 $\text{GF}(2)$ 上。求以此 $a(x)$ 所对应的 MS 多项式所构成的循环码。

令 $\alpha \in \text{GF}(2^4)$ 是一个本原域元素, 它是 $x^4 + x + 1$ 的根。由定理 5.8.4 可知: $a_0 = a_0^2$, $a_1^2 = a_2$, $a_2^2 = a_4$, $a_4^2 = a_8$, $a_8^2 = a_1$, $a_3^2 = a_6$, $a_6^2 = a_{12}$, $a_{12}^2 = a_9$, $a_9^2 = a_3$, $a_5^2 = a_{10}$, $a_{10}^2 = a_5$, $a_7^2 = a_{14}$, $a_{14}^2 = a_{13}$, $a_{13}^2 = a_{11}$, $a_{11}^2 = a_7$ 。由此可知, $a_0^2 = 1$ 或 0 , 而 a_1 可选 $\text{GF}(2^4)$ 中任何一个, 从而 a_2 , a_4 , a_8 也跟着确定。因已假设 $\partial \circ a(x) < 9$, 故 $a_9 = a_3 = a_6 = a_{12} = 0$, 类似地, $a_5 = a_{10} = 0$, $a_{11} = a_{13} = a_{14} = a_7 = 0$, 所以, $a(x)$ 为

$$a(x) = a_1^8 x^8 + a_1^4 x^4 + a_1^2 x^2 + a_1 x + a_0$$

考虑与 $a(x)$ 相应的 MS 多项式 $A(z)$

$$\begin{aligned} A(z) &= A_{15} z^{14} + A_{14} z^{13} + \cdots + A_1 z + A_0 \\ &= a_1^8 [(a^8 z)^{14} + (a^8 z)^{13} + \cdots + a^8 z] + a_1^4 [(a^4 z)^{14} + (a^4 z)^{13} + \cdots + a^4 z] \\ &\quad + a_1^2 [(a^2 z)^{14} + (a^2 z)^{13} + \cdots + a^2 z] + a_1 [(a z)^{14} + (a z)^{13} + \cdots + a z] \\ &\quad + a_0 [z^{14} + z^{13} + \cdots + 1] \end{aligned}$$

因为 $a_3 = a_5 = a_6 = a_7 = a_9 = a_{10} = a_{11} = a_{12} = a_{13} = a_{14} = 0$, 由定理 5.8.3 可知, 相应的 $A(\alpha^{-3}) = A(\alpha^{-5}) = A(\alpha^{-6}) = A(\alpha^{-7}) = A(\alpha^{-9}) = A(\alpha^{-10}) = A(\alpha^{-11}) = A(\alpha^{-12}) = A(\alpha^{-13}) = A(\alpha^{-14}) = 0$, 所以, $A(z)$ 以 $\alpha^{-3}, \alpha^{-5}, \alpha^{-6}, \alpha^{-7}, \alpha^{-9}, \alpha^{-10}, \alpha^{-11}, \alpha^{-12}, \alpha^{-13}, \alpha^{-14}$ 为根, 因而产生的循环码也必以上述元素为根。

$a(x)$ 中 a_0 的可能选择有两种: 0 和 1, a_1 的选择有 16 种, 所以有 32 种可能的 $a(x)$ 多项式, 相应地也有 32 种 $A(z)$, 这 32 个 $A(z)$ 就组成了 2^5 个码字, 因而这是一个 $[15, 5]$ 码。它的生成多项式 $g(x)$ 以 $\alpha^{-3}, \alpha^{-5}, \alpha^{-6}, \alpha^{-7}, \alpha^{-9}, \alpha^{-10}, \alpha^{-11}, \alpha^{-12}, \alpha^{-13}, \alpha^{-14}$ 为根。从这里可以看出, $a(x)$ 相应于信息多项式, 而相应的 $A(z)$ 多项式就是码多项式。相反, 当已知码多项式 $A(z)$ 时, 根据反演公式也可以很快地确定信息多项式 $a(x)$ 。 ■

*§ 5.9 序列线性复杂度与勃拉哈特(Blahut)定理

序列的线性复杂度在循环码的译码和密码分析中扮演着重要角色。而 Blahut 定理则把线性复杂度与码的最小距离紧密联系起来。它们在计算码的最小距离限时, 起着重要作用。

一、序列的线性复杂度

如果把 $[n, k]$ 线性码的每一个码字 $C = (c_{n-1}, c_{n-2}, \dots, c_1, c_0)$ 看成一个 n 长的序列, 则我们可以从序列的角度来研究编码器。

由 k 级编码器的编码可知, 从已知的 k 个信息元 $(c_{n-1}, \dots, c_{n-k})$, 通过线性递推式(5.7.2)可以得到 $n-k$ 个校验位。如果把 $(c_{n-1}, \dots, c_{n-k})$ 作为初值, 放入图 5-15 所示的 k 级编码电路, 则由此线性递推电路产生一个周期为 n 的周期序列

$$(C^n) = (c_{n-1}, c_{n-2}, \dots, c_1, c_0, c_{n-1}, c_{n-2}, \dots)$$

该序列满足由式(5.7.2)决定的线性递推式

$$c_{n-i-k} = - \sum_{j=0}^{k-1} c_{n-i-j} h_j \quad i = 1, 2, \dots, n-k$$

一般而言, 若序列 $(a^n) = (a_{n-1}, a_{n-2}, \dots, a_1, a_0)$ 满足以下线性递推式:

$$a_i = - \sum_{j=1}^l h_j a_{i-j}$$

或

$$a_i + h_1 a_{i-1} + h_2 a_{i-2} + \dots + h_l a_{i-l} = 0 \quad l \leq i \leq n \quad (5.9.1)$$

如图 5-18 所示。它意味着存在有一组不全为 0 非负的整数 $h_1, h_2, \dots, h_l, h_l \neq 0$, 使上式成立。当上式成立时, 则

$$a_i = - \sum_{j=1}^L h_j a_{i-j} \quad L > l, h_{l+1} = \dots = h_L = 0 \quad (5.9.2)$$

也同样成立。因此, 必须定义一个能使式(5.9.1)成立的最小 L , 定义这个最小 $L=l$ 为序列 (a^n) 的 **线性复杂度**。

定义 5.9.1 若 l 是使序列 (a^n) 满足线性递推方程式(5.9.1)的最小 l , 则称序列的线性复杂度 $L(a^n) = l$, 称 $h'(x) = x^l + h_1 x^{l-1} + \dots + h_{l-1} x + h_l$ 为序列 (a^n) 的 **联接多项式**或**特征多项式**, 因此 $L(a^n)$ 也就是特征多项式的次数或线性反馈移存器的**级数**。

序列的线性复杂度, 说明生成该序列最少可以用几级线性反馈移位寄存器生成。例如, 把 [7, 4, 3] 码的每一码字看成一个序列, 则可以由图 5-16 所示的 4 级线性反馈移位寄存器生成, 也就是每一码字(序列)满足:

$$c_{7-3-i} = - \sum_{j=0}^3 c_{7-(j+1)-i} h_j \quad i = 1, 2, 3, 4$$

或

$$c_{4-i} = - \sum_{j=1}^4 c_{7-(j+1)-i} h_{j-1} \quad i = 1, 2, 3, 4$$

式中, $h_0 = h_1 = h_2 = h_4 = 1$, $h_3 = 0$, 特征多项式 $h'(x) = x^4 + x^3 + x^2 + 1$, 它就是 $h(x)$ 的互反多项式。如何寻找一个序列或码字的线性复杂度与循环码的译码密切相关, 这将在 BCH 码译码时谈到。

引理 5.9.1 当且仅当为全 0 序列时, 序列的线性复杂度为 0。

$$L(a^n) = 0 \Leftrightarrow (a^n) = (00\dots 0) = (\mathbf{0}^n)$$

引理 5.9.2 n 长序列的线性复杂度至多为 n , 即 $L(a^n) \leq n$ 。

引理 5.9.3 当且仅当 (a^n) 是除了最后一位为非 0, 其余都是 0 时, (a^n) 的线性复杂度为 n 。

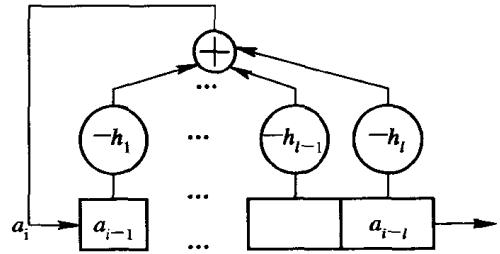


图 5-18 线性反馈移存器

$$L(\mathbf{a}^n) = n \Leftrightarrow (\mathbf{a}^n) = (\mathbf{0}^{n-1} \Delta^+)$$

式中, Δ^+ 表示 $GF(q)$ 上的非0元素。

证明 若 $(\mathbf{a}^n) = (\mathbf{0}^{n-1} \Delta^+)$ 则对于任何 $l < n$, $i = n - 1$ 都不能使式(5.9.1)满足。反之, 如果 $(\mathbf{a}^n) \neq (\mathbf{0}^{n-1} \Delta^+)$, 则在 a_0, a_1, \dots, a_{n-2} 中或者必存在有一个非0的数据, 对于 $l = n - 1$ 时, 使式(5.9.1)满足, 或者 $L(\mathbf{a}^n) = 0$ 。 ■

引理 5.9.4 若 $GF(q)$ 上, 序列 (\mathbf{a}^n) 的线性复杂度是 l , 则当 (\mathbf{a}^n) 看成是扩域 $GF(q^m)$ 上的序列时, 其线性复杂度也是 l 。

引理 5.9.5 如果序列 (\mathbf{a}^n) 由 3 个序列: (Δ^j) , (\mathbf{b}^m) 和 (Δ^{n-j-m}) 串接而成, 即 $(\mathbf{a}^n) = (\Delta^j \mathbf{b}^m \Delta^{n-j-m})$, 则 $L(\mathbf{a}^n) \geq L(\mathbf{b}^m)$, 这里 Δ 是指 $GF(q)$ 中的任意元素。类似地, 对半无限序列 $(\mathbf{a}^\infty) = (a_0, a_1, \dots) = (\Delta^j \mathbf{b}^m \Delta^\infty)$, 也有 $L(\mathbf{a}^\infty) \geq L(\mathbf{b}^m)$ 。

证明 令 $L(\mathbf{a}^n) = l$ 。若 $m \leq l$, 则由引理 5.9.2, $L(\mathbf{b}^m) \leq m \leq l = L(\mathbf{a}^n)$, 引理得证。下面证明 $m > l$ 时的情形。由 (\mathbf{a}^n) 序列的结构形式可知, $a_k = b_{k-j}$, 对所有 k , $j \leq k < m + j$ 。对于所有 i , $l \leq i < n$, 式(5.9.1)对这些 a_i 是成立的, 这意味着式(5.9.1)对所有这些 a_i , $l + j \leq i < m + j$, 也同样成立, 而这等价于对于 $l \leq i < m$, 式(5.9.1)对 (\mathbf{b}^m) 序列同样成立。 ■

定理 5.9.1^[4] 若序列 (\mathbf{b}^{n-M+1}) 是由 n 重 (\mathbf{a}^n) 根据以下规则得到:

$$b_{i-M} = \sum_{j=1}^M \beta_j a_{i-j} \quad i = M, M+1, \dots, n$$

式中, $\beta_1, \beta_2, \dots, \beta_M \in GF(q)$, 且 (\mathbf{a}^n) 满足式(5.9.1), 对于 $l \leq i < n$ 。因而 (\mathbf{b}^{n-M+1}) 与 (\mathbf{a}^n) 满足式(5.9.1), 有相同的系数 h_1, h_2, \dots, h_l , 对于 $l \leq i < n - M + 1$, 特别是

$$L(\mathbf{b}^{n-M+1}) \leq L(\mathbf{a}^n) \tag{5.9.3}$$

该定理说明一个序列如果是另一个序列的移位组合之和, 则得到的线性复杂度不可能超过原序列。

二、矩阵的秩与线性复杂度

下面我们从矩阵的秩来讨论序列的线性复杂度。

定理 5.9.2 令 $(\mathbf{a}^n) = (a_{n-1}, \dots, a_1, a_0)$, 则 (\mathbf{a}^n) 的线性复杂度等于以下矩阵:

$$\mathbf{G}_a = \begin{bmatrix} a_0 & a_1 & a_2 & \cdots & a_{n-3} & a_{n-2} & a_{n-1} \\ a_1 & a_2 & a_3 & \cdots & a_{n-2} & a_{n-1} & \Delta_1 \\ a_2 & a_3 & a_4 & \cdots & a_{n-1} & \Delta_1 & \Delta_2 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ a_{n-1} & \Delta_1 & \Delta_2 & \cdots & \Delta_{n-3} & \Delta_{n-2} & \Delta_{n-1} \end{bmatrix} \tag{5.9.4}$$

的最小的秩。即

$$L(\mathbf{a}^n) = \text{rank}[\mathbf{G}_a] \tag{5.9.5}$$

这里所谓最小是按矩阵中的任意符号 $\Delta_1, \Delta_2, \dots, \Delta_{n-1}$ 取遍所有可能的值的情况。

证明 令 \mathbf{G}_a 矩阵的第一行至第 n 行, 分别标以第 0 至第 $n-1$ 。若 $L(\mathbf{a}^n) = l$, 则由式(5.9.1)可知, \mathbf{G}_a 的第 i 行 ($i = l, l+1, \dots, n-1$) 可以通过合适地选择任意元素 Δ , 用第 l 行以前各行线性组合表示。因此 \mathbf{G}_a 的最小秩不会大于 l 。但是由线性复杂度的定义, l 是使式(5.9.1)满足的最小整数。因此最小的 l 是使第 l 行的前 $n-l$ 个分量, 能被写成前面各行

的对应位分量的线性组合, 因此前面 l 行必是线性无关的, 所以 G_a 的最小秩不小于 l 。 ■

该定理把矩阵的秩与有限长序列的线性复杂度联系了起来, 从而可以通过计算矩阵的秩找线性复杂度。

以 $(\mathbf{a}^\infty) = (a_0, a_1, \dots)$ 表示一个半无限序列, 而用 $(\mathbf{a}^n)^\infty = (a_0, a_1, \dots, a_{n-1}, a_0, a_1, \dots, a_{n-1}, a_0, \dots)$ 表示周期半无限序列。下面讨论这种周期半无限序列的线性复杂度。

定理 5.9.3 一个周期半无限序列 $(\mathbf{a}^n)^\infty$ 的线性复杂度等于以下矩阵:

$$\mathbf{M}(\mathbf{a}^n) = \begin{bmatrix} a_0 & a_1 & \cdots & a_{n-1} \\ a_1 & a_2 & \cdots & a_0 \\ \vdots & \vdots & & \vdots \\ a_{n-1} & a_0 & \cdots & a_{n-2} \end{bmatrix} = \begin{bmatrix} (\mathbf{a}^n) \\ S(\mathbf{a}^n) \\ \vdots \\ S^{n-1}(\mathbf{a}^n) \end{bmatrix} \quad (5.9.6)$$

的秩。即

$$L((\mathbf{a}^n)^\infty) = \text{rank}[\mathbf{M}(\mathbf{a}^n)] \quad (5.9.7)$$

这里 $S^i(\mathbf{a}^n)$ 表示 (\mathbf{a}^n) 序列向左循环移位 i 次得到的序列, 称 $\mathbf{M}(\mathbf{a}^n)$ 阵为循环矩阵。

证明 在周期半无限序列 $(\mathbf{a}^n)^\infty = (a_0, a_1, \dots, a_{n-1}, a_0, a_1, \dots)$ 中, $a_i = a_{i+n}$, 对所有 $i \geq 0$, 由式(5.9.1)知, $(\mathbf{a}^n)^\infty$ 的线性复杂度至多为 n 。如果 $L((\mathbf{a}^n)^\infty) = l$, 则意味着对所有 $l \leq i < \infty$, 式(5.9.1)必须成立。当且仅当对 n 个连续的 i 位, 例如说 $i=l, l+1, \dots, l+n-1$, 式(5.9.1)成立时, 该周期序列的线性复杂度是 l , 为了对 $i \geq l$, $(\mathbf{a}^n)^\infty$ 也满足式(5.9.1), 则 (\mathbf{a}^n) 必须满足以下的矢量方程:

$$S^l(\mathbf{a}^n) + h_1 S^{l-1}(\mathbf{a}^n) + \cdots + h_l(\mathbf{a}^n) = (\mathbf{0}^n) \quad (5.9.8)$$

因此, $L((\mathbf{a}^n)^\infty)$ 是使式(5.9.8)成立的最小 l 。

如果 $L((\mathbf{a}^n)^\infty) = l$, 则 l 是使 $\mathbf{M}(\mathbf{a}^n)$ 的第 $S^i(\mathbf{a}^n)$ 行是前 l 行线性组合, 即使式(5.9.8)成立的最小整数, 因此, $\text{rank}[\mathbf{M}(\mathbf{a}^n)] \geq l$ 。另一方面, 对式(5.9.8)两边乘以 S^j , 得

$$S^{l+j}(\mathbf{a}^n) + h_1 S^{l+j-1}(\mathbf{a}^n) + \cdots + h_l S^j(\mathbf{a}^n) = S^j(\mathbf{0}^n) = (\mathbf{0}^n) \quad 0 \leq j < n-l$$

该式指出, $\mathbf{M}(\mathbf{a}^n)$ 的每一行是与以前各行线性相关的, 因此, $\text{rank}[\mathbf{M}(\mathbf{a}^n)] \leq l$ 。由此得到 $\text{rank}[\mathbf{M}(\mathbf{a}^n)] = l$ 。 ■

定理 5.9.4^[4] 若 N 重 (\mathbf{b}^N) 是 N 重 (\mathbf{a}^N) 的 N 个循环移位的线性组合, 则满足 $(\mathbf{a}^N)^\infty$ 的任何线性递推方程, 也满足 $(\mathbf{b}^N)^\infty$ 。特别是

$$L((\mathbf{b}^N)^\infty) \leq L((\mathbf{a}^N)^\infty)$$

该定理是定理 5.9.1 的推广。该定理也意味着生成 $(\mathbf{a}^N)^\infty$ 的线性移位寄存器也能生成 $(\mathbf{b}^N)^\infty$ 。

例如, $\text{GF}(2^3)$ 上的周期为 7 的序列

$$(\mathbf{a}^7) = (0, 1, \alpha^3, \alpha^5, \alpha^2, \alpha, \alpha^6)$$

设 (\mathbf{b}^7) 序列是 (\mathbf{a}^7) 序列和它的二次循环移位的线性组合, 且组合数字 $\beta_1 = \alpha$, $\beta_2 = 1$, 即

$$\begin{aligned} \beta_1(\mathbf{a}^7) &= \alpha(\mathbf{a}^7) = (0, \alpha, \alpha^4, \alpha^6, \alpha^3, \alpha^2, 1) \\ \beta_2 S^2(\mathbf{a}^7) &= S^2(\mathbf{a}^7) = (\alpha^3, \alpha^5, \alpha^2, \alpha, \alpha^6, 0, 1) \\ \beta_1(\mathbf{a}^7) + \beta_2 S^2(\mathbf{a}^7) &= (\mathbf{b}^7) = (\alpha^3, \alpha^6, \alpha, \alpha^5, \alpha^4, \alpha^2, 0) \end{aligned}$$

这里 $\alpha \in \text{GF}(2^3)$ 是本原元, 可以检查 $(\mathbf{a}^7)^\infty$ 满足

$$a_i = \alpha^3 a_{i-1} + \alpha a_{i-2} \quad i \geq 2$$

$L(\mathbf{a}^7)=2$ 。由引理 5.9.3 和引理 5.9.4 可知 $L((\mathbf{b}^7)^\infty) \geq 2$ ，并且由定理 5.9.4 可知：

$$b_i = \alpha^3 b_{i-1} + \alpha b_{i-2} \quad i \geq 2$$

由此可得 $b_0 = \alpha^3$, $b_1 = \alpha^6$, $b_2 = \alpha^3 \alpha^6 + \alpha \alpha^3 = \alpha$, $b_3 = \alpha^3 \alpha + \alpha \alpha^6 = \alpha^5$, $b_4 = \alpha^3 \alpha^5 + \alpha \alpha = \alpha^4$, $b_5 = \alpha^3 \alpha^4 + \alpha \alpha^3 = \alpha^2$, $b_6 = \alpha^3 \alpha^2 + \alpha \alpha^4 = 0$, $b_7 = \alpha^3 0 + \alpha \alpha^2 = \alpha^3 = b_0 \dots$ 。

下面的定理给出了周期半无限序列的采样(抽样)序列的线性复杂度。

定理 5.9.5^[4](采样定理) 对周期半无限序列 $(\mathbf{a}^N)^\infty$ 的每一 K 位采样组成的采样序列，如果 $(K, N)=1$ ，则采样序列的线性复杂度与原序列相同。若 $(K, N) \neq 1$ ，则采样序列的线性复杂度不会增加。

例如， $(\mathbf{a}^N)^\infty = (\mathbf{a}^7)^\infty = (0, 1, \alpha^3, \alpha^5, \alpha^2, \alpha, \alpha^6, 0, 1, \alpha^3, \dots)$ ，每一 $K=2$ 采样得到采样序列

$$(\mathbf{b}^N)^\infty = (\mathbf{b}^7)^\infty = (0, \alpha^3, \alpha^2, \alpha^6, 1, \alpha^5, \alpha, 0, \alpha^3, \dots)$$

它的线性复杂度 $L((\mathbf{b}^7)^\infty) = L((\mathbf{a}^7)^\infty) = 2$ ，因为 $(2, 7) = 1$ 。

三、Blahut 定理

定理 5.9.6 (Blahut 定理) 若 $\text{GF}(q)$ 域含有一个 n 级单位根，则 n 长有限序列 (\mathbf{a}^n) 的汉明重量 $w(\mathbf{a}^n) = L((A^n)^\infty)$ ，这里 (A^n) 是 (\mathbf{a}^n) 的离散傅里叶变换，或 MS 多项式。反之， n 长的有限序列 (A^n) 的汉明重量 $w((A^n)) = L((\mathbf{a}^n)^\infty)$ ，这里 (\mathbf{a}^n) 是 (A^n) 的离散傅里叶变换的反变换。

证明 由定理 5.8.2 可知：

$$\text{rank}[\mathbf{M}(A^n)] = w((A^n))$$

$$\text{rank}[\mathbf{M}(\mathbf{a}^n)] = w((\mathbf{a}^n))$$

由定理 5.9.3 可以有：

$$L((A^n)^\infty) = \text{rank}[\mathbf{M}(A^n)]$$

$$L((\mathbf{a}^n)^\infty) = \text{rank}[\mathbf{M}(\mathbf{a}^n)]$$

由此可得：

$$L((A^n)^\infty) = w((A^n))$$

$$L((\mathbf{a}^n)^\infty) = w((\mathbf{a}^n))$$

Blahut 定理 说明一个周期序列的线性复杂度，等于它的一个周期的 DFT 的汉明重量。该定理在分析码的最小重量和码的最小距离限时将起重要作用。

有关序列线性复杂度的更多性质和详细证明，及其与循环码的关系参阅 [4]。

例 5.19 $\text{GF}(2)$ 上 $[7, 3, 4]$ 码的一个码多项式 $a(x) = x^5 + x^3 + x^2 + x$ ，相应的码序列 $(\mathbf{a}^7) = (a_0 a_1 \dots a_6) = (0111010)$ ，它的 MS 多项式 $A(z) = \alpha^4 z^4 + \alpha^2 z^2 + \alpha z$ ，所以 $w(A(z)) = 3$ ，可知循环码码字序列的线性复杂度 $L((\mathbf{a}^7)^\infty) = w(A(z)) = 3$ 。反之，由 DFT 性质可知，由 $A(z)$ 系数 $(A_0, A_1, \dots, A_6) = (0, \alpha, \alpha^2, 0, \alpha^4, 0, 0)$ 组成的序列，它的线性复杂度 $L((A^7)^\infty) = w((\mathbf{a}^7)) = 4$ 。

到此为止我们所讨论的循环码都是等保护能力的，也就是码对每个信息元所提供的保护或纠错能力是相等的。很明显，由于循环码的循环性质，对于系统循环码而言，必是等保护能力码，不可能存在不等保护能力的系统循环码。但是，对于非系统循环码而言，就可能存在。已证明许多非系统循环码具有不等保护能力(UEP)，在文献[6]中，利用计算机

已搜索了码长 ≤ 65 奇数长度的所有非系统循环码的不等保护能力。由于篇幅所限，这里不再介绍 UEP 循环码的性质与构造，可参阅有关文献^[5, 6]。

习 题

1. 令 $g(x) = x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1$ 是 $(15, 5)$ 循环码的生成多项式，
 - (1) 求出该码的校验多项式；
 - (2) 写出该码的系统码形式的 G 和 H 矩阵；
 - (3) 构造 k 级编码器。
2. 求 $GF(2^5)$ 上以 α, α^3 为根的二进制循环码：
 - (1) 求出生成多项式 $g(x)$ ，确定码长 n 和信息位个数 k ；
 - (2) 写出该码系统码形式的 G 和 H 矩阵；
 - (3) 求出该码的 R 和最小距离。
3. 令 n 是 $g(x) | (x^n - 1)$ 的最小整数。现用该 $g(x)$ 生成为 n 的循环码，证明码的最小距离至少为 3。
4. 设一个 $[n, k]$ 循环码的生成多项式 $g(x)$ ，且 n 是奇数， $x+1$ 不是 $g(x)$ 的因子，试证全为 1 的 n 重是一码字。
5. 在第 4 题中若 $x+1$ 是 $g(x)$ 的一个因子，证明全为 1 的 n 重不是码字；但若 n 是偶数，则全为 1 的 n 重是一个码字。
6. 求以 $C(x) = x^{12} + x^9 + x^6 + x^5 + x^4 + x^3 + x^2$ 作为码字时，有最小码长 n 的二进制循环码。求出它的 $g(x)$ 、 n 、 k ，并画出编码器。
7. 令 $[n, k]$ 循环码的校验多项式是
$$h(x) = x^k + h_{k-1}x^{k-1} + h_{k-2}x^{k-2} + \cdots + h_1x + 1$$
它的生成矩阵 $G = [I \ p]$ ，证明矩阵 p 的第一列是 $(1, h_1, \dots, h_{k-1})^T$ 。
8. 设 C 是以 $g(x)$ 为生成多项式的二进制 $[n, k]$ 循环码，它具有如下性质：当 $C_1 = (c_0, c_1, \dots, c_{n-1})$ 是 C 的一个码字时， $C'_1 = (\bar{c}_0, \bar{c}_1, \dots, \bar{c}_{n-1}) \in C$ ，这里， $\bar{c}_i = 0$ ，若 $c_i = 1$ ； $\bar{c}_i = 1$ ，若 $c_i = 0$ 。问 $g(x)$ 应满足什么条件？
9. 令 β_1 和 β_2 是 $GF(2^m)$ 伽逻华域中的两个不同的非零元素，且令 $\varphi_1(x)$ 和 $\varphi_2(x)$ 分别是 β_1 和 β_2 的最小多项式，有一个以 $g(x) = \varphi_1(x)\varphi_2(x)$ 作为生成多项式的循环码吗？若你认为有，则找出以 $g(x)$ 作为生成多项式的最短循环码。
10. 令 C_1 和 C_2 分别是由 $g_1(x)$ 和 $g_2(x)$ 生成的两个长为 n 的循环码。证明 C_1 和 C_2 码的和 $C_1 + C_2 = C_3$ 码有生成多项式 $g(x) = \text{GCD}(g_1(x), g_2(x))$ ，这里， $C_1 + C_2 = \{c_1(x) + c_2(x), c_1(x) \in C_1, \text{且 } c_2(x) \in C_2\}$ 。设 C_1 和 C_2 码的最小距离分别是 d_1 和 d_2 ， C_3 码的最小距离是什么？
11. 令 C_1 和 C_2 分别由 $g_1(x)$ 和 $g_2(x)$ 生成的两个长为 n 的循环码。证明既属于 C_1 又属于 C_2 码的公共码多项式形成了另一循环码 $C_3 = C_1 \cap C_2$ ，确定 C_3 码的生成多项式。设 C_1 和 C_2 码的最小距离分别是 d_1 和 d_2 ，你能谈谈关于 C_3 码的最小距离吗？
12. 找码长为 7 的所有循环码的生成多项式和幂等多项式。
13. 若 $x^n - 1 = g(x)h(x)$ ，由 $g(x)$ 生成的码有幂等多项式 $e(x)$ ，证明由 $h(x)$ 生成的码有幂等多项式 $1 + e(x)$ 。

14. 令 C_1 和 C_2 是循环码，它们的幂等多项式分别是 $e_1(x)$ 和 $e_2(x)$ ，证明当且仅当 $e_1(x)e_2(x) = e_1(x)$ 时， $C_1 \subseteq C_2$ 。
15. 已知(15, 5)码的生成多项式 $g(x) = x^{10} + x^8 + x^6 + x^4 + x^2 + x + 1$ ，求出 $x^2g(x)$ 的 MS 多项式，画出它的谱图。
16. 第15题中的 $x^2g(x)$ 码多项式相对应的码序列的线性复杂度是多少？画出生成此序列的线性反馈移存器电路图。
17. 从频谱的观点证明[7, 3, 4]扩张汉明码的最小距离是4。
18. 设 $g(x)$ 以 $\alpha, \alpha^2, \alpha^3, \dots, \alpha^{\delta-1}$ 为根。从谱和序列线性复杂度的观点证明该码的最小距离至少是 δ 。（提示：首先求出码多项式的 MS 多项式，把此 MS 多项式看成是 $GF(q^n)$ 上的 n 长序列，引用定理 5.9.3 和定理 5.9.6，求出此序列的线性复杂度。）

参 考 文 献

- [1] W. W. Peterson and E. J. Weldon, Jr., Error-Correcting Codes, MIT Press, Cambridge, 1972.
- [2] [美]林舒、科斯特洛：《差错控制编码：基础和应用》（王育民、王新梅译），人民邮电出版社，1986。
- [3] V. Pless, Introduction to the Theory of Error-Correcting Codes, John Wiley and Sons, 1982.
- [4] T. Schaub, A Linear Complexity Approach to Cyclic Codes, Dissertation for Ph. D, ADAG, 1988.
- [5] W. J. V. Gils., "Two topics on linear unequal error protection codes bound on their length and cyclic code classes", IEEE Trans. on IT, No. 6, pp. 866—876, 1983.
- [6] M. C. Lin, C. C. Lin, and S. Lin, "Computer search for binary cyclic UEP codes of odd length up to 65", IEEE Trans. on IT, No. 4, pp. 924—935, 1990.
- [7] G. Castagnoli, J. L. Massey et al., "On repeated-root cyclic codes", IEEE Trans. on IT, No. 2, pp. 337—342, 1991.

第六章 循环码的译码

由前一章可知，循环码的编码电路可用 k 级或 $n-k$ 级移位寄存器简单实现。但是，它的译码电路却往往比较复杂。如同所有的线性或非线性分组码一样，译码问题始终是纠错码理论特别是循环码理论与实际中最关键、最有吸引力的问题之一。译码器的运算速度及其实现的复杂性，往往成为纠错码或循环码是否实用的关键。研究出快速、简单、经济和译码错误概率小的译码方法，仍是当今纠错码理论和实际中最重要的研究课题之一。

如果把码字看成是时间域或频率域上的信号，则纠错码的译码可以分为利用 DFT 和信号处理技术的频域译码以及时域译码。时域译码是频域译码的基础，因此本章讨论循环码的时域译码，下一章再讨论有关码的频域译码。而在时域译码中，一般分为两类：一是利用码的代数结构的代数译码；二是不仅利用码的代数结构，而且还利用信道干扰统计特性的概率译码。本章先介绍循环码的一般译码原理，然后介绍几类比较简单的代数译码方法如捕错译码、大数逻辑译码等，最后介绍常用的概率译码方法如广义最小距离译码、契斯(Chase)译码等。

§ 6.1 循环码译码的一般原理

设发送的码字是 $C(x) = (c_{n-1}x^{n-1} + \dots + c_1x + c_0)$ (今后不再严格区分码字与码多项式)，通过 q 进制输入和输出的信道后，译码器输入端得到的是

$$R(x) = C(x) + E(x) = (r_{n-1}x^{n-1} + \dots + r_1x + r_0) \quad r_i = c_i + e_i$$

式中， $E(x) = (e_{n-1}x^{n-1} + \dots + e_1x + e_0)$ 是信道产生的错误图样，应当指出，上述这些式中的 c_i 、 r_i 、 e_i 均是 $GF(q)$ 中的元素，也就是我们这里仅讨论硬判决时的译码方法。

译码器的主要任务就是如何从 $R(x)$ 中得到正确的估计错误图样 $\hat{E}(x) = E(x)$ ，然后得到 $C(x)$ ，并由此得到信息组 $m(x)$ 。

如同所有线性分组码的译码一样，循环码的译码也分为以下 3 步：

- (1) 计算 $R(x)$ 的伴随式 $S(x)$ ；
- (2) 根据伴随式 $S(x)$ 找出估计错误图样 $\hat{E}(x)$ ；

(3) $R(x) - \hat{E}(x) = \hat{C}$ ，得到译码器输出的估值码字 \hat{C} ，并送出译码器给用户。若 $\hat{C}=C$ ，则译码正确，否则译码错误。

如果是非系统码，则还必须由 \hat{C} 中得到估值信息组 \hat{m} ；如果是系统码，这一步可省略。

由于循环码的循环特性，在上述各步运算中，往往比非循环码的计算要简单。

一、伴随式计算和错误的检测

设发送的码字 $C = (c_{n-1}, c_{n-2}, \dots, c_1, c_0)$ ，信道产生的错误图样为 $E = (e_{n-1}, e_{n-2}, \dots, e_1, e_0)$ ，译码器收到的 n 重

$$\begin{aligned}\mathbf{R} = \mathbf{C} + \mathbf{E} &= (c_{n-1} + e_{n-1}, c_{n-2} + e_{n-2}, \dots, c_1 + e_1, c_0 + e_0) \\ &= (r_{n-1}, r_{n-2}, \dots, r_1, r_0) \quad r_i = c_i + e_i\end{aligned}$$

由伴随式定义可知，相应的伴随式是

$$\mathbf{S} = \mathbf{R} \cdot \mathbf{H}^T = (\mathbf{C} + \mathbf{E}) \mathbf{H}^T = \mathbf{E} \mathbf{H}^T$$

可知伴随式 \mathbf{S} 仅与错误图样有关，而与发送的码字无关，由它可计算出错误图样 \mathbf{E} 。

设 $[n, k]$ 循环码的生成多项式为 $g(x)$ ，且 $x^n - 1 = g(x)h(x)$ ， $\deg g(x) = n-k$ 。该码的一致校验矩阵由式 (5.1.9) 可知为

$$\mathbf{H} \equiv [\tilde{x}^{n-1} \tilde{x}^{n-2} \cdots \tilde{x} \tilde{1}^T] \pmod{g(x)}$$

所以

$$\begin{aligned}\mathbf{S} = \mathbf{R} \cdot \mathbf{H}^T &= (r_{n-1}, r_{n-2}, \dots, r_1, r_0) \begin{bmatrix} \tilde{x}_{n-1} \\ \tilde{x}_{n-2} \\ \vdots \\ \tilde{x}_1 \\ \tilde{x}_0 \end{bmatrix} \pmod{g(x)} \\ &= (c_{n-1}, c_{n-2}, \dots, c_1, c_0) \begin{bmatrix} \tilde{x}_{n-1} \\ \vdots \\ \tilde{x}_0 \end{bmatrix} \pmod{g(x)} \\ &\quad + (e_{n-1}, \dots, e_1, e_0) \begin{bmatrix} \tilde{x}_{n-1} \\ \vdots \\ \tilde{x}_0 \end{bmatrix} \pmod{g(x)}\end{aligned}$$

由此式可知相应的多项式表示为

$$S(x) \equiv C(x) + E(x) \equiv R(x) \equiv E(x) \pmod{g(x)} \quad (6.1.1)$$

或

$$S(x) = R_g(x) + g(x)q(x) = E_g(x) + g(x)q_1(x) \quad (6.1.2)$$

式中， $R_g(x)$ 和 $E_g(x)$ 分别是 $R(x)$ 和 $E(x)$ 被 $g(x)$ 除后所得的余式。两式表明循环码的伴随式计算电路就是一个 $g(x)$ 除法电路，伴随式 $S(x)$ 就是 $g(x)$ 除 $R(x)$ 后所得的余式。如果接收的矢量 $R(x)$ 没有错误， $E(x) = 0$ ，则 $S(x) = 0$ ；否则， $S(x) \neq 0$ （在码的检错能力以内）。因此，循环码的检错电路非常简单，就是一个 $g(x)$ 除法电路。收到 $R(x)$ 后送入 $g(x)$ 除法电路运算，若最后得到的余式为 0，则说明 $E(x) = 0$ ，接收到的 $R(x)$ 就是一个码字；若不为 0，则说明接收到的 $R(x)$ 不是码字。

由式(6.1.1)或式(6.1.2)看出，若 $\deg E(x) < \deg g(x) = n-k$ ，或 $E(x) = x^i E_1(x)$ 且 $\deg E_1(x) < \deg g(x)$ ，则 $S(x) \not\equiv 0 \pmod{g(x)}$ 。这说明 $[n, k]$ 循环码至少能检测长度等于 $n-k$ 的突发错误，以及检测使 $S(x) \equiv E(x) \neq 0 \pmod{g(x)}$ 的所有错误图样 $E(x)$ 。

二、伴随式计算电路性质及一般译码器

用 $g(x)$ 除法电路计算伴随式的电路（伴随式计算电路）有如下一个很重要的特点。

定理 6.1.1 若 $S(x)$ 是 $R(x)$ 的伴随式，则 $R(x)$ 的循环移位 $xR(x)$ （在模 $x^n - 1$ 运算

下)的伴随式 $S_1(x)$, 是 $S(x)$ 在伴随式计算电路中无输入时(自发运算)右移一位的结果, 即

$$S_1(x) \equiv xS(x) \pmod{g(x)} \quad (6.1.3)$$

证明 由伴随式定义可知 $xR(x)$ 之伴随式为

$$S_1(x) \equiv xR(x) \pmod{g(x)} = xR_g(x) + q_1(x)g(x) \quad (6.1.4)$$

由式(6.1.2)可知:

$$xS(x) = xR_g(x) + xq(x)g(x)$$

该式减去式(6.1.4)可得:

$$xS(x) - S_1(x) = g(x)(xq(x) - q_1(x)) \equiv 0 \pmod{g(x)}$$

因此

$$S_1(x) \equiv xS(x) \pmod{g(x)} \quad \blacksquare$$

推论 6.1.1 $x^jR(x)$ 的伴随式 $S_j(x) \equiv x^jS(x) \pmod{g(x)}$, $j = 0, 1, \dots, n-1$ 。而任意多项式 $a(x)$ 乘 $R(x)$ 所对应的伴随式

$$S_a(x) \equiv a(x)S(x) \pmod{g(x)} \quad (6.1.5)$$

伴随式计算电路的这些性质, 在循环码的译码运算中非常有用。若 $C(x)$ 是循环码 C 的一个码字, 则 $xC(x) \in C$ 。因此, 若 $S(x) \equiv E(x) \pmod{g(x)}$ 是 $R(x) = C(x) + E(x)$ 的伴随式, 则 $xS(x) \equiv xE(x) \pmod{g(x)}$ 就是 $xR(x) = xC(x) + xE(x)$ 的伴随式。也就是说若 $E(x)$ 是一个可纠正的错误图样(陪集首), 则 $E(x)$ 的循环移位 $xE(x)$ 也是一个可纠正的错误图样, 一般说来 $x^iE(x)$ ($1 \leq i \leq n-1$) 也是可纠正的错误图样。这样我们就可以根据这种循环关系划分错误图样, 把任一特定的错误图样及其所有循环移位作为一类。如可将错误图样 $100\dots0, 0100\dots0, 0010\dots0, \dots, 00\dots01$ 作为一类, 并以第一位开头为非 0 的错误图样 $100\dots0$, 作为此类错误图样的代表。在 q 进制时, 若码要纠正 $\leq t$ 个错误, 则错误图样代表共有

$$N_1 = \sum_{j=1}^t (q-1)^j \binom{n-1}{j-1} \quad (6.1.6)$$

个。译码时, 只要知道此代表图样的伴随式, 该类中其它错误图样的伴随式都可由此代表图样伴随式在伴随式计算电路中得到。这样, 就使得循环码译码器的错误图样识别电路大为简化, 由原来识别

$$N_2 = \sum_{j=1}^t \binom{n}{j} (q-1)^j \quad (6.1.7)$$

个图样减少到 N_1 个。

例如二进制码, $n=63$, $t=4$, 由式(6.1.6)和式(6.1.7)计算译码器所需识别的错误图样个数如表 6-1 所示。

表 6-1 N_1, N_2 比较表

| t | 1 | 2 | 3 | 4 |
|-------|----|-------|--------|---------|
| N_1 | 1 | 63 | 1 954 | 39 774 |
| N_2 | 63 | 2 016 | 41 727 | 637 382 |

一般说来纠错码译码设备的复杂性, 主要决定于由伴随式找出错误图样的识别电路或

组合逻辑电路的复杂性。由表 6-1 可知，虽然循环码识别错误图样的个数比一般非循环码大为减少，但随着 n 、 t 的加大，需要识别的错误代表个数 N_1 仍增加很快，以致难以实现。因此，利用组合逻辑线路识别错误图样代表的方法，仅适合于 n 、 t 较小的情况。若 n 和 t 都较大，则必须利用循环码的其它特点，寻找更为简单和巧妙的译码方法，这正是纠错码理论研究中和它的实际应用中最引人注目的问题之一。

例 6.1 二进制 $[7, 4, 3]$ 循环汉明码，它的 $g(x) = x^3 + x + 1$ ，相应的校验矩阵

$$\mathbf{H} = [\tilde{x}^6 \tilde{x}^5 \tilde{x}^4 \tilde{x}^3 \tilde{x}^2 \tilde{x}^1 \tilde{x}^0] \pmod{g(x)} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

由式(6.1.6)知，构造此译码器的错误图样识别电路时，只要识别一个图样 $E_6 = (1000000)$ 就够了，该图样的伴随式就是 \mathbf{H} 的第一列 (101) 。可知，识别 E_6 错误图样的识别电路就是一个检测伴随式是否是 (101) 的电路。由此可得如图 6-1 所示的译码电路。图中的伴随式计算电路就是一个 $g(x) = x^3 + x + 1$ 的除法电路，而有 3 个输入端的与门和反相器，组成了识别 (101) 的伴随式识别器。

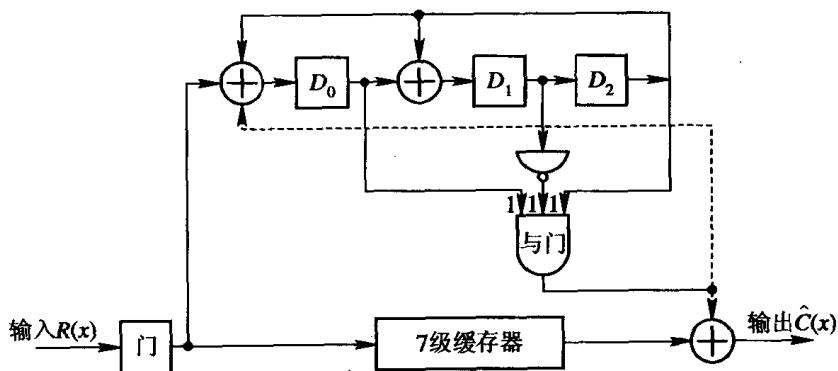


图 6-1 $[7, 4, 3]$ 循环汉明码译码器

译码器的译码过程如下：

(1) 开始译码时门开，移存器内容全为 0。收到的 $R(x) = r_6x^6 + \dots + r_0$ ，以高次项系数(r_6)至低次项系数的次序，一方面送入 7 级缓冲器，一方面送入 $g(x)$ 除法电路计算伴随式。7 次移位后， $R(x)$ 的系数全部存入缓存器， $g(x)$ 电路也得到了伴随式 $S_0(x)$ ，此时门关，禁止输入。

(2) 若 $S_0(x) \equiv 1 + x^2 \equiv x^6 \pmod{g(x)}$ ，说明 $E(x) = x^6$ ， r_6 位有错，伴随式计算 ($g(x)$ 除法器) 电路中的 D_0 、 D_1 、 D_2 存贮的值是 (101) ，它就是 $S_0(x) = 1 + x^2$ 之系数。 D_1 的 0 经反相后成了 1，与门 3 个输入端全为 1，呈打开状态。这时译码器继续移位， r_6 从缓存器输出，与门也输出一个信号“1”与 r_6 相加，使 r_6 由原来的 1 变成 0，或由 0 变成 1，纠正了 r_6 的错误： $r_6 + 1 = c_6 + e_6 + 1 = c_6 + 1 + 1 = c_6$ ，得到了原来发送的码元。此时与门的纠错信号“1”也反馈到伴随式计算电路输入端(图中虚线所示)，对伴随式进行修正，以消去该错误对伴随式的影响。这由于

$$R(x) = r_{n-1}x^{n-1} + \dots + r_1x + r_0$$

相应的伴随式是 $S_0(x)$ 。纠错后 $R(x)$ 成为

$$R_1(x) = (r_{n-1} + 1)x^{n-1} + \cdots + r_1x + r_0$$

与 $R_1'(x)$ 相应的伴随式

$$S_1'(x) \equiv S_0(x) + x^{n-1} \pmod{g(x)}$$

因为纠错是在第 $n+1$ 次移位进行的，所以 $R_1'(x)$ 成为

$$R_1(x) = xR_1'(x) \equiv r_{n-2}x^{n-1} + \cdots + r_0x + r_{n-1} + 1 \pmod{x^n - 1}$$

相应的伴随式

$$S_1(x) \equiv xS_1'(x) \equiv xS_0(x) + x^n \equiv xS_0(x) + 1 \pmod{g(x)}$$

由于 $S_1(x)$ 是 $xR_1'(x)$ 的伴随式，而 $xS_0(x)$ 是 $xR(x)$ 的伴随式，也就是 $xE(x)$ 的伴随式，因此为了得到真正的 $xR(x)$ 的伴随式，就必须从 $S_1(x)$ 中消去“1”，也就是在伴随式计算电路输入端加 1。

例如，第 7 次移位后若 $S_0(x) = 1 + x^2$ ，说明 r_6 有错，第 8 次移位时对 r_6 进行纠错，纠错信号“1”输入到 $g(x)$ 电路输入端，结果使 $g(x)$ 移存器中的内容成为(000)，消除了 e_6 的影响。

(3) 若 $E(x) = x^5$ ，则 $S_0(x) \equiv x^5 \equiv x^2 + x + 1 \pmod{g(x)}$ ，此时与门不打开，说明 r_6 正确。这时伴随式计算电路和缓存器各移位一次， r_6 输出， r_5 移到缓存器最右一级，伴随式计算电路得到的伴随式是

$$S_1(x) \equiv xS_0(x) \equiv xE(x) \equiv x^2 + 1 \pmod{g(x)}$$

因此再移动一次，与门输出的纠正信号“1”正好与缓存器输出的 $r_5 = c_5 + 1$ 相加，得到了 c_5 ，从而完成了纠错。若 r_5 不错，则重复上述过程一直到译完一个码字为止。

该译码过程可用表 6-2 表示，已知 $R(x) = x^6 + x + 1$, $E(x) = x^4$ 。由该表知，到第 10 个节拍，与门输出一个“1”纠正 r_4 ，最后译码器输出码字(1010011)。

表 6-2 图 6-1 译码器译码过程

| 节拍 | 输入 $R(x)$ | 伴随式计算电路 | | | 与门输出 | 缓存输出 | 译码器输出 |
|----|-----------|---------|-------|-------|------|------|-------|
| | | D_0 | D_1 | D_2 | | | |
| 0 | | 0 | 0 | 0 | | | |
| 1 | $1(x^6)$ | 1 | 0 | 0 | | | |
| 2 | $0(x^5)$ | 0 | 1 | 0 | | | |
| 3 | $0(x^4)$ | 0 | 0 | 1 | | | |
| 4 | $0(x^3)$ | 1 | 1 | 0 | | | |
| 5 | $0(x^2)$ | 0 | 1 | 1 | | | |
| 6 | $1(x)$ | 0 | 1 | 1 | | | |
| 7 | $1(x^0)$ | 0 | 1 | 1 | | | |
| 8 | | 1 | 1 | 1 | | 1 | 1 |
| 9 | | 1 | 0 | 1 | | 0 | 0 |
| 10 | | 0 | 0 | 0 | 1 | 0 | 1 |
| 11 | | 0 | 0 | 0 | | 0 | 0 |
| 12 | | 0 | 0 | 0 | | 0 | 0 |
| 13 | | 0 | 0 | 0 | | 1 | 1 |
| 14 | | 0 | 0 | 0 | | 1 | 1 |

从上述译码过程可知，译一组码共需 $14(2n=14)$ 个节拍，仅当第一组的 $R(x)$ 移出 7 级缓存器后，才能接收第二组的 $R(x)$ 。为了使译码连续，必须再加一个伴随式计算电路，如图 6-2。

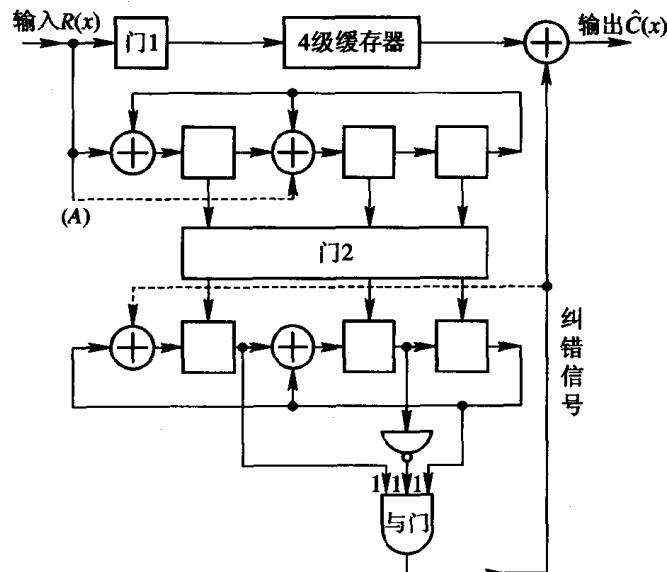


图 6-2 [7, 4] 码完整译码器

开始工作时，所有移存器的存数全为 0，门 1 开、门 2 关。当 $k=4$ 次移位后，4 级缓存器接收了前面的 4 个信息位（对系统码而言），此时门 1 关，并使 4 级缓冲器停止移位。再移动 $n-k=3$ 次后， $g(x)$ 除法电路得到了伴随式 $S_0(x)$ ，此时门 2 开，把上边 $g(x)$ 除法电路中的伴随式送到下面的伴随式计算电路中，随即门 2 关闭，且上边 $g(x)$ 除法电路立即清洗为 0。门 1 再次打开，4 级缓存器一边送出第一组的信息，一边接收第二组 $R(x)$ 的前 k 位信息组。与此同时，上边伴随式计算电路计算第二组 $R(x)$ 的伴随式，而下边伴随式计算电路，对第一组 $R(x)$ 中的信息元进行纠错。

显然，上述译码电路仅适应于系统码。若为非系统码， k 级缓存器必须变成 n 级，且还需要从已纠错过的 $\hat{C}(x)$ 中取出 k 个信息元 $\hat{m}(x)$ 。因为对非系统码而言， $C(x) = m(x)g(x)$, $m(x) = C(x)g^{-1}(x)$ 。可知： $\hat{m}(x) = \hat{C}(x)g^{-1}(x)$ 。说明译码器输出 $\hat{C}(x)$ 后，把 $\hat{C}(x)$ 再通过 $g(x)$ 除法电路，所得的商就是最终所需的估值信息组 $\hat{m}(x)$ 。

由上述讨论可得出系统循环码的一般译码器，如图 6-3。这种译码器也称梅吉特 (Meggit) 通用译码器，它的复杂性由组合逻辑电路决定。

三、扩展汉明码的译码

[$2^m, 2^m - 1 - m, 4$] 扩展汉明码是由 [$2^m - 1, 2^m - 1 - m, 3$] 汉明码加一个全校验位得到。它的码字 $(c_{n-1}, \dots, c_0, c_\infty)$ 中前 n 个码元 (c_{n-1}, \dots, c_0) 是汉明码的一个码字， c_∞ 是全校验位。扩展汉明码的码长是 8 的整数倍，特别适用于计算机或微机组的数据处理或数据传输系统中。

扩展汉明码能纠正一个错误同时发现两个错误，虽然它不是循环码，但它译码电路的

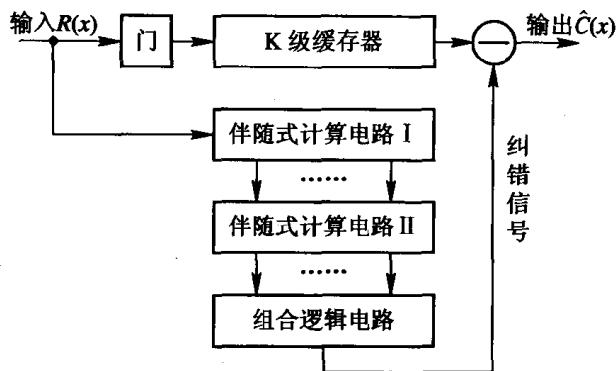


图 6-3 循环码的通用译码器(梅吉特译码器)

主要部分与循环汉明码的译码器相同，只要加上检错电路即可。

如[8, 4, 4]扩展码，只要在[7, 4, 3]循环汉明码译码器中，加一个检错电路即成，如图 6-4。图中，(a)部分的电路基本上与图 6-1 同，是循环汉明码的译码器，不同的是多加了一个全校验位检查电路，它由一个级移存器加一个模 2 加法器组成。图中，(b)部分电路是一个检错电路。该译码器的译码过程如下：

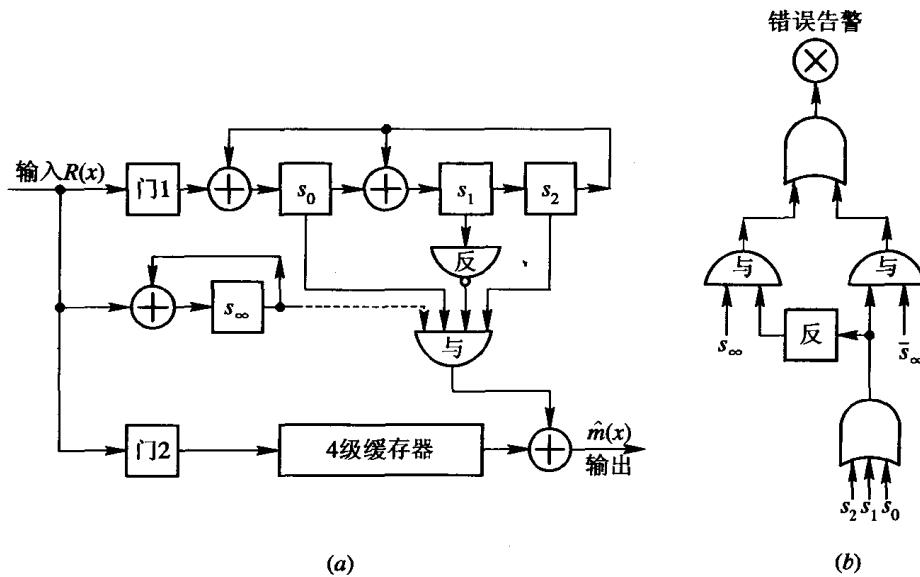


图 6-4 [8, 4, 4]扩展码译码器

(1) 开始时所有寄存器中的内容为 0，门 1 和门 2 开。移位 4 次后门 2 关， $R(x) = r_6 x^6 + \dots + r_0 + r_\infty$ 中的前 4 位(r_6, r_5, r_4, r_3)存入 4 级缓存器中，它就是待纠错的 4 个信息元。移动 7 次后门 1 关， $R(x)$ 的前 7 个码元($r_6, r_5, r_4, r_3, r_2, r_1, r_0$)，已全部送入[7, 4, 3]码所决定的伴随式计算电路中，得到了伴随式(s_2, s_1, s_0)。第 8 次移位后，在全校验位检查电路中得到了全校验的结果 s_∞ ，此时译码器不再输入。

(2) 当 $s_\infty = 0$ 、 $(s_0, s_1, s_2) = (000)$ 时，译码器认为接收 $R(x)$ 无误，把 4 级缓存器中的信息元输出。

(3) 当 $s_\infty = 1$ 、 $(s_0, s_1, s_2) \neq (000)$ 时，译码器认为有一个错误，此时纠错部分的译码电路，按上面讲的汉明码的方法进行纠错译码，4 次移位后已全部输出已纠正过的信息元。

(4) $s_\infty = 0$ 、 $(s_0, s_1, s_2) \neq (000)$ ，译码器认为出现了偶数个错误，错误告警电路输出一信号给用户，表示检测到错误。

(5) $s_\infty = 1$ 、 (s_0, s_1, s_2) 全为 0 时，译码器认为出现了一个以上的奇数个错误，错误告警电路也输出一个信号给用户。

当然，为了使译码连续，在图 6-4 的(a)部分电路中，也必须有两个伴随式计算电路，这与图 6-2 相同。

[$2^m - 1, 2^m - 2 - m, 4$] 增余删信汉明码的译码电路与扩展汉明码的译码电路基本相同，只不过全校验位的结果 s_∞ 也要输入到错误图样的识别电路与门中，对 [7, 3, 4] 码来说就是输入到图 6-4(a) 中有 3 个输入端的与门，如虚线所示，其它情况相同。

四、缩短循环码的译码

缩短 i 个信息位的 [$n - i, k - i$] 缩短循环码，是在 [n, k] 循环码中选前 i 个信息位为 0 的码字组成。若 [n, k] 循环码的码字

$$C(x) = c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \cdots + c_0$$

则 [$n - i, k - i$] 缩短循环码的码字

$$C'(x) = c_{n-1-i}x^{n-1-i} + \cdots + c_0$$

因此，缩短循环码的译码器必须在原 [n, k] 循环码译码器基础上作如下修正：

(1) k 级缓存器改为 $k - i$ 级；

(2) 为了与(1)的改动相适应， $R(x)$ 应自动乘以 x^i ，然后再输入伴随式计算电路。

如 [7, 4] 循环汉明码缩短一位变成 [6, 3] 码，它的译码器就是把图 6-2 中的译码器作如下变动： $R(x)$ 从图中(A)虚线所示的地方输入，这相当于 $R(x)$ 自动乘以 x ，4 级缓存器变成 3 级。

§ 6.2 捕 错 译 码

循环码译码器的复杂性，主要取决于由伴随式确定错误图样的组合电路的简单与否。对某些码来说，这种组合电路可以做得比较简单，如循环汉明码。除此以外，利用信息集译码中的捕错译码方法^[13]，也能够用较简单的组合逻辑电路实现译码。它特别适用于纠突发错误码、纠单个随机错误码，以及某些低码率和码长较短、纠错能力较弱的码。与其它译码方法相比，它更为简单，因而特别受到工程技术人员的欢迎。但这种译码方法对长码或高纠错能力的码来说，没有多大效果。

一、基本工作原理

设码字 $C(x)$ 是某一纠 t 个错误的 [n, k] 循环码的码字，当它通过有扰信道到达接收端译码器时成为 $R(x) = C(x) + E(x)$ 。相应的伴随式

$$S(x) \equiv R(x) \equiv E(x) \equiv E_t(x) + E_p(x) \pmod{g(x)}$$

式中

$$E_I(x) = e_{n-1}x^{n-1} + \cdots + e_{n-k}x^{n-k}$$

$$E_p(x) = e_{n-k-1}x^{n-k-1} + \cdots + e_0$$

分别是在码字信息组(或前 k 位)和校验位(或后 $n - k$ 位)上的错误图样。若 $\partial \cdot E(x) = n - k - 1$, 即所有 $\leq t$ 个错误集中在校验元的 $n - k$ 位上, 则 $E_I(x) = 0$, $E(x) = E_p(x)$ 。 $E_p(x)$ 的最高次数是 $n - k - 1$, 而 $\partial \cdot g(x) = n - k$, 所以当 $E(x) = E_p(x)$ 被 $g(x)$ 除后的余式仍为 $E_p(x)$, 即

$$S(x) \equiv E(x) = E_p(x) \pmod{g(x)}$$

说明错误图样就是 $S(x)$ 。这样, 只要把 $R(x)$ 直接减去 $S(x)$ 就进行了纠错, 得到了 $\hat{C}(x)$ 。

当然, 要所有错误全部集中在码字后 $n - k$ 位上, 实际上是很少有的情况。但是, 只要错误集中地出现在任意的连续 $n - k$ 位码段以内, 根据循环码的特点以及定理 6.1.1 所确定的伴随式性质, 把 $R(x)$ 和相应的伴随式 $S_0(x)$, 一起同时在译码器的各自移位寄存器中移位 i 次, 使这些错误全部集中到 $x^i R(x)$ 的后 $n - k$ 位上。一旦检测电路检测到伴随式 $S_i(x)$ 的重量 $w(S_i(x)) \leq t$ 时, 就认为此时的 $S_i(x) \equiv x^i S_0(x) \pmod{g(x)}$, 就是 $x^i R(x) \pmod{x^n - 1}$ 的伴随式, 并且错误已全部集中到 $x^i R(x) = R_i(x) \pmod{x^n - 1}$ 的后 $n - k$ 位以内。此时, 只要在 $R_i(x)$ 中减去 $S_i(x)$, 就对 $R_i(x)$ 进行了纠错。然后, 再把已纠错过的 $\hat{R}_i(x)$, 再循环移位 x^{n-i} 次, 即 $x^{n-i} x^i \hat{R}(x) = x^n \hat{R}(x) = \hat{R}(x) \pmod{x^n - 1}$, 便恢复到原来接收的 $R(x)$ 中各码元的顺序关系, 从而纠正了 $R(x)$ 中的错误。由于这种译码方法是通过 $S_0(x)$ 与 $R(x)$ 的循环移位运算, 把错误捕获到 $n - k$ 位低次位码元段(从 x^{n-k-1} 至 x^0 次)内, 然后再进行纠错, 故称为**捕错译码**。

显然, 只有当所有 $\leq t$ 个错误全部集中在连续 $n - k$ 位以内时, 才是这种译码方法可纠正的错误图样。

对于纠 t 个错误的循环码来说, 必须使 t 个错误能连续地出现在 $n - k$ 位以内, 这等价于要求有连续 k 位码元无错, 或错误图样中连续 k 位的值为 0。由于 t 个错误均匀分布在 n 位上时最难满足连续 k 位无错这一要求, 因此可以用捕错方法译码的 $[n, k]$ 循环码, n, k, t 之间必须满足下列条件:

$$k < n/t \quad \text{或} \quad t < n/k, \quad \text{或} \quad R < 1/t \quad (6.2.1)$$

下面一个问题是在译码过程中, 如何判断错误已全部集中在 $R_i(x)$ 的最低次的 $n - k$ 位以内。对 $[n, k]$ 循环码来说, 下面定理给出了这种判断准则。

定理 6.2.1 纠正 t 个错误的 $GF(q)$ 上的 $[n, k]$ 循环码, 捕错译码过程中已把 t 个错误集中在 $R_i(x)$ 的最低次 $n - k$ 位以内的充要条件是此时的伴随式重量

$$w(S_i(x)) \leq t \quad (6.2.2)$$

证明 若错误已集中在 $n - k$ 位低次位码元段以内, 则

$$S_i(x) \equiv x^i E(x) = E_i(x) = E_{ip}(x) \pmod{g(x)}$$

$$S_i(x) = E_{ip}(x)$$

码只能纠正 t 个错误, 若错误图样 $E(x)$ 是一个可纠正的错误图样, 则 $w(E(x)) \leq t$, 因而 $E(x)$ 的循环移位 i 次的错误图样 $E_i(x)$ 的重量也必小于等于 t , 所以

$$w(S_i(x)) = w(E_i(x)) \leq t$$

反之，若 $w(S_i(x)) \leq t$ ，则错误一定集中在 $n - k$ 位低次位码元段内。设错误没有集中在该段以内，则 $\partial \cdot E_i(x) \geq \partial \cdot (x)g(x)$ ，由此

$$E_i(x) = q(x)g(x) + S_i(x)$$

$$E_i(x) - S_i(x) = q(x)g(x) = C_i(x)$$

$C_i(x)$ 是 $g(x)$ 的倍式，由循环码性质知它必是 $[n, k]$ 循环码的一个码字。因而

$$\begin{aligned} w(E_i(x) + (-S_i(x))) &= w(C_i(x)) \geq d \\ &= 2t + 1 \end{aligned}$$

由三角不等式(3.1.2)式可知

$$\begin{aligned} w(E_i(x)) + w(-S_i(x)) &\geq w(E_i(x) + (-S_i(x))) \\ &\geq 2t + 1 \end{aligned}$$

因为

$$w(E_i(x)) \leq t$$

所以

$$w(-S_i(x)) \geq t + 1 > t$$

由于 $w(-S_i(x)) = w(S_i(x))$ ，因此上式与假设 $w(S_i(x)) \leq t$ 相矛盾，因而错误没有集中在 $n - k$ 位低次位以内的反证法假设不能成立，故错误集中在 $n - k$ 位低次位内。■

例 6.2 二进制 $[15, 7, 5]$ 循环码，生成多项式 $g(x) = x^8 + x^7 + x^6 + x^4 + 1$ ，能纠正两个错误。

$$t = 2 < 15/7$$

满足捕错译码的必要条件式(6.2.1)，可以用捕错译码方法译码，它的译码电路如图 6-5。其译码过程如下：

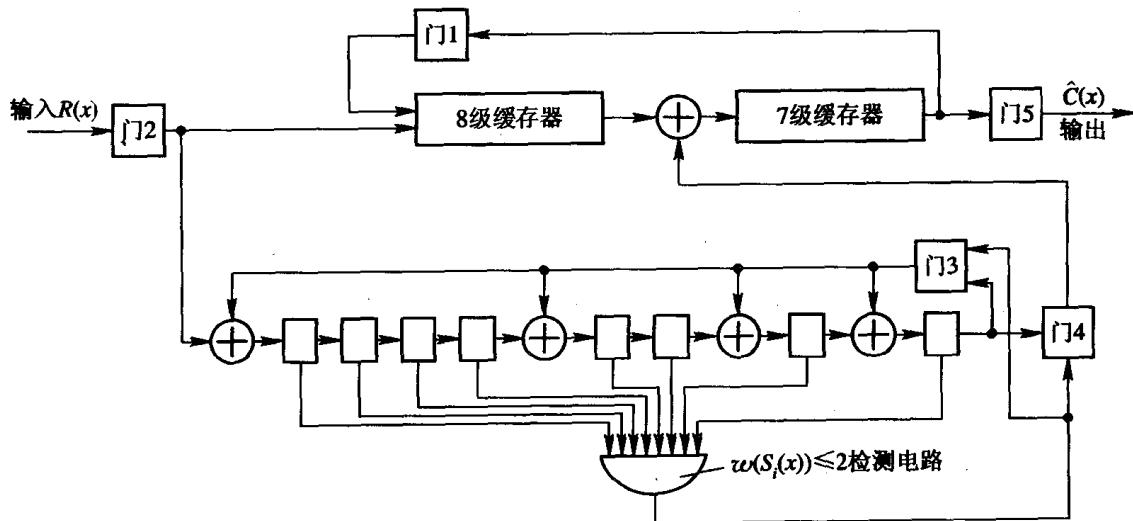


图 6-5 $[15, 7, 5]$ 循环码捕错译码电路

(1) 开始工作时，所有移存器和缓存器清洗为 0，门 2、门 3 开，门 1、门 4 和门 5 关闭。 $n=15$ 次移位后， $R(x)$ 的 15 个码元全部移入 $(8+7)=15$ 级缓存器，信息元在前 7 级，同时伴随式计算电路也完成了伴随式计算得到了 $S_0(x)$ 。若 $S_0(x) = 0$ ，说明无错，打开门

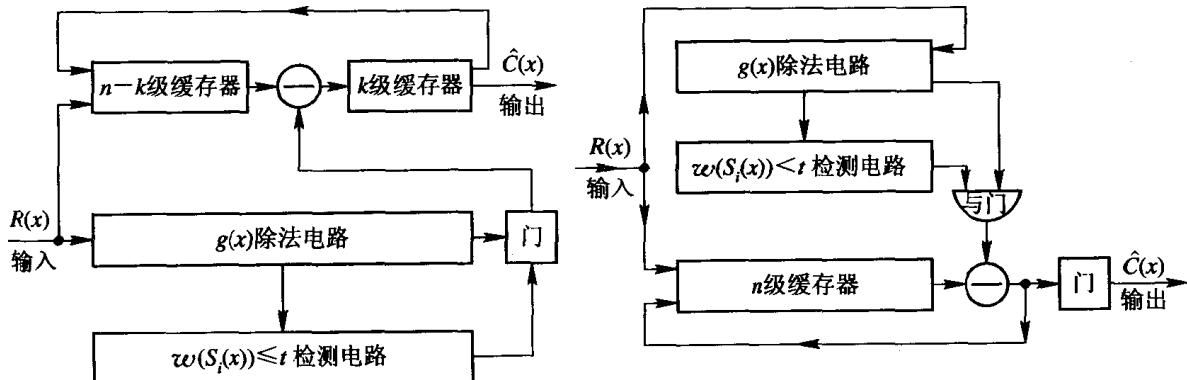
5, 输出缓存器中的 7 个信息元。若 $S_0(x) \neq 0$, 则进行以下各步。

(2) 此时门 2 关, 门 1 开, 若 $w(S_0(x)) \leq 2$, 检测电路检测到伴随式的重量 ≤ 2 , 打开门 4, 关闭门 3。此时存在伴随式寄存器中的内容就是 $R(x)$ 后 8 位内的错误图样, 移位 15 次, 在移位过程中, 伴随式中的错误图样 $S_0(x) = E_p(x)$ 与 $R(x)$ 中的错误图样逐位模 2 加, 完成了纠错。然后门 1 关闭, 再移位 15 次, 输出已纠错的 $\hat{C}(x)$ 。

(3) 若 $w[S_0(x)] > 2$, 则 15 级缓存器和 $g(x)$ 除法电路都循环移位一次, 并检查 $w[S_1(x)]$ 的重量, 若仍大于 2, 则继续循环移位。如果第 i 次循环移位后, 重量检测电路检测到 $w[S_i(x)] \leq 2$, 则说明错误已捕获到此时 $R_i(x) = x^i R(x) \pmod{x^n - 1}$ 的最后 8 位中, 这时门 3 关闭, 门 4 打开, 继续移位 $n-i=15-i$ 次, 伴随式寄存器中输出的错误图样与缓存器中 $R_i(x)$ 的后 8 位中的码元逐位模 2 加, 以纠正其中的错误。15-i 次移位完毕后纠错结束, 并且 $r_{14} = \hat{c}_{14}$ 又处在 7 级缓存器的最前(右)一级, 此时, 门 1 关, 门 5 和门 2 打开, 译码器再移动 15 次, 就把缓存器中的结果 $\hat{C}(x)$ 全部送出, 同时缓存器接收下一组的输入 $R(x)$ 。2n 次移位后, 若始终没有 $w(S_i(x)) \leq 2$, 则说明 $R(x)$ 中有不可纠错误。■

从上述纠错过程看到, 要完成一个码组的纠错, 译码器共需移动 $2n$ 次, 若再把 $\hat{C}(x)$ 串行输出, 还需要移动 n 次, 共需 $3n$ 次移位节拍才能输出一个已纠错的码组。当然, 为了使译码器连续工作, 像图 6-3 那样必须有两套伴随式计算电路工作。

$[n, k]$ 循环码的一般捕错译码器如图 6-6 和图 6-7, 它们分别称为第一类和第二类捕错译码器。



第二类译码器与第一类的差别, 仅在于第二类译码器是把接收到的 $R(x)$ 自动乘以 x^{n-k} 后, 再进入 $g(x)$ 除法电路计算伴随式, 即 $R(x)$ 从 $g(x)$ 电路的最高次位送入。所以

$$\begin{aligned} R(x)x^{n-k} &\equiv (r_{n-1}x^{n-1} + r_{n-2}x^{n-2} + \dots + r_0)x^{n-k} \\ &\equiv r_{n-1}x^{n-k-1} + r_{n-2}x^{n-k-2} + \dots + r_k + r_{k-1}x^{n-1} \\ &\quad + \dots + r_1x^{n-k+1} + r_0x^{n-k} \pmod{x^n - 1} \end{aligned} \quad (6.2.3)$$

由该式看出, 第二类译码电路对 $R(x)$ 的前 r_{n-1} 至 r_k 位中的错误进行纠正。当接收完 $R(x)$ 后, 伴随式重量检测电路若检测到 $w(S_0(x)) \leq t$, 则说明错误产生在 r_{n-1} 至 r_k 位中, 此时检测器控制 $g(x)$ 电路中的 $S_0(x)$ 串行输出, 与 n 级缓存器的输出进行相减以纠正错误。若 $w(S_0(x)) > t$, 则 $g(x)$ 除法电路与 n 级缓存器各自循环移位一次, 再检查 $w(S_1(x))$, 直

至移位检查 n 次，对 $R(x)$ 的每一位循环检查了一遍。然后，打开输出控制门，再移动 n 次，便把 n 级缓存器中的已纠错过的码字全部送给用户。因此，第二类译码器也需 $3n$ 次移位才能输出已纠错的码字。

由于第二类译码器首先对 r_{n-1} 至 r_k 纠错，因此对系统码来说，在纠错过程中可以一边纠错一边送出已纠正过的信息元 $\hat{c}_{n-1} = \hat{m}_{k-1}$, $\hat{c}_{n-2} = \hat{m}_{k-2}$ …直接给用户，故仅需 $2n$ 次移位。译码器就完成了纠错并把信息送至用户。但这样做了后，重量 $\leq t$ 的处在码字首尾的循环错误图样不能纠正。

捕错译码方法不仅能应用于某些纠随机错误的循环码，而且还是纠突发错误循环码的主要译码方法，这将在以后讨论。

由上讨论可知，捕错译码要求接收码字中连续 k 位码元无错，而这仅是信息集译码的充分条件，而并非必要，因此捕错译码是信息集译码的一种易于工程实现的特殊情况^[13]。

二、捕错译码的修正

捕错译码是假定错误能集中在 $n - k$ 段以内的前提下进行的，要求 $[n, k]$ 循环码必须满足式(6.2.1)的必要条件。但是，能满足此条件的纠随机错误的循环码很少，只有纠正一个错误的循环汉明码和 $[15, 7, 5]$ 码等，而绝大部分循环码均不满足。可是，有些循环码如 $[15, 5, 7]$ 码， $[23, 12, 7]$ Golay 码及 $[17, 9, 5]$ QR 码等，虽不满足式(6.2.1)的条件，但 k 仅比 n/t 稍大或相等，也就是说在译码过程中不能把错误全部集中在 $n - k$ 个码元段以内，而有个别错误可能在其它码元位上。为了解决此问题，必须对捕错译码加以适当修正。

译码过程中，若译码器能把大部分错误捕捉到 $n - k$ 个码元段以内，同时使个别错误进入某几个预先指定的位上，则也能确定此时的错误图样。当然，为了实现这种运算必须附加一些电路，以确定错误是否在某几个指定的位置。

设 $[n, k]$ 循环码能纠正 t 个错误，生成多项式是 $g(x)$ 。译码器收到 $R(x)$ 后，计算伴随式

$$\begin{aligned} S_o(x) &\equiv E(x) = E_I(x) + E_p(x) \equiv S_I(x) + E_p(x) \pmod{g(x)} \\ S_I(x) &\equiv E_I(x) = e_{n-1}x^{n-1} + \dots + e_{n-k}x^{n-k} \pmod{g(x)} \\ E_p(x) &\equiv S_o(x) - S_I(x) \pmod{g(x)} \end{aligned} \quad (6.2.4)$$

式中， $S_I(x)$ 是 $R(x)$ 前 k 位(对系统码来说就是信息位)码段中错误图样 $E_I(x)$ 的伴随式。由上式知，若 $E_I(x)$ 及其 $S_I(x)$ 是已知的，则可根据式(6.2.4)得到 $R(x)$ 中后 $n - k$ 位内(校验位)的错误图样 $E_p(x)$ ，从而确定出原来的错误图样 $E(x) = E_I(x) + E_p(x)$ 。

设 $\{Q_j(x)\}$ 是次数小于等于 $k - 1$ 次的二进制多项式集合，

$$S_{I_j}(x) \equiv x^{n-k}Q_j(x) \pmod{g(x)} \quad I_j = 0, 1, \dots$$

是 $x^{n-k}Q_j(x)$ 的伴随式，即如果在前 k 位上错误图样 $E_I(x) = x^{n-k}Q_j(x)$ ，则 $S_{I_j}(x)$ 就是它的伴随式。因此，如果任何一个重量 $\leq t$ 的错误图样 $E(x)$ ，或它的 i 次循环移位 $x^iE(x) = E_i(x)$ ，在前 k 位码段内与 $\{Q_j(x)\}$ 中的任一个相一致，则把此时的 $S_{I_j}(x)$ 与此时 $E_i(x)$ 的伴随式 $S_i(x)$ 相减，由式(6.2.4)知：

$$E_{p_i}(x) = S_i(x) - S_{I_j}(x) \quad (6.2.5)$$

它就是此时 $R_i(x) = x^iR(x)$ 的后 $n - k$ 位上的错误图样。由于码能纠正 t 个错误，且

$w(E(x)) \leq t$, 所以在某一时刻连续 k 位中的错误图样的重量为 $w(Q_j(x))$, 则此时后面连续 $n - k$ 位中错误图样 $E_{p_i}(x) = E_i(x) - x^{n-k}Q_j(x)$, 它的重量

$$w(E_{p_i}(x)) = w(S_i(x) - S_{I_j}(x)) \leq t - w(Q_j(x)) \quad (6.2.6)$$

所以, 当式(6.2.6)成立时, 就认为错误图样 $E_i(x)$ 中前 k 个连续位上的图样与 $Q_j(x)$ 相一致。由此可从式(6.2.5)得到原来的错误图样

$$E(x) \equiv x^{n-i}E_i(x) \equiv x^{n-i}[S_i(x) - S_{I_j}(x) + x^{n-k}Q_j(x)] \pmod{x^n - 1}$$

如果 $R(x)$ 是从 $g(x)$ 的高次位送入(第二类译码器), 相当于 $R(x)x^{n-k}$, 则此时

$$E(x) \equiv x^{n-k-i}(S_i(x) - S_{I_j}(x) + x^{n-k}Q_j(x)) \pmod{x^n - 1}$$

由上讨论不难知道, 对任何一个 $w(E(x)) \leq t$ 的错误图样, 若在译码过程中前 k 位内的错误图样能被 $\{Q_j(x)\}$ 所包含或覆盖(所以称 $Q_j(x)$ 为覆盖多项式), 也就是 $E(x)$ 或 $E(x)$ 的任何循环移位 $x^iE(x)$ 中除了在 $Q_j(x)$ 所确定的相应位置上的值不为 0 以外, 其它信息位置上均为 0, 则就可以用这种修正捕错误码方法译码。其译码过程如下:

- (1) $R(x)$ 一方面送入 $g(x)$ 电路计算伴随式 $S_0(x)$, 一面送入 n 级缓存器;
- (2) 在 $g(x)$ 除法电路中进行无输入的循环移位运算(自发运算), 由 $S_0(x)$ 计算 $S_i(x)$ ($i=1, 2, \dots, n-1$), 同时 n 级缓存器也相应地进行循环移位运算(模 $x^n - 1$ 运算);
- (3) 对每一个 i 计算

$$w(S_i(x) - S_{I_j}(x)) \quad j = 1, 2, \dots, N$$

式中, N 是 $\{Q_i(x)\}$ 中的多项式数目。若对某一个 i 和 j 有

$$w(S_i(x) - S_{I_j}(x)) \leq t - w(Q_j(x))$$

则说明此时 $E_i(x)$ 的前 k 位中的错误图样与 $Q_j(x)$ 图样相一致, 所以总的错误图样

$$E_i(x) = S_i(x) - S_{I_j}(x) + x^{n-k}Q_j(x) = E_{p_i}(x) + E_{I_j}(x)$$

把 $R_i(x)$ 减去此 $E_i(x)$, 就进行了纠错, 得到了此时的 $\hat{C}_i(x)$:

$$\hat{C}_i(x) = R_i(x) - E_i(x) \equiv x^i\hat{C}(x) \pmod{x^n - 1}$$

再把 $\hat{C}_i(x)$ 循环移位 $n-i$ 次, 就得到了已纠正的码字

$$\hat{C}(x) \equiv x^i\hat{C}(x)x^{n-i} \pmod{x^n - 1}$$

- (4) 循环移位 n 次后, 对所有 $j=1, 2, \dots, N$, 若均有

$$w(S_i(x) - S_{I_j}(x)) > t - w(Q_j(x))$$

则表示 $R(x)$ 中的错误图样是不可纠正的, 从而发现了错误。

例 6.3 [23, 12, 7]Golay 码, 它的生成多项式 $g(x) = x^{11} + x^{10} + x^6 + x^5 + x^4 + x^2 + 1$ 。该码若用循环码的其它方法译码比较复杂, 但用修正捕错译码则比较简单。

该码的 n, k 与 t 之间关系不满足式(6.2.1), 不能用捕错译码, 必须用修正方法。

首先选择在前 k 位中的覆盖多项式集合 $\{Q_j(x)\}$, 经分析可知应选 $\{0, x^5, x^6\}^{(2)}$ 。其中, x^5, x^6 是特定在信息位置上的错误(对系统码而言), 而 0 表示错误能全部集中在后 $n-k$ 位内, 不需要在前 k 位上指定错误的情况。

由 $\{0, x^5, x^6\}$ 可得:

$$x^{n-k}Q_1(x) = x^{11}0 = 0$$

$$x^{n-k}Q_2(x) = x^{11}x^5 = x^{16}$$

$$x^{n-k}Q_3(x) = x^{11}x^6 = x^{17}$$

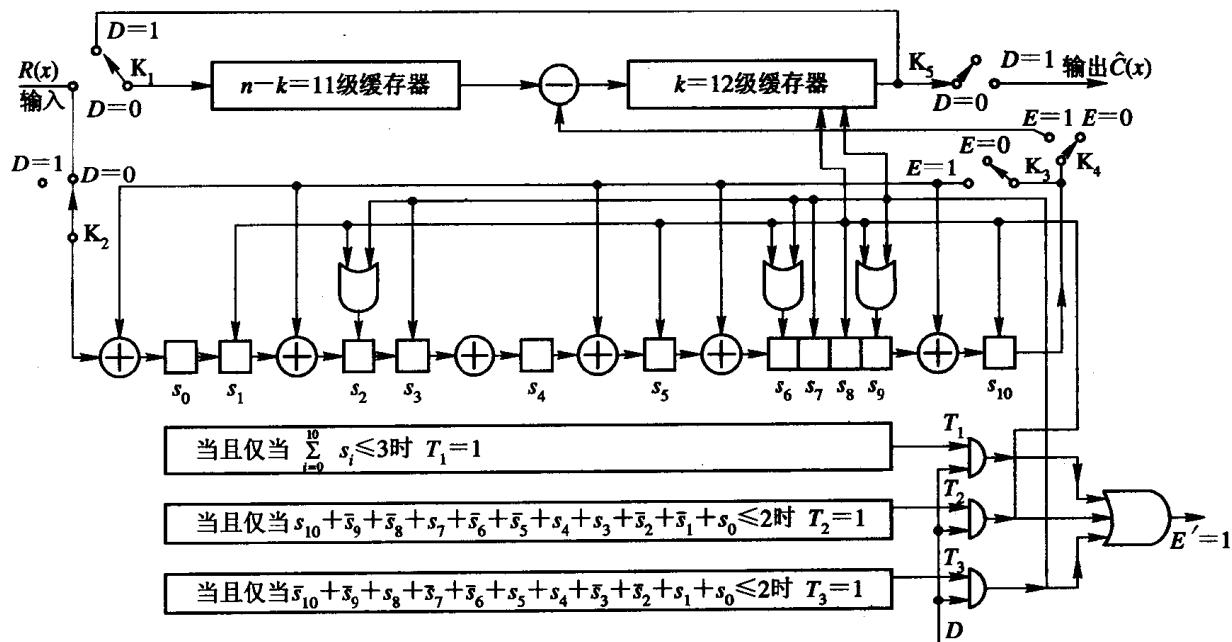
由此得到相应的

$$S_{I_0}(x) \equiv 0 \pmod{g(x)}$$

$$S_{I_1}(x) \equiv x^{16} \equiv x^9 + x^8 + x^6 + x^5 + x^2 + x \pmod{g(x)}$$

$$S_{I_2}(x) \equiv x^{17} \equiv x^{10} + x^9 + x^7 + x^6 + x^3 + x^2 \pmod{g(x)}$$

[23, 12]Golay 码的修正捕错译码器如图 6-8, 图中, $\bar{S}_i = 1 + S_i$ 表示取 S_i 的加法逆元。译码过程如下:



(1) 开始时所有移存器中的存数均清洗为 0, 开关 K_1 、 K_2 和 K_5 接至 $D=0$ 的位置, K_3 和 K_4 接至 $E=0$ 的位置。接收到的 $R(x)$ 一方面送入 23($=11+12$) 级缓存器中, 一方面送入伴随式计算电路计算伴随式。23 次移位后, 得到伴随式

$$S_0(x) = s_{10}x^{10} + s_9x^9 + \cdots + s_0$$

把它送至 3 个错误图样检测电路;

(2) 开关 K_1 、 K_2 和 K_5 接到 $D=1$ 的位置, K_3 和 K_4 仍在 $E=0$ 的位置, 由错误图样检测电路检测可纠正的错误图样:

① 若 $w(S_0(x)) \leq 3$, 认为错误全在后 11 位中, 且错误图样就是 $S_0(x)$, 此时 $T_1 = 1$ 导致 $E' = 1$, K_3 和 K_4 接至 $E=1$ 位置, 并在移位过程中对 11 级缓存器的输出进行纠错。

② 若 $w(S_0(x) - S_{I_1}(x)) \leq 2$, 则

$$\begin{aligned} S_0(x) - S_{I_1}(x) &= (s_{10}x^{10} + s_9x^9 + \cdots + s_1x + s_0) - (x^9 + x^8 + x^6 + x^5 + x^2 + x) \\ &= s_{10}x^{10} + \bar{s}_9x^9 + \bar{s}_8x^8 + s_7x^7 + \bar{s}_6x^6 + \bar{s}_5x^5 \\ &\quad + s_4x^4 + s_3x^3 + \bar{s}_2x^2 + \bar{s}_1x + s_0 \end{aligned}$$

它就是此时在后 11 位上的错误图样，而在前 12 位中的错误图样相应地为 x^{16} 。上式中 $\bar{s}_i = s_i + 1 = s_i - 1$ (二进制情况)。此时 $T_2=1$ 导致 $E'=1$ ，控制 K_3 和 K_4 接至 $E=1$ 的位置，由 T_2 引出连线将 x^{16} 位置上的错误纠正，同时由 $S_0(x)$ 减去 $S_{I_1}(x)$ ，把伴随式中的存数变成

$$E_{P_1}(x) = S_0(x) - S_{I_1}(x)$$

在移位过程中，缓存器输出减去伴随式移存器的输出，实现了纠错。

③ 若 $w(S_0(x) - S_{I_2}(x)) \leq 2$ ，则

$$\begin{aligned} S_0(x) - S_{I_2}(x) &= (s_{10}x^{10} + s_9x^9 + \dots + s_1x + s_0) - (x^{10} + x^9 + x^7 + x^6 + x^5 + x^2) \\ &= \bar{s}_{10}x^{10} + \bar{s}_9x^9 + s_8x^8 + \bar{s}_7x^7 + \bar{s}_6x^6 + s_5x^5 \\ &\quad + s_4x^4 + \bar{s}_3x^3 + \bar{s}_2x^2 + s_1x + s_0 \end{aligned}$$

就是在后 11 位上的错误图样，而在前 12 位中的错误图样相应地为 x^{17} 。此时 $T_3=1$ ，使 $E'=1$ ， K_3 和 K_4 接至 $E=1$ 的位置，由 T_3 引出连线送一信号将 x^{17} 位上的错误纠正。同时用 $S_0(x)$ 减去 $S_{I_2}(x)$ ，使伴随式移存器中的存数变为

$$E_{P_2}(x) = S_0(x) - S_{I_2}(x)$$

在移位过程中，缓存器的输出减去伴随式移存器的输出而实现纠错。

④ 若 $w(S_0(x)) > 3$ ，或 $w(S_0(x) - S_{I_1}(x)) > 2$ ，或 $w(S_0(x) - S_{I_2}(x)) > 2$ ，则进行下一步；

(3) 把 23 级缓存器和伴随式计算电路同时循环移位一次，对新得到的伴随式 $S_1(x)$ ，进行如同第(2)步的检查；

(4) 当缓存器和伴随式计算电路共移位 $2n$ 次后，译完了一个码字。此时 K_1 、 K_2 和 K_5 接至 $D=0$ 的位置， K_3 和 K_4 接到 $E=0$ 的位置；

(5) 送入第二组 $R(x)$ ，同时第一组已纠过错的码元由 23 级缓存器输出。所以，接收到的 $R(x)$ 从输入译码器直至输出完为止，共需移位 $3n(3 \times 23)$ 次。

通常情况下，若移位 $3n$ 次（若 $R(x)$ 从 $g(x)$ 的高次位输入，则仅需 $2n$ 次）后伴随式移存器中的存数不为全 0，则表示发现不可纠的错误。此时，译码器应输出一信号，告知用户这组码字有不可纠的错误，并将电路恢复到初始状态。本例中，由于 [23, 12] 码是完备码，不可能出现这种情况，所以当错误个数大于 3 时，译码器将产生错译。 ■

虽然上面讨论的捕错译码及其修正仅局限于二进制码，但事实上这些译码方法能适应于 q 进制循环码。最近的研究表明^[3]，码率 $R < 2/3$ 的纠 2 个和纠 3 个错误的非二进制码，也可以用修正捕错译码方法译码，但译码原理和过程与二进制码稍有不同。这种译码方法比目前已知的非二进制码的其它译码方法都要简单。

三、覆盖多项式的数目与建立

无论是对二进制还是 q 进制码，修正捕错译码器的复杂性主要决定于覆盖多项式集合 $\{Q_j(x)\}$ 中，覆盖多项式数目 j 的多少，若 j 过大，则译码器太复杂而不能实用。对于一个特定的循环码是否一定能找到覆盖多项式？若能找到，则应如何找 $\{Q_j(x)\}$ ？最小的 j 应是多少？这些是捕错译码能否实用的关键问题。

对于满足式(6.2.1)的码，也就是错误能全部集中到连续的 $n-k$ 位以内，或能保证连续 k 位无错，则覆盖多项式必定存在，就是 $Q_0(x) = 0$ ， $j=1$ ；如果码不满足式(6.2.1)，但

满足以下关系式：

$$R < 2/t \quad \text{或} \quad 2n/t > k \quad (6.2.7)$$

也就是所有 t 个错误的图样都能保证至少有 $\lceil k/2 \rceil$ 个连续位无错，(这 $\lceil x \rceil$ 表示大于或等于 x 的最小整数)。在这种情况下覆盖多项式集合一定存在，并且当 $t=2$ 和 $t=3$ 时，可以计算 j 的大小以及找出覆盖多项式。但是，为了分析问题简单起见，我们这里及以后所指的 $\{Q_j(x)\}$ 集合中，所有 $Q_j(x)$ 都是单项式 0 和 $Q_j(x) = x^{n-k+a}$ 的形式， a 是正整数，而不讨论 $Q_j(x) = x^i + x^j + \dots$ 的多项式形式。

引理 6.2.1 对于纠 t 个错误的 $GF(q)$ 上的 $[n, k]$ 循环码，当且仅当 $R < 2/t$ 时，覆盖多项式集合必存在。

证明 若 $t \geq R/2$ ，则不能找到一个覆盖多项式集合 $\{Q_j(x)\} = \{0\} \cup \{x^{n-k+a}, 0 \leq a \leq k\}$ ，它能覆盖所有的 t 个错误图样。因为至少存在有这样一个图样，两个错误元间的间隔必小于 $k/2$ ，因此不论何种单项式覆盖多项式 $Q_j(x) = x^{n-k+a}$ ，均不能保证除了 $E(x)$ 或 $x^i E(x)$ 中的连续 k 位内，只有 x^{n-k+a} 位置上为非 0，而其它位置上为 0。反之，若 $t < 2/R$ ，则 t 个或小于 t 个错误图样中必至少有 3 个连续错误，第一个和第三个错误至少间隔 $\lceil 2n/t \rceil$ 位，因 $\lceil 2n/t \rceil > k$ ，因此集合 $\{0\} \cup \{x^{n-k+a}, 0 \leq a < k\}$ 必是覆盖多项式集合。 ■

定理 6.2.2 对于 $t=2$ ，覆盖多项式 $\{Q_j(x)\}$ 中，最少的多项式数目为

$$j = \lceil n/(2(n-k)) \rceil$$

证明 或者集合 $\{0, x^{2(n-k)-1}, x^{3(n-k)-1}, \dots, x^{b(n-k)-1}\}$ ($b = \lceil n/(2(n-k)) \rceil$) 是覆盖多项式集合，或者两个错误之间分隔 1, 2, … 或 $n-1$ 位无错。一个单项式的覆盖多项式 x^{n-k+a} 至多覆盖 $2(n-k)$ 位，即： $a+1, a+2, \dots, a+n-k, n-a-1, n-a-2, \dots, k-a$ 位，因此至少需 $\lceil n/(2(n-k)) \rceil$ 个覆盖单项式。 ■

定理 6.2.3 对于 $t=3$ ，且 $R < 2/3$ ，则以下步骤能决定覆盖多项式集合的大小，并找出覆盖多项式集合。令 $b_0 = \lceil n/3 \rceil$, $b_1 = k+1-b_0$ 。对于 $i \geq 2$ ，令 $b_i = b_{i-2} + b_{i-1} - (n-k+1)$ 。因为 $k < (2/3)n$ ，我们有 $n-k+1 > b_0 \geq b_1 > b_2 > b_3 \dots$ 。令 c 是使 $b_c > 0$, $b_{c+1} \leq 0$ ，则 $1+c$ 是 $\{Q_j(x)\}$ 中的最小 j 。并且 $\{Q_j(x)\} = \{0, x^{n-k-1+b_1}, x^{n-b_2}, x^{n-k-1+b_3}, x^{n-b_4}, \dots, x^{n-k-1+b_c}, (或 x^{n-b_c})\}$ 是最小的覆盖多项式集合。若 c 是奇数，则 $\{Q_j(x)\}$ 中的最后一项为 $x^{n-k-1+b_c}$ ；若是偶数，则为 x^{n-b_c} 。

该定理的证明比较复杂，请参阅 [3]。

例如 $[17, 9, 5]$ QR 码， $t=2$, $R=0.53 < 1$ 。由定理 6.2.2 可知， $j = \lceil 17/16 \rceil = 2$, $Q_j(x) = \{0, x^{2(n-k)-1=15}\}$ 。

对于循环码来说，覆盖多项式集合不是唯一的，如 $[17, 9, 5]$ QR 码的覆盖多项式也可以是 $\{0, x^8\}$ 。又例如 $[23, 12, 7]$ Golay 码， $t=3$, $R=0.52 < 2/3$ 。由定理 6.2.3 可得 $b_0 = \lceil 23/3 \rceil = 8$, $b_1 = k+1-b_0 = 5$, $b_2 = b_0 + b_1 - (n-k+1) = 8+5-12=1$, $b_3 = b_1 + b_2 - (n-k+1) = 5+1-(12)=-6$ ，所以 $c=2$ ，因此 $j=1+c=3$, $\{Q_j(x)\} = \{0, x^{15}, x^{22}\}$ ，但在例 6.3 中的 $Q_j(x) = \{0, x^{11+5}, x^{11+6}\}$ 。

表 6-3^[3, 4] 给出了某些二进制循环码的覆盖多项式集合及大小，表中仅列出了 $\{Q_j(x)\}$ 中除 0 以外的所有单项式，并且表中还列出了某些 $t > 3$ 二进制循环码的覆盖多项式，有关寻找这些覆盖多项式的详细讨论，请参阅文献 [3, 4]。

表 6-3 某些二进制循环码的覆盖多项式

| n | k | t | $Q_j(x)$ | $j-1$ | n | k | t | $Q_j(x)$ | $j-1$ |
|-----|-----|-----|--|-------|-----|-----|-----|----------------------------|-------|
| 15 | 5 | 3 | 10(表示 x^{10} , 下同) | 1 | 39 | 26 | 2 | 19(20, 21, 22, 23, 24) | 1 |
| 17 | 9 | 2 | 8(10等) | 1 | 39 | 15 | 4 | 24, 29, 34 | 3 |
| 21 | 9 | 3 | 15(14)(表示 x^{15} 或 x^{14} , 下同) | 1 | 39 | 13 | 5 | 28, 31, 34 | 3 |
| 21 | 7 | 3 | 14 | 1 | 43 | 29 | 2 | 21(22, 23, 24, 25, 26) | 1 |
| 23 | 12 | 3 | 15, 22(表示 x^{15} 和 x^{22} , 下同) | 2 | 45 | 28 | 2 | 28(27, 26, 25, 24, 23, 22) | 1 |
| 23 | 11 | 3 | 15 | 1 | 45 | 27 | 2 | 27(26, 25, 24, 23, 22) | 1 |
| 31 | 21 | 2 | 19 | 1 | 45 | 24 | 2 | 24(23, 22) | 1 |
| 31 | 20 | 2 | 21 | 1 | 45 | 23 | 3 | 30, 44 | 2 |
| 31 | 16 | 3 | 20, 30 | 2 | 45 | 22 | 3 | 30 | 1 |
| 31 | 15 | 3 | 20 | 1 | 45 | 21 | 3 | 30 | 1 |
| 31 | 11 | 5 | 21, 23, 25, 27 | 4 | 45 | 18 | 3 | 31(32, 33, 34等) | 1 |
| 33 | 10 | 4 | 21, 25, 29 | 3 | 45 | 16 | 4 | 31, 38 | 2 |
| 33 | 11 | 5 | 24, 27, 30 | 3 | 45 | 9 | 5 | 36(37, 38等) | 1 |
| 35 | 26 | 2 | 17 | 1 | 51 | 35 | 2 | 35(25, 26等) | 1 |
| 35 | 16 | 3 | 25 | 1 | 51 | 27 | 3 | 34, 44 | 2 |
| 35 | 13 | 3 | 23(24, 25)(表示 x^{23} 或 x^{24} 或 x^{25} , 下同) | 1 | 51 | 19 | 4 | 37, 44 | 2 |
| 35 | 7 | 6 | 30 | 1 | 51 | 17 | 5 | 37, 41, 45 | 3 |

§ 6.3 大数逻辑译码原理

循环码译码器的复杂性主要决定于码的代数结构。纠单个错误的汉明码、纠单个突发错误的循环码，以及某些纠错能力较低的码，可以用以前所介绍的比较简单的方法如捕错译码。但是，对绝大多数纠多个随机错误的循环码来说，却没有如此简单的译码方法。不过，有一类循环码的子类，码的结构比较特殊，可以用较简单的大数逻辑方法译码，这类码称为**大数逻辑可译码**。

大数逻辑译码是 1954 年里德(Reed)，在译里德-马勒尔(Reed - Muller, RM)码时提出的一种较简单的译码方法^[1, 2]。可是对于一般码长而言，能用大数逻辑方法译码的大数逻辑可译码，其纠错能力和码率都稍次于有相同参数的其它码如 BCH 码，但由于它的译码算法和设备均比相应的 BCH 码译码方法要简单，因此在实际中有较广的应用。

一、大数逻辑译码的基本概念

通过以下例子介绍二进制循环码的大数逻辑译码的基本思想。 $[7, 3, 4]$ 增余删信汉明

码，它的生成多项式 $g(x) = (x+1)(x^3+x+1) = x^4 + x^3 + x^2 + 1$ ，校验矩阵是

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

设接收的 n 重 $\mathbf{R} = \mathbf{C} + \mathbf{E}$ ， $\mathbf{C} = (c_6, c_5, \dots, c_0)$ 是发送的码字， $\mathbf{E} = (e_6, e_5, \dots, e_0)$ 是错误图样。相应的伴随式

$$\mathbf{S}^T = \mathbf{H} \cdot \mathbf{E}^T = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} e_6 \\ e_5 \\ \vdots \\ e_0 \end{bmatrix} = \begin{bmatrix} s_3 \\ s_2 \\ s_1 \\ s_0 \end{bmatrix}$$

对伴随式的分量 s_0, s_1, s_2 和 s_3 进行线性组合，也就是对 \mathbf{H} 矩阵的行进行线性组合，得到以下一组校验方程：

$$\begin{aligned} A_1 &= s_3 = e_6 + e_4 + e_3 \\ A_2 &= s_1 = e_6 + e_5 + e_1 \\ A_3 &= s_2 + s_0 = e_6 + e_2 + e_0 \end{aligned} \quad (6.3.1)$$

显然，上述方程的系数所组成的 3 个七重数组：(1011000)，(1100010) 和 (1000101) 是 \mathbf{H} 行的线性组合，在 \mathbf{H} 的行空间中。用 A_1, A_2 和 A_3 代表这 3 个七重，则码字 \mathbf{C} 满足：

$$\mathbf{C} \cdot \mathbf{A}_1^T = \mathbf{C} \cdot \mathbf{A}_2^T = \mathbf{C} \cdot \mathbf{A}_3^T = 0$$

或

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_6 \\ c_5 \\ \vdots \\ c_0 \end{bmatrix} = \mathbf{H}_0 \cdot \mathbf{C}^T = 0 \quad (6.3.2)$$

可知：

$$\begin{aligned} A_1 &= c_6 + c_4 + c_3 = 0 \\ A_2 &= c_6 + c_5 + c_1 = 0 \\ A_3 &= c_6 + c_2 + c_0 = 0 \end{aligned}$$

得到一组新的校验方程。它们有以下特点： c_6 含在每一方程之中， c_5, c_4, c_3, c_2, c_1 和 c_0 只含在某一方程中。有这种特点的校验方程称为正交于 c_6 （或 x^6 、 e_6 ）码元位的正交校验方程（或正交一致校验和式），由它们的系数所组成的矩阵 \mathbf{H}_0 ，称为正交一致校验矩阵。所以，由式 (6.3.2) 可得该码的正交校验矩阵

$$\mathbf{H}_0 = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

定义 6.3.1 若某一特定码元位（如 x^{n-i} ）出现在 \mathbf{H}_0 矩阵中 J 行的每一行中，而其它码元位至多在其中一行出现，则称 \mathbf{H}_0 为正交于该码元位 (x^{n-i}) 的正交一致校验矩阵。

[7, 3, 4] 码能纠正一个错误同时发现两个错误，由上面的 \mathbf{H}_0 矩阵或式 (6.3.2) 可以

发现：

- (1) 发生一个错误，且在正交码元位上： $e_6=1$ ，则可得 $A_1=1, A_2=1, A_3=1$ 。
- (2) 发生一个错误但不在正交码元位上： $e_6=0, e_i=1$ (i 取 5, 4, 3, 2, 1, 0 中的某一个)，则只有一个 $A_j=1$ ($j=1, 2, 3$)，其它两个 A 均为 0。
- (3) 发生两个错误，其中一个在正交位上，另一个在其它位(如 e_5)： $e_6=1, e_5=1$ ，则 $A_1=1, A_2=0, A_3=1$ 。
- (4) 发生两个错误，都不在正交位上，设 $e_5=1, e_4=1$ ，则 $A_1=1, A_2=1, A_3=0$ 。

可知，若发生一个错误且在正交位(e_6 或 x^6)上，则 A_i 中 1 的数目大于 2 为 3；若发生在其余码元位上，则 A_i 中 1 的数目小于 2 为 1。若同时发生两个错误，则不论发生在什么位置， A_i 中 1 的数目均是 2。因此，可以根据 A_i 中取值为 1 的个数多少，对 x^6 码元位的值 r_6 纠错。由于是循环码，当对 x^6 码元位校验后，再循环移位对 x^5, x^4, \dots, x^0 码元位依照同样方法校验，计算相应的 A_1, A_2, A_3 中取 1 数目的多少，可对它们逐位纠错。像这种根据正交方程中取 1 数目的多少，进行译码的方法称为**大数逻辑译码**，能用这种方法译码的码称为**大数逻辑可译码**。

上例中的[7, 3, 4]码，它的正交一致校验和式(6.3.1)式仅对 $e_6(x^6)$ 码元位正交，因此，只要用一次大数逻辑判决就能完成该位的译码，这种码称为**一步大数逻辑可译码**。

下面讨论正交校验和式的个数 J 与码纠错能力之间的关系。

定理 6.3.1 一个循环码若在任一位上能建立 J 个正交一致校验和式，则该码能纠正 $t \leq \lfloor J/2 \rfloor$ 个错误(式中， $\lfloor x \rfloor$ 表示取 x 的整数部分)。

证明

(1) 若码组中产生了 $t \leq \lfloor J/2 \rfloor$ 个错误，其中之一发生在正交码元位上，其余的 $t-1 \leq \lfloor J/2 \rfloor - 1$ 个错误在其它位上，则至多有 $\lfloor J/2 \rfloor - 1$ 个正交校验和式为 0(因为此时取 0 的正交方程中有两个错误，一个在正交位上，另一个在其它位上)，至少有 $J - (\lfloor J/2 \rfloor - 1) = \lfloor J/2 \rfloor + 1$ 个 1，所以根据 1 的数目多于一半的事实，对正交位纠错。

(2) 若 $t = \lfloor J/2 \rfloor$ 个错误均不在正交码元位上，最坏情况下均匀分布在每一方程中，则只能使 $\lfloor J/2 \rfloor$ 个方程取值 1，1 的个数没有超过一半，不会对正交码元位进行纠错。

综上所述，有 J 个正交校验和式的码能纠正 $t \leq \lfloor J/2 \rfloor$ 个错误。 ■

一步大数逻辑可译码的纠错个数 t ，与码参数之间有如下关系⁽²⁾：

$$t \leq \frac{n-1}{2(d_{ev}-1)} \quad (6.3.3)$$

式中， n 是码长， d_{ev} 是该码对偶码的最小距离。

由上式可知，若希望一个码的纠错能力尽可能大，则希望它的对偶码的纠错能力尽可能小，但实际上有这种特性的码不多。例如[23, 12]Golay 码的对偶码有 $d_{ev}=8$ ，若用大数逻辑译码方法译此码则只能纠一个错误，但若用上节所讲的修正捕错方法译就可纠 3 个错误。当然，实际上也有一些码的纠错能力达到式(6.3.3)所给定的限，如例举的[7, 3, 4]码。今后称码的最小距离 $d=J+1$ 的码为**一步完备可正交码**。

[$2^m-1, 2^m-1-m$] 循环汉明码的对偶码是一个 [$2^m-1, m$] 码，有最小距离 $d=2^{m-1}$ ，称为**极长码**。可以验证它满足式(6.3.3)的限，并且是一步完备可正交码，能用一步大数逻辑译码方法纠正 $2^{m-2}-1$ 个错误。

一步大数逻辑可译码的译码方法虽然简单，但遗憾的是此类码不多，且纠错能力也不高。为了提高大数逻辑译码方法的应用范围和改善码的纠错能力，可把对某一码元位正交的概念推广到对某一码元位置集合正交，并通过逐次大数逻辑译码，最后完成对某一码元位的译码。为此先定义对某一码元位置集合正交的概念。

定义 6.3.2 如果某一码元位置集合 $\{c_{i1}, c_{i2}, \dots, c_{il}\}$ 的线性组合

$$a_{i1}c_{i1} + a_{i2}c_{i2} + \dots + a_{il}c_{il}$$

在 A_1, A_2, \dots, A_J 的一致校验和式中均出现，而其余码元位置集合至多在其中一个校验和式中出现，则说 A_1, A_2, \dots, A_J 在集合 $\{c_{i1}, c_{i2}, \dots, c_{il}\}$ 上正交，称 A_1, A_2, \dots, A_J 是正交于该码元位置集合的正交一致校验和式，上式中的 a_{ij} 是非0值。

例如[7, 4, 3]循环汉明码， $g(x) = x^3 + x^2 + 1$ 。它的校验矩阵

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

由 $\mathbf{S} = \mathbf{E} \cdot \mathbf{H}^T = (s_2, s_1, s_0)$ 得到：

$$\begin{cases} A_{11}^1 = s_2 = (e_6 + e_4) + e_3 + e_2 \\ A_{12}^1 = s_1 = (e_6 + e_4) + e_5 + e_1 \end{cases} \quad (6.3.4)$$

$$\begin{cases} A_{21}^1 = s_1 = (e_6 + e_5) + e_4 + e_1 \\ A_{22}^1 = s_2 + s_0 = (e_6 + e_5) + e_2 + e_0 \end{cases} \quad (6.3.5)$$

组成了 $J=2$ 组对 $\{(e_6, e_4) = E_1^1\}$ 和对 $\{(e_6, e_5) = E_2^1\}$ 码元位置集合正交的正交一致校验和式。若码组中仅出现一个错误，则根据 A_{11}^1 和 A_{12}^1 中取值是1的个数多少，正确译得 $(e_6 + e_4) = A_1^1$ 的值。由 A_{21}^1 和 A_{22}^1 中取值是1的个数多少，正确译得 $(e_6 + e_5) = A_2^1$ 的值。然后再根据 A_1^1, A_2^1 中取值为1的个数，译出 x^6 码元位是否错误。

如 x^6 码元位发生了错误， $e_6 = 1$ 。由式(6.3.4)和式(6.3.5)有：

$$\begin{cases} A_{11}^1 = (e_6 + e_4) + e_3 + e_2 = 1 \\ A_{12}^1 = (e_6 + e_4) + e_5 + e_1 = 1 \end{cases}$$

$$\begin{cases} A_{21}^1 = (e_6 + e_5) + e_4 + e_1 = 1 \\ A_{22}^1 = (e_6 + e_5) + e_2 + e_0 = 1 \end{cases}$$

可得：

$$\begin{cases} A_1^1 = (e_6 + e_4) = 1 \\ A_2^1 = (e_6 + e_5) = 1 \end{cases}$$

此两方程正交于 e_6 ，可以根据 A_1^1 和 A_2^1 中取1个数的多少，决定出 e_6 的值。这里 $A_1^1 = A_2^1 = 1$ ，有两个1，所以 $e_6 = 1$ 。

从该例看出，为了译出一个码元需要进行两次大数判决。每次需要两个正交校验和式，第一次是对码元位置集合 $\{e_6, e_4\}$ 和 $\{e_6, e_5\}$ 正交，第二次是对两个码元的位置集中的 $e_6(x^6)$ 码元位正交，所以是两步大数逻辑译码。 $[7, 4, 3]$ 循环汉明码就称为**两步大数逻辑可译码**。

一般情况，若能在一个循环码中挑选出几个码元位置集合： E_1^1, E_2^1, \dots ，且对每个码元位置集合至少可构成 J 个正交于它的一致校验和式，如果码组中发生的错误个数 $\leq \lfloor J/2 \rfloor$ ，则可以根据大数逻辑译码方法，对 E_1^1, E_2^1, \dots 作出正确估值，得到 A_1^1, A_2^1, \dots 的值。其中对每一个 A_i^1 ，当且仅当大于 $\lfloor J/2 \rfloor$ 个正交校验和式的值为 1 时， A_i^1 的值为 1；否则取 0。得到 A_1^1, A_2^1, \dots 的值以后，再用它来估计在 E_1^1, E_2^1, \dots 中选出来的更小码元位置集合 $E_1^2 \subset E_1^1, E_2^2 \subset E_2^1, \dots$ 中的值。如果第二次所选的每个集合 E_1^2, E_2^2, \dots ，也至少能构成 J 个正交于它的一致校验和式，那么就可以用大数逻辑译码方法对 E_1^2, E_2^2, \dots 中的错误进行正确估计，得到估值 A_1^2, A_2^2, \dots 。这个过程一直进行下去，所选的码元位置集合越来越小，最后获得至少含有 J 个正交于一个码元位置 e_i 的正交校验和，从而根据大数逻辑方法最后译出 e_i 的值。由于码是循环码，可用这种方法确定每一位的错误值从而对每一码元位译码。像这种需要 L 步大数逻辑译码才能最后译出一个码元的方法，称为 L 步大数逻辑译码；能用 L 步大数逻辑译码的码称为 L 步大数逻辑可译码或 L 步可正交码；若 $J+1=d$ ，则称为 L 步完备可正交码。该例中 [7, 4, 3] 循环汉明码就是一个二步完备可正交码。

二、大数逻辑译码器

大数逻辑译码器分为两类：I型和II型，它们没有本质上的不同。下面先介绍一步大数逻辑译码器，再介绍 L 步大数逻辑译码器。

1. I型大数逻辑译码器 以[7, 3, 4]码的译码器为例，说明这种类型的一步大数逻辑译码器的构成和工作过程。

由式(6.3.1)和式(6.3.2)可知：

$$H_0 \cdot E^T = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \end{bmatrix} = \begin{bmatrix} s_3 \\ s_1 \\ s_2 + s_0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} e_6 \\ e_5 \\ e_4 \\ e_3 \\ e_2 \\ e_1 \\ e_0 \end{bmatrix}$$

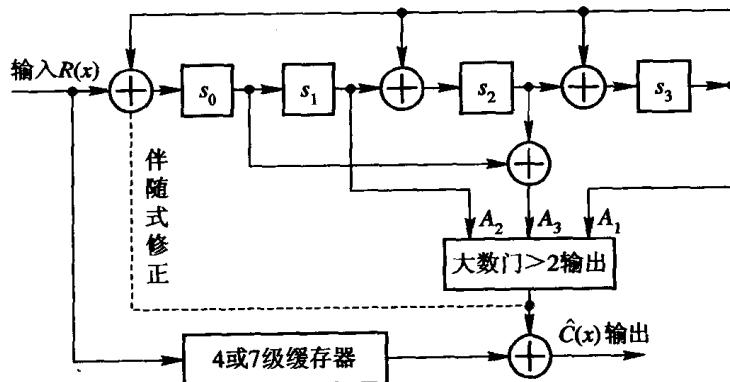


图 6-9 [7, 3, 4] 码 I 型大数逻辑译码器

由此构成的 I 型大数逻辑译码器如图 6-9。其工作过程如下：

(1) 由 $g(x) = x^4 + x^3 + x^2 + 1$ 的除法电路，计算 $R(x)$ 的伴随式 $S(x)$ 。 $n(7)$ 次移位后得到了伴随式 (s_0, s_1, s_2, s_3) ，存在伴随式寄存器中，此时 $R(x)$ 也全部进入 7 级（若为系统码也用 4 级）缓存器中。

(2) 停止对译码器输入，并开始对 $r_{n-1} = r_6$ 进行检验，也就是检查 $A_1 = s_3$ 、 $A_2 = s_1$ 和 $A_3 = s_2 + s_0$ 中的 1 的个数。若 1 的个数是 3（一般情况下是大于等于 $\lfloor J/2 \rfloor + 1$ ），大数门（为 3 个输入端的与门）输出 1。此时缓存器移动一次输出 r_6 ，对它进行纠错。若 $A_1 + A_2 + A_3 < 3$ ，则大数门没有输出， r_6 直接送出译码器。

(3) 伴随式计算器循环移位一次对 r_5 进行检验，此时在伴随式寄存器中的内容是对应于 r_5 的计算结果。若大数门输出 1，则对 r_5 进行纠错，否则， r_5 直接送出译码器。

(4) 重复上述步骤直至 $n=7$ 次为止。

(5) 第 $n(7)$ 次移位完毕后，若伴随式寄存器中的存数全为 0，则说明 $R(x)$ 中的错误是可以纠正的；否则说明是不可纠正的。若不可纠正，译码器送出一信号至用户，表示收到的 $R(x)$ 中的错误不能纠。然后，重新清洗译码器到初始状态，准备接收第二个码组的译码。

图中的虚线是把大数门输出的“1”反馈到伴随式计算电路，以消除该错误码元对伴随式的影响。

图 6-10 画出了 $[n, k, d=J+1]$ 码的 I 型一步大数逻辑译码器的一般组成。

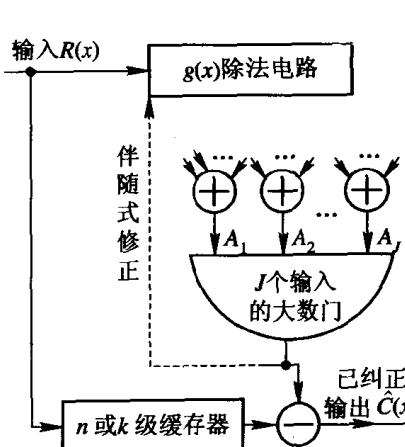


图 6-10 $[n, k, d=J+1]$ 码的 I 型大数逻辑译码器

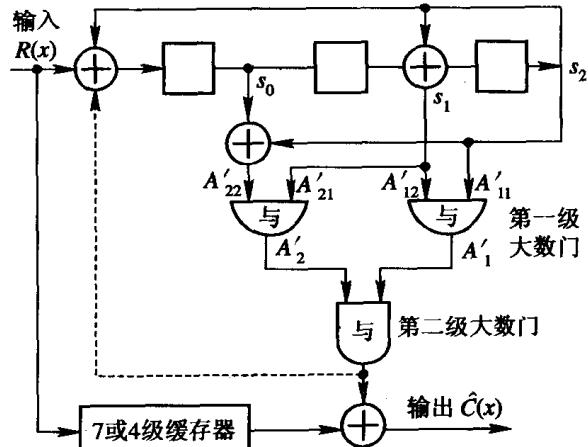


图 6-11 $[7, 4, 3]$ 码 I 型二步大数逻辑译码器

下面介绍这种类型的二步大数逻辑译码器，以 $[7, 4, 3]$ 循环汉明码为例说明。它的 I 型二步大数逻辑译码器画于图 6-11 中，其工作过程与一步大数逻辑译码器基本相似，这里不再重复。

由图 6-9 至 6-11 可发现，I 型大数逻辑译码器的正交校验和式 $A_i (i=1, 2, \dots, J)$ ，是由伴随式分量 $s_0, s_1, \dots, s_{n-k-1}$ 的线性组合得到的。

2. I 型大数逻辑译码器 仍以 $[7, 4, 3]$ 码为例说明这类大数逻辑译码器的工作原理。由式(6.3.1)得：

$$S = H_0 E^T = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} e_6 \\ e_5 \\ e_4 \\ e_3 \\ e_2 \\ e_1 \\ e_0 \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \end{bmatrix}$$

$$A_1 = s_3 = e_6 + e_4 + e_3$$

$$A_2 = s_1 = e_6 + e_5 + e_1$$

$$A_3 = s_2 + s_0 = e_6 + e_2 + e_0$$

由此式得到该码的Ⅰ型大数逻辑译码器，如图 6-12 所示。这类译码器的工作过程与Ⅰ型相似，它们的区别仅在于当 $n=7$ 次移位完毕后， $R(x)$ 已全部进入到 7 级移存器中，此时开关 K 由 b 接至 a，对码元 r_6 纠错后， r_6 一方面送至用户，一方面反馈到 7 级移存器的第一级中，以便参加对 r_5 的检验。同理，其它码元在纠错以后也要反馈到移存器中，以便参加对下一位码元的检验。

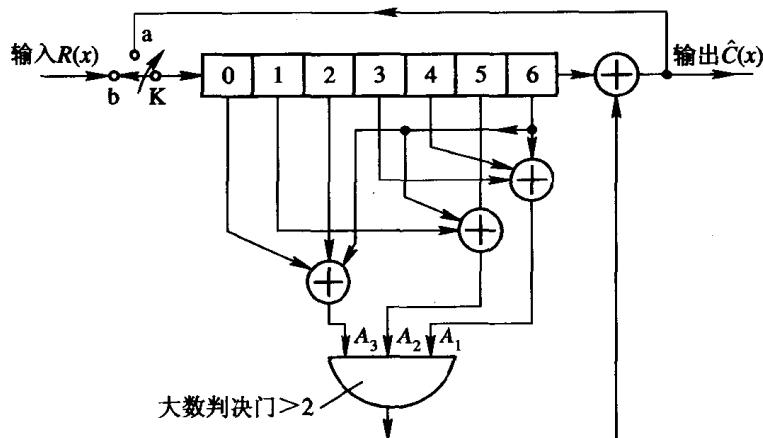


图 6-12 [7, 4, 3] 码Ⅰ型大数逻辑译码器

$[n, k, d=J+1]$ 循环码的Ⅰ型大数逻辑译码器的一般框图，如图 6-13 所示。

下面以 [7, 4, 3] 循环码为例，说明Ⅰ型二步大数逻辑译码器的组成。

由式(6.3.4)和式(6.3.5)有：

$$\begin{cases} A_{11}^1 = (e_6 + e_4) + e_3 + e_2 \\ A_{12}^1 = (e_6 + e_4) + e_5 + e_1 \\ A_{21}^1 = (e_6 + e_5) + e_4 + e_1 \\ A_{22}^1 = (e_6 + e_5) + e_2 + e_0 \end{cases}$$

可直接得到此码的Ⅰ型大数逻辑译码器，如图 6-14。它的工作步骤大致与一步的相同，这里不再赘述。

L 步Ⅰ型大数逻辑译码器的结构，类似于图 6-14 所示的二步结构形式。但是，无论是Ⅰ型还是Ⅱ型译码器，它们设备的复杂性都随 L 指数增加，因此在实际中很少用到 $L>2$ 步译码器。

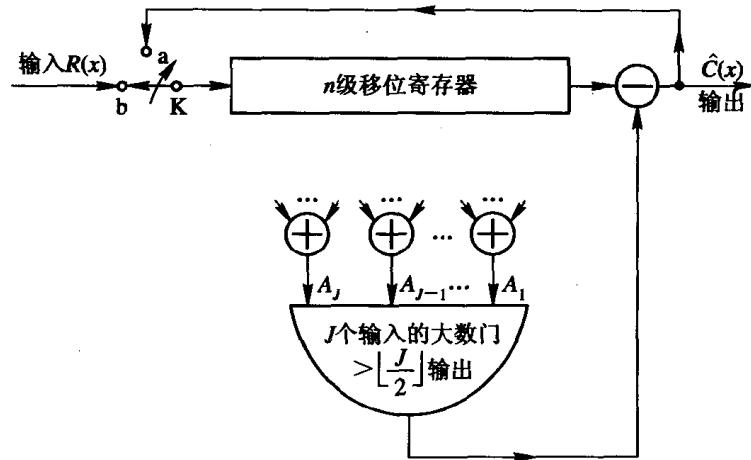


图 6-13 $[n, k, d=J+1]$ 码的 I 型大数逻辑译码器

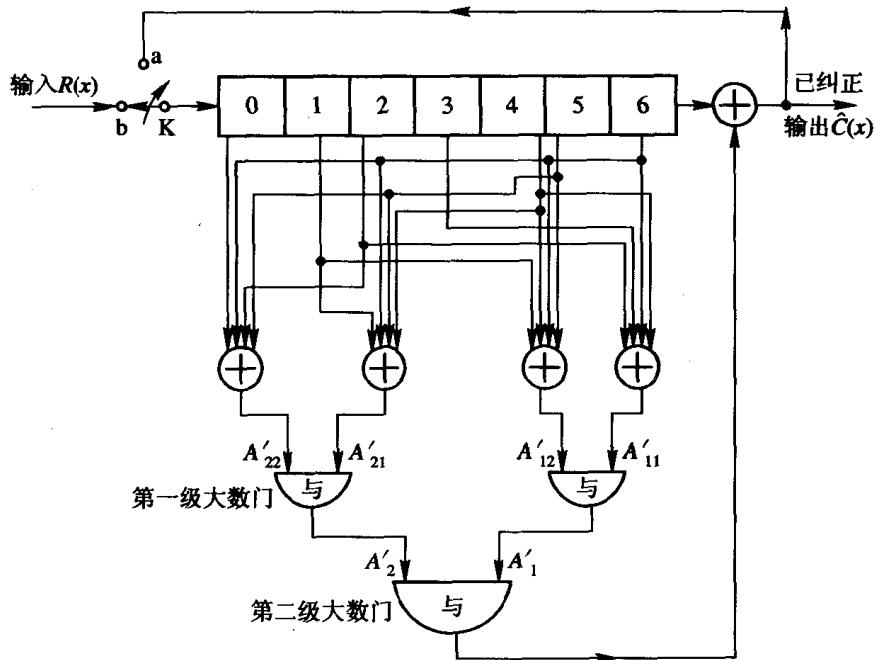


图 6-14 $[7, 4, 3]$ 码的 I 型二步大数逻辑译码器

§ 6.4 大数逻辑可译码的构造

大数逻辑可译码的构造涉及到欧氏几何、射影几何和组合数学中的一些问题，利用这些数学方法，能相应地构造出一大类欧氏几何(EG)码^[2](如 Reed - Muller 码等)、射影几何(PG)码^[2](如极长码和差集码等)和利用组合数学构造的码(如区组设计码和复数旋转码^[5]等)。由于篇幅所限，这里仅介绍可用一步大数逻辑译码的极长码、差集码和复数旋转码，而其它码类可参阅文献[2]，[5]。

一、极长码

极长码是一阶 PG 码，这里我们仅讨论 $p=2$ 的二进制码。

定义 6.4.1 对任何整数 $m \geq 2$ ，均存在有如下参数的极长码： $n = 2^m - 1$, $k = m$, $d = 2^{m-1}$ 。码的生成多项式

$$g(x) = \frac{x^n - 1}{p(x)}$$

式中， $p(x)$ 是一个 m 次本原多项式。

该码有一个全为 0 码字， $2^m - 1$ 个重量为 2^{m-1} 的码字，因而最小距离是 2^{m-1} ，这是一个等重码。由第五章可知，所有以 m 次本原多项式作为码的生成多项式 $g(x)$ ，则每一个都生成一个 $[2^m - 1, 2^m - 1 - m]$ 汉明码，因而极长码是该汉明码的对偶码。由于用极长码的校验多项式 $p^*(x)$ ($p(x)$ 的互反多项式) 作为电路的联接多项式时，电路所产生的序列就是一个 m 序列，它就是极长码的一个码字，因为 m 级线性移位寄存器，可能产生的最长序列就是 m 序列，故称该码为极长码。

下面我们讨论这类码的大数逻辑译码方法，为此必须首先确定这类码的正交校验和式。

由上面可知，该码的零空间是汉明码。考虑以下形式的汉明码码字集合：

$$Q = \{W(x) = x^i + x^j + x^{n-1}, \quad 0 \leq i < j < n-1\}$$

即 Q 中的所有码字，除在 x^{n-1} 位外，没有任何两个码字有相同项。因为若有相同项如：

$$\begin{aligned} W_1(x) &= x^i + x^j + x^{n-1} \\ W_2(x) &= x^i + x^k + x^{n-1} \end{aligned}$$

则

$$W_1(x) + W_2(x) = x^j + x^k$$

由于 $W_1(x) + W_2(x)$ 也是汉明码的一个码字，可是它的重量小于 3，这与汉明码的距离要求相矛盾，因而 $W_1(x)$ 与 $W_2(x)$ 有相同项的假设不能成立。因此， Q 中包含了用 $p^*(x)$ 生成的、且与 $g(x)$ 生成的码字正交于 x^{n-1} 的所有多项式。所以，可以由 Q 得到正交校验矩阵与正交校验和式。

下面讨论如何得到 Q 中的正交多项式。首先在 x^{n-1} 至 x^0 位中，由任何两项 x^{n-1} 和 x^i ，求 x^i ，使

$$\begin{aligned} x^{n-1} + x^i &\equiv 0 \pmod{p^*(x)} \\ x^{n-1} + x^i &= q_1(x)p^*(x) \\ x^{n-1} + x^i &= q_1(x)p^*(x) + x^i \end{aligned}$$

所以， x^i 就是 $x^{n-1} + x^i$ 被 $p^*(x)$ 除后，得到仅有的一项的余项。得到 x^i 后，再用 x^{n-1} 和 $x^k (k \neq i, j)$ 两项，同理得到

$$x^{n-1} + x^k \equiv q_2(x)p^*(x) + x^l$$

x^l 和 x^k 项组成 Q 中的另一个正交多项式 $x^{n-1} + x^k + x^l$ ，依此类推，一直进行下去，一共可找到 $(n-1)/2$ 个正交于 x^{n-1} 位的正交校验多项式。所以

$$J = \frac{n-1}{2} = \frac{2^m - 2}{2} = 2^{m-1} - 1$$

码的最小距离

$$d = J + 1 = 2^{m-1}$$

例 6.4 求 $m=4$, $p(x)=1+x+x^4$ 的极长码及其正交校验多项式。

码有如下参数: $n=2^m-1=15$, $k=4$, $d=8$, 能用大数逻辑译码纠正 3 个随机错误。现在确定正交校验多项式集合 Q 。 Q 中共有 $(n-1)/2=7$ 个正交于 x^{14} 的正交校验和式 $W_i(x)$ 。 $p^*(x) = x^4 + x^3 + 1$, 所以由

$$x^{14} + x^{13} = x^{10}(x^4 + x^3 + 1) + x^{10}$$

得到余式 x^{10} , 因而可得:

$$W_1(x) = x^{14} + x^{13} + x^{10}$$

同理可得:

$$W_2(x) = x^{14} + x^{12} + x^6$$

$$W_3(x) = x^{14} + x^{11} + 1$$

$$W_4(x) = x^{14} + x^9 + x^4$$

$$W_5(x) = x^{14} + x^8 + x$$

$$W_6(x) = x^{14} + x^7 + x^5$$

$$W_7(x) = x^{14} + x^3 + x^2$$

由此得到以下 7 个正交于 x^{14} 位的正交校验和式:

$$A_1 = e_{14} + e_{13} + e_{10} = s_{10}$$

$$A_2 = e_{14} + e_{12} + e_6 = s_6$$

$$A_3 = e_{14} + e_{11} + e_0 = s_0$$

$$A_4 = e_{14} + e_9 + e_4 = s_4 + s_9$$

$$A_5 = e_{14} + e_8 + e_1 = s_1 + s_8$$

$$A_6 = e_{14} + e_7 + e_5 = s_5 + s_7$$

$$A_7 = e_{14} + e_3 + e_2 = s_2 + s_3$$

上式中伴随式分量的系数, 可根据 § 6.3 所述的 I 型与 II 型大数逻辑译码器之间的关系得到, 即 s_i 前的系数 a_i 与 e_i 前的系数满足以下关系:

$$a_i = e_i \quad i = 0, 1, \dots, n-k-1 = 0, 1, \dots, 10$$

由上面 A_1, A_2, \dots, A_7 方程可得到该码的 I 型和 II 型大数逻辑译码器。 ■

二、差集循环码

差集循环码是一阶 PG 码。但它最原始的构造方法是从差集角度出发的, 为此我们首先定义差集的概念。

定义 6.4.2 令 p 为素数, s 是任意正整数, $n = p^{2s} + p^s + 1$ 。在整数集合 $\{0, 1, 2, \dots, p^s(p^s + 1)\}$ 中, 选择 $p^s + 1$ 个整数组成集合 D : $\{0 \leq l_1 < l_2 < l_3, \dots, < l_{p^s}\}$, 由此整数集合构成 $p^s(p^s + 1)$ 个有序差:

$$D = \{l_j - l_i, i \neq j\}$$

显然，在 D 中差的一半是正数，一半是负数，称 D 为 p^s 阶单纯完备差集。当且仅当满足以下性质：

- (1) D 中所有正差均不同；
- (2) D 中所有负差均不同；
- (3) 若 $l_j - l_i$ 是 D 中的负差，则 $p^s(p^s + 1) - (l_j - l_i) + 1$ 不等于 D 中任何正差。

由该定义可知， D 中所有元素正差和负差构成了1到 $p^s(p^s + 1)$ 的任何整数。

例 6.5 $p=2, s=2, p^s=4, p^s(p^s+1)=20$ ，则 $n=p^s(p^s+1)+1=21$ ，在 $\{0, 1, \dots, 21\}$ 个整数中，若我们选取 $p^s+1=2^2+1=5$ 个数组成

$$D: \{0, 2, 7, 8, 11\}$$

则有：

$$\begin{aligned} 2 - 0 &= 2, \quad 0 - 2 = -2 \equiv 19 \pmod{21} \\ 7 - 0 &= 7, \quad 0 - 7 = -7 \equiv 14 \pmod{21} \\ 8 - 0 &= 8, \quad 0 - 8 = -8 \equiv 13 \pmod{21} \\ 11 - 0 &= 11, \quad 0 - 11 = -11 \equiv 10 \pmod{21} \\ 7 - 2 &= 5, \quad 2 - 7 = -5 \equiv 16 \pmod{21} \\ 8 - 2 &= 6, \quad 2 - 8 = -6 \equiv 15 \pmod{21} \\ 11 - 2 &= 9, \quad 2 - 11 = -9 \equiv 12 \pmod{21} \\ 8 - 7 &= 1, \quad 7 - 8 = -1 \equiv 20 \pmod{21} \\ 11 - 7 &= 4, \quad 7 - 11 = -4 \equiv 17 \pmod{21} \\ 11 - 8 &= 3, \quad 8 - 11 = -3 \equiv 18 \pmod{21} \end{aligned}$$

因而 D 是一个 $2^s=4$ 阶单纯完备差集。 ■

下面讨论用差集概念构造循环码的方法。仅讨论 $p=2$ 的情况，即 2^s 阶单纯差集所构成的码。

定义 6.4.3 令 $p=\{l_0=0, l_1, \dots, l_{2^s}\}$ 是一个 2^s 阶单纯完备差集，并定义下列多项式：

$$Z(x) = x^0 + x^{l_1} + x^{l_2} + \dots + x^{l_{2^s}} = 1 + x^{l_1} + \dots + x^{l_{2^s}}$$

且 $n = p^{2^s} + p^s + 1 = 2^{2^s} + 2^s + 1 = 2^s(2^s + 1) + 1$

$$h(x) = \text{GCD}(Z(x), x^n + 1) = 1 + h_1x + h_2x^2 + \dots + h_{k-1}x^{k-1} + x^k$$

则长为 n 的差集循环码由下列 $g(x)$ 生成：

$$\begin{aligned} g(x) &= \frac{x^n - 1}{h(x)} \\ &= 1 + g_1x + g_2x^2 + \dots + x^{n-k} \end{aligned}$$

该码有如下参数：

$$n = 2^{2^s} + 2^s + 1$$

$$n - k = \binom{p + 1}{2}^s + 1 = 3^s + 1$$

$$d = 2^s + 2$$

下面讨论该码的正交校验多项式 $\{W(x)\}$ 是如何得到的。由正交校验多项式的性质可知， $W(x)$ 必在由 $g(x)$ 生成的码的零空间中，并对某一码元位如 x^{n-1} 正交。为此：

(1) 先求 $h(x)$ 的互反多项式 $h^*(x)$, $\partial \circ h^*(x) = k$, 则由此 $h^*(x)$ 作生成多项式生成的 $[n, n-k]$ 码一定在 $g(x)$ 生成码的零空间中。

(2) 令 $Z^*(x) = x^{l_2} Z(x^{-1}) = x^{l_2} + x^{l_2 - l_1} + \dots + 1$ 是 $Z(x)$ 的互反多项式。由于 $h(x) = \text{GCD}(Z(x), x^n - 1)$, 因而 $h(x) | Z(x)$, 所以 $h^*(x) | Z^*(x)$ 。因此, $Z^*(x)$ 也必在 $g(x)$ 生成的码的零空间中。

(3) 令

$$\begin{aligned} W(x) &= x^{n-1-l_2} Z^*(x) \\ &= x^{n-1-l_2} + x^{n-1-l_2+l_1} + \dots + x^{n-1-l_2} \\ &= x^{n-1} + x^{n-1-l_1} + \dots + x^{n-1-l_2} \end{aligned}$$

由 $h^*(x) | Z^*(x)$ 可知, $h^*(x) | W(x)$ 。所以 $W(x)$ 也在由 $g(x)$ 生成的码的零空间中。

(4) 令

$$\begin{aligned} W_i(x) &= x^{l_i} W(x) \\ &= x^{l_i} (x^{n-1} + x^{n-1-l_1} + \dots + x^{n-1-l_2}) \\ &= x^{l_i} (x^{-1} + x^{-1-l_1} + \dots + x^{-1-l_2}) \\ &= x^{l_i-1} + x^{l_i-l_1-1} + \dots + x^{l_i-l_{i-1}-1} + x^{n-1} + x^{l_i-l_{i+1}-1} + \dots + x^{l_i-l_2-1} \end{aligned}$$

显而易见, $W_i(x)$ 是 $W(x)$ 左移循环移位 l_i 次后的关系式。因为 $l_i \in D = \{0, l_1, \dots, l_2\}$, D 是一个单纯完备差集, 所以, $W_i(x) = x^{l_i} W(x)$ 与 $W_j(x) = x^{l_j} W(x)$ ($i \neq j$), 除在 x^{n-1} 项之外, 其它项都互不相同。因此, $W(x), W_1(x), W_2(x), \dots, W_{l_2}(x)$ 构成了正交于 x^{n-1} 码元位的 $J=2^s+1$ 个正交校验和式, 因而差集循环码的 $d=J+1=2^s+2$ 。

例 6.6 以例 6.5 的差集为例。 $p=2, s=2, n=2^{2s}+2^s+1=21, n-k=3^2+1=10, k=11, d=2^s+2=6$, 得到一个[21, 11]差集循环码。

$$D = \{0, 2, 7, 8, 11\}, l_2 = 11$$

$$Z(x) = 1 + x^2 + x^7 + x^8 + x^{11}$$

$$h(x) = \text{GCD}\{Z(x), x^{21} + 1\} = 1 + x^2 + x^7 + x^8 + x^{11}$$

码的生成多项式

$$g(x) = \frac{x^{21} - 1}{h(x)} = 1 + x^2 + x^4 + x^6 + x^7 + x^{10}$$

现在根据上面所设的方法求正交校验多项式。

(1) 求 $h(x)$ 的互反多项式 $h^*(x)$

$$h^*(x) = x^{11} h(x^{-1}) = 1 + x^3 + x^4 + x^9 + x^{11}$$

(2) 求 $Z(x)$ 的互反多项式 $Z^*(x)$

$$Z^*(x) = x^{11} Z(x^{-1}) = x^{11} + x^9 + x^4 + x^3 + 1$$

(3) 求正交校验和式 $W(x)$

$$\begin{aligned} W(x) &= x^{n-1-l_2} Z^*(x) = x^{21-1-11} (x^{11} + x^9 + x^4 + x^3 + 1) \\ &= x^{20} + x^{18} + x^{13} + x^{12} + x^9 \end{aligned}$$

(4) 求 $W_l(x), l=2, 7, 8, 11$

$$\begin{aligned} W_1(x) &= W_{l_2}(x) = x^2 (x^{20} + x^{18} + x^{13} + x^{12} + x^9) \\ &= x^{20} + x^{15} + x^{14} + x^{11} + x \end{aligned}$$

$$\begin{aligned}
W_2(x) &= W_{l_7}(x) = x^7(x^{20} + x^{18} + x^{13} + x^{12} + x^9) \\
&= x^{20} + x^{19} + x^{16} + x^6 + x^4 \\
W_3(x) &= W_{l_8}(x) = x^{20} + x^{17} + x^7 + x^5 + x^0 \\
W_4(x) &= W_{l_{11}}(x) = x^{20} + x^{10} + x^8 + x^3 + x^2
\end{aligned}$$

共得到了 $W(x)$, $W_1(x)$, ..., $W_4(x)$ 等 $J=2^2+1=5$ 个正交于 x^{20} 码元位的正交一致校验和式。由上可得以下的正交校验和式：

$$\begin{aligned}
A_1 &= e_{20} + e_{18} + e_{13} + e_{12} + e_9 = s_9 \\
A_2 &= e_{20} + e_{15} + e_{14} + e_{11} + e_1 = s_1 \\
A_3 &= e_{20} + e_{19} + e_{16} + e_6 + e_4 = s_6 + s_4 \\
A_4 &= e_{20} + e_{17} + e_7 + e_5 + e_0 = s_7 + s_5 + s_0 \\
A_5 &= e_{20} + e_{10} + e_8 + e_3 + e_2 = s_8 + s_3 + s_2
\end{aligned}$$

由此可得到 [21, 11] 差集循环码的 I 型和 II 型大数逻辑译码器。

为了实际使用方便，表 6-4 中列出了某些用差集方法构造的一步大数逻辑可译码^[2]。对高次码元位 (x^{n-1}) 正交的 $J=d-1$ 个正交检验和式，可以从表中“有关的差集”这一栏中得到。如 [7, 3, 4] 码，查表得到的“有关的差集”是 {0, 2, 3}，表示正交校验多项式

$$h_1(x) = x^6(x^{-3} + x^{-2} + x^0) = x^6 + x^4 + x^3$$

表 6-4 某些一步大数逻辑可译码

| s | n | k | d | t | 生成多项式 $g(x)$ | 有关的差集 |
|-----|------|-----|-----|-----|--|--|
| 1 | 7 | 3 | 4 | 1 | 0, 2, 3, 4 | 0, 2, 3 |
| 2 | 21 | 11 | 6 | 2 | 0, 2, 4, 6, 7, 10 | 0, 2, 7, 8, 11 |
| 3 | 73 | 45 | 10 | 4 | 0, 2, 4, 6, 8, 12, 16, 22, 25, 28 | 0, 2, 10, 24, 25, 29, 36, 42, 45 |
| 4 | 273 | 191 | 18 | 8 | 0, 4, 10, 18, 22, 24, 34, 36, 40, 48, 52, 56, 66, 67, 71, 76, 77, 82 | 0, 18, 24, 46, 50, 67, 103, 112, 115, 126, 128, 159, 166, 167, 186, 196, 201 |
| 5 | 1057 | 813 | 34 | 16 | 0, 1, 3, 4, 5, 11, 14, 17, 18, 22, 23, 26, 27, 28, 32, 33, 35, 37, 39, 41, 43, 45, 47, 48, 51, 52, 55, 59, 62, 68, 70, 71, 72, 74, 75, 76, 79, 81, 83, 88, 95, 96, 98, 101, 103, 105, 106, 108, 111, 114, 115, 116, 120, 121, 122, 123, 124, 126, 129, 131, 132, 135, 137, 138, 141, 142, 146, 147, 149, 150, 151, 153, 154, 155, 158, 160, 161, 164, 165, 166, 167, 169, 174, 175, 176, 177, 178, 179, 180, 182, 183, 184, 186, 188, 189, 191, 193, 194, 195, 198, 199, 200, 201, 202, 203, 208, 209, 210, 211, 212, 214, 216, 222, 224, 226, 228, 232, 234, 236, 242, 244 | 0, 1, 3, 7, 15, 31, 54, 63, 109, 127, 138, 219, 255, 277, 298, 338, 348, 439, 452, 511, 528, 555, 597, 677, 697, 702, 754, 792, 879, 905, 924, 990, 1023 |

注意：每个生成多项式是用其非零项的指数代表的。例如 {0, 2, 3, 4} 代表 $g(x) = x^4 + x^3 + x^2 + 1$ 。

其余两个正交校验多项式是 $h_1(x)$ 的平方或循环移位，如第二个正交校验多项式

$$\begin{aligned} h_2(x) &= x^2(x^6 + x^4 + x^3) = x^8 + x^6 + x^5 \\ &\equiv x^6 + x^5 + x \pmod{x^7 - 1} \end{aligned}$$

是 $h_1(x)$ 的循环移位二次(乘 x^2)，而第三个正交校验多项式

$$\begin{aligned} h_3(x) &= x^3h_1(x) = xh_2(x) \\ &= x(x^6 + x^5 + x) \equiv x^6 + x^2 + 1 \pmod{x^7 - 1} \end{aligned}$$

是 $h_1(x)$ 的循环移位 3 次。

该表中列出的码都是一步完备可正交码，因此可从“有关的差集”这一列内，通过上面介绍的 $h_1(x)$ 平方和循环移位，得到 $J=d-1$ 个正交校验和式。

三、复数旋转码

复数旋转码是利用区组设计构造的一类一步大数逻辑可译码。有以下参数：

| | |
|---------|-----------------------------|
| 码 长 | $n = p^2 + p(d - 1)$ |
| 信 息 位 | $k = p^2$ |
| 最 小 距 离 | $d = q + 1 \leqslant p + 2$ |
| 码 率 | $R = k/n = p/(p + d - 1)$ |

这里， p 是奇素数。

该码的编码规则如下。把 $k=p^2$ 个信息元排成下列的 $p \times p$ 阶方阵：

$$M_p = \begin{bmatrix} m_{0,0} & \cdots & m_{0,p-1} \\ \vdots & & \vdots \\ m_{p-1,0} & \cdots & m_{p-1,p-1} \end{bmatrix} \quad (6.4.1)$$

然后在 M_p 阵后面附上一个 $p \times q$ 阶校验元矩阵

$$N_{p,q} = \begin{bmatrix} n_{0,0} & \cdots & n_{0,q-1} \\ \vdots & & \vdots \\ n_{p-1,0} & \cdots & n_{p-1,q-1} \end{bmatrix} \quad (6.4.2)$$

式中， $0 \leqslant q \leqslant p+1$ ，根据对 d 的要求而定。上式中的各校验元由下式求得：

$$\begin{aligned} n_{i,j} &= \sum_{k=0}^{p-1} m_{r,k} \\ r &\equiv i + jk \pmod{p} \quad 0 \leqslant j \leqslant p-1 \end{aligned} \quad (6.4.3)$$

如果 $N_{p,q}$ 中最后一列的 $j=q$ ，则其校验位按

$$n_{i,q} = \sum_{s=0}^{p-1} m_{s,i} \quad 0 \leqslant i \leqslant p-1 \quad (6.4.4)$$

求出，也就是 M_p 矩阵的各列的奇偶校验。上面两式中的 \sum 为模 2 和。

例如， $p=3, k=3^2=9, d=5$ ，则 $n=p^2+p(d-1)=21$ ，得到一个[21, 9, 5]码，它的

$$M_3 = \begin{bmatrix} m_{00} & m_{01} & m_{02} \\ m_{10} & m_{11} & m_{12} \\ m_{20} & m_{21} & m_{22} \end{bmatrix}$$

$$N_{3,3} = \begin{bmatrix} n_{00} & n_{01} & n_{02} & n_{03} \\ n_{10} & n_{11} & n_{12} & n_{13} \\ n_{20} & n_{21} & n_{22} & n_{23} \end{bmatrix}$$

根据式(6.4.3)和式(6.4.4)可得 $N_{3,3}$ 中的各元素为:

$$\begin{array}{ll} n_{00} = m_{00} + m_{01} + m_{02} & n_{01} = m_{00} + m_{11} + m_{22} \\ n_{10} = m_{10} + m_{11} + m_{12} & n_{11} = m_{10} + m_{21} + m_{02} \\ n_{20} = m_{20} + m_{21} + m_{22} & n_{21} = m_{20} + m_{01} + m_{12} \\ n_{02} = m_{00} + m_{21} + m_{12} & n_{03} = m_{00} + m_{10} + m_{20} \\ n_{12} = m_{10} + m_{01} + m_{22} & n_{13} = m_{01} + m_{11} + m_{21} \\ n_{22} = m_{20} + m_{11} + m_{02} & n_{23} = m_{02} + m_{12} + m_{22} \end{array}$$

由此可得每个信息元的 4 个正交校验和式。如 m_{00} 的 4 个正交校验和式:

$$\begin{aligned} m_{00} + m_{01} + m_{02} + n_{00} &= 0 \\ m_{00} + m_{11} + m_{22} + n_{01} &= 0 \\ m_{00} + m_{12} + m_{21} + n_{02} &= 0 \\ m_{00} + m_{10} + m_{20} + n_{03} &= 0 \end{aligned} \quad (6.4.5)$$

设信息元为(101011100), 则由 M_3 和 $N_{3,3}$ 组成的矩阵如下:

$$[M_3 \ N_{3,3}] = \left[\begin{array}{ccc|cccc} 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{array} \right] \quad (6.4.6)$$

由式(6.4.3)和式(6.4.4)不难看出, 其编码有以下规律:

- (1) 每一个校验元与 p 个信息元有关, 而每一个信息元被 $p+1$ 个校验元监督, 每一列的校验元正好把所有信息元校验一次。
- (2) 每一个信息元的变化, 都会使具有 q 列的校验元矩阵中有 q 个校验元随之变化, 也就是最小码距 $d=q+1$ 。特别当取 $q=2t$ 时, $d=2t+1$ 。
- (3) 当 $q=p+1$ 时, 每两个信息元正好有一次机会被同一校验元所校验, 从而使纠错能力达到最大, 为 $t_m=(p+1)/2$ 。

传输时可以按矩阵 $[M_p \ N_{p,q}]$ 的列的次序, 也可以按行的次序传送。收端收到码元后, 按 $[M_p \ N_{p,q}]$ 的规则重新构造, 然后按照式(6.4.3)和式(6.4.4)的规则进行大数逻辑译码。由此两式或式(6.4.5)可以看到, 这类码并非循环码, 但并不会妨碍其译码的简单性。特别当用软件实现这类码的编译码时, 软件的编译更为简单, 译码速度可以做得很快。

此外对这类码稍加改进, 还可以构造出 LUEP 码, 有关这类码以及复数旋转码的详细构造与分析可参阅文献[5], [6]。

§ 6.5 软判决译码的基本原理

上几节所介绍的循环码的各种译码方法都是硬判决译码, 即解调器供给译码器作为译码用的每个码元只取 0 或 1 两个值(二进制情况, 若非特别说明, 下面的讨论仅局限于二进制码)。在调相和相干解调(DPSK)系统中, 若 0、1 信号的电压 $\pm \sqrt{E_s}$, 则相当于匹配滤波器输出端的判决门限取 0。若接收电压的幅度(或抽样电压幅度)小于 0, 则解调器输出为 0, 若大于或等于 0, 则输出为 1。这种判决结果当然会损失掉接收信号中所包含的有用信息。为了充分利用接收信号波形中的信息, 使译码器能以更大的正确概率判决所发的

码字，就把解调器输出的抽样电压进行分层或量化。因而由解调器输出供给译码器的值就不止两个，而有 Q (通常 $Q = 2^m$)个。另一方面若译码器直接利用解调器输出的未量化的模拟电压或其变换进行译码，称为**模拟译码**。无论译码器利用 Q 进制序列译码，还是利用模拟量的模拟译码，统称为**软判决译码**(或简称**软译码**)。而这时的编码信道称为二进制输入 Q 进制输出的离散信道，一般若信道中的噪声仅为白高斯噪声，则称为**离散无记忆信道**(DMC)，相应于模拟译码的信道则称为**模拟信道**。因此，软判决译码又为数字通信系统设计工程师提供了一个设计自由度。

通常，译码器利用附加的软判决信息进行软译码时比硬译码能得到额外的 2~3 dB 的增益。一般，若利用 $Q=8$ 或 $Q=16$ 电平量化，所得到的性能约与最大似然译码的性能差不多，但其译码器的复杂度却比最大似然译码要低得多。因此，随着数字通信质量的日益提高，软判决译码将越来越得到普遍应用。

由于软判决译码要利用每个码元的可信信息，因此软译码器比硬译码器要复杂得多。一般情况下它较适合于中等码长和中等纠错能力的码，并且更适应于信噪比在很宽范围内变化的信道，如高频和散射信道。

众所周知，应用纠错码进行差错控制时，对信道的质量有一定要求。信道质量越好，则利用纠错码所获得的效果就越明显；反之，若信道原始误码率很高如为 10^{-2} ，则利用纠错码后不但没有好处，有时反而变得更坏。但软判决译码不同，在误码率为 $10^{-2} \sim 10^{-3}$ 范围内，能提供几乎最佳的性能，这正与一般的硬判决译码器相反。

在软判决译码中有两个最佳译码准则：一是使**码字错误概率最低**的软判决译码，这种方法，是以接收序列与发送码字的“软判决距离”为基础进行的；另一准则是使**码元错误概率最低**的软判决译码，这种方法是以每个码元似然函数比为基础进行的。与这两个准则相联系的性能，实质上是一致的。因为最小码字错误率通常也使码元错误概率最小，反之亦然。提出这两种准则的一个重要意义，在于使我们可以从不同角度进行软译码。

使符号(码元)错误概率最小的软判决译码方法中最重要的有：1963年由梅西(Massy)在他的“Threshold decoding(门限译码)”一书中提出的 APP 译码(后验概率译码)算法^[7]；哈特曼(Hartman)和鲁道夫(Rudolph)于 1976 年提出的逐位译码算法(简称 HR 算法)^[8]；以及 1971 年由韦尔登(Weldon)提出的重量删除译码算法(WED 算法)^[9]等。

最早使码字错误概率最小的软判决译码算法，是 1966 年由福尼(Forney)提出的广义**最小距离译码**(GMD 译码)^[10]，它其实就是硬译码中最小汉明距离译码和删除译码的推广。GMD 译码与稍后由契斯(Chase)于 1972 年提出的 Chase 算法^[11]，在本质上完全相同，其实就是 GMD 译码的一种实用算法。

本节介绍与软判决译码直接有关的信道量化中的几个问题，在下一节分别介绍比较实用的 GMD 与 Chase 算法。至于使码元错误概率最小的译码算法如 APP、WED 等^[32]，以及多进制码(如 RS 码)的软判决译码算法^[33]，可参阅有关文献。

一、软判决译码的基本概念

设在数传系统中分别用反相信号(BPSK)

$$s_1(t) = \sqrt{2E_s/T} \sin(2\pi f_0 t + \pi/2)$$

$$s_0(t) = \sqrt{2E_s/T} \sin(2\pi f_0 t - \pi/2)$$

传送 0、1 两个码元，式中 E_s 为信号能量， $T=1/f_0$ ，信号通过均值为 0、方差为 $\sigma^2=N_0/2$ 的加性高斯白噪声(AWGN)信道后到达接收端解调器的输入端。利用相干解调，则解调器匹配滤波器输出的抽样电压是一均值为 $\pm\sqrt{E_s}$ 、方差 $\sigma^2=N_0/2$ 的高斯随机变量。所以，输出电压 \tilde{r}_i 的概率密度函数为：

$$P(\tilde{r}_i|1) = \frac{\exp[-(\tilde{r}_i - \sqrt{E_s})^2/N_0]}{\sqrt{\pi N_0}} \quad (6.5.1)$$

$$P(\tilde{r}_i|0) = \frac{\exp[-(\tilde{r}_i + \sqrt{E_s})^2/N_0]}{\sqrt{\pi N_0}} \quad (6.5.2)$$

在硬判决解调器中，最佳判决门限为 0。当匹配滤波器输出的电压为负，则输出为 -1(相当于 0 或 1 码元)；若为正，则输出为 +1(相当于 1 或 0 码元)。很容易证明⁽¹¹⁾，此时该系统的误码率

$$p_e = Q[(2E_s/N_0)^{1/2}] \quad (6.5.3)$$

式中

$$Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{-x^2/2} dx \quad (6.5.4)$$

在数传系统中与硬判决解调器相应的译码器是一个只纠错的硬判决译码器。如果解调器输出有 3 个值 ± 1 和 0(相当于 0, 1, x)，则与此相应的译码器是一个纠错纠删译码器。若解调器的输出是一个介于 ± 1 之间的实数序列 \tilde{R} ，则与此相应的译码器是一个模拟译码器或软判决译码器。

在硬判决译码器和纠错纠删译码器中，通常利用汉明距离 d_h 来度量码的纠错能力。在软判决译码中，与软判决译码器相应的解调器的输出是 $Q(=2^n)$ 进制值，或介于 ± 1 之间的实数值。所以，与此相应的必须定义两个 n 长序列或码字之间的软判决距离，或其它适合于软判决译码的距离函数。

下面我们首先介绍几种有关的译码方法及距离函数，然后介绍信道量化。

第一章中介绍了最大似然译码、最大后验概率译码等译码错误概率最小的最佳译码方法，下面将介绍其它几类最佳译码的方法。

设 C 是 $[n, k]$ 二进制线性分组码， $\tilde{C}_i = (\tilde{c}_{i1}, \dots, \tilde{c}_{in}) \in \tilde{C}$ ，是被传输码字 $C_i = (c_{i1}, \dots, c_{in}) \in C$ 的传输映射，这里 $\tilde{c}_{il} = (-1)^{c_{il}}$ ， $l=1, 2, \dots, n$ 。因此 \tilde{C} 在由 $+1, -1$ 组成的 n 维线性空间的一个 k 维子空间中，因此群 $\{C, +\}$ 和群 $\{\tilde{C}, \times\}$ 之间建立了同构对应。这里“+”表示各分量之间按模 2 和运算，“ \times ”表示各分量之间按模 2 乘，即

$$\tilde{C}_1 \cdot \tilde{C}_2 = (\tilde{c}_{11}\tilde{c}_{21}, \dots, \tilde{c}_{1n}\tilde{c}_{2n})$$

当码字在均值为 0、方差为 $\sigma^2=N_0/2$ 的白高斯噪声信道中传输时，它可能被干扰。设从解调器送至译码器的 n 长实数序列为 \tilde{R} ，则由于噪声对每一码元(或符号)的影响是独立的，因而发送 $\tilde{C}_i = (\tilde{c}_{i1}, \dots, \tilde{c}_{in})$ ，接收为 $\tilde{R} = (\tilde{r}_1, \dots, \tilde{r}_n)$ 的概率 $P(\tilde{R}|\tilde{C}_i)$ 是 n 个高斯密度函数之积

$$P(\mathbf{R}|\tilde{\mathbf{C}}_l) = \prod_{i=1}^n [(1/\sqrt{\pi N_0}) e^{-(\tilde{r}_i - \tilde{c}_{li})^2/N_0}] = [1/\sqrt{\pi N_0}]^n e^{-\sum_{i=1}^n (\tilde{r}_i - \tilde{c}_{li})^2/N_0}$$

由最大似然译码可知，译码器选择一个有最大似然函数 $P(\mathbf{R}|\tilde{\mathbf{C}}_l)$ 的码字 $\tilde{\mathbf{C}}_l$ 作为输出。从上式看这等效于

$$\max P(\mathbf{R}|\tilde{\mathbf{C}}_l) \Rightarrow \min \sum_{i=1}^n (\tilde{r}_i - \tilde{c}_{li})^2 \quad l = 1, \dots, 2^k \quad (6.5.5)$$

定义

$$d_E^2 = \sum_{i=1}^n (\tilde{r}_i - \tilde{c}_{li})^2 \quad (6.5.6)$$

为接收序列 $\tilde{\mathbf{R}}$ 与码字 $\tilde{\mathbf{C}}_l$ 之间的欧几里德距离(或称毕达哥拉斯距离)。若 $\tilde{\mathbf{C}}_l$ 与 $\tilde{\mathbf{R}}$ 之间的欧几里德距离最小，则译码器把 $\tilde{\mathbf{R}}$ 译成 $\tilde{\mathbf{C}}_l$ ，称这种译码器为**欧几里德译码器**或**最小欧几里德距离译码器**。由此可知，最大似然译码也就等价于最小欧几里德距离译码。

下面定义两个实数序列 $\tilde{\mathbf{R}}$ 、 $\tilde{\mathbf{C}}_l$ 之间的内积(点积) $\tilde{\mathbf{R}} \cdot \tilde{\mathbf{C}}_l$ 为

$$\tilde{\mathbf{R}} \cdot \tilde{\mathbf{C}}_l = \sum_{i=1}^n \tilde{r}_i \cdot \tilde{c}_{li} \quad (6.5.7)$$

由此，式(6.5.6)可写成

$$\min d_E^2 = \sum_{i=1}^n (\tilde{r}_i - \tilde{c}_{li})^2 = \min(\tilde{\mathbf{R}} \cdot \tilde{\mathbf{R}} - 2\tilde{\mathbf{R}} \cdot \tilde{\mathbf{C}}_l + \tilde{\mathbf{C}}_l \cdot \tilde{\mathbf{C}}_l) \Rightarrow \max \tilde{\mathbf{R}} \cdot \tilde{\mathbf{C}}_l \quad (6.5.8)$$

这是因为 $\tilde{\mathbf{R}} \cdot \tilde{\mathbf{R}}$ 与 l 无关，而 $\tilde{\mathbf{C}}_l \cdot \tilde{\mathbf{C}}_l$ 对每个码字来说都是 n ，所以最小欧几里德译码就等价于**最大内积译码**。由于两个 n 长序列 $\tilde{\mathbf{R}}$ 与 $\tilde{\mathbf{C}}_l$ 之间的相关函数 $\rho(\tilde{\mathbf{R}}, \tilde{\mathbf{C}}_l)$ ，通常定义为

$$\rho(\tilde{\mathbf{R}}, \tilde{\mathbf{C}}_l) = \sum_{i=1}^n \tilde{r}_i \tilde{c}_{li} = \tilde{\mathbf{R}} \cdot \tilde{\mathbf{C}}_l \quad (6.5.9)$$

因此最大内积译码就是**最大相关译码**。所以，当码字等概发送，噪声为白高斯噪声时，最大后验概率译码就等价于最大似然译码，也就等价于最小欧几里德距离译码或等价于最大相关译码。

由式(6.5.8)可得

$$\begin{aligned} d_E^2 &= \tilde{\mathbf{R}} \cdot \tilde{\mathbf{R}} - 2\tilde{\mathbf{R}} \cdot \tilde{\mathbf{C}}_l + \tilde{\mathbf{C}}_l \cdot \tilde{\mathbf{C}}_l \\ &= n - 2(n - d_h - d_h) + n = 4d_h \end{aligned} \quad (6.5.10)$$

所以

$$d_E = 2\sqrt{d_h} \quad (6.5.11)$$

这里， d_h 是 $[n, k]$ 码的最小汉明距离 d 。由上式可知，在 BSC 信道中(这时把解调器的输出作硬判决，只取 $+1, -1$ 两个值，故 $\tilde{\mathbf{R}} \cdot \tilde{\mathbf{R}} = n$)，最小欧几里德距离译码也就是最小汉明距离译码。所以在 BSC 中，最大后验概率译码、最大似然译码、最大相关译码都是等价的，它们都是使译码错误概率最小的最佳译码方法。

但是要实现上述各种最佳译码，译码器都必须计算、比较 2^k 个后验概率或者似然函数、相关函数和最小距离。当 $n=k/R$ 很大时，所有这些最佳译码方法都是不切实际，难以实现的。另一方面，由前几节可知，利用码的代数结构进行硬判决译码的硬译码器却比较简单，易于工程实现。因此，我们必须在这些最佳译码方法与硬判决译码之间设立一种折衷方法，既能充分利用波形信道输出的各种有用信息，又使译码器不太复杂。而软判决译码就提供了这种所希望的折衷方法，它的性能接近最佳译码，但是复杂性并不大，易于工程实现。

二、模拟电压的量化及其距离函数

无论是最大似然译码、最小欧几里德距离译码还是最大相关译码，都要用解调器输出的实数序列 $\tilde{\mathbf{R}}$ 进行计算。由于 $\tilde{\mathbf{R}}$ 的每个元素 \tilde{r}_i 可任意取值，是一模拟量，而在实际上要利用模拟量进行计算是很难实现的，因此必须予以量化。若量化级数有限且有足够的量化电平级数，则与原来的模拟量基本相同。当然，从工程实用观点看，希望量化级数少一些，这样可使模数转换器的成本较低，且所需的代表量化电平的码元数目较少，计算较易。通常除了二进制量化（硬判决）外，最通常的是线性（均匀）八电平量化，如图 6-15(a) 所示。

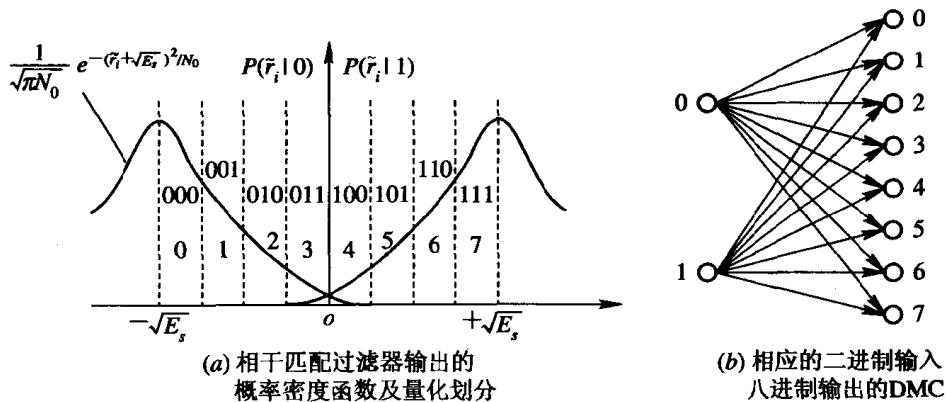


图 6-15 线性量化与相应的 DMC

在二进制输入， Q 进制输出的信道中，由匹配滤波器输出的结果完全由转移概率密度函数 $P(j|x)$ 决定，即给定输入为 x （0 或 1 或相应的 $1, -1$ ）时，输出电压幅度落在电平区间 j 的概率。由于仅讨论对称信道，因而 $P(j|0) = P((Q-1-j|1))$ ，如图 6-15(b) 所示， $P(2|0) = P(5|1)$ 。

在 BSC 信道中，通常用汉明距离度量码字之间的区别程度。而在二进制输入 Q 进制输出的离散无记忆信道（DMC）中，我们必须寻找适合于这种信道特点的、度量码字之间差别的距离度量。

首先，我们定义接收码元 \tilde{r}_i 与发送码元 c_i 之间的软判决距离

$$d_s = j_i \quad (6.5.12)$$

这里， j_i 是接收码元 \tilde{r}_i 落入某一量化电平区间的区间号。如图 6-15(a) 中所示，若 \tilde{r}_i 的值在第 5 区间，则 j_i 等于 5。可以证明^[12]，这种定义基本上能反映出接收码元相对发送码元的似然函数值的比。由此，可定义接收序列 $\tilde{\mathbf{R}}$ 与码字 $\tilde{\mathbf{C}}_i$ 之间的软判决距离

$$d_s(\tilde{\mathbf{R}}, \tilde{\mathbf{C}}_i) = d_s(\tilde{\mathbf{C}}_i, \tilde{\mathbf{R}}) = \sum_{i=1}^n d_{si} = \sum_{i=1}^n j_i \quad (6.5.13)$$

可以证明^[13]

$$\max \log_b P(\mathbf{R}|\mathbf{C}_i) = \min \sum_{i=1}^n j_i = \min d_s(\tilde{\mathbf{R}}, \tilde{\mathbf{C}}_i) \quad (6.5.14)$$

也就是说在二进制输入 Q 进制输出的 DMC 中，若用这样定义的软判决距离，则最大似然译码也就近似等价于最小软判决距离译码。

下面定义发送码字 $C_i = (c_{i1}, \dots, c_{in})$ 的软判决重量(下简称软重量) w_{is} 。首先, 定义发送码元 c_{ii} 的软重量 w_{isi} , 在二进制输入 Q 进制输出的 DMC 中, 定义

$$w_{isi} = \begin{cases} Q - 1 & \text{当 } c_{ii} = 1 \\ 0 & \text{当 } c_{ii} = 0 \end{cases} \quad (6.5.15)$$

所以 C_i 的软重量定义为

$$w_{is} = \sum_{i=1}^n w_{isi} \quad (6.5.16)$$

若把 C_i 的每个分量 c_{ii} 用它的软重量 w_{isi} 表示, 则 C_i 的软重量表示

$$c_{is} = (w_{i1}, \dots, w_{in}) \quad (6.5.17)$$

如在八进制输出的 DMC 中, 码字 $C_i = (1101000)$ 的软重量表示 $c_{is} = (7707000)$, 因而它的软重量

$$w_{is} = \sum_{i=1}^7 w_{isi} = 21$$

一个码字或序列的软重量与汉明重量(硬重量) w_{ih} 之间的关系为

$$w_{is} = (Q - 1)w_{ih} \quad (6.5.18)$$

由码字和序列的软重量表示, 可以定义两个码字或序列 C_i 与 C_k 之间的软判决距离(软距离)为

$$d_s(C_i, C_k) = d_s(C_k, C_i) = |w_{is} - w_{ks}| = \sum_{i=1}^n |w_{isi} - w_{kni}| \quad (6.5.19)$$

它与汉明距离 $d_h(C_i, C_k)$ 的关系为

$$d_s(C_i, C_k) = (Q - 1)d_h(C_i, C_k) \quad (6.5.20)$$

在二进制输入八进制输出信道中, 若接收码字 $R_s = (0034060)$, 其软重量为 13。若发送码字 $C_i = (0077070)$, 则它们之间的软距离 $d_s(R_s, C_i) = 8$, 它就是

$E_s = C_i - R_s = (0 - 0, 0 - 0, 7 - 3, 7 - 4, 0 - 0, 7 - 6, 0 - 0) = (0043010)$ 的软重量。定义

$$E_s = R_s - C_i = (|w_{r1} - w_{i1}|, \dots, |w_{rn} - w_{in}|) = (e_{s1}, \dots, e_{sn})$$

为接收码字 R_s 与发送码字 C_i 之间的软错误图样, 其中 w_{rsi} 为 R_s 中的第 i 个分量 r_{si} 的软重量, e_{si} 为八进制数字, 是 E_s 中的第 i 个分量的软重量。由此可知, 最小软距离译码就是寻找一个有最小软重量的软错误图样的译码, 这种概念完全与最小汉明距离译码相同。

下面定理说明, 码的纠错能力与软判决距离之间的关系。

定理 6.5.1 一个有最小距离为 d_h 的二进制 $[n, k]$ 线性分组码, 有最小软距离 $d_s = (Q - 1)d_h$, 它一定能纠正软重量为

$$t_s < \frac{(Q - 1)d_h}{2} \quad (6.5.21)$$

的任何错误图样。

证明 设在信道中出现了一个软重量小于 $(Q - 1)d_h/2$ 的错误图样 E , 则接收码组 R_s 与发送码字 C_i 之间的软距离 $d_s(R_s, C_i)$ 小于 $(Q - 1)d_h/2$, 它等于 E 的软重量 w_{Es} 。假设有另一码字 C_k , 它与接收码组的软距离 $d_s(R_s, C_k)$ 小于等于 w_{Es} , 则由距离三角不等式可知, C_i 与 C_k 之间的软距离

$$\begin{aligned}
 d_s(C_{ls}, C_{ks}) &\leq d_s(C_{ls}, R_s) + d_s(R_s, C_{ks}) \\
 &< (Q - 1)d_h/2 + (Q - 1)d_h/2 \\
 &= d_h(Q - 1)
 \end{aligned}$$

这与码有最小距离为 $d_h(Q - 1)$ 的假设相矛盾。因此不可能存在另一码字 C_k , 它与接收码字的软距离小于 $(Q - 1)d_h/2$, 故能用最小软距离译码器把 R 正确地译为 C_l 。 ■

推论 6.5.1 有最小距离为 d_h 的二进制线性分组码, 在 Q 进制输出的 DMC 中, 当信噪比很高时, 应用最小软判决距离译码, 能纠正 $d_h - 1$ 个硬判决错误。

证明 在高信噪比下, 引起硬判决错误的最大可能的错误分量的软重量是 $Q/2$, 但码能纠正 $((Q - 1)d_h/2) - 1$ 的软重量错误图样, 所以码能纠正

$$\left[\frac{(Q - 1)d_h}{2} - 1 \right] / (Q/2) \approx d_h - 1$$

个硬判决错误。 ■

三、码元可信度与量化电平之间的关系

在最大似然译码中, 每个码元的似然函数 $\log_b p(\tilde{r}_i | \tilde{c}_{li})$ 越大, 说明接收到的 \tilde{r}_i 是 \tilde{c}_{li} 的可能性越大。因此, $\log_b p(\tilde{r}_i | \tilde{c}_{li})$ 反映了接收码元是发送码元 c_{li} 的可信程度(以下简称可信度)。由前面介绍可知, 在 Q 进制输出的 DMC 中, 量化电平间隔的值 j 大致反映了接收码元相对发送码元的似然函数值之比, 因而 j 也同样能反映出接收码元的可信度。由图 6-15(a) 可知, j 越大说明接收码元是 1 的可能性越大, 是 0 的可能性越小; 反之, 则说明是 0 的可能性越大。

现在以八($Q = 2^m = 2^3$)电平量化的 DMC 为例, 进一步说明码元可信度与量化电平之间的关系。可把图 6-15(a)画成图 6-16。

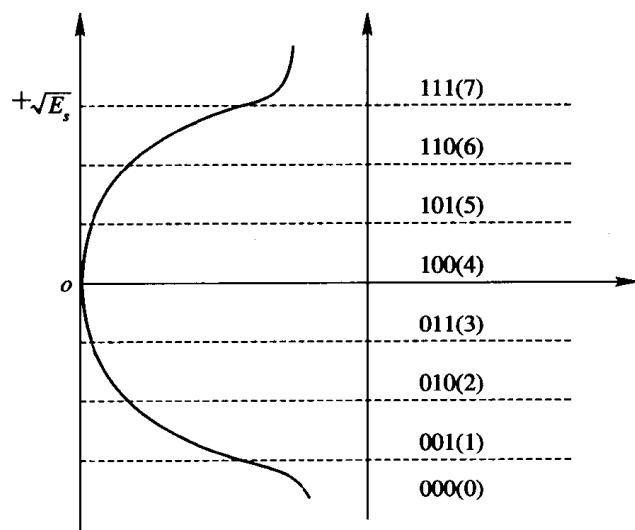


图 6-16 量化电平与码元可信度之关系

由此图看到, 在硬判决情况下, 判决门限以 0 为界。若解调器输出的电压大于 0 则判为 1(或 0), 小于或等于 0 则判为 0(或 1)。从该图还可看到, 若接收码元的电压落在 $j=7$

区间，则发送码元是 1 的可信度最大，为 0 的可信度最小(即与1的软距离最小，与 0 的软距离最大)。 j 越小，发送码元是 1 的可信度越小，是 0 的可信度越大。若以 3 个二进制码元表示八进制量化的电平值，则其第一位码元表示硬判决的结果，而后二位则表示此判决的可信度。如 111，表示 $j=7$ ，第一位二进制码元为 1，表示硬判决值为 1，后二位为 11，表示此硬判决的可信度最高，为 11(3)。同理， $j=6, 5, 4$ 时，它们的二进制表示分别为 110, 101, 100，其硬判决值均为 1，硬判决可信度分别为 10(2), 01(1) 和 00(0)。当 $j=4$ 时，判为 1 的可信度最低。同样，当 $j=0, 1, 2$ 和 3 时，硬判决结果均为 0，但其判决可信度分别为 3, 2, 1 和 0，它们正好是三位二进制码元表示中后二位码元取值的补。因此，若以 α_i 表示第 i 个码元的可信度，则 α_i 与 j 的二进制表示的后 $m-1$ ($Q = 2^m$ 进制量化) 个码元的关系如下：

$$\alpha_i = \begin{cases} j \text{ 的后 } m-1 \text{ 个码元所表示的值,} & \text{若第一个码元为 1} \\ j \text{ 的后 } m-1 \text{ 个码元取补所表示的值,} & \text{若第一个码元为 0} \end{cases}$$

如收到的序列为 (0, 6, 5, 4, 3)，则相应的二进制表示为 (000, 110, 101, 100, 011)，因而硬判决结果为 (01110)，可信度序列 $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5) = (00+11=11(3), 10(2), 01(1), 00(0), 11+11=00(0)) = (3, 2, 1, 0, 0)$ ，说明最后两个码元的可信度最低，第一个码元的可信度最高。

综上所述，现在可以有两种方法表示接收序列的可信度：一是直接用量化电平间隔 j 表示，如在八电平量化中，若 j 为 7、0，则可信度最高，6、1 次之，5、2 再次之，而 4、3 可信度最低。另一方法是用两个序列，第一个序列为硬判决序列，第二个序列 α 是说明第一个序列中每个码元可信度的信道测量信息序列。除此以外，还有其它表示方法，这将在后面介绍。

由上可知，对解调器的输出进行量化后所得之量化电平值 $j(0, 1, \dots, 2^m-1)$ ，近似地表示了接收码元似然函数值的相对大小。显然，量化电平越多，则越能精确地接近似然函数，越能准确地反映出接收码元的可信度，从而使译码器的译码性能更接近于最大似然译码。但是随着量化电平数目的增多，模数转换器和译码器的复杂性很快增长，以致难以实现。分析和计算机模拟表明，一般用八电平，最多十六电平量化，其译码性能就可达到(或接近)无限量化时的性能。

另一方面，由于似然函数并非是线性的，因此用非线性量化代替线性(均匀)量化，应能改善译码器的性能。但是计算机模拟表明，当用非线性量化代替线性量化时，对译码器性能的影响并不大。一般情况下用八电平线性量化，其译码器性能与无限量化时相当接近。因此，工程实用上，采用八电平至多十六电平量化也就足够了。

四、编码增益与软判决增益

这一节讨论利用纠错码的硬判决译码或软判决译码后，每传输一个信息元所能获得的能量节省。众所周知，在通信系统性能分析中最有用的图是信噪比 E_b/N_0 与误码率 p_e 之间的关系图。在某一误码率下，应用一特定的纠错码系统后，相对没有应用时所能获得的信噪比的减少分贝数，定义为此纠错码的**编码增益**。因此编码增益不仅与所用的纠错码有关，而且与信噪比有关。

决定编码增益的常用方法，是对应用和未应用纠错码的同一系统画出 E_b/N_0 与 p_e 的关

系曲线。例如，图 6-17 中画出了利用 [23, 12]Golay 码硬判决译码时的性能曲线和未用编码时的性能曲线。

由该图看到，当 p_e 为 10^{-5} 时，编码增益为 2.15 dB ，而在 p_e 等于 10^{-3} 时为 1.35 dB 。当信噪比非常低时，编码增益变为负值，这说明利用纠错码后性能更差。这种门限效应是所有利用纠错码的差错控制系统所共有的。所以，在实际工程中应用纠错码时，必须对信噪比提出一个最低要求，若低于此值，则必须设法改善信道质量，以达到要求，否则就不能应用纠错码来改善通信质量。

当然在某些情况下，信噪比并不能完全决定编码增益。例如，在某些信道如高频、散射等信道中，单纯利用增加功率并不能使误码率减低多少，而利用某一种纠错码后却能使误码率显著减低，这时编码增益将很大。

在性能分析中，另一个比较有用的量

是系统在高信噪比时所获得的编码增益，称这种增益为渐近编码增益，它与所用码的码速率 R 及最小距离 d_h 有关，且易于计算，它可以在无限量化信道（模拟信道或未量化信道）或硬判决信道中定义。

在高信噪比下，使用码速率为 R ，纠 t 个错误的码，在加性白高斯噪声（AWGN）信道中，用硬判决译码时，其误码率为^[7]

$$p_b \approx K p_e^{t+1} = K \{Q((2RE_b/N_0)^{1/2})\}^{t+1} \quad (6.5.22)$$

式中， K 是小于 1 的系数。没有应用纠错码的数传系统，如差分相干解调（DPSK），在 AWGN 信道中的误码率为

$$p'_b = K Q[(2E'_b/N_0)^{1/2}] \quad (6.5.23)$$

在高信噪比时， Q 函数可以用下式渐近表示：

$$Q(x) \approx e^{-x^2/2} \quad (6.5.24)$$

当 $p_b = p'_b$ 时，所需信噪比的减低，即为纠错码系统所获得的渐近编码增益。把式(6.5.24)代入式(6.5.23)和式(6.5.22)得

$$K \{\exp(-\frac{2RE_b}{N_0} \frac{1}{2})\}^{t+1} = K \{\exp(-\frac{2E'_b}{N_0} \frac{1}{2})\}$$

两边取对数，当 E_b/N_0 很大时由上式可知：

$$E'_b/E_b = R(t+1)$$

由此可知渐近编码增益

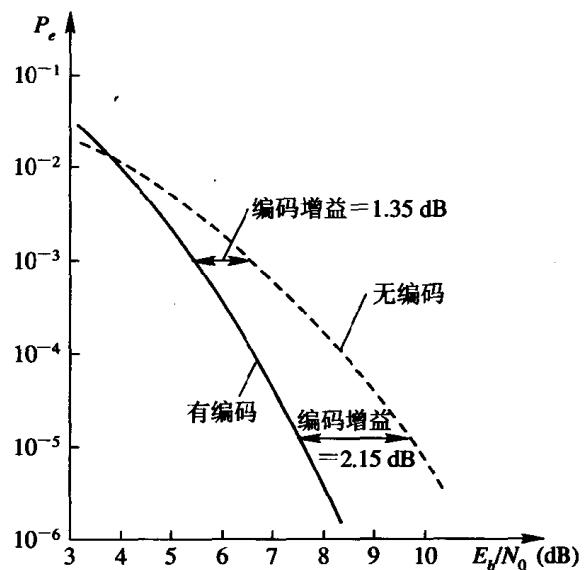


图 6-17 应用与未应用[23, 12]码时， E_b/N_0 与 p_e 之关系曲线

$$G = 10 \lg R(t+1) = 10 \lg R \left(\frac{d_h + 1}{2} \right) \quad (6.5.25)$$

对 $R=1/2$ 并纠两个错误的码来说, 可知渐近编码增益约为 1.7 dB。由于渐近编码增益是在 $E_b/N_0 \rightarrow \infty$ 时获得的, 因此在中等信噪比时, 实际所能获得的编码增益要小于此值, 所以式(6.5.25)是编码增益的上限。

应用类似方法可以证明^[7], 未量化信道的渐近编码增益

$$G_s = 10 \lg R d_h \quad (6.5.26)$$

与式(6.5.25)比较可知, 当高信噪比时, 无限量化的软判决译码比硬判决译码要好 3 dB, 称此增益为软判决增益。当然, 在中等信噪比和量化级数为有限(例如八或十六电平)情况下, 所能获得的软判决增益要小于 3 dB。通常, 在八电平量化和中等信噪比下, 约能获得 2dB 的软判决增益(对中等码长和中等纠错能力的码而言)。但是, 若为非 AWGN 信道, 则软判决增益要超过此值。

§ 6.6 码字错误概率最小的软判决译码

使码字错误概率最小的译码方法, 其性能接近最大似然译码, 且实现比较简单, 因此是目前用得最为普遍的一类软判决译码方法。这类译码方法是对收到的整个矢量(\mathbf{R})进行总的判决, 以决定与接收的 \mathbf{R} 最似然的码字。

在这类算法中, 最主要的有广义最小距离(GMD)算法和契斯(Chase)算法。下面先介绍 GMD 算法, 然后介绍 Chase 算法。

一、GMD 译码算法

GMD 算法是1966年首先由福尼(Forney)提出的一类逐字判决的、软判决译码方法。它是所有其它逐字判决软判决译码算法的基础。下面先介绍与此有关的距离函数, 然后介绍它的译码方法。

1. 广义距离与广义最小距离译码 除了用量化电平来度量接收码元可信度之外, 还可以定义另一种码元度量。

设由解调器输出的实数序列 $\tilde{\mathbf{R}} = (\tilde{r}_1, \tilde{r}_2, \dots, \tilde{r}_n)$, 送入一个似然比估值器, 它能计算码元的对数似然函数比

$$L_i = \log_b \frac{P(\tilde{r}_i | 0)}{P(\tilde{r}_i | 1)} \quad (6.6.1)$$

再把 L_i 送入一个有特征为 $g(L_i)$ 的量化器, 使其输出

$$\alpha_i = q(L_i)$$

最后把 $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)$ 序列送入译码器译码。由于 L_i 可取任意实数, 因此 α 序列中的每一分量 α_i 也可取任意实数。这样, 可把 α 序列看成是 n 维实数空间中的一个点。

对量化器特征 $g(L_i)$ 的不同选择, 就得到了不同的码元度量。例如, 若选

$$g(L_i) = \begin{cases} 1 & L_i \geq 0 \\ -1 & L_i < 0 \end{cases}$$

则为汉明距离度量。若选

$$g(L_i) = \begin{cases} 1 & L_i \geq T \\ 0 & -T < L_i < T \\ -1 & L_i \leq -T \end{cases}$$

式中, T 是一门限, 则相应于有删除区间的情况。若选

$$g(L_i) = \begin{cases} 1 & L_i \geq T \\ L_i/T & -T < L_i < T \\ -1 & L_i \leq -T \end{cases} \quad (6.6.2)$$

则由此可知, 量化器输出 α_i 在 ± 1 之间取值。因此, 除了两端点的硬限幅之外, 该量化器输出的 α_i 表示了码元对数似然函数比。

我们把 α_i 定义为两个接收码元之间差别程度的度量, 称为码元之间的广义距离。由此可见, 广义距离 α_i 是一个可在 ± 1 之间任意取值的实数, 而不像汉明距离和软距离那样是整数。显然, α_i 的绝对值 $|\alpha_i|$ 越大, 则说明接收码元是 0 或 1 的可能性越大, 码元的可信度越大。

在二进制汉明距离译码中, 若 $[n, k]$ 码的最小汉明距离为 d_h , 则它至多只能纠正 $t \leq \left\lfloor \frac{d_h-1}{2} \right\rfloor$ 个错误。这相应于至多只有一个码字 C_i 满足

$$d_h(\tilde{R}_h, \tilde{C}_i) \leq \frac{1}{2}(d_h - 1) \quad (6.6.3)$$

式中, \tilde{R}_h 是硬判决序列 R_h 的传输映射序列, \tilde{C}_i 是 C_i 的传输映射序列。由式(6.5.10)知, 式(6.6.3)成立, 意味着至多只有一个码字 C_i 满足

$$d_E^2(\tilde{R}_h, \tilde{C}_i) \leq 2(d_h - 1) \quad (6.6.4)$$

在二进制情况下, $\tilde{R}_h \times \tilde{R}_h = n$, 因此由式(6.5.10)可知:

$$d_E^2(\tilde{R}_h, \tilde{C}_i) = 2(n - \tilde{R}_h \cdot \tilde{C}_i)$$

由该式和式(6.6.4)知, 最多只有一个码字 C_i 满足

$$\tilde{R}_h \cdot \tilde{C}_i \geq n - (d_h - 1) > n - d_h \quad (6.6.5)$$

能实现式(6.6.5)译码的译码器, 显然是一个最大似然译码器。这种译码方法是一种不完备译码, 也就是说译码器可能找不到一个满足式(6.6.5)的码字 C_i , 此时译码失败, 说明在接收序列 \tilde{R} 中, 存在一个不可纠正的错误图样码。

若量化器的输出只取 ± 1 和 0 三种情况, 即删除情况, 则同样可以证明, 一个纠错纠删译码器只能找到一个 C_i (倘若存在的话)码字满足式(6.6.5)。

下面定理说明, 如果量化器的输出是一个在 ± 1 之间任意取值的实数序列 $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)$, 则也有类似的结论。

定理 6.6.1 令 $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)$ 是输入至译码器的实数序列, 其中 $|\alpha_i| \leq 1$, 则至多只有一个码字 $C_i \in (n, k, d_h)$ 满足

$$\alpha \cdot \tilde{C}_i > n - d_h \quad (6.6.6)$$

证明 设 C_i 是满足上面不等式的码字。若还有另一个码字 C'_i 也满足此不等式, 则 C_i 和 C'_i 之间至少有 d_h 个不同的分量。令:

$$s = \{i, c_{hi} \neq c'_{hi}\}$$

$$\bar{s} = \{i, c_{hi} = c'_{hi}\}$$

因而

$$\alpha \cdot \tilde{C}_l = \sum_{i \in \tilde{s}} \alpha_i \tilde{c}_{li} + \sum_{i \in s} \alpha_i \tilde{c}_{li} = A_1 + A_2$$
$$\alpha \cdot \tilde{C}_l = \sum_{i \in \tilde{s}} \alpha_i \tilde{c}'_{li} + \sum_{i \in s} \alpha_i \tilde{c}'_{li} = A_1 - A_2$$

但是

$$A_1 = \sum_{i \in \tilde{s}} \alpha_i \tilde{c}_{li} \leq n - d_h$$

又因为 $\alpha \cdot \tilde{C}_l > n - d_h$, 所以 $A_2 > 0$, 由此可知:

$$\alpha \cdot \tilde{C}_l \leq n - d_h$$

所以, \tilde{C}_l 不满足式(6.6.6)。 ■

我们定义能完成式(6.6.6)的译码器, 称为**广义最小距离译码器**, 这种译码方法称为**广义最小距离译码**, 简称 GMD。

例 6.7 二进制[3, 1]码, 两个码字为 $C_0 = (000)$, $C_1 = (111)$, 相应的 $\tilde{C}_0 = (111)$, $\tilde{C}_1 = (-1 - 1 - 1)$, 它们之间的 $d_h = 3$ 。由量化器输出的码元似然函数比矢量 $L = (2.0 - 0.4 - 0.2)$, 最大似然译码器将做以下内积:

$$L \cdot \tilde{C}_0 = (2.0 - 0.4 - 0.2) \cdot (111) = 1.4$$

$$L \cdot \tilde{C}_1 = (2.0 - 0.4 - 0.2) \cdot (-1 - 1 - 1) = -1.4$$

因此, 译成 $\hat{C}_0 = C_0 = (000)$ 。

但是如果用硬判决译码器, 则量化器的输出 $R_h = (011)$, 译码器错译成 $C_1 = (111)$ 。

如果用纠错纠删译码器, 并选门限 $T = 1$, 则在此时量化器输出的值 $R = (100)$, 因而纠错纠删译码器译为 $C_0 = (000)$ 。

若用 GMD 译码器, 量化器在门限 $T = 1$ 时的输出为 $(1 - 0.4 - 0.2)$ 。这时 GMD 译码器把最少可信度码元, 即 $|\alpha_i|$ 最小的码元作删除处理, 得到一个试探序列 $C'_0 = (100)$, 从而纠错纠删译码器译得 $\hat{C}_0 = C_0 = (000)$ 码字, 最后检验

$$\alpha \cdot \tilde{C}_0 = (1 - 0.4 - 0.2) \cdot (111) = 0.4 > n - d_h = 0$$

因此 GMD 译码器输出 $C_0 = (000)$ 。 ■

2. GMD 译码过程及流程图 从上面讨论中可以把 GMD 译码器的译码过程归结如下:

第一步, 从量化器输出的 α 序列中删去 i 个最不可信的码元, 并按照 α_i 的正负号, 把 α 序列变换为由 $-1, 1, 0$ 元素组成的试探序列 α' 。

第二步, 把 α' 序列送入纠错纠删译码器, 得到一个已译码字 C'_l 。

第三步, 检验 $\alpha \cdot \tilde{C}_l > n - d_h$ 是否满足。若满足则输出 \hat{C}_l , 它就是 GMD 译码器给出的码字; 若不满足, 则在 α 序列中删除 $i+1$ 个最不可信码元, 重新回到第一步。

第四步, 若试探译码到最后, 仍没有找到一个满足 $\alpha \cdot \tilde{C}_l > n - d_h$ 的码字 C_l , 则译码器宣告译码失败, 并输出一个特定信号通知用户。

在上述译码过程中, 一个很自然的问题是: 如果存在有满足式(6.6.6)的码字, 那么要进行多少次试探译码才能找到该码字呢? 由第三章可知, $[n, k, d_h]$ 线性分组码至多能纠正 $d_h - 1$ 个删除错误。因此, 用删除最不可信码元进行纠错译码寻找满足式(6.6.6)码字的试探次数, 至多只能进行 d_h 次(第一次不删除)。下面定理说明了这点。

定理 6.6.2 若 $\alpha \cdot \tilde{C}_l > n - d_h$, 则对码字 C_l 至少有一个 a , $a=0, 1, 2, \dots, d_{h-1}$, 有 $\alpha \cdot \tilde{C}_l^{(a)} > n - d_h$, 式中的 $\tilde{C}_l^{(a)}$ 是第 a 次试探序列得到的码字。

证明 设 $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)$, 按照 $|\alpha_i|$ 的大小, 把 α 中的每个分量排列如下:

$$|\alpha_{i1}| \leq |\alpha_{i2}| \leq \dots \leq |\alpha_{in}|.$$

令

$$s(l) = \begin{cases} 1 & \alpha_i \geq 0 \\ -1 & \alpha_i < 0 \end{cases}$$

再设

$$q_a(\alpha_{ij}) = \begin{cases} 0 & 1 \leq j \leq a \\ s(\alpha_{ij}) & a+1 \leq j \leq n, 0 \leq a \leq n \end{cases}$$

则序列

$$\mathbf{q}_a = (q_a(\alpha_1) q_a(\alpha_2) \dots q_a(\alpha_n))$$

是有 $n-a$ 个分量等于 ± 1 , a 个分量等于 0 的 n 维实数空间中的一个点。所以, 若有一个码字与它的距离小于 d_h , 则可用纠删译码器找到它。 \mathbf{q}_0 相应于无删除情况, 它相应于硬判决序列, 而 \mathbf{q}_1 相应于删除一个最不可信码元的试探序列, \mathbf{q}_2 是删除两个最小可信度码元的试探序列等等。令:

$$\begin{aligned} \lambda_0 &= |\alpha_{i1}| \\ \lambda_a &= |\alpha_{i, a+1}| - |\alpha_{i, a}| \quad 1 \leq a \leq n-1 \\ \lambda_n &= 1 - |\alpha_{in}| \end{aligned}$$

因而, $0 \leq \lambda_a \leq 1$, 且

$$\sum_{a=0}^n \lambda_a = 1$$

所以 λ_a 具有概率性质, 且

$$\sum_{a=0}^{j-1} \lambda_a = |\alpha_{ij}|$$

所以

$$\alpha = \sum_{a=0}^n \lambda_a \mathbf{q}_a(\alpha_a)$$

假设对所有 a , $\alpha \cdot \mathbf{q}_a \leq n - d_h$, 则

$$\alpha \cdot \tilde{C}_l = \sum_{a=0}^n \lambda_a \mathbf{q}_a \cdot \tilde{C}_l \leq (n-d) \sum_{a=0}^n \lambda_a = n - d_h$$

这与假设 $\alpha \cdot \tilde{C}_l > n - d_h$ 相矛盾。所以, 对某一个 a , 必有

$$\alpha \cdot \mathbf{q}_a = \alpha \cdot \tilde{C}_l^{(a)} > n - d_h$$

■

这个定理是 GMD 译码的充分条件, 而定理 6.6.1 则是必要条件。可以看到, GMD 译码器的试探次数至多为 $n+1$ 次, 事实上, 这当然是不可能的。正如前面提到过的, 一个纠删译码器至多只能纠正 $d_h - 1$ 个删除, 故至多只能有 d_h 次试探。因此, 当 $a > d_h - 1$ 时, 不存在任何 \mathbf{q}_a 序列。事实上, 试探次数比 $d_h - 1$ 还要少, 例如对任何 \mathbf{q}_a 序列, 如果相应的 $\lambda_a = 0$, 则表示相邻两个分量有相同大小的 $|\alpha_i|$, 因此该两个符号可同时删除, 这时 \mathbf{q}_{a+1} 就不必试验了。一般而言, 有相同大小 $|\alpha_i|$ 的码元都可同时删除, 当然 $|\alpha_i| = 1$ 的码元决不能

删除。

如果在译码中利用纠错纠删译码器，则由第三章知，对于一个纠正 t 个错误 s 个删除的译码器，要求码的距离至少为 $2t+s+1$ 。所以，对任一 q_a ，如果 $d_h - a$ 为偶数，则仅当 q_a 中的错误个数 $t \leq \lfloor (d_h - a)/2 \rfloor - 1$ 时，才能由纠错纠删译码器给出正确的码字 C_t 。但是，这种存在于 q_a 序列中的错误图样，也同样能在 q_{a+1} 作试探序列时被纠错纠删译码器正确纠正，而给出正确码字 C_t 。所以，如果在试探译码过程中，以每次删除 $2i$ 个最少可信度码元的速度进行试探译码，则试探次数几乎可减少一半。

推论 6.6.1 为了由 GMD 译码器译出满足式(6.6.6)的码字 C_t ，如果它确实存在的话，则利用纠错纠删译码器译码的试探次数，最多不超过 $(d_h + 1)/2$ 次。

为了对 GMD 译码过程有一清晰的概念，在图 6-18 中给出了译码流程图，由此图可以很容易用计算机软件或微处理器实现译码。

GMD 译码算法是一种最基本的软判决译码算法。近几年来不少作者^[14, 18, 19]研究了它与 WED、APP 等译码算法的关系，并探讨了放松式(6.6.6)条件的可能性。已证明对二进制输入 $Q (= 2^n)$ 进制输出的 DMC 中，如果进行如下变换：

$$\alpha_i = \frac{2h_i}{Q-1} - 1$$

$$h_i = 0, 1, 2, \dots, Q-1$$

式中， h_i 是第 i 个接收码元所处的电平区间，则对这种量化信道，同样可以用式(6.6.6)作为判决条件，进行软判决译码^[18]。

二、Chase 译码算法

1972年契斯(Chase)提出了另一种使码字错误概率最小的软判决译码算法^[10]，它的性能接近最大似然译码。该算法与 GMD 算法的区别是：用硬判决纠错译码器代替 GMD 中的纠删或纠错纠删译码器，并且试探序列的选择也与 GMD 稍有不同。因此，实际上 Chase 算法是 GMD 算法的修正，二者之间无本质差别。若无特别说明，下面所讨论的算法都是

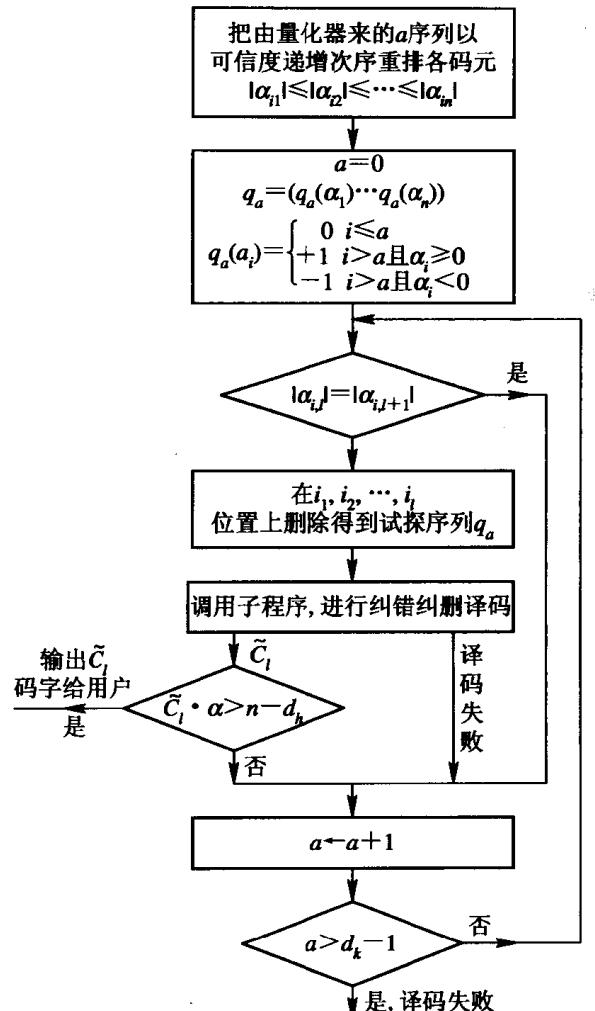


图 6-18 GMD 译码器流程图

针对二进制码的。

1. Chase 算法的基本原理 在该算法中，利用硬判决译码器，根据不同的试探序列产生几个候选码字，然后把它们与接收序列进行比较，挑选一个与接收序列有最近软距离的候选码字作为译码器的输出码字。

设发送码字 $\mathbf{C}_l = (c_{l1}, c_{l2}, \dots, c_{ln})$ ，接收序列 $\tilde{\mathbf{R}} = (\tilde{r}_1, \tilde{r}_2, \dots, \tilde{r}_n) = \tilde{\mathbf{C}}_l + \tilde{\mathbf{E}} = (\tilde{c}_{l1} + \tilde{e}_1, \tilde{c}_{l2} + \tilde{e}_2, \dots, \tilde{c}_{ln} + \tilde{e}_n)$ ，它的硬判决序列 $\mathbf{R}_h = (r_1, r_2, \dots, r_n)$ 。信道错误图样 $\tilde{\mathbf{E}} = \tilde{\mathbf{R}}_h - \tilde{\mathbf{C}}_l = (\tilde{e}_1, \tilde{e}_2, \dots, \tilde{e}_n)$ ，相应的硬判决错误图样 $\mathbf{E} = (e_1, e_2, \dots, e_n)$ ，此错误图样的汉明重量 $w_h(\mathbf{E}) = \sum_{i=1}^n e_i$ 。

二进制硬判决译码器的任务是寻找一个码字，或等价于寻找一个错误图样 \mathbf{E}_m ，使

$$w_h(\mathbf{E}_m) \leq \lfloor (d_h - 1)/2 \rfloor \quad (6.6.7)$$

若此不等式满足，则译码器能找到一个唯一的码字输出；否则就找不到。这时译码器指出接收序列中有不可纠正的错误图样，这种译码就是通常所说的不完备译码。但是，对于完备译码器来说，由于译码器必须进行判决，输出一个码字，因此完备译码器是寻找一个码组使

$$\min w_h(\mathbf{R}_h - \mathbf{C}_l) \quad l = 1, 2, \dots, 2^k \quad (6.6.8)$$

所以完备译码器总能找到一个码字 \mathbf{C}_j ，即使 $w_h(\mathbf{R}_h - \mathbf{C}_j) > \lfloor (d_h - 1)/2 \rfloor$ 也在所不顾。

应用上述的类似方法，可以定义一个利用信道测量信息或码元可信度信息的完备软译码器，它译出的码字满足：

$$\min d_s(\mathbf{R}_h - \mathbf{C}_l) = \min_l w_s(\mathbf{E}_l) \quad (6.6.9)$$

也就是寻找一个与接收序列的量化序列 \mathbf{R}_h 有最小软距离的码字 \mathbf{C}_l ，或者寻找软重量最小的错误图样。称完成式(6.6.9)功能的译码器为完备信道测量信息译码器，简称 Chase 译码器。

Chase 译码器的基本工作原理如图 6-19 所示。在 $GF(2)$ 上的 n 维线性空间中， n 长的接收序列 \mathbf{R}_h 和码字 $\mathbf{C}_A, \mathbf{C}_B, \mathbf{C}_C$ 和 \mathbf{C}_D 之间的硬距离如图中所示，每一码字用一个半径为 $\lfloor (d_h - 1)/2 \rfloor$ 的球包围着，该码字位于码球的中心。若 \mathbf{R}_h 在 \mathbf{C}_A 的球内，则就把 \mathbf{R}_h 译成 \mathbf{C}_A ，显然这种译码是唯一的（由于各个码球互不相交），也就是说这种译码方法能唯一地确定错误图样 $\mathbf{E}_A = \mathbf{R}_h - \mathbf{C}_A$ 。而利用信道测量信息（码元可信度）的目的，是寻找相对少量的错误图样集合，而不仅仅是一个错误图样，并且在此集合中选择一个有最小软重量的错误图样，作为译码器所确定的错误图样。由此可见，利用码元可信度信息的 Chase 译码器，首先必须确定一组错误图样集合，然后挑选一个有最小软重量的图样，作为译码器最后所确定的错误图样。

由此可得 Chase 译码器的工作过程如下：

(1) 从解调器输出的接收序列 $\tilde{\mathbf{R}}$ 中，得到一个硬判决序列 \mathbf{R}_h 和可信序列 α 。

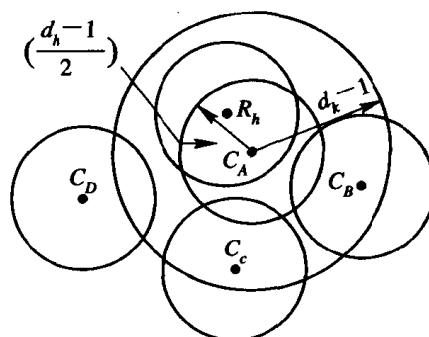


图 6-19 Chase 译码器的原理图

(2) 选择一个试探序列 T , 把它与 R_h 相加, 得到一个新的序列 $R'_h = R_h + T$ 。

(3) 把 R'_h 送入硬判决译码器进行译码, 得到一个错误图样 $E' = R'_h + C_i$, 这里 C_i 是硬判决译码器所译得的一个候选码字。此时译码器认为存在于 R_h 中候选错误图样是 $E_t = T + E'$, 因为 $E = R_h - C_i = R'_h + T - C_i = E' + C_i + T - C_i = E' + T$ 。

(4) 重复步骤(2)、(3), 直到根据错误图样集合所确定的试探序列全部试探完毕, 得到一组候选错误图样集合 $\{E_t\}$ 。

(5) 译码器在 $\{E_t\}$ 集合中, 挑选一个有最小软重量的错误图样 E_{mi} , 作为译码器最后所确定的错误图样。最后, 译码器输出已译码字 $\hat{C}_i = R_h - E_{mi}$ 。

图 6-20 给出了上述的 Chase 译码器的工作流程图。由图可见, 决定该译码器复杂性的是试探序列集合的大小和硬判决译码器。由于硬判决译码器仅完成式(6.6.7)确定的纠错能力, 因此有可能对所有试探序列, 译码器均得不到合适的错误图样, 这时如果不挑选与重量最轻的错误图样相对应的候选码字作为译码器输出的话, 译码器就把接收到的序列 R_h 直接送给用户, 并指出它有不可纠正的错误。但是, 目前通用的 Chase 译码器, 都是挑一个与重量最轻的错误图样相应的候选码字, 作为译码器的输出码字。

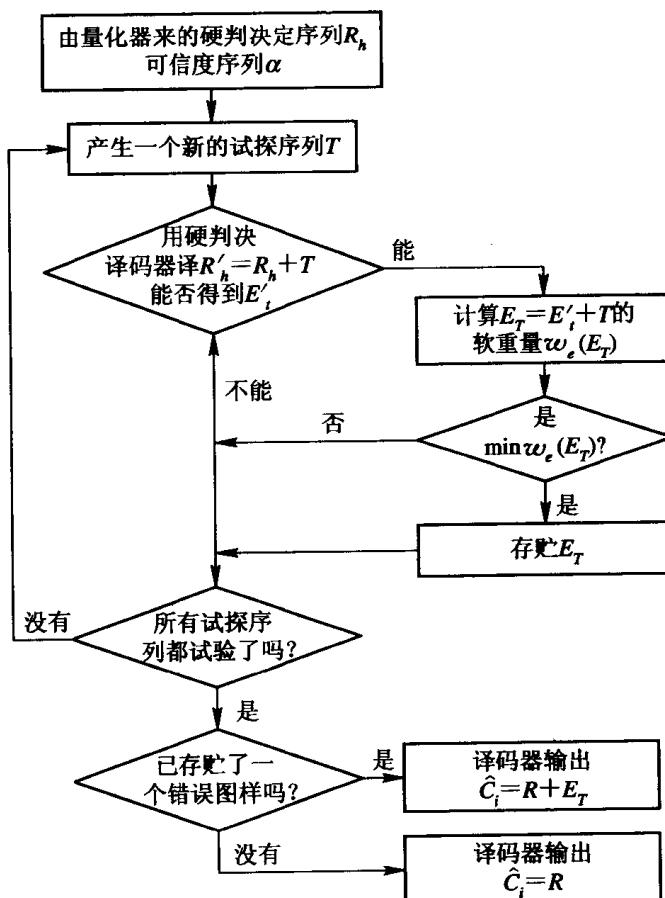


图 6-20 Chase 译码器的工作流程图

2. 试探序列集合的大小及相应算法 试探序列选择的原则, 应该包括能产生以 R_h 为中心, $d_h - 1$ 为半径的球内的所有 n 重。因此, 原则上讲, 应该选择与 R_h 距离为 $d_h - 1$ 的

所有错误图样。当然实际上根据不同的选择方法，试探序列的数目可大大减少。可以有以下三种方法：

(1) Chase 算法1(CHS1)。该译码器产生 $\binom{n}{\lfloor d_h/2 \rfloor} = C_n^{d_h/2}$ 个，所有重量为 $\lfloor d_h/2 \rfloor$ 的试探序列。因为试探序列 \mathbf{T} 中有 $\lfloor d_h/2 \rfloor$ 个 1，如果 \mathbf{R}_h 中有等于或小于 $d_h - 1$ 个错误，则 \mathbf{R}'_h 中的错误数目等于或小于 $\lfloor (d_h - 1)/2 \rfloor$ ，这正是硬判决译码器所能纠正的。由于 $C_n^{d_h/2}$ 个试探序列所产生的 \mathbf{R}'_h ，包括了所有与 \mathbf{R}_h 距离为 $d_h - 1$ 的 n 重，因此，这种算法能提供最好的译码性能，但译码器也最复杂。故这种算法仅适用于短码和 d_h 较小的码。

(2) Chase 算法2(CHS 2)。为了减少 CHS 1 算法的试探序列数目，仅考虑 $\lfloor d_h/2 \rfloor$ 个最少可信度码元位置的错误情况，从而使试探错误数目大为减少，由于 \mathbf{R}_h 中含有等于或小于 $d_h - 1$ 个错误，其中 $\lfloor d_h/2 \rfloor$ 个最可能发生在可信度最小的码元位上，若试探序列 \mathbf{T} 在这些位置取 1，则 \mathbf{R}'_h 中的错误数目减少到等于或小于 $\lfloor (d_h - 1)/2 \rfloor$ 个，所以硬判决译码器可以纠正。

在 $\lfloor d_h/2 \rfloor$ 个可信度最小的码元位上，所有可能的错误图样组合有 $2^{\lfloor d_h/2 \rfloor}$ 个，故译码器必须产生 $2^{\lfloor d_h/2 \rfloor}$ 个试探序列，由于一般情况下 $n \gg d_h$ ，因而 $2^{\lfloor d_h/2 \rfloor} < C_n^{d_h/2}$ ，所以 CHS2 算法的复杂性比 CHS1 要低，特别当 n 很大时更是如此，但其性能比 CHS 1 稍差。

(3) Chase 算法 3(CHS 3)。此算法与 CHS 2 算法基本相同，但试探序列的数目是 $\lfloor (d_h/2) + 1 \rfloor$ ，而不是 $2^{\lfloor d_h/2 \rfloor}$ 。每一个试探序列，在 i 个可信度最低的码元位置上取 1。若 d_h 为偶数，则 i 取 $0, 1, 3, \dots, d_h - 1$ 。若 d_h 为奇数，则 i 取 $0, 2, 4, \dots, d_h - 1$ 。这里 $i=0$ ，相应于全 0 试探序列。

如 $d_h=4$ ，则 i 取 $0, 1, 3$ 。即译码器产生三个试探序列 $\mathbf{T}_0, \mathbf{T}_1, \mathbf{T}_3$ 。 \mathbf{T}_0 的重量为 0， \mathbf{T}_1 量为 1， \mathbf{T}_3 的重量为 3。 \mathbf{T}_3 中的 3 个 1，对应于 \mathbf{R}_h 序列中 3 个可信度最小的码元位置。

与前两种算法相比，此算法所需的试探序列数目最少，计算量和复杂性也最小，且有相似的性能，所以在实际上 CHS 3 算法更有希望。特别当码的最小距离 d_h 很大时，此算法的优点更为突出，因为它的试探序列数目随 d_h 线性增加，而不像前两种算法那样指数增长。

从上述三种方法不难看出，Chase 算法的基本思想与 GMD 基本相同，本质上都是最近邻域译码，正如图 6-19 中所表示的，Chase 的 3 种算法代表了 3 种产生最近邻域子集的方法。定理 6.5.1 指出，任何一个 $[n, k, d_h]$ 线性分组码，一定能纠正软重量小于 $(1/2)(Q - 1)d_h$ 的错误图样。因此，若接收序列 \mathbf{R}_h 中包含有小于或等于 $d_h - 1$ 个硬判决错误时，若用重量为 $\lfloor d_h/2 \rfloor$ 的试探序列 \mathbf{T} 与之模 2 加，就有极大可能消除 $\lfloor d_h/2 \rfloor$ 个错误，而剩下的 $\lfloor (d_h - 1)/2 \rfloor$ 个错误，用硬判决译码器把它纠正，从而纠正了 $d_h - 1$ 个错误。

例 6.8 设利用 $[15, 7, 5]$ 码。发送的是全 0 序列，接收序列的量化序列 $\mathbf{R}_h = (000000044540000)$ ，利用 CHS 2 算法进行译码。

码的 $d_h=5$ ，故译码器需产生 $2^{\lfloor 5/2 \rfloor}=4$ 个试探序列： $\mathbf{T}_0, \mathbf{T}_1, \mathbf{T}_2$ 和 \mathbf{T}_3 。 \mathbf{T}_0 是全 0 序列， \mathbf{T}_1 是重量为 1 的序列，它在可信度最低的码元位置 r_{8s} 取值为 1，其它为 0，即 $\mathbf{T}_1 = (000000010000000)$ ， \mathbf{T}_2 在另一最低可信度码元位 r_{9s} 取 1，即 $\mathbf{T}_2 = (000000001000000)$ ， \mathbf{T}_3 在 r_{8s}, r_{9s} 位上取值为 1，即 $\mathbf{T}_3 = (000000011000000)$ 。相应的 4 个试探图样分别为：

$$\mathbf{R}'_0 = \mathbf{R}_h + \mathbf{T}_0 = (000000011110000)$$

$$\mathbf{R}'_1 = \mathbf{R}_h + \mathbf{T}_1 = (000000001110000)$$

$$\mathbf{R}'_2 = \mathbf{R}_h + \mathbf{T}_2 = (000000010110000)$$

$$\mathbf{R}'_3 = \mathbf{R}_h + \mathbf{T}_3 = (000000000110000)$$

上述 4 个试探图样送入硬判决译码器后，只有 \mathbf{R}'_3 译成全 0 码字 \mathbf{R}_3 ，其它 3 个将错译成重量为 5 的其它码字。而 $d_s(\mathbf{R}_3, \mathbf{R}_s) = 17$ ，但其它 3 个已译码序列与 \mathbf{R}_s 的软距离 $d_s(\mathbf{R}_{0s}, \mathbf{R}_s), d_s(\mathbf{R}_{1s}, \mathbf{R}_s)$ 和 $d_s(\mathbf{R}_{2s}, \mathbf{R}_s)$ ，肯定都大于 17。其中最小的一个是 $d_s(\mathbf{R}_{0s}, \mathbf{R}_s) = 7 + (7 - 4) + (7 - 4) + (7 - 5) + (7 - 4) = 18$ 。因而译码器最终送出全 0 码字，纠正了 \mathbf{R}_h 中的 4 个错误。

若用 CHS 3 算法译码，则产生 $\lfloor (d_h/2) + 1 \rfloor = 3$ 个试探序列： \mathbf{T}_0 是全 0 序列， $\mathbf{T}_1 = (000000011000000)$ ， $\mathbf{T}_2 = (000000011110000)$ 。结果 $\mathbf{R}'_2 = \mathbf{R}_h + \mathbf{T}_2$ 送入硬译码器后得到全 0 序列，译码器正确译码。 ■

后面的图 6-21，给出了用 Chase 的 3 种算法译 [24, 12, 8] 码时，用计算机模拟的性能曲线 (AWGN 信道)。由该图看到，CHS 1、CHS 2 算法的性能很接近，CHS 3 算法的性能稍差。由于 CHS 2 算法的试探数目大大少于 (当 n 和 d 较大时) CHS 1 算法的数目，且二者的译码错误概率很接近，因此，目前 CHS 2 算法在实际中用得较广泛。

分析表明，Chase 算法不仅能纠正软重量小于 $(1/2)(Q - 1)d_h$ 的错误图样，而且也能纠正软重量大于该值的许多错误图样，但另一种较简单的 WED 算法却不能做到这点，所以 Chase 算法比 WED 算法更接近于最大似然译码。

虽然 Chase 算法的性能较好，译码器也不太复杂，但在某些场合如利用微机实现时，仍嫌试探次数太多，以致微机运算速度赶不上译码速度的要求，限制了整个系统处理数据的速率。因此，如何减少 Chase 算法特别是 CHS 2 算法的试探次数，日益引起人们的注意。1981 年哈斯盖特 (Hackett)，在利用 CHS 2 算法译 [24, 12, 8] 码时，提出了一种修正算法，能使试探次数减少一半^[20]。但遗憾的是，这种修正算法仅适用于最小距离为偶数的码。另一种加快 Chase 译码速度的方法，是在进行试探译码过程中，建立一个门限标准^[11]，当达到该标准时结束试探，不再继续进行译码。但是，如何建立门限以及门限与译码性能的关系在文献[11]中并没有讨论。最近作者^[21]提出了与 GMD 算法结合的广义门限 Chase 算法，利用类似于式(6.6.5)的判别式作为译码试探是否结束的标准，从而加快了译码速度，且对性能并没有多大影响。除此以外，还有不少作者提出了其它修正算法，有兴趣的读者请参阅文献[22~25]。

3. Chase 译码算法的性能 下面介绍 Chase 译码算法在 AWGN 信道和瑞利分布衰落信道中的性能，但不作推导。若需了解推导过程，请参阅文献[11]。

设利用 PSK 信号，通过 AWGN 信道，接收端应用相干解调，则 Chase 算法和相关译码的错组率 p_{eas} 为

$$n_d Q(\sqrt{2\gamma_b R})^{d_h} (1 - Q(\sqrt{2\gamma_b R})) < p_{eas} < \binom{n}{d_h} Q(\sqrt{2d_h \gamma_b R}) \quad (6.6.10)$$

式中， n_d 是 $[n, k, d_h]$ 码中重量为 d_h 的码字数， R 是码率， $\gamma_b = (E_b/N_0)$ 为信噪比， $Q(x)$ 是由式(6.5.4)确定的 Q 函数。

当信噪比很大时，式(6.6.10)中的上、下限趋向相同，为

$$P_{eas} \approx K \exp(-\gamma_b d_h R) \quad (6.6.11)$$

式中

$$K = \exp(\gamma_b o(\gamma_b))$$

是 γ_b 的弱函数, $o(\gamma_b)$ 是小 o 函数, 随 γ_b 的增加趋近 0。

对硬判决译码, 错组率

$$P_{ech} \approx K \exp(-\gamma_b R(t+1)) \quad (6.6.12)$$

该式与式(6.5.22)在本质上相同, 只是表示方式不一样。

在瑞利衰落信道中, Chase 算法和相关译码的错组率

$$n_d \left(\frac{1-\mu}{2} \right)^{d_h} \left(\frac{1+\mu}{2} \right)^{n-d_h} < P_{echR} < \frac{1}{2} \binom{n}{d_h} \left[1 - \mu \sum_{i=0}^{d_h-1} \binom{2i}{i} \left(\frac{1-\mu^2}{4} \right)^i \right] \quad (6.6.13)$$

式中

$$\mu = \frac{\gamma_b R}{\gamma_b R + 1}$$

当信噪比增加时, 上、下限趋近相同。

$$P_{echR} \approx K' \exp(-\ln \gamma_b d_h) = K' (1/\gamma_b)^{d_h} \quad (6.6.14)$$

式中

$$K' = \exp[-o(\gamma_b) \ln(\gamma_b)]$$

对硬判决译码而言, 错组率

$$P_{echR} \approx K (1/\gamma_b)^{t+1} \quad (6.6.15)$$

而误码率

$$P_{eR} = \frac{1}{2} (1 - \mu) \quad (6.6.16)$$

把式(6.6.11)与式(6.6.12)及式(6.6.14)与式(6.6.15)进行比较, 可知在高信噪比下, 用软判决译码比硬判决译码, 其纠错能力几乎增加了一倍, 这与推论 6.5.1 的结果相同。在 AWGN 信道中, 这相当于系数因子变化 $2 \approx d_h/(t+1)$, 故信噪比改进约 3 dB。而对瑞利衰落信道而言, 其信噪比改善程度要大于 3 dB。图 6-21 和图 6-22, 分别给出了在这两种信道中, 用计算机模拟[24, 12, 8]码的硬、软判决译码时的性能曲线。

由图 6-21 可以看出, 在 AWGN 信道中, 当信道误码率 $p_e = 10^{-5}$ 时, 硬判决译码大约提供了 2 dB 的编码增益, 而相关译码或 Chase 算法, 大约提供了 4 dB, 其中软判决增益为 2 dB。

从图 6-22 的曲线可以看出, 在瑞利衰落信道中, Chase 的 3 种译码算法的性能曲线比较分散, 但无论编码增益还是软判决增益, 均比 AWGN 信道中的情况要大得多。因此, 在散射和高频信道中, 利用软判决译码能取得较好的结果^[28]。

以上讨论的软判决译码方法, 大都是针对二进制码, 原则上讲, 这些方法也可推广到多进制码如 RS 码, 但实际上会遇到一些困难。目前, RS 码的软判决译码有两类: 一类是基于删除译码基础上的译码, 其特点是删除译码次数与量化比特数有关; 另一类是将 $GF(2^m)$ 上的 RS 码转化成二元码的 Chase 译码问题, 或者把 Chase 方法推广到多进制情况。有关 RS 码的这些软判决译码方法请参阅文献[33]。

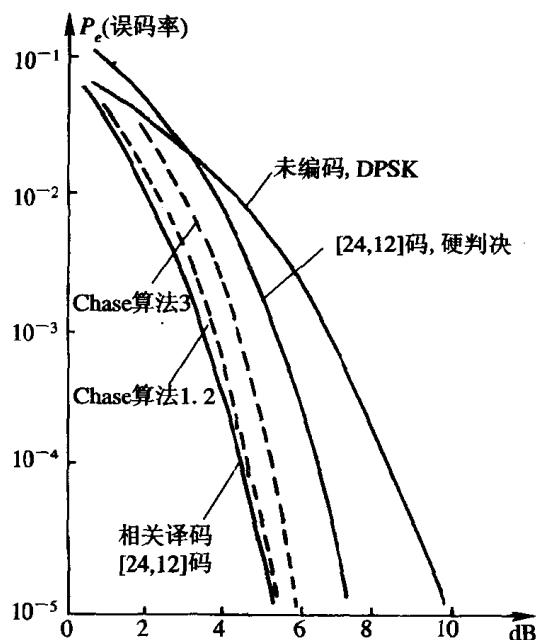


图 6-21 AWGN 信道中 $[24, 12, 8]$ 码的性能

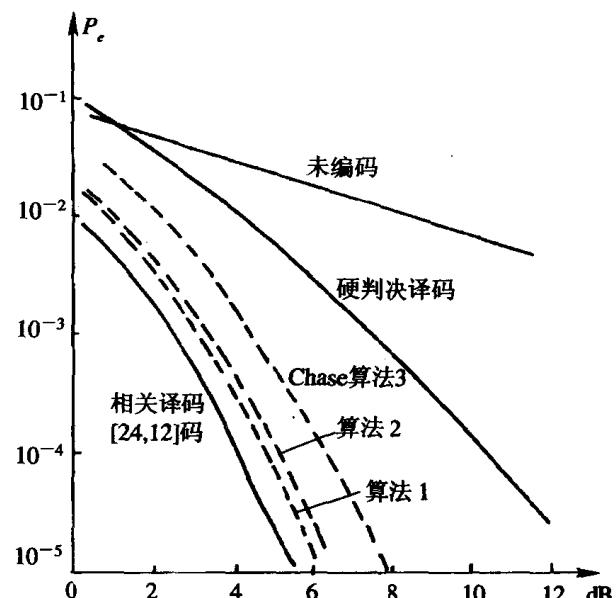


图 6-22 瑞利衰落信道中 $[24, 12, 8]$ 码的性能

习题

- 设计一个由 $g(x) = x^4 + x^3 + 1$ 生成的 $[15, 11]$ 循环汉明码编译码器。
- 构造由第 1 题的 $[15, 11]$ 码缩短三位的 $[11, 8]$ 码译码器。
- 证明 $g(x) = 1 + x^2 + x^4 + x^6 + x^7 + x^{10}$ 生成一个 $[21, 11]$ 循环码，作出此码的伴随式计算电路，令 $R(x) = 1 + x^5 + x^{17}$ 是接收多项式，计算 $R(x)$ 的伴随式，列出 $R(x)$ 的每一接收数据移入伴随式计算电路后，伴随式寄存器中的内容。
- 构造 $GF(2)$ 上以 α, α^3 为根的循环码；这里 $\alpha \in GF(2^4)$ 中的本原元。求出该码的生成多项式 $g(x)$ 以及码长 n 和 k 。设计出该码的编码电路，求计算伴随式的电路。
- 构造 $[15, 5, 7]$ 码的译码器，它的生成多项式 $g(x) = 1 + x + x^2 + x^4 + x^5 + x^8 + x^{10}$ ，该码能纠正 3 个错误。设用简单的捕错译码器译码。
 - 证明所有 2 个错误能被捕获；
 - 能捕获所有 3 个错误的图样吗？若不能，则有多少种 3 个错误图样不能被捕获；
 - 作出该码的简单捕错译码器。
- 作出第 5 题中 $[15, 5, 7]$ 码的修正捕错译码器，叙述其译码过程。
- 已知 $[17, 9, 5]$ QR 码的生成多项式 $g(x) = x^8 + x^7 + x^6 + x^4 + x^2 + x + 1$ ，求出该码利用修正捕错译码的 $\{Q_j(x)\}$ ，并作出捕错译码器，说明译码过程。
- 考虑 $[31, 5]$ 极长码，它的一致校验多项式是 $h(x) = 1 + x^2 + x^5$ ，求正交于 x^{30} 码元位的全部正交多项式。画出该码的 I 型和 II 型大数逻辑译码器。
- 作出表 6-4 中的 $[21, 11, 6]$ 码的 I 型大数逻辑译码器。

10. 构造 $p=3, d=3$ 复数旋转码，画出编译码电路图。
11. 第 5 题中的 [15, 7, 5] 码是一步大数逻辑可译码，画出该码的 I 型和 II 型大数逻辑译码器。
12. 考虑 [11, 6] 线性码，它的一致校验矩阵是
- $$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$
- (a) 证明该码的距离恰好为 4。
(b) 令 $\mathbf{E} = (e_0, e_1, e_2, \dots, e_{10})$ 是错误矢量，求出以此错误矢量表示的伴随式码元。
(c) 对 $i=5, 6, 7, 8, 9, 10$ ，求出正交于每一消息数据 e_i 的全部可能的一致校验和。
(d) 该码是一步完备可正交码吗？
13. 作出 [15, 7, 5] 码的 CHS 2 译码器，已知接收 $\tilde{\mathbf{R}} = (000007004400020)$ ，求发送码字。
14. 作出利用 CHS 2 算法译 [17, 9, 5] 码的软判决译码流程图。

参 考 文 献

- [1] W. W. Peterson and E. J. Weldon, Jr., Error - Correcting Codes, 2nd ed., MIT Press, Cambridge, MA, 1971.
- [2] S. Lin and D. J. Costello, Jr., Error Control Coding—Fundamentals and Applications, 1984.
中译本：王育民、王新梅译，《差错控制码：基础和应用》，人民邮电出版社，1986。
- [3] V. K. Wei, “An error - trapping decoder for nonbinary cyclic codes”, IEEE Trans. IT -30, No. 3, pp. 538—541, 1984.
- [4] 于康友：求取一类循环码的覆盖多项式集合的一种方法，《通信学报》(待发表)。
- [5] 斯蕃：新型复数旋转码的特性分析，《通信学报》，No. 2, pp. 62—69, 1986.
- [6] 斯蕃：《组合设计与编码》，西南交通大学出版社，1990.
- [7] J. L. Massey, Threshold Decoding, MIT Press, Cambridge, MA, 1963.
- [8] C. R. P. Hartman, C. P. Rudolph, An optimum symbol - by - symbol decoding rule for linear codes, IEEE Trans. on IT, pp. 514—517, Sept. 1976.
- [9] F. J. Weldon, Jr., “Decoding binary block codes on Q -ary output channels”, IEEE Trans. on IT, pp. 713—718, Nov. 1971.
- [10] G. D. Forney, Concatenated Codes, MIT Press, Cambridge, MA, 1966.
- [11] D. Chase, “A class of algorithms for decoding block codes with channel measurement information”, IEEE Trans. on IT, pp. 170—182, Jan. 1972.
- [12] A. J. Viterbi, J. K. Omura, Principles of Digital Communication and Coding, McGraw - Hill, New York, 1979.
- [13] G. C. Clark, Jr., J. B. Cain, Error - Correcting Coding for Digital Communications, Plenum Press, New York, 1981.
- [14] A. Tanaka, K. Furusawa, S. Kaneko, “A novel approach to soft decoding of threshold decodable codes”, IEEE Trans. on IT, pp. 244—246, March 1980.

- [15] H. Greenberger, "An iterative algorithm for decoding block codes transmitted over a memoryless channel", JPL OSN Progress Report 42-47, Jet Propulsion Laboratory, July and August, 1978.
- [16] L. D. Rudolph, C. R. P. Hartman, "Algebraic analog decoding of linear binary codes", IEEE Trans. on IT, pp. 430-440, July 1979.
- [17] C. R. P. Hartman, L. D. Rudolph, R. G. Mehrotra, "Asymptotic performance of optimum bit - by - bit decoding for the white Gaussian channel", IEEE Trans. on IT, pp. 520-522, July 1977.
- [18] C. C. H. Yu, D. J. Costello, Jr., "Generalized minimum distance decoding algorithms for Q - ary output channels", IEEE Trans. on IT, pp. 238-244, March 1980.
- [19] T. Y. Hwang, S. U. Guan, "Generalized minimum distance decoding on majority logic decodable code", IEEE Trans. on IT, pp. 790-792, Sep. 1982.
- [20] C. M. Hackett, "An efficient algorithm for soft - decision decoding of the (24, 12) extended Golay code", IEEE Trans. on IT, pp. 909-911, June 1981.
- [21] 王新梅: 广义门限契斯算法,《电子科学学刊》, No. 6, 1986.
- [22] R. W. Boyd, Harris Corporation, Government Systems Group, Melbourne, Florida, May 1979.
- [23] 平川秀治等, ブック誤ソ訂正符号の軟判定復号 PWI“ソラム”電子通讯学会通信方式研究会資料 OSFF-11, pp. 25-32, 1977.
- [24] E. R. Berlekamp, "The construction of fast, high-rate, soft decision decoders", IEEE Trans. on IT, pp. 372-377, May 1983.
- [25] N. N. Tendolkar, Asymptotically Optimum Soft Decision Decoding, Dissertation, August 1983.
- [26] W. Stieritz, J. K. Wolf, "Improvement in two soft decision maximum likelihood decoding algorithms for linear", NTC[2], pp. 26.7.1-26.7.6, 1977.
- [27] J. K. Wolf, "Efficient maximum likelihood decoding of linear coder using a trellies", IEEE Trans. on IT, pp. 76-80, Jan. 1978.
- [28] P. G. Farrell and R. M. F. Goodman, "Soft decision error control for HF data transmission", IEEP, Part F, Vol. 127, No. 5, pp. 389-400, 1980.
- [29] N. Montague, "An application of error protection to a vocoded speech channel", IERE Conference on Digital Processing of Signals in Communication, Sept. 1977.
- [30] R. M. F. Goodman, A. D. Green, "Microprocessor controlled soft - decision decoding of error - correcting block codes", IERE Conference on Processing, Sept. 1977.
- [31] 王新梅: 软判决译码综述,《通信学报》, 1985. No. 6.
- [32] 王新梅:《纠错码与差错控制》,人民邮电出版社, 1989.
- [33] 马建华: 分组码的软判决译码,《西安电子科技大学博士论文》, 1990.

第七章 BCH 码与 Goppa 码

自 1950 年汉明发表了纠正单个随机错误的码以来，几乎过了近 10 年时间，才于 1959 年由霍昆格姆(Hocquenghem)，1960 年由博斯(Bose)和雷-查德胡里(Ray - Chaudhuri)分别提出了纠正多个随机错误的循环码——BCH 码的构造方法。BCH 码是迄今为止所发现的一类很好的线性纠错码类。它的纠错能力很强，特别在短和中等码长下，其性能很接近于理论值，并且构造方便，编码简单。特别是它具有严格的代数结构，因此它在编码理论中起着重要作用。BCH 码是迄今为止研究得最为详尽，分析得最为透彻，取得的成果也是最多的码类之一。

1960 年彼德逊(Peterson)从理论上解决了二进制 BCH 码的译码算法，奠定了 BCH 码译码的理论基础。稍后，格林斯坦(Gorenstein)和齐勒尔(Zierler)把它推广到多进制。1966 年伯利坎普(Berlekamp)利用迭代算法译 BCH 码，从而大大加快了译码速度，从实际上解决了 BCH 码的译码问题。由于 BCH 码性能优良，结构简单，编译码设备也不太复杂，使得它在实际使用中受到工程技术人员的欢迎，是目前用得最为广泛的码类之一。

1966 年斯利华斯特华(Srivastava)首先利用有理分式表示码字，并构造了一类有理分式码。在此基础上，70 年代初戈帕(Goppa)较系统地构造了另一类有理分式码——Goppa 码。Goppa 码的最主要优点是，对某些 Goppa 码而言，能达到 Shannon 编码定理所给出的性能。正由于它的良好性能，因此自该码提出以后，许多学者对它的性质、编码方法进行了深入研究，探讨了它与其它码类的关系，推动了编码理论的发展，提出并构造了一系列广义码，如交替码、GBCH 码等。特别是 80 年代初，戈帕由 Goppa 码引出了代数几何码，麦克依理斯(MacEliece)用 Goppa 码构造了一类公开密钥系统，使得 Goppa 码日益引起了人们的极大兴趣。因此，研究 Goppa 码有非常重要的理论意义。

本章首先讨论 BCH 码的各种编译码方法，然后介绍经典 Goppa 码及其广义码以及与 BCH 码的关系，至于现代 Goppa 码和代数几何码，将在下一章介绍。

§ 7.1 BCH 码的描述及其距离限

一、BCH 码的定义

BCH 码是纠正多个随机错误的循环码，可以用生成多项式 $g(x)$ 的根描述。

定义 7.1.1 给定任一有限域 $GF(q)$ 及其扩域 $GF(q^m)$ ，其中 q 是素数或素数的幂， m 为某一正整数。若码元取自 $GF(q)$ 上的一循环码，它的生成多项式 $g(x)$ 的根集合 R 中含有以下 $\delta-1$ 个连续根：

$$R \supseteq \{\alpha^{m_0}, \alpha^{m_0+1}, \dots, \alpha^{m_0+\delta-2}\}$$

时，则由 $g(x)$ 生成的循环码称为 q 进制 BCH 码。

其中, $\alpha \in GF(q^m)$ 是域中的 n 级元素, $\alpha^{m_0+i} \in GF(q^m)$ ($0 \leq i \leq \delta - 2$), m_0 是任意整数, 但对于最常见的情况 $m_0=0$ 或 1 。若 $m_0=1$, 则称这类 BCH 码为**狭义 BCH 码**。

设 $m_i(x)$ 和 e_i 分别是 α^{m_0+i} ($i=0, 1, \dots, \delta-2$) 元素的最小多项式和级, 则由式(5.2.3) 和式(5.2.4) 可知, BCH 码的生成多项式和码长分别是:

$$g(x) = \text{LCM}(m_0(x), m_1(x), \dots, m_{\delta-2}(x)) \quad (7.1.1)$$

$$n = \text{LCM}(e_0, e_1, \dots, e_{\delta-2}) \quad (7.1.2)$$

如果生成多项式 $g(x)$ 的根中, 有一个 $GF(q^m)$ 中的本原域元素, 则 $n=q^m-1$, 称这种码长 $n=q^m-1$ 的 BCH 码为**本原 BCH 码**; 否则, 称为**非本原 BCH 码**。 $GF(q^m)$ 中元素的级一定是 q^m-1 的因子, 所以非本原 BCH 码的码长也一定是 q^m-1 的因子。

二、BCH 码的距离限

一个码的纠错能力, 完全由它的最小汉明距离 d_m 决定, 而 BCH 码的 d_m 则完全由 $g(x)$ 的根决定, 正由于 BCH 码的根与 d_m 有密切关系, 研究这些关系不仅有重要的理论意义, 而且有很大的实际意义。自 1959 年到 1960 年发现 BCH 码至今, 研究 d_m 与 $g(x)$ 的根的关系一直是一个非常引人注目的研究课题, 到目前为止有 4 个限: BCH 限、HT 限、鲁斯(Roos)限和 LW 限, 论述了这种关系。下面介绍的就是 $GF(q)$ 上 BCH 码或循环码的这 4 个限。

定理 7.1.1(BCH 限) BCH 码的最小距离 d_{BCH} 至少为 δ 。

BCH 码限的证明有两种方法, 一种是从码的校验矩阵 H 出发, 另一种是从码的 DFT 或 MS 多项式出发。下面分别证明之。

证明

(1) 设 $C(x) = q(x)g(x)$ 是 BCH 码的任一码字, 则它必以 $\alpha^{m_0}, \alpha^{m_0+1}, \dots, \alpha^{m_0+\delta-2}$ 等 $\delta-1$ 个连续元素为根, 即若码字

$$C(x) = c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \dots + c_1x + c_0$$

则

$$\begin{aligned} C(\alpha^i) &= c_{n-1}(\alpha^i)^{n-1} + c_{n-2}(\alpha^i)^{n-2} + \dots + c_1(\alpha^i) + c_0 = 0 \\ i &= m_0, m_0+1, \dots, m_0+\delta-2 \end{aligned}$$

因此由式(5.2.2)可知 BCH 码的校验矩阵

$$H = \begin{bmatrix} (\alpha^{m_0})^{n-1} & (\alpha^{m_0})^{n-2} & \cdots & \alpha^{m_0} & 1 \\ (\alpha^{m_0+1})^{n-1} & (\alpha^{m_0+1})^{n-2} & \cdots & \alpha^{m_0+1} & 1 \\ \vdots & \vdots & & \vdots & \vdots \\ (\alpha^{m_0+\delta-2})^{n-1} & (\alpha^{m_0+\delta-2})^{n-2} & \cdots & \alpha^{m_0+\delta-2} & 1 \end{bmatrix} \quad (7.1.3)$$

要证明满足此 H 的码的距离至少为 δ , 由定理 3.3.1 可知, 只要证明 H 矩阵中任意 $\delta-1$ 列线性无关即可。现在从此 H 矩阵中任意取出 $\delta-1$ 列, 并计算它的行列式之值

$$D = \begin{bmatrix} (\alpha^{m_0})^{j_1} & (\alpha^{m_0})^{j_2} & \cdots & (\alpha^{m_0})^{j_{\delta-1}} \\ (\alpha^{m_0+1})^{j_1} & (\alpha^{m_0+1})^{j_2} & \cdots & (\alpha^{m_0+1})^{j_{\delta-1}} \\ \vdots & \vdots & & \vdots \\ (\alpha^{m_0+\delta-2})^{j_1} & (\alpha^{m_0+\delta-2})^{j_2} & \cdots & (\alpha^{m_0+\delta-2})^{j_{\delta-1}} \end{bmatrix}$$

$$= \alpha^{m_0(j_1+j_2+\cdots+j_{\delta-1})} \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha^{j_1} & \alpha^{j_2} & \cdots & \alpha^{j_{\delta-1}} \\ (\alpha^{j_1})^2 & (\alpha^{j_2})^2 & \cdots & (\alpha^{j_{\delta-1}})^2 \\ \vdots & \vdots & & \vdots \\ (\alpha^{j_1})^{\delta-2} & (\alpha^{j_2})^{\delta-2} & \cdots & (\alpha^{j_{\delta-1}})^{\delta-2} \end{bmatrix} \quad (7.1.4)$$

该式的行列式就是以下形式的范得蒙行列式：

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_s \\ x_1^2 & x_2^2 & \cdots & x_s^2 \\ \vdots & \vdots & & \vdots \\ x_1^{s-1} & x_2^{s-1} & \cdots & x_s^{s-1} \end{bmatrix} = \prod_{i>j}^{s-1} \prod_{j=1}^{s-2} (x_i - x_j) \quad (7.1.5)$$

由于生成多项式 $g(x)$ 没有重根，且 α 是 n 级元素，则对任何 $k \neq r$, $\alpha^{j_k} \neq \alpha^{j_r}$ (k, r 均不大于 n)。由此可知，式(7.1.4)行列式的值不为 0，所以式(7.1.3)的 H 矩阵任意 $\delta-1$ 列线性无关，码的最小距离 $d_{\text{BCH}} \geq \delta$ 。

(2) BCH 码的码字 $C(x) = g(x)m(x)$, $g(x)$ 含有 $J-1$ 个连续根： $\alpha^{m_0}, \alpha^{m_0+1}, \dots, \alpha^{m_0+\delta-2}$ ，因此， $C(x)$ 也有 $\delta-1$ 个连续根。由定理 5.8.3 可知， $C(x)$ 的 MS 多项式 $A(Z) = A_0 + A_1Z + \cdots + A_{n-1}Z^{n-1}$ 中的系数 $A_{m_0} = A_{m_0+1} = \cdots = A_{m_0+\delta-2} = 0$ 。如果把 $A(Z)$ 的系数 A_0, A_1, \dots, A_{n-1} 看成是一个 $\text{GF}(q^m)$ 上的序列 (A^n) ，则 $(A^n) = (\Delta^j O^{\delta-1} \Delta^+ \Delta^{n-j-\delta})$ ，由引理 5.9.5 和 5.9.3 可知，半无限序列 $(A^n)^\infty$ 的线性复杂度 $L((A^n)^\infty) = (\Delta^j O^{\delta-1} \Delta^+ \Delta^\infty) \geq \delta$ 。由 Blahut 定理可知， $C(x)$ 的系数 $(c_0, c_1, \dots, c_{n-1})$ 组成的序列，也就是 BCH 码的码字，有汉明重量 $w(C^n) = L((A^n)^\infty) \geq \delta$ 。 ■

推论 7.1.1 若 BCH 码生成多项式 $g(x)$ 的根集 $R \subseteq \{\alpha^{m_0+ic} \mid i=0, 1, \dots, \delta-2\}$ ，且 $(n, c)=1$ ，则码的 $d_{\text{BCH}} \geq \delta$ 。

证明 令 $\alpha=\beta$, $(c, n)=1$ ，所以 β 也是 n 级单位元。 $\alpha^{m_0}=\beta^j$ ，由此可得 $g(x)$ 的根集为 $\{\beta^{j+i} \mid i=0, 1, \dots, \delta-2\}$ ，这就归结为定理 7.1.1 的情况。 ■

称 $\{\alpha^{m_0+ic} \mid i=0, 1, \dots, \delta-2\}$ 为 $g(x)$ 的广义连续根集。

定理 7.1.1 所指出的 BCH 码的距离 δ ，称为码的设计距离(也称 BCH 限)。但事实上很多码的实际最小距离 d 往往超过 δ 。例如，由第五章可知，[23, 12]Golay 码以 $\beta, \beta^2, \beta^3, \beta^4, \beta^6, \beta^8, \beta^9, \beta^{12}, \beta^{13}, \beta^{16}, \beta^{18}$ 为根，它是一个非本原 BCH 码， $23 \mid (2^{11}-1)$ ， $\beta = \alpha^{89} \in \text{GF}(2^{11})$ ， α 是 $\text{GF}(2^{11})$ 的本原域元素。最大的连续根数为 4，由 BCH 码限可知它的设计距离 $\delta=5$ ，但事实上，由于它有 3 个连续根集： $\{\beta, \beta^2, \beta^3, \beta^4\}$, $\{\beta^8, \beta^9\}$, $\{\beta^{12}, \beta^{13}\}$ ，因而实际最小距离 $d=7$ 。这说明仅考虑一个最大连续根集是不够的。所以，为了较精确估计 BCH 码的实际距离，必须对其他的非最大连续根集进行考虑，哈德曼与曾开明(Hartmann & Tzeng)于 1972 年提出了 HT 限，改进了 BCH 码限。

定理 7.1.2(HT 限) 若 BCH 码的生成多项式 $g(x)$ 的根集 R 中含有 s 组 $\delta-1$ 个 α 的连续元素： $R \subseteq \{\alpha^{m_0+ia+j} \mid i=0, 1, \dots, s-1; j=0, 1, \dots, \delta-2\}$ 且 $(n, a)=1$, $\alpha \in \text{GF}(q^m)$ 是 n 级元素，则码的最小距离 $d_{\text{HT}} \geq \delta + s - 1$ 。

证明 从码的 MS 多项式或谱表示证明。以码字的频域分量或 MS 多项式的系数组成

的半无限序列为

$$(A^n)^\infty = (\cdots \triangle 0 0 \cdots 0 \triangle \cdots \triangle 0 0 \cdots 0 \triangle \cdots \triangle 0 0 \cdots 0 \triangle \cdots)$$

↑ ↑ ↑ ↑

位置指示: $m \quad m+\delta-2 \quad m+a \quad m+(s-1)a$

这里, $\triangle \in GF(q^m)$ 。由 (A^n) 的循环移位组成以下矩阵:

$$M(A^n) = \begin{bmatrix} \cdots \triangle 0 0 \cdots 0 \triangle \cdots \triangle 0 0 \cdots 0 \triangle \cdots \cdots \triangle 0 0 \cdots 0 \triangle \cdots \cdots \\ \cdots 0 0 \cdots 0 \triangle \cdots \triangle 0 0 \cdots 0 \triangle \cdots \cdots \triangle 0 0 \cdots 0 \triangle \cdots \cdots \\ \vdots \quad \vdots \quad \vdots \\ \cdots 0 0 \triangle \cdots \triangle 0 0 \cdots 0 \triangle \cdots \cdots \triangle 0 0 \cdots 0 \triangle \cdots \cdots \cdots \cdots \end{bmatrix}$$

↑ ↑ ↑

$m \quad m+a \quad m+(s-1)a$

$$= \begin{bmatrix} (A^n) \\ s(A^n) \\ \vdots \\ s^{\delta-2}(A^n) \\ \vdots \end{bmatrix} \quad (7.1.6)$$

式中, s^i 表示 (A^n) 往左循环移位 i 位。由定理 5.9.3 和定理 5.9.2 可知, $M(A^n)$ 矩阵的首 $\delta-1$ 行是线性无关的, 在位置 $m, m+a, \dots, m+(s-1)a$ 上, 这些行中的元素均为 0。把 $M(A^n)$ 中的首 $\delta-1$ 行进行线性组合, 我们可以得到一个周期重复序列 $(A'^n)^\infty$, 线性组合的系数如下选择: 使得 $(A'^n)^\infty$ 序列在 $\delta-2$ 个位置: $m+sa, m+(s+1)a, \dots, m+(s+\delta-3)a$ 都是 0, 且 $(A'^n)^\infty \neq 0^\infty$ (因为 $M(A^n)$ 的首 $\delta-1$ 行线性无关)。

由上可知 $(A'^n)^\infty$ 含有 $s+\delta-2$ 个零, 每两个 0 之间间隔为 ac 位 (a 是正整数)。现在对序列 $(A'^n)^\infty$ 采样, 当 $(n, a)=1$ 时, 每隔 c 位对 $(A'^n)^\infty$ 进行采样所得到的序列 $(A''^n)^\infty$ 含有一组 $s+\delta-2$ 个连续的 0。所以, 由引理 5.9.3, $L(A''^n)^\infty \geq s+\delta-1$, 由采样定理 5.9.5 可知 $L((A''^n)^\infty) = L((A'^n)^\infty) \geq s+\delta-1$, 再根据序列的移位和相加定理 5.9.4 可知 $L((A^n)^\infty) \geq L((A'^n)^\infty) \geq s+\delta-1$, 再根据 Blahut 定理立即可得 $w(a^n) = L((A^n)^\infty) \geq s+\delta-1$ 。 ■

如 [17, 9] 非本原 BCH 码, 它以 $GF(2^8)$ 中的 $\alpha^{15}=\beta, \beta^2, \beta^4, \beta^8, \beta^9, \beta^{13}, \beta^{15}, \beta^{16}$ 元素为根, $g(x)=x^8+x^7+x^6+x^4+x^2+x+1$ 。由 BCH 限可知码的最小距离 $d \geq \delta=3$, 但由于码有 3 组 2 个连续元素为根: $\beta, \beta^2, \beta^8, \beta^9, \beta^{15}, \beta^{16}$, 且 $c=7, (7, 15)=1$, 由 HT 限可知码的最小距离 $d_n \geq 3+2=5$, 实际上该码的真正最小距离就是 5, 能纠正两个随机错误。

如同推论 7.1.1 一样, 也可把 HT 限推广到广义连续根集的情况, 也即 $g(x)$ 的根集: $R \supseteq \{\alpha^{a_0+a_1i+a_2j} \mid i=0, 1, \dots, s-1, j=0, 1, \dots, \delta-2\}, (n, a_1)=1, (n, a_2)=1$, 但是对 a_2 的条件还可放宽, 得到以下的广义 HT 限 (GHT)。

推论 7.1.2 (GHT 限) 若 BCH 码的 $g(x)$ 的根集 $R \supseteq \{\alpha^{a_0+a_1i+a_2j}, 0 \leq i \leq s-1; 0 \leq j \leq \delta-2\}, (n, a_1) < \delta, (n, a_2)=1, \delta \geq 2$, 则 $d_{GHT} \geq \delta+s-1$ 。

例如, $n=51$ 的二进制循环码, $g(x)=m_1(x)m_5(x)m_9(x)$, 根集 $R=\{\alpha^i, i=1, 2, 4, 5, 7, 8, 9, 10, 13, 14, 15, 16, 18, 20, 21, 26, 28, 29, 30, 32, 33, 36, 40, 42\}, \alpha^{51}=1$,

$\alpha \in GF(2^6)$ 。因为根集 $R \supseteq \{\alpha^{7+i+6j} | i=0,1; j=0,1,2,3\}$, $a_1=1$, $a_2=6$, $(51,6)=3 < \delta=5$, 故由 GHT 限得到码的 $d_{GHT} \geq 5+1=6$ 。但由 HT 限, $R \supseteq \{\alpha^{7+i+7j} | i=0,1; j=0,1,2\}$, $c_1=1$, $c_2=7$, $(1,51)=1$, $(7,51)=1$, 可知码的 $d_{HT} \geq 4+1=5$ 。

无论是 HT 限还是 GHT 限, 它们仅适用于 $g(x)$ 的 s 组 $\delta-1$ 个元素的根集中, 每组根之间的间隔相等, 都为 a 。如第一组根的第一个元素为 α^{m_0} , 则第二组、第三组根的第一个元素分别为 α^{m_0+a} , α^{m_0+2a} 等等。如果在 s 组中缺了几组, 即每组之间间隔不相等, 则不能用 HT 限。针对这种间隔不相等情况, 鲁斯(Roos)于 1983 年提出了 Roos 限。

定理 7.1.3(Roos 限) 若 BCH 码的 $g(x)$ 根集 R 含有 s 组 α 的 $\delta-1$ 个连续元素 $R \supseteq \{\alpha^{m_0+ai+j} | i \text{ 只取 } 0 \text{ 至 } s+\mu-1 \text{ 中的 } s \text{ 个整数}; j=0,1,\dots,\delta-2\}$, 且 $(a,n)=1$, $a^s=1$, $\alpha \in GF(q^m)$, 则码的最小距离 $d_R \geq \delta+s-1$ 。

证明 设根集 R 中 s 组 α 的 $\delta-1$ 个连续元素的第一个元素分别为 α^{m_0} , $\alpha^{m_0+i_1a}$, $\alpha^{m_0+i_2a}$, \dots , $\alpha^{m_0+i_{s-1}a}$, 则如同式(7.1.6)的 $M(A^n)$ 的首 $\delta-1$ 行, 这些行不仅线性无关, 并且每行的第 m_0 , m_0+i_1a , m_0+i_2a , \dots , $m_0+i_{s-1}a$ 位置均为 0 元素, 用证明 HT 限时所用的方法, 我们首先构造一个 $(A')^\infty$ 序列, 它是矩阵的首 $\delta-1$ 行的线性组合。并且在某些合适的位置中再附加 $\delta-2$ 个 0, 其中 μ 个 0 补在空缺组的第一个元素的位置上, 而 $\delta-2-\mu$ 个 0 在最后面, 这样得到的序列与证明 HT 限时中得到的 $(A')^\infty$ 序列有相同的形式, 因而其余的证明如同 HT 限。 ■

虽然 Roos 限适合于 $g(x)$ 的根集为缺组情况, 但要求 s 组根中的每一组的元素个数相同且连续。如果 s 组根中有几组的元素数目不同, 也就是有几组不完整, 则不能用 Roos 限, 必须对它进行修正, 得到了广义 Roos 限(GR 限)。

定理 7.1.4(GR 限)^[4] 设 BCH 码 $g(x)$ 的根集 $R \supseteq \{\alpha^{m_0}, \alpha^{m_0+i_1}, \dots, \alpha^{m_0+i_{s-2}}; \alpha^{m_0+j_1a}, \alpha^{m_0+i_1+j_1a}, \dots, \alpha^{m_0+i_{s-2}+j_1a}; \dots, \alpha^{m_0+j_{s-1}a}, \alpha^{m_0+i_1+j_{s-1}a}, \dots, \alpha^{m_0+i_{s-2}+j_{s-1}a}\}$, 这里 $(a,n)=1$, $i_{s-2} > i_{s-1} > \dots > i_1 > 0$, $j_{s-1} > j_{s-2} > \dots > j_1 > 0$, $\alpha \in GF(q^m)$, $a^s=1$ 。

若不完整组中缺 α 幂次的数目 $\gamma = i_{s-1} - \delta + 2$, 缺组的数目 $\mu = j_{s-1} - s + 1$, 且同时满足以下不等式:

- (1) $\gamma < s$
- (2) $s + \mu \leq d$

这里 d 是关于最小距离 d_m 的下限 ($d_m \geq d$), 则 $d_{GR} = d_{m_1} \geq \delta + s - 1$ 。

例如, [21, 9]二进制循环码的 $g(x)$ 的根集 $R \supseteq \{\alpha^1, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^8, \alpha^9, \alpha^{11}, \alpha^{12}, \alpha^{15}, \alpha^{16}, \alpha^{18}\}$, $\alpha^{21}=1$, $\alpha \in GF(2^6)$ 。把这些根按以下次序排列: $\alpha^6 - \alpha^8, \alpha^9; \dots; \alpha^{21+1}, \alpha^{21+2}, \alpha^{21+3}, \alpha^{21+4}; \alpha^{21+9}, \dots, \alpha^{21+11}, \alpha^{21+12}$ 。可见在四组等间隔元素集合中, 只有三组是根集, 缺了一组 $\{\alpha^{14}, \alpha^{15}, \alpha^{16}, \alpha^{17}\}$, $\mu=1$ 。并且在两组中分别缺了一个元素 α^7 和 α^{21+10} , $\gamma=1$, 且 $(8, 21)=1$, $s=3 > \gamma=1$, 满足不等式(1)。由于码以 $\alpha, \alpha^2, \alpha^3, \alpha^4$ 为根。由 BCH 限知 $d_m = d_{BCH} = 5$, 所以 $s + \mu = 3 + 1 = 4 < 5$, 满足不等式(2)。可知, $d_{GR} \geq 4 + 3 - 1 = 6$, 该码的实际最小距离就是 6。

如果在 $g(x)$ 根集 R 中, 包含的 s 组元素不仅不等间隔(缺组情况), 而且每组中的元素不连续(缺元素), 在这种情况下, GR 限也无力处理。为此必须对 GR 限再作进一步的推广(ER 限)。

推论 7.1.3(ER 限)^[4] 若循环码的 $g(x)$ 根集含有 s 组不等间隔(间隔为 a), 而且不完

整的 $\delta-1$ 个 α 的连续元素，即 $R \supseteq \{\alpha^{m_0+ja}, \alpha^{m_0+i_1+ja}, \dots, \alpha^{m_0+i_{\delta-2}+ja} \mid j=0, 1, \dots, s-1\}$ ，且 $(a, n)=1, i_{\delta-2} > i_{\delta-1} > \dots > i_1 > 0$ ，缺 α 幂次的元素数目 $\gamma = i_{\delta-1} - \delta + 2$ ，满足 $\gamma < s$ ，则 $d_{\text{ER}} \geq \delta + s - 1$ 。

例如，[21, 12]二进制循环码的根集 $R \supseteq \{\alpha^0=1, \alpha^3, \alpha^6, \alpha^7, \alpha^9, \alpha^{12}, \alpha^{14}, \alpha^{15}, \alpha^{18}\}$ ， $\alpha^{21}=1, \alpha \in \text{GF}(2^6)$ ，可知 $g(x)$ 的根含有以下元素： $\alpha^7, \dots, \alpha^9, \alpha^{12}, \dots, \alpha^{14}$ 。 $a=5, (5, 21)=1, s=2, \gamma=1, \delta-1=2, \gamma < s$ ，由此可知 $d_{\text{ER}} \geq 3+2-1=4$ ，实际距离也为 4。事实上，由 HT 限也可得 $d_{\text{HT}} \geq 4$ 。

对于 $g(x)$ 根集 R 中元素分布更为一般情况，1986 年范林特和威尔逊 (Vanlent & Wilson) 提出了 LW 限^[5]。此限对估计 BCH 码的最小距离更为精确，但计算更为复杂。

定理 7.1.5(LW 限)^[5] 设 BCH 码 $g(x)$ 的根集 $R \supseteq AB$ ， $A=\{\alpha^j \mid j=l, k, \dots\}$ ， $B=\{\alpha^c \mid i=0, a, b, \dots\}$ ，且 $(n, c)=1$ 。支持 I 为非 0 码字的最小重量。构造以下矩阵

$$\begin{aligned} M(A)_{I_1} &= \begin{bmatrix} 1 & \alpha^{i_1} & \alpha^{2i_1} & \cdots & \alpha^{(n-1)i_1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \alpha^{i_{I_1}} & \alpha^{2i_{I_1}} & \cdots & \alpha^{(n-1)i_{I_1}} \end{bmatrix} \\ M(B)_{I_1} &= \begin{bmatrix} 1 & \alpha^i_1 & \alpha^{2i_1} & \cdots & \alpha^{(n-1)i_1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \alpha^{i_{I_2}} & \alpha^{2i_{I_2}} & \cdots & \alpha^{(n-1)i_{I_2}} \end{bmatrix} \end{aligned}$$

若 $\text{rank}(M(A)_{I_1}) + \text{rank}(M(B)_{I_1}) > |I_1|$ ，则重选 I_2 ，构造 $M(A)_{I_2}, M(B)_{I_2}$ 再按上式检验，直至

$$\text{rank}(M(A)_{I_i}) + \text{rank}(M(B)_{I_i}) \leq |I_i|$$

则

$$d_{\text{LW}} \geq |I_i|$$

式中， $\text{rank}(M)$ 为矩阵 M 的秩， $|I_i|$ 为第 i 次估计的距离。

例如，[21, 7] 非本原 BCH 码， $g(x)$ 的根集 $R \supseteq \{\alpha^j, j=1, 2, 3, 4, 6, 7, 8, 9, 11, 12, 14, 15, 16, 18\}$ 。 $\alpha^{21}=1, \alpha \in \text{GF}(2^6)$ 。

由 BCH 限可得 $d_B \geq 5$ ，由 HT 限 $d_{\text{HT}}=5+1=6$ ，由 GR 限可知， $d \geq \delta+s-1=7$ ，但由 LW 限可知： $A=\{\alpha^j \mid j=3, 4\}$ ， $B=\{\alpha^{i_j}, i=0, 1, 2, 3, 5, 6, (21, 4)=1\}$ 。由 GR 限可知，应选 $|I_1|=7$ ，构造 $M(A)_{I_1}$ 和 $M(B)_{I_1}$ 矩阵，计算它们的秩分别为 2 和 6，所以

$$\text{rank}(M(A)_{I_1}) + \text{rank}(M(B)_{I_1}) = 2 + 6 = 8 > |I_1| = 7$$

再选 $I_2=8$ ，构造 $M(A)_{I_2}$ 和 $M(B)_{I_2}$ 矩阵，它们的秩分别为 2 和 6，所以

$$\text{rank}(M(A)_{I_2}) + \text{rank}(M(B)_{I_2}) = 2 + 6 = 8 = |I_2|$$

可知 $d_{\text{LW}}=8$ ，此码的实际最小距离就是 8。

用 LW 限估计 BCH 码的最小距离是很精确的，在码长 $n < 63$ 以内，除了两个码以外，都可用上述介绍的限精确估计，表 7-1 给出了 $n < 63$ 的部分 BCH 码的实际最小距离及用各种限估计的距离。

当然很多 BCH 码的实际距离是等于它的设计距离的。例 5.5 中的 [15, 5] 码，它以 $\alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6, \alpha^8, \alpha^9, \alpha^{10}, \alpha^{12}$ 为根，最大连续根数是 6，由 BCH 限可知，该码的设计距

离 $\delta=7$, 这与码的实际距离一样。又如循环汉明码, 它以 $\alpha, \alpha^2, \alpha^4, \dots, \alpha^{2^{m-1}}$ 为根, $\delta=3$ 与实际距离也一样。那么, 码应该满足什么条件, 才能使它的设计距离等于实际距离呢? 下面几个定理给出了二进制和 q 进制 BCH 码的设计距离等于实际距离的条件^[1]。

表 7-1 $n < 63$ 的部分 BCH 码的距离限比较

| 序号 | n | k | d | d_{BCH} | 根集 $\{i; \alpha^i\}$ | 所用的限 | 序号 | n | k | d | d_{BCH} | 根集 $\{i; \alpha^i\}$ | 所用的限 |
|----|-----|-----|-----|------------------|----------------------|------------|----|-----|-----|-----|------------------|----------------------|----------|
| 1 | 15 | 9 | 4 | 3 | 3, 5 | HT | 24 | 43 | 29 | 6 | 4 | 1 | Roos |
| 2 | 17 | 9 | 5 | 4 | 1 | HT | 25 | 43 | 15 | 13 | 7 | 1, 3 | HT |
| 3 | 21 | 13 | 4 | 3 | 3, 7, 9 | HT | 26 | 45 | 35 | 4 | 3 | 5, 9 | HT |
| 4 | 21 | 12 | 4 | 3 | 0, 3, 7, 9 | d 为偶 | 27 | 45 | 27 | 6 | 5 | 1, 3, 15 | Roos |
| 5 | 21 | 9 | 6 | 5 | 1, 3, 9 | LW | 28 | 45 | 23 | 7 | 6 | 1, 5, 21 | LW |
| 6 | 21 | 9 | 8 | 6 | 0, 1, 3, 7 | LW | 29 | 45 | 21 | 8 | 7 | 1, 5, 9, 15 | LW |
| 7 | 21 | 7 | 8 | 5 | 1, 3, 7, 9 | Roos | 30 | 45 | 17 | 8 | 7 | 1, 3, 5, 9, 15 | HT |
| 8 | 23 | 12 | 7 | 5 | 1 | LW | 31 | 45 | 15 | 10 | 8 | 1, 7, 9, 15 | Roos |
| 9 | 23 | 11 | 8 | 6 | 0, 1 | 8 号码的偶重量子码 | 32 | 45 | 14 | 10 | 8 | 0, 1, 7, 9, 15 | 31 号码的子码 |
| 10 | 31 | 21 | 5 | 4 | 1, 5 | HT | 33 | 45 | 9 | 12 | 9 | 1, 5, 7, 9, 15 | LW |
| 11 | 31 | 16 | 7 | 5 | 1, 5, 7 | Roos | 34 | 47 | 24 | 11 | 5 | 1 | LW |
| 12 | 31 | 11 | 11 | 7 | 1, 3, 5, 11 | LW | 35 | 51 | 41 | 4 | 3 | 3, 17 | HT |
| 13 | 33 | 21 | 4 | 3 | 3, 11 | HT | 36 | 51 | 35 | 5 | 4 | 1, 9 | HT |
| 14 | 33 | 13 | 10 | 5 | 1, 3 | LW | 37 | 51 | 34 | 6 | 4 | 0, 1, 5 | LW |
| 15 | 33 | 11 | 11 | 8 | 1, 3, 11 | LW | 38 | 51 | 27 | 8 | 5 | 1, 3, 9 | LW |
| 16 | 35 | 25 | 4 | 3 | 5, 7, 15 | HT | 39 | 51 | 25 | 10 | 9 | 1, 3, 5, 17 | LW |
| 17 | 35 | 20 | 6 | 5 | 1, 5 | LW | 40 | 51 | 25 | 10 | 5 | 1, 3, 17, 19 | LW |
| 18 | 35 | 16 | 7 | 6 | 1, 5, 7 | Roos | 41 | 51 | 19 | 10 | 9 | 1, 5, 9, 19 | LW |
| 19 | 35 | 13 | 8 | 6 | 1, 5, 7, 15 | LW | 42 | 51 | 17 | 12 | 6 | 1, 3, 9, 19 | LW |
| 20 | 39 | 25 | 4 | 3 | 3, 13 | HT | 43 | 51 | 17 | 14 | 9 | 1, 3, 5, 17, 19 | LW |
| 21 | 39 | 15 | 10 | 7 | 1, 3 | LW | 44 | 51 | 11 | 15 | 9 | 1, 3, 5, 11, 19 | LW |
| 22 | 39 | 13 | 12 | 7 | 1, 3, 13 | LW | 45 | 55 | 35 | 5 | 4 | 1 | HT |
| 23 | 41 | 21 | 9 | 6 | 1 | LW | 46 | 55 | 21 | 15 | 8 | 1, 5, 11 | LW |

定理 7.1.6 二进制本原 BCH 码的码长 $n=2^m-1$, 设计距离 δ 若为 $2t+1$, 则如:

$$(1) 2^m < \sum_{i=1}^{t+1} (2_i^m - 1)$$

或

$$(2) m > \log_2((t+1)!) + 1 \sim t \log_2\left(\frac{t}{e}\right)$$

当 $t \rightarrow 0$, 则码的实际距离等于设计距离。

如 $m=5$, $t=1, 2, 3$, 得到的 [31, 26], [31, 21], [31, 16] 等 BCH 码, 由于

$\sum_{i=0}^{t+1} \binom{31}{i} > 2^{5t}$, 所以 δ 等于实际距离。

定理 7.1.7 若码长 $n=ab$, 且二进制 BCH 码的设计距离 $\delta=a$, 则 a 是码的实际距离。

例如, [15, 7, 5] 码和 [15, 11, 3] 码, $15=5\times 3$, $a=5=\delta$ 和 $a=3=\delta$ 都等于码的实际距离。

定理 7.1.8 长为 $n=q^m-1$ 的 $GF(q)$ 上的 BCH 码, 若有设计距离 $\delta=q^h-1$, 则码的实际距离等于 δ 。

如 $q=2$, $m=3$, $h=2$, 就是一个长为 7 的 $\delta=3$ 的 BCH 码, 这是一个 [7, 4, 3] 循环汉明码, 它的实际距离就等于 δ 。

下面定理给出了 BCH 码实际最小距离的上限。

定理 7.1.9 设计距离为 δ 的 $GF(q)$ 上的本原 BCH 码, 它的实际最小距离 $d \leq q\delta + q - 2$ 。

由此定理可知, 二进制本原 BCH 码的实际最小距离不会超过设计距离的一倍。

上面这些定理虽然给出了码的实际距离 d 等于 δ 的条件。但 $d=\delta$ 时, n 的充分必要条件是什么, 这个问题至今仍没有解决。

定理 7.1.5 至定理 7.1.8 的证明, 需要其它预备知识, 可参阅[1]。

由循环码的理论可知, BCH 码的生成多项式

$$g(x) = \text{LCM}(m_0(x)m_1(x)\cdots m_{\delta-2}(x))$$

式中, $m_0(x), m_1(x), \dots, m_{\delta-2}(x)$ 分别是 $\alpha^{m_0}, \alpha^{m_0+1}, \dots, \alpha^{m_0+\delta-2}$ 的最小多项式。由于 $\alpha^{m_0+i} \in GF(q^m)$ 的最小多项式的次数最多不会超过 m , 因此 $\deg g(x) \leq m(\delta-1)$, 由此可知 BCH 码的信息元个数

$$K \geq n - m(\delta - 1) \quad (7.1.7)$$

从这里看出 BCH 码的生成多项式 $g(x)$ 的次数不能任意取值, 因而它的信息位数和最小距离也往往不能任意变化。

三、BCH 码的覆盖半径

码的覆盖半径是衡量码纠错性能的另一重要参数。由于证明比较复杂, 因此下面仅给出 BCH 码的覆盖半径的主要结果, 定理中的 t 均指由 BCH 码限所决定。

定理 7.1.10^[6] 令 $g(x) = m_{i_1}(x)m_{i_2}(x)\cdots m_{i_t}(x)$ 是 $[n, k, 2t+1]$ 二进制本原 BCH 码 C_B 的生成多项式, 且 $g(x)$ 无重根, $D = \max\{i_1, i_2, \dots, i_t\}$ 。如果 $2^m \geq (D-1)^{t+2}$, 则码 C_B 的覆盖半径 t $[n, k] < 2t+1 = d$ 。

但当码长 $n=2^m-1$ 充分大时, 本原 BCH 码的覆盖半径可以减小。

定理 7.1.11^[7] 存在一个与 t 有关的整数 m_0 , 当 $m > m_0$ 时, 长为 2^m-1 的二进制本原 BCH 码的覆盖半径是 $2t-1$ 。

循环汉明码是纠 1 个错误的本原 BCH 码, 由第三章知, 它是完备码。因此, 纠 1 个错误的本原 BCH 码的覆盖半径是 1。已证明^[8] 纠 2 个错误的二进制 BCH 码是准完备码, 因此, 它们的覆盖半径是 3。

推论 7.1.4 对于 n 是 2^m-1 因子的二进制 $[n, k, 3]$ 和 $[n, k, 5]$ BCH 码, 它们的覆盖半径分别是 1 和 3。

定理 7.1.12^[9] 所有二进制纠 3 个错误的本原 BCH 码，覆盖半径是 5。

非本原 BCH 码的码长 $n = (2^m - 1)/a$ 是 $2^m - 1$ 的因子，这里 a 是正整数。下面定理给出了非本原 BCH 码覆盖半径。

定理 7.1.13^[6] 令二进制非本原 BCH 码的码长 $n = (2^m - 1)/a$ ，生成多项式 $g(x) = m_a(x)m_{3a}(x)\cdots m_{(2t-1)a}(x)$ ，如果 $2^m \geq ((2t-1)a-1)^{4t+2}$ ，则码的覆盖半径 $t[n, k]$

$$2t-1 \leq t[n, k] \leq 2t+1$$

如果码长更长，则上述限还可以进一步改进。

定理 7.1.14^[7] 令 $a \neq 1$ 是 $2^m - 1$ 的因子，若 $m > (4t+2)\log_2(2t-1)a$ ，则非本原 BCH 码的覆盖半径是 $2t$ 。

由式(7.1.1)知，纠 t 个错误 BCH 码的生成多项式 $g(x) = \text{LCM}(m_1(x)m_2(x)\cdots m_u(x))$ ，因此，纠 1 个错误的 BCH 码 C_{B1} ，一般含有纠 2 个错误的 BCH 码 C_{B2} 作为子码，同样 C_{B2} 中含有 C_{B3} 等。所以，如果 $C_{B1} \supset C_{B2} \supset C_{B3} \supset \dots$ ，则可以证明，BCH 码 C_{B2} 的覆盖半径 $t[C_{B2}]$ 有以下性质^[10]：

$$t[C_{B2}] \geq d_{B1} = 3$$

$$t[C_{B3}] \geq d_{B2} = 5$$

$$t[C_{B4}] \geq d_{B3} = 7$$

...

这里， d_{Bi} 是纠 i 个错误的 C_{Bi} 码的设计距离。

四、码的重量分布

BCH 码的重量分布是衡量 BCH 码纠、检错性能的重要参数。目前对于纠 2 个和 3 个错误的本原 BCH 码，以及某些低码率 BCH 码的重量分布是已知外，对于其它码仍不完全清楚。

表 7-2 和表 7-3 分别给出了纠 2 个和 3 个错误的本原 BCH 码对偶码的重量分布。

表 7-2 $[2^m - 1, k, 5]$ 本原 BCH 码对偶码的重量分布

| 奇数 $m \geq 3$ | |
|---------------------------|--------------------------------------|
| 重量 i | 重量为 i 的码矢数 B_i |
| 0 | 1 |
| $2^{m-1} - 2^{(m+1)/2-1}$ | $[2^{m-2} + 2^{(m-1)/2-1}](2^m - 1)$ |
| 2^{m-1} | $(2^m - 2^{m-1} + 1)(2^m - 1)$ |
| $2^{m-1} + 2^{(m+1)/2-1}$ | $[2^{m-2} - 2^{(m-1)/2-1}](2^m - 1)$ |

| 偶数 $m \geq 4$ | |
|---------------------------|---|
| 重量 i | 重量为 i 的码矢数 B_i |
| 0 | 1 |
| $2^{m-1} - 2^{(m+2)/2-1}$ | $2^{(m-2)/2-1}[2^{(m-2)/2} + 1](2^m - 1)/3$ |
| $2^{m-1} - 2^{m/2-1}$ | $2^{(m+2)/2-1}(2^{m/2} + 1)(2^m - 1)/3$ |
| 2^{m-1} | $(2^{m-2} + 1)(2^m - 1)$ |
| $2^{m-1} + 2^{m/2-1}$ | $2^{(m+2)/2-1}(2^{m/2} - 1)(2^m - 1)/3$ |
| $2^{m-1} + 2^{(m+2)/2-1}$ | $2^{(m-2)/2-1}[2^{(m-2)/2} - 1](2^m - 1)/3$ |

表 7-3 $[2^m-1, k, \gamma]$ 本原 BCH 码对偶码的重量分布

| 奇数 $m \geq 5$ | |
|-------------------------|---|
| 重量 i | 重量为 i 的码字数目 B_i |
| 0 | 1 |
| $2^{m-1} - 2^{(m+1)/2}$ | $2^{(m-5)/2} [2^{(m-3)/2} + 1] (2^{m-1} - 1) (2^m - 1) / 3$ |
| $2^{m-1} - 2^{(m-1)/2}$ | $2^{(m-3)/2} [2^{(m-1)/2} + 1] (5 \cdot 2^{m-1} + 4) (2^m - 1) / 3$ |
| 2^{m-1} | $(9 \cdot 2^{2m-4} + 3 \cdot 2^{m-3} + 1) (2^m - 1)$ |
| $2^{m-1} + 2^{(m-1)/2}$ | $2^{(m-3)/2} [2^{(m-1)/2} - 1] (5 \cdot 2^{m-1} + 4) (2^m - 1) / 3$ |
| $2^{m-1} + 2^{(m+1)/2}$ | $2^{(m-5)/2} [2^{(m-3)/2} - 1] (2^{m-1} - 1) (2^m - 1) / 3$ |

| 偶数 $m \geq 6$ | |
|---------------------------|---|
| 重量 i | 重量为 i 的码字数目 B_i |
| 0 | 1 |
| $2^{m-1} - 2^{(m+4)/2-1}$ | $[2^{m-1} + 2^{(m+4)/2-1}] (2^m - 4) (2^m - 1) / 960$ |
| $2^{m-1} - 2^{(m+2)/2-1}$ | $7[2^{m-1} + 2^{(m+2)/2-1}] 2^m (2^m - 1) / 48$ |
| $2^{m-1} - 2^{m/2-1}$ | $2(2^{m-1} + 2^{m/2-1}) (3 \cdot 2^m + 8) (2^m - 1) / 15$ |
| 2^{m-1} | $(29 \cdot 2^{2m} - 4 \cdot 2^m + 64) (2^m - 1) / 64$ |
| $2^{m-1} + 2^{m/2-1}$ | $2(2^{m-1} - 2^{m/2-1}) (3 \cdot 2^m + 8) (2^m - 1) / 15$ |
| $2^{m-1} + 2^{(m+2)/2-1}$ | $7[2^{m-1} - 2^{(m+2)/2-1}] 2^m (2^m - 1) / 48$ |
| $2^{m-1} + 2^{(m+4)/2-1}$ | $[2^{m-1} - 2^{(m+4)/2-1}] (2^m - 4) (2^m - 1) / 960$ |

由这些表，再应用 MacWilliams 恒等式，就可计算得到纠 2 个和 3 个错误的本原 BCH 码的重量分布。

$[23, 12, 7]$ 二进制 Golay 码在实际中经常用到，它的重量算子为

$$A(x) = 1 + 253x^7 + 506x^8 + 1288x^{11} + 1288x^{12} + 506x^{15} + 253x^{16} + x^{23}$$

它的 $[24, 12, 8]$ 扩展 Golay 码的重量算子为

$$A'(x) = 1 + 759x^8 + 2576x^{12} + 759x^{16} + x^{24}$$

§ 7.2 二进制 BCH 码及其扩展

一、二进制 BCH 码

在实际中应用得最多的是码元取自 $GF(2)$ 中的二进制 BCH 码。由 BCH 码的定义可知，对任一个正整数 m ，一定可以构造出以下的二进制码。

取 $m_0=1$, $\delta=2t+1$, 又设 α 是 $GF(2^m)$ 的本原域元素，则由 BCH 码的定义可知：若码以 $\alpha, \alpha^2, \dots, \alpha^{2t}$ 为根，则二进制 BCH 码的生成多项式

$$g(x) = \text{LCM}(m_1(x)m_2(x)\dots m_{2t}(x)) \quad (7.2.1)$$

式中， $m_i(x)$ 是 α^i ($1 \leq i \leq 2t$) 的最小多项式，该 BCH 码一定能纠正 t 个错误。

由第四章知，在特征为 2 的 $GF(2^m)$ 域上， α^{2i} 的最小多项式与 α^i 的相同，所以，式 (7.2.1) 也可写成

$$g(x) = m_1(x)m_3(x)\cdots m_{2t-1}(x) \quad (7.2.2)$$

因此, 二进制 BCH 码以 $\alpha, \alpha^3, \alpha^5, \dots, \alpha^{2t-1}$ 为根, 码长

$$n = \text{LCM}(e_1, e_3, \dots, e_{2t-1}) \quad (7.2.3)$$

码的校验矩阵是

$$\mathbf{H} = \begin{bmatrix} \alpha^{n-1} & \alpha^{n-2} & \cdots & \alpha & 1 \\ (\alpha^3)^{n-1} & (\alpha^3)^{n-2} & \cdots & \alpha^3 & 1 \\ \vdots & \vdots & & \vdots & \vdots \\ (\alpha^{2t-1})^{n-1} & (\alpha^{2t-1})^{n-2} & \cdots & \alpha^{2t-1} & 1 \end{bmatrix} \quad (7.2.4)$$

式中, $e_1, e_3, \dots, e_{2t-1}$ 分别是 $\alpha, \alpha^3, \dots, \alpha^{2t-1}$ 元素的级。显然, 二进制 BCH 码的码长或者是 $2^n - 1$ 或者是它的一个因子。我们把上述结果归结成如下定理。

定理 7.2.1 对任何正整数 m 和 t , 一定存在一个二进制 BCH 码, 它以 $\alpha, \alpha^3, \dots, \alpha^{2t-1}$ 为根, 其码长 $n=2^m - 1$ 或是 $2^m - 1$ 的因子, 能纠正 t 个随机错误, 校验位数目至多为 $\delta g(x) = mt$ 个。

例 7.1 $m=4, \alpha \in \text{GF}(2^4)$ 是本原域元素, 它是 $x^4 + x + 1$ 的根。求码长 $n=2^4 - 1 = 15$ 的二进制 BCH 码。

(1) $t=1$, 则码以 $\alpha, \alpha^2, \alpha^4, \alpha^8$ 为根, α 的最小多项式 $m_1(x) = x^4 + x + 1$, 所以码的生成多项式

$$g(x) = m_1(x) = x^4 + x + 1$$

校验元数目是 $\delta g(x) = 4$, 得到一个 $[15, 11, 3]$ BCH 码, 这是一个纠正单个错误的循环汉明码。可知, 纠正单个错误的本原 BCH 码就是循环汉明码。

(2) $t=2$, 则码以 α, α^3 为根, α^3 的最小多项式为 $m_3(x) = x^4 + x^3 + x^2 + x + 1$, 所以

$$\begin{aligned} g(x) &= m_1(x)m_3(x) = (x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1) \\ &= x^8 + x^7 + x^6 + x^4 + 1 \end{aligned}$$

$$n = \text{LCM}(15, 5) = 15$$

有 8 个校验元, 得到一个 $[15, 7, 5]$ 码, 能纠正 2 个随机错误。

(3) $t=3$, 则码以 $\alpha, \alpha^3, \alpha^5$ 为根, α^5 的最小多项式 $m_5(x) = x^2 + x + 1$, 所以

$$\begin{aligned} g(x) &= m_1(x)m_3(x)m_5(x) \\ &= (x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1)(x^2 + x + 1) \\ &= x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1 \end{aligned}$$

得到一个 $[15, 5, 7]$ BCH 码。

(4) $t=4$ 。码以 $\alpha, \alpha^3, \alpha^5, \alpha^7$ 为根, α^7 的最小多项式 $m_7(x) = x^4 + x^3 + 1$, 所以码的生成多项式

$$\begin{aligned} g(x) &= m_1(x)m_3(x)m_5(x)m_7(x) \\ &= x^{14} + x^{13} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^7 \\ &\quad + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1 \end{aligned}$$

得到一个 $[15, 1, 15]$ 码, 这是一个重复码, 最小距离 $d=15$, 能纠正 7 个随机错误。该码虽以 $\alpha, \alpha^3, \alpha^5, \alpha^7$ 为根, 但由于共轭根系的原因, 该码实际上以 $\alpha^i, 1 \leq i \leq 14$, 14 个连续元素为根, 故设计距离 $\delta=15$, 实际最小距离也为 15。 ■

由这一系列例子看出, BCH 码的信息位、纠错个数并不是连续变化, 而是跳跃式变化的。上述这些码, 码长 $n=2^m-1$, 所以都是本原 BCH 码。

例 7.2 求码长 $n=21$, 纠 2 个随机错误的 BCH 码。

$n=21$ 不是 2^m-1 的类型, 故必是 2^m-1 的一个因子。 $2^6-1=21\times 3$, 所以 $\text{GF}(2^6)$ 是含有 21 级元素的最小域。

设 $\alpha \in \text{GF}(2^6)$ 是本原域元素, 它是 x^6+x+1 的根。令 $\beta=\alpha^3$, 则 β 的级是 21。要求纠正 2 个错误, 则 $g(x)$ 以 β, β^3 为根。 $\beta=\alpha^3$ 的最小多项式

$$m_1(x) = x^6 + x^4 + x^2 + x + 1$$

$\beta^3=(\alpha^3)^3=\alpha^9$ 的最小多项式 $m_3(x) = x^3 + x^2 + 1$, 所以码的生成多项式

$$\begin{aligned} g(x) &= m_1(x)m_3(x) \\ &= (x^6 + x^4 + x^2 + x + 1)(x^3 + x^2 + 1) \\ &= x^9 + x^8 + x^7 + x^5 + x^4 + x + 1 \end{aligned}$$

$$n = \text{LCM}(21, 7) = 21$$

得到一个[21, 12, 5]非本原 BCH 码。 ■

二、BCH 码的扩展

由第三章可知, 任何一个奇重量线性码, 增加 1 个全校验位后最小重量增加 1。若我们对奇最小距离的 $[n, k, d]$ BCH 码增加 1 个全校验位, 则变成一个 $[n+1, k, d+1]$ 码, 称它为**扩展 BCH 码**。若对 $[2^m-1, k, d]$ 本原 BCH 码增加 1 个全校验位, 则变成 $[2^m, k, d+1]$ 码, 称为**扩展本原 BCH 码**。

设 $[n, k, \delta]$ BCH 码以 $\alpha, \alpha^2, \dots, \alpha^{\delta-1}$ 为根。由式(7.1.3)可知, 码的校验矩阵

$$\mathbf{H} = \begin{bmatrix} \alpha^{n-1} & \alpha^{n-2} & \cdots & \alpha & 1 \\ (\alpha^2)^{n-1} & (\alpha^2)^{n-2} & \cdots & \alpha^2 & 1 \\ \vdots & \vdots & & \vdots & \vdots \\ (\alpha^{\delta-1})^{n-1} & (\alpha^{\delta-1})^{n-2} & \cdots & \alpha^{\delta-1} & 1 \end{bmatrix}$$

由 \mathbf{H} 矩阵每行每列的性质可知, $[n+1, k, \delta+1]$ 扩展码的校验矩阵为

$$\mathbf{H}_E = \begin{bmatrix} \alpha^{n-1} & \alpha^{n-2} & \cdots & \alpha & 1 & 0 \\ (\alpha^2)^{n-1} & (\alpha^2)^{n-2} & \cdots & \alpha^2 & 1 & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ (\alpha^{\delta-1})^{n-1} & (\alpha^{\delta-1})^{n-2} & \cdots & \alpha^{\delta-1} & 1 & 0 \\ 1 & 1 & \cdots & 1 & 1 & 1 \end{bmatrix} \quad (7.2.5)$$

显然, 这相当于码增加了以 $\alpha^0=1$ 为根。但是必须注意, 这种扩展方法与增加以 1 为根, 减少 1 个信息位的增余删信 BCH 码不一样, 它们的最小距离虽然都增加了 1, 但扩展 BCH 码的码长增加了 1, 且每个码字之间不一定存在循环关系, 而增余删信 BCH 码的码长与原码相同, 且仍是循环码。

如二进制 [7, 4, 3] 本原 BCH 码, 它以 $\alpha, \alpha^2, \alpha^4$ 为根, $g(x) = x^3 + x + 1$, 校验矩阵

$$\mathbf{H} = [\alpha^6 \ \alpha^5 \ \alpha^4 \ \alpha^3 \ \alpha^2 \ \alpha^1 \ 1]$$

该码增加一个全校验位后变成[8, 4, 4]扩展本原 BCH 码，它的校验矩阵

$$\mathbf{H}_E = \begin{bmatrix} \alpha^6 & \alpha^5 & \alpha^4 & \alpha^3 & \alpha^2 & \alpha & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

因[7, 4, 3]码是一个汉明码，所以[8, 4, 4]码也称为扩展汉明码。

若[7, 4, 3]码增加以 $\alpha^0=1$ 为根，则码的生成多项式 $g(x) = (x+1)(x^3+x+1) = x^4 + x^3 + x^2 + 1$ ，此码的校验矩阵为

$$\mathbf{H} = \begin{bmatrix} \alpha^6 & \alpha^5 & \alpha^4 & \alpha^3 & \alpha^2 & \alpha & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

这是一个[7, 3, 4]BCH 码，它其实就是一个增余删信汉明码。

码长 $n \leq 255$ 的二进制本原 BCH 码，它的 k 、 d 和生成多项式 $g(x)$ 列于表 7-4 中。表 5-2 中列出的 QR 码（除[7, 4]和[31, 16]码以外）都是非本原 BCH 码，此外还有一些非本原 BCH 码，它们的 d 和 $g(x)$ 列于表 7-5 中。

表 7-4 和表 7-5 中生成多项式 $g(x)$ 栏下面的数字，代表 $g(x)$ 的二进制系数的八进制数表示。例如，八进制数 13 的二进制数表示为 001011，因而代表 $g(x) = x^3 + x + 1$ 。 b 和 z 分别表示该码的纠突发错误能力和最佳程度。这将在第九章中讲到。表 7-5 中的 (QR) 表示该码是平方剩余码。

在 b 栏下有 * 者，表示该码若增加一个全校验位变成扩展本原 BCH 码，码的纠突发能力将增加 1。

我们可把表 7-4 中 n 、 k 、 t 之间的关系，画成如图 7-1 所示的曲线。为了更容易看出 BCH 码的纠错能力 t/n 与码率 $k/n=R$ 之间的关系，也画出了码长直到 65 535 位时，它们之间的关系。同时，为了便于比较，也画出了线性码纠错能力的 3 个限。

由该图可以明显看出：

(1) 当码的纠错能力 t/n 固定时，码长越短，码的 R 越大；当 $n \leq 31$ 时接近汉明限；而 $n \leq 1023$ 时，接近 V-G 限；但是随着 $n \rightarrow \infty$, $R \rightarrow 0$ 。

(2) 当码率 R 固定时，码长 ≤ 31 时，码的纠错能力接近汉明限； $n \leq 1023$ 时，接近 V-G 限；但是随着 $n \rightarrow \infty$, 码的纠错能力 $t/n \rightarrow 0$ 。

由此可知，BCH 码的性能在码长 ≤ 1023 时很好，都在 V-G 限以上；特别在短码时性能更好，接近汉明限；但随着码长的增加，性能变坏。因此 BCH 码做不到香农编码定理所要求的能力。

虽然图 7-1 是根据表 7-4 的 BCH 限得到的，但由定理 7.1.8 可知，BCH 码的实际距离

$$d \leq q\delta + q - 2 \approx q\delta = 2\delta \quad (\text{二进制情况})$$

因此，实际上当 R 固定时，BCH 码的纠错能力随着 n 的增加仍趋向于 0。即使如此，在中、短码长时 BCH 码仍不愧是一个很好的码。

表 7-4 码长 $n \leq 255$ 的二进制本原 BCH 码表

| n | k | t | b | $r/n(\%)$ | $z/2(\%)$ | $g(x)$ |
|-----|-----|-----|-----|-----------|-----------|---------------------|
| 7 | 4 | 1 | 1* | 42.9 | 33.3 | 13 |
| 15 | 11 | 1 | 1* | 26.7 | 25.0 | 23 |
| 15 | 7 | 2 | 4 | 53.3 | 50.0 | 721 |
| 15 | 5 | 3 | 5 | 66.7 | 50.0 | 2467 |
| 31 | 26 | 1 | 1* | 16.1 | 20.0 | 45 |
| 31 | 21 | 2 | 4* | 32.3 | 40.0 | 3551 |
| 31 | 16 | 3 | 7 | 43.4 | 46.7 | 107657 |
| 31 | 11 | 5 | 10 | 64.5 | 50.0 | 5423325 |
| 31 | 6 | 7 | 12* | 80.6 | 48.0 | 313365047 |
| 31 | 26 | 1 | 1* | 16.1 | 20.0 | 75 |
| 31 | 21 | 2 | 4* | 32.3 | 40.0 | 2303 |
| 31 | 16 | 3 | 6 | 48.4 | 40.0 | 135273 |
| 31 | 11 | 5 | 10 | 64.5 | 50.0 | 6163305 |
| 31 | 6 | 7 | 12* | 80.6 | 48.0 | 331722561 |
| 31 | 26 | 1 | 1* | 16.1 | 20.0 | 67 |
| 31 | 21 | 2 | 3* | 32.3 | 30.0 | 3557 |
| 31 | 16 | 3 | 7 | 48.4 | 46.7 | 141225 |
| 31 | 11 | 5 | 9* | 64.5 | 45.0 | 6715141 |
| 31 | 6 | 7 | 11* | 80.6 | 44.0 | 230745335 |
| 63 | 57 | 1 | 1* | 9.5 | 16.7 | 103 |
| 63 | 51 | 2 | 4* | 19.0 | 33.3 | 12471 |
| 63 | 45 | 3 | 5 | 28.6 | 27.8 | 1701317 |
| 63 | 39 | 4 | 11 | 38.1 | 45.8 | 166623567 |
| 63 | 36 | 5 | 12* | 42.9 | 44.4 | 1033500423 |
| 63 | 30 | 6 | 15 | 52.4 | 45.5 | 157464165547 |
| 63 | 24 | 7 | 17* | 61.9 | 43.6 | 17323260404441 |
| 63 | 18 | 10 | 21* | 71.4 | 46.7 | 1363026512351725 |
| 63 | 16 | 11 | 22 | 74.6 | 46.8 | 6331141367235453 |
| 63 | 10 | 13 | 25* | 84.1 | 47.2 | 472622305527250155 |
| 63 | 7 | 15 | 28 | 88.9 | 50.0 | 5231045543503271737 |
| 63 | 57 | 1 | 1* | 9.5 | 16.7 | 147 |
| 63 | 51 | 2 | 3 | 19.0 | 25.0 | 11253 |
| 63 | 45 | 3 | 7 | 28.6 | 38.9 | 1431377 |
| 63 | 39 | 4 | 11 | 38.1 | 45.8 | 156615307 |
| 63 | 36 | 5 | 13 | 42.9 | 40.7 | 1715374561 |
| 63 | 30 | 6 | 15 | 52.4 | 36.4 | 105065105421 |
| 63 | 24 | 7 | 1 | 61.9 | 46.2 | 10611427654563 |
| 63 | 18 | 10 | 2 | 71.4 | 46.7 | 1207106757642651 |

(续表)

| <i>n</i> | <i>k</i> | <i>t</i> | <i>b</i> | <i>r/n(%)</i> | <i>z/2(%)</i> | <i>g(x)</i> |
|----------|----------|----------|----------|---------------|---------------|--|
| 63 | 16 | 11 | 20 | 74.6 | 42.6 | 6625720617154137 |
| 63 | 10 | 13 | 25* | 84.1 | 49.1 | 743065712726034051 |
| 63 | 7 | 15 | 28 | 88.9 | 50.0 | 4567515266076214705 |
| 63 | 57 | 1 | 1* | 9.5 | 16.7 | 155 |
| 63 | 51 | 2 | 4 | 19.0 | 33.3 | 16223 |
| 63 | 45 | 3 | 8 | 28.6 | 44.4 | 1125063 |
| 63 | 39 | 4 | 10 | 38.1 | 41.7 | 102673553 |
| 63 | 36 | 5 | 12* | 42.9 | 44.4 | 1537210637 |
| 63 | 30 | 6 | 14* | 52.4 | 42.4 | 106054077561 |
| 63 | 24 | 7 | 16* | 61.9 | 41.0 | 14225100247067 |
| 63 | 18 | 10 | 22 | 71.4 | 48.9 | 1142177532557273 |
| 63 | 16 | 11 | 20* | 74.6 | 42.6 | 7456576205014441 |
| 63 | 10 | 13 | 25* | 84.1 | 47.2 | 755334022316461443 |
| 63 | 7 | 15 | 27* | 88.9 | 48.2 | 6534604245447336175 |
| 127 | 120 | 1 | 1* | 5.5 | 14.3 | 211 |
| 127 | 113 | 2 | 4* | 11.0 | 28.6 | 41567 |
| 127 | 106 | 3 | 8* | 16.5 | 38.1 | 11554743 |
| 127 | 99 | 4 | 12 | 22.0 | 42.9 | 447023271 |
| 127 | 92 | 5 | 14* | 27.6 | 40.0 | 624730022327 |
| 127 | 85 | 6 | 19 | 33.1 | 45.2 | 130704476322273 |
| 127 | 78 | 7 | 21 | 38.6 | 42.9 | 26230002166130115 |
| 127 | 71 | 9 | 27 | 44.1 | 48.2 | 6255010713253127753 |
| 127 | 64 | 10 | 29* | 49.6 | 46.0 | 1206534025570773100045 |
| 127 | 57 | 11 | 34 | 55.1 | 48.6 | 335265252505705053517721 |
| 127 | 50 | 13 | 37* | 60.6 | 48.1 | 54446512523314012421501421 |
| 127 | 43 | 14 | 40 | 66.1 | 47.6 | 17721772213651227521220574343 |
| 127 | 36 | 15 | 45 | 71.7 | 49.5 | 3146074666522075044764574721735 |
| 127 | 29 | 21 | 46* | 77.2 | 46.9 | 403114461367670603667530141176155 |
| 127 | 22 | 23 | 52 | 82.7 | 49.5 | 123376070404722522435445626637647043 |
| 127 | 15 | 27 | 55* | 88.2 | 49.1 | 22057042445604554770523013762217604353 |
| 127 | 8 | 31 | 59* | 93.7 | 49.6 | 7047264052751030651476224271567733130217 |
| 127 | 120 | 1 | 1* | 5.5 | 14.3 | 217 |
| 127 | 113 | 2 | 4* | 11.0 | 28.6 | 54505 |
| 127 | 106 | 3 | 8* | 16.5 | 38.1 | 14517623 |
| 127 | 99 | 4 | 12 | 22.0 | 42.9 | 2320637377 |
| 127 | 92 | 5 | 15* | 27.6 | 42.9 | 616051466261 |
| 127 | 85 | 6 | 18* | 33.1 | 42.9 | 152055627024155 |
| 127 | 78 | 7 | 24 | 38.6 | 49.0 | 35647104545000377 |

(续表)

| <i>n</i> | <i>k</i> | <i>t</i> | <i>b</i> | <i>r/n(%)</i> | <i>z/2(%)</i> | <i>g(x)</i> |
|----------|----------|----------|----------|---------------|---------------|--|
| 127 | 71 | 9 | 24* | 44.1 | 42.9 | 6402400420033061235 |
| 127 | 64 | 10 | 30* | 49.6 | 47.6 | 1346342546425521305535 |
| 127 | 57 | 11 | 34 | 55.1 | 48.6 | 257671620113233366110015 |
| 127 | 50 | 13 | 37 | 60.6 | 48.1 | 72364124311247042327752451 |
| 127 | 43 | 14 | 41 | 66.1 | 48.8 | 16563411316762141523202565773 |
| 127 | 36 | 15 | 45 | 71.7 | 49.5 | 3033145113365036627465666704563 |
| 127 | 29 | 21 | 48 | 77.2 | 49.0 | 403456765606274161324061641535467 |
| 127 | 22 | 23 | 52 | 82.7 | 49.5 | 104324444272233501517170527173574417 |
| 127 | 15 | 27 | 54* | 88.2 | 48.2 | 37071231012177064120650613540236515175 |
| 127 | 8 | 31 | 59* | 93.7 | 49.6 | 4220564640737462343050754765226654156257 |
| 127 | 120 | 1 | 1* | 5.5 | 14.3 | 235 |
| 127 | 113 | 2 | 4* | 11.0 | 28.6 | 76533 |
| 127 | 106 | 3 | 6* | 16.5 | 28.6 | 10513165 |
| 127 | 99 | 4 | 12 | 22.0 | 42.9 | 2113100037 |
| 127 | 92 | 5 | 16 | 27.6 | 45.7 | 530405706075 |
| 127 | 85 | 6 | 18 | 33.1 | 42.9 | 145007126304221 |
| 127 | 78 | 7 | 23 | 38.6 | 46.9 | 30222671041133777 |
| 127 | 71 | 9 | 26 | 44.1 | 46.4 | 5056513565374533677 |
| 127 | 64 | 10 | 29* | 49.6 | 46.0 | 1337626055235540411717 |
| 127 | 57 | 11 | 32 | 55.1 | 45.7 | 222602703023045367232111 |
| 127 | 50 | 13 | 38 | 60.6 | 49.4 | 73066070324015476437747471 |
| 127 | 43 | 14 | 40 | 66.1 | 47.6 | 16156210716167256615031425161 |
| 127 | 36 | 15 | 45* | 71.7 | 49.5 | 3145167034442312151474354252557 |
| 127 | 29 | 21 | 47* | 77.2 | 48.0 | 411034220540056004036332365536535 |
| 127 | 22 | 23 | 51 | 82.7 | 48.6 | 151571237655357367520454667705462071 |
| 127 | 15 | 27 | 56 | 88.2 | 50.0 | 24220353103706645134226343657675776433 |
| 127 | 8 | 31 | 58* | 93.7 | 48.7 | 6772370523071332110623245010363565460527 |
| 127 | 8 | 31 | 59* | 93.7 | 49.6 | 6225355160704241323261563507571370117651 |
| 255 | 247 | 1 | 1* | 3.1 | 12.5 | 435 |
| 255 | 239 | 2 | 5* | 6.3 | 31.2 | 267543 |
| 255 | 231 | 3 | 9* | 9.4 | 37.5 | 156720665 |
| 255 | 223 | 4 | 11* | 12.5 | 34.4 | 75626641375 |
| 255 | 215 | 5 | 17 | 15.7 | 42.5 | 23157564726421 |
| 255 | 207 | 6 | 21* | 18.8 | 43.8 | 16176560567636227 |
| 255 | 199 | 7 | 26 | 22.0 | 46.4 | 7633031270420722341 |
| 255 | 191 | 8 | 27* | 25.1 | 42.2 | 2663470176115333714567 |
| 255 | 187 | 9 | 27* | 26.7 | 39.7 | 52755313540001322236351 |
| 255 | 179 | 10 | 35* | 29.8 | 46.1 | 22624710717340432416300455 |

(续表)

| <i>n</i> | <i>k</i> | <i>t</i> | <i>b</i> | <i>r/n(%)</i> | <i>z/2(%)</i> | <i>g(x)</i> |
|----------|----------|----------|----------|---------------|---------------|---|
| 255 | 171 | 11 | 39* | 32.9 | 46.4 | 15416214212342356077061630637 |
| 255 | 163 | 12 | 43* | 36.1 | 46.7 | 7500415510075602551574724514601 |
| 255 | 155 | 13 | 47 | 39.2 | 47.0 | 3757513005407665015722506464677633 |
| 255 | 147 | 14 | 50* | 42.4 | 46.3 | 1642130173537165525304165305441011711 |
| 255 | 139 | 15 | 55* | 45.5 | 47.4 | 461401732060175561570722730247453567445 |
| 255 | 131 | 18 | 60* | 48.6 | 48.4 | 2157133314715101512612502774421420241654 71 |
| 255 | 123 | 19 | 64 | 51.8 | 48.5 | 1206140522420660037172103265161412262725 06267 |
| 255 | 115 | 21 | 68 | 54.9 | 48.6 | 6052666557210024726363640460027635255631 3472737 |
| 255 | 107 | 22 | 70* | 58.0 | 47.3 | 2220577232206625631241730023534742017657 4750154441 |
| 255 | 99 | 23 | 75* | 61.2 | 48.1 | 1065666725347317422274141620157433225241 1076432303431 |
| 255 | 91 | 25 | 80* | 64.3 | 48.8 | 6750265030327444172723631724732511075550 762720724344561 |
| 255 | 87 | 26 | 83 | 65.9 | 49.4 | 1101367634147432364352316343071720462067 22545273311721317 |
| 255 | 79 | 27 | 86* | 69.0 | 48.9 | 6670003563765750002027034420736617462101 5326711766541342355 |
| 255 | 71 | 29 | 90 | 72.2 | 48.9 | 2402471052064432151555417211233116320544 4250362557643221706035 |
| 255 | 63 | 30 | 93* | 75.3 | 48.4 | 1075447505516354432531521735770700366611 17264552676136567025433014 |
| 255 | 55 | 31 | 99 | 78.4 | 49.5 | 731542520350110013301527530603205430541 326755010557044426035473617 |
| 255 | 47 | 42 | 103 | 81.6 | 49.5 | 2533542017062646563033041377406233175123 334145446045005066024552543173 |
| 255 | 45 | 43 | 104 | 82.4 | 49.5 | 1520205605523416113110134637642370156367 0024470762373033202157025051541 |
| 255 | 37 | 45 | 107* | 85.5 | 49.1 | 5136330255067007414177447245437530420735 706174323432347644354737403044003 |
| 255 | 29 | 47 | 111 | 88.6 | 49.1 | 3025715536673071465527064012361377115342 242324201174114060254657410403565037 |
| 255 | 21 | 55 | 116 | 91.8 | 49.6 | 1256215257060332656001773153607612103227 341405653074542521153121614466510173725 |
| 255 | 13 | 59 | 120* | 94.9 | 49.6 | 4641732005052564544426573714250066004330 67744547656140317467721357026134460500 547 |

表 7-5 某些二进制非本原 BCH 码

| n | k | t | b | $r/n(\%)$ | $z/2(\%)$ | $g(x)$ |
|-----|-----|-----|-----|-----------|-----------|--------------|
| 17 | 9 | 2 | 3 | 47.1 | 37.5 | 727 (QR) |
| 21 | 12 | 2 | 4 | 42.9 | 44.4 | 1663 |
| 23 | 12 | 3 | 5 | 47.8 | 45.5 | 5343 |
| 33 | 22 | 2 | 3 | 33.3 | 27.3 | 5145 |
| 41 | 21 | 4 | 9 | 48.8 | 45.0 | 6647133 (QR) |
| 47 | 24 | 5 | 11 | 48.9 | 47.8 | 43073357 |
| 65 | 53 | 2 | 3 | 18.5 | 25.0 | 10761 |
| 65 | 40 | 4 | 10 | 38.5 | 40.0 | 354300067 |
| 73 | 46 | 4 | 12 | 37.0 | 44.4 | 1717773537 |

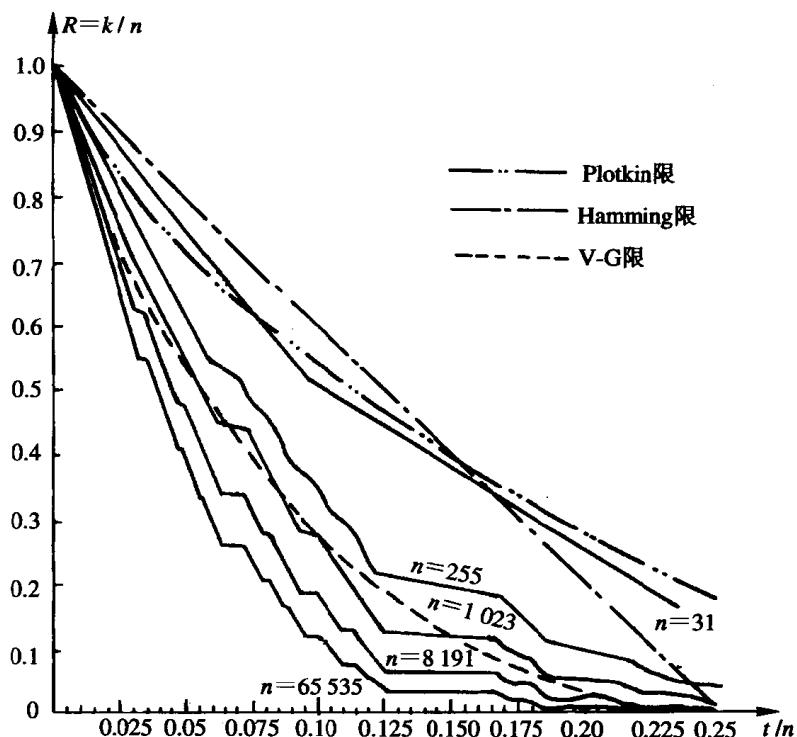


图 7-1 某些二进制 BCH 码的纠错能力

§ 7.3 Reed—Solomon(RS)码

RS 码是一类有很强纠错能力的多进制 BCH 码，也是一类典型的代数几何码。它首先由里德(Reed)和索洛蒙(Solomon)应用 MS 多项式于 1960 年构造出来的。当然，用 MS 多项式构造的 RS 码是非系统码，而用 BCH 码构造方法能产生系统码。

一、RS 码的时域编码

定义 7.3.1 $\text{GF}(q)(q \neq 2)$ 上，码长 $N=q-1$ 的本原 BCH 码称为 RS 码。

可知 RS 码最主要特点之一是码元取自 $\text{GF}(q)$ 上，而它的生成多项式的根也在 $\text{GF}(q)$

上, 所以 RS 码是码元的符号域与根域一致的 BCH 码。

因为 $x^{q-1} - 1 = \prod_{\alpha^i \in \text{GF}(q)} (x - \alpha^i)$, α^i 的最小多项式 $m_i(x) = x - \alpha^i$ 。所以, 长为 $N = q-1$, 设计距离为 δ 的 RS 码, 由 BCH 码的定义可知, 它的生成多项式

$$g(x) = (x - \alpha^{m_0})(x - \alpha^{m_0+1}) \cdots (x - \alpha^{m_0+\delta-2}) \quad (7.3.1)$$

通常情况下取 $m_0=1$, 此时

$$g(x) = (x - \alpha)(x - \alpha^2) \cdots (x - \alpha^{\delta-1}) \quad (7.3.2)$$

由此生成一个 q 进制的 $[q-1, q-\delta]$ RS 码, 有最小距离为 δ 。由于线性码的最大可能的最小距离是校验元的个数加 1, 而 RS 码恰好做到了这一点。因此, 称 RS 码为**极大最小距离可分码**, 简称 MDS 码。显然, RS 码的设计距离 δ 与实际距离 D 是一致的。

例 7.3 设码的符号取自 $\text{GF}(q) = \text{GF}(2^3)$ 中的元素, $\alpha \in \text{GF}(2^3)$ 是本原域元素, 它是 x^3+x+1 的根, 构造 $D=5$ 的 RS 码。 $\text{GF}(2^3)$ 的元素同表 4-1。

要求码的 $D=5$, 则码必以 $\alpha, \alpha^2, \alpha^3, \alpha^4$ 为根, 因此由式(7.3.2)可知, 码的生成多项式

$$g(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^3)(x - \alpha^4) = x^4 + \alpha^3x^3 + x^2 + \alpha x + \alpha^3$$

由此生成一个 $\text{GF}(2^3)$ 上的八进制 $[7, 3, 5]$ 本原 BCH 码, 也就是 $[7, 3, 5]$ RS 码。它的系统码生成矩阵由式(5.1.7)可知为

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & \alpha^4 & 1 & \alpha^4 & \alpha^5 \\ 0 & 1 & 0 & \alpha^2 & 1 & \alpha^6 & \alpha^6 \\ 0 & 0 & 1 & \alpha^3 & 1 & \alpha & \alpha^3 \end{bmatrix}$$

相应的校验多项式

$$h(x) = \frac{x^7 - 1}{g(x)} = x^3 + \alpha^3x^2 + \alpha^2x + \alpha^4$$

校验矩阵为:

$$\mathbf{H} = \begin{bmatrix} \alpha^4 & \alpha^2 & \alpha^3 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ \alpha^4 & \alpha^6 & \alpha & 0 & 0 & 1 & 0 \\ \alpha^5 & \alpha^6 & \alpha^3 & 0 & 0 & 0 & 1 \end{bmatrix}$$

或

$$\mathbf{H} = \begin{bmatrix} \alpha^6 & \alpha^5 & \alpha^4 & \alpha^3 & \alpha^2 & \alpha & 1 \\ (\alpha^2)^6 & (\alpha^2)^5 & (\alpha^2)^4 & (\alpha^2)^3 & (\alpha^2)^2 & (\alpha^2) & 1 \\ (\alpha^3)^6 & (\alpha^3)^5 & (\alpha^3)^4 & (\alpha^3)^3 & (\alpha^3)^2 & (\alpha^3) & 1 \\ (\alpha^4)^6 & (\alpha^4)^5 & (\alpha^4)^4 & (\alpha^4)^3 & (\alpha^4)^2 & (\alpha^4) & 1 \end{bmatrix}$$

一般情况下, $\text{GF}(q)$ 上系统码形式 $[q-1, q-D]$ RS 码的 \mathbf{H} 矩阵

$$\mathbf{H} = \begin{bmatrix} \alpha^{n-1} & \alpha^{n-2} & \cdots & \alpha & 1 \\ (\alpha^2)^{n-1} & (\alpha^2)^{n-2} & \cdots & \alpha^2 & 1 \\ \vdots & \vdots & & \vdots & \\ (\alpha^{D-1})^{n-1} & (\alpha^{D-1})^{n-2} & \cdots & \alpha^{D-1} & 1 \end{bmatrix} \quad (7.3.3)$$

或由式(5.1.9)可得 \mathbf{H} 的标准形式为

$$\mathbf{H} \equiv [\tilde{x}^{T_{n-1}} \tilde{x}^{T_{n-2}} \cdots \tilde{x}^T 1^T] \pmod{g(x)}$$

$GF(2^m)$ 上的 RS 码是 2^m 进制码，它的编码电路可用 k 或 $n-k$ 级 2^m 进制移位寄存器实现。该例中 [7, 3, 5]RS 码的编码器，若用 $k=3$ 级乘法电路实现，则如图 7-2 所示。图中的移存器必须是能寄存八进制的元件组成，这可用 3 级触发器组成的移存器完成。 α^2 , α^3 , α^4 常乘器也可用模 2 加法器构成。

现以乘 α^2 为例，说明八进制常乘器的组成。 $GF(2^3)$ 中每个元素均可表示成它的自然基底 1, α , α^2 的线性组合： $a_2\alpha^2 + a_1\alpha + a_0$ ，在乘以 α^2 后，则

$$\begin{aligned} \alpha^2(a_2\alpha^2 + a_1\alpha + a_0) &= a_2\alpha^4 + a_1\alpha^3 + a_0\alpha^2 = a_2(\alpha^2 + \alpha) + a_1(\alpha + 1) + a_0\alpha^2 \\ &= (a_2 + a_0)\alpha^2 + (a_2 + a_1)\alpha + a_1 = a'_2\alpha^2 + a'_1\alpha + a'_0 \end{aligned}$$

式中

$$a'_2 = a_2 + a_0$$

$$a'_1 = a_2 + a_1$$

$$a'_0 = a_1$$

所以，乘 α^2 电路如图 7-3 所示。

由此可把图 7-2 的八进制 [7, 3, 5]RS 码的编码电路，用二进制的部件实现，如图 7-4 所示。它的工作过程如下：

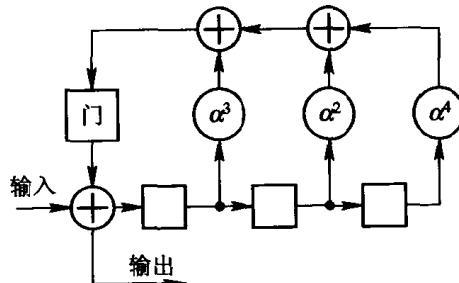


图 7-2 [7,3,5] 八进制 RS 码编码器

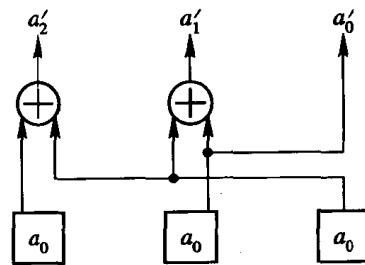


图 7-3 $GF(2^3)$ 中乘 α^2 电路

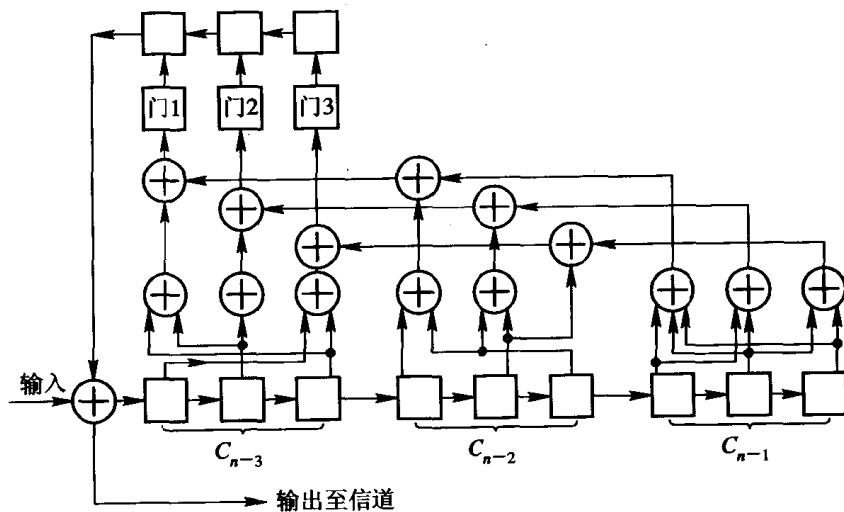


图 7-4 [7, 3, 5]RS 码编码器

(1) 门 1、门 2、门 3 关闭, 所有移存器清洗为 0。然后将 3 个八进制信息符号 $c_{n-1}=c_6$, $c_{n-2}=c_5$, $c_{n-3}=c_4$, 一方面送入八进制寄存器中, 另一方面送入信道。注意每一节拍移动 1 个八进制符号。

(2) 3 个八进制信息元送入后, 门 1、门 2、门 3 打开, 移动一个节拍后, 这时移存器右移一位送出 c_6 , 同时得到第一个校验元 c_3 , 它一方面送至信道, 另一方面存入八进制寄存器的第一级中(最左面)。这时在八进制寄存器中存贮的数据自左至右是 c_3 , c_4 , c_5 。再移动 1 个节拍, 得到第二个校验元 c_2 , 这样依次类推移完 7 个节拍后, 得到了所有 4 个校验元, 并且跟随在信息元后送至信道, 完成了一个码字的编码过程。

(3) 清洗寄存器, 重新关闭门 1、门 2、门 3, 重复(1)、(2)过程, 对第二组信息元进行编码。

如果 $\text{GF}(q)$ 上 RS 码的生成多项式为

$$\begin{aligned} g(x) &= (x - \alpha^0)(x - \alpha^{q+1}) \cdots (x - \alpha^{q^{t-2}}) \\ &= g_0 + g_1 x + \cdots + g_{t-1} x^{q^{t-1}} \\ &= g_0 + g_1 x + \cdots + g_{2t} x^{2t} \end{aligned}$$

这里, $\alpha \in \text{GF}(q)$, $\alpha^{q-1}=1$, 若 $\alpha^e \alpha^{e+2t-1}=1$, 则生成多项式 $g(x)$ 的系数具有如下特点: $g_0=g_{2t}=1$, $g_1=g_{2t-1}$, $g_{t-1}=g_{t+1}$, \cdots 。生成多项式的系数有对称性, $g(x)$ 的互反多项式 $g^*(x)=g(x)$ 。对这种 RS 码的编码器, 只要 t 个乘法器就够了。最近, 林舒等应用域元素的对偶基表示, 使 RS 码的编译码更为简单, 有关这方面的知识可参阅[11]。

二、RS 码的频域编码

设 α 是 $\text{GF}(q)$ 的本原域元素, $M_{K-1}, M_{K-2}, \cdots, M_1, M_0$ 是信息元, $M_i \in \text{GF}(q)$, $0 \leq i \leq K-1$ 。作信息多项式

$$F(x) = M_{K-1}x^{K-1} + M_{K-2}x^{K-2} + \cdots + M_1x + M_0 \quad (7.3.4)$$

把 $\text{GF}(q)$ 中的 $q-1$ 个非 0 元素分别代入上式得

$$\left. \begin{aligned} F(\alpha^{q-2}) &= M_{K-1}(\alpha^{q-2})^{K-1} + M_{K-2}(\alpha^{q-2})^{K-2} + \cdots + M_1\alpha^{q-2} + M_0 \\ F(\alpha^{q-3}) &= M_{K-1}(\alpha^{q-3})^{K-1} + M_{K-2}(\alpha^{q-3})^{K-2} + \cdots + M_1\alpha^{q-3} + M_0 \\ &\vdots \\ F(\alpha) &= M_{K-1}\alpha^{K-1} + M_{K-2}\alpha^{K-2} + \cdots + M_1\alpha + M_0 \\ F(1) &= M_{K-1} + M_{K-2} + \cdots + M_1 + M_0 \end{aligned} \right\} \quad (7.3.5)$$

我们定义下列 $\text{GF}(q)$ 上的 $q-1$ 重:

$$(F(\alpha^{q-2}), F(\alpha^{q-3}), \cdots, F(\alpha), F(1)) \quad (7.3.6)$$

是 RS 码的频域上的一个码字, 与其相应的码多项式是:

$$\begin{aligned} C(x) &= F(\alpha^{q-2})x^{q-2} + F(\alpha^{q-3})x^{q-3} + \cdots + F(\alpha)x + F(1) \\ &= c_{q-2}x^{q-2} + c_{q-3}x^{q-3} + \cdots + c_1x + c_0 \end{aligned} \quad (7.3.7)$$

显然, 它就是信息多项式 $F(x)$ 的 MS 多项式。

由定理 5.8.3 可知, 要证明 RS 码以 $\alpha, \alpha^2, \cdots, \alpha^{q-1}$ 为根, 只要证明码多项式 $C(x)$ 中, $F(\alpha), F(\alpha^2), \cdots, F(\alpha^{q-1})$ 系数均为 0, 其它系数均不为 0 即可。

由定理 5.8.1 可知, 信息多项式 $F(x)$ 的系数

$$M_i = \frac{1}{n} C(\alpha^i) = \frac{1}{q-1} F(\alpha^{-i})$$

$$F(\alpha^{-i}) = (q-1)M_i$$

若 $M_i=0$, 则 $F(\alpha^{-i})=0$; $M_i\neq0$, $F(\alpha^{-i})\neq0$ 。由于信息组 $(M_{K-1}, M_{K-2}, \dots, M_1, M_0)$ 通常不全为 0, 故相应的 $F(\alpha^{-(K-1)})=F(\alpha^{a-K})$, $F(\alpha^{-(K-2)})=F(\alpha^{a-K+1})$, \dots , $F(\alpha^{a-2})$, $F(\alpha^{a-1})$ 均不全为 0。而 $F(x)$ 的其它系数 $M_K, M_{K+1}, \dots, M_{q-2}$ 均为 0, 所以, 相应的 $F(\alpha^{-K})=F(\alpha^{a-K-1})$, $F(\alpha^{-(K+1)})=F(\alpha^{a-K-2})$, \dots , $F(\alpha^{-(q-2)})=F(\alpha)$ 均为 0。由定理 5.8.3 可知, 码字 $C(x)$ 以 $\alpha, \alpha^2, \dots, \alpha^{a-K-1}=a^{a-K}$ 为根, 这相当于码的生成多项式

$$g(x)=(x-\alpha)(x-\alpha^2)\cdots(x-\alpha^{a-K})=(x-\alpha)(x-\alpha^2)\cdots(x-\alpha^{D-1})$$

与用 BCH 码定义的 RS 码完全等价。但用这种频域编码方法得到的 RS 码是非系统码。

例 7.4 仍以 $GF(2^3)$ 上的 $[7, 3, 5]$ RS 码为例。设信息组为 $M_2=1, M_1=0, M_0=0$ 。

作信息多项式 $F(x)=M_2x^2+M_1x+M_0=x^2$, 并求它的 MS 多项式。为此先求 MS 多项式的系数: $F(\alpha^6)=(\alpha^6)^2=\alpha^5, F(\alpha^5)=(\alpha^5)^2=\alpha^3, F(\alpha^4)=(\alpha^4)^2=\alpha, F(\alpha^3)=(\alpha^3)^2=\alpha^6, F(\alpha^2)=(\alpha^2)^2=\alpha^4, F(\alpha)=\alpha^2, F(1)=1$ 。所以 $F(x)$ 的 MS 多项式为

$$\begin{aligned} C(x) &= F(\alpha^6)x^6 + F(\alpha^5)x^5 + \cdots + F(\alpha)x + F(1) \\ &= \alpha^5x^6 + \alpha^3x^5 + \alpha x^4 + \alpha^6x^3 + \alpha^4x^2 + \alpha^2x + 1 \end{aligned}$$

相应的频域码矢是: $(\alpha^5, \alpha^3, \alpha, \alpha^6, \alpha^4, \alpha^2, 1)$ 。

可以验证, 该码字 $C(x)$ 必以 $\alpha, \alpha^2, \alpha^3, \alpha^4$ 为根。如将 α 代入 $C(x)$, 则

$$\begin{aligned} C(\alpha) &= \alpha^5\alpha^6 + \alpha^3\alpha^5 + \alpha\alpha^4 + \alpha^6\alpha^3 + \alpha^4\alpha^2 + \alpha^2\alpha + 1 \\ &= \alpha^4 + \alpha + \alpha^5 + \alpha^2 + \alpha^6 + \alpha^3 + 1 = 0 \end{aligned}$$

同理可得到 $C(\alpha^2)=C(\alpha^3)=C(\alpha^4)=0$ 。

由上讨论可知, RS 码的时域与频域编码在本质上并没有什么不同, 而仅仅是处理编码方式的不同, 但在第八章中我们将看到 RS 码的频域表示与代数几何码密切相关。

如果用二进制符号代替多进制 RS 码中的每个符号, 则 $q=2^m$ 进制的 $[q-1, K, D]$ RS 码变成了 $[m(q-1), Km, D]$ 二进制码。如例 7.3 或例 7.4 中的 $[7, 3, 5]$ 八进制 RS 码, 可变成 $[21, 9, 5]$ 二进制分组码。例如, 若 RS 码的八进制码组是: $(\alpha^5, \alpha^3, \alpha, \alpha^6, \alpha^4, \alpha^2, 1)$, 则相应的 $[21, 9, 5]$ 码的码字为: $(111\ 011\ 010\ 101\ 110\ 100\ 001)$ 。该二进制分组码, 能纠正任何 2 个随机错误, 以及长度 ≤ 4 的突发错误, 和大量的其它错误图样。

这种用二进制符号来表示 2^m 进制 RS 码的方法, 称为由 $GF(2^m)$ 上的码映射成 $GF(2)$ 上的码。由于 $GF(2^m)$ 上的每个元素 α^i , 可以用它的一组自然基底 $1, \alpha, \alpha^2, \dots, \alpha^{m-1}$ 的线性组合表示如下:

$$\alpha^i = a_0 + a_1\alpha + a_2\alpha^2 + \cdots + a_{m-1}\alpha^{m-1} \quad a_i \in GF(2)$$

因此, 这种映射是把线性码映射成线性码。虽然, 循环码不一定映射成循环码, 但一定映射成 $n_0=m$ 的准循环码。

三、RS 码的扩展

由第三章可知, 对线性码增加 1 个全校验位后, 若原码的最小距离是奇数, 则最小距离增加了 1, 否则不会增加。但对 RS 码来说, 增加 1 个全校验位后, 总可使最小距离增加 1。下面的定理说明这点。

定理 7.3.1 令 C_{RS} 是用生成多项式

$$g(x) = (x - \alpha)(x - \alpha^2) \cdots (x - \alpha^{D-1})$$

生成的 $[N=q^m-1, K, D]$ RS 码，则对 C_{RS} 的每一个码字 $C = (c_N, c_{N-1}, \dots, c_2, c_1)$ 加上 1 个全校验位

$$c_0 = - \sum_{i=1}^N c_i$$

后，得到一个 $[N+1, K, D+1]$ 码，称它是原 RS 码的扩展 RS 码。

证明 设 $C \in C_{\text{RS}}$ 是码中最小重量为 D 的非全零码字，现证明进行扩展后，重量增加到 $D+1$ 。如果我们能做到

$$-c_0 = \sum_{i=1}^N c_i \neq 0$$

设 C 的码字多项式

$$C(x) = c_N x^{N-1} + c_{N-1} x^{N-2} + \cdots + c_2 x + c_1 = g(x)a(x)$$

增加 1 个全校验位后，码字 C 变成码字 C' ，相应的码多项式：

$$\begin{aligned} C'(x) &= c_N x^N + c_{N-1} x^{N-1} + \cdots + c_1 x + c_0 \\ C(1) &= c_N + c_{N-1} + \cdots + c_2 + c_1 = -c_0 \end{aligned}$$

所以，要证明 $-c_0$ 不等于 0，只要证明 $C(1) \neq 0$ 即可。由于

$$C(1) = g(1)a(1)$$

显然， $g(1) \neq 0$ 。若 $a(1) = 0$ ，则说明码字 $C(x)$ 含有以 1 为根， $C(x) = b(x)(x-1)g(x)$ ，由 BCH 限可知，码字 C 的重量是 $D+1$ ，这与 C 的重量是 D 相矛盾，故 $a(1) \neq 0$ ， $C(1) = -c_0 \neq 0$ 。所以，扩展 RS 码的最小重量增加了 1。 ■

例如可把例 7.4 中的 $[7, 3, 5]$ RS 码，增加 1 个全校验位后，扩展成 $[8, 3, 6]$ 码，设 $[7, 3, 5]$ 码的一个码字是 $[\alpha^5, \alpha^3, \alpha, \alpha^6, \alpha^4, \alpha^2, 1]$ ，则扩展后成为 $[\alpha^5, \alpha^3, \alpha, \alpha^6, \alpha^4, \alpha^2, 1, c_0]$ ，其中

$$c_0 = -(\alpha^5 + \alpha^3 + \alpha + \alpha^6 + \alpha^4 + \alpha^2 + 1) = 0$$

扩展码的码字成为 $(\alpha^5, \alpha^3, \alpha, \alpha^6, \alpha^4, \alpha^2, 0)$ 。这里， c_0 之所以是 0，是由于码字 $(\alpha^5, \alpha^3, \alpha, \alpha^6, \alpha^4, \alpha^2, 1)$ 不是码中重量最轻者之故。若是一个重量最轻的码字如 $(1, 0, 0, \alpha^4, 1, \alpha^4, \alpha^5)$ ，则扩展后

$$c_0 = -(1 + \alpha^4 + 1 + \alpha^4 + \alpha^5) = \alpha^5$$

得到扩展码的码字是 $(1, 0, 0, \alpha^4, 1, \alpha^4, \alpha^5, \alpha^5)$ ，码字的重量增加了 1。

由式(7.2.5)可知， $[N+1=q^m, K, D+1]$ 扩展 RS 码的校验矩阵

$$H_E = \begin{bmatrix} \alpha^{N-1} & \alpha^{N-2} & \cdots & \alpha & 1 & 0 \\ (\alpha^2)^{N-1} & (\alpha^2)^{N-2} & \cdots & \alpha^2 & 1 & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ (\alpha^{D-1})^{N-1} & (\alpha^{D-1})^{N-2} & \cdots & \alpha^{D-1} & 1 & 0 \\ 1 & 1 & \cdots & 1 & 1 & 1 \end{bmatrix} \quad (7.3.8)$$

由于 RS 码的 $D=(q^m-1)-K+1$ ，且令 $\alpha^i=\alpha_i$ ，则满足

$$\mathbf{H}_{1E} = \begin{bmatrix} \alpha_{N-1} & \alpha_{N-2} & \cdots & \alpha & 1 & 0 \\ \alpha_{N-1}^2 & \alpha_{N-2}^2 & \cdots & \alpha^2 & 1 & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ \alpha_{N-1}^{q^m-K-1} & \alpha_{N-2}^{q^m-K-1} & \cdots & \alpha^{q^m-K-1} & 1 & 0 \\ 1 & 1 & \cdots & 1 & 1 & 1 \end{bmatrix} \quad (7.3.9)$$

是一个 $[q^m, K, q^m-K+1]$ 扩展 RS 码。由 \mathbf{H} 矩阵的性质可知，交换 \mathbf{H} 矩阵的行或列并不会改变码的距离特性，且由 $\alpha^{q^m-1}=1$ ，可知式(7.3.9)的 \mathbf{H}_{1E} 也可写成如下形式：

$$\mathbf{H}_{1E} = \begin{bmatrix} 1 & 1 & \cdots & 1 & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_{q^m-1} & 0 \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_{q^m-1}^2 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ \alpha_1^{q^m-K-1} & \alpha_2^{q^m-K-1} & \cdots & \alpha_{q^m-1}^{q^m-K-1} & 0 \end{bmatrix} \quad (7.3.10)$$

我们还可以对上述的扩展 RS 码再增加 1 个校验位，产生一个 $[q^m+1, K, q^m-K+2]$ 双扩展 RS 码，其校验矩阵是在 \mathbf{H}_{1E} 阵中加上一行一列构成为：

$$\mathbf{H}_{2E} = \begin{bmatrix} 1 & 1 & \cdots & 1 & 1 & 0 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_{q^m-1} & 0 & 0 \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_{q^m-1}^2 & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ \alpha_1^{q^m-K} & \alpha_2^{q^m-K} & \cdots & \alpha_{q^m-1}^{q^m-K} & 0 & 1 \end{bmatrix} \quad (7.3.11)$$

下面只要证明 \mathbf{H}_{2E} 中任何 q^m-K+1 列线性无关，也就是 \mathbf{H}_{2E} 中任何 q^m-K+1 列组成一个满秩矩阵。事实上， \mathbf{H}_{2E} 中除最右两列的任何 q^m-K+1 列组成一个范德蒙矩阵，因而是满秩的。类似地，包含最后一列或二列的任何 q^m-K+1 列矩阵，最后也可以化成一个范德蒙矩阵，因而也是满秩矩阵。

扩张和双扩张 RS 码的每个码字之间，不一定存在着循环关系。但存在着有同样参数的循环码。由上面看出，扩张和双扩张 RS 码也都是极大最小距离码。

下面我们进行另一类扩展。设 $(c_{N-1}, c_{N-2}, \dots, c_0)$ 是 $[N=2^m-1, K, D]$ RS 码的一个码字。把它的每一码元符号映射成二进制 m 重，且对每一 m 重附加 1 个全校验位，则得到以下参数的一个二进制码：

$$n = (m+1)(2^m - 1) \quad k = mK, d \geq 2D = 2(2^m - K) \quad (7.3.12)$$

对任何 $K=1, 2, \dots, 2^m-2$ 都成立。称这类码为 RS 码的映射扩展码。

可把这种映射扩展方法，施行于扩张码，得到一个

$$[(m+1)2^m, mK, d \geq 2(2^m - K + 1)] \quad (7.3.13)$$

二进制分组码，称为 RS 码的双重映射扩展码。

定理 7.3.2 以 $\alpha, \alpha^2, \dots, \alpha^{D-1}$ 为根的 $[N=2^m-1, K, D]$ GF(2^m)上的 RS 码，包含了码长为 N 、距离等于 D 的二进制本原 BCH 码。类似地，扩张 RS 码包含了扩张 BCH 码。

证明 若 C 是 BCH 码的一个二进制码矢，且 $C(\alpha) = C(\alpha^2) = \cdots = C(\alpha^{D-1}) = 0$ ，则 C 必是以 $\alpha, \alpha^2, \dots, \alpha^{D-1}$ 为根的 RS 码的一个码矢。

同理可证，若 C 是二进制 BCH 码的一个码矢，且 $C(1) = C(\alpha) = \cdots = C(\alpha^{D-1}) = 0$ ，

则 C 也必是扩张 RS 码的一个码矢。 ■

所以 RS 码的最小距离至多等于 BCH 码的最小距离。由此我们可以证明，用 RS 码映射成的映射扩张和双重映射扩张码，所构成的长二进制码也并不是好码。

小结 $GF(q^m)$ 上的 $[q^m-1, K, q^m-K]_{q^m}$ 进制 RS 码，是一个 MDS 和本原 BCH 码，其 $g(x) = (x - \alpha^b)(x - \alpha^{b+1}) \cdots (x - \alpha^{b+q^m-K-2})$ (b 通常取 1)， $\alpha \in GF(q^m)$ 是本原域元素。它可以扩展成 $[q^m, K, q^m-K+1]$ 码和 $[q^m+1, K, q^m-K+2]$ 码，以及映射成相应的二进制码。

四、RS 码的推广

我们可以对式(7.3.6)的非系统 RS 码的码字，作进一步的推广。

定义 7.3.2 设 $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_N)$ ， α_i 是 $GF(q^m)$ 中的不同元素。 $v = (v_1, v_2, \dots, v_N)$ ， v_i 是 $GF(q^m)$ 中的非 0 元素(不一定不同)，因而如下所有矢量：

$$(v_1F(\alpha_1), v_2F(\alpha_2), \dots, v_nF(\alpha_N)) \quad (7.3.14)$$

的集合称为广义 RS 码，记以 $GRS_K(\alpha, v)$ ，它也是一类代数几何码。

这里， $F(x)$ 是系数取自 $GF(q^m)$ 上的，次数小于 K 次的信息多项式。由于 $F(x)$ 至多有 $K-1$ 个根，由定理 5.8.3 可知，该码的最小距离至少是 $N-K+1$ ，所以， $GRS_K(\alpha, v)$ 码也是一个 $GF(q^m)$ 上的有极大最小距离的分组码，当 $v_1=v_2=\dots=v_N=1$ ，且 $\alpha_i=\alpha^i$ 时，则广义 RS 码就是普通的 RS 码。

定理 7.3.3 $GRS_K(\alpha, v)$ 码的对偶码也是一个对某一 v' 的 $GRS_{N-K}(\alpha, v')$ 码。

证明 首先假定 $K=N-1$ ，且令 B 码是 $GRS_{N-1}(\alpha, v)$ 码的对偶码。所以，B 码的 $K=1$ ，它的码字必是某一固定矢量 $v' = (v'_1 v'_2 \cdots v'_N)$ 的所有纯量倍数组成。由于它们互为对偶码，由对偶码的性质可知，它们之间每对码字的内积应为 0，若我们能证明满足

($v_1F(\alpha_1), v_2F(\alpha_2), \dots, v_nF(\alpha_N)$) • ($v'_1F'(1), v'_2F'(1), \dots, v'_nF'(1)$) = 0 $\quad (7.3.15)$
的 $v' = (v'_1, v'_2, \dots, v'_N)$ 中的每一个 v'_i 均不为 0，则由定义 7.3.2 可知，B 码必是一个 $GRS_1(\alpha, v')$ 广义 RS 码。

由式(7.3.15)得

$$v_1F(\alpha_1)v'_1F'(1) + v_2F(\alpha_2)v'_2F'(1) + \cdots + v_nF(\alpha_N)v'_nF'(1) = 0$$

注意到 $F'(1) = 1$ ，因而上式可写成：

$$\left\{ \begin{array}{l} v_1v'_1(1 + \alpha_1 + \alpha_1^2 + \cdots + \alpha_1^{N-2}) + \\ v_2v'_2(1 + \alpha_2 + \alpha_2^2 + \cdots + \alpha_2^{N-2}) + \\ \vdots \\ v_nv'_N(1 + \alpha_N + \alpha_N^2 + \cdots + \alpha_N^{N-2}) = 0 \end{array} \right.$$

或

$$\left\{ \begin{array}{l} v_1v'_1 + v_2v'_2 + \cdots + v_nv'_N = 0 \\ \alpha_1v_1v'_1 + \alpha_2v_2v'_2 + \cdots + \alpha_Nv_Nv'_N = 0 \\ \vdots \\ \alpha_1^{N-2}v_1v'_1 + \alpha_2^{N-2}v_2v'_2 + \cdots + \alpha_N^{N-2}v_Nv'_N = 0 \end{array} \right. \quad (7.3.16)$$

写成矩阵形式为

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_N \\ \vdots & \vdots & & \vdots \\ \alpha_1^{N-2} & \alpha_2^{N-2} & \cdots & \alpha_N^{N-2} \end{bmatrix} \begin{bmatrix} v_1 v'_1 \\ v_2 v'_2 \\ \vdots \\ v_N v'_N \end{bmatrix} = \mathbf{0} \quad (7.3.17)$$

如果某一个 v'_i , 例如 v'_1 等于 0, 则上式成为:

$$\begin{cases} v_2 v'_2 + v_3 v'_3 + \cdots + v_N v'_N = 0 \\ \alpha_2 v_2 v'_2 + \alpha_3 v_3 v'_3 + \cdots + \alpha_N v_N v'_N = 0 \\ \vdots \\ \alpha_2^{N-2} v_2 v'_2 + \alpha_3^{N-2} v_3 v'_3 + \cdots + \alpha_N^{N-2} v_N v'_N = 0 \end{cases}$$

或

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha_2 & \alpha_3 & \cdots & \alpha_N \\ \vdots & \vdots & & \vdots \\ \alpha_2^{N-2} & \alpha_3^{N-2} & \cdots & \alpha_N^{N-2} \end{bmatrix} \begin{bmatrix} v_2 v'_2 \\ v_3 v'_3 \\ \vdots \\ v_N v'_N \end{bmatrix} = \mathbf{0}$$

上式左边第一个方阵是范德蒙矩阵, 因而要使上式成立, 仅只有所有的 $v_i v'_i = 0$, 但 $v_i \neq 0$, 所以, 要求所有的 $v'_i = 0$, 而这是不可能的, 因而所有的 v'_i 均不为 0, 故 D 是一个 $\text{GRS}_1(\alpha, v')$ 码。

对所有 $K < N - 1$, 与式(7.3.16)类似可得

$$\sum_{i=1}^N (\alpha_i v_i) (\alpha_i' v'_i) = \sum_{i=1}^N \alpha_i^{s+t} v_i v'_i = 0 \quad s \leq K-1, t \leq N-K-1 \quad (7.3.18)$$

同理, 可证所有 v'_i 不为 0, 因而 $\text{GRS}_K(\alpha, v)$ 的对偶码是 $\text{GRS}_{N-K}(\alpha, v')$ 码。 ■

由定理 7.3.3 可知, $\text{GRS}_K(\alpha, v)$ 的校验矩阵就是 $\text{GRS}_{N-K}(\alpha, v')$ 的生成矩阵:

$$\begin{aligned} & \begin{bmatrix} v'_1 & \cdots & v'_N \\ \alpha_1 v'_1 & \cdots & \alpha_N v'_N \\ \vdots & & \vdots \\ \alpha_1^{N-K-1} v'_1 & \cdots & \alpha_N^{N-K-1} v'_N \end{bmatrix} \\ &= \begin{bmatrix} 1 & \cdots & 1 \\ \alpha_1 & \cdots & \alpha_N \\ \vdots & & \vdots \\ \alpha_1^{N-K-1} & \cdots & \alpha_N^{N-K-1} \end{bmatrix} \begin{bmatrix} v'_1 & 0 \\ v'_2 & \ddots \\ 0 & v'_N \end{bmatrix} \quad (7.3.19) \end{aligned}$$

五、子域子码

如果 $\text{GF}(q^m)$ 上的码 C_q^m 中的某些码字集合 C_{qs}^m , 它们组成了 C_q^m 中的一个子群, 则称 C_{qs}^m 是 C_q^m 中的一个子码。

如 $\text{GF}(2)$ 上的由 $g(x) = x^3 + x + 1$ 生成的 $[7, 4, 3]$ 汉明码 C_2^1 , 它的 16 个码字集合中, 由 $g'(x) = (x+1)(x^3+x+1)$ 生成的 8 个码字集合 $C_{2s}^1 \subset C_2^1$, 有偶数重量, 它就是 C_2^1 中的一个子群。因此, C_{2s}^1 码是 C_2^1 码的一个子码, 其实就是 $[7, 3, 4]$ 增余删信汉明码。

定义 7.3.3 若 C_q^m 是 $\text{GF}(q^m)$ 上的 $[n, k, d]$ 码, 如果在 C_q^m 中的、码字分量均在 $\text{GF}(q)$

上码字集合 C_{qs} 是 C_q^m 中的一个子群，则称 $C_{qs} \subset C_q^m$ 是 C_q^m 码的一个子域子码，这是一个 $[n, k', d']$ 码。这里， $n - m(n - k) \leq k' \leq k$, $d' \geq d$ 。

例如， $\text{GF}(2^3)$ 上的由 $g(x) = x + \alpha$ 生成的 $[7, 6, 2]$ RS 码 C_2^3 ，这里， $\alpha \in \text{GF}(2^3)$ 是本原元，其中码字 $g'(x) = g(x)(x + \alpha^2)(x + \alpha^4) = (x + \alpha)(x + \alpha^2)(x + \alpha^4) = x^3 + x + 1$ ，在 C_2^3 码的子码 C_{2^3} 中。所以，由 $g'(x)$ 生成的 $\text{GF}(2)$ 上的码就是 C_2^3 码的子域子码 C_{2^3} ，它其实也就是 $[7, 4, 3]$ 汉明码。

定理 7.3.4(Delsarte)^[18] 子域子码的对偶码是原码对偶码的迹，即

$$C_{qs}^\perp = \text{Tr}(C_q^{m\perp}) \quad (7.3.20)$$

可知，若原码的码字 $(c_{n-1}, c_{n-2}, \dots, c_0) \in C_q^m$ ，它的对偶码的码字为 $(c'_{n-1}, c'_{n-2}, \dots, c'_0) \in C_q^{m\perp}$ ，则与其相对应的子域子码 C_{qs} 的对偶码 C_{qs}^\perp 中的码字是

$$(\text{Tr}(c'_{n-1}), \text{Tr}(c'_{n-2}), \dots, \text{Tr}(c'_0)) \in C_{qs}^\perp$$

例如，由生成多项式 $g(x) = (x + \alpha^6)$ 产生的 $[7, 6, 2]$ RS 码， $\alpha \in \text{GF}(2^3)$ 是本原元。它的对偶码的生成多项式 $g'(x) = (x^7 + 1)/(x + \alpha^6) = x^6 + \alpha^6 x^5 + \alpha^5 x^4 + \alpha^4 x^3 + \alpha^3 x^2 + \alpha^2 x + \alpha$ ，而 $\text{Tr}(g'(x)) = x^6 + x^5 + x^3 + 1$ ，它就是 $[7, 6, 2]$ RS 码的子域子码 $[7, 4, 3]$ 码的对偶码 $[7, 3, 4]$ 码的一个码字： $(10\ 01\ 10\ 11) = (\text{Tr}(1), \text{Tr}(\alpha^6), \text{Tr}(\alpha^5), \text{Tr}(\alpha^4), \text{Tr}(\alpha^3), \text{Tr}(\alpha^2), \text{Tr}(\alpha))$ 。

子码和子域子码的概念，在研究码的最小距离、重量分布和译码时有重要作用。

由上面讨论可以看出，GRS 码是一类范围非常广的码，在式(7.3.19)中的 α_i, v_i 取不同的值时就得到了不同的码类，如以后将介绍的交替码就是 GRS 码的一类子域子码。

§ 7.4 BCH 码的一般译码方法

BCH 码的译码问题，一直是编码理论研究中最感兴趣的课题之一。一个码能否在实际中应用，往往取决于译码器是否简单、快速、经济和译码错误概率小。BCH 码在短和中等码长下，具有很好的纠错性能，构造容易，故在实际中得到广泛应用。此外，在以后将看到，BCH 码与其它各类码有极其密切的关系，因此研究 BCH 码的译码无论在实际上，还是在理论上都有着重要意义。

1960 年彼得逊奠定了二进制 BCH 码译码的理论基础，稍后戈雷(Gore)和齐勒尔(Zierler)推广到多进制情况。1965 年福尼(Forney)解决了 BCH 码的纠错纠删码。1966 年伯利坎普提出了迭代译码算法，节省了计算量，加快了译码速度，因而从实际上解决了 BCH 码的译码问题。自此以后，很多作者又不断提出了新的译码方法如欧几里德、连分式等译码算法。1978 年勃莱哈特用数字信号处理技术中常用的频谱方法，基于码的 MS 多项式，提出了频域译码。自 70 年代中以后，很多作者研究了超 BCH 码限的超设计距离的译码问题，如能达到 HT 限和 Roos 限的译码问题。但目前用得最普遍、实用上最重要的还是时域中的迭代译码法。这一节我们将讨论 BCH 码的译码理论，下一节介绍迭代译码算法、频域译码和超设计距离译码。

一、BCH 码译码的基本概念

BCH 码的译码与第三章讲的线性码的译码步骤相同分为 3 步：第一步是由接收到的

$R(x)$ 计算出伴随式 S ; 第二步由伴随式找出错误图样 $\hat{E}(x)$; 第三步由 $R(x) - \hat{E}(x)$ 得到最可能发送的码字 $\hat{C}(x)$, 完成译码。

设 $GF(q)$ 上的 $[n, k, d]$ BCH 码以 $\alpha^{m_0}, \alpha^{m_0+1}, \dots, \alpha^{m_0+2t-1}$ 为根, 则它的生成多项式

$$g(x) = (x - \alpha^{m_0})(x - \alpha^{m_0+1}) \cdots (x - \alpha^{m_0+2t-1}) \quad d = 2t + 1$$

$\alpha \in GF(q^m)$ 为 n 级域元素。发送的码字 $C(x) = q(x)g(x)$, 接收的 n 重为 $R(x) = C(x) + E(x)$ 。设错误图样

$$E(x) = e_{n-1}x^{n-1} + e_{n-2}x^{n-2} + \cdots + e_1x + e_0$$

若信道产生 t 个错误, 则

$$E(x) = Y_t x^{l_t} + Y_{t-1} x^{l_{t-1}} + \cdots + Y_1 x^{l_1} = \sum_{i=1}^t Y_i x^{l_i}$$

式中, $Y_i \in GF(q)$, x^{l_i} 称为错误位置数, 说明错误发生在 $R(x)$ 中的第 $n-l_i$ (x^{n-1} 的系数算第一位) 位, 错误值是 Y_i 。如果 $R(x)$ 中有 γ 个错误, 则 $E(x)$ 共有 γ 项 $Y_i x^{l_i}$, $i=1, 2, \dots, \gamma$ 。

由伴随式定义和式(7.1.3)可知:

$$S^T = H \cdot R^T = H \cdot E^T$$

$$\begin{aligned} &= \begin{bmatrix} (\alpha^{m_0})^{n-1} & (\alpha^{m_0})^{n-2} & \cdots & \alpha^{m_0} & 1 \\ (\alpha^{m_0+1})^{n-1} & (\alpha^{m_0+1})^{n-2} & \cdots & \alpha^{m_0+1} & 1 \\ \vdots & \vdots & & \vdots & \vdots \\ (\alpha^{m_0+2t-1})^{n-1} & (\alpha^{m_0+2t-1})^{n-2} & \cdots & \alpha^{m_0+2t-1} & 1 \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ Y_t \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} Y_1(\alpha^{m_0})^{l_1} + Y_2(\alpha^{m_0})^{l_2} + \cdots + Y_t(\alpha^{m_0})^{l_t} \\ Y_1(\alpha^{m_0+1})^{l_1} + Y_2(\alpha^{m_0+1})^{l_2} + \cdots + Y_t(\alpha^{m_0+1})^{l_t} \\ \vdots \\ Y_1(\alpha^{m_0+2t-1})^{l_1} + Y_2(\alpha^{m_0+2t-1})^{l_2} + \cdots + Y_t(\alpha^{m_0+2t-1})^{l_t} \end{bmatrix} \\ &= \begin{bmatrix} s_{m_0} = R(\alpha^{m_0}) = E(\alpha^{m_0}) \\ s_{m_0+1} = R(\alpha^{m_0+1}) = E(\alpha^{m_0+1}) \\ \vdots \\ s_{m_0+2t-1} = R(\alpha^{m_0+2t-1}) = E(\alpha^{m_0+2t-1}) \end{bmatrix} \quad (7.4.1) \end{aligned}$$

式中

$$s_j = \sum_{i=1}^t Y_i (\alpha^j)^{l_i} = R(\alpha^j) \quad j = m_0, m_0 + 1, \dots, m_0 + 2t - 1$$

若令 $(\alpha^{m_0+i})^{l_i} = x_i^{m_0+i}$, 则上式可写成

$$s_j = \sum_{i=1}^t Y_i x_i^j \quad j = m_0, m_0 + 1, \dots, m_0 + 2t - 1 \quad (7.4.2)$$

或

$$\begin{aligned} s_{m_0} &= Y_1 x_1^{m_0} + Y_2 x_2^{m_0} + \dots + Y_t x_t^{m_0} = \sum_{k=1}^t Y_k x_k^{m_0} \\ s_{m_0+1} &= Y_1 x_1^{m_0+1} + Y_2 x_2^{m_0+1} + \dots + Y_t x_t^{m_0+1} = \sum_{k=1}^t Y_k x_k^{m_0+1} \\ &\vdots \\ s_{m_0+2t-1} &= Y_1 x_1^{m_0+2t-1} + \dots + Y_t x_t^{m_0+2t-1} = \sum_{k=1}^t Y_k x_k^{m_0+2t-1} \end{aligned} \quad (7.4.3)$$

通常情况下取 $m_0=1$, 则

$$\begin{aligned} s_1 &= Y_1 x_1 + Y_2 x_2 + \dots + Y_t x_t = \sum_{k=1}^t Y_k x_k \\ s_2 &= Y_1 x_1^2 + Y_2 x_2^2 + \dots + Y_t x_t^2 = \sum_{k=1}^t Y_k x_k^2 \\ &\vdots \\ s_{2t} &= Y_1 x_1^{2t} + Y_2 x_2^{2t} + \dots + Y_t x_t^{2t} = \sum_{k=1}^t Y_k x_k^{2t} \end{aligned} \quad (7.4.4)$$

称式中的 s_j 为 x 的加权幂和对称函数。

我们的目的是要由式(7.4.4)中的 $2t$ 个方程求出 $2t$ 个未知数 $x_i, Y_i, (i=1, 2, \dots, t)$ 。要直接解上述方程比较困难, 所以分两步进行, 先解出错误位置数 x_i , 再求错误值 Y_i 。为此引入错误位置多项式

$$\sigma(x) = (1 - x_1 x)(1 - x_2 x) \cdots (1 - x_t x)$$

若第 k 个错误位置 $x=x_k^{-1}$, 则 $\sigma(x_k^{-1})=0$ 。因此, 求错误位置就是求解错误位置多项式 $\sigma(x)$ 的根。

把 $\sigma(x)$ 展开:

$$\begin{aligned} \sigma(x) &= (1 - x_1 x)(1 - x_2 x) \cdots (1 - x_t x) = \prod_{i=1}^t (1 - x_i x) \\ &= 1 - (x_1 + x_2 + \cdots + x_t)x + (x_1 x_2 + x_1 x_3 + \cdots + x_{t-1} x_t)x^2 \\ &\quad - \cdots + (-1)^t x_1 x_2 \cdots x_t x^t \end{aligned} \quad (7.4.5)$$

令

$$\begin{aligned} \sigma_1 &= -(x_1 + x_2 + \cdots + x_t) \\ \sigma_2 &= (x_1 x_2 + x_1 x_3 + \cdots + x_{t-1} x_t) \\ &\vdots \\ \sigma_t &= (-1)^t x_1 x_2 \cdots x_t \end{aligned} \quad (7.4.6)$$

则式(7.4.5)成为

$$\sigma(x) = 1 + \sigma_1 x + \sigma_2 x^2 + \cdots + \sigma_t x^t = \prod_{i=1}^t (1 - x_i x) \quad (7.4.7)$$

称式(7.4.6)是 x_i 的初等幂和对称函数。若 x_k^{-1} 为错误位置, 则

$$\sigma(x_k^{-1}) = 1 + \sigma_1 x_k^{-1} + \sigma_2 x_k^{-2} + \cdots + \sigma_t x_k^{-t} = 0$$

两边乘以 x_k^t , 则

$$x_k^t + \sigma_1 x_k^{t-1} + \sigma_2 x_k^{t-2} + \cdots + \sigma_t = 0 \quad k = 1, 2, \dots, t$$

上式两边再乘以 $Y_k x_k^j$, $j = m_0, m_0+1, \dots, m_0+t-1$, 则

$$Y_k x_k^{j+t} + \sigma_1 Y_k x_k^{j+t-1} + \cdots + \sigma_t Y_k x_k^j = 0 \quad k = 1, 2, \dots, t$$

对 k 求和得

$$\sum_{k=1}^t Y_k x_k^{j+t} + \sigma_1 \sum_{k=1}^t Y_k x_k^{j+t-1} + \cdots + \sigma_t \sum_{k=1}^t Y_k x_k^j = 0 \quad k = 1, 2, \dots, t$$

由式(7.4.2) s_j 的定义知上式成为

$$s_{j+t} + \sigma_1 s_{j+t-1} + \cdots + \sigma_t s_j = 0 \quad j = m_0, m_0+1, \dots, m_0+t-1 \quad (7.4.8)$$

把上式展开, 则

$$s_{m_0+t} + \sigma_1 s_{m_0+t-1} + \sigma_2 s_{m_0+t-2} + \cdots + \sigma_t s_{m_0} = 0$$

$$s_{m_0+1+t} + \sigma_1 s_{m_0+1+t} + \sigma_2 s_{m_0+1+t-1} + \cdots + \sigma_t s_{m_0+1} = 0$$

⋮

$$s_{m_0+2t-1} + \sigma_1 s_{m_0+2t-2} + \sigma_2 s_{m_0+2t-3} + \cdots + \sigma_t s_{m_0+t-1} = 0 \quad (7.4.9)$$

或

$$\begin{bmatrix} s_{m_0+t-1} & s_{m_0+t-2} & \cdots & s_{m_0} \\ s_{m_0+t} & s_{m_0+t-1} & \cdots & s_{m_0+1} \\ \vdots & \vdots & & \vdots \\ s_{m_0+2t-2} & s_{m_0+2t-3} & \cdots & s_{m_0+t-1} \end{bmatrix} \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_t \end{bmatrix} = - \begin{bmatrix} s_{m_0+t} \\ s_{m_0+t+1} \\ \vdots \\ s_{m_0+2t-1} \end{bmatrix} \quad (7.4.10)$$

$$[\mathbf{M}][\boldsymbol{\sigma}] = - [\mathbf{S}]$$

式(7.4.9)是一组线性方程, 有 t 个方程和 t 个未知数, 该方程组有解的充要条件是式(7.4.10)中的 \mathbf{M} 矩阵满秩。若实际产生的错误个数为 t , 则下面定理证明系数矩阵 \mathbf{M} 为满秩。

定理 7.4.1 如果 $s_j (j = m_0, m_0+1, \dots, m_0+t-1)$ 是由 t 个不同的非零数对 (x_i, Y_i) 组成, 则矩阵

$$\begin{bmatrix} s_{m_0+t-1} & s_{m_0+t-2} & \cdots & s_{m_0} \\ s_{m_0+t} & s_{m_0+t-1} & \cdots & s_{m_0+1} \\ \vdots & \vdots & & \vdots \\ s_{m_0+2t-2} & s_{m_0+2t-3} & \cdots & s_{m_0+t-1} \end{bmatrix} \quad (7.4.11)$$

满秩, 否则为非满秩。

证明 将式(7.4.3)中的 s_{m_0+j} 代入 \mathbf{M}

$$\mathbf{M} = \begin{bmatrix} \Sigma Y_k x_k^{m_0+t-1} & \Sigma Y_k x_k^{m_0+t-2} & \cdots & \Sigma Y_k x_k^{m_0} \\ \Sigma Y_k x_k^{m_0+t} & \Sigma Y_k x_k^{m_0+t-1} & \cdots & \Sigma Y_k x_k^{m_0+1} \\ \vdots & \vdots & & \vdots \\ \Sigma Y_k x_k^{m_0+2t-2} & \Sigma Y_k x_k^{m_0+2t-3} & \cdots & \Sigma Y_k x_k^{m_0+t-1} \end{bmatrix}$$

$$\begin{aligned}
&= \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_t \\ \vdots & \vdots & & \vdots \\ x_1^{t-1} & x_2^{t-1} & \cdots & x_t^{t-1} \end{bmatrix} \begin{bmatrix} Y_1 x_1^{m_0} & & & \\ & Y_2 x_2^{m_0} & & \\ & & \mathbf{0} & \\ & & & Y_t x_t^{m_0} \end{bmatrix} \begin{bmatrix} 1 & x_1 & \cdots & x_1^{t-1} \\ 1 & x_2 & \cdots & x_2^{t-1} \\ \vdots & \vdots & & \vdots \\ 1 & x_t & \cdots & x_t^{t-1} \end{bmatrix} \\
&= [\mathbf{x}] [\mathbf{Yx}] [\mathbf{x}]^T
\end{aligned}$$

式中, $[\mathbf{x}]$ 和 $[\mathbf{x}]^T$ 是范德蒙矩阵, 只要每个元素不为 0 且不同, 则它们的行列式不为 0。而 $[\mathbf{Yx}]$ 矩阵, 仅只有所有数对 $Y_i x_i^{m_0}$ 不为 0 时, 行列式不为 0。若发生 t 个错误, 则有 t 个不同且不为 0 的 x_i 和 Y_i , 因而 \mathbf{M} 矩阵满秩, 否则不满秩。 ■

如果接收的 $R(x)$ 中, 仅有 $\gamma < t$ 个错误, 则 $s_j = \sum_{k=1}^{\gamma} Y_k x_k^j$, $t \times t$ 阶 \mathbf{M} 矩阵非满秩, 就必须把 \mathbf{M} 矩阵降至 $\gamma \times \gamma$ 阶满秩为止, 才能求得 $\sigma_1, \sigma_2, \dots, \sigma_r$ 未知数。

例 7.5 二进制 $[15, 5, 7]$ BCH 码。它以 $\alpha, \alpha^3, \alpha^5$ 为根, $g(x) = m_1(x)m_3(x)m_5(x) = x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1$ 。接收的 $R(x) = x^{10} + x^3$, 求发出的码字 $C(x)$, 这里 α 是 $p(x) = x^4 + x + 1$ 的根。

首先计算伴随式

$$S^T = \mathbf{H} \cdot R^T = \mathbf{H} \cdot E^T = \begin{bmatrix} \alpha^{14} & \alpha^{13} & \cdots & \alpha & 1 \\ (\alpha^3)^{14} & (\alpha^3)^{13} & \cdots & \alpha^3 & 1 \\ (\alpha^5)^{14} & (\alpha^5)^{13} & \cdots & \alpha^5 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{bmatrix}$$

式中, $s_1 = \alpha^{10} + \alpha^3 = \alpha^{12}$, $s_3 = \alpha^7$, $s_5 = \alpha^{10}$, 由二进制 BCH 码的编码理论可知, 若码以 α 为根, 则必以 $\alpha^2, \alpha^4, \dots, \alpha^{2^{m-1}}$ 为根, 因而

$$s_2 = s_1^2 = \alpha^9, s_4 = s_2^2 = \alpha^3, s_6 = s_3^2 = \alpha^{14}$$

上面 α^i 和 α^j 之间的运算按表 4-2 所给出的值进行(下同)。把求得的 s_i 值代入式(7.4.10)中, 并取 $m_0=1$, 得

$$\begin{bmatrix} s_3 & s_2 & s_1 \\ s_4 & s_3 & s_2 \\ s_5 & s_4 & s_3 \end{bmatrix} \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \end{bmatrix} = \begin{bmatrix} s_4 \\ s_5 \\ s_6 \end{bmatrix}$$

首先，计算 3×3 阶的 M 矩阵行列式是否为 0，

$$\begin{vmatrix} s_3 & s_2 & s_1 \\ s_4 & s_3 & s_2 \\ s_5 & s_4 & s_3 \end{vmatrix} = \begin{vmatrix} \alpha^7 & \alpha^9 & \alpha^{12} \\ \alpha^3 & \alpha^7 & \alpha^9 \\ \alpha^{10} & \alpha^3 & \alpha^7 \end{vmatrix} = 0$$

因此实际产生的错误个数小于 3， $\sigma_3=0$ ， 3×3 阶 M 矩阵必须降至 2×2 阶矩阵，得到

$$\begin{bmatrix} s_3 & s_2 \\ s_4 & s_3 \end{bmatrix} \begin{bmatrix} \sigma_1 \\ \sigma_2 \end{bmatrix} = \begin{bmatrix} s_4 \\ s_5 \end{bmatrix}$$

再计算

$$\begin{vmatrix} s_3 & s_2 \\ s_4 & s_3 \end{vmatrix} = \begin{vmatrix} \alpha^7 & \alpha^9 \\ \alpha^3 & \alpha^7 \end{vmatrix} = \alpha^5 \neq 0$$

所以矩阵满秩，方程组有解，经计算求得：

$$\sigma_1 = \alpha^{12} \quad \sigma_2 = \alpha^{13}$$

代入错误位置多项式式(7.4.7)可得：

$$\sigma(x) = 1 + \alpha^{12}x + \alpha^{13}x^2$$

解得它的 2 个根 $x_1^{-1}=\alpha^{-3}$, $x_2^{-1}=\alpha^{-10}$ ；相应的错误位置数 $x_1=\alpha^3$, $x_2=\alpha^{10}$ 。由此可知，第 5 位(即 x^{10} 次位)和第 12 位(即 x^3 次位)产生了错误。因此 $\hat{E}(x)=x^{10}+x^3$, $\hat{C}(x)=R(x)-\hat{E}(x)=0$ ，发的码字是一个全为 0 码字。 ■

通常情况下，若产生一个错误， $t=1$ ，取 $m_0=1$ ，则由式(7.4.10)可得：

$$\begin{aligned} s_1\sigma_1 &= s_2 \\ \sigma_1 &= -s_2/s_1 \\ \sigma(x) &= 1 + (-s_2/s_1)x \end{aligned} \tag{7.4.12}$$

若产生 2 个错误，则由式(7.4.10)可得：

$$\begin{aligned} \begin{bmatrix} s_2 & s_1 \\ s_3 & s_2 \end{bmatrix} \begin{bmatrix} \sigma_1 \\ \sigma_2 \end{bmatrix} &= \begin{bmatrix} s_3 \\ s_4 \end{bmatrix} \\ \sigma_1 &= \frac{s_3s_2 - s_1s_4}{s_2^2 - s_1s_3} \quad \sigma_2 = \frac{s_2s_4 - s_3^2}{s_2^2 - s_2s_3} \\ \sigma(x) &= 1 + \sigma_1x + \sigma_2x^2 \end{aligned} \tag{7.4.13}$$

求得了 $\sigma(x)$ ，解出它的根得到错误位置数 x_1, x_2, \dots, x_t 以后，把它们的值代入式(7.4.3)，当 $m_0=1$ 时，代入式(7.4.4)得：

$$\begin{bmatrix} x_1 & x_2 & \cdots & x_t \\ x_1^2 & x_2^2 & \cdots & x_t^2 \\ \vdots & \vdots & & \vdots \\ x_1^t & x_2^t & \cdots & x_t^t \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_t \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_t \end{bmatrix}$$

或

$$[\mathbf{x}][\mathbf{Y}] = [\mathbf{s}]$$

上式的 $[\mathbf{x}]$ 矩阵是范德蒙矩阵，只有 $x_1 \neq x_2 \neq \cdots \neq x_t$ ，且不为 0，则上述方程组有解，可求出错误值

$$Y_i = \begin{vmatrix} x_1 & \cdots & s_1 x_{i+1} & \cdots & x_t \\ x_1^2 & \cdots & s_2 x_{i+1}^2 & \cdots & x_t^2 \\ \vdots & & \vdots & & \vdots \\ x_1^t & \cdots & s_t x_{i+1}^t & \cdots & x_t^t \end{vmatrix}_{|x|}$$

$$i = 1, 2, \dots, t; m_0 = 1 \quad (7.4.14)$$

在二进制情况下，错误值只可能是 1，故求 Y_i 这一步可以省略。

二、钱(Chien)搜索和伴随式计算电路

用式(7.4.10)求得 $\sigma(x)$ 后，下一步的问题是从工程观点看，如何简单地求出它的根即错误位置。1964 年钱闻天提出了一个求 $\sigma(x)$ 根的实用方法，解决了这个问题。

解 $\sigma(x)$ 的根，就是确定 $R(x) = r_{n-1}x^{n-1} + r_{n-2}x^{n-2} + \dots + r_1x + r_0$ ，为了要检验第一位 r_{n-1} 是否错误，相当于译码器要确定 α^{n-1} 是否是错误位置数，这等于检验 $\alpha^{-(n-1)}$ 是否是 $\sigma(x)$ 的根。若 $\alpha^{-(n-1)} = \alpha$ 是 $\sigma(x)$ 的根，则

$$\sigma(\alpha^{-(n-1)}) = \sigma(\alpha) = 1 + \sigma_1\alpha + \sigma_2\alpha^2 + \dots + \sigma_t\alpha^t = 0$$

或

$$\sigma_1\alpha + \sigma_2\alpha^2 + \dots + \sigma_t\alpha^t = -1$$

所以得到了 $\sigma(x)$ 后，为了译 r_{n-1} ，译码器首先计算 $\sigma_1\alpha, \sigma_2\alpha^2, \dots, \sigma_t\alpha^t$ ，然后计算它们的和是否为 -1 。若是，则 α^{n-1} 是错误位置数， r_{n-1} 码元是错误的；否则 r_{n-1} 是正确的。换言之

$$\begin{aligned} \sigma_1\alpha + \sigma_2\alpha^2 + \dots + \sigma_t\alpha^t = -1, & \quad r_{n-1} \text{ 有错} \\ \sigma_1\alpha + \sigma_2\alpha^2 + \dots + \sigma_t\alpha^t \neq -1, & \quad r_{n-1} \text{ 正确} \end{aligned} \quad (7.4.15)$$

同理，为了译 r_{n-l} 译码器必须计算 $\sigma_1x^l, \sigma_2x^{2l}, \dots, \sigma_tx^{tl}$ ，并计算它们的和，若

$$\begin{aligned} \sigma_1\alpha^l + \sigma_2\alpha^{2l} + \dots + \sigma_t\alpha^{tl} = -1 & \quad r_{n-l} \text{ 有错} \\ \sigma_1\alpha^l + \sigma_2\alpha^{2l} + \dots + \sigma_t\alpha^{tl} \neq -1 & \quad r_{n-l} \text{ 正确} \end{aligned} \quad (7.4.16)$$

这样依次对每一个 r_{n-l} ($l=1, 2, \dots, n$) 进行检验，就求得了 $\sigma(x)$ 的根，这个过程称为钱搜索。

在工程上，钱搜索过程可用图 7-5 所示的电路实现，它的工作过程如下：

- (1) t 个寄存器寄存 $\sigma_1, \sigma_2, \dots, \sigma_t$ ，当错误个数 $\gamma < t$ ，则 $\sigma_{\gamma+1} = \sigma_{\gamma+2} = \dots = \sigma_t = 0$ 。
- (2) r_{n-1} 正要从缓冲存贮器读出之前， t 个乘法器由移位脉冲控制行乘法运算，且 $\sigma_1\alpha, \sigma_2\alpha^2, \dots, \sigma_t\alpha^t$ 存在 σ 寄存器中，并送入 A 中进行

$$\sigma_1\alpha + \sigma_2\alpha^2 + \dots + \sigma_t\alpha^t$$

运算和检验。若等于 -1 ，则 A 输出一个信号，控制门打开，把错误值 Y_{n-1} 与缓存器输出的 r_{n-1} 相减，得到 $r_{n-1} - Y_{n-1} = c_{n-1}$ 。在二进制情况下，A 输出 1 与 r_{n-1} 进行模 2 加，即完成对 r_{n-1} 的纠错。

- (3) r_{n-1} 译完后，再进行一次相乘，此时 $\sigma_1\alpha^2, \sigma_2(\alpha^2)^2, \dots, \sigma_t(\alpha^t)^2$ 存在 σ 寄存器中，并在 A 中进行相加运算和检验，对 r_{n-2} 进行纠错。

- (4) 其余码元同(2)一样纠错。

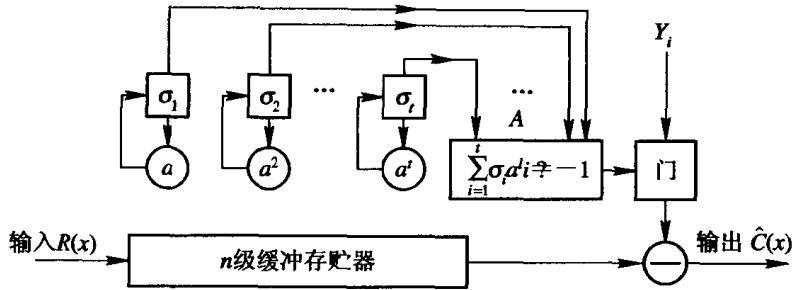


图 7-5 钱搜索电路图

由上面讨论可知, BCH 码的译码过程可分以下几步:

- (1) 由接收到的 $R(x)$, 求得 $s_j = \sum_i Y_i x_i^j$, $j = m_0, m_0 + 1, \dots, m_0 + 2t - 1$ 。
- (2) 由 s_i 求出错误位置多项式 $\sigma(x)$ 。
- (3) 用钱搜索解出 $\sigma(x)$ 的根, 得到错误位置数, 确定出错误位置。
- (4) 由错误位置数求得错误值, 从而得到错误图样 $\hat{E}(x)$ 。
- (5) $R(x) - \hat{E}(x) = \hat{C}(x)$, 完成纠错过程。

上述译码过程中, 第(1)步求伴随式 s_i , 在工程上实现比较简单。设码的生成多项式

$$g(x) = m_1(x)m_2(x)\cdots m_{2t}(x)$$

式中, $m_i(x)$ 是 α 的最小多项式, 则

$$R(x) = q_j(x)m_j(x) + r_j(x)$$

由于 $m_j(x)$ 是以 α^j 为根, 因而当 $x=\alpha^j$ 时, 则由式(7.4.2)可知:

$$s_j = R(\alpha^j) = q_j(\alpha^j)m_j(\alpha^j) + r_j(\alpha^j) = r_j(\alpha^j)$$

下面用例 7.5 中的二进制 [15, 5] 码为例, 说明求 s_j 的电路。 $[15, 5]$ 码的 $g(x) = m_1(x)m_3(x)m_5(x)$, $m_1(x) = x^4 + x + 1$, $m_3(x) = x^4 + x^3 + x^2 + x + 1$, $m_5(x) = x^2 + x + 1$ 。为计算 $s_1, s_2, s_3, s_4, s_5, s_6$ 可先写出:

$$R(x) = q_1(x)m_1(x) + r_1(x) \quad 0 \leqslant \partial r_1(x) < \partial m_1(x)$$

$$R(x) = q_3(x)m_3(x) + r_3(x) \quad 0 \leqslant \partial r_3(x) < \partial m_3(x)$$

$$R(x) = q_5(x)m_5(x) + r_5(x) \quad 0 \leqslant \partial r_5(x) < \partial m_5(x)$$

所以计算 $r_1(x)、r_3(x)、r_5(x)$ 的电路分别是 $m_1(x)、m_3(x)、m_5(x)$ 的除法电路。如图 7-6 所示。由于 $s_1 = r_1(\alpha)$, 故 $m_1(x)$ 除法电路中寄存器内的数值就是 s_1 。但 $s_2 = s_1^2 = r_1(\alpha^2)$, $s_3 = r_3(\alpha^3)$, $s_4 = s_2^2 = r_1(\alpha^4)$, $s_5 = r_5(\alpha^5)$, $s_6 = s_3^2 = r_3(\alpha^6)$, 则需由 $r_1(\alpha)、r_3(\alpha)$ 和 $r_5(\alpha)$ 分别变换后才能求得。这种变换方法与 5.6 节中所讲的方法相同。现仅以求 $s_2 = s_1^2 = r(\alpha^2)$, $s_3 = r_3(\alpha^3)$ 和 $s_5 = r_5(\alpha^5)$ 为例说明变换方法及具体电路。

令 $r(x) = a_3x^3 + a_2x^2 + a_1x + a_0$, 则有:

$$r(\alpha^2) = a_3\alpha^6 + a_2\alpha^4 + a_1\alpha^2 + a_0 = a_3\alpha^3 + (a_3 + a_1)\alpha^2 + a_2\alpha + (a_2 + a_0)$$

$$r(\alpha^3) = b_3\alpha^9 + b_2\alpha^6 + b_1\alpha^3 + b_0 = (b_3 + b_2 + b_1)\alpha^3 + b_2\alpha^2 + b_3\alpha + b_0$$

$$r(\alpha^5) = c_1\alpha^5 + c_0 = c_1\alpha^2 + c_1\alpha + c_0$$

需注意的是计算 s_4 和 $s_2 = r_1(\alpha^2)$ 时的 $r(x)$ 电路是用求 $r_1(x)$ 电路变换, 计算 s_6 和 s_3 时的

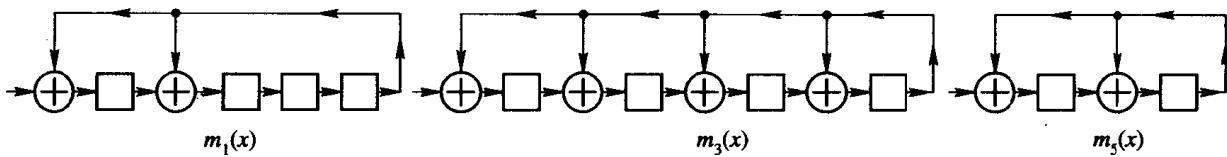


图 7-6 $m_1(x)$ 、 $m_3(x)$ 、 $m_5(x)$ 除法电路

$r(x)$ 电路是用求 $r_3(x)$ 的电路变换，计算 s_5 时的电路是用求 $r_5(x)$ 的电路变换。图 7-7 画出了计算 s_1 至 s_6 的电路。

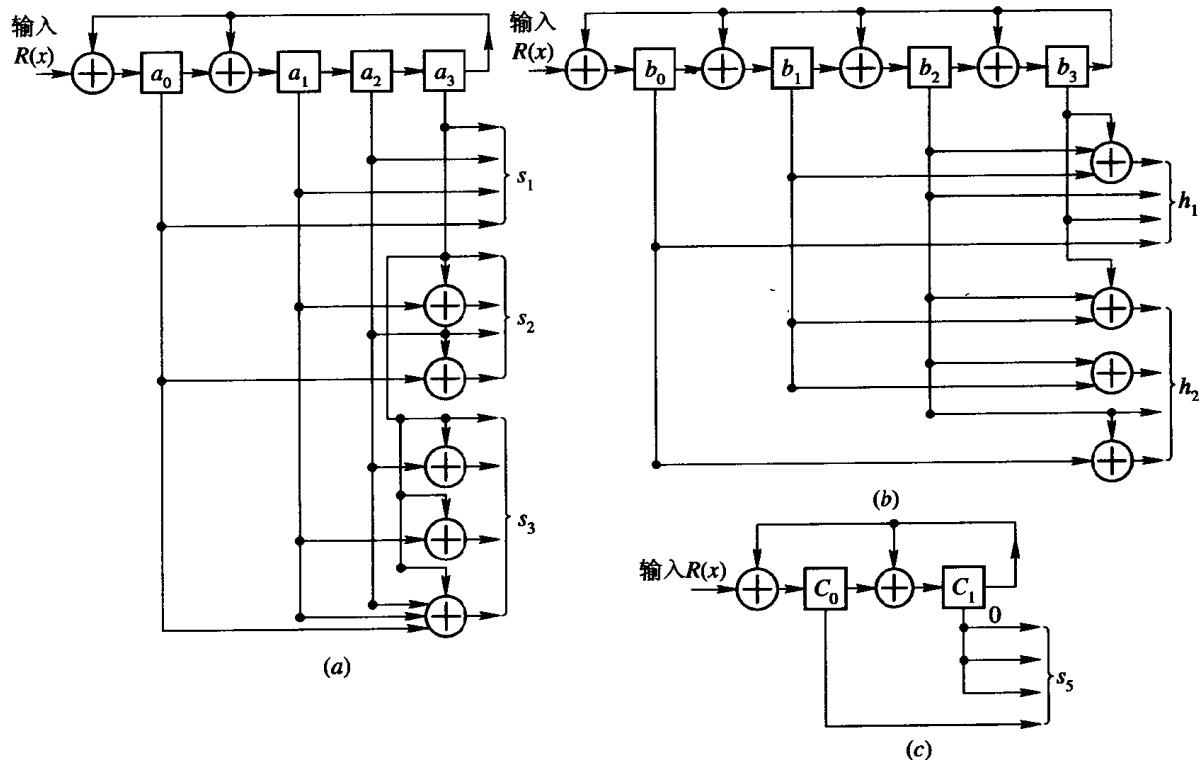


图 7-7 计算 s_1 至 s_6 电路

由上面讨论可以看到，译码过程中以第(2)步由 s_i 求 $\sigma(x)$ 的计算量最大。通常要解式(7.4.9)的线性方程组，它的计算量和系数矩阵阶数的三次方成正比。因此，当码长较长，纠错能力较大时，计算量很大，要求译码器的运算速度很高。并且如果实际产生的错误数 γ 远小于码的纠错能力 t 时，这一步的计算量不但没有减小，反而增加。它必须首先计算式(7.4.10)中 $[M]$ 矩阵的行列式，若为 0，必须降阶再计算行列式的值。这样降阶一次计算一次，直到 $[M]$ 矩阵行列式的值不为 0 为止，共需降阶计算 $t-\gamma$ 次，然后再求解方程组，解出 $\sigma_1, \sigma_2, \dots, \sigma_\gamma$ 。而第(3)步求 $\sigma(x)$ 的根，则可以用钱搜索较简单地实现。第(4)步也必须解一组线性方程组，但对工程上常用的二进制 BCH 码来说，这一步可节省。

§ 7.5 BCH 码的迭代译码算法

由上一节讨论看到，决定译码器复杂性和速度的主要因素在于第二步求 $\sigma(x)$ ，如何简化和加快这一步是 BCH 码译码的关键。1966 年伯利坎普提出了由 S 求 $\sigma(x)$ 的迭代译码算法，极大地加快了求 $\sigma(x)$ 的速度，实现比较简单，且易于用计算机完成译码，因而从工程上解决了 BCH 码的译码问题。1969 年梅西(Massey)指出了迭代译码算法与序列的最短线性移位寄存器综合之间的关系，并进行了简化，自此以后把这种译码算法称为 BM 迭代译码算法。

一、迭代译码算法的基本原理

由式(7.4.7)可知：

$$\sigma(x) = 1 + \sigma_1x + \sigma_2x^2 + \cdots + \sigma_tx^t = \prod_{i=1}^t (1 - x_i x) \quad (7.5.1)$$

设

$$S(x) = 1 + s_1x + s_2x^2 + \cdots = \sum_{i=0}^{\infty} s_i x^i = \sum_{i=0}^{\infty} \left(\sum_{j=1}^t Y_j x_j^i \right) x^i \quad (7.5.2)$$

式中， $s_0=1$ 。作乘积 $S(x)\delta(x)$ ，并用

$$\omega(x) = 1 + \omega_1x + \omega_2x^2 + \cdots \quad (7.5.3)$$

表示该乘积

$$S(x)\sigma(x) = \omega(x)$$

将式(7.5.1)和式(7.5.2)代入上式并展开

$$\begin{aligned} S(x)\sigma(x) &= 1 + (s_1 + \sigma_1)x + (s_2 + \sigma_1s_1 + \sigma_2)x^2 + \cdots + (s_t + \sigma_1s_{t-1} + \cdots + \sigma_t)x^t \\ &\quad + (s_{t+1} + \sigma_1s_t + \cdots + \sigma_ts_1)x^{t+1} + \cdots \\ &= 1 + \omega_1x + \omega_2x^2 + \cdots + \omega_tx^t + \omega_{t+1}x^{t+1} + \cdots \end{aligned} \quad (7.5.4)$$

由式(7.4.9)可知，若 $m_0=1$ ，则有：

$$\begin{aligned} s_{t+1} + \sigma_1s_t + \cdots + \sigma_ts_1 &= 0 \\ s_{t+2} + \sigma_1s_{t+1} + \cdots + \sigma_ts_2 &= 0 \\ &\vdots \\ s_{2t} + \sigma_1s_{2t-1} + \cdots + \sigma_ts_t &= 0 \end{aligned} \quad (7.5.5)$$

所以，式(7.5.4)中大于 x^t 的系数均为 0。由于 $\sigma(x)$ 的次数为 t ，而式(7.5.4)中的最高次数为 t ，所以 $\delta\omega(x) \leq t$ ，或 $\delta\omega(x) \leq \delta\sigma(x)$ 。由式(7.5.4)看出，式(7.5.2)中的 $S(x)$ 及 $\sigma(x)$ 都只要选到 x^t 次项即可，因而 $\delta S(x)\sigma(x) \leq 2t$ 。所以式(7.5.4)成为

$$S(x)\sigma(x) \equiv \omega(x) \pmod{x^{2t+1}} \quad (7.5.6)$$

称它是求 $\sigma(x)$ 的关键方程。

如 $t=1$ ，由式(7.5.6)可得：

$$\begin{aligned} 1 + (s_1 + \sigma_1)x + (s_1\sigma_1 + s_2 + \sigma_2)x^2 &\equiv 1 + \omega_1x + \omega_2x^2 \pmod{x^3} \\ 1 + (s_1 + \sigma_1)x + (s_1\sigma_1 + s_2 + \sigma_2)x^2 - (1 + \omega_1x + \omega_2x^2) &\equiv 0 \pmod{x^3} \end{aligned}$$

或

$$1 + (s_1 + \sigma_1)x + (s_1\sigma_1 + s_2 + \sigma_2)x^2 - (1 + \omega_1x + \omega_2x^2) = x^3Q(x) \equiv 0 \pmod{x^3}$$

因为右边方程 $Q(x)x^3$ 的最低次为三次, 故左边方程的常数项, x 的一次、二次项系数均为 0, 即

$$\begin{aligned}s_1 + \sigma_1 - \omega_1 &= 0 \\ s_1\sigma_1 + s_2 + \sigma_2 - \omega_2 &= 0\end{aligned}$$

注意到 $t=1$, $\sigma_2=0$, $\omega_2=0$, 所以有:

$$\begin{aligned}\sigma_1 &= -s_2/s_1 & \omega_1 &= s_1 - s_2/s_1 \\ \sigma(x) &= 1 + (-s_2/s_1)x\end{aligned}$$

这与用求解式(7.4.10)得到的结果式(7.4.12)完全相同。

用展开式(7.5.6)两边的方法求解 $\sigma(x)$, 若错误个数 t 比较大则仍很麻烦。下面我们讨论用迭代方法解式(7.5.6), 求 $\sigma_1, \sigma_2, \dots, \sigma_t$ 。

所谓用迭代方法求解式(7.5.6), 就是首先选择一组或两组合理的初值如 $\sigma^{(0)}(x)$ 和 $\omega^{(0)}(x)$, 然后开始第一次迭代运算求得 $\sigma^{(1)}(x)$ 和 $\omega^{(1)}(x)$, 并用 $\sigma^{(0)}(x)$ 和 $\omega^{(0)}(x)$ 表示它们。这样依次进行, 由 $\sigma^{(i)}(x)$ 及 $\omega^{(i)}(x)$ 求得 $\sigma^{(i+1)}(x)$ 和 $\omega^{(i+1)}(x)$, 也就是首先计算出满足式(7.5.6)的 $\sigma(x)$ 和 $\omega(x)$ 的低次项, 然后通过迭代逐步求得 $\sigma(x)$ 和 $\omega(x)$ 的高次项, 最后解出满足式(7.5.6)的 $\sigma(x)$ 和 $\omega(x)$ 。

下面对 $i=-1, 0, 1, 2, \dots, 2t$ 求解:

$$S(x)\sigma(x) \equiv \omega(x) \pmod{x^{i+1}} \quad (7.5.7)$$

其中设 $i=-1$, 完全是人为的, 目的是为了求得一组初始值。先从 $i=-1$ 开始, 此时

$$S(x)\sigma^{(-1)}(x) \equiv \omega^{(-1)}(x) \pmod{x^0}$$

得到一组初始值 $\sigma^{(-1)}(x) = -1$, $\omega^{(-1)}(x) = 0$ 。然后再取 $i=0$, 显然 $\sigma^{(0)}(x) = 1$, $\omega^{(0)}(x) = 1$ 是一个解, 由此得到迭代算法所需的另一组初始值。由此开始进入迭代第一步, 求满足式(7.5.6)的以 x^2 为模的 $\sigma^{(1)}(x)$ 和 $\omega^{(1)}(x)$ 。再进入迭代第二步, 求满足以 x^3 为模的 $\sigma^{(2)}(x)$, $\omega^{(2)}(x)$ 等等。直到第 $2t$ 步迭代, 求得满足式(7.5.6)的以 x^{2t+1} 为模的 $\sigma^{(2t)}(x)$ 和 $\omega^{(2t)}(x)$ 时为止。其迭代过程如下:

$$\begin{aligned}S(x)\sigma^{(1)}(x) &\equiv \omega^{(1)}(x) \pmod{x^2} \\ S(x)\sigma^{(2)}(x) &\equiv \omega^{(2)}(x) \pmod{x^3} \\ &\vdots \\ S(x)\sigma^{(2t)}(x) &\equiv \omega^{(2t)}(x) \pmod{x^{2t+1}}\end{aligned} \quad (7.5.8)$$

由此看到, 用迭代方法求解式(7.5.6), 就是根据式(7.5.7)由已知的第 i 步的解 $\sigma^{(i-1)}(x)$ 和 $\omega^{(i-1)}(x)$, 求解满足 $\pmod{x^{i+1}}$ 的 $\sigma^{(i)}(x)$ 和 $\omega^{(i)}(x)$, 最后求得满足 $\pmod{x^{2t+1}}$ 的解 $\sigma^{(2t)}(x)$ 和 $\omega^{(2t)}(x)$, 它就是式(7.5.6)的解。

在求解过程中, 如果 $s_1 = s_2 = \dots = s_{j-1} = 0$, 而 $s_j \neq 0$, 则当 $i < j$ 时

$$\sigma^{(i)} = 1 \quad \omega^{(i)} = 1$$

是满足式(7.5.7)的一个解, 而当 $i=j$ 时, 则至少有 2 个可能的解:

$$\sigma^{(i)}(x) = 1 - s_j x^j \quad \omega^{(i)}(x) = 1$$

和

$$\sigma^{(i)}(x) = 1 \quad \omega^{(i)}(x) = 1 + s_j x^j$$

这说明解不是唯一的。一般说来, 用这种方法求解时, 每一步都不是唯一的, 为了使解唯

一，必须加以条件限制。

在 q 进制无记忆离散对称信道中，若信道的转移概率 $p < 1/q$ ，则信道产生错误个数少的可能性最大，也就是说 $\sigma(x)$ 的次数越低的可能性越大。因此使译码错误概率最小的最大似然译码器，在求 $\sigma(x)$ 的迭代过程中，为了使解唯一，必须保证 $\sigma(x)$ 的次数最小，且 $\omega(x)$ 的次数不大于 $\sigma(x)$ 的次数。也就是在迭代过程中必须始终满足以下条件：

- (1) $\partial \circ \sigma^{(i)}(x) \geq \partial \circ \omega^{(i)}(x)$
- (2) $\partial \circ \sigma^{(i)}(x) = \min \partial \circ \sigma^{(i)}(x)$

或

$$D^*(i) = \min D(i) \quad (7.5.9)$$

式中， $D^*(i)$ 是第*i*+1次迭代过程中得到的 $\sigma^{(i)}(x)$ 中，最低次数的 $\sigma^{(i)}(x)$ 的次数。如果在迭代过程中不能保证满足上述两个条件，则无法保证求得的解是唯一的。

用迭代方法求解到第*j*步时，求得满足

$$S(x)\sigma^{(j)}(x) \equiv \omega^{(j)}(x) \pmod{x^{j+1}} \quad (7.5.10)$$

的解 $\sigma^{(j)}(x)$ 和 $\omega^{(j)}(x)$ 。在通常情况下，第*j*+1步，即以 x^{j+2} 为模时，上式中的 $\sigma^{(j)}(x)$ 和 $\omega^{(j)}(x)$ 不再满足式(7.5.7)。发生这种情况时，我们定义一个第*j*+1步与第*j*步的差值 d_j ，使下式成立：

$$S(x)\sigma^{(j)}(x) \equiv \omega^{(j)}(x) + d_j x^{j+1} \pmod{x^{j+2}} \quad (7.5.11)$$

为了得到 d_j ，将上式左边展开，得

$$\begin{aligned} & (1+s_1x+s_2x^2+\cdots)(1+\sigma_1^{(j)}x+\sigma_2^{(j)}x^2+\cdots) \\ &= 1+(s_1+\sigma_1^{(j)})x+(s_2+s_1\sigma_1^{(j)}+\sigma_2^{(j)})x^2+\cdots \\ & \quad +(s_j+s_{j-1}\sigma_1^{(j)}+s_{j-2}\sigma_2^{(j)}+\cdots+\sigma_j^{(j)})x^j \\ & \quad +(s_{j+1}+s_j\sigma_1^{(j)}+\cdots+\sigma_{j+1}^{(j)})x^{j+1} \end{aligned}$$

由此可知：

$$d_j = s_{j+1} + \sum_{i=1}^{\partial \circ \sigma^{(j)}(x)} s_{j+1-i} \sigma_i^{(j)} \quad (7.5.12)$$

式中， $\sigma_i^{(j)}$ 是

$$\sigma^{(j)}(x) = 1 + \sigma_1^{(j)}x + \sigma_2^{(j)}x^2 + \cdots + \sigma_{D(j)}^{(j)}x^{D(j)}$$

中 x^i 项的系数，这里， $D(j) = \partial \circ \sigma^{(j)}(x)$ 。

把式(7.5.12)代入式(7.5.6)并经化简可得：

$$\sigma^{(j+1)}(x) = \sigma^{(j)}(x) - d_j d_i^{-1} x^{j-i} \sigma^{(i)}(x) \quad (7.5.13)$$

$$\omega^{(j+1)}(x) = \omega^{(j)}(x) - d_j d_i^{-1} x^{j-i} \omega^{(i)}(x) \quad (7.5.14)$$

这里*i*是*j*前面的某一行，并满足*i*- $D(i)$ 最大，且 $d_i \neq 0$ ，而

$$D(j+1) = \max(D(j), j - i + D(i))$$

通过两组初值 $\sigma^{(-1)}(x)$ 、 $\omega^{(-1)}(x)$ 和 $\sigma^{(0)}(x)$ 、 $\omega^{(0)}(x)$ ，以及式(7.5.12)、式(7.5.13)和式(7.5.14)，利用迭代算法就可求得满足式(7.5.6)的解 $\sigma(x)$ 。由此可知，用迭代方法计算就是从假定的 $\gamma=0$ 开始，按方程(7.5.8)的顺序逐步增加 $\omega(x)$ 的次数，并根据关键方程，逐步进行修正求出修正项 d_j ，最终求得 $\sigma^{(2t)}(x)$ 。其迭代步骤如下：

(1) 由初值

$$\sigma^{(-1)}(x)=1, \omega^{(-1)}(x)=0, D(1)=0, d_{-1}=1$$

$$\sigma^{(0)}(x)=1, \quad \omega^{(0)}(x)=1, \quad D(0)=0, \quad d_0=s_1$$

开始迭代。

(2) 按式(7.5.12)计算 d_j , 若 $d_j=0$, 则有:

$$\sigma^{(j+1)}(x)=\sigma^{(j)}(x)$$

$$\omega^{(j+1)}(x)=\omega^{(j)}(x)$$

$$D^*(j+1)=D^*(j)$$

并计算 d_{j+1} , 再进行下一次迭代。

如果 $d_j \neq 0$, 则找出 j 之前的某一行 i , 它在所有 j 行之前各行中的 $i-D(i)$ 最大, 且 $d_i \neq 0$, 于是按照式(7.5.13)和式(7.5.14)分别计算 $\sigma^{(j+1)}(x)$ 和 $\omega^{(j+1)}(x)$, 这就是第 $j+1$ 步的解。

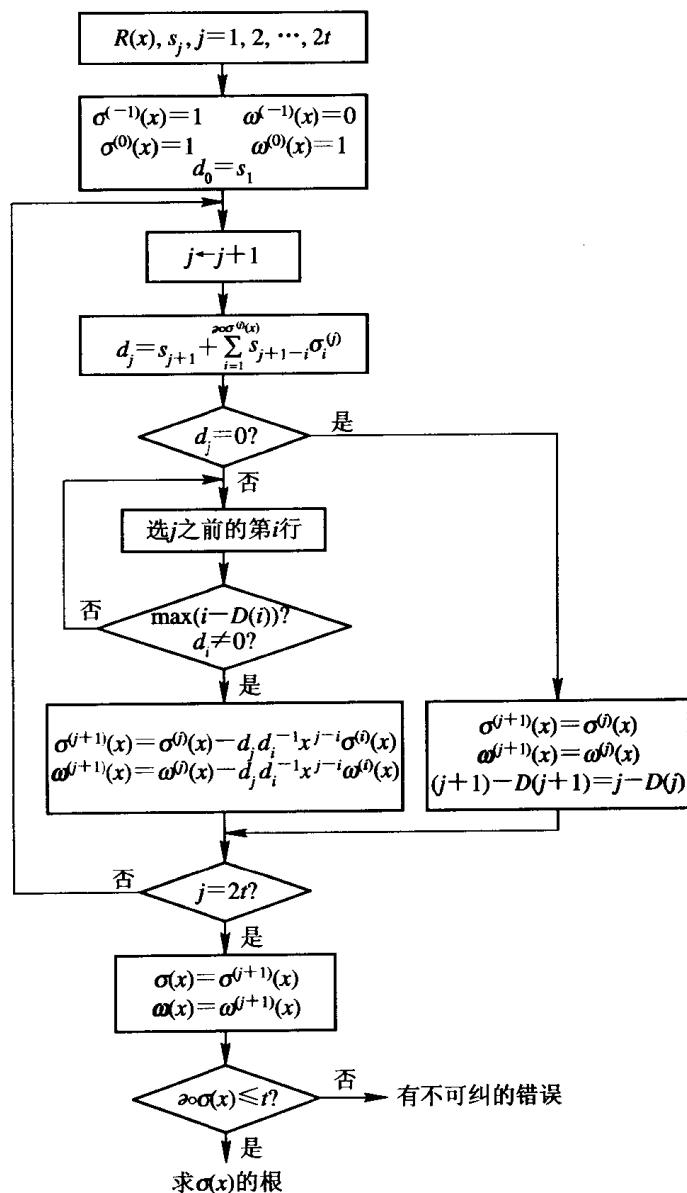


图 7-8 求 $\sigma(x)$ 的迭代算法流程图

(3) 计算 d_{j+1} , 重复(2)进行下一次迭代。这样 $2t$ 次迭代后得到的 $\sigma^{(2t)}(x)$ 和 $\omega^{(2t)}(x)$ 即为所求的 $\sigma(x)$ 和 $\omega(x)$ 。

$\omega(x)$ 是在求 $\sigma(x)$ 过程中得到的辅助多项式, 在求解错误值时将用到, 故称 $\omega(x)$ 为错
误值多项式。

上述迭代过程如图 7-8 所示。可列成表 7-6, 进行迭代时逐步把表格填满即可。

表 7-6 求 $\sigma(x)$ 的迭代过程表

| j | $\sigma^{(j)}(x)$ | $\omega^{(j)}(x)$ | $D(j)$ | $j-D(j)$ | d_j |
|----------|-------------------|-------------------|--------|----------|-------|
| -1 | 1 | 0 | 0 | -1 | 1 |
| 0 | 1 | 1 | 0 | 0 | s_1 |
| 1 | | | | | |
| 2 | | | | | |
| \vdots | | | | | |
| $2t$ | | | | | |

例 7.6 GF(2^4)上的[15, 9, 7]RS 码, 它的生成多项式 $g(x)=(x-\alpha)(x-\alpha^2)(x-\alpha^3)(x-\alpha^4)(x-\alpha^5)(x-\alpha^6)$, 能纠正 3 个错误。已知收到的 $R(x)=\alpha^7x^3+\alpha^{11}x^{10}$, 求错误位置。

首先计算出伴随式: $s_1=\alpha^7$, $s_2=\alpha^{12}$, $s_3=\alpha^6$, $s_4=\alpha^{12}$, $s_5=\alpha^{14}$, $s_6=\alpha^{14}$ 。这里 α 之间的运算按表 4-2 规则。用迭代方法求得 $\sigma(x)$ 如表 7-7 所示。得到 $\sigma(x)=1+\alpha^{12}x+\alpha^{13}x^2$, 求得错误位置数是 α^3 和 α^{10} , 所以 x^3 与 x^{10} 位置发生了错误。 ■

例 7.7 [31, 11, 11]二进制 BCH 码, $g(x)=(x^5+x^2+1)(x^5+x^4+x^3+x^2+1)(x^5+x^4+x^2+x+1)(x^5+x^3+x^2+x+1)$, 能纠正 5 个错误。设 $R(x)=x^{29}+x^{11}+x^9$, 求 $C(x)$ 。

表 7-7 求[15, 9, 7]RS 码 $\sigma(x)$ 的迭代过程表

| j | $\sigma^{(j)}(x)$ | $\omega^{(j)}(x)$ | $D(j)$ | $j-D(j)$ | d_j |
|------|---------------------------------|---------------------------------|--------|----------|-----------------------|
| -1 | 1 | 0 | 0 | -1 | 1 |
| 0 | 1 | 1 | 0 | 0 | $s_1=\alpha^7$ |
| 1 | $1+\alpha^7x$ | 1 | 1 | 0 | α^5 |
| 2 | $1+\alpha^5x$ | $1+\alpha^{13}x$ | 1 | 1 | α^3 |
| $3a$ | $1+\alpha^7x+\alpha^5x^2$ | 1 | 2 | 1 | α^5 (取 $i=1$) |
| $3b$ | $1+\alpha^5x+\alpha^{11}x^2$ | $1+\alpha^{13}x+\alpha^{11}x^2$ | 2 | 1 | α^2 (取 $i=0$) |
| 4 | $1+\alpha^{12}x+\alpha^{13}x^2$ | $1+\alpha^2x+x^2$ | 2 | 2 | 0 |
| 5 | $1+\alpha^{12}x+\alpha^{13}x^2$ | $1+\alpha^2x+x^2$ | 2 | 3 | 0 |

首先计算伴随式: $s_1=\alpha^{18}$, $s_2=\alpha^5$, $s_3=\alpha^{15}$, $s_4=\alpha^{10}$, $s_5=\alpha^{13}$, $s_6=\alpha^{30}$, $s_7=\alpha^{22}$, $s_8=\alpha^{20}$, $s_9=\alpha^{17}$, $s_{10}=\alpha^{26}$ 。这里 $\alpha \in GF(2^5)$ 是本原元, 它是 $p(x)=x^5+x^2+1$ 的根。因为二进制情况下不需要求错误值, 故不必求 $\omega(x)$ 。用迭代法求得 $\sigma(x)$ 如表 7-8 所示。

表 7-8 求[31, 11, 11]二进制 BCH 码 $\sigma(x)$ 的迭代过程表

| j | $\sigma^{(j)}(x)$ | $D(j)$ | $j-D(j)$ | d_j |
|-----|--|--------|----------|---------------------|
| -1 | 1 | 0 | -1 | 1 |
| 0 | 1 | 0 | 0 | $s_1 = \alpha^{18}$ |
| 1 | $1 + \alpha^{18}x$ | 1 | 0 | $0(i=1)$ |
| 2 | $1 + \alpha^{18}x$ | 1 | 1 | α^{20} |
| 3 | $1 + \alpha^{18}x + \alpha^2x^2$ | 2 | 1 | $0(i=0)$ |
| 4 | $1 + \alpha^{18}x + \alpha^2x^2$ | 2 | 2 | α^{19} |
| 5 | $1 + \alpha^{18}x + \alpha^{28}x^2 + \alpha^{17}x^3$ | 3 | 2 | $0(i=2)$ |
| 6 | $1 + \alpha^{18}x + \alpha^{28}x^2 + \alpha^{17}x^3$ | 3 | 3 | 0 |
| 7 | $1 + \alpha^{18}x + \alpha^{28}x^2 + \alpha^{17}x^3$ | 3 | 4 | 0 |
| 8 | $1 + \alpha^{18}x + \alpha^{28}x^2 + \alpha^{17}x^3$ | 3 | 5 | 0 |
| 9 | $1 + \alpha^{18}x + \alpha^{28}x^2 + \alpha^{17}x^3$ | 3 | 6 | 0 |
| 10 | $1 + \alpha^{18}x + \alpha^{28}x^2 + \alpha^{17}x^3$ | 3 | 7 | 0 |

由此可知, $\sigma(x) = 1 + \alpha^{18}x + \alpha^{28}x^2 + \alpha^{17}x^3$, 用钱搜索方法求得 3 个根分别是 α^{-29} , α^{-11} , α^{-9} , 所以错误图样 $\hat{E}(x) = x^{29} + x^{11} + x^9$, 故 $\hat{C}(x) = R(x) - \hat{E}(x) = 0$, 是一个全为 0 码字。 ■

由上面分析和两个例中看出, 与一般译码方法相比, 迭代译码有以下显著优点:

- (1) 计算比较简单, 特别是可以用微处理机及软件实现译码, 且程序不复杂。
- (2) 实际产生的错误个数 $\gamma < t$ 时, 计算量不仅不会增加, 反而减少。
- (3) 无论实际发生的错误个数 $\gamma \leq t$ 是大是小, 用迭代方法计算所需的时间较为一致。这对译码设备的设计和实现比较有利。
- (4) 在二进制情况下, 迭代步骤可以减少一半。

二、二进制 BCH 码迭代译码算法的简化

由例 7.7 看出, 迭代译码过程中, 若 j 为奇数时 $d_j = 0$, 因而迭代次数可以减少一半, 下面我们将证明这一点。

若码为二进制 BCH 码, 则 $E(x)$ 中的错误值 Y_i 均为 1, 因而式(7.4.3)中的 s_j 成为

$$s_j = \sum_i^t x_i^j \quad (7.5.15)$$

由加权幂和对称函数变为一般的初等幂和对称函数。另一方面, 由二进制 BCH 码的校验矩阵特点可知 $s_{2j} = s_j^2$ 。

由关键方程式(7.5.6)出发, 把它展开并考虑到式(7.5.5), 可得

$$\begin{aligned} S(x)\sigma(x) &\equiv 1 + (s_1 + \sigma_1)x + (s_2 + \sigma_1s_1 + \sigma_2)x^2 + (s_3 + s_2\sigma_1 + s_1\sigma_2 + \sigma_3)x^3 \\ &\quad + \cdots + (s_t + \sigma_1s_{t-1} + \cdots + \sigma_t)x^t \pmod{x^{2t+1}} \end{aligned} \quad (7.5.16)$$

由式(7.4.6)和式(7.5.15)可知, 有如下的牛顿恒等式:

$$\begin{aligned} s_1 - \sigma_1 &= 0 \\ s_2 - s_1\sigma_1 + 2\sigma_2 &= 0 \\ s_3 - s_2\sigma_1 + s_1\sigma_2 - 3\sigma_3 &= 0 \end{aligned}$$

⋮

在任意域中成立。所以在 GF(2) 中上式成为：

$$\begin{aligned}
 s_1 + \sigma_1 &= 0 \\
 s_2 + \sigma_1 s_1 &= 0 \\
 s_3 + s_2 \sigma_1 + s_1 \sigma_2 + \sigma_3 &= 0 \\
 &\vdots
 \end{aligned} \tag{7.5.17}$$

将式(7.5.17)代入式(7.5.16)得

$$S(x)\sigma(x) = 1 + \sigma_2 x^2 + \sigma_4 x^4 + \cdots + \sigma_{2t} x^{2t} \equiv \omega(x) \pmod{x^{2t+1}} \tag{7.5.18}$$

这说明关键方程的奇数次项的系数均为 0，仅有偶次项系数。基于上式我们不难相信，在二进制 BCH 码的迭代译码过程中有如下特点：

- (1) $\omega^{(i)}(x)$ 等于 $\sigma^{(i)}(x)$ 的偶部多项式 $\sigma_e^{(i)}(x)$ ；
- (2) i 为奇数时的 $d_i = 0$ 。

为了证明上述两个特点，在特征为 2 的域中引入逆生成函数 $R(x)$ ，它定义为

$$S(x)R(x) = 1 \tag{7.5.19}$$

式中， $R(x) = 1 + R_o x + R_e x^2 + \cdots$ ，是一个有常数项的幂级数。我们分别用 S_o 和 R_o 记为 S 和 R 的奇次幂部分，用 S_e 和 R_e 记为它们的偶次幂部分。

引理 7.5.3 在特征为 2 的域中， $R_e = 0$ 的充要条件是 $S_e = S^2$ 。

证明 将 S 、 R 写成：

$$S = 1 + S_o + S_e \quad R = 1 + R_o + R_e$$

代入式(7.5.19)展开得

$$\begin{aligned}
 (1 + S_o + S_e)(1 + R_o + R_e) \\
 = 1 + S_e + R_e + S_o R_o + S_e R_e + (1 + S_e)R_o + (1 + R_e)S_o = 1
 \end{aligned}$$

比较奇偶部分可得：

$$\text{偶部: } 1 + S_e + R_e + S_o R_o + S_e R_e = 1$$

$$\text{奇部: } (1 + S_e)R_o + (1 + R_e)S_o = 0$$

消去 R_o 得

$$R_e = (S_e + S^2)/(1 + S^2)$$

所以，若 $S_e = S^2$ ，则 $R_e = 0$ ；反之若 $R_e = 0$ ，则 $S_e = S^2$ 。 ■

定理 7.5.2 对二进制 BCH 码的迭代译码算法有：

$$\omega^{(i)}(x) = \sigma_e^{(i)}(x) \quad i = 0, 1, 2, \dots, 2t$$

和

$$d_{2k+1} = 0 \quad k = 0, 1, 2, \dots, t - 1$$

证明 先证明第一部分，对 i 用归纳法。 $i=0$ 时，由初始条件 $\sigma^{(0)}(x)=1$ 和 $\omega^{(0)}=1$ ，可知定理为真，当 $i=1$ 时有：

$$\begin{aligned}
 \sigma^{(1)}(x) &= 1 - s_1 x \\
 \omega^{(1)}(x) &= \sigma_e^{(1)}(x) = 1
 \end{aligned}$$

所以定理亦真。现假设：

$$\omega^{(2i)}(x) = \sigma_e^{(2i)}(x) \tag{7.5.20}$$

$$d_{2i-1} = 0$$

不失一般性, 认为 $d_{2i} \neq 0$, 这时由式(7.5.14)有

$$\omega^{(2i+1)}(x) = \omega^{(2i)}(x) - d_{2i}d_{2h}^{-1}x^{2(i-h)}\omega^{(2h)}(x)$$

上式一般情况下 $2h$ 应用 l 代替, 但由于选 $l < 2h$ 时, 必须有 $d_l \neq 0$, 由归纳法假设它必是偶数。把式(7.5.20)代入上式得

$$\omega^{(2i+1)}(x) = \sigma_e^{(2i)}(x) - d_{2i}d_{2h}^{-1}x^{2(i-h)}\sigma_e^{(2h)}(x) = \sigma_e^{(2i+1)}(x)$$

这就证明了定理的第一部分。

下面证明 $d_{2i+1}=0$ 。为此考虑:

$$S(x)\sigma^{(2i+1)}(x) \equiv \omega^{(2i+1)}(x) + d_{2i+1}x^{2i+2} \pmod{x^{2i+3}} \quad (7.5.21)$$

对二进制 BCH 码有 $s_{2j}=s_j^2$ 或 $S_e(x)=S^2(x)$, 由引理 7.5.3 知 $R_e(x)=0$ 或 $R(x)=R_o(x)$ 。把式(7.5.21)的两边乘以 $R(x)=1+R_o(x)$, 并考虑到 $\omega^{(i)}(x)=\sigma_e^{(i)}(x)$ 及式(7.5.19)和引理 7.5.3 可得

$$\begin{aligned} \sigma^{(2i+1)}(x) &\equiv \sigma_e^{(2i+1)}(x) + R_o(x)\sigma_e^{(2i+1)}(x) + d_{2i+1}x^{2i+2} \\ &\quad + R_o(x)d_{2i+1}x^{2i+2} \pmod{x^{2i+3}} \end{aligned} \quad (7.5.22)$$

由于

$$R_o(x) = R_1x + R_3x^3 + \dots$$

所以

$$R_o(x)x^{2i+2} \equiv 0 \pmod{x^{2i+3}}$$

把式(7.5.22)移项, 考虑到上式以及 $\sigma^{(j)}(x)=\sigma_0^{(j)}(x)+\sigma_e^{(j)}(x)$, 可得

$$\sigma_0^{(2i+1)}(x) + R_o(x)\sigma_e^{(2i+1)}(x) \equiv d_{2i+1}x^{2i+2} \pmod{x^{2i+3}}$$

上式左边为奇函数, 右边为偶函数, 这是不可能的, 所以只有 $d_{2i+1}=0$ 。 ■

根据以上定理, 可以给出二进制 BCH 码的求 $\sigma(x)$ 的迭代算法:

(1) 迭代初始值。 $\sigma^{(-1/2)}(x)=1$, $D(1/2)=0$, $d_{-1/2}=1$; $\sigma^{(0)}(x)=1$, $D(0)=0$, $d_0=s_1$ 。

(2) 计算

$$d_j = s_{2j+1} + \sum_{i=1}^{\partial \cdot \sigma^{(j)}(x)} s_{2j+1-i} \sigma_i^{(j)} \quad (7.5.23)$$

如果 $d_j=0$, 则 $\sigma^{(j+1)}(x)=\sigma^{(j)}(x)$, $D(j+1)=D(j)$ 并进行下一次迭代。如果 $d_j \neq 0$, 则找出第 j 行的前一行, 例如说第 i 行 ($i < j$), 它的 $2i-D(i)$ 为最大且 $d_i \neq 0$, 于是

$$\sigma^{(j+1)}(x) = \sigma^{(j)}(x) + d_j d_i^{-1} x^{2(j-i)} \sigma_i^{(i)}(x) \quad (7.5.24)$$

然后再计算 d_{j+1} , 作下一次迭代。这过程一直继续到算出 $\sigma^{(t)}(x)$ 时为止。 $\sigma^{(t)}(x)$ 即为所求之 $\sigma(x)$ 。迭代过程可用表 7-9 表示。

表 7-9 二进制 BCH 码求 $\sigma(x)$ 迭代算法过程表

| j | $\sigma^{(j)}(x)$ | $D(j)$ | $2j-D(j)$ | d_j |
|----------|-------------------|--------|-----------|-------|
| -1/2 | 1 | 0 | -1 | 1 |
| 0 | 1 | 0 | 0 | s_1 |
| 1 | | | | |
| 2 | | | | |
| \vdots | | | | |
| t | | | | |

例 7.8 仍以例 7.7 中的二进制[31, 11, 11]BCH 码为例, $R(x)=x^{29}+x^{11}+x^9$ 。求得的 s_1, s_2, \dots, s_{10} 同前。计算 $\sigma(x)$ 的迭代过程如表 7-10 所示, 所求得的 $\sigma(x)$ 与例 7.7 中的结果完全相同。

表 7-10 二进制[31, 11, 11]BCH 码求 $\sigma(x)$ 迭代过程

| j | $\sigma^{(j)}(x)$ | $D(j)$ | $2j-D(j)$ | d_j |
|-----|--|--------|-----------|---------------------|
| 1/2 | 1 | 0 | -1 | 1 |
| | 1 | 0 | 0 | $s_1 = \alpha^{18}$ |
| 1 | $1+\alpha^{18}x$ | 1 | 1 | α^{20} |
| 2 | $1+\alpha^{18}x+\alpha^2x^2$ | 2 | 2 | α^{19} |
| 3 | $1+\alpha^{18}x+\alpha^{28}x^2+\alpha^{17}x^3$ | 3 | 3 | 0 |
| 4 | $1+\alpha^{18}x+\alpha^{28}x^2+\alpha^{17}x^3$ | 3 | 5 | 0 |
| 5 | $1+\alpha^{18}x+\alpha^{28}x^2+\alpha^{17}x^3$ | 3 | 7 | 0 |

从上述讨论看到, 对于非二进制纠 t 个错误的 BCH 码, 求 $\sigma(x)$ 需 $2t$ 次迭代。1981 年陈金龙证明^[12]并提出了一种改进的算法, 当信道中实际产生的错误个数 $\gamma < t$ 时, 求 $\sigma(x)$ 仅需 $t + \gamma$ 次迭代, 而对于二进制码, 则仅需 $(t + \gamma)/2$ 次迭代, 并猜想这是求 $\sigma(x)$ 所需的最低迭代次数。当 t 很大时, 陈金龙所提的改进算法具有实际意义, 能加快译码速度。

此外, 在实际的无记忆对称信道中, 码元错误概率 $p_e < 1/q$, 因此产生重量轻的错误图样的概率比产生重量重的要小, 特别产生一个错误的错误图样的概率很大。因此由式(7.4.1)可知, 如果

$$s_{j+1}/s_j = \alpha^{n-i} \quad j = 1, \dots, 2t-1 \quad (7.5.25)$$

则可知这时信道仅产生一个错误, 且错误产生在 x^{n-i} 位置上, 错误值

$$e_{n-i} = s_1 \alpha^{-(n-i)} = s_1 \alpha^i \quad m_0 = 1 \quad (7.5.26)$$

从而不必进行 $2t$ 次迭代运算求 $\sigma(x)$, 就能得到错误图样。这种方法对加快大纠错能力码的译码速度具有一定的实际意义。

三、迭代译码算法与序列的最短线性反馈移位寄存器综合及线性复杂度之间的关系

这里将简单讨论一下求 $\sigma(x)$ 的迭代算法, 与序列的最短线性反馈移位寄存器(LFSR)综合及序列的线性复杂度之间的关系。

由上节讨论可知, 要解式(7.5.5)

$$\begin{aligned} s_{t+1} + \sigma_1 s_t + \dots + \sigma_t s_1 &= 0 \\ s_{t+2} + \sigma_1 s_{t+1} + \dots + \sigma_t s_2 &= 0 \\ &\vdots \\ s_{2t} + \sigma_1 s_{2t-1} + \dots + \sigma_t s_t &= 0 \end{aligned}$$

线性方程组可用式(7.5.6)

$$S(x)\sigma(x) \equiv \omega(x) \pmod{x^{2t+1}}$$

利用迭代方法求解 $\sigma(x)$ 。在上述方程组中，如果已知 $\sigma_1, \sigma_2, \dots, \sigma_t$ 和 s_1, s_2, \dots, s_t ，则可通过计算求得 s_{t+1} ，依次类推就可求得 $s_{t+2}, s_{t+3}, \dots, s_{2t}$ 。这个递推过程可用图 7-9 的电路实现，与图 5-18 比较可知它们完全相同。

式(7.5.5)显然是一组线性递推关系式。因此只要在电路中放入初始值 s_1, s_2, \dots, s_t ，通过移位就可在输出端得到一组序列 s_1, s_2, \dots, s_{2t} 。现在的问题是已知 s_1, s_2, \dots, s_{2t} ，问电路应如何联接，并且使用的级数最少，才能使输出的序列为 s_1, s_2, \dots, s_{2t} 。这个问题就是 LFSR 的综合问题，它与 BCH 码迭代译码求 $\sigma(x)$ 的关系，首先由梅西于 1969 年发现。

从 BCH 码译码的角度考虑，上述问题就是已知伴随式 s_1, s_2, \dots, s_{2t} ，如何确定电路的联接多项式 $\sigma(x) = 1 + \sigma_1x + \sigma_2x^2 + \dots + \sigma_tx^t$ ，也就是求错误位置多项式的系数 $\sigma_1, \sigma_2, \dots, \sigma_t$ ，且要求 $\partial \circ \sigma(x)$ 最低。

如已知 $s_1, s_2, \sigma(x)$ 应是什么才能使输出的序列为 s_1, s_2 。如果我们对产生序列的 LFSR 的级数没有限制，则解有无穷多个，例如可用两级移存器构成，把 s_1, s_2 放入，则前两次移位输出的序列就是 s_1, s_2 。而两级移存器可以随便联接，也可以任意加 n 级，因而解有无穷多个。但如果要求移位寄存器的级数最小，这相当于要求电路的联接多项式 $\sigma(x)$ 的次数最低，则解就不会有无穷多个。由此看出，求产生序列 s_1, s_2, \dots, s_{2t} 电路特征多项式或联接多项式 $\sigma(x)$ ，就是迭代译码算法中由 s_1, s_2, \dots, s_{2t} 求错误位置多项式。显然只有 $\sigma_1 = -s_2/s_1$ ，也就是联接多项式 $\sigma(x) = -(s_2/s_1)x$ 时，才能由已知的 s_1 ，得到 s_1, s_2 序列，如图 7-10 所示。

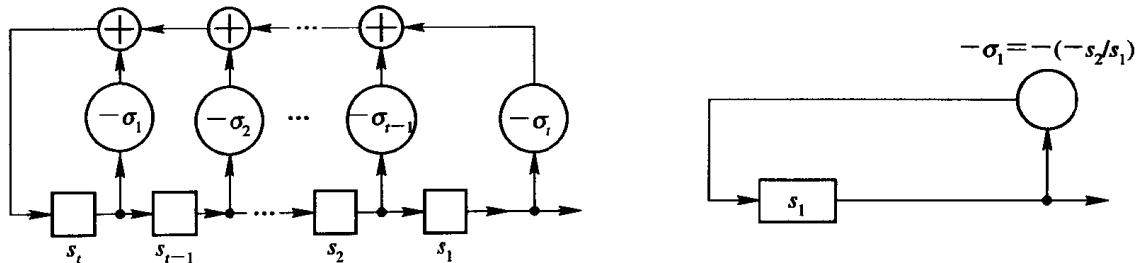


图 7-9 解(7.5.5)式的电路图

图 7-10 得到 s_1, s_2 序列的最短线性移存器

伴随式序列 $S^{(2t)} = \{s_1, s_2, \dots, s_{2t}\}$ 满足式(7.5.5)所确定的线性递推关系，因此已知伴随式序列 $S^{(2t)}$ ，用 BM 算法求其联接多项式 $\sigma(x)$ ，由 § 5.9 节知，就是求 $S^{(2t)}$ 的线性复杂度，且

$$\partial \circ \sigma(x) = L(S^{(2t)}) \quad (7.5.27)$$

这说明 BM 算法不仅在 BCH 码的译码中起着关键作用，而且也是目前已知的求序列线性复杂度的最快与最好的方法之一。

四、错误值的计算

BCH 码译码的最后一步就是求错误值 Y_i 。这一步对二进制码来说可省略，但对多进制码来说是必需的。下面求出的 Y_i ，都是针对 $m_0=1$ 时的情况，若 $m_0 \neq 1$ ，则应稍作修正。

设实际产生的错误个数 $\gamma \leq t$ ，则由式(7.4.5)和式(7.4.7)可知：

$$\sigma(x) = \prod_{i=1}^{\gamma} (1 - x_i x) = \sum_{i=0}^{\gamma} \sigma_i x^i \quad (7.5.28)$$

式中, $\sigma_0=1$, 定义如下函数

$$\sigma_j(x) = \frac{\sigma(x)}{1 - x_j x} \quad (7.5.29)$$

把式(7.5.28)代入上式

$$\sigma_j(x) = \prod_{i \neq j} (1 - x_i x) = \sum_{i=0}^{\gamma-1} \sigma_{ji} x^i \quad (7.5.30)$$

由式(7.5.29)和式(7.5.30)

$$\sigma(x) = \sigma_j(x)(1 - x_j x) = (1 - x_j x) \sum_{i=0}^{\gamma-1} \sigma_{ji} x^i$$

由式(7.5.28)与上式可知:

$$\sum_{i=0}^{\gamma} \sigma_{ji} x^i = (1 - x_j x) \sum_{i=0}^{\gamma-1} \sigma_{ji} x^i = \sum_{i=0}^{\gamma-1} \sigma_{ji} x^i - \sum_{i=0}^{\gamma-1} \sigma_{ji} x_j x^{i+1}$$

比较上式两边同次项系数可得:

$$\begin{aligned} \sigma_{j0} &= \sigma_0 = 1 \\ \sigma_i &= \sigma_{ji} - \sigma_{j(i-1)} x_j \quad i=1, 2, \dots, \gamma-1 \end{aligned}$$

或

$$\sigma_{ji} = \sigma_i + \sigma_{j(i-1)} x_j, \quad i = 1, 2, \dots, \gamma-1 \quad (7.5.31)$$

作

$$\sum_{i=0}^{\gamma-1} \sigma_{ji} s_{\gamma-i} = \sum_{i=0}^{\gamma-1} \sigma_{ji} \left(\sum_{k=1}^{\gamma} Y_k x_k^{\gamma-i} \right)$$

变更求和的顺序

$$\sum_{i=0}^{\gamma-1} \sigma_{ji} s_{\gamma-i} = \sum_{k=1}^{\gamma} Y_k x_k^{\gamma} \sum_{i=0}^{\gamma-1} \sigma_{ji} x_k^{-i} = \sum_{k=1}^{\gamma} Y_k x_k^{\gamma} \sigma_j(x_k^{-1})$$

由于 $\sigma_j(x_k^{-1})$ 仅当 $k=j$ 时不为零, 所以

$$\sum_{i=0}^{\gamma-1} \sigma_{ji} s_{\gamma-i} = Y_j x_j^{\gamma} \sigma_j(x_j^{-1}) = Y_j \sum_{i=0}^{\gamma-1} \sigma_{ji} x_j^{\gamma-i}$$

由此可得

$$Y_j = \frac{\sum_{i=0}^{\gamma-1} \sigma_{ji} s_{\gamma-i}}{x_j \sum_{i=0}^{\gamma-1} \sigma_{ji} x_j^{\gamma-i-1}} \quad (7.5.32)$$

它只和 σ_i 和 $\sigma(x)$ 的根 x_i 及 s_k 有关, 因此当用迭代法求得 $\sigma(x)$ 后就可立即计算 Y_j , 它比用式(7.4.14)计算要简单得多。由式(7.5.32)计算 Y_j , 其计算量与 t^2 成正比, 而解线性方程组则与 t^3 成正比。

例 7.9 以例 7.6 中的 [15, 9] RS 码为例, 已求得 $\sigma(x) = 1 + \alpha^{12} x + \alpha^{13} x^2$, 两个根是 $x_1^{-1} = \alpha^{-3}$ 和 $x_2^{-1} = \alpha^{-10}$, 相应的错误位置是 $x_1 = \alpha^3$, $x_2 = \alpha^{10}$ 。求 Y_1 和 Y_2 。

由方程(7.5.31)可得: $\sigma_{10} = \sigma_{20} = 1$, $\sigma_{11} = x_2 = \alpha^{10}$, $\sigma_{21} = x_1 = \alpha^3$, 因而可得:

$$Y_1 = \frac{\sigma_{10}s_2 + \sigma_{11}s_1}{\sigma_{10}x_1^2 + \sigma_{11}x_1} = \alpha^7$$

$$Y_2 = \frac{\sigma_{20}s_2 + \sigma_{21}s_1}{\sigma_{20}x_2^2 + \sigma_{21}x_2} = \alpha^{11}$$

由此可知错误图样 $\hat{E}(x) = \alpha^7 x^3 + \alpha^{11} x^{10}$, 发送的码矢 $\hat{C}(x) = R(x) - \hat{E}(x) = 0$, 是一个全为

0 码字。 ■

我们也可以用 $\omega(x)$ 求得错误值，由于在迭代过程中 $\omega(x)$ 是已知的，因此，用这种方法求 Y_i 可能较为简单。

由式(7.4.4)可知：

$$s_i = \sum_{k=1}^t Y_k x_k^i$$

若实际产生的错误个数为 $\gamma \leq t$ ，则：

$$\begin{aligned} s_i &= \sum_{k=1}^{\gamma} Y_k x_k^i \\ S(x) &= 1 + s_1 x + s_2 x^2 + \cdots = 1 + \sum_{i=1}^{\infty} s_i x^i \\ &= 1 + \sum_{i=1}^{\infty} \left(\sum_{k=1}^{\gamma} Y_k x_k^i \right) x^i = 1 + \sum_{i=1}^{\infty} \left(\sum_{k=1}^{\gamma} Y_k \sum_{j=1}^{\infty} (x_k x)^j \right) \end{aligned}$$

由

$$\sum_{j=1}^{\infty} a^j = a/(1-a) \quad |a| < 1$$

可知：

$$S(x) = 1 + \sum_{k=1}^{\gamma} \frac{Y_k x_k x}{1 - x_k x}$$

所以

$$\begin{aligned} S(x)\sigma(x) &= \left(1 + \sum_{k=1}^{\gamma} \frac{Y_k x_k x}{1 - x_k x} \right) \left(\prod_{i=1}^{\gamma} (1 - x_i x) \right) \\ &= \prod_{i=1}^{\gamma} (1 - x_i x) + \left(\frac{Y_1 x_1 x}{1 - x_1 x} + \frac{Y_2 x_2 x}{1 - x_2 x} + \cdots + \frac{Y_{\gamma} x_{\gamma} x}{1 - x_{\gamma} x} \right) \prod_{i=1}^{\gamma} (1 - x_i x) \\ &= \prod_{i=1}^{\gamma} (1 - x_i x) + \prod_{i=1}^{\gamma} (1 - x_i x) \frac{Y_1 x_1 x}{1 - x_1 x} + \cdots + \prod_{i=1}^{\gamma} (1 - x_i x) \frac{Y_{\gamma} x_{\gamma} x}{1 - x_{\gamma} x} \\ &= \prod_{i=1}^{\gamma} (1 - x_i x) + Y_1 x_1 x \prod_{i=2}^{\gamma} (1 - x_i x) + \cdots + \prod_{i \neq j}^{\gamma} (1 - x_i x) (Y_j x_j x) \\ &\quad + \cdots + Y_{\gamma} x_{\gamma} x \sum_{i=1}^{\gamma-1} (1 - x_i x) \\ &= \sigma(x) + \sum_{j=1}^{\gamma} Y_j x_j x \prod_{\substack{i=1 \\ i \neq j}}^{\gamma} (1 - x_i x) \\ &\equiv \omega(x) \pmod{x^{2\gamma+1}} \end{aligned} \tag{7.5.33}$$

令 $x=x_i^{-1}$ ，则 $\sigma(x_i^{-1})=0$ ，因而上式成为

$$Y_i x_i x_i^{-1} \prod_{\substack{j=1 \\ i \neq j}}^{\gamma} (1 - x_i^{-1} x_j) = \omega(x_i^{-1})$$

由此可得

$$Y_i = \frac{\omega(x_i^{-1})}{\prod_{\substack{j=1 \\ i \neq j}}^{\gamma} (1 - x_i^{-1} x_j)} \tag{7.5.34}$$

要求得上式必须知道 $\sigma(x)$ 的所有根。这在实际中是很不方便的，为此求另一种形式。

$$\begin{aligned} S(x)\sigma(x) &= \left(1 + \sum_{k=1}^{\gamma} \frac{Y_k x_k x}{1 - x_k x}\right) \sigma(x) \\ &= \sigma(x) + \sigma(x) \sum_{k=1}^{\gamma} \frac{Y_k x_k x}{1 - x_k x} \\ &\equiv \omega(x) \pmod{x^{2\gamma+1}} \end{aligned}$$

由于恒等式左边最高次数为 2γ ，故上式成为

$$\begin{aligned} \sigma(x) \sum_{k=1}^{\gamma} \frac{Y_k x_k x}{1 - x_k x} &= \omega(x) - \sigma(x) \\ \frac{Y_i x_i x}{1 - x_i x} \sigma(x) + \sigma(x) \sum_{\substack{j=1 \\ j \neq i}}^{\gamma} \frac{Y_j x_j x}{1 - x_j x} &= \omega(x) - \sigma(x) \end{aligned} \quad (7.5.35)$$

求 $\sigma(x)$ 的形式导数

$$\sigma'(x) = \left(\prod_{j=1}^{\gamma} (1 - x_j x) \right)' = - \sum_{i=1}^{\gamma} x_i \prod_{\substack{j=1 \\ j \neq i}}^{\gamma} (1 - x_j x)$$

令 $x = x_i^{-1}$ ，则上式成为

$$\sigma'(x_i^{-1}) = - x_i \prod_{\substack{j=1 \\ j \neq i}}^{\gamma} (1 - x_j x_i^{-1})$$

所以

$$-\sigma'(x_i^{-1}) x_i^{-1} = \prod_{\substack{j=1 \\ j \neq i}}^{\gamma} (1 - x_j x_i^{-1}) \quad (7.5.36)$$

由式(7.5.35)可得

$$Y_i x_i x \prod_{\substack{j=1 \\ j \neq i}}^{\gamma} (1 - x_j x) + \sigma(x) \sum_{\substack{j=1 \\ j \neq i}}^{\gamma} \frac{Y_j x_j x}{1 - x_j x} = \omega(x) - \sigma(x)$$

令 $x = x_i^{-1}$ ，则上式成为

$$Y_i x_i x_i^{-1} \prod_{\substack{j=1 \\ j \neq i}}^{\gamma} (1 - x_j x_i^{-1}) = \omega(x_i^{-1})$$

把式(7.5.36)代入上式便知

$$Y_i x_i x_i^{-1} (-\sigma'(x_i^{-1})) (x_i^{-1}) = \omega(x_i^{-1})$$

所以错误值

$$Y_i = \frac{-x_i \omega(x_i^{-1})}{\sigma'(x_i^{-1})} \quad (7.5.37)$$

把 7.6 例中已知 $\sigma(x) = 1 + \alpha^{12}x + \alpha^{13}x^2$ ，则 $\sigma'(x) = \alpha^{12}$ ，由 7.6 例中求得 $\omega(x) = 1 + \alpha^2x + x^2$ ，并已知 2 个逆根(即错误位置数) $x_1 = \alpha^3$, $x_2 = \alpha^{10}$ 。所以

$$Y_1 = \frac{\alpha^3(1 + \alpha^2\alpha^{-3} + \alpha^{-6})}{\alpha^{12}} = \alpha^7$$

同理可求得 $Y_2 = \alpha^{11}$ ，这与用式(7.5.32)求得的值完全相同。

现在我们已得到了求 Y_i 的式(7.5.32), 式(7.5.34), 式(7.5.37)3 个表达式。应当指出，当 $g(x)$ 的 $\delta-1$ 个连续根中若 $m_0 \neq 1$ 时，则这 3 个公式的右边必须乘 $x_i^{-m_0+1}$ 。由于用式(7.5.32)计算仅需进行四则运算，因此它通常应用于实际译码器设计中。而式(7.5.34)

与式(7.5.37)的计算比较方便, 所以通常可用以检验所求的 Y_i 值是否正确。

上面我们讨论了 BCH 码译码的主要步骤和算法。表 7-11 给出了应用专用译码器纠正 t 个错误的 q 进制 BCH 码时, 每一主要步骤所需的移存器级数和所需的移位节拍次数。

表 7-11 BCH 码译码主要步骤的复杂性和时间

| 译码步骤 | 移存器级数 | 移位次数 | 计算量 | |
|----------------|----------|---------------|------------|--------------|
| | | | 乘法 | 加法 |
| 计算伴随式 | $2mt$ | n | $2tn$ | $2t(n-1)$ |
| 计算 $\sigma(x)$ | $4mt$ | $2mt$ | $2t^2$ | $2t^2$ |
| 求 x_i | mt | mn (或 k) | nt | kt |
| 计算错误值 Y_i | mt | $6mt$ | $2t^2$ | $3t^2$ |
| 存贮码字 | $2n$ | | | |
| 总计 | $2n+8mt$ | $(1+m)n+8mt$ | $3nt+4t^2$ | $2nt+5t^2-t$ |

对一个本原 BCH 码来说, $m \approx \log_q n$ 。若 $t \approx pn$, $0 < p < 0.25$, 那么 BCH 码译码器的复杂性和所需时间可用以下公式进行估计:

$$\text{移存器级数} = n(2 + 7p \log_q n)$$

$$\text{移位次数} = n(1 + \log_q n(1 + 8p))$$

因此译码器的译码速度和复杂性都随 $n \log_q n$ 而增加, 对大的 n 来说, 复杂性仅比随码长的线性增长稍大一点, 这为 BCH 码用于实际提供了良好的条件。若我们不采用专用译码设备而应用通用计算机完成译码, 则运算速度大约随 $n^2 \log_q n$ 增加。例如, 若译码设备完成加法指令需 $2 \mu s$, 其它指令均需 $.2 \mu s$, 则它的比特传输率大约可达 10^5bit/s , 若用通用计算机处理大约每秒可达几千比特。

图 7-11 给出了 BCH 码译码器的主要组成部分。若为二进制 BCH 码, 则第四步计算 Y_i 可以省略, 此框图可以从图中去掉。

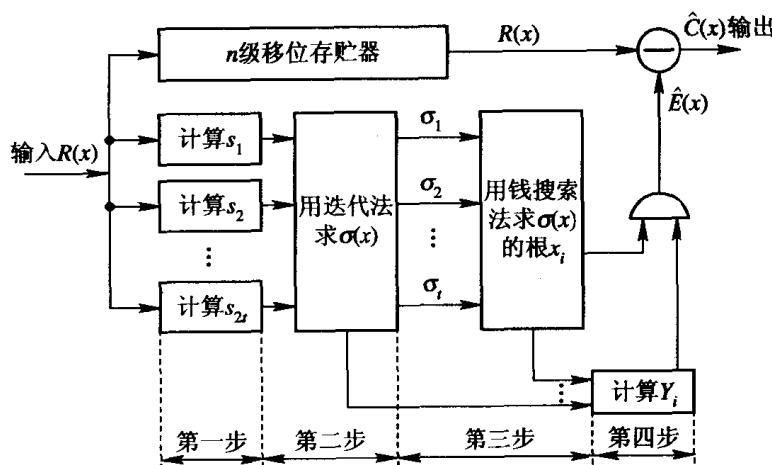


图 7-11 BCH 码译码框图

姚明余等于 1986 年提出了一种逐位判决逐位纠正的 BCH 码译码算法，这种算法的译码速度较迭代译码稍快，但要求有较多的存贮设备，有关这种算法可参阅[13]。

虽然用 BM 算法能较快地得到 $\sigma(x)$ ，但电路实现仍较复杂。因此对于纠错能力 $t \leq 3$ 的码，用解式(7.4.10)的线性方程组方法，可能更为简单^[14]。

* § 7.6 BCH 码的纠错纠删译码

距离为 $d = 2t + e + 1$ 的线性分组码，能纠正 t 个错误同时纠正 e 个删除，若全用来纠删则可以纠正 $d - 1$ 个删除。在二进制码中，纠错纠删译码比较简单和直观，能用迭代算法较快地求得 $\sigma(x)$ 。

设接收 n 重 $R(x)$ 中含有 $\leq t$ 个错误和 $\leq e$ 个删除。删除的位置是已知的，仅不知其值的大小，故可在删除位置上任意填上 e 个二进制（如为 q 进制码，则为 q 进制）数字，为简单起见填上 e 个 0。如果发的码字在这 e 个删除位置中有 f 个 1、 $e - f$ 个 0，则用全为 0 填充的删除位置中有 f 个错误码元，再加上原有的 t 个错误共有 $t + f$ 个错误。若 $t + f > \lfloor (d - 1)/2 \rfloor$ ，则码不能纠正，这时在删除位置中把全为 0 改为全为 1，此时填充删除位置后的 $R(x)$ 中有 $t + e - f$ 个错误，由于

$$d = 2t + e + 1$$

且已知

$$t + f > \lfloor (d - 1)/2 \rfloor \quad f > \lfloor (d - 1)/2 \rfloor - t$$

所以

$$t + e - f = d - 1 - (t + f) \leq d - 1 - \lfloor (d - 1)/2 \rfloor = \lfloor (d - 1)/2 \rfloor$$

说明此时 $R(x)$ 中的错误个数必小于 $\lfloor (d - 1)/2 \rfloor$ ，这在码的纠错能力以内，可以用迭代算法找到 $\sigma(x)$ 。

由上可知，二进制 BCH 码的纠错纠删译码很简单，把收到的 $R(x)$ 中删除位置全填上 0，并送到译码器译码。若译码器的纠错个数 $\leq \lfloor (d - 1)/2 \rfloor$ ，则说明此时译码器译码正确，送出译完的码字；反之，则说明译码错误，这时把 $R(x)$ 中的删除位置全填上 1，再送入译码器译码，译得的码字即为所求。

但对多进制码并非如此简单。首先，必须对伴随式进行修正，并利用迭代算法求出错误位置多项式 $\sigma(x)$ 。但与前节介绍的方法并不相同，它必须包含两个错误位置多项式：一是删除位置多项式，另一是错误位置多项式。现在举例说明译码过程。

设 GF(2⁴) 上的 [15, 7, 9] RS 码，它以 $\alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6, \alpha^7, \alpha^8$ 为根， α 是 GF(2⁴) 中的本原元，它是 $x^4 + x + 1$ 的根。码的生成多项式

$$\begin{aligned} g(x) &= \prod_{i=1}^8 (x - \alpha^i) \\ &= x^8 + \alpha^{14}x^7 + \alpha^2x^6 + \alpha^4x^5 + \alpha^2x^4 + \alpha^{13}x^3 + \alpha^5x^2 + \alpha^{11}x + \alpha^6 \end{aligned}$$

该码的 $d = 9$ ，能纠正 2 个随机错误和 4 个删除。

设接收的 $R(x) = \alpha^{13}x^{14} + \alpha^9x^{13} + e_{12}x^{12} + \alpha^{12}x^{11} + e_{10}x^{10} + \alpha^7x^9 + \alpha^9x^7 + \alpha^{13}x^6 + \alpha^{10}x^5 + \alpha x^4 + \alpha^{14}x^3 + e_2x^2 + \alpha^4x$ ，式中， e_{12}, e_{10}, e_2 表示这几位是删除。

令在删除位置上的符号置零(当然可置其它值,但置零时计算方便)。由此可得伴随式: $s_1 = \alpha^6$, $s_2 = \alpha^2$, $s_3 = \alpha^6$, $s_4 = \alpha^5$, $s_5 = \alpha^{14}$, $s_6 = \alpha^5$, $s_7 = \alpha^{13}$, $s_8 = \alpha^5$ 。作删除位置多项式

$$\begin{aligned}\sigma_e(x) &= \prod_{j=1}^e (1 - x_j x) = 1 + \sigma_{e1}x + \cdots + \sigma_{ee}x^e \\ &= (1 - \alpha^2 x)(1 - \alpha^{10} x)(1 - \alpha^{12} x) = 1 + \alpha^6 x + \alpha^{13} x^2 + \alpha^9 x^3\end{aligned}\quad (7.6.1)$$

得 $\sigma_{e1} = \alpha^6$, $\sigma_{e2} = \alpha^{13}$, $\sigma_{e3} = \alpha^9$ 。

定义修正伴随式

$$T_j = \sum_{i=0}^e \sigma_{ei} s_{j-i} \quad j = e+1, e+2, \dots, d-1 \quad (7.6.2)$$

式中, e 是实际产生删除的个数, 该例中是 3。由此可得:

$$\begin{aligned}T_4 &= s_4 + s_3 \sigma_{e1} + s_2 \sigma_{e2} + s_1 \sigma_{e3} = \alpha^{14} \\ T_5 &= s_5 + s_4 \sigma_{e1} + s_3 \sigma_{e2} + s_2 \sigma_{e3} = \alpha^9 \\ T_6 &= s_6 + s_5 \sigma_{e1} + s_4 \sigma_{e2} + s_3 \sigma_{e3} = \alpha^{14} \\ T_7 &= s_7 + s_6 \sigma_{e1} + s_5 \sigma_{e2} + s_4 \sigma_{e3} = \alpha^8 \\ T_8 &= s_8 + s_7 \sigma_{e1} + s_6 \sigma_{e2} + s_5 \sigma_{e3} = \alpha\end{aligned}$$

现在可以用迭代方法求错误位置多项式 $\sigma_t(x)$ 。利用式(7.5.13)和式(7.5.12)求 $\sigma^{(j+1)}(x)$ 和 d_j , 但计算时用 T_{e+i} 代替这两式中的 s_i ($i=1, 2, \dots, e$), 其计算过程如表 7-12 所示。求得

$$\sigma_t(x) = 1 + \alpha^3 x + \alpha^6 x^2 = (1 - \alpha^8 x)(1 - \alpha^{13} x)$$

表 7-12 求 $\sigma_t(x)$ 的迭代过程

| j | $\sigma_t^{(j)}(x)$ | $D(j)$ | $j-D(j)$ | d_j | i |
|-----|---------------------------------------|--------|----------|-------------------------|-----|
| -1 | 1 | 0 | -1 | 1 | |
| 0 | 1 | 0 | 0 | $T_{e+1} = \alpha^{14}$ | |
| 1 | $1 + \alpha^{14} x$ | 1 | 0 | α^{10} | -1 |
| 2 | $1 + \alpha^{10} x$ | 1 | 1 | α^9 | 0 |
| 3 | $1 + \alpha^{11} x + \alpha^{13} x^2$ | 2 | 1 | α^{14} | 1 |
| 4 | $1 + \alpha^3 x + \alpha^6 x^2$ | 2 | 2 | | |

为了求得错误值 Y_i 及删除值 e_i , 作多项式

$$\begin{aligned}\sigma(x) &= \sigma_e(x) \sigma_t(x) \\ &= 1 + \sigma_{e1}x + \sigma_{e2}x^2 + \cdots + \sigma_{et}x^{t+e} \\ &= (1 + \alpha^6 x + \alpha^{13} x^2 + \alpha^9 x^3)(1 + \alpha^3 x + \alpha^6 x^2) \\ &= 1 + \alpha^2 x + \alpha^7 x^2 + \alpha^{10} x^3 + \alpha^6 x^4 + x^5\end{aligned}$$

可知:

$$\sigma_1 = \alpha^2, \sigma_2 = \alpha^7, \sigma_3 = \alpha^{10}, \sigma_4 = \alpha^6, \sigma_5 = 1$$

$\partial \cdot \sigma(x) = 5$ 说明错误加删除有 5 个, 通过线搜索可求得错误位置数:

$$X_1 = \alpha^2, X_2 = \alpha^7, X_3 = \alpha^{10}, X_4 = \alpha^{12}, X_5 = \alpha^{13}$$

计算:

$$\begin{aligned}
\sigma_{11} &= \sigma_1 - X_1 = 0, & \sigma_{12} &= \sigma_2 - \sigma_{11}X_1 = \alpha^7 \\
\sigma_{21} &= \sigma_1 - X_2 = 1, & \sigma_{22} &= \sigma_2 - \sigma_{21}X_2 = \alpha^{11} \\
\sigma_{31} &= \sigma_1 - X_3 = \alpha^4, & \sigma_{32} &= \sigma_2 - \sigma_{31}X_3 = \alpha \\
\sigma_{41} &= \sigma_1 - X_4 = \alpha^7, & \sigma_{42} &= \sigma_2 - \sigma_{41}X_4 = \alpha^3 \\
\sigma_{51} &= \sigma_1 - X_5 = \alpha^{14}, & \sigma_{52} &= \sigma_2 - \sigma_{51}X_5 = \alpha^2 \\
\sigma_{13} &= \sigma_3 - \sigma_{12}X_1 = \alpha^{13}, & \sigma_{14} &= \sigma_4 - \sigma_{13}X_1 = \alpha^{13} \\
\sigma_{23} &= \sigma_3 - \sigma_{22}X_2 = \alpha^2, & \sigma_{24} &= \sigma_4 - \sigma_{23}X_2 = \alpha^7 \\
\sigma_{33} &= \sigma_3 - \sigma_{32}X_3 = \alpha^{14}, & \sigma_{34} &= \sigma_4 - \sigma_{33}X_3 = \alpha^5 \\
\sigma_{43} &= \sigma_3 - \sigma_{42}X_4 = \alpha^5, & \sigma_{44} &= \sigma_4 - \sigma_{43}X_4 = \alpha^3 \\
\sigma_{53} &= \sigma_3 - \sigma_{52}X_5 = \alpha^5, & \sigma_{54} &= \sigma_4 - \sigma_{53}X_5 = \alpha^2
\end{aligned}$$

代入式(7.5.32)求得:

$$Y_1 = \alpha^8, Y_2 = \alpha^2, Y_3 = \alpha^5, Y_4 = \alpha^{11}, Y_5 = \alpha^5$$

最后得到错误图样

$$\hat{E}(x) = \alpha^5x^{13} + \alpha^{11}x^{12} + \alpha^5x^{10} + \alpha^2x^8 + \alpha^8x^2$$

译码器输出的码字

$$\begin{aligned}
\hat{C}(x) &= R(x) - \hat{E}(x) \\
&= \alpha^3x^{14} + \alpha^6x^{13} + \alpha^{11}x^{12} + \alpha^{12}x^{11} \\
&\quad + \alpha^5x^{10} + \alpha^2x^8 + \alpha^9x^7 + \alpha^{13}x^6 + \alpha^{10}x^5 \\
&\quad + \alpha x^4 + \alpha^{14}x^3 + \alpha^8x^2 + \alpha^4x
\end{aligned}$$

可以检验它是 $g(x)$ 的倍式, 译码器输出是合理的。

可以把以上的纠错纠删译码过程用图 7-12 的流程表示, 结果表明纠删纠错译码并不比纠错译码简单, 甚至更复杂, 这说明码的纠错能力与复杂性是矛盾的。

§ 7.7 BCH 码的频域译码

上两节讨论的译码方法是在时域中进行的, 也就是把接收矢量 $R=C+E$, 以及 C 和 E 都看成是时域中的信号, 时域译码器的任务就是从 R 中以最小的译码错误概率确定出 \hat{E} , 然后由 R 和 \hat{E} 得到译码器的输出 $\hat{C}=R-\hat{E}$ 。在这种译码过程中, 无论是求 $\sigma(x)$ 和确定出错误位置和错误值都是在时域中进行, 中间没有经过任何的频谱变换。

时域译码中的第一步是由 $R=C+E$ 计算伴随式, 由式(7.4.1)可知:

$$\begin{aligned}
S &= R \cdot H^T = E \cdot H^T \\
&= (s_{m_0}, s_{m_0+1}, \dots, s_{m_0+2t-1}) = (s_1, s_2, \dots, s_{2t})
\end{aligned}$$

这里设 $m_0=1$, 式中

$$\begin{aligned}
s_j &= \sum_{i=0}^{n-1} e_i (\alpha^j)^i = \sum_{i=1}^t e_{l_i} (\alpha^j)^i \quad j = 1, 2, \dots, 2t \\
E &= (e_{n-1}, e_{n-2}, \dots, e_0) \quad e_{l_i} \neq 0, i = 1, 2, \dots, t
\end{aligned} \tag{7.7.1}$$

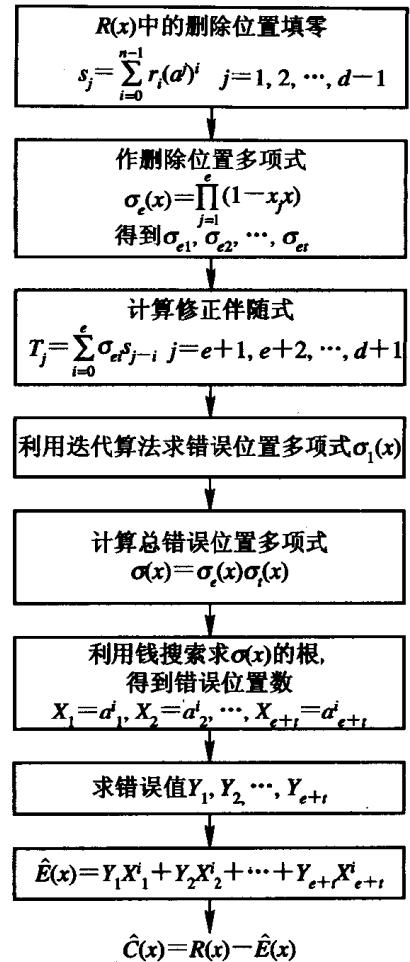


图 7-12 纠删纠错译码流程图

把式(7.7.1)与式(5.8.3)比较可知, $2t$ 个伴随式分量恰好是 \mathbf{E} 的 MS 多项式离散傅里叶变换(DFT)中的 $2t$ 个分量。

如果对 $\mathbf{R}=\mathbf{C}+\mathbf{E}$ 进行离散傅里叶变换, 则由 § 5.8 节中 DFT 或 MS 多项式的性质 2 可知 \mathbf{R} 的 DFT

$$\tilde{\mathbf{R}} = \tilde{\mathbf{C}} + \tilde{\mathbf{E}} \quad (7.7.2)$$

式中, $\tilde{\mathbf{C}}$ 和 $\tilde{\mathbf{E}}$ 分别是 \mathbf{C} 和 \mathbf{E} 的 DFT。每个分量的 DFT 为

$$\tilde{r}_i = \tilde{c}_i + \tilde{e}_i \quad i = 0, 1, \dots, n-1$$

而伴随式 $s_j (j=1, 2, \dots, 2t)$ 是 $\tilde{\mathbf{R}}$ 中从 l_1 到 l_1+2t-1 的 $2t$ 个量。由 BCH 码的性质可知, 码以 $\alpha, \alpha^2, \dots, \alpha^{2t}$ 为根, 根据定理 5.8.3 可知 $\tilde{\mathbf{C}}$ 中的 $2t$ 个频率分量等于零。

$$\tilde{c}_i = 0 \quad i = l_1, l_1 + 1, l_1 + 2, \dots, l_1 + 2t - 1$$

因而

$$s_j = \tilde{r}_{j+l_1-1} = \tilde{e}_{j+l_1-1} \quad j = 1, 2, \dots, 2t \quad (7.7.3)$$

可知 $2t$ 个伴随式分量就是 $\tilde{\mathbf{E}}$ 中的 $2t$ 个分量。如果码的距离 $d_{\text{BCH}} \geq 2t+1$, 则当 $w(\mathbf{E}) \leq t$ 时, 那么通过这 $2t$ 个分量我们就能唯一地确定出 $\tilde{\mathbf{E}}$, 并由 $\tilde{\mathbf{R}} - \tilde{\mathbf{E}} = \tilde{\mathbf{C}}$, 再经 DFT^{-1} 由 $\tilde{\mathbf{C}}$ 恢复出 \mathbf{C} 。这就是 BCH 码频域译码的基本原理。

现在的问题是如何由 $\tilde{\mathbf{E}}$ 中的已知 $2t$ 个分量, 求出 $\tilde{\mathbf{E}}$ 的所有 n 个分量的序列 (\tilde{e}_i) , 且在 $l_1, l_1+1, \dots, l_1+2t-1$ 位置上与 s_1, s_2, \dots, s_{2t} 相同。这可以用线性递归扩展方法求得。

设 $e \leq t$ 个错误, 产生在位置 $\alpha^i, i=1, \dots, e$, 则错误位置多项式

$$\sigma(x) = \prod_{i=1}^e (1 - x\alpha^i) = 1 + \sigma_1x + \sigma_2x^2 + \dots + \sigma_ex^e$$

如果把 $\sigma(x)$ 的系数写成向量形式 $\sigma = (\sigma_0, \sigma_1, \dots, \sigma_e, 0, 0, \dots, 0)$, 并且把它看成是频域表示。由于 $\sigma(\alpha^{-i}) = 0$, 因此当对 σ 进行 DFT^{-1} 时, 得到的序列 $\lambda = (\lambda_0, \lambda_1, \dots, \lambda_{n-1})$, 就是 σ 的时域表示, 且 $\lambda_i = 0$, 对于 $i=1, 2, \dots, e$, 而其它分量值不为零。对序列 $\mathbf{E} = (e_0, e_1, \dots, e_{n-1})$ 来说, 若 $e_i \neq 0, i=1, 2, \dots, e$, 则相应的 $\lambda_i = 0$; 反之, 若 $e_i = 0$, 则 $\lambda_i \neq 0$ 。因此, 在时域上有以下关系:

$$\lambda_i e_i = 0 \quad i = 0, 1, \dots, n-1 \quad (7.7.4)$$

根据 DFT 的性质, 即 § 5.8 中 MS 的性质 3(卷积定理)可知, 它们在频域中卷积为零。

$$\sum_{j=0}^{n-1} \sigma_j \tilde{e}_{l-j} = 0 \quad l = 0, 1, \dots, n-1$$

又 $\sigma(x) \leq t$, 所以对 $j > t$ 有 $\sigma_j = 0$, 故有

$$\sum_{j=0}^t \sigma_j \tilde{e}_{l-j} = 0$$

$\sigma_0 = 1$, 所以上式可写成

$$\tilde{e}_l = - \sum_{j=1}^t \sigma_j \tilde{e}_{l-j} \quad l = 0, 1, \dots, n-1 \quad (7.7.5)$$

这是一个有 n 个方程的线性递推式, 共有 $n-t$ 个未知量, 它们是 $\sigma(x)$ 的 t 个系数: $\sigma_1, \dots, \sigma_t$ 和 $\tilde{\mathbf{E}}$ 的 $n-2t$ 个分量。而该式中 $\tilde{\mathbf{E}}$ 的 $2t$ 个已知分量由式(7.7.3)可知是伴随式。

$$s_l = - \sum_{j=1}^t \sigma_j s_{l-j} \quad l = t+1, \dots, 2t \quad (7.7.6)$$

与式(7.5.5)完全相同, 仅含有已知的 $2t$ 个伴随式和未知的 t 个系数: $\sigma_1, \dots, \sigma_t$ 。正如在

§ 7.5节中所指出的，如果把已知的伴随式看成是一个长为 $2t$ 的序列 $S^{(2t)}$ ，则用 BM 算法得到的 $\sigma(x)$ ，就是由已知 $\{s_1, s_2, \dots, s_t\}$ 得到序列 $S^{(2t)}$ 的最短 LFSR 的联接多项式，且由式 (7.5.27) 可知：

$$\partial \circ \sigma(x) = L(S^{(2t)})$$

因此可以通过 BM 算法得到 $\sigma(x)$ ，且 \tilde{E} 中的其余的 $n-2t$ 个分量可以由式 (7.7.5) 求出。也就是由已知的 $s_t, s_{t+1}, \dots, s_{2t}$ ，得到 s_{2t+1} ，由 $s_{t+1}, s_{t+2}, \dots, s_{2t}, s_{2t+1}$ 得到 s_{2t+2} 等等。这就是求满足式 (7.7.5) 的线性递归扩展方法。由于用 $\sigma(x)$ 联接多项式所组成的 LFSR 是最短的，因此由该移存器产生的序列 \tilde{E} 的线性复杂度也是最小的，所以由 Blahut 定理可知，由 \tilde{E} 的 DFT^{-1} 得到的 \hat{E} 有最小的汉明重量，这正是最大似然译码器所要求的 \hat{E} ，由此我们就求得了时域中的错误矢量 \hat{E} ，并由 $R - \hat{E} = \hat{C}$ 得到译码器的输出 \hat{C} 。

当然，我们也可以由 $\tilde{R} - \tilde{E} = \tilde{C}$ ，得到频域的 \tilde{C} ，然后再通过 DFT^{-1} 得到时域译码器输出 \hat{C} 。

从上讨论可知，频域译码器的关键一步也就是利用 BM 算法，求序列的联接多项式 $\sigma(x)$ ，这与时域译码中的迭代译码完全相同；不同的是时域译码中得到 $\sigma(x)$ 后还需求出它的根，以得到 $E(x)$ 的错误位置以及解出错误值。而在频域译码中，得到 $\sigma(x)$ 后，利用式 (7.7.5) 得到 \tilde{E} 的其余 $n-2t$ 个分量，然后利用 $\tilde{R} - \tilde{E} = \tilde{C}$ ，再进行 DFT^{-1} ，由 \tilde{C} 得到 \hat{C} 。或者由 \tilde{E} 进行 DFT^{-1} 得到 \hat{E} ，而不必再求错误值和错误位置。

DFT 与 DFT^{-1} 有很多已知的快速算法，因此对某些码长的 RS 码来说，频域译码可以做得比时域译码器更快。但是对于二进制码来说，时域译码器似乎比频域译码器要简单，译码速度可能也更快。

综上讨论可知，有 4 种 BCH 码的编—译码方式：时域—时域，频域—频域，频域—时域和时域—频域。在实际中要何种类型的编—译码器视具体情况而定。

到此为止，我们讨论了 BCH 码的时域和频域译码。对时域的最大似然译码器来说，就是由接收矢量 $R = C + E$ 中寻找有最轻重量的错误图样 \hat{E} ；而对频域的最大似然译码器来说，就是由 $\tilde{R} = \tilde{C} + \tilde{E}$ 中寻找线性复杂度最低的序列 (\tilde{E}) ，由 Blahut 定理可知 $w(\hat{E}) = L(\tilde{E})$ ，因此这两种译码方法完全等价。而这两种译码方法的关键一步都是利用 BM 算法求 $\sigma(x)$ 。

§ 7.8 超 BCH 限译码

由以上讨论可知，无论是时域或频域译码，或者无论是彼得逊的基本译码还是 BM 译码，在求解错误位置多项式时，仅考虑了 s_1 至 s_{2t} 个连续的伴随式分量，因此仅能译到 BCH 限所决定的纠错能力。如果在译码时不仅考虑 $2t$ 个连续的伴随式分量而且利用了所有（连续与不连续的）伴随式分量，则可以预料此时应能译到码的实际最小距离所决定的纠错能力。另一方面从频域译码或从序列的最短线性反馈移存器综合来看，达到 BCH 限的译码问题，就是求产生序列 $S^{(2t)}$ 的最短 LFSR，或它的联接多项式 $\sigma(x)$ ，也就是求一个序列的

最短 LFSR 的综合问题。

至于要达到 HT 限、Roos 限或 LW 限甚至实际最小距离的译码问题，由于在分析距离限时，利用了根集中的多个连续根集，因此，译码时必须考虑多个序列的最短线性移存器综合问题。该问题最初由哈特曼和曾开明在 1974 年译 $d_{HT} \leq 4$ 以及 $d_{HT} = s+3$ 的 BCH 码时所考虑，到了 1981 年冯贵良推广了他们的结果解决了达到 HT 限的 BCH 码的译码问题。以后，冯贵良和曾开明对该问题继续进行深入研究，利用广义迭代译码算法和广义欧几里德算法，或广义彼得逊译码算法解决了多序列的最短 LFSR 的综合问题^[15]，利用这些方法可译到 d_{Roos} 甚至实际最小距离的 BCH 码的纠错能力。

下面我们从频域角度讨论达到 HT 限的 BCH 码译码或多个序列的最短 LFSR 的综合问题。

定理 7.8.1(多序列线性复杂度定理) 设 $(a^n)^\infty$ 是周期为 n 的半无限长序列，它含有 s 个长度均为 m 的子序列，且相继两个子序列间的相应元素均相隔 a 位， $(n, a) = 1$ 。若 $l(l < m)$ 是每个子序列满足系数为 h_1, h_2, \dots, h_l 的线性递推式(5.9.1)的最小整数，则或者整个序列 $(a^n)^\infty$ 满足有同样系数的式(5.9.1)，且 $L((a^n)^\infty) = l$ ；或者 $L((a^n)^\infty) \geq m+s-l$ 。

从 LFSR 综合来说，若最小组数为 l 的同一个 LFSR，生成如上所述的 s 个子序列中的每一个，则或者该 LFSR 生成整个 $(a^n)^\infty$ 序列，且 $L((a^n)^\infty) = l$ ；或者 $L((a^n)^\infty) \geq m+s-l$ 。

定理 7.8.2(多序列译码定理) 设一个周期序列 $(\tilde{E}^n)^\infty$ ，它含有 s 个子序列，且 $L((\tilde{E}^n)^\infty) \leq \lfloor (m+s-1)/2 \rfloor$ 。若每个子序列的长度为 m ，且相继子序列的相应元素相隔 a 位， $(n, a) = 1$ 。如果 $l(l < m)$ 是每个子序列满足系数为 h_1, h_2, \dots, h_l 的线性递推式(5.9.1)的最小整数，则 $(\tilde{E}^n)^\infty$ 亦满足带有同样系数的线性递推式，且 $L((\tilde{E}^n)^\infty) = l$ 。

从 LFSR 综合来说，当给定周期序列 $(\tilde{E}^n)^\infty$ 的 s 个子序列，这里 $L((\tilde{E}^n)^\infty) \leq \lfloor (m+s-1)/2 \rfloor$ ，若每个子序列长为 m ，且相继子序列的对应元素相隔 a 位， $(n, a) = 1$ ，且 l 是生成每个子序列的 LFSR 的最小组数，则同一个 LFSR 亦生成 $(\tilde{E}^n)^\infty$ ，且 $L((\tilde{E}^n)^\infty) = l$ 。

上述两个定理的证明可参看^[4]。注意，上述定理中，条件 $L((\tilde{E}^n)^\infty) \leq \lfloor (m+s-1)/2 \rfloor$ ，保证了当 $s < m$ 时， $L((\tilde{E}^n)^\infty) = l < m$ 。然而，如果 $s > m$ ，则对 $(\tilde{E}^n)^\infty$ 序列每隔 a 位采样，得到一个采样序列 $(\tilde{E}_a^n)^\infty$ ，根据采样定理 5.9.5 可知， $L((\tilde{E}_a^n)^\infty) = L((\tilde{E}^n)^\infty)$ ，利用生成成长为 s 的 m 个已知序列的 LFSR 能生成 $(\tilde{E}_a^n)^\infty$ 序列。而对 $(\tilde{E}_a^n)^\infty$ 序列每隔 $a^{-1}(\bmod n)$ 位采样，可以得到序列 $(\tilde{E}^n)^\infty$ ，这里 a^{-1} 是 a 的乘法逆元。

由多序列译码定理可以得到能译至 HT 限的 BCH 码的译码算法。如果码满足 HT 限，则生成多项式 $g(x)$ 所产生的频域序列 $(G^n)^\infty$ 中，含有 s 个 $(0^m)(\Delta^{m-n}) = (p^m)$ 图样，且 $d_{HT} = m+s$ 。所以译码器能纠正直至 $\lfloor (m+s-1)/2 \rfloor$ 个错误。换言之，当 $L((\tilde{E}^n)^\infty) \leq \lfloor (m+s-1)/2 \rfloor$ 时，译码器必定能找到有最低线性复杂度的 $(\tilde{E}^n)^\infty$ 序列。在 $(G^n)^\infty$ 中的图样 (P^n) ，对 HT 限来说，保证了 $(\tilde{E}^n)^\infty$ 的 s 个已知序列，满足多序列译码定理所要求的假设条件。因此，生成已知 s 个子序列的最短 LFSR 能生成 $(\tilde{E}^n)^\infty$ 序列。

由此可以得到能译至 HT 限的 BCH 码译码算法：

- (1) 把接收序列 $R = C + E$ 通过 DFT 变换成频域序列 $\tilde{R} = \tilde{C} + \tilde{E}$ 。
- (2) 由 $S^T = H \cdot R^T = H \cdot E^T$ 可知， $(\tilde{E}^n)^\infty$ 序列中的 s 个子序列 $(\tilde{E})_1, (\tilde{E})_2, \dots, (\tilde{E})_s$ 等于 $(\tilde{R})^\infty$ 中的相应的子序列。
- (3) 若子序列的数目 s 大于子序列的长度 m ，则对 $(E^n)^\infty$ 序列每隔 a 位采样得到采样序

列 $(\tilde{\mathbf{E}}_a^m)^\infty$, 并且交换 s 和 m 。

(4) 找出能生成子序列 $(\tilde{\mathbf{E}})_1, (\tilde{\mathbf{E}})_2, \dots, (\tilde{\mathbf{E}})_s$ 的最短 LFSR $[\sigma(x), l]$, 即它有联接多项式为 $\sigma(x)$, $\sigma \circ \sigma(x)=l$, 由 l 级移存器组成。

(5) 用 LFSR $[\sigma(x), l]$ 生成整个序列 $(\tilde{\mathbf{E}}^m), (\tilde{\mathbf{E}}_a^m)$ 。

(6) 若采用第(3)步的采样序列, 则 $(\tilde{\mathbf{E}}_a^m)^\infty$ 用 a^{-1} 采样, 得到采样序列 $(\tilde{\mathbf{E}}^m)$ 。

(7) 译码 $\tilde{\mathbf{C}}=\tilde{\mathbf{R}}-\tilde{\mathbf{E}}$, 对 $\tilde{\mathbf{C}}$ 进行 DFT $^{-1}$ 得到 $\hat{\mathbf{C}}$ 。

现以综合 $s=2$ 个子序列为例, 说明译码过程。设每个子序列长为 m , $s=2$ 。由 HT 限可知, 相应码的最小距离 $d_{HT} \geq m+2$, 能纠正 $t=\lfloor(m+1)/2\rfloor$ 个错误, 也就是我们能够找到 $\tilde{\mathbf{E}}^m$, 它的线性复杂度 $L(\hat{\mathbf{E}}^m)^\infty \leq \lfloor(m+1)/2\rfloor$ 。

设输入频域译码器的伴随式序列

$$\begin{array}{c} \tilde{e}_i \tilde{e}_{i+1} \cdots \tilde{e}_{i+m-1} \Delta_{i+m} \cdots \Delta_{i+a-1} e_{i+a} e_{i+a-1} \cdots e_{i+a+m-1} \Delta_{i+a+m} \\ | \quad | \\ \leftarrow (\tilde{\mathbf{E}})_1 \rightarrow \quad \quad \quad \leftarrow (\tilde{\mathbf{E}})_2 \rightarrow \end{array}$$

由式(7.7.3)可知, 其中 \tilde{e}_i 为相应的伴随式。首先对 $(\tilde{\mathbf{E}})_1$ 序列应用 BM 算法, 得到一个 LFSR $[\sigma_1(x), l_1]$ 。如果 LFSR $[\sigma_1(x), l_1]$ 亦生成序列 $(\tilde{\mathbf{E}})_2$, 则译码的第(4)步完成, 并用 LFSR $[\sigma_1(x), l_1]$ 生成整个 $\tilde{\mathbf{E}}^m$ 序列。若 LFSR $[\sigma_1(x), l_1]$ 不能生成 $(\tilde{\mathbf{E}})_2$, 则我们可以证明 $(\tilde{\mathbf{E}}^m)^\infty$ 的线性复杂度是 $l_2 \geq m-l_1+1$ 。另一方面 $l_2 \geq l_1$, 因此 $l_2 \geq \max\{l_1, m-l_1+1\}$, 所以仅当 $l_2=l_1=\lfloor(m+1)/2\rfloor$ 时, 才能满足 $l_1 \leq \lfloor(m+1)/2\rfloor$ 和 $l_2 \leq \lfloor(m+1)/2\rfloor$ 的假设。

如果 LFSR $[\sigma_1(x), l_1]$ 生成 $(\tilde{\mathbf{E}})_1$ 和 $(\tilde{\mathbf{E}}^m)$ 的第 $(i+m+1)$ 个分量 \tilde{e}_{i+m} , 但不能生成 $(\tilde{\mathbf{E}})_2$, 则 $l_2 \geq \max\{l_1, m-l_1+2\}$, 这与我们的假设相矛盾, 所以由 LFSR $[\sigma_1(x), l_1]$ 产生的下一个差值 $d_m \neq 0$, 根据 BM 算法, 新的联接多项式由式(7.5.13)可知为

$$\sigma_2(x) = \sigma_1(x) - d_m d_i^{-1} x^{m-i} \sigma^{(i)}(x)$$

式中, d_i 是第 i 步的差值, $\sigma^{(i)}(x)$ 是相应的联接多项式。唯一能够自由选取的值是 d_m , 选择 d_m 是等价于选择 Δ_{i+m} , 但 d_m 可以用 $\sigma_2(x)$ 生成 $(\tilde{\mathbf{E}})_2$ 时找到。最后得到的 LFSR $[\sigma_2(x), l_2]$ 就是所求的 LFSR, 由它生成整个序列 $(\tilde{\mathbf{E}}^m)$, 完成了译码的第(5)步。

例 7.10 二进制 $[17, 9, 5]$ BCH 码, 它的生成多项式 $g(x)$ 的根集为 $\{\beta^1, \beta^2, \beta^4, \beta^8, \beta^9, \beta^{13}, \beta^{15}, \beta^{16}\}$, 这里, $\beta^{17}=1$, $\beta \in GF(2^8)$, $\beta=\alpha^{15}$, $\alpha \in GF(2^8)$ 是域中的本原元, $\alpha^{255}=1$, α 是 $p(x)=x^8+x^4+x^3+x^2+1$ 的根。

由码的根集可知, $d_{BCH} \geq 3$, 由 HT 限可得 $d_{HT} \geq 5$, 该码的实际距离就是 5。由码字的 DFT 所得的谱可知, 每一码字的谱表示中一定有以下形式的图样: $[\Delta 00 \Delta 0 \Delta \Delta \Delta 00 \Delta \Delta \Delta 0 \Delta 00]$, 它含有 $s=3$ 组长为 $m=2$ 的 0 串。且每一个 0 串之间相隔 $a=7$ 位, $(17, 7)=1$ 。

设接收到的序列 $\mathbf{R}=(1100111111111111)$, 则译码器完成以下译码步骤:

(1) 由 \mathbf{R} 计算 DFT 得到

$$(\tilde{\mathbf{R}}^m) = [1, \alpha^{63}, \alpha^{126}, \alpha^{121}, \alpha^{252}, \alpha^{94}, \alpha^{242}, \alpha^{203}, \alpha^{249}, \alpha^{159}, \alpha^{188}, \alpha^{47}, \alpha^{229}, \alpha^{207}, \alpha^{151}, \alpha^{231}, \alpha^{243}]$$

(2) $s=3$ 个子序列为: $(\tilde{\mathbf{E}})_1 = (\alpha^{63}, \alpha^{126})$, $(\tilde{\mathbf{E}})_2 = (\alpha^{249}, \alpha^{159})$, $(\tilde{\mathbf{E}})_3 = (\alpha^{231}, \alpha^{243})$ 。

(3) 因为 $s=3 > m=2$, 所以对 $(\tilde{\mathbf{R}}^n)$ 序列, 每隔 $\alpha=7$ 位采样, 得到采样序列后, 可得

$$(\tilde{\mathbf{E}}_7)_1 = (\alpha^{63}, \alpha^{249}, \alpha^{231}) \text{ 和 } (\tilde{\mathbf{E}}_7)_2 = (\alpha^{126}, \alpha^{159}, \alpha^{243})$$

(4) 用 BM 算法, 得到能生成 $(\tilde{\mathbf{E}}_7)_1$ 序列的 LFSR₁, 它的联接多项式

$$\sigma_1(x) = 1 + \alpha^{186}x + \alpha^{100}x^2 \quad l = 2$$

用 $\sigma_1(x)$ 于 $(\tilde{\mathbf{E}}_7)_2$, 证明它不能生成 $(\tilde{\mathbf{E}}_7)_2$, 所以

$$\sigma_2(x) = \sigma_1(x) + d_m d_i^{-1} x^{m-i} \sigma^{(i)}(x)$$

这里, 取 $d_i^{-1} = \alpha^{-163}$, $m-i=1$, $\sigma^{(i)}(x) = 1 + \alpha^{186}x$, 所以

$$\sigma_2(x) = 1 + (\alpha^{186} + \alpha^{92}d_m)x + (\alpha^{100} + \alpha^{23}d_m)x^2$$

用 $\sigma_2(x)$ 于 $(\tilde{\mathbf{E}}_7)_2$, 得到以下求 d_m 的方程:

$$\alpha^{226} + \alpha^{149}d_m + \alpha^{90} + \alpha^{251}d_m = \alpha^{243}$$

可得 $d_m = \alpha^{162}$, 最后求得

$$\sigma_2(x) = 1 + \alpha^{203}x + \alpha^{15}x^2$$

(5) 用 LFSR₂[$\sigma_2(x)$, $l_2=2$]产生整个 $(\tilde{\mathbf{E}}_7^n)$ 序列

$$(\tilde{\mathbf{E}}_7^n)^\infty = [0, \alpha^{203}, \alpha^{151}, \alpha^{252}, \alpha^{47}, \alpha^{63}, \alpha^{249}, \alpha^{231}, \alpha^{94}, \alpha^{229}, \alpha^{126}, \alpha^{159}, \alpha^{243}, \alpha^{242}, \alpha^{207}, \alpha^{121}, \alpha^{188}, \dots]$$

(6) 对 $(\tilde{\mathbf{E}}_7^n)^\infty$ 序列用 $\alpha^{-1}=7^{-1}=5 \pmod{17}$ 采样, 得到

$$(\tilde{\mathbf{E}}^n)^\infty = [0, \alpha^{63}, \alpha^{126}, \alpha^{121}, \alpha^{252}, \alpha^{94}, \alpha^{242}, \alpha^{203}, \alpha^{249}, \alpha^{159}, \alpha^{188}, \alpha^{47}, \alpha^{229}, \alpha^{207}, \alpha^{151}, \alpha^{231}, \alpha^{243}, \dots]$$

(7) 计算

$$\hat{\mathbf{C}} = \tilde{\mathbf{R}} - (\tilde{\mathbf{E}}^n) = (1000000000000000000)$$

对 $\hat{\mathbf{C}}$ 进行 DFT⁻¹ 得到

$$\hat{\mathbf{C}} = (1111111111111111111)$$

除了用频域方法进行超 BCH 限的译码外, 也可以用时域方法^[15], 并用解线性方程组 (7.5.5) 式, 直接求出 $\sigma(x)$ 的系数 $\sigma_1, \sigma_2, \dots, \sigma_t$ ^[16], 并且还可以译至码所具有的某些信息位上的更高的纠错能力^[18]。有关这些方法可参阅相应的文献。

§ 7.9 Goppa 码的一般描述

一、码的有理分式表示

正如可以用一个多项式 $C(x) = c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \dots + c_1x + c_0$ ($c_i \in GF(q)$) 表示一个 n 维线性空间中的一个矢量 $(c_{n-1}, c_{n-2}, \dots, c_1, c_0)$ 一样, 也可以用有理分式来表示该矢量。在多项式表示中, 多项式中的次数 x^{n-i} 表示矢量分量的位置是第 i 位(由左向右), 而它的系数 c_{n-i} 表示该分量的取值。

在有理分式表示中, 则用有理分式的参数表示矢量分量的位置和取值大小。如 n 维线

性空间中的一个矢量 $(c_{n-1}, c_{n-2}, \dots, c_1, c_0)$ ($c_i \in GF(q)$)，可以用以下有理分式表示：

$$C(z) = \sum_{i=0}^{n-1} \frac{c_i}{z - \alpha_i} \quad (7.9.1)$$

式中， c_i 是矢量的 $n-i$ 位分量的值， $\alpha_i \in GF(q^n)$ 表示矢量的 $n-i$ 位分量的位置。由此可见，一个 n 重矢量完全可用式(7.9.1)的有理分式确定；反之，一个如式(7.9.1)所示的有理分式也完全确定了一个矢量。

由第五章知，在码的多项式表示中，若 $C(x)$ 是一个由 $g(x)$ 生成的 $[n, k]$ 循环码的一个码字，则 $C(x) \equiv 0 \pmod{g(x)}$ 。同理，在码的有理分式表示中，若 $C(z)$ 是一个由 $g(z)$ 多项式生成的 C_g 码的一个码字，如果

$$C(z) = \sum_{i=0}^{n-1} \frac{c_i}{z - \alpha_i} \equiv 0 \pmod{g(z)} \quad (7.9.2)$$

则称 C_g 是一个有理分式码， $C(z)$ 是由 $g(z)$ 生成的有理分式码 C_g 的一个码字， $g(z)$ 称为 C_g 的生成多项式。

设 C_g 是一个有理分式码， $a(z)$ 与 $b(z)$ 是 C_g 的两个码字，则：

$$\begin{aligned} a(z) &= \sum_{i=0}^{n-1} \frac{a_i}{z - \alpha_i} \equiv 0 \pmod{g(z)} \\ b(z) &= \sum_{i=0}^{n-1} \frac{b_i}{z - \alpha_i} \equiv 0 \pmod{g(z)} \end{aligned}$$

显然

$$a(z) + b(z) = \sum_{i=0}^{n-1} \frac{a_i + b_i}{z - \alpha_i} = \sum_{i=0}^{n-1} \frac{c_i}{z - \alpha_i} \equiv 0 \pmod{g(z)}$$

式中， $c_i = a_i + b_i$ ， $i = 0, 1, 2, \dots, n-1$ 。所以 $a(z) + b(z)$ 也是 C_g 的一个码字，满足加法封闭性，同理可证 C_g 中有恒等元 0，加法逆元，且结合律成立。因而满足(7.9.2)式的有理分式码 C_g 是一个群码，当然必是一个线性码。

下面我们讨论码字的有理分式表示与多项式表示之间的关系。

首先分析下式：

$$\sum_{z=1}^4 \frac{1}{z} \equiv 0 \pmod{5} \quad (7.9.3)$$

的意义。由第二章知 $\{0, 1, 2, 3, 4\}$ 构成一个模 5 的剩余类环，由于 5 是素数，所以该环就成为有限域。式(7.9.3)左边的 $1/z = z^{-1}$ ，而 z^{-1} 就是 z 的逆元，为有限域中的一个元素。同理可知，式(7.9.3)左边的每一个有理分式都是有限域中的一个元素：如 $1/3 = 3^{-1} \equiv 2$ ， $1/4 = 4^{-1} \equiv 1 \pmod{5}$ 等。因而(7.9.3)式成为

$$\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} = 1 + 2^{-1} + 3^{-1} + 4^{-1} = 4 + 3 + 2 + 1 = 10 \equiv 0 \pmod{5}$$

同理，在(7.9.2)等式左边的每一项 $1/(z - \alpha_i)$ ，也是模 $g(z)$ 多项式环中的一个元素。若 $g(z)$ 是一个既约多项式，则由有限域理论可知，模 $g(z)$ 多项式环就是一个有限域。该有限域中的每一元素是一个次数 $< \deg g(z)$ 的多项式。因此

$$\frac{1}{z - \alpha_i} \equiv p_i(z) \pmod{g(z)}$$

而

$$\sum_{i=0}^{n-1} \frac{1}{z - \alpha_i} \equiv \sum_{i=0}^{n-1} p_i(z) \equiv 0 \pmod{g(z)}$$

表示等式右边之和的多项式是 $g(z)$ 的倍式。现在的问题是 $1/(z - \alpha_i) = (z - \alpha_i)^{-1}$ 。也就是 $(z - \alpha_i)$ 的逆元 $p_i(z)$ 是否存在。由于 $g(z)$ 是一个既约多项式， $(g(z), (z - \alpha_i)) = 1$ ，由欧几里德算法

$$p_i(z)(z - \alpha_i) + c(z)g(z) = 1$$

所以

$$p_i(z)(z - \alpha_i) \equiv 1 \pmod{g(z)}$$

因而 $p_i(z)$ 是 $(z - \alpha_i)$ 的逆元。因此只要 $g(z)$ 是一个域 $GF(q)$ 上的既约多项式，那么 $(z - \alpha_i)^{-1}$ 是有意义的，(7.9.2) 等式左边是一个多项式且是 $g(z)$ 的倍式。所以，用有理分式(7.9.2)式表示一个码字，与用多项式表示一个码字，本质上都相同，二者都用多项式表示，仅仅是表现形式不同而已。

二、Goppa 码的定义与描述

定义 7.9.1 设 $0 < n \leq q^m$ (q 为素数或素数幂， $m > 0$ 的整数)， $L = \{\alpha_1, \alpha_2, \dots, \alpha_n\}$ 是一个有序集合， $\alpha_i \in GF(q^m)$ ，且对任何 $i \neq j$ 恒有 $\alpha_i \neq \alpha_j$ ， $i, j \leq n$ 。又设 $GF(q)$ 上的 n 维线性空间为 V_n ，且矢量 $C = (c_1, c_2, \dots, c_n) \in V_n$ ，则与 C 对应的 $GF(q^m)$ 上的 z 的有理分式表示是

$$R_c(z) = \sum_{i=1}^n \frac{c_i}{z - \alpha_i}$$

又设 $g(z)$ 是系数在 $GF(q^m)$ 上的 z 多项式，它的根不在 L 中，则以下 n 重矢量集合

$$\{C | R_c(z) \equiv 0 \pmod{g(z)}, C \in V_n\} \quad (7.9.4)$$

或等价地说由 $g(z)$ 生成的多项式环中使 $R_c(z) \equiv 0 \pmod{g(z)}$ 的多项式集合 $\{R_c(z)\}$ ，称为由 $g(z)$ 生成的 Goppa 码，称 $g(z)$ 是码的生成多项式或 Goppa 多项式。若 $g(z)$ 在 $GF(q^m)$ 上既约，则称为既约 Goppa 码。

由此定义看到，Goppa 码是 n 维线性空间中，满足式(7.9.4)的所有 n 重矢量 C 的集合，它显然是一个线性子空间，因而 Goppa 码是一个线性码。

例 7.11 设 $L = \{\alpha_1, \alpha_2, \dots, \alpha_8\}$ ， $\alpha_i \in GF(2^3)$ ， $n = 2^3 = 8$ ，码的 Goppa 多项式 $g(z) = z^2 + z + 1$ ，已知八重矢量 $C = (11110001)$ ，则 C 的有理分式表示为

$$\begin{aligned} R_c(z) &= \frac{1}{z - \alpha_1} + \frac{1}{z - \alpha_2} + \frac{1}{z - \alpha_3} + \frac{1}{z - \alpha_4} + \frac{1}{z - \alpha_8} \\ &= \frac{\sum_j \prod_{i \neq j} (z - \alpha_i)}{(z - \alpha_1)(z - \alpha_2)(z - \alpha_3)(z - \alpha_4)(z - \alpha_8)} = \frac{f'(z)}{f(z)} \end{aligned}$$

式中， $f(z) = (z - \alpha_1)(z - \alpha_2)(z - \alpha_3)(z - \alpha_4)(z - \alpha_8)$ ， $f'(z)$ 是 $f(z)$ 的形式导数。由此可知，对任一 Goppa 码，若

$$R_c(z) = \frac{f'(z)}{f(z)} \equiv 0 \pmod{g(z)} \quad (7.9.5)$$

则 C 是由 $g(z)$ 生成的 Goppa 码的一个码字。要满足 $R_c(z) \equiv 0 \pmod{g(z)}$ ，只要检查 $f'(z)$

是否能被 $g(z)$ 除尽。若 $f'(z)$ 能被 $g(z)$ 除尽，则 \mathbf{C} 是一个码字；否则不是。 ■

由以上定义和例子可以看到，决定一个 Goppa 码的是两个参数：一个是有序集合 $L = \{\alpha_1, \alpha_2, \dots, \alpha_n\}$ ；另一个是 Goppa 多项式 $g(z)$ 。所以，又称 Goppa 码为 $\Gamma(L, g)$ 码。由于 L 决定了码元的位置，所以又称 L 为码的位置集合。

由式(7.9.5)

$$R_c(z) = \frac{f'(z)}{f(z)} \equiv 0 \pmod{g(z)}$$

看出， $f(z) = \prod_{\substack{i=1 \\ i \text{ 取值 } c_i \neq 0}}^n (z - \alpha_i)$ 的根含在 L 之中，但由定义 7.9.1 知 $g(z)$ 的根不在 L 中，因而

$(f(z), g(z)) = 1$ 这就是对 Goppa 多项式 $g(z)$ 的要求。

若 $g(z)$ 的根与 L 的根都在 $GF(q^n)$ 域中，那么码长 $n = q^n - \deg g(z)$ ，如果 $g(z)$ 的根不在 $GF(q^n)$ 中，则

$$\begin{aligned} \{L\} &= \{\alpha_1, \alpha_2, \dots, \alpha_{q^n}\} & \alpha_i \in GF(q^n) \\ n &= q^n \end{aligned}$$

一般情况下， $g(z)$ 的根不在位置集 L 所处的域中，而在该域的扩域中，此时能得到最长码长的码，它的码长 $n = q^n$ 。

三、Goppa 码的校验矩阵与最小距离

由 Goppa 码定义 7.9.1 可知：

$$R_c(z) = \sum_{i=0}^{n-1} \frac{c_i}{(z - \alpha_i)} \equiv 0 \pmod{g(z)} \quad (7.9.6)$$

该式可写成

$$\left[\frac{1}{z - \alpha_0} \quad \frac{1}{z - \alpha_1} \quad \cdots \quad \frac{1}{z - \alpha_{n-1}} \right] \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-1} \end{bmatrix} = \mathbf{H}_1 \mathbf{C}^T \equiv 0 \pmod{g(z)}$$

称

$$\mathbf{H}_1 = \left[\frac{1}{z - \alpha_0} \quad \frac{1}{z - \alpha_1} \quad \cdots \quad \frac{1}{z - \alpha_{n-1}} \right] \quad (7.9.7)$$

为 Goppa 码的校验矩阵。我们可把 \mathbf{H}_1 化成其它形式，为此把 \mathbf{H}_1 矩阵中的每一个元素进行化简。

因为

$$\frac{g(z)}{z - \alpha_i} \equiv 0 \pmod{g(z)}$$

所以

$$\frac{-1}{z - \alpha_i} \equiv \frac{g(z) - g(\alpha_i)}{z - \alpha_i} g^{-1}(\alpha_i) \pmod{g(z)}$$

把 \mathbf{H}_1 中的每个元素都用上式代入得

$$\mathbf{H}_2 = - \left[\frac{g(z) - g(\alpha_0)}{z - \alpha_0} g^{-1}(\alpha_0), \frac{g(z) - g(\alpha_1)}{z - \alpha_1} g^{-1}(\alpha_1), \dots, \frac{g(z) - g(\alpha_{n-1})}{z - \alpha_{n-1}} g^{-1}(\alpha_{n-1}) \right] \quad (7.9.8)$$

上式中每一元素的负号都在矩阵外面，由于负号并不影响式(7.9.8)两边的关系，所以今后可把此负号去掉。

若 $C(z) = c_{n-1}z^{n-1} + c_{n-2}z^{n-2} + \dots + c_0$ 是由 $g(z)$ 生成的 Goppa 码的一个码字，则由 $\mathbf{H}_2 \cdot \mathbf{C}^T = 0$ 可知

$$\sum_{i=0}^{n-1} c_i \frac{g(z) - g(\alpha_i)}{z - \alpha_i} g^{-1}(\alpha_i) = 0$$

对上式中的 $(g(z) - g(\alpha_i))g(\alpha_i)^{-1}/(z - \alpha_i)$ 进行化简后，可把 \mathbf{H}_2 矩阵写成以下形式：

$$\begin{aligned} \mathbf{H}_2 &= \begin{bmatrix} g_r & 0 & \cdots & 0 \\ g_{r-1} & g_r & \cdots & 0 \\ g_{r-2} & g_{r-1} & g_r & 0 \\ \vdots & \vdots & \vdots & \vdots \\ g_1 & g_2 & \cdots & g_r \end{bmatrix} \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha_0 & \alpha_1 & \cdots & \alpha_{n-1} \\ \alpha_0^2 & \alpha_1^2 & \cdots & \alpha_{n-1}^2 \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_0^{-1} & \alpha_1^{-1} & \cdots & \alpha_{n-1}^{-1} \end{bmatrix} \\ &\quad \cdot \begin{bmatrix} g^{-1}(\alpha_0) & & & \mathbf{0} \\ & g^{-1}(\alpha_1) & & \\ & & \mathbf{0} & \\ & & & g^{-1}(\alpha_{n-1}) \end{bmatrix} \\ &= A\alpha B \end{aligned} \quad (7.9.9)$$

该式中的 A 矩阵最后可化为只有主对角线元素，而其它元素均为 0 的矩阵，因此它的存在与否不会影响码的纠错能力。所以式(7.9.9)的 \mathbf{H}_2 矩阵最后可化简为

$$\begin{aligned} \mathbf{H}_3 &= \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha_0 & \alpha_1 & \cdots & \alpha_{n-1} \\ \alpha_0^2 & \alpha_1^2 & \cdots & \alpha_{n-1}^2 \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_0^{-1} & \alpha_1^{-1} & \cdots & \alpha_{n-1}^{-1} \end{bmatrix} \begin{bmatrix} g^{-1}(\alpha_0) & & & \\ & g^{-1}(\alpha_1) & & \mathbf{0} \\ & & \mathbf{0} & \\ & & & g^{-1}(\alpha_{n-1}) \end{bmatrix} = \alpha B \\ &= \begin{bmatrix} g^{-1}(\alpha_0) & g^{-1}(\alpha_1) & \cdots & g^{-1}(\alpha_{n-1}) \\ \alpha_0 g^{-1}(\alpha_0) & \alpha_1 g^{-1}(\alpha_1) & \cdots & \alpha_{n-1} g^{-1}(\alpha_{n-1}) \\ \vdots & \vdots & & \vdots \\ \alpha_0^{-1} g^{-1}(\alpha_0) & \alpha_1^{-1} g^{-1}(\alpha_1) & \cdots & \alpha_{n-1}^{-1} g^{-1}(\alpha_{n-1}) \end{bmatrix} \end{aligned} \quad (7.9.10)$$

显然在 α 矩阵中任意取出 r 列所组成的矩阵是一个范德蒙矩阵，其秩为 r 。因此，只要 $\alpha_1, \alpha_2, \dots, \alpha_n$ 不相等且不为 0，则 α 矩阵的秩为 r ，而 B 矩阵显然是一个满秩矩阵。由此可知， \mathbf{H}_3 矩阵的秩为 r ，即 \mathbf{H}_3 矩阵中各行线性无关，且任意 r 列线性无关。由定理 3.3.1 知满足 \mathbf{H}_3 矩阵的 Goppa 码有最小距离为

$$d = r + 1 = \partial \cdot g(z) + 1 \quad (7.9.11)$$

如果 \mathbf{H}_3 中的每个元素都用基域 $GF(q)$ 中的元素表示，则 \mathbf{H}_3 矩阵至多有 $m\partial \cdot g(z)$ 行，且至少 r 列线性无关。综上所述可得如下定理。

定理 7.9.1 由 r 次多项式 $g(z)$ 生成的 q 进制 Goppa 码，至多有 $m\partial \cdot g(z)$ 个校验位，有最小距离 $d \geq \partial \cdot g(z) + 1$ ，其位置域 L 中的元素为 $\alpha_i \in GF(q^m)$, $i = 0, 1, 2, \dots, n-1$ 。如果 $g(z)$ 的根不在 $GF(q^m)$ 中，则生成一个 $[q^m, k \geq q^m - rm, d \geq r+1]$ 的既约 Goppa 码。

把 $\text{GRS}_{n-r}(\alpha, \gamma)$ 的校验矩阵(7.3.19)式与 Goppa 码的校验矩阵 H_3 进行比较可知, 若广义 RS 码的 $\gamma_i = g^{-1}(\alpha_i)$, 则 $\text{GF}(q)$ 上的 $\text{GRS}_{n-r}(\alpha, \gamma)$ 码就是 Goppa 码, 即 Goppa 码是 $\text{GRS}_r(\alpha, \gamma)$ 码的对偶码的子域子码。

定理 7.9.2 给定码率 $R > 0$ 和任意小 $\epsilon > 0$, 则码长 $n \rightarrow \infty$ 时, 一定可以找到 $\text{GF}(q)$ 上的既约 Goppa 多项式 $g(z)$, 由它生成的 Goppa 码满足以下关系:

$$\frac{d}{n} > \varphi^{-1}(1 - R) - \epsilon \quad (7.9.12)$$

这里

$$\varphi(x) = x \log_q(q-1) - x \log_q x - (1-x) \log_q(1-x)$$

由此定理可知, 当 $n \rightarrow \infty$ 时, 一定存在有满足 $d/n > 0$, $R > 0$ 的既约 Goppa 码, 也就是能达到 V-G 限的码。这说明在 Goppa 码中一定存在有能达到香农信道编码定理所指出的 Shannon 码, 但是如何选择 $g(z)$, 使码能达到这种性能仍没有解决。

四、二进制 Goppa 码

由 Goppa 码的定义和(7.9.6)式可知:

$$\begin{aligned} R_e(z) &= \sum_{i=0}^{n-1} \frac{c_i}{z - \alpha_i} \equiv 0 \pmod{g(z)} \\ &= \frac{c_0(z - \alpha_1)(z - \alpha_2) \cdots (z - \alpha_{n-1}) + \cdots + (z - \alpha_0)(z - \alpha_1) \cdots (z - \alpha_{n-2})c_{n-1}}{(z - \alpha_0)(z - \alpha_1) \cdots (z - \alpha_{n-1})} \\ &= \frac{f_1(z)}{f(z)} = \epsilon(z) \end{aligned} \quad (7.9.13)$$

式中, $f_1(z) = f'(z)$ 是 $f(z)$ 的形式导数, 在二进制情况下它是一个偶函数 $R_e(z)$ 。如

$$f(z) = f_0 + f_1 z + f_2 z^2 + f_3 z^3 + f_4 z^4 + f_5 z^5 + \cdots$$

则

$$f'(z) = f_1 + 2f_2 z + 3f_3 z^2 + 4f_4 z^3 + 5f_5 z^4 + \cdots \equiv f_1 + f_3 z^2 + f_5 z^4 + \cdots \pmod{2}$$

因此若 $g(z) | R_e(z)$, 则等效于 $g(z) | f'(z)$ 。

设 $\bar{g}(z)$ 是除尽 $g(z)$ 的次数最低的完全偶多项式, 即 $\bar{g}(z) | g(z)$ 。由上式可知 $\bar{g}(z) | f'(z)$, 因而(7.9.12)式

$$\epsilon(z) = \frac{f'(z)}{f(z)} \equiv 0 \pmod{g(z)}$$

可写成

$$\epsilon(z) = \frac{f'(z)}{f(z)} \equiv 0 \pmod{\bar{g}(z)}$$

在特征为 2 的域 $\text{GF}(2^n)$ 上的偶多项式 $f'(z) = [F(x)]^2$, 因而若 $g(z)$ 无重根, 则 $\bar{g}(z) = g^2(z)$, 由此可知, 二进制 Goppa 码的最小距离

$$d \geqslant 2 \cdot \bar{g}(z) + 1 = 2\theta \cdot g(z) + 1$$

有信息位 $k = n - \theta \cdot g(z)m$ 。

例 7.12 $g(z) = z^2 + z + 1$, $L = \text{GF}(2^3) = \{0, 1, \alpha, \dots, \alpha^6\}$, $\alpha \in \text{GF}(2^3)$ 是本原域元素, α 的本原多项式是 $x^3 + x + 1$ 。 $g(\alpha^i) \neq 0$, $i = 0, 1, \dots, 6$, $g(0) \neq 0$ 。因而 $g(z)$ 在 $\text{GF}(2^3)$ 上既约, 所以 $g(z)$ 的根必在 $\text{GF}(2^2)$ 、 $\text{GF}(2^4)$ 、 $\text{GF}(2^6)$ 等域中。由此得到一个既约 Goppa 码, 有

码长 $n=2^3=8$, $k \geq 8-3 \cdot 2=2$, 有最小距离 $d \geq 5$ 。由(7.9.10)式可知该码的校验矩阵

$$\begin{aligned} \mathbf{H} &= \begin{bmatrix} g^{-1}(0) & g^{-1}(1) & g^{-1}(\alpha) & \cdots & g^{-1}(\alpha^6) \\ 0g^{-1}(0) & 1g^{-1}(1) & \alpha g^{-1}(\alpha) & \cdots & \alpha^6 g^{-1}(\alpha^6) \end{bmatrix} \\ &= \begin{bmatrix} 1 & 1 & \alpha^2 & \alpha^4 & \alpha^2 & \alpha & \alpha & \alpha^4 \\ 0 & 1 & \alpha^3 & \alpha^6 & \alpha^5 & \alpha^5 & \alpha^6 & \alpha^3 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} \end{aligned}$$

由此得 4 个码字为:

$$\begin{array}{ll} \text{码元位置: } & 0 \ 1 \ \alpha \ \alpha^2 \ \alpha^3 \ \alpha^4 \ \alpha^5 \ \alpha^6 \\ \text{码字: } & \begin{array}{ccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 \end{array} \end{array}$$

得到一个[8, 2, 5]二进制既约 Goppa 码。从上面 4 个码字中任意取出两个非 0 码字, 进行初等变换后即可得到码的标准生成矩阵和系统码形式的码字集合。 ■

§ 7.10 Goppa 码的扩展及其它特殊子类

这一节主要讨论扩展 Goppa 码与循环码的关系, 并介绍几类特殊的有理分式码, 以及它们与 BCH 码的关系。

一、扩展 Goppa 码

由前可以看出, Goppa 码虽是线性码, 但并不是循环码。可是它与循环码有密切关系。

若我们把例 7.12 中的[8, 2, 5]Goppa 码的每个码字中附加一位全校验位, 并适当地交换列的位置, 则 4 个码字成为:

$$\begin{array}{ll} \text{码元位置: } & 1 \ \alpha^4 \ \alpha^6 \ \infty \ \alpha^2 \ \alpha^5 \ 0 \ \alpha \ \alpha^3 \\ \text{码字: } & \begin{array}{cccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \end{array} \end{array}$$

得到一个[9, 2, 6]码, 可以检验它是一个循环码。码元位置中的 ∞ 就是附加的全校验位。称[9, 2, 6]码是[8, 2, 5]码的扩展 Goppa 码。

设 $\Gamma(L, g)$ 是一个由 $GF(q^m)$ 上的 r 次既约多项式 $g(z)$ 生成的 q 进制 Goppa 码, 有码长 $n=q^m$, $L=GF(q^m)=\{0, 1, \alpha, \dots, \alpha^{r-1}\}$ 。由式(7.9.10)知, 若 $a=(a(0), a(1), \dots, a(\alpha^{r-1}))$ 是 $\Gamma(L, g)$ 码的一个码字, 则

$$\sum_{\beta \in GF(q^m)} \frac{\beta a(\beta)}{g(\beta)} = 0 \quad i = 0, 1, 2, \dots, r-1 \quad (7.10.1)$$

若对此码附加一个全校验位 $a(\infty)$, 且

$$a(\infty) = - \sum_{\beta \in GF(q^m)} a(\beta)$$

或

$$\sum_{\beta \in GF(q^m) \cup \{\infty\}} a(\beta) = 0 \quad (7.10.2)$$

由此得到码字 $\hat{a} = (a(0), a(1), \dots, a(\alpha^{r-1}), a(\infty))$ 。由所有扩展码字组成的集合, 称为扩展 Goppa 码, 用 $\hat{\Gamma}(L, g)$ 表示。由式(7.9.10)可知, $\hat{\Gamma}(L, g)$ 码的校验矩阵

$$H = \begin{bmatrix} g^{-1}(0) & g^{-1}(1) & g^{-1}(\alpha) & \cdots & g^{-1}(\alpha^{r-1}) & 0 \\ 0 & g^{-1}(1) & \alpha g^{-1}(\alpha) & \cdots & \alpha^{r-1} g^{-1}(\alpha^{r-1}) & 0 \\ 0 & g^{-1}(1) & \alpha^2 g^{-1}(\alpha) & \cdots & (\alpha^{r-1})^2 g^{-1}(\alpha^{r-1}) & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & g^{-1}(1) & \alpha^{r-1} g^{-1}(\alpha) & \cdots & (\alpha^{r-1})^{r-1} g^{-1}(\alpha^{r-1}) & 0 \\ 1 & 1 & 1 & \cdots & 1 & 1 \end{bmatrix} \quad (7.10.3)$$

由此可知, 若 \hat{a} 是扩展 Goppa 码的一个码字, 则它必须满足 $\hat{a} \cdot H^T = H \cdot \hat{a}^T = \mathbf{0}$ 或

$$\sum_{\beta \in GF(q^m) \cup \{\infty\}} \frac{\beta a(\beta)}{g(\beta)} = 0 \quad i = 0, 1, 2, \dots, r \quad (7.10.4)$$

下面定理说明了扩展 Goppa 码是一个循环码时, 它的 Goppa 多项式 $g(z)$ 所必须满足的充要条件。

定理 7.10.1 若 $L = GF(q^m)$, $n = q^m$ 。则由 $g(z) = [m(z)]^a$ (a 是大于 0 的整数, $m(z) \in GF(q^m)[z]$) 生成的 $\Gamma(L, g)$ Goppa 码的扩张码 $\hat{\Gamma}(L, g)$ 码, 为循环码的充要条件是 $m(z)$ 为二次多项式。

该定理的证明比较复杂, 有兴趣的读者可参阅有关文献^[18], 下面举例说明之。

例 7.13 $L = GF(2^5)$, $g(z) = z^2 + z + 1$ 。由此 $g(z)$ 生成一个 $[32, 22, 5]$ 二进制既约 Goppa 码, 对该码加一个全校验位后, 得到一个 $[33, 22, 6]$ 扩展既约 Goppa 码, 可以验证它就是由 $g(x) = x^{11} + x^9 + x^6 + x^5 + x^2 + 1$ 生成的 $[33, 22, 6]$ 循环码。 ■

二、累积码

定义 7.10.1 Goppa 多项式 $g(z)$ 只有一个根的 Goppa 码称为累积码。

累积码有如下参数: $g(z) = (z - \beta)^r$, $L = \{GF(q^m) - \{\beta\}\}$, $n \leq q^m - 1$, $k = n - rm$, $d = \partial \circ g(z) + 1$, $n = q^m - 1$ 是累积码的最大可能的码长。由此可知, 累积码的生成多项式 $g(z)$ 的根与位置集 L 都在 $GF(q^m)$ 域中。

若 $\beta = 0$, 则 $g(z) = z^r$ 。 $g(z) = z^r$ 以 0 元素为根, 当 $n = q^m - 1$ 时生成的码是一类较特殊的累积码。这类累积码的校验矩阵由(7.9.10)式可知为

$$H = \begin{bmatrix} g^{-1}(\alpha_1) & g^{-1}(\alpha_2) & \cdots & g^{-1}(\alpha_n) \\ \alpha_1 g^{-1}(\alpha_1) & \alpha_2 g^{-1}(\alpha_2) & \cdots & \alpha_n g^{-1}(\alpha_n) \\ \vdots & \vdots & & \vdots \\ \alpha_1^{r-1} g^{-1}(\alpha_1) & \alpha_2^{r-1} g^{-1}(\alpha_2) & \cdots & \alpha_n^{r-1} g^{-1}(\alpha_n) \end{bmatrix}$$

$$= \begin{bmatrix} \alpha_1^{-r} & \alpha_2^{-r} & \cdots & \alpha_n^{-r} \\ \alpha_1^{-r+1} & \alpha_2^{-r+1} & \cdots & \alpha_n^{-r+1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{-1} & \alpha_2^{-1} & \cdots & \alpha_n^{-1} \end{bmatrix} \quad (7.10.5)$$

所以该累积码以 $\alpha^{-r}, \alpha^{-r+1}, \dots, \alpha^{-1}$ 等 r 个连续元素为根，与 BCH 码的校验矩阵式(7.1.3)比较可知，若 $\alpha^{m_i} = \alpha_i^{-r}$, $\alpha_r^{-n} = (\alpha^r)^n = 1$ ，则式(7.1.3)与式(7.10.5)相同。式(7.10.5) H 矩阵的行列式为

$$(\alpha_1^{-r} \alpha_2^{-r} \cdots \alpha_n^{-r}) \begin{vmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{-1} & \alpha_2^{-1} & \cdots & \alpha_n^{-1} \end{vmatrix}$$

所以该码的距离 $d=r+1$ ，这与 BCH 限决定的完全相同。由此可知，以 0 为根的、 $n=q^m-1$ 的累积码就是本原 BCH 码。

定理 7.10.2 以 $L=\{1, \alpha, \alpha^2, \dots, \alpha^{r-1}\}$, $\alpha \in GF(q^m)$ 是本原元素, $n=q^m-1$, $g(z)=z^r$ 生成的 Goppa 码，是累积码类中一的循环码。

证明 $g(z)=z^r$ 生成的 Goppa 码是 BCH 码，显然是循环码，现证唯一性。如果 $g'(z)=(z-\beta)^r$ ($\beta \neq 0$) 生成的 Goppa 码也是循环码，也就是若 $C(z)$ 是一个码字的话， $\alpha C(z)$ 也应是一个码字。即若

$$\epsilon_\alpha(z) = \sum_{i=1}^n \frac{c_i}{z - \alpha_i} = \frac{f'(z)}{f(z)} \equiv 0 \pmod{g'(z) = (z - \beta)^r}$$

成立，则下式也应成立：

$$\epsilon_{\alpha\beta}(z\alpha) = \sum_{i=1}^n \frac{\alpha c_i}{z - \alpha_i} = \frac{\varphi(z)}{f(z)} \equiv 0 \pmod{g'(z) = (z - \beta)^r}$$

这说明该码除了有 $g'(z)$ 的所有根之外，还有 $\alpha\beta$ 形式的全部根，即 $\partial \circ \varphi(z) \geq n$ ，而这是不可能的。所以 $g'(z)=(z-\beta)^r$, ($\beta \neq 0$) 生成的 Goppa 码不是循环码。■

下面定理说明所有累积码，不论 $g(z)$ 是以 $\beta=0$ 还是以 $\beta \neq 0$ 为根，它们都具有相同的性质。因此要研究累积码的性质，只要研究用 $g(z)=z^r$ 产生的码即可。

定理 7.10.3 生成多项式的次数均等于 r 的累积码有相同的重量分布。

证明 累积码的 (L, g) 是 $(\{GF(q^m) - \{\beta\}\}, (z-\beta)^r)$ 。设 $\epsilon_c(z)$ 是该码的一个码字，则

$$\epsilon_c(z) = \frac{\varphi(z)}{f(z)} \equiv 0 \pmod{(z - \beta)^r}$$

进行线性变量代换 $z=z+\beta$ ，则上式成为

$$\epsilon_c(z + \beta) = \frac{\varphi(z + \beta)}{f(z + \beta)} \equiv 0 \pmod{(z)^r}$$

所以 $\Gamma(\{GF(q^m) - \{\beta\}\}, (z-\beta)^r)$ 码的码字 $\epsilon_c(z)$ 与 $\Gamma(\{GF(q^m) - \{\beta\}\}, z^r)$ 码的码字 $\epsilon_c(z + \beta)$ 完全等价。■

所以对累积码的研究可以归结为由 $g(z)=z^r$ 产生的码的研究，也就是归结为本原 BCH 码的研究，因而有相同的重量分布。

虽然 $g(z)=z^r$ 生成的累积码就是本原 BCH 码，用 Goppa 码估计的码的最小距离 $d=\partial \circ g(z)+1=r+1$ ，好像与 BCH 限的估计相同，但事实上，对很多累积码而言，用 BCH 限的估计比用 Goppa 码限的估计要好得多，例如 [63, 15] 和 [63, 12] 码，用 Goppa 码的方法

估计 d 分别是 17 和 19, 但用 BCH 码限估计分别是 24 和 28。

三、可分码

定义 7.10.2 Goppa 多项式 $g(z)$ 如果没有重根, 则由此 $g(z)$ 生成的 Goppa 码称为可分码。

由此可知, 可分码的生成多项式有如下形式:

$$g(z) = (z - \beta_1)(z - \beta_2) \cdots (z - \beta_r) \quad (7.10.6)$$

定理 7.10.4 由式(7.10.6)的 $g(z)$ 生成的 Goppa 码, 其校验矩阵有如下形式:

$$\mathbf{H} = \left[\begin{array}{cccc} \frac{1}{z - \alpha_1} & \frac{1}{z - \alpha_2} & \cdots & \frac{1}{z - \alpha_n} \end{array} \right] \quad (7.10.7)$$

或

$$\mathbf{H}_{S_a} = \left[\begin{array}{cccc} \frac{1}{\beta_1 - \alpha_1} & \frac{1}{\beta_1 - \alpha_2} & \cdots & \frac{1}{\beta_1 - \alpha_n} \\ \vdots & \vdots & & \vdots \\ \frac{1}{\beta_r - \alpha_1} & \frac{1}{\beta_r - \alpha_2} & \cdots & \frac{1}{\beta_r - \alpha_n} \end{array} \right] \quad (7.10.8)$$

式中, $\alpha \in GF(q^m)$, $L = \{\alpha_1, \alpha_2, \alpha_3, \dots, \alpha_n\}$ 。

证明 由 Goppa 码定义可知, 若 $C = (c_n, c_{n-1}, \dots, c_1)$ 是由 $g(z) = (z - \beta_1)(z - \beta_2) \cdots (z - \beta_r)$ 生成的 Goppa 码的一个码字, 则

$$\epsilon_c(z) = \sum_{i=1}^n \frac{c_i}{z - \alpha_i} = \frac{\varphi(z)}{f(z)} \equiv 0 \pmod{g(z)}$$

写成矩阵形式为

$$\left[\begin{array}{cccc} \frac{1}{z - \alpha_1} & \frac{1}{z - \alpha_2} & \cdots & \frac{1}{z - \alpha_n} \end{array} \right] \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix} = 0 \quad (7.10.9)$$

由此得知码的校验矩阵是式(7.10.7)。

若 C_g 是由 $g(z)$ 生成的一个码, 则由 Goppa 码的定义可知, $g(z) | \varphi(z)$ 。所以若 $g(z)$ 以 $\beta_1, \beta_2, \dots, \beta_r$ 为根, 则 $\varphi(z)$ 和 $\epsilon_c(z)$ 也必以 $\beta_1, \beta_2, \dots, \beta_r$ 为根。因而由式(7.10.9)可得

$$\left[\begin{array}{cccc} \frac{1}{\beta_1 - \alpha_1} & \frac{1}{\beta_1 - \alpha_2} & \cdots & \frac{1}{\beta_1 - \alpha_n} \\ \frac{1}{\beta_2 - \alpha_1} & \frac{1}{\beta_2 - \alpha_2} & \cdots & \frac{1}{\beta_2 - \alpha_n} \\ \vdots & \vdots & & \vdots \\ \frac{1}{\beta_r - \alpha_1} & \frac{1}{\beta_r - \alpha_2} & \cdots & \frac{1}{\beta_r - \alpha_n} \end{array} \right] \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix} = 0$$

由此可知该码的校验矩阵为式(7.10.8)。 ■

若 $g(z)$ 在 $GF(q^m)$ 上既约, 则这类可分码是既约 Goppa 码。对这类既约的可分码有如下参数: $n = q^m, k \geq q^m - mr, d \geq \partial \cdot g(z) + 1$ 。若 $q = 2$, 则 $n = 2^m, k \geq 2^m - mr, d \geq 2\partial \cdot g(z) + 1$ 。

例 7.14 例 7.11 中 $(8, 2, 5)$ 码是一个可分码。该码的生成多项式 $g(z) = z^2 + z + 1 = (z - \alpha^{21})(z - \alpha^{42})$, $\alpha \in GF(2^6)$ 是域中的生成元。该码的校验矩阵

$$\mathbf{H} = \begin{bmatrix} \frac{1}{\alpha^{21} - \alpha_1} & \frac{1}{\alpha^{21} - \alpha_2} & \cdots & \frac{1}{\alpha^{21} - \alpha_8} \\ \frac{1}{\alpha^{42} - \alpha_1} & \frac{1}{\alpha^{42} - \alpha_2} & \cdots & \frac{1}{\alpha^{42} - \alpha_8} \end{bmatrix}$$

式中, $\alpha_i \in GF(2^3)$, $L = \{GF(2^3)\}$ 。由此 \mathbf{H} 化简后所得的表示式与例 7.12 中的表示式相同。 ■

四、Srivastava 码(S 码)

1966 年斯利华斯特华(Srivastava)首先利用有理分式表示码，并构造出一类 Srivastava 码(简称 S 码)，它与 Goppa 码有密切关系。

定义 7.10.3 有下列校验矩阵：

$$\mathbf{H}_s = \begin{bmatrix} \frac{z_1}{\alpha_1 - w_1} & \frac{z_2}{\alpha_2 - w_1} & \cdots & \frac{z_n}{\alpha_n - w_1} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{z_1}{\alpha_1 - w_r} & \frac{z_2}{\alpha_2 - w_r} & \cdots & \frac{z_n}{\alpha_n - w_r} \end{bmatrix} \quad (7.10.10)$$

的 q 进制码，称为 Srivastava 码。式中 α_i, w_i 是 $GF(q^m)$ 中的 $n+r$ 个不同的元素，而 z_1, z_2, \dots, z_n 是 $GF(q^m)$ 中的非 0 元素，该码有如下参数： $n \leq q^m - r$, $k \geq n - mr$, $d = r+1$ 。

在式(7.10.10)中，若令 $z_i = \alpha_i^l$ ，则成为

$$\mathbf{H}'_s = \begin{bmatrix} \frac{\alpha_1^{l-1}}{1 - \alpha_1^{-1}w_1} & \frac{\alpha_2^{l-1}}{1 - \alpha_2^{-1}w_1} & \cdots & \frac{\alpha_n^{l-1}}{1 - \alpha_n^{-1}w_1} \\ \vdots & \vdots & & \vdots \\ \frac{\alpha_1^{l-1}}{1 - \alpha_1^{-1}w_r} & \frac{\alpha_2^{l-1}}{1 - \alpha_2^{-1}w_r} & \cdots & \frac{\alpha_n^{l-1}}{1 - \alpha_n^{-1}w_r} \end{bmatrix} \quad (7.10.11)$$

这就是原始定义的 S 码的校验矩阵。

在 \mathbf{H}_s 中若令 $z_i = 1$ ，或在 \mathbf{H}'_s 中取 $l=2$ ，则就成为可分码的校验矩阵 \mathbf{H}_{sa} 。因此， $z_i = 1$ ($i=1, 2, \dots, n$) 或 $l=2$ 的 S 码就是 Goppa 码中的可分码。

1972 年海盖特(Helgert)把 S 码进行推广，得到一类广义 q 进制 Srivastava 码(简称 GS 码)，它的校验矩阵

$$\mathbf{H}_{GS} = \begin{bmatrix} \mathbf{H}_{s1} \\ \mathbf{H}_{s2} \\ \vdots \\ \mathbf{H}_{ss} \end{bmatrix} \quad (7.10.12)$$

式中

$$\mathbf{H}_{si} = \begin{bmatrix} \frac{z_1}{(\alpha_1 - w_i)^t} & \cdots & \frac{z_n}{(\alpha_n - w_i)^t} \\ \vdots & & \vdots \\ \frac{z_1}{(\alpha_1 - w_i)^t} & \cdots & \frac{z_n}{(\alpha_n - w_i)^t} \end{bmatrix} \quad i = 1, 2, \dots, s \quad (7.10.13)$$

式中， $\alpha_1, \alpha_2, \dots, \alpha_n, w_1, w_2, \dots, w_s$ 是 $GF(q^m)$ 中的 $n+s$ 个不同元素， z_1, z_2, \dots, z_n 是

$\text{GF}(q^m)$ 中的非 0 元素。

GS 码是有以下参数的线性码: $n \leq q^m - s$, $k \geq n - mst$, $d = st + 1$ 。显然, $t=1$, $s=r$ 的 GS 码就是 S 码。

例 7.15 构造有如下参数的二进制 GS 码: $n=8$, $m=6$, $r=2$, $s=1$, $t=2$ 。设 $\alpha \in \text{GF}(2^6)$ 是本原域元素, $\alpha_1, \alpha_2, \dots, \alpha_8 \in \text{GF}(2^3) \subset \text{GF}(2^6)$, 所以 $\alpha_1, \alpha_2, \dots, \alpha_8$ 是 $\text{GF}(2^6)$ 中的 $0, 1, \alpha^9, \alpha^{18}, \alpha^{27}, \alpha^{36}, \alpha^{45}, \alpha^{54}$ 元素, 其中 α^9 是 $\text{GF}(2^3)$ 中的本原元素, α 的本原多项式是 $p(x)=1+x+x^6$ 。

由式(7.10.12)、式(7.10.13)可知, 该 GS 码的校验矩阵

$$\mathbf{H}_{\text{GS}} = \begin{bmatrix} \frac{z_1}{\alpha_1 - w_1} & \frac{z_2}{\alpha_2 - w_1} & \cdots & \frac{z_8}{\alpha_8 - w_1} \\ \frac{z_1}{(\alpha_1 - w_1)^2} & \frac{z_2}{(\alpha_2 - w_1)^2} & \cdots & \frac{z_8}{(\alpha_8 - w_1)^2} \end{bmatrix}$$

令 $z_i=1$, $i=1, 2, \dots, 8$, $w_1=\alpha$, 则上式成为

$\mathbf{H}_{\text{GS}} =$

$$\begin{bmatrix} \frac{1}{0-\alpha} & \frac{1}{1-\alpha} & \frac{1}{\alpha^9-\alpha} & \frac{1}{\alpha^{18}-\alpha} & \frac{1}{\alpha^{27}-\alpha} & \frac{1}{\alpha^{36}-\alpha} & \frac{1}{\alpha^{45}-\alpha} & \frac{1}{\alpha^{54}-\alpha} \\ \frac{1}{(0-\alpha)^2} & \frac{1}{(1-\alpha)^2} & \frac{1}{(\alpha^9-\alpha)^2} & \frac{1}{(\alpha^{18}-\alpha)^2} & \frac{1}{(\alpha^{27}-\alpha)^2} & \frac{1}{(\alpha^{36}-\alpha)^2} & \frac{1}{(\alpha^{45}-\alpha)^2} & \frac{1}{(\alpha^{54}-\alpha)^2} \end{bmatrix}$$

上式矩阵中的第二行元素是第一行对应元素的平方, 由定理 5.2.1 可知, 它们线性相关, 因而矩阵中的第二行多余, 可以去掉, 由此

$$\begin{aligned} \mathbf{H}_{\text{GS}} &= \begin{bmatrix} \frac{1}{0-\alpha} & \frac{1}{1-\alpha} & \frac{1}{\alpha^9-\alpha} & \frac{1}{\alpha^{18}-\alpha} & \frac{1}{\alpha^{27}-\alpha} & \frac{1}{\alpha^{36}-\alpha} & \frac{1}{\alpha^{45}-\alpha} & \frac{1}{\alpha^{54}-\alpha} \end{bmatrix} \\ &= [a^{62} \ a^{57} \ a^{14} \ a^{15} \ a^{55} \ a^{49} \ a^{25} \ a^{11}] \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \end{bmatrix} \end{aligned}$$

所以码有 6 个校验位, 得到一个 $[8, 2, 5]$ GS 码, 它的 4 个码字是: (00000000) , (11110001) , (01011110) , (10101111) 。这与例 7.11 中的 $[8, 2, 5]$ 码完全等价。 ■

该例中的 $\alpha_1, \alpha_2, \dots, \alpha_8$ 是 $\text{GF}(2^6)$ 中的一个循环子群, 它们恰好组成一个子域 $\text{GF}(2^3)$ 。从原则上讲, 这 8 个元素可以在 $\text{GF}(2^6)$ 中任意挑选, 但不一定所挑选的元素组成的 \mathbf{H}_{GS} , 所得到的码有最大的最小距离, 只能保证由 GS 限给出的最小距离 $d \geq 3$ 。但事实上按我们例中的方法挑选, 可得到 $d=5$ 。位置域中的元素 $\alpha_1, \alpha_2, \dots, \alpha_n$ 应如何挑选, 才能保证得到的 GS 码有最大的最小距离, 这个问题仍没有完全解决。

§ 7.11 交替码(Alternant 码)和 GBCH 码

交替码简称 AT 码, 最先由海盖特于 1972 年提出。这是一类非常广泛的线性码。它包

括前节所提到的各类码。

一、交替码

定义 7.11.1 由下列校验矩阵

$$\mathbf{H}_{AT} = \begin{bmatrix} y_1g_1(x_1) & y_2g_1(x_2) & \cdots & y_ng_1(x_n) \\ \vdots & \vdots & & \vdots \\ y_1g_r(x_1) & y_2g_r(x_2) & \cdots & y_ng_r(x_n) \end{bmatrix} \quad (7.11.1)$$

所决定的码称为 q 进制交替码，记以 $A_q(\mathbf{x}, \mathbf{y})$ 。式中

$$g_i(x) = c_{i1} + c_{i2}x + \cdots + c_{ir}x^{r-1} \quad i = 1, 2, \dots, r$$

是 $GF(q^m)$ 域中的多项式。 $x_i (i=1, 2, \dots, n)$ 是 $GF(q^m)$ 域中的不同元素； y_1, y_2, \dots, y_n 是 $GF(q^m)$ 中的非0元素。该码的码长为 n ，信息位 $k=n-mr$ ，最小距离 $d \geq r+1$ 。

由式(7.11.1)及 $g_i(x)$ 的定义可知， \mathbf{H}_{AT} 也可写成：

$$\mathbf{H}_{AT} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1r} \\ c_{21} & c_{22} & \cdots & c_{2r} \\ \vdots & \vdots & & \vdots \\ c_{r1} & c_{r2} & \cdots & c_{rr} \end{bmatrix} \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ \vdots & \vdots & & \vdots \\ x_1^{r-1} & x_2^{r-1} & \cdots & x_n^{r-1} \end{bmatrix} \begin{bmatrix} y_1 & & & \mathbf{0} \\ & y_2 & & \\ & & \ddots & \\ \mathbf{0} & & & y_n \end{bmatrix} = \mathbf{CXY} \quad (7.11.2)$$

可以证明上式中的 \mathbf{C} 矩阵对码的最小距离并无影响，不同的 \mathbf{C} 矩阵，只得到不同的等价码而已。因而从码的最小的距离角度考虑，式(7.11.2)也可写成：

$$\mathbf{H}_{AT} = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1n} \\ \vdots & \vdots & & \vdots \\ h_{r1} & h_{r2} & \cdots & h_{rn} \end{bmatrix} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ \vdots & \vdots & & \vdots \\ x_1^{r-1} & x_2^{r-1} & \cdots & x_n^{r-1} \end{bmatrix} \begin{bmatrix} y_1 & & & \mathbf{0} \\ & y_2 & & \\ & & \ddots & \\ \mathbf{0} & & & y_n \end{bmatrix} = \mathbf{XY} \quad (7.11.3)$$

把该式与广义 RS 码的校验矩阵式(7.3.19)比较可知，若 \mathbf{H}_{AT} 中取 $x_i = \alpha_i$, $y_i = \gamma_i$ ，则 $A_q(\alpha, \gamma)$ 码就是 $GRS_k(\alpha, \gamma)$ 码中的所有 q 进制子码所组成的子域子码。AT 码不仅与 GRS 码密切相关，而且也与前节所述的各类码有密切关系。

例 7.16 取 $h_{ij} = \alpha^{(i-1)(j-1)}$ 或 $y_1 = y_2 = \cdots = y_7 = 1$, $x_j = \alpha^{j-1}$, $i=1, 2, j=1, 2, \dots, 7$, α 是 $GF(2^3)$ 上的本原元素，它是 x^3+x+1 的根。由此可知， $A_2(\alpha, 1)$ 交替码的 \mathbf{H} 矩阵由式(7.11.3)可知是

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 \end{bmatrix}$$

每个元素若用相应的二进制三重表示，则

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

得到一个[7, 3, 4]码，它就是增余删信循环汉明码。

该例中若取 $h_{ij} = \alpha^{(j-1)}$ 或 $y_j = x_j = \alpha^{j-1}$, $i=1, 2, j=1, 2, \dots, 7$ 。则 $A_2(\alpha, \alpha)$ 交替码的校验矩阵为

$$H = \begin{bmatrix} 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 \\ 1 & \alpha^2 & \alpha^4 & \alpha^6 & \alpha & \alpha^3 & \alpha^5 \end{bmatrix}.$$

该阵中的第二行是第一行的 $2(q)$ 次幂，所以是多余的，可以去掉。因而

$$H = [1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6] = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

得到一个[7, 4, 3]循环汉明码。 ■

由这些例子看出，在交替码的 H_{AT} 矩阵中， h_{ij} 或 x_i, y_i 的不同取值就得到了不同的码类，如表 7-13 所示。

表 7-13 各码类校验矩阵中 h_{ij} 元素的取值及条件

| 码类 | h_{ij} | 条件 | 构造者 | 注记 |
|-------|---|---|----------------------------------|-----------|
| GRS | $\alpha_i^{j-1} r_i$ | r_i 是 $GF(Q^m)$ 的非零元 | Delsart | Q^m 进制码 |
| AT | $\alpha_i^{j-1} r_i$ | r_i 是 $GF(Q^m)$ 的非零元 | Helgert | Q 进制码 |
| GBCH | $\alpha_i^{j-1} r_i$ | $\{\alpha_i\}$ 是 $GF(Q^m)$ 中加法群的子群 | Chien-Chen | Q 进制码 |
| Goppa | $\alpha_i^{j-1} \frac{1}{g(\alpha_i)}$ | $g(z)$ 是 t 次任意多项式 | Goppa | Q 进制码 |
| S | $\alpha_i^{j-1} \frac{\alpha_i^{m_0}}{g(\alpha_i)}$ | $g(z) = \prod_{u=1}^t (z - z_u)$ $z_u \in GF(q^m)$, m_0 是整数 | Srivastava | Q 进制码 |
| BCH | $\alpha_i^{j-1} \frac{\alpha_i^{m_0}}{g(\alpha_i)}$ | $g(z) = z^t$, m_0 是整数 | Bose Chaudhuri Hocquenghem | Q 进制码 |

AT 码、GS 码、Goppa 码以及 GBCH 码等各码类之间的相互关系如图 7-13 所示。图中的 GBCH 码是 BCH 码的推广，称为广义 BCH 码，它也是交替码的一个子类，所有上述这些码都是 GRS 码的一个子类，它们都是 GRS 码的子域子码。

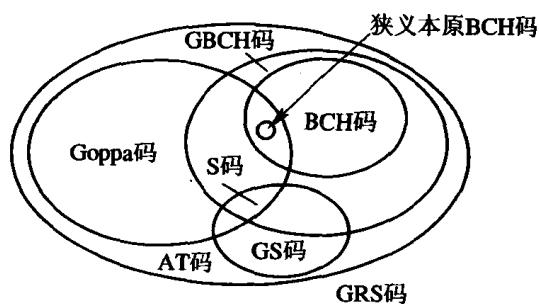


图 7-13 交替码各子类之间的关系

二、GBCH 码

1975年钱和陈(Chien & Chen)基于 MS 多项式提出了广义 BCH 码的概念，它也是包含很多子类的广义码之一。

定义 7.11.2 设 $p(z)$ 和 $g(z)$ 都是 $GF(q^n)$ 上的多项式， $\deg p(z) \leq n-1$, $\deg g(z) = r \leq n-1$, 且 $(z^n - 1, p(z)) = (z^n - 1, g(z)) = 1$ 。又设 $a(x)$ 是 $GF(q)$ 上的 $n-1$ 次多项式，它的 MS 多项式为 $A(z)$ ，且

$$[A(z)p(z)]_n \equiv 0 \pmod{g(z)} \quad (7.11.4)$$

则我们把 $GF(q)$ 上的所有满足式(7.11.4)的 $a(x)$ 多项式集合，称为一个 q 进制 GBCH 码，并用 $GBCH(p, g)$ 表示。上式中 $[A(z)p(z)]_n$ 表示两个多项式之积被 $z^n - 1$ 除后所得之余式。

可以证明 $GBCH(p, g)$ 码有如下形式的校验矩阵^[18]:

$$\begin{aligned} H &= \begin{bmatrix} p_0/g_0 & p_1\alpha/g_1 & \cdots & p_{n-1}\alpha^{n-1}/g_{n-1} \\ p_0/g_0 & p_1\alpha^2/g_1 & \cdots & p_{n-1}\alpha^{2(n-1)}/g_{n-1} \\ \vdots & \vdots & & \vdots \\ p_0/g_0 & p_1\alpha^r/g_1 & \cdots & p_{n-1}\alpha^{r(n-1)}/g_{n-1} \end{bmatrix} \\ &= \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \alpha^{r-1} & \alpha^{2(r-1)} & \cdots & \alpha^{(n-1)(r-1)} \end{bmatrix} \begin{bmatrix} p_0/g_0 \\ p_1\alpha/g_1 \\ \vdots \\ p_{n-1}\alpha^{n-1}/g_{n-1} \end{bmatrix} \end{aligned} \quad (7.11.5)$$

式中， p_i 、 g_i 分别是 $p(z)$ 和 $g(z)$ 多项式的系数：

$$p(z) = p_0 + p_1 z + p_2 z^2 + \cdots + p_{n-1} z^{n-1}$$

$$g(z) = g_0 + g_1 z + g_2 z^2 + \cdots + g_{n-1} z^{n-1}$$

由式(7.11.5)可知， $GBCH(p, g)$ 码有以下参数： $n, k \geq n - mr, d \geq r + 1$ 。

把 GBCH 码的校验矩阵(7.11.5)式与 AT 码的校验矩阵 H_{AT} 比较可知，若在 H_{AT} 阵中取 $y_i = p_{i-1}\alpha^{i-1}/g_{i-1}$, $x_i = \alpha^{i-1}$, $i = 1, 2, \dots, n$, α 是 $GF(q^n)$ 上的 n 阶单位根，则此时的 AT 码就是 GBCH 码。因此 GBCH 码是交替码的一个子类，如图 7-13 所示。

随着 $p(z)$ 和 $g(z)$ 多项式的不同取值，可得到 GBCH 码的不同子类。例如，在二进制情况下，若取 $p(z) = z^{n-1}$, $g(z) = g(z)$ ，则得到的 GBCH 码就是 Goppa 码，其位置集 L 由 $GF(2^n)$ 上的 n 阶单位根组成。

若 $p(z) = z^{\delta+b-2}$, $g(z) = z^{\delta-1}$ 。由此得到的 GBCH 码的校验矩阵由式(7.11.5)可知为

$$H = \begin{bmatrix} 1 & \alpha^b & \alpha^{2b} & \cdots & \alpha^{(n-1)b} \\ 1 & \alpha^{b+1} & \alpha^{2(b+1)} & \cdots & \alpha^{(n-1)(b+1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \alpha^{b+\delta-2} & \alpha^{2(b+\delta-2)} & \cdots & \alpha^{(n-1)(b+\delta-2)} \end{bmatrix}$$

显然，它就是以 $\alpha^b, \alpha^{b+1}, \dots, \alpha^{b+\delta-2}$ 连续元素为根的 BCH 码的校验矩阵。因此，GBCH 码完全包含了 BCH 码。

例 7.17 取 $n=15$, $p(z)=z$, $g(z)=z^2$ 。由式(7.11.5)可得 GBCH(z, z^2)码的 H 矩阵为

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 & \alpha^7 & \alpha^8 & \alpha^9 & \alpha^{10} & \alpha^{11} & \alpha^{12} & \alpha^{13} & \alpha^{14} \end{bmatrix}$$

式中, α 是 $GF(2^4)$ 上的本原元素, 相应地本原多项式是 x^4+x+1 。由此矩阵可知, 这是一个[15, 10, 4]二进制 BCH 码, 它以 $1, \alpha, \alpha^2$ 为根, 是一个增余删信汉明码。

又如, 取 $p(z)=z^5$, $g(z)=z^2$, $n=15$, 则该 GBCH 码的校验矩阵是

$$H = \begin{bmatrix} 1 & \alpha^4 & \alpha^8 & \alpha^{12} & \alpha & \alpha^5 & \alpha^9 & \alpha^{13} & \alpha^2 & \alpha^6 & \alpha^{10} & \alpha^{14} & \alpha^3 & \alpha^7 & \alpha^{11} \\ 1 & \alpha^5 & \alpha^{10} & 1 & \alpha^5 & \alpha^{10} \end{bmatrix}$$

得到一个[15, 9, 3]二进制 BCH 码, 它以 α, α^5 为根, 该码的最小距离 $d \geq 3$ 。 ■

在线性码的构造中, 除了上面提到的 AT 码、GBCH 码、Goppa 码等以外, 自 1976 年以来曾开明(K. M. Tzeng)和齐默尔曼(Zimmerman)等还利用拉格郎日内插公式对 Goppa 码进行了推广。并详细分析了拉格郎日内插与有理分式表示码之间的关系, 以及和 MS 多项式、孙子定理、交替矩阵之间的关系, 并在孙子定理基础上构造了一类范围更广的码——超码。有关这类码这里不再介绍。

可以证明长的交替码是好码, 并存在有能满足 V-G 限的交替码, 有关此问题的证明可参阅文献[18]。

* § 7.12 交替码的欧几里德译码算法

这一节将讨论交替码的译码。由于交替码是一类包含有很多子类的广义码, 因此讨论它的译码问题有普遍意义。

如同 BCH 码的译码, 交替码译码也分为 3 步: 第一步由接收到的 $R=(r_1, r_2, \dots, r_n)$ 计算伴随式 $S(z)$; 第二步由 $S(z)$ 寻找错误位置多项式 $\sigma(z)$ 与错误值多项式 $\omega(z)$; 第三步由 $\sigma(z)$ 求出它的根找到错误位置, 并与 $\omega(z)$ 一起计算错误值大小, 并纠正 R 中的错误。在这三步中, 求 $\sigma(z)$ 和 $\omega(z)$ 的第二步是最关键的一步, 它可以用 § 7.5 中所讲的迭代译码算法求解, 也可用欧几里德算法求解 $\sigma(z)、\omega(z)$ 。为了更好地理解多项式之间的欧几里德算法, 下面我们将证明它。

定理 7.12.1(多项式欧几里德算法) 令 $r_{-1}(z)$ 和 $r_0(z)$ 是多项式, 且 $\partial \circ r_0(z) \leq \partial \circ r_{-1}(z)$, 它们之间的最高公因子是 $h(z)$ 。则必存在有一对多项式 $U(z)$ 和 $V(z)$, 使

$$U(z)r_{-1}(z) + V(z)r_0(z) = h(z) \quad (7.12.1)$$

成立。这里 $\partial \circ U(z)$ 和 $\partial \circ V(z)$ 均小于 $\partial \circ r_{-1}(z)$ 。

证明 设两个多项式 $U_i(z)$ 和 $V_i(z)$ 的初值定义如下:

$$\begin{aligned} U_{-1}(z) &= 0 & U_0(z) &= 1 \\ V_{-1}(z) &= 1 & V_0(z) &= 0 \end{aligned} \quad (7.12.2)$$

且规定:

$$\begin{aligned} U_i(z) &= q_i(z)U_{i-1}(z) + U_{i-2}(z) \\ V_i(z) &= q_i(z)V_{i-1}(z) + V_{i-2}(z) \end{aligned} \quad (7.12.3)$$

因而

$$\begin{aligned}
\begin{bmatrix} U_i(z) & U_{i-1}(z) \\ V_i(z) & V_{i-1}(z) \end{bmatrix} &= \begin{bmatrix} U_{i-1}(z) & U_{i-2}(z) \\ V_{i-1}(z) & V_{i-2}(z) \end{bmatrix} \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \\
&= \begin{bmatrix} U_{i-2}(z) & U_{i-3}(z) \\ V_{i-2}(z) & V_{i-3}(z) \end{bmatrix} \begin{bmatrix} q_{i-1}(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} = \dots \\
&= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} q_1(z) & 1 \\ 1 & 0 \end{bmatrix} \dots \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix}
\end{aligned} \tag{7.12.4}$$

上式也可写成：

$$\begin{aligned}
\begin{bmatrix} r_{i-2}(z) \\ r_{i-1}(z) \end{bmatrix} &= \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} r_{i-1}(z) \\ r_i(z) \end{bmatrix} \\
\begin{bmatrix} r_{i-3}(z) \\ r_{i-2}(z) \end{bmatrix} &= \begin{bmatrix} q_{i-1}(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} r_{i-1}(z) \\ r_i(z) \end{bmatrix} \\
&\vdots \\
\begin{bmatrix} r_{-1}(z) \\ r_0(z) \end{bmatrix} &= \begin{bmatrix} q_1(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} q_2(z) & 1 \\ 1 & 0 \end{bmatrix} \dots \begin{bmatrix} q_i(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} r_{i-1}(z) \\ r_i(z) \end{bmatrix}
\end{aligned}$$

由式(7.12.4)可知上式变成为

$$\begin{bmatrix} r_{-1}(z) \\ r_0(z) \end{bmatrix} = \begin{bmatrix} U_i(z) & U_{i-1}(z) \\ V_i(z) & V_{i-1}(z) \end{bmatrix} \begin{bmatrix} r_{i-1}(z) \\ r_i(z) \end{bmatrix}$$

式(7.12.4)矩阵的行列式值是 $(-1)^i$, 因而两边乘以逆阵得

$$\begin{bmatrix} r_{i-1}(z) \\ r_i(z) \end{bmatrix} = (-1)^i \begin{bmatrix} V_{i-1}(z) & -U_{i-1}(z) \\ -V_i(z) & U_i(z) \end{bmatrix} \begin{bmatrix} r_{-1}(z) \\ r_0(z) \end{bmatrix} \tag{7.12.5}$$

写成一般形式为

$$r_j(z) = (-1)^j [-V_j(z)r_{-1}(z) + U_j(z)r_0(z)] \tag{7.12.6}$$

它就是(7.12.1)式。因而：

$$\begin{aligned}
\partial \circ U_i(z) &= \sum_{k=1}^i \partial \circ q_k(z) \\
\partial \circ r_{i-1}(z) &= \partial \circ r_{-1}(z) - \sum_{k=1}^i \partial \circ q_k(z) \\
\partial \circ U_i(z) &= \partial \circ r_{-1}(z) - \partial \circ r_{i-1}(z) < \partial \circ r_{-1}(z)
\end{aligned}$$

类似地可知 $\partial \circ V_i(z) < \partial \circ r_{-1}(z)$ 。

下面我们将采用欧几里德算法求出 $\sigma(z)$ 。设 $A(x, y)$ 是 $\text{GF}(q)$ 上的一个交替码，它的校验矩阵 $H=XY$ ，码的最小距离 $d=2t+1$ 。设在接收矢量 R 中以下码元位置： $z_1=x_{i1}$ ， $z_2=x_{i2}$ ， \dots ， $z_t=x_{it}$ 上发生了错误，错误值大小是： $e_{i1}=Y_{i1}$ ， $e_{i2}=Y_{i2}$ ， \dots ， $e_{it}=Y_{it}$ 。由此可得伴随式

$$S = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ \vdots & \vdots & & \vdots \\ x_1^{r-1} & x_2^{r-1} & \cdots & x_n^{r-1} \end{bmatrix} \begin{bmatrix} y_1 & & & \\ & y_2 & & \\ & & \ddots & \\ & & & y_n \end{bmatrix} = \begin{bmatrix} \vdots \\ y_{i1} \\ y_{i2} \\ \vdots \\ y_{it} \\ \vdots \end{bmatrix} = \begin{bmatrix} s_0 \\ s_1 \\ \vdots \\ s_{r-1} \end{bmatrix}$$

式中, $r=2t$, 且:

$$s_\mu = \sum_{\gamma=0}^t x_\gamma^\mu Y_\gamma a_\gamma \quad \mu = 0, 1, 2, \dots, r-1 \quad (7.12.7)$$

$$S(z) = \sum_{\mu=0}^{r-1} s_\mu z^\mu \quad (7.12.8)$$

由 $S(z)$ 可得到错误位置多项式 $\sigma(z)$, 以及错误伴随多项式 $\omega(z)$,

$$\sigma(z) = \sum_{i=1}^t (1 - x_i z) = \sum_{i=0}^t \sigma_i z^i \quad \sigma_0 = 1 \quad (7.12.9)$$

该式中各符号的意义与式(7.4.7)相同。与式(7.5.6)的关键方程类似。可得到 $S(z)$ 、 $\omega(z)$ 与 $\sigma(z)$ 之间有如下关系:

$$S(z)\sigma(z) \equiv \omega(z) \pmod{z'} \quad (7.12.10)$$

称该式是译 $A_q(x, y)$ 码的关键方程。

我们的目的就是在已知 $S(z)$ 条件下, 求解满足(7.12.10)式的 $\sigma(z)$ 和 $\omega(z)$ 。式(7.12.10)是一个同余式, 因而解不是唯一的。为了使解唯一必须对 $\sigma(z)$ 与 $\omega(z)$ 规定一些条件, 这些条件与 BCH 码译码中, 用迭代法求解 $\sigma(x)$ 与 $\omega(x)$ 时所限制的条件相同。这就是在求解式(7.12.10)过程中, 始终要保持 $\sigma(z)$ 的次数尽可能小, 且 $\deg \omega(z) < \deg \sigma(z)$ 。由于码仅能纠正 t 个错误, 因而最后所求得的 $\deg \sigma(z) \leq t$, $\deg \omega(z) \leq t-1$ 。

下面我们利用欧几里德算法求满足式(7.12.10)的 $\sigma(z)$ 与 $\omega(z)$ 。式(7.12.10)可写成:

$$S(z)\sigma(z) = q_i(z)z' + \omega(z) \quad (7.12.11)$$

或

$$\omega(z) = -q_i(z)z' + S(z)\sigma(z) \quad (7.12.12)$$

由欧几里德算法可知 $\omega(z)$ 是 $S(z)$ 与 z' 的最大公因子。为了由 $S(z)$ 与 z' 求得 $\omega(z)$ 与 $\sigma(z)$, 就可用欧几里德算法逐步进行。把式(7.12.12)与(7.12.6)比较可得如下算法:

(1) 令

$$r_{-1}(z) = z' \quad r_0(z) = S(z)$$

(2) 利用欧几里德算法由式(7.12.2)所确定的初值以及(7.12.3)式, 用欧几里德算法逐步填满下表:

| j | $U_j(z)$ | $V_j(z)$ | $r_j(z)$ | $q_j(z)$ |
|----------|----------|----------|---------------|----------|
| -1 | 0 | 1 | z' , $r=2t$ | |
| 0 | 1 | 0 | $S(z)$ | |
| \vdots | | | | |
| k | | | | |

直至计算到 $r_k(z)$, 满足

$$\partial \circ r_{k-1}(z) \geq t \quad \partial \circ r_k(z) \leq t-1 \quad (7.12.13)$$

为止。由式(7.12.6)和式(7.12.11)、式(7.12.12)可知, 错误位置和错误值多项式为:

$$\sigma(z) = \delta U_k(z) \quad (7.12.14)$$

$$\omega(z) = (-1)^k \delta r_k(z) \quad (7.12.15)$$

这里 δ 是一个常数, 由条件 $\sigma_0=1$ 得到。

显然以上两式满足:

$$\omega(z) \equiv \sigma(z) S(z) \pmod{z^t} \quad (7.12.16)$$

$$\partial \circ \sigma(z) \leq t \quad \partial \circ \omega(z) \leq t-1 \quad (7.12.17)$$

(3) 类似于 § 7.5 节中由 $\omega(x)$ 求错误值的过程相似, 可由 $\omega(z)$ 得到错误值

$$Y_{ij} = \frac{\omega(x_i^{-1})}{y_j \prod_{\substack{j=1 \\ i \neq j}}^r (1 + x_i^{-1} x_{jr})} \quad (7.12.18)$$

对二进制码来说这一步可以省略。上述译码过程, 如图 7-14 所示。

例 7.18 例 7.16 中的二进制 $A(\alpha, 1)$ 码, 校验矩阵为

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 \end{bmatrix}$$

$y_1=y_2=\cdots=y_7=1$ 。能纠正一个错误 $t=1$, $r=2^t=2$ 。设接收到的 $R=(0010000)$, 求发送的码字。

由 $S=H \cdot R^T$ 可得:

$$S(z) = 1 + \alpha^2 z$$

下面用欧几里德算法求 $\sigma(z)$, 如下表所示:

| j | $U_j(z)$ | $V_j(z)$ | $r_j(z)$ | $q_j(z)$ |
|-----|-------------------------|----------|-------------------------|-------------------------|
| -1 | 0 | 1 | $z' = z^2$ | |
| 0 | 1 | 0 | $S(z) = 1 + \alpha^2 z$ | |
| 1 | $\alpha^5 z + \alpha^3$ | 1 | α^3 | $\alpha^5 z + \alpha^3$ |

由此可得:

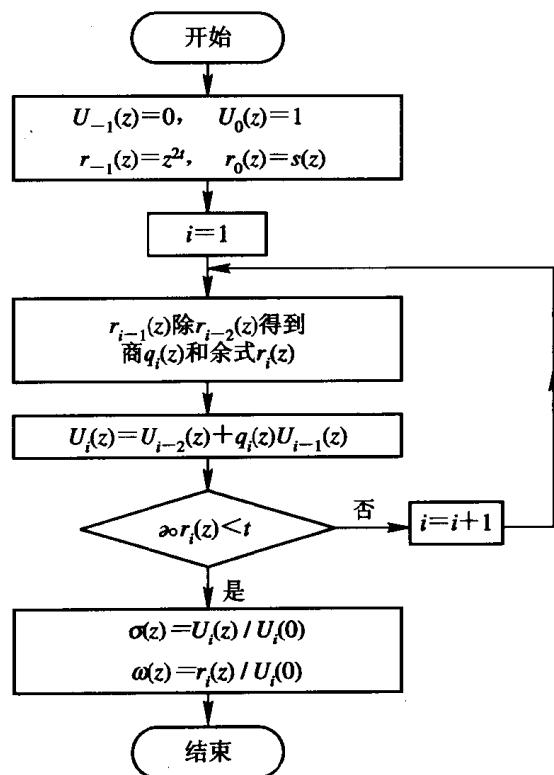


图 7-14 欧几里德译码算法流程图

$$S\sigma(z) = \alpha^5 z + \alpha^3 = \alpha^3(z\alpha^2 + 1) \quad z = \alpha^{-2}$$

因而 α^2 位置发生了错误，故 C 是一个全为 0 码字。 ■

上面我们简单介绍了欧几里德译码算法，由这种算法过程可知，这种译码算法可用计算机程序在计算机上较方便地实现。但由于这种译码算法需进行除法运算，并记录大量的中间结果，因而其译码速度比 BM 迭代译码算法要慢，所需的设备也较多。

习 题

1. 证明用 $g(x) = g^*(x)$ 生成的，且含有以 1 为根的二进制 BCH 码，最小距离至少为 6。
2. 若 BCH 码以 $\alpha, \alpha^{m_0+1}, \dots, \alpha^{m_0+\delta-2}$ 为根，现用另一个与 α 同级元素 α' 代替 α ，证明这种代换结果得到一个等价码。
3. 确定 $n=15$ 的所有本原二进制 BCH 码的生成多项式。
4. 确定 $n=31$ 的所有本原二进制 BCH 码的生成多项式，找出各码的 R, d 。
5. 构造一个 $n=21, d \geq 7$ 的码率最大的 BCH 码，确定它的生成多项式 $g(x)$ 。
6. 对 $m \geq 3, t < 2^{m-1}$ ，存在有一个长为 2^m+1 纠 t 个错误的二进制本原 BCH 码吗？若有找出它的 $g(x)$ 。
7. 设 $[n, k]$ 二进制本原 BCH 码，能纠正 t 个错误，若 $2t+1$ 是 n 的因子，证明码的最小距离恰好为 $2t+1$ 。
8. 已知二进制 $[21, 7]$ 码 $g(x)$ 的根集 $R \supseteq \{\alpha^i | i=1, 2, 3, 4, 6, 7, 8, 9, 11, 12, 14, 15, 16, 18\}$ ，分析该码的最小距离。
9. 已知二进制 $[31, 11]$ 码 C_1 的 $g_1(x)$ 的根集 $R \supseteq \{\alpha, \alpha^3, \alpha^7, \alpha^{15}\}$ ，另一个 C_2 码 $g_2(x)$ 的根集 $R_2 \supseteq \{\alpha, \alpha^3, \alpha^{11}\}$ ，用已知各种限分别求出 C_1 码和 C_2 码的距离。
10. 用迭代译码，译二进制 $[15, 7]$ 码的接收码矢 $R(x) = x + x^7$ ，求出发送码字。
11. 构造一个 $GF(2^4)$ 上的长为 $n=15$ 的纠两个错误的 RS 码，找出它的生成多项式和 k 。且用迭代译码法译接收矢量 $R(x) = \alpha x^3 + \alpha^{11} x^7$ 。
12. 一个 $GF(2^m)$ 上的纠正 t 个错误的 RS 码，若它的生成多项式为

$$g(x) = (x + \alpha)(x + \alpha^2) \cdots (x + \alpha^{2^t})$$
 这里， $\alpha \in GF(2^m)$ 是一个本原域元素，证明该码最小距离恰好为 $2t+1$ 。
13. 试画出纠正两个错误的 $[31, 21]$ 二进制 BCH 码的伴随式计算电路。
14. 试画出 $GF(2^4)$ 上的长为 15 的纠单个错误 RS 码的编码电路，并画出该码的伴随式计算电路。
15. 设 $\alpha \in GF(2^5)$ 是本原域元素，令 $\beta = \alpha^3$ ，若码的生成多项式以 $\beta, \beta^2, \beta^3, \beta^4$ 为根，求出该码的生成多项式，码长和信息位个数。
16. 在纠 t 个错误的 BCH 码中，证明接收码字中仅产生一个错误的充要条件是式 (7.5.25) 成立。
17. 用频域译码方法译题 6 中的接收矢量。
18. 用解线性方程组的方法译第 10 题中的接收矢量。
19. 用频域译码方法译第 11 接收矢量 $R(x)$ ，找出发送码字。
20. 构造一个 $n=15$ 纠两个错误的二进制既约 Goppa 码，并与 $[15, 7, 5]$ BCH 码比

较，能得出什么结论？试用欧几里德译码方法译接接收矢量 $R=(00100010000000)$ 。

参 考 文 献

- [1] W. W. Peterson and E. J. Weldon, Jr., Error Correcting Codes, Cambridge, MA, 1972.
- [2] R. E. Blahut:《差错控制码的理论与实践》徐秉铮等译, 华南理工大学出版社, 1988.
- [3] 王新梅:《纠错码与差错控制》, 人民邮电出版社, 1989.
- [4] T. Schaub, "A linear complexity approach to cyclic codes", A Dissertation for the Degree of Ph. Doctor, 1988.
- [5] J. H. VanLint, R. M. Wilson, "On the minimum distance of cyclic codes", IEEE Trans. on IT, No. 1, pp. 23—40, 1986.
- [6] T. Helleseth, "On covering radius of cyclic linear codes and arithmetic codes", Discr. Appl. Math., Vol. 11, pp. 157—173, 1985.
- [7] S. G. Vleduts and A. N. Skorobogatv, "Covering radius for long BCH codes", Probl. Peredachi Inform. Vol. 25, No. 1, pp. 28—34, 1989.
- [8] D. C. Gorenstein, W. W. Peterson and W. Znerler, "Two — error correcting BCH codes are quasi — perfect", Inform. Control, Vol. 3, pp. 291—294, 1960.
- [9] T. Helleseth, "All Binary 3 — error — correcting BCH codes of length $2^m - 1$ have covering radius 5", IEEE Trans. on IT, No. 2, pp. 257—258, 1978.
- [10] G. D. Cohen et al., "Covering radius—survey and recent results", IEEE Trans. on IT, No. 3, pp. 328—343, 1985.
- [11] S. Lin, T. Kasam., "Encoding and decoding of reed — solomon codes in dual basis", 《电子学报》, No. 4, pp. 6—20, 1986.
- [12] C. L. Chen, "High—speed decoding of BCH codes", IEEE Trans. on IT, No. 2, pp. 254—256, 1981.
- [13] 姚明余. 忻鼎稼: BCH 码的一种新的译码方法, 《通信学报》, No. 5, pp. 10—14, 1989.
- [14] A. M. Michelson, A. H. Levesque, Error—Control Techniques For Digital Communication, John Wiley & Sons, New York, 1985.
- [15] G. L. Feng and K. K. Tzeng, "A generalized euclidean algorithm for multisequence shift—register synthesis", IEEE Trans. on IT, No. 3, pp. 584—594, 1989.
- [16] P. Bours, et al. ', "Algebraic decoding beyond e_{BCH} of some binary cyclic codes, when $e > e_{BCH}$ ", IEEE Trans. on IT, No. 1, pp. 214—222, 1990.
- [17] P. Stevens, "Extension of the BCH decoding algorithm to decode binary cyclic codes up to their maximum error correction capacities", IEEE Trans. on IT, No. 5, pp. 1332—1341, 1988.
- [18] F. J. MacWilliams and N. J. A. Sloane, The Theory of Error Correcting Codes, North—Holland, 1977.

* 第八章 代数几何码

代数几何码理论的确立归功于苏联数学家戈帕(Goppa)。他在 70 年代末及 80 年代初所发表的一系列重要论文，将代数几何的理论与方法系统地应用于编码理论中，发现代数几何中的许多概念与结论均可转换成纠错编码的相应性质，使得原来线性码中的重要参数，诸如码长、距离、维数等，具有全新的几何意义及算术意义。继此之后，许多著名编码学家，诸如莱查特(Lachaud)，范林特(Van Lint)，斯普林格(Springer)及特斯法斯曼(Tsfasman)等人，在这一领域中做了许多重要的推进工作。特别是特斯法斯曼等人的工作，基于代数几何与 Goppa 码的思想，利用模曲线构造了在性能上优于 V—G 限，因而也优于 J 码的码序列。这是多年来关于 V—G 限和 Shannon 码构造上在理论方面的重大突破，在编码学界产生了轰动效应。

由于上述的一系列奠基性工作，使代数几何码形成了编码理论中具有领导性的重要研究领域。尽管代数几何码走向实用阶段尚有一段艰难的历程，但是许多编码学家都预言这可能是本世纪最后 10 年的事情。

代数几何作为现代数学的重要研究领域在我国仍处于初级阶段，特别是这一学科目前尚不为我国广大编码学者所熟悉。考虑到本书的对象，我们只能对代数几何码的基本概念及重要结论作一个简明的介绍，不追求理论的严谨与系统，许多重要结果仅以实例说明。有志于深入研究这一领域的读者可去查阅有关文献及专著。

§ 8.1 代数几何的研究对象

代数几何是几何学中的一个重要研究领域，它研究平面代数曲线、空间代数曲线和代数曲面，更一般地，研究 n 维空间的代数簇。所谓代数簇，就是由一组代数方程所确定的点集以及由这些点集通过一定的规则导出的对象。

例如，在普通直角坐标中，由代数方程

$$F(x, y) = 0$$

所决定的曲线即为平面代数曲线，这里 $F(x, y)$ 是关于变量 x, y 的二元多项式。平面上的直线、圆锥曲线都是代数曲线，但是 $y - \sin x = 0$ 所决定的正弦曲线便不是代数曲线。类似地，由三元多项式

$$F(x, y, z) = 0$$

决定的点集即为代数曲面。两个无关且相容的三元代数方程组

$$F_1(x, y, z) = 0$$

$$F_2(x, y, z) = 0$$

所决定的点集，即为空间代数曲线。一般地，由 n 元代数方程组

$$F_i(x_1, x_2, \dots, x_n) = 0 \quad i = 1, 2, \dots, m$$

所决定的点集即为代数簇。研究一次曲线(直线)及一次曲面(平面)，以及二次曲线和曲面是普通解析几何中的内容。在上一世纪以前，代数几何是从研究三次及四次曲线及曲面的分类开始的。从19世纪末开始，人们才开始研究一般代数簇的系统结构。在代数几何的研究中采用拓扑学及抽象代数方法则是本世纪的事情。

在代数几何学中，主要的工具是定义于有限域上的代数曲线，特别是仿射曲线及射影曲线。因此，以下我们着重介绍与此有关的基本概念。

§ 8.2 仿射空间与仿射变换

假设 K 是一个域， V_K^n 代表 K 上定义的 n 维向量空间。借助于 V_K^n 可引入一个 n 维仿射空间。

定义 8.2.1 一个 n 维仿射空间 A_K^n 是点 P, Q, R, \dots 的集，满足：

1° 每一个有序点偶 (P, Q) ，恰有 $v \in V_K^n$ 与之对应，记为 $\overrightarrow{PQ} = v$ 。

2° 每一个点 $P \in A_K^n$ 及每一向量 $v \in V_K^n$ ，恰有一点 $Q \in A_K^n$ 使 $\overrightarrow{PQ} = v$ 。

3° 对于 A_K^n 中任意三点 P, Q, R ，恒有

$$\overrightarrow{PQ} + \overrightarrow{QR} = \overrightarrow{PR}$$

设 $v \in V_K^n$ 。由定义 8.2.1 之 2°，对于每一点 $P \in A_K^n$ ，恰有一点 $Q \in A_K^n$ 使 $\overrightarrow{PQ} = v$ 。由此定义了 A_K^n 上的一个映射

$$T_v : P \mapsto Q, \quad PT_v = Q$$

称此映射 T_v 为 A_K^n 上的一个平移(见图 8-1)。

在 A_K^n 中任选一点 O ，称为原点。于是由定义 8.2.1，对于每一个 $P \in A_K^n$ ，存在 $v \in V_K^n$ 使 $\overrightarrow{OP} = v$ ，称 v 为点 P 的位置向量。若 v 在 V_K^n 的一个基底 e_1, e_2, \dots, e_n 之下的坐标为 (x_1, x_2, \dots, x_n) ，则称点 P 具有的坐标为 (x_1, x_2, \dots, x_n) 。特别是，点 O 之坐标为 $(0, 0, \dots, 0)$ 。 A_K^n 的仿射坐标系记为 $\{\mathbf{0}, e_1, e_2, \dots, e_n\}$ 。

仿射坐标是比普通直角坐标更为一般的坐标系。

A_K^n 中点的平移变换可通过 V_K^n 中的(位置)向量的平移表达。

事实上，从 A_K^n 中的点 P 与 V_K^n 中位置向量 $\overrightarrow{OP} = v$ 之间的一一对应关系，平移变换 $PT_b = Q$ ($b \in V_K^n$)，即 $\overrightarrow{PQ} = b$ ，可视为向量 \overrightarrow{OP} 在变换 T_b 之下变换为向量 \overrightarrow{OQ} ，记为 $\overrightarrow{OPT}_b = \overrightarrow{OQ}$ 。由于 $\overrightarrow{OP} = v$ ， $\overrightarrow{OQ} = \overrightarrow{OP} + \overrightarrow{PQ} = v + b$ ，因此由图 8-2 所示：

$$vT_b = v + b$$

进一步，我们还可以建立仿射子空间的概念。

定义 8.2.2 设 α^n 为 n 维仿射空间 A_K^n 的非空子集， S^m 为 V_K^n 的 m 维向量空间。如

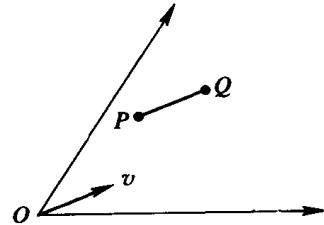


图 8-1 A_K^n 上的一个平移

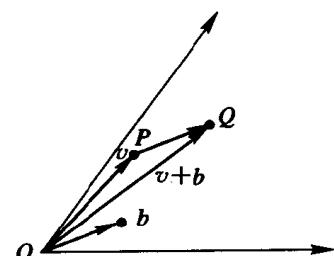


图 8-2 A_K^n 上的平移变换

果在 A_K^n 与 V_K^n 的相应关系下, α^m 恰好是相应于 S^m 的一个仿射空间, 则称 α^m 是 A_K^n 的一个 m 维仿射子空间。

A_K^n 的一维仿射子空间 α^1 称为直线, 二维仿射子空间 α^2 称为平面。

依据仿射坐标, 我们可以对仿射空间中的几何图像建立相应的方程。

例如, 考虑 A_K^2 中一条直线, 它通过点 $P =$

(p_1, p_2) , 且该直线(作为一维仿射空间)相应的一维向量空间以 $v = (v_1, v_2)$ 为基底, 如图 8-3 所示。设 $X = (x_1, x_2)$ 为该直线上任意一点。于是 \overrightarrow{PX} 必为该直线相应的一维向量空间中的向量, 因而可写成 $\overrightarrow{PX} = \rho v$ ($\rho \in K$)。又因 $\overrightarrow{PX} = \overrightarrow{OX} - \overrightarrow{OP} = (x_1, x_2) - (p_1, p_2) = (x_1 - p_1, x_2 - p_2)$, $\rho v = (\rho v_1, \rho v_2)$, 故

$$(x_1 - p_1, x_2 - p_2) = (\rho v_1, \rho v_2)$$

由于 $v \neq 0$ (基向量), 不妨设 $v_1 \neq 0$ 。于是, 由 $x_1 - p_1 = \rho v_1$ 得

$$\rho = \frac{x_1 - p_1}{v_1}$$

将 ρ 代入 $x_2 - p_2 = \rho v_2$ 中, 便有

$$v_2 x_1 - v_1 x_2 + (v_1 p_2 - v_2 p_1) = 0$$

令 $a_1 = v_2$, $a_2 = -v_1$, $a_0 = v_1 p_2 - v_2 p_1$, 便有

$$a_1 x_1 + a_2 x_2 + a_0 = 0 \quad (8.2.1)$$

式中, a_1 、 a_2 不全为 0。

反过来, 每一个形如式(8.2.1)的一次方程均代表 A_K^2 中的一条直线。事实上, 考虑满足方程式(8.2.1)的所有的点 $X = (x_1, x_2)$ 。由于 a_1 , a_2 不全为 0, 不妨设 $a_1 \neq 0$ 。于是 $x_1 = -a_1^{-1}(a_0 + a_2 x_2)$ 。选取 $P = (-a_1^{-1}a_0, 0)$, $v = (-a_2, a_1) \neq 0$, 便有

$$x = (-a_1^{-1}a_0, 0) + \rho(-a_2, a_1) = p + \rho v \quad \rho = a_1^{-1}x_2$$

此处点 x 与点 p 为点 X 与 P 的位置向量。这表明方程式(8.2.1)代表通过点 P 且由向量 v 构成的直线。

类似于上述方法, 我们可以建立仿射空间中的高次曲线的方程。仿射空间中的曲线称为仿射曲线。

在本节之末, 我们引入仿射变换的概念。

设 e_1, e_2, \dots, e_n 为 n 维向量空间 V_K^n 的基底。设 $\{\mathbf{0}, e_1, e_2, \dots, e_n\}$ 为相应的 n 维仿射空间 A_K^n 的坐标系。设 $X = (x_1, x_2, \dots, x_n)$ 为 A_K^n 中任意一点。所谓仿射变换是指线性变换

$$Y = CX + \mathbf{b} \quad (8.2.2)$$

式中

$$C = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{nn} \end{bmatrix}$$

为域 K 上的非异矩阵, $\mathbf{b} = (b_1, \dots, b_n)$ 为 A_K^n 中一点。因此, 仿射变换即为 A_K^n 上的非异齐

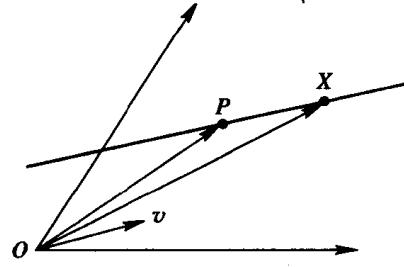


图 8-3 A_K^2 中的直线

次线性变换再加上平移。在变换式(8.2.2)之下,点 $X=(x_1, \dots, x_n)$ 变为点 $Y=(y_1, \dots, y_n)$ 。

仿射几何就是研究在仿射变换之下几何图形的不变性质及不变量。例如,在仿射变换下,直线仍变成直线。因此,直线是仿射几何的研究对象。但是,在 A_k^2 中椭圆 $b^2x_1^2+a^2x_2^2-a^2b^2=0$ 经过仿射变换 $x_1=r^{-1}ax'_1, x_2=r^{-1}bx'_2$ 可变为圆 $x'_1^2+x'_2^2-r^2=0$ 。因此,在仿射几何中圆与椭圆是一回事(仿射等价)。

§ 8.3 射影空间与射影变换

引入 n 维仿射空间的出发点是 n 维向量空间。引入 n 维射影空间的出发点则是域 K 上的 $n+1$ 维向量空间 V_K^{n+1} 。

设 $v, w \in V_K^{n+1}$ 。若存在 $r \neq 0 (r \in K)$ 使 $w = rv$, 则称向量 v 与 w 等价。

在等价的意义下, V_K^{n+1} 中的全部向量被分成等价类。零向量 $\mathbf{0}$ 构成由自身代表的一类。记为 $[\mathbf{0}]$ 。如果一个类中包含向量 v , 则此类用 $[v]$ 代表。

定义 8.3.1 在 V_K^{n+1} 中所建立的每一个异于 $[\mathbf{0}]$ 的类称为射影点。所有射影点的全体所构成之集称为域 K 上的 n 维射影空间, 记作 P_K^n 。

因此, P_K^n 中的点是 V_K^{n+1} 中的一维子空间。 V_K^{n+1} 中的二维子空间称为 P_K^n 中的射影直线, 三维子空间称为 P_K^n 中的射影平面, $m (m > 3)$ 维子空间称为 P_K^n 中的 $(m-1)$ 维超平面。

在射影空间中可以引进齐次坐标的概念。

设 e_1, \dots, e_n, e_{n+1} 是 V_K^{n+1} 的一个基底。于是 P_K^n 中每一点 $[X]$ 皆可表为

$$[X] = [x_1e_1 + \dots + x_ne_n + x_{n+1}e_{n+1}] = \left[\sum_{j=1}^{n+1} x_j e_j \right]$$

称 $(x_1, \dots, x_n, x_{n+1})$ 为点 $[X]$ 的射影齐次坐标。

显然, 若 $(x_1, \dots, x_n, x_{n+1})$ 为点 $[X]$ 的齐次坐标, 则对于任意 $r \in K, r \neq 0$, $(rx_1, \dots, rx_n, rx_{n+1})$ 亦为点 $[X]$ 的齐次坐标, 因为 $[X] = [rX]$ 。所以, 在射影空间中的点由坐标比 $x_1 : \dots : x_n : x_{n+1}$ 决定。反之, 每一个比值 $x_1 : \dots : x_n : x_{n+1}$ (x_1, \dots, x_n, x_{n+1} 不全为0) 决定 P_K^n 中唯一的一点。点 X 的齐次坐标有时也表示为 $(x_1 : \dots : x_n : x_{n+1})$ 。

作为例子, 我们建立射影平面 P_K^2 中直线的齐次坐标方程。

设 $[Y] = (y_1, y_2, y_3)$, $[Z] = (z_1, z_2, z_3)$ 为该直线上两点, $[X] = (x_1, x_2, x_3)$ 为该直线上任意一点。这表明相应的 V_K^3 中向量 X, Y, Z 线性相关, 即存在不全为0的 $\lambda_1, \lambda_2, \lambda_3$ 使 $\lambda_1X + \lambda_2Y + \lambda_3Z = 0$ 。即齐次方程组:

$$\begin{aligned}\lambda_1x_1 + \lambda_2y_1 + \lambda_3z_1 &= 0 \\ \lambda_1x_2 + \lambda_2y_2 + \lambda_3z_2 &= 0 \\ \lambda_1x_3 + \lambda_2y_3 + \lambda_3z_3 &= 0\end{aligned}$$

有非零解 $(\lambda_1, \lambda_2, \lambda_3)$ 。而这只有在条件

$$\begin{vmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ z_1 & z_2 & z_3 \end{vmatrix} = 0 \quad (8.3.1)$$

之下才成立。而式(8.3.1)可写成

$$u_1x_1 + u_2x_2 + u_3x_3 = 0 \quad (8.3.2)$$

式中

$$u_1 = y_2z_3 - y_3z_2 \quad u_2 = y_3z_1 - y_1z_3 \quad u_3 = y_1z_2 - y_2z_1$$

并且 u_1, u_2, u_3 不全为 0(否则 Y 与 Z 将线性相关)。

另一方面, 若点 $[X]$ 不在该直线上, 则 X, Y, Z 必线性独立, 因而行列式(8.3.1)必不为 0, 即 $[X]$ 之坐标不满足式(8.3.2)。

式(8.3.2)即为 P_k^2 中射影直线之一般方程。类似地, 我们可建立射影空间中高次曲线的方程。射影空间中的曲线称为**射影曲线**。

我们现在考查射影空间与仿射空间之间的关系。

假设用以决定射影空间 P_k^n 中点坐标的基底为 $\{e_1, \dots, e_n, e_{n+1}\}$ (张成 V_k^{n+1}), 而产生仿射空间 A_k^n 之基底为 $\{e_1, e_2, \dots, e_n\}$ 。现将 P_k^n 中的点分为两类: 第一类, $x_{n+1} \neq 0$; 第二类, $x_{n+1} = 0$ 。对于第一类的点可写为

$$(x_1, \dots, x_n, x_{n+1}) = \left(\frac{x_1}{x_{n+1}}, \dots, \frac{x_n}{x_{n+1}}, 1 \right)$$

于是在 P_k^n 中的第一类点与 A_k^n 中的点之间可建立如下的一一对应关系:

$$(y_1, \dots, y_n, 1) \in P_k^n \leftrightarrow (y_1, \dots, y_n) \in A_k^n$$

在 A_k^n 中没有一个点对应于 P_k^n 中的第二类点。

由于射影空间 P_k^n 中的第一类点包含了所有对应于仿射空间 A_k^n 中的点, 同时在射影空间中还有第二种点, 因此射影空间 P_k^n 可视为由仿射空间 A_k^n 中的点添加第二种点, 即所谓**虚点**而获得的模型。由于所有虚点均对应于 $x_{n+1} = 0$ 的射影点, 因此这种射影点位于一个射影超平面上, 并且全部这种点对应于仿射空间 A_k^n 的模像是 $(n-1)$ 维线性子空间, 即所谓 $(n-1)$ 维流型。在 A_k^2 中是直线, 在 A_k^3 中是平面, 等等。在射影空间中, 上述的虚点也称为**无穷远点**。

作为例子, 我们指出, 在仿射几何中有两条直线平行的概念, 而在相应的射影几何中任何两条直线均相交于唯一的一点。事实上, 设仿射平面 A_k^2 中直线 l 的方程为

$$a_1x_1 + a_2x_2 + a_3 = 0 \quad a_1, a_2 \text{ 不全为 } 0$$

则所有 A_k^2 中与 l 平行的直线方程均可写成上述形式、其中 a_1, a_2 固定, a_3 任意变化。将坐标齐次化便得到相应的射影平面 P_k^2 中对应的直线方程:

$$a_1 \frac{x_1}{x_3} + a_2 \frac{x_2}{x_3} + a_3 = 0$$

或

$$a_1x_1 + a_2x_2 + a_3x_3 = 0$$

显然, 位于该直线上具有 $x_3 = 0$ 的点只有一个, 其齐次坐标为 $(a_2, -a_1, 0)$ 。由于 $(a_2, -a_1, 0)$ 中不出现 a_3 , 故所有与已知直线 l 平行的直线均含有该点。因此, 在射影平面的仿射模型中, 所有平行的直线均相交于唯一的虚点, 并且每一个虚点均为一直线以及与其平行的所有直线的交点。因此, 在射影平面上, 每两条直线均相交于唯一的一点(实点或虚点)。

同样, 在射影空间 P_k^3 中, 所有平行的平面均相交于唯一的一条虚直线, 并且每条虚直线是一个平行平面簇的交线。

综上所述, P_k^1 的模型是由 A_k^1 添加一个虚点构成。 P_k^2 的模型是由 A_k^2 添加一条虚直线(其上的点均为虚点)而成。 P_k^3 的模型是由 A_k^3 添加一个所谓虚平面(其上的点均为虚点)而成。一般, P_k^n 的模型是由 A_k^n 添加一个 $(n-1)$ 维虚超平面(其上的点均为虚点)而成。

最后, 介绍射影变换的概念。

设 $X = (x_1, \dots, x_n, x_{n+1})$ 与 $rX = (rx_1, \dots, rx_n, rx_{n+1})$ ($r \neq 0$) 为向量空间 V_k^{n+1} 中两个彼此等价的向量。设 L_A 为 V_k^{n+1} 的一个非异线性变换, 该变换的矩阵为 $(n+1) \times (n+1)$ 阶非异矩阵 A 。于是

$$(rX)L_A = (rX)A = (Xr)A = X(rA) = XL_{rA}$$

式中, L_{rA} 是 V_k^{n+1} 上以非异矩阵 rA 为变换矩阵的线性变换。由此可见, 如果我们把成比例的两个非异矩阵 A 与 rA ($r \neq 0$) 视为等价, 相应地, 把两个非异线性变换 L_A 与 L_{rA} 视为等价, 则上式表明: V_k^{n+1} 中等价的线性变换把 V_k^{n+1} 中等价的向量变为等价向量。由此可知, 在 V_k^{n+1} 中的非异线性变换把 m 维子空间仍变成 m 维子空间。特别是, 这一变换把一维子空间仍变为一维子空间, 这就导致了射影空间 P_k^n 中点的变换。

上述讨论指出, V_k^{n+1} 中非异线性变换等价类把 P_k^n 中的点仍变成 P_k^n 中的点, 这种对应还是一一对应。

称 V_k^{n+1} 中非异线性变换等价类为射影空间 P_k^n 的一个射影变换。

在 P_k^n 中, 射影变换的齐次坐标表达式可写成:

$$\begin{aligned} & (\rho x'_1, \dots, \rho x'_n, \rho x'_{n+1}) \\ &= (x_1, \dots, x_n, x_{n+1}) \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & a_{1,n+1} \\ \vdots & \vdots & & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & a_{n,n+1} \\ a_{n+1,1} & a_{n+1,2} & \cdots & a_{n+1,n} & a_{n+1,n+1} \end{bmatrix} = XA \end{aligned}$$

式中, A 为相应的射影变换矩阵。

研究在射影变换下不变的图形性质与不变量, 是射影几何学的课题。射影几何学有丰富的研究内容。

§ 8.4 在有限域上的仿射曲线与射影曲线

设 $F_q(\text{GF}(q))$ 代表 q 阶有限域。又设 \bar{F}_q 代表 F_q 的代数闭包, 即包含 F_q 的最小代数闭域。可以证明:

$$\bar{F}_q = \bigcup_{m=1}^{\infty} F_{q^m}$$

今后, 将以 \bar{F}_q 代替前几节讨论的任意域 K , 并且考虑点的坐标取在 \bar{F}_q 上的仿射平面与射影平面, 分别用 $A^2(\bar{F}_q)$ 及 $P^2(\bar{F}_q)$ 代表。

设 $F(x, y)$ 代表 \bar{F}_q 上的二元多项式, 它在 \bar{F}_q 上的全部根便定义为仿射平面 $A^2(\bar{F}_q)$ 上的一条仿射曲线。多项式 $F(x, y)$ 的次数称为该曲线的阶。

定义 8.4.1 在仿射平面上 $A^2(\bar{F}_q)$ 的点 (a, b) , 若 $a, b \in F_q$, 则称点 (a, b) 为 $A^2(\bar{F}_q)$ 上的有理点。

对于 m 次二元多项式 $F(x, y)$, 它定义了一条 m 阶仿射曲线, 记为 C 。经过齐次化:

$z^m F(x/z, y/z)$ 便得到一个三元 m 次齐次多项式, 记为 $F(x, y, z)$ 。

例如, 考虑三次二元多项式

$$F(x, y) = y^2 - x^2(x + 1)$$

它定义了仿射平面上一条三阶仿射曲线。经过齐次化:

$$z^3 F(x/z, y/z) = F(x, y, z) = y^2 z - x^3 - x^2 z$$

齐次化的过程相当于引进齐次坐标。

一般, 由 m 次二元多项式 $F(x, y)$ 经过齐次化得到的三元 m 次齐次多项式 $F(x, y, z)$, 便定义了射影平面上的一条 m 阶代数曲线, 称为 m 阶射影曲线。它是由仿射曲线 C 上的点添加某些无穷远点所构成的射影平面上的 m 阶代数曲线。

反过来, 每一条 m 阶射影曲线 $F(x, y, z) = 0$ 也可通过非齐次化手续化为 m 阶仿射曲线 $F(x, y, 1) = 0$ 。例如五阶射影曲线

$$F(x, y, z) = x^5 + y^5 - z^5 = 0$$

可化为

$$F(x, y) = x^5 + y^5 - 1 = 0$$

这相当于由原来的 m 阶射影曲线去掉某些无穷远点所产生的 m 阶仿射曲线。

定义 8.4.2 在射影平面 $P^2(\bar{F}_q)$ 上的点 (a, b, c) , 当 $c \neq 0$ 时, $a/c, b/c \in F_q$; 当 $c = 0$ 时(此时, a, b 中至少有一不为 0), $a/b \in F_q (b \neq 0)$ 或 $b/a \in F_q (a \neq 0)$; 则称点 (a, b, c) 为 $P^2(\bar{F}_q)$ 上的有理点。

曲线上这些有理点有时可枚举出来。

例 8.1 在 $P^2(\bar{F}_4)$ 上找出曲线

$$y^2 z + yz^2 = x^3 + x^2 z + xz^2 + z^3 \quad (8.4.1)$$

上的全部有理点。

表 8-1 式(8.4.1)所示曲线上的全都有理点

| | Q | P_1 | P_2 | P_3 | P_4 | P_5 | P_6 | P_7 | P_8 |
|-----|---|----------|---------|-------|-------|----------|----------|----------|---------|
| x | 0 | 0 | 0 | 1 | 1 | α | α | β | β |
| y | 1 | α | β | 0 | 1 | α | β | α | β |
| z | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

熟知, $F_4 = \{0, 1, \alpha, \beta\}$, $\beta = \alpha + 1 = \alpha^2$, 经过计算, 该曲线上的全部有理点共有 9 个, 见表 8-1。

点 $Q(0, 1, 0)$ 是无穷远点。式(8.4.1)所确定的曲线是亏格为 1 的代数曲线, 称为椭圆曲线。椭圆曲线是编码理论中要讨论的重要曲线, 以后我们会进一步解释。■

在实践上, 真正算出给定曲线的全部有理点, 或退一步, 算出这些有理点的个数, 均是相当困难的工作。

如果多项式 $F(x, y)$ 在 F_q 的任何扩域 F_{q^m} 上均无异于常数的因式, 则称相应的曲线为不可约曲线(既约曲线)。判断一条曲线的不可约性, 也是一件相当困难的事情。

§ 8.5 RS 码与 Goppa 码

为了讨论一般的代数几何码, 我们首先来回顾一下经典的 RS 码及 Goppa 码。

正如定义 7.3.1 所述, RS 码是 F_q ($q \neq 2$) 上码长为 $n=q-1$ 的本原 BCH 码, 该码的生成多项式为

$$g(x) = \prod_{i=1}^{d-1} (x - \alpha^i)$$

式中, α 为 F_q 的本原域元素。RS 码是码长为 $n (= q-1)$, 信息位数为 $n-d+1$, 最小距离为 d 的极大最小距离可分码 $[n, n-d+1, d]$, 亦即 MDS 码。

像所有线性码一样, 对于 $[n, n-d+1, d]$ RS 码增加一个全校验位后, 便成为扩展 RS 码。它是码长为 $n+1 (= q)$, 信息位数仍为 $n-d+1$ 的 $[n+1, n-d+1]$ 码。由定理 7.3.1, 这个码的最小距离为 $d+1$, 因而扩展 RS 码仍为 MDS 码。

我们现在遵循里德与索洛蒙原来的编码方法, 也就是用 M-S 多项式构造 RS 码的方法来建立扩展 RS 码。

为了方便, 我们取 $n=q$ 。 F_q 中的元素记成

$$\alpha_i = \alpha^i \quad 0 \leq i \leq q-2$$

式中, α 为 F_q 的本原域元素。令

$$L = \{f \in F_q[x] \mid \partial \circ f \leq k-1\} \quad k < n$$

我们定义一个码 C 为

$$C = \{(f(\alpha_0), f(\alpha_1), \dots, f(0)) \mid f \in L\} \quad (8.5.1)$$

正如 § 7.3 中所指出的码 C 是用频域方法编出的扩展 RS 码, 码长 $n=q$, 信息位数为 k , 即 L (作为线性空间) 的维数。又因 L 中的多项式次数 $\leq k-1$, 故不可能多于 $k-1$ 个零点, 从而码 C 的最小距离 $d_m \geq n-k+1$ 。另一方面, 推论 3.2.1 指出: 任何线性码的最小距离至多为该码的校验位数加 1, 即

$$d_m \leq n-k+1$$

因此, $d_m = n-k+1$ 。这表明码 C 是 $[n, k, n-k+1]$ 线性码, 亦即 MDS 码。

不难看出, 码 C 等价于扩展 RS 码。事实上, 设 $f(x) = \sum_{j=0}^{k-1} a_j x^j$, 且令 $c_i = f(\alpha_i)$, $0 \leq i \leq q-2$ 。于是, 如果 $1 \leq l \leq q-k-1$, 则

$$\sum_{i=0}^{q-2} c_i (\alpha^l)^i = \sum_{i=0}^{q-2} (\alpha^l)^i \sum_{j=0}^{k-1} a_j (\alpha^i)^j = \sum_{j=0}^{k-1} a_j \sum_{i=0}^{q-2} (\alpha^{i+j})^i$$

注意到

$$x^{q-1} - 1 = (x-1) \sum_{i=0}^{q-2} x^i$$

从而在 F_q 中每个元素 $\alpha \neq 0, 1$, 均满足 $\sum_{i=0}^{q-2} \alpha^i = 0$ 。由于 $1 \leq l+j \leq q-k-1+k-1 = q-2$, 故 $\sum_{i=0}^{q-2} (\alpha^{i+j})^i = 0$ 。因此

$$\sum_{i=0}^{q-2} c_i (\alpha^l)^i = 0$$

这表明 $c = (c_0, c_1, \dots, c_{q-2})$ 是最小距离为 $q-k$ 的 RS 码中的码字。将此码再添加 1 个全校验位, 即为扩展 RS 码。RS 码在磁盘与光盘的纠错技术中很有用处。

为了进一步讨论代数几何码, 我们将 RS 码的第二种定义再加以推广。

设 $\xi_i (\neq 0) \in F_q (i=1, \dots, n)$, $\{\alpha_1, \dots, \alpha_n\} \subset \{1, \alpha, \dots, \alpha^{q-2}\}$, 且 $\alpha_1, \dots, \alpha_n$ 彼此不同, 于是可定义一个码为

$$C = \{(\xi_1 f(\alpha_1), \dots, \xi_n f(\alpha_n)) \mid f \in L\} \quad (8.5.2)$$

这个码的构成相当于如下的线性映射:

$$\begin{array}{ccc} L & \rightarrow & (F_q)^n \\ \downarrow & & \\ f & \rightarrow & (\xi_1 f(\alpha_1), \dots, \xi_n f(\alpha_n)) \end{array}$$

显然, C 仍为最小距离为 $n-k+1$ 的线性码 $[n, k, n-k+1]$, 即为 MDS 码。

为增大码长, 上面所构成的线性码还可在 F_q 的扩域 F_{q^m} 上考虑。设

$$L = \{f \in F_{q^m}[x] \mid \deg f \leq k-1\} \quad k < n$$

这里 $\deg f = \partial \circ f$ 是多项式 f 的次数。再设 $\xi_i (\neq 0) \in F_{q^m} (i=1, 2, \dots, n)$, $\xi = (\xi_1, \dots, \xi_n)$, 并且取 F_{q^m} 中 n 个不同元素 $\alpha_1, \dots, \alpha_n$, 置 $\alpha = (\alpha_1, \dots, \alpha_n)$, 于是可构造一个线性码

$$C = \{(\xi_1 f(\alpha_1), \dots, \xi_n f(\alpha_n)) \mid f \in L\}$$

码 C 仍为 $[n, k, n-k+1]$ 线性码, 即为 MDS 码。由定义 7.3.2 可知该码就是广义 RS 码, 记为 $\text{GRS}_k(\alpha, \xi)$ 。

现在我们再来考察 Goppa 码。

设 F_q 为基域, $g(z) \in F_{q^m}[z]$, $L = \{\alpha_1, \dots, \alpha_n\}$, $\alpha_i \in F_{q^m} (i=1, \dots, n)$, $\alpha_i \neq \alpha_j (i \neq j)$ 。设 $g(z)$ 的根不在 L 中, 即 $g(\alpha_i) \neq 0 (i=1, \dots, n)$, 则由定义 7.9.1 可知:

$$C_G = \left\{ (c_1, \dots, c_n) \in (F_q)^n \mid \sum_{i=1}^n \frac{c_i}{z - \alpha_i} \equiv 0 \pmod{g(z)} \right\}$$

就是 § 7.9 中所讨论的经典 Goppa 码 $\Gamma(L, g)$, 其中 $g(z)$ 称为 Goppa 多项式。

为使导入代数曲线上定义的线性码更为自然, 我们将 Goppa 码的定义进一步推广。

假设 $f(z) = \varphi(z)/\psi(z)$ 是 F_{q^m} 上的有理函数, 即 $\varphi(z), \psi(z) \in F_{q^m}[z]$ 。现在考虑 F_{q^m} 上所有具备下列性质的有理函数全体:

1° $f(z)$ 以 $g(z)$ 的所有零点(根)为零点($g(z)$ 即为前述定义中的 F_{q^m} 上的 Goppa 多项式), 且 $f(z)$ 在这些零点上的级至少为 $g(z)$ 在这些零点上相应的级;

2° $f(z)$ 除了在 $L = \{\alpha_1, \dots, \alpha_n\}$ 中的某些点上可能具有一级极点外别无其它极点。

这里 $f(z)$ 的极点、极点的级, 以及下面提到的 $f(z)$ 在极点 α_i 上的留数 $\text{Res}_{\alpha_i} f$ 与普通复变函数论中的定义类似, 此处不再重复。

当 $f(z)$ 取遍具有上述性质 1°, 2° 的全部 F_{q^m} 上的有理函数时(注意, 这些有理函数的全体构成一个线性空间), 在 F_{q^m} 上便可定义一个线性码, 它由形如

$$(\text{Res}_{\alpha_1} f, \dots, \text{Res}_{\alpha_n} f) \quad (8.5.3)$$

的 n 重构成。

因为 Goppa 码是定义在 F_q 上的线性码, 显然, Goppa 码是上述定义的码的子域 ($F_q \subset F_{q^m}$) 子码。在 Goppa 码中

$$R_c(z) = \sum_{i=1}^n \frac{c_i}{z - \alpha_i}$$

即为 F_{q^m} 上的有理函数, 它满足条件 1°, 2°, 并且

$$c_i = \text{Res}_{\alpha_i} R_c(z) \quad i = 1, 2, \dots, n$$

为了考察 Goppa 码与广义 RS 码之间的关系，我们首先需找出 Goppa 码 $\Gamma(L, g)$ 一致校验矩阵的另一种表达式。设 $g(z) = \sum_{i=0}^t g_i z^i$ 。于是

$$\varphi(z) = \frac{g(z) - g(x)}{z - x} = \sum_{0 \leq i+j \leq t-1} g_{i+j+1} x^j z^i$$

是关于变量 z 的一个次数 $< t$ 的多项式（对于任何 x ）。由于

$$\frac{1}{z - \alpha_i} \equiv \frac{-1}{g(\alpha_i)} \left[\frac{g(z) - g(\alpha_i)}{z - \alpha_i} \right] \pmod{g(z)}$$

从而依据 $(z - x)\varphi(z) = g(z) - g(x) \equiv -g(x) \pmod{g(z)}$ ，再令 $\xi_i = 1/g(\alpha_i)$ ，关系式

$$\sum_{i=1}^n \frac{c_i}{z - \alpha_i} \equiv 0 \pmod{g(z)}$$

可改写成

$$\sum_{i=1}^n c_i \xi_i \sum_{0 \leq i+j \leq t-1} g_{i+j+1} (\alpha_i)^j z^i = 0$$

对于 $0 \leq i \leq t-1$ ，上式中 z^i 的系数均为 0。我们看出， $\mathbf{c} = (c_1, \dots, c_n)$ 必与下列矩阵之每一行向量内积为 0：

$$\mathbf{H}_4 = \begin{bmatrix} \xi_1 g_t & \cdots & \xi_n g_t \\ \xi_1 (g_{t-1} + g_t \alpha_1) & \cdots & \xi_n (g_{t-1} + g_t \alpha_n) \\ \vdots & & \vdots \\ \xi_1 (g_1 + g_2 \alpha_1 + \cdots + g_t \alpha_1^{t-1}) & \cdots & \xi_n (g_1 + g_2 \alpha_n + \cdots + g_t \alpha_n^{t-1}) \end{bmatrix} \quad (8.5.4)$$

式中，第一行对应于上述和式中 z^{t-1} 之系数；第二行对应于 z^{t-2} 的系数，等等；最后，第 t 行对应于 z^0 的系数。与式(7.9.9)相比可知：若这里的 $t=r$ ，则上面的 \mathbf{H}_4 矩阵就是 Goppa 码的 \mathbf{H}_2 矩阵。

如所周知，对上述矩阵做初等行变换并不影响 \mathbf{c} 与该矩阵行的正交性。由此便得到 Goppa 码 $\Gamma(L, g)$ 的一致校验矩阵为

$$\mathbf{H}_5 = \begin{bmatrix} \xi_1 & \xi_2 & \cdots & \xi_n \\ \xi_1 \alpha_1 & \xi_2 \alpha_2 & \cdots & \xi_n \alpha_n \\ \vdots & \vdots & & \vdots \\ \xi_1 \alpha_1^{t-1} & \xi_2 \alpha_2^{t-1} & \cdots & \xi_n \alpha_n^{t-1} \end{bmatrix} \quad (8.5.5)$$

现在我们再回过来考虑广义 RS 码 $\text{GRS}_k(\alpha, \xi)$ ，亦即

$$\mathcal{C} = \{(\xi_1 f(\alpha_1), \dots, \xi_n f(\alpha_n)) \mid f \in L\}$$

的生成矩阵。在 $\text{GRS}_k(\alpha, \xi)$ 中取 $k=t$ ，且设 $f(z) = \sum_{i=0}^{t-1} f_i z^i$ 。于是 \mathcal{C} 中的码字可写成

$$(\xi_1 \sum_{i=0}^{t-1} f_i \alpha_1^i, \dots, \xi_n \sum_{i=0}^{t-1} f_i \alpha_n^i) \quad (8.5.6)$$

$$(f_0, f_1, \dots, f_{t-1}) \begin{bmatrix} \xi_1 & \xi_2 & \cdots & \xi_n \\ \xi_1 \alpha_1 & \xi_2 \alpha_2 & \cdots & \xi_n \alpha_n \\ \vdots & \vdots & & \vdots \\ \xi_1 \alpha_1^{t-1} & \xi_2 \alpha_2^{t-1} & \cdots & \xi_n \alpha_n^{t-1} \end{bmatrix}$$

这表明 Goppa 码 $\Gamma(L, g)$ 的一致校验矩阵恰为 $\text{GRS}_k(\alpha, \xi)$ 的生成矩阵。这正如 § 7.9 中所

指出的, Goppa 码恰为一个 GRS 码的对偶码的子域子码。

§ 8.6 代数几何码的构成

在上一节构造 RS 码及广义 RS 码中所使用的多项式 $f(z)$ 可写成

$$z^\delta + a_{\delta-1}z^{\delta-1} + \cdots + a_1z + a_0 \quad \delta \leq k-1$$

或引进齐次坐标 $z \rightarrow x/y$, 上式可改写成

$$\frac{x^\delta + a_{\delta-1}x^{\delta-1}y + \cdots + a_1xy^{\delta-1} + a_0y^\delta}{y^\delta} \quad \delta \leq k-1 \quad (8.6.1)$$

考虑射影直线 $X = P^1(\bar{F}_q)$, 以下有时简写为 P^1 。设 $Q = (1, 0)$ 代表 P^1 上的无穷远点。设 $L((k-1) \cdot Q)$ 代表对于点 Q 至多为 $k-1$ 级极点的形如式(8.6.1)的有理函数全体。

在 RS 码定义中的 $\alpha_1, \alpha_2, \dots, \alpha_n$ 对应于 $X = P^1(\bar{F}_q)$ 上的 F_q 上的有理点 $P_1 = (\alpha_1, 1), P_2 = (\alpha_2, 1), \dots, P_n = (\alpha_n, 1)$ 。在 $X = P^1(\bar{F}_q)$ 上的全部有理点共有 $q+1$ 个: $(0, 1), (1, 1), (\alpha, 1), (\alpha^2, 1), \dots, (\alpha^{q-2}, 1), (1, 0)$ 。因此, 从几何的观点来看, RS 码相当于从 $L((k-1) \cdot Q)$ 到 $(F_q)^n$ 的一个映射, 在这一映射之下, 将 $L((k-1) \cdot \infty)$ 上的每一个对于点 $Q = (1, 0)$, 至多为 $(k-1)$ 级极点的形如式(8.6.1)的有理函数, 取其在 $X = P^1(\bar{F}_q)$ 上 n 个指定点 P_1, \dots, P_n 的值, 便得到 $(F_q)^n$ 中的一个向量 $(f(P_1), \dots, f(P_n))$, 即为一个码字。图 8-4 给出了这种码的几何解释。

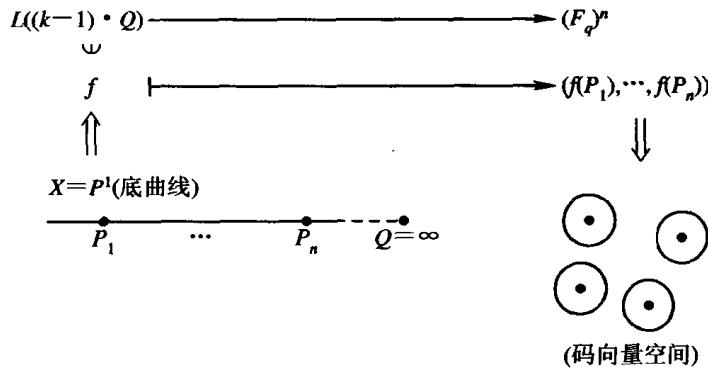


图 8-4 RS 码的几何解释

从上述 RS 码的几何解释中, 我们看到, 这种码是由在射影直线 $X = P^1$ 上的有理函数及有理点 P_1, \dots, P_n 所确定的。称 $X = P^1$ 为底曲线。

激发起构造代数几何码想法的出发点就是将射影直线 $X = P^1$ 由射影平面 P^2 上的代数曲线来取代, 并且考虑以这条代数曲线上的指定点 Q_1, \dots, Q_l 分别至多为 m_1, \dots, m_l 级极点的全部有理函数, 记为 $L(G)$, $G = \sum_{j=1}^l m_j Q_j$ 。在这条曲线上取定 n 个 F_q 上的有理点 P_1, \dots, P_n , 从而便构造出一个新的线性码

$$C = \{(f(P_1), \dots, f(P_n)) \mid f \in L(G)\}$$

如图 8-5 所示。在这里, 前述代数曲线也称为底曲线。

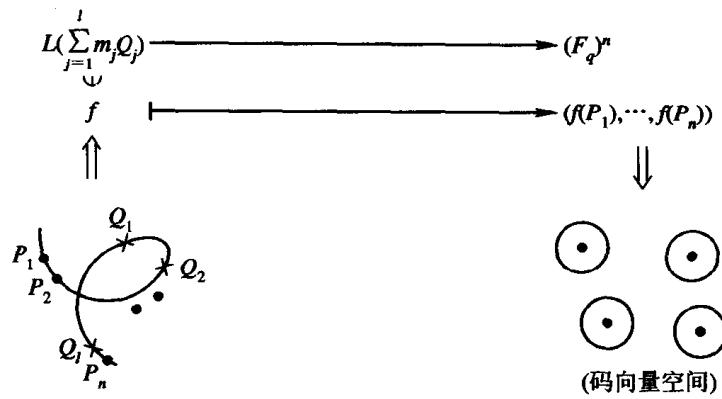


图 8-5 用 P^2 上的代数曲线构造的线性码

例 8.2 考察例 8.1 中所示之椭圆曲线

$$y^2z + yz^2 = x^3 + x^2z + xz^2 + z^3$$

已知该曲线上共有 9 个有理点: Q, P_1, P_2, \dots, P_8 , 如表 8-1 所示。

现令 $G=4Q, L(G)=L(4Q)=\{f|f$ 为系数取自 F_4 上、且点 Q 至多为其 4 级极点的 X_1 上的有理函数}, 其中 X_1 代表椭圆曲线

$$X_1 = \{(x, y, z) \in P^2(\bar{F}_4) | y^2z + yz^2 = x^3 + x^2z + xz^2 + z^3\}$$

不难看出, $L(G)$ 为四维线性空间, 它的一组基底是

$$\psi_1 = 1 \quad \psi_2 = \frac{x}{z} \quad \psi_3 = \frac{y}{z} \quad \psi_4 = \frac{x^2}{z^2}$$

事实上,

$$\psi_2 = \frac{x}{z} = \frac{x^3}{zx^2} = \frac{y^2z + yz^2 + x^2z + xz^2 + z^3}{zx^2} = \frac{y^2 + yz + x^2 + xz + z^2}{x^2}$$

因此 ψ_2 以点 $Q=(0, 1, 0)$ 为二级极点。进一步,

$$\psi_4 = \left(\frac{x}{z}\right)^2 = \frac{y^4 + y^2z^2 + x^4 + x^2z^2 + z^4}{x^4}$$

这表示 ψ_4 以点 $Q=(0, 1, 0)$ 为四级极点。

考虑如下的线性映射 φ_L :

$$\begin{aligned} L(G) &\rightarrow (F_4)^8 \\ \Downarrow \\ \varphi_L: \quad f &\rightarrow (f(P_1), \dots, f(P_8)) \end{aligned}$$

为了找出所构成的码

$$C = \{(f(P_1), \dots, f(P_8)) | f = \sum_{i=1}^4 f_i \psi_i, f_i \in F_4\} \quad (8.6.2)$$

的生成多项式, 只需注意

$$\begin{aligned} c = (f(P_1), \dots, f(P_8)) &= \left(\sum_{i=1}^4 f_i \psi_i(P_1), \dots, \sum_{i=1}^4 f_i \psi_i(P_8) \right) \\ &= (f_1, f_2, f_3, f_4) \begin{bmatrix} \psi_1(P_1) & \psi_1(P_2) & \cdots & \psi_1(P_8) \\ \psi_2(P_1) & \psi_2(P_2) & \cdots & \psi_2(P_8) \\ \psi_3(P_1) & \psi_3(P_2) & \cdots & \psi_3(P_8) \\ \psi_4(P_1) & \psi_4(P_2) & \cdots & \psi_4(P_8) \end{bmatrix} \quad (8.6.3) \end{aligned}$$

由此可见码 C 的生成矩阵为

$$\begin{bmatrix} \psi_1(P_1) & \psi_1(P_2) & \cdots & \psi_1(P_8) \\ \psi_2(P_1) & \psi_2(P_2) & \cdots & \psi_2(P_8) \\ \psi_3(P_1) & \psi_3(P_2) & \cdots & \psi_3(P_8) \\ \psi_4(P_1) & \psi_4(P_2) & \cdots & \psi_4(P_8) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & \alpha & \alpha & \beta & \beta \\ \alpha & \beta & 0 & 1 & \alpha & \beta & \alpha & \beta \\ 0 & 0 & 1 & 1 & \beta & \beta & \alpha & \alpha \end{bmatrix} \quad (8.6.4)$$

进一步, 如果把码 C 的全部 $4^3=64$ 个码字找出来, 我们便可发现码 C 有最小距离为 4。因此, 码 C 是一个四进制的(8, 4, 4)线性码。■

这一例子说明, 我们确实能够利用代数曲线来构造线性码。这使我们为构造各种线性码找到了新的源泉, 从而使编码理论的研究迈入了一个全新的阶段。

今后, 用 $\text{Rat}(X)$ 代表(F_q 上的)不可约代数曲线 X 的有理函数全体所构成的有理函数域。 $\text{Rat}(X)$ 中的元素在射影曲线的情形下可表为两个同次多项式之比:

$$f = \frac{A(x, y, z)}{B(x, y, z)}$$

并且如果另有 $f_1 = A_1(x, y, z)/B_1(x, y, z)$ 满足:

$$f - f_1 \equiv 0 \pmod{X}$$

则视 f 与 f_1 为等同。注意, 此处 X 代表一个代数曲线。

例如, 设 X 代表 F_2 上的不可约代数曲线

$$X = F(x, y, z) = y^2z + yz^2 + x^3 + x^2z$$

我们来计算 $f = (y^2 + yz)/xz$ 于点 $P = (0, 0, 1)$ 之值。注意到

$$\frac{y^2 + yz}{xz} - \frac{x^2 + xz}{z^2} \equiv 0 \pmod{F(x, y, z)}$$

故 $f = (y^2 + yz)/xz \equiv x(x+z)/z^2$ 于点 $P = (0, 0, 1)$ 之值 $f(P) = 0$ 。

对于 $\text{Rat}(X)$ 中之元 f , 若它的零点为 $\hat{P}_1, \dots, \hat{P}_r$, 相应的级数为 n_1, \dots, n_r , 而它的极点为 q_1, \dots, q_s , 相应的级数为 m_1, \dots, m_s , 则以 $\text{div}(f)$ 代表形式和

$$(f) \triangleq \sum_{i=1}^r n_i \hat{P}_i - \sum_{j=1}^s m_j q_j \quad (8.6.5)$$

并称 (f) 为 f 的除子(divisor)。

例如, 在例 8.2 中, 对于 $\psi_2 = x/z$, 我们有

$$\text{div}(\psi_2) = P_1 + P_2 - 2Q$$

§ 8.7 代数曲线中的一些重要概念

为了进一步介绍代数几何码, 本节将引入代数曲线理论中的一些重要概念。需要说明的是, 为了要严格地定义这些概念, 必须深入介绍代数几何理论的进一步知识。限于篇幅及读者对象, 我们在叙述这些概念时将不追求严格, 并尽量以实例说明。为了易于理解, 读者可将本节中讨论的有限域 F_q 的代数闭包 \bar{F}_q 想象成普通的复数域 C 。

一、局部环与局部参数

在本节中用 K 代表 F_q 的代数闭包 \bar{F}_q 。设 X 是一个仿射曲线或射影曲线, P 是曲线 X

上的一点, $U(P)$ 是 P 点的一个邻域(为定义邻域概念, 需在射影空间 P^n 或仿射空间 A^n 中引进所谓 Zariski 拓扑, 此处从略)。在 $U(P)$ 上定义的函数 $\varphi = f/g$ 称为于点 P 正则, 如果 $g(P) \neq 0$, 其中 f, g 均为具有相同次数的齐次多项式。两个在点 P 正则的函数 φ_1, φ_2 称为是等价的, 如果 φ_1, φ_2 在点 P 的某一邻域内相等。用 $O(U)$ 代表在 $U(P)$ 的每一点均为正则的函数全体, 显然 $O(U)$ 构成一个环。将 $O(U)$ 中的函数按上述等价概念分类, 又得到一个环, 记为 $O_P(X)$ 。

定义 8.7.1 $O(U)$ 中正则函数等价类的集合所构成的环 $O_P(X)$ 称为在 X 上点 P 的局部环(local ring)。

这里定义的局部环概念正是近世代数中的局部环概念。事实上, 不难证明 $O_P(X)$ 含有一个在点 P 等于 0 的函数类的集合, 记为 $m_P(X)$, 它是 $O_P(X)$ 中唯一的一个极大理想。作为练习, 建议读者证明这一结论。

在 A^2 中, 若一曲线由 $F(x, y)=0$ 定义, $P(a, b)$ 是该曲线上一点。若偏导数 $F_x(P)$ 与 $F_y(P)$ 至少有一个不为 0, 则称点 P 为该曲线的非奇异点。这表明该曲线在点 P 有切线 $F_x(P)(x-b)+F_y(P)(y-b)=0$ 。若曲线上每一点均为非奇异点, 则称该曲线是非奇异的或光滑的。今后, 我们总假定所讨论的曲线是光滑的。在一般情形下, 如果 m_P/m_P^2 (作为 K 上的向量空间看待)具有维数 1, 则称相应的代数曲线在点 P 光滑。

因为 m_P/m_P^2 之维数为 1, 故该空间有一生成元 t 。我们同样也把 t 作为对应于 m_P 中之元。于是可断定, $O_P(X)$ 中的每一元素 z 均可表为 $z=ut^m$ 的形式, 其中 u 是 $O_P(X)$ 中的单位(即乘法逆元), m 是某一整数。具备这一性质的元素 t , 作为函数看待, 称之为在点 P 的局部参数。如果 $m>0$, 则点 P 是 z 的 m 级零点; 如果 $m<0$, 则点 P 是 z 的 m 级极点。我们引入记号

$$m = \text{Ord}_P(z) = v_P(z)$$

对于一般在 X 上定义的有理函数 $\varphi=f/g$, 我们可定义

$$v_P(f/g) = v_P(f) - v_P(g) \quad (8.7.1)$$

例 8.3 假设曲线 X 是 A^2 中之圆 $x^2+y^2-1=0$, 且设 $P=(1, 0)$, 考虑 $z=z(x, y)=1-x$, 显然 $z(P)=0$, 因此 $z \in m_P$ 。由于直线 $y=0$ 与 X 之交点在点 P 的重数为 1, 故 y 即为局部参数。进一步, 在 X 上我们有

$$z = 1 - x = \frac{y^2}{1+x}$$

而 $(1+x)^{-1}$ 是 $U_P(X)$ 的单位。因此, $v_P(z)=2$ 。

如果在 P^2 中考虑上述例子, 此时曲线 X 为 $x^2+y^2-z^2=0$, $P=(1, 0, 1)$, 我们考虑 $U_P(X)$ 中的函数 $(z-x)/z$, 它也是 m_P 中之元素。于是在点 P 的局部参数为 $t=y/z$ 。我们有

$$\frac{z-x}{z} = \frac{z^2-x^2}{z(z+x)} = \frac{y^2 z}{z^2(z+x)} = t^2 \frac{z}{z+x}$$

式中, $z/(z+x)$ 是局部环 $O_P(X)$ 中之单位。因此, $v_P((z-x)/z)=2$ 。 ■

二、除子

除子(divisor)的概念在代数几何中占有重要地位。

以下总假定 X 是 K 上的光滑射影曲线。

定义 8.7.2 在曲线 X 上的有限个点所作的形式整系数线性组合

$$D = \sum_{P \in X} n_P P \quad n_P \text{ 是整数} \quad (8.7.2)$$

即除去 X 上的有限个 $n_P=0$ 的点 P 外, 称之为除子。如果式(8.7.2)中所有 $n_P \geq 0$, 则称除子 D 是有效的, 记为 $D > 0$ 。最后, 在式(8.7.2)中称 $\sum n_P$ 为除子 D 的次数, 记为

$$\deg(D) = \sum n_P \quad (8.7.3)$$

两个除子 $D_1 = \sum n_i P_i$ 及 $D_2 = \sum m_i P_i$ 可进行加减运算:

$$D_1 \pm D_2 = \sum (n_i \pm m_i) P_i$$

例如, 若 $D_1 = P_1 + 2P_2 - P_3$, $D_2 = P_1 - 3P_3 + P_4$, 则

$$D_1 - D_2 = 2P_2 + 2P_3 - P_4$$

因此, 按上述的形式加法运算, X 上定义的全部除子构成一个加法群(X 上的自由 Abel 群)。

定义 8.7.3 X 上全部除子所构成的加法群称为曲线 X 的除子群, 记为 $\text{div}(X)$ 。

对于 X 上的有理函数 f , 在 § 8.6 中已经定义过 f 的除子 (f) 。使用本节中的记号 $v_P(\cdot)$, 我们可以重新定义 (f) 如下。

定义 8.7.4 对于 X 上的有理函数 f , f 不恒为 0, 定义 f 的除子为

$$(f) = \sum_{P \in X} v_P(f) P$$

对于 $f=0$, 我们规定:

$$(f) = \sum_{P \in X} 0P$$

在某种意义上说, 函数 f 的除子可看作是一种登记表, 它告诉我们这个数在曲线 X 上有哪些零点与极点以及它们相应的级数。因为 f 是分子与分母次数相同的有理函数, 又因 K 是代数闭域, 所以不难想象, f 具有相同数目的零点与极点(连同其级数一起计算)。因此, 直观上可看出下述定理成立。

定理 8.7.1(Bezout 定理) 对于 X 上的任一有理函数 f , 恒有 $\deg(f)=0$ 。

现在, 我们在除子群 $\text{div}(X)$ 上建立一种等价关系。设 $D, D' \in \text{div}(X)$, 如果对于某一有理函数 f , 有 $D - D' = (f)$, 则称除子 D 与 D' 线性等价, 记为 $D \equiv D'$ 。不难证明, 这确实是一个等价关系, 即满足:

1. $D \equiv D'$;
2. $D \equiv D' \Rightarrow D' \equiv D$;
3. $D \equiv D', D' \equiv D'' \Rightarrow D \equiv D''$ 。

定义 8.7.5 称商群 $\text{Div}(X)/\{(f) \mid f \in \text{Rat}(X)\}$ 为曲线 X 的 Picard 群, 记为

$$\text{Pic}(X) = \text{Div}(X)/\{(f) \mid f \in \text{Rat}(X)\}$$

Picard 群也叫除子类群。

Picard 群在代数曲线理论中起着重要的作用。

类似于 § 8.5 中用有理函数所构成的线性空间定义推广了的 Goppa 码的情形一样, 我们可在曲线 X 上做同样的事情。

设 D 为曲线 X 的一个除子。考察集合

$$\mathcal{L}(D) = \{f | f \in \text{Rat}(X), f \neq 0, (f) + D > 0\} \cup \{0\} \quad (8.7.4)$$

式中 $>$ 表示所有系数 ≥ 0 , 如果设 $D = \sum_{i=1}^r n_i P_i - \sum_{j=1}^s m_j Q_j$, 所有 n_i, m_j 均 ≥ 0 , 于是不难看出, $\mathcal{L}(D)$ 中的函数 f , 它或者为0, 或者是以诸 Q_j ($1 \leq j \leq s$)至少为 m_j 级零点, 并且除去可能以诸 P_i ($1 \leq i \leq r$)至多为 n_i 级极点外别无其它极点的有理函数。很明显, $\mathcal{L}(D)$ 构成一个域 K 上的线性空间。

定义 8.7.6 设 $D \in \text{div}(X)$, 称

$$\mathcal{L}(D) = \{f | f \in \text{Rat}(X), f \neq 0, (f + D) > 0\} \cup \{0\}$$

为由曲线 X 的除子 D 所决定的线性空间。

可以证明, $\mathcal{L}(D)$ 必为有限维线性空间, 以 $\dim_K \mathcal{L}(D)$ 代表 $\mathcal{L}(D)$ 的维数。我们可以有下面的定理。

定理 8.7.2 对于线性空间 $\mathcal{L}(D)$, 下述论断成立:

- 1° 若 $\deg(D) < 0$, 则 $\mathcal{L}(D) = \{0\}$;
- 2° $\dim_K \mathcal{L}(D) \leq 1 + \deg(D)$ 。

今后, 用 $l(D)$ 代表 $\dim_K \mathcal{L}(D)$, $l(D)$ 与下一节要叙述的一个重要定理有直接的关系。

三、在曲线上的微分

考察 A^2 上由方程 $F(x, y) = 0$ 定义的曲线 X , 且设 $P = (a, b)$ 为 X 上一点。曲线 X 在点 P 的切线 T_P 由 $d_P F = 0$ 定义, 其中 $d_P F$ 由下式定义:

$$d_P F = F_x(a, b)(x - a) + F_y(a, b)(y - b) \quad (8.7.5)$$

给定 $P \in X$, 则映射 d_P 把每一个函数 F 变成一个线性函数 $d_P(F)$ 。同样, 给定函数 F , 则映射 $d_P F$ 把每一点 $P \in X$ 皆变成一个线性函数 $d_P F$ 。

称线性算子 $d_P F$ 为 F 在点 P 的微分。若考察每一点 $P \in X$ 的微分, 则记为 dF 。它具有下述性质:

- 1° $d(F+G) = dF + dG$;
- 2° $da = 0$, $a \in K$;
- 3° $d(FG) = FdG + GdF$ 。

定义 8.7.7 设 X 是闭域 K 上的仿射曲线。在 X 上定义 $\Omega(X)$ 为满足下述条件的集合:

$\Omega(X)$ 中的每一元素在每一点 $P \in X$ 的邻域 $U(P)$ 内均可表示为 $\sum_{i=1}^m f_i dg_i$, 式中, f_i, g_i

均在 $U(P)$ 正则; 并且 dg_i 满足上述条件1°, 2°, 3°。

称 $\Omega(X)$ 中的元素为正则微分形式。

利用正则微分形式容易建立有理微分形式的概念。

定义 8.7.8 考虑序偶 (U, ω) 与 (V, η) , 设 ω, η 分别是邻域 U, V 上的正则微分形式。定义等价关系:

$$(U, \omega) \sim (V, \eta) \Leftrightarrow \omega = \eta \text{ 在 } U \cap V$$

按这一等价关系决定的等价类称为有理微分形式。

对于有理微分形式, 我们有: $d(g/h) = (hdg - gdh)/h^2$, 其中 $g, h \neq 0$ 。

从现在起，我们称 X 上的有理微分形式为 **微分**。仍用 $\Omega(X)$ 代表这种微分的全体所构成的空间。可以证明， $\Omega(X)$ 的维数为 1。因此，在具有局部参数 t 的点 P 的一个邻域内，微分 ω 可以表示成 $\omega = f dt$ 的形式，式中 f 是有理函数。

定义 8.7.9 微分 ω 的除子定义为

$$(\omega) = \sum_{P \in X} V_P(f_P) P \quad (8.7.6)$$

此处 $\omega = f_P dt_P$ 是 ω 的局部表示。

可以证明除子 (ω) 的定义依赖于局部参数的选取，并且上面的和式仅有有限项系数不为 0。

称 $W = (\omega)$ 为 **标准除子**。可以证明，这种标准除子构成 Picard 群 $\text{Pic}(X)$ 中的一个类。因此按定义 8.7.6，可以定义线性空间 $\mathcal{L}(W)$ 及其维数 $l(W)$ 。

定义 8.7.10 设 X 为闭域 K 上的光滑射影曲线。称 $g = l(W)$ 为曲线 X 的 **亏格** (genus)。

亏格是代数几何理论的重要概念。亏格是代数几何中内蕴的概念，而不是外在强加的。直观上说，这种流形在奇点附近形成若干个“洞”，这些“洞”的数目就是该流形的亏格。例如，拓扑学中已经指出，任何可定向的二维紧流形均同胚于一个具有若干环柄的球面(图 8-6)。这些环柄的个数 g ，就是一个拓扑不变量，就是亏格。

下面的定理给出了计算曲线亏格的算法。

定理 8.7.3(Plücker 公式) 若 X 是 P^2 上一个光滑 d 次射影曲线，则

$$g = \frac{1}{2}(d-1)(d-2) \quad (8.7.7)$$

例如，若 $X = P^1$ ，则 $g=0$ 。由此可见，经典的 Goppa 码就是利用亏格为 0 的射影直线构造的线性码。

为了在一般的代数曲线上构造 Goppa 码，我们需要引进微分 ω 在点 P 的留数概念。

假设 P 是曲线 X 上一点， t 是在点 P 邻域的局部参数， $\omega = f dt$ 是 ω 的局部表示。于是函数 f 可展开成 Laurent 级数 $\sum_{-\infty}^{\infty} a_i t^i$ 。我们定义 ω 在点 P 的留数为 a_{-1} ，记为

$$\text{Res}_P(\omega) = a_{-1} \quad (8.7.8)$$

类似于复变函数论中的留数定理，可以同样证明下述定理。

定理 8.7.4 如果 ω 是光滑射影曲线 X 上的微分，则

$$\sum_{P \in X} \text{Res}_P(\omega) = 0 \quad (8.7.9)$$

§ 8.8 Riemann – Roch 定理

有了上一节的准备，我们便可介绍著名的 Riemann – Roch 定理了。

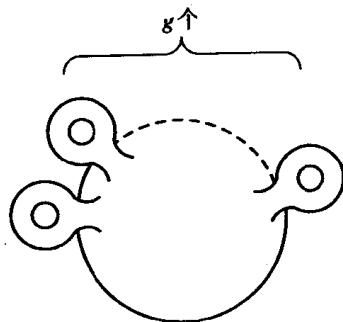


图 8-6 亏格的拓扑结构

定理 8.8.1 (Riemann - Roch 定理) 设 D 是亏格为 g 的光滑射影曲线上的一个除子。于是, 对于任意标准除子 W , 有

$$l(D) - l(W - D) = \deg(D) - g + 1 \quad (8.8.1)$$

这一定理不但在代数几何及其相关的理论(诸如数论等)中占有中心地位, 它同时也是在编码理论中获得具有重大意义结果的关键所在。

利用这一定理可以决定标准除子的次数。

推论 8.8.1 对于标准除子 W , 有

$$\deg(W) = 2g - 2 \quad (8.8.2)$$

证明 每一个在射影曲线上的正则函数均为常数, 从而 $\mathcal{L}(0) = k$, $l(0) = 1$ 。在定理 8.9.1 中置 $D = W$ 便可得到 $\deg(W) = 2g - 2$ 。 ■

推论 8.8.2 设 D 是亏格为 g 的光滑射影曲线的除子, 并且假设 $\deg(D) > 2g - 2$ 。于是

$$l(D) = \deg(D) - g + 1 \quad (8.8.3)$$

证明 由推论 8.8.1, $\deg(W - D) = \deg(W) - \deg(D) < (2g - 2) - (2g - 2) = 0$ 。由定理 8.8.2 之 1°, $\mathcal{L}(W - D) = \{0\}$, 从而 $l(W - D) = 0$, 再由定理 8.8.1 便得

$$l(D) = \deg(D) - g + 1 \quad \blacksquare$$

在定理 8.8.1 中之 $l(W - D)$ 可用微分的概念加以解释。我们引进定义 8.8.1 对于微分情形的相应概念。

定义 8.8.1 设 D 为曲线 X 上的一个除子, 我们定义

$$\Omega(D) = \{\omega \in \Omega(X) : (\omega) - D > 0\}$$

并且用 $\delta(D)$ 表示 $\dim_K \Omega(D)$, 称之为 D 的指标(index)。

可以证明:

$$\delta(D) = l(W - D) \quad (8.8.4)$$

为了说明 Riemann - Roch 定理的应用, 我们再回到 § 8.5 中关于代数几何码的构造问题上来。

设 X 是 F_q 上的光滑射影曲线。设 P_1, P_2, \dots, P_n 为 X 上的有理点, 并且 D 是除子 $P_1 + P_2 + \dots + P_n$ 。进一步, 假设 G 是某一另外的除子, 并且除子 G 由不同于诸 P_i 的有理点构成, 并且满足条件

$$2g - 2 < \deg(G) < n \quad (8.8.5)$$

定义 8.8.2 在 F_q 上长为 n 的线性码 $L(D, G)$ 是线性映射 $\varphi_L: \mathcal{L}(G) \rightarrow (F_q)^n$ 之下的像, φ_L 由下式定义:

$$\varphi_L(f) = (f(P_1), f(P_2), \dots, f(P_n))$$

称这种码为广义的几何 RS 码。

定理 8.8.2 $L(D, G)$ 码具有维数(信息位) $k = \deg(G) - g + 1$ 及最小距离 $d \geq n - \deg(G)$ 。

证明

1°. 如果 f 属于映射 φ_L 之核 $\text{Ker } \varphi_L = \{f \in \mathcal{L}(G) \mid \varphi_L(f) = 0\}$, 即 $f(P_i) = 0$, $i = 1, 2, \dots, n$, 再注意到 $f \in \mathcal{L}(G)$, 则必有 $f \in \mathcal{L}(G - D)$, 由式(8.8.5)有 $\deg(G - D) = \deg G - \deg D = \deg G - n < 0$ 。因此, 由定理 8.7.2 之 1°, $\mathcal{L}(G - D) = \{0\}$, 即 $f = 0$ 。这表明 φ_L

为 $\mathcal{L}(G)$ 到 $(F_q)^n$ 的单射。因此

$$l(G) = \dim_k \mathcal{L}(G) = L(D, G) \text{ 码的维数 } k$$

又由式(8.8.5)及推论 8.8.2, 得 $l(G) = \deg G - g + 1$ 。

2. 如果 $\alpha(f)$ 具有重量为 d , 则诸 P_i 中必有 $n-d$ 个点 $P_{i_1}, P_{i_2}, \dots, P_{i_{n-d}}$ 使 $f(P_{i_k}) = 0$, $k=1, 2, \dots, n-d$ 。因此, 对于除子 $E = P_{i_1} + \dots + P_{i_{n-d}}$, 必有 $f \in \mathcal{L}(G-E)$ (注意 $f \in \mathcal{L}(G)$)。从而 $(f) + (G-E) > 0$, 亦即

$$\deg(f) + \deg(G) - \deg(E) = \deg(G) - n + d \geq 0$$

此处由定理 8.7.1, $\deg(f) = 0$ 。 ■

上述的广义的几何 RS 码, 也称为第一类几何码。第二类几何码是所谓广义的几何 Goppa 码, 可定义如下。

定义 8.8.3 广义的几何 Goppa 码是 F_q 上码长为 n 的线性码 $L^*(D, G)$, 它是线性映射

$$\varphi_L^* : \Omega(G-D) \rightarrow (F_q)^n$$

之下的像, φ_L^* 由下式定义:

$$\varphi_L^*(\eta) = (\text{Res}_{P_1}(\eta), \text{Res}_{P_2}(\eta), \dots, \text{Res}_{P_n}(\eta)) \quad (8.8.6)$$

线性码 $L^*(D, G)$ 的相参数由下述定理给出。

定理 8.8.3 线性码 $L^*(D, G)$ 具有维数 $k^* = n - \deg(G) + g - 1$, 最小距离 $d^* \geq \deg(G) - 2g + 2$ 。

这一定理的证明与定理 8.8.2 类似, 也是定理 8.8.1(Riemann-Roch 定理)的直接推论。定理的证明留给读者。

正如经典的 Goppa 码与 RS 码的情形一样, 我们有下述的定理。

定理 8.8.4 $L(D, G)$ 与 $L^*(D, G)$ 互为对偶码。

证明 由定理 8.8.2 与定理 8.8.3, $k+k^*=n$ 。因此只需证明 $L(D, G)$ 与 $L^*(D, G)$ 中的码字互为正交, 即内积为 0。设 $f \in \mathcal{L}(G)$, $\eta \in \Omega(G-D)$, 由定义 8.8.2 与定义 8.8.3 可知微分 $f\eta$ 除在点 P_1, P_2, \dots, P_n 可能有一级极点外别无另外的极点, 并且

$$\text{Res}_{P_i}(f\eta) = f(P_i)\text{Res}_{P_i}(\eta)$$

由定理 8.7.4

$$\sum_{i=1}^n f(P_i)\text{Res}_{P_i}(\eta) = 0 \quad \blacksquare$$

由前面的讨论可以看到, 曲线 X 上有理点的个数与相应的线性码的码长有密切的关系。关于这一点我们介绍下述的著名结果。

定理 8.8.5(Weil 限) 设 X 是 F_q 上亏格为 g 的曲线, $N_q(X)$ 代表 X 上有理点的数目。于是有

$$|N_q(X) - (q+1)| \leq 2g\sqrt{q} \quad (8.8.7)$$

由上式可看出:

$$N_q(X) \leq q+1 + \lfloor 2g\sqrt{q} \rfloor \quad (8.8.8)$$

塞里(Serre)于 1983 年将此上限改进为

$$N_q(X) \leq q + 1 + g\lfloor 2\sqrt{q} \rfloor \quad (8.8.9)$$

§ 8.9 椭圆曲线码

在本章之末，我们对椭圆曲线码作一大致的介绍。

亏格为 1 的平面代数曲线称为椭圆曲线。椭圆曲线 X 在 P^2 上的一般齐次表达式为

$$y^2z + a_1xyz + a_3yz^2 = x^3 + a_2x^2z + a_4xz^2 + a_6z^3 \quad (8.9.1)$$

其中 $a_1, a_2, a_3, a_4, a_6 \in F_q$, 称

$$F(x, y, z) = y^2z + a_1xyz + a_3yz^2 - x^3 - a_2x^2z - a_4xz^2 - a_6z^3 \quad (8.9.2)$$

为 Weirstrass 多项式。

人们研究椭圆曲线的目的，在于它在有限域上能提供取之不尽的有限 Abel 群，并且由于这种群的丰富结构，还是能够实际计算的。

有许多方式在椭圆曲线 X 上构造 Abel 加法群。例如，取点 $Q = (0, 1, 0)$ (它在椭圆曲线上)作为加法单位元。如图 8-7 所示，对于曲线上任意两点 P_1, P_2 ，设连结 P_1, P_2 两点的直线与该曲线相交于点 R 。若连结 R, Q 的直线与该曲线相交于一点 P_3 ，则定义 $P_3 = P_1 \oplus P_2$ 。对于点 P_1, P_2 相重，或者有一个为 Q ，可类似地定义加法 \oplus 。对于曲线上的点 P ，我们还可以定义加法逆元 P^* ，满足 $P \oplus P^* = Q$ 。因此。曲线 X 上的点按上述加法运算 \oplus 构成加法群。不难看出 X 上的有理点全体构成上述加法群的一个子群。例如，在 § 8.4 中之例 8.1 中， $P_1 = (0, \alpha, 1)$ ，从而 $P_1^* = P_2 = (0, \beta, 1)$ ，同理， $P_3^* = P_4$ ， $P_5^* = P_6$ ， $P_7^* = P_8$ 。

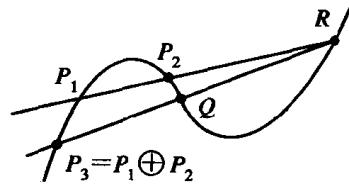


图 8-7 椭圆曲线上点的一种加法运算

表 8-2 F_4 上椭圆曲线码与 BCH 码的比较

| n | d | r_0 | r_1 |
|-----|-----|-------|-------|
| 25 | 8 | 16 | 13 |
| | 10 | 19 | 17 |
| 81 | 12 | 33 | 28 |
| | 14 | 37 | 34 |
| | 16 | 45 | 37 |
| 289 | 18 | 61 | 49 |
| | 20 | 71 | 57 |
| | 26 | 91 | 61 |

以椭圆曲线为底曲线构成的线性码，称为椭圆曲线码。

利用椭圆曲线上的有理点所构成的群，可以研究以椭圆曲线为底曲线所构成的线性码 $L(D, G)$ 与 $L^*(D, G)$ 的许多性质。

德林科特(Y. Driencourt)等人于 1987 年曾证明，对于码长为 n 的椭圆曲线码

$L(D, G)$, 其最小距离 d 介于 $n - \deg G$ 与 $n - \deg G + 1$ 之间(针对特征为 2 的域):

$$n - \deg G \leq d \leq n - \deg G + 1 \quad (8.9.3)$$

经验表明, 当码长 n 不太大时, 椭圆曲线码比 BCH 码的编码效率要高。表 8-2 针对 F_4 给出了这两种码的比较。表中 n 为码长, d 为最小距离, r_0 代表 BCH 码的校验位数目, r_1 代表椭圆曲线码的校验位数目。

关于椭圆曲线码的重量分布、覆盖半径以及各种码限的研究, 引起了不少学者的重视, 由于篇幅所限, 这里不再介绍, 请参阅[1]。

代数几何码的研究是当代编码理论界关注的焦点之一。人们利用各种代数曲线试图构造性能及参数较好的线性码, 从理论上探讨代数几何码的一般性质, 讨论曲线上有理点的个数(与码长有关)及码限问题。对于代数几何码, 编码问题比译码问题容易。为使代数几何码走向实用化, 寻找高效率的译码算法是这一领域中最为重要的课题。

编写本章的目的在于促进更多的读者了解代数几何码这一新兴的领域。当然, 真正走向这一领域的研究, 还需要一段艰苦的历程。代数几何码的出现, 正如著名编码学家范林特所指出的那样, 它证明了一个重要的事实, 任何数学, 不管它多么高深, 最终总是会在应用领域中放出异彩。

习 题

1. 试将下列仿射平面上的代数曲线方程变为射影平面上的代数曲线方程:

$$(1) y^2 + y = x^3 + x^2 + x + 1$$

$$(2) x^r + y^r + 1 = 0$$

2. 试将下列射影平面上的代数曲线方程变换为仿射平面上的代数曲线方程:

$$(1) x^3y + y^3z + z^3x = 0$$

$$(2) x^5 + y^2z^3 + yz^4 + z^5 = 0$$

3. 对于例 8.2 中的 ψ_4 , 试求出 $\text{div}(\psi_4)$ 。

4. 证明 § 8.7 中的 $m_P(X)$ 是环 $U_P(X)$ 中唯一的极大理想。

5. 在 F_4 的代数闭域 \bar{F}_4 上考察 P^2 中由方程

$$f(x, y, z) = x^3y + y^3z + z^3x = 0$$

定义的代数曲线(Klein 四次曲线)。试求出该曲线在 F_4 上的全部有理点。

6. 证明曲线 X 上的所有次数为 0 的除子(即 $\deg(D)=0$)构成 $\text{div}(X)$ 的子群。

7. 证明在 § 8.7 中对于 $\text{div}(X)$ 中建立的关系 $D \equiv D'$ ($D, D' \in \text{div}(X)$) 是一等价关系。

并且进一步证明:

$$1^\circ D \equiv 0 \Leftrightarrow D = (f)$$

$$2^\circ D \equiv D' \Leftrightarrow \deg(D) = \deg(D')$$

$$3^\circ D \equiv D', D_1 \equiv D'_1 \Leftrightarrow D + D_1 \equiv D' + D'_1$$

8. 证明 § 8.7 中的定理 8.7.2 之 1°。

9. 证明 § 8.8 中的定理 8.8.3。

参 考 文 献

- [1] J. H. Van Lint, Algebraic Geometric Codes, Coding Theory and Design Theory, Part I Coding Theory, pp. 137—162, 1990.
- [2] 小西健司, 代数几何的符号理论, 《数理科学》, No. 303, Sept. 1988.
- [3] V. D. Goppa, Geometry and Codes, Kluwer Academic Publishers, 1988.

第九章 纠突发错误循环码

大部分实际信道如短波、散射、有线等信道中所产生的错误是突发性的；某些数据存储系统中的磁带、磁盘、磁鼓等由于伤痕、读写头的接触不良等所产生的错误，大部分也是突发性的，而不是随机性的。这就是说这些信道中所产生的错误是突发错误，或突发错误与随机错误并存。通常称这类信道为有记忆信道或突发信道。

针对突发信道的最好差错控制方法是利用 ARQ 方式，但由于 ARQ 方式的固有缺点，在某些情况下不能使用，必须采用 FEC 方式。一般而言，前面各章所讨论的纠随机错误码，除了与交错等方式结合外，对于纠正突发错误并没有什么效果。因此，必须专门设计一种用来纠正突发错误的纠错码。本章所讨论的这类码，就是纠正突发错误或纠正突发和随机错误的循环分组码，这种码类在实际中比较有使用价值。

§ 9.1 基本码限

第一章中已定义了长为 b 的突发错误图样。如果以多项式表示突发错误图样，则在长为 n 的码字内，长度 $\leq b$ 的突发错误图样是

$$E(x) = x^i b(x) \quad i = 0, 1, 2, \dots, n - b$$

式中， x^i 表示突发开始的位置， $b(x)$ 代表突发错误图样，且 $b(x) \leq b-1$ 。如 $b(x) = x^3 + x^2 + 1$ ，表示突发错误图样是(1101)，这是一个长为 4 的突发错误。对循环码码字中的突发错误图样，还可定义循环(首尾相接)突发错误图样：

$$E(x) = x^i b(x) \quad i = 0, 1, 2, \dots, n - 1 \pmod{x^n - 1}$$

如(10…00101)，就是一个长为 4 的循环突发，它的错误图样多项式

$$E(x) = x^{n-1} + x^2 + 1$$

循环移位一次后成为(00…01011)，相应的

$$E'(x) = xE(x) = x^n + x^3 + x \equiv x^3 + x + 1 \pmod{x^n - 1}$$

今后，我们所指的纠突发错误循环码，若没有特别说明，也可纠正循环突发错误。

任一个 $[n, k]$ 分组码，若能纠正码组中长度 $\leq b$ 的所有突发错误图样，则称 b 为该码的纠突发能力。在给定的 n 与 b 下，显然要求构成的 $[n, k]$ 码有最少的多余码元。下面定理给出了 n, b 与校验元数目之间的关系。

定理 9.1.1 一个 q 进制 $[n, k]$ 线性分组码，若要发现(或检测)所有长度 $\leq b$ 的突发错误，其充要条件是需要 b 个校验元。

证明

(1) 任何一个长度 $\leq b$ 的突发错误图样(下简称突发)不能作为一个码字。因为 $[n, k]$ 是线性码，必有一个全 0 码字 C_0 ，在传输过程中，若信道产生的突发是长度 $\leq b$ 的 n 重 E_b ，则译码器收到的 n 重 $R = C_0 + E_b = E_b$ 是一个仅含有长度 $\leq b$ 的突发，若任一长度 $\leq b$ 的突发

\mathbf{E}_b 作为一个码字 C_1 , 则译码器不能区分收到的 \mathbf{E}_b 是 $C_0 + \mathbf{E}_b$ 产生的, 还是发的就是码字 $C_1 = \mathbf{E}_b$ 。因此长度 $\leq b$ 的突发不能作为一个码字。

(2) 任何两个具有相同位置的, 长度 $\leq b$ 的突发 $\mathbf{E}_1, \mathbf{E}_2$ 不能在同一陪集中。若在同一陪集中, 则 $\mathbf{E}_1 + \mathbf{E}_2 = \mathbf{E}_3$ 也是一个长度 $\leq b$ 的突发, 且由陪集性质知 \mathbf{E}_3 也是一个码字, 这与(1)相矛盾。

(3) 在某一固定位置上, 长度 $\leq b$ 的突发共有 $q^b - 1$ 个, 而 q 进制 $[n, k]$ 线性码有 q^{n-k} 个陪集, 所以只有 $q^{n-k} \geq q^b$ 时, 才能使得每个长度 $\leq b$ 的突发处在不同的陪集中, 从而发现这些错误。所以

$$n - k \geq b \quad (9.1.1)$$

(4) 反之, 若码有 b 个校验元, 则码的一致校验矩阵有 b 行、 n 列, 可如下构造:

$$\begin{array}{ccccccccccccccccccccc} 1 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 0 \cdots & 0 & 0 & 1 & 0 \cdots & 0 & 0 & 1 & 0 \cdots & 0 & 0 & 1 & 0 \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 1 \end{array}$$

$\underbrace{\hspace{1cm}}_b \quad \underbrace{\hspace{1cm}}_b \quad \underbrace{\hspace{1cm}}_b$

因此任何长度 $\leq b$ 的突发 \mathbf{E}_b , 用 \mathbf{H} 进行检验后, 其伴随式 S_b 为

$$S_b = \mathbf{E}_b \cdot \mathbf{H}^T \neq 0$$

故可检测到错误。 ■

推论 9.1.1 任何 $[n, k]$ 线性码, 能发现所有长度 $\leq n - k$ 的突发错误。

定理 9.1.2 一个 q 进制 $[n, k]$ 线性码, 若

(1) 要纠正所有长度 $\leq b$ 的突发, 则至少需 $2b$ 个校验元, 即

$$n - k \geq 2b \quad (9.1.2)$$

(2) 要纠正所有长度 $\leq b$, 且同时发现所有长度 $\leq d, \geq b$ 的突发, 至少需 $b+d$ 个校验元, 即

$$n - k \geq b + d \quad (9.1.3)$$

证明

(1) 任何一个长度 $\leq 2b$ 的突发不能作为一个码字。

设有一长度 $\leq 2b$ 的突发 \mathbf{E} 是一个码字, 则该码字可以看成是两个长度 $\leq b$ 的突发 $\mathbf{e}_1, \mathbf{e}_2$ 之和, $\mathbf{E} = \mathbf{e}_1 + \mathbf{e}_2$ 。由陪集性质可知, $\mathbf{e}_1, \mathbf{e}_2$ 必在同一陪集中。若 \mathbf{e}_1 (或 \mathbf{e}_2) 作为陪集首, 则 \mathbf{e}_2 (或 \mathbf{e}_1) 将是不能纠正的长度 $\leq b$ 的突发。所以, 为了使码能纠正任何长度 $\leq b$ 的突发错误图样, 任何一个长度 $\leq 2b$ 的突发, 或任何两个长度 $\leq b$ 的突发的组合, 不能作为一个码字。

(2) 在某一固定位置的长度 $\leq 2b$ 的突发, 共有 $q^{2b} - 1$ 个, 而 q 进制 $[n, k]$ 线性码共有 q^{n-k} 个陪集, 所以只有当

$$q^{n-k} \geq q^{2b} \quad \text{或} \quad n - k \geq 2b$$

时, 才能使得每个长度 $\leq 2b$ 的突发处在不同的陪集之中。

(3) 类似地, 每个长度 $\leq b+d$ 的突发, 可以看成是一个长度 $\leq d$, 与另一个长度 $\leq b$ 的突发的组合。若码要检测长度 $\leq d$, 并同时纠正长度 $\leq b$ 的突发, 则它们之和不能是一个码字, 即它们也应处在不同的陪集之中, 因此至少需 $b+d$ 个校验元。 ■

一般称式(9.1.2)为赖格尔(Rieger)限, 简称 R 限。这是每个能纠正长度 $\leq b$ 的突发错

误线性码所必须满足的必要条件。

推论 9.1.2 若 $[n, k]$ 线性码要具有能纠正任何长度 $\leq b$ 的突发错误能力，其充要条件是任何两个长度 $\leq b$ 的突发的任意组合不能作为一个码字。

证明 由证明定理 9.1.2 中的(1)可知其必要性。现证明充分性。设错误图样为

$$\mathbf{E}_b = \mathbf{e}_{b_1} + \mathbf{e}_{b_2}$$

式中， \mathbf{e}_{b_1} 、 \mathbf{e}_{b_2} 分别是两个长度 $\leq b$ 的突发。若 \mathbf{E}_b 不是 $[n, k]$ 码的一个码字，则

$$\mathbf{E}_b \mathbf{H}^T = \mathbf{e}_{b_1} \mathbf{H}^T + \mathbf{e}_{b_2} \mathbf{H}^T \neq 0$$

式中， \mathbf{H} 是 $[n, k]$ 码的一致校验矩阵。由该式可知：

$$-\mathbf{e}_{b_1} \mathbf{H}^T \neq \mathbf{e}_{b_2} \mathbf{H}^T \neq 0$$

因此， \mathbf{e}_{b_1} 的伴随式 $\mathbf{e}_{b_1} \mathbf{H}^T$ 不等于 \mathbf{e}_{b_2} 的伴随式 $\mathbf{e}_{b_2} \mathbf{H}^T$ ，且不为零，由此可知任何两个长度 $\leq b$ 的突发必能区分和纠正。 ■

推论 9.1.3 具有纠突发能力为 b 的 $[n, k]$ 线性码，能检测任何两个长度 $\leq b$ 的突发错误的所有组合；反之亦然。

推论 9.1.4 具有纠突发能力为 b 的 $[n, k]$ 线性码，能纠正任何两个长度 $\leq b$ 的突发删除错误。

证明 两个长度 $\leq b$ 的突发删除错误的位置是已知的，因此可以在这两个位置上放入任意码元组合，然后计算伴随式是否为0。由推论 9.1.3 可知，该码能检测任意两个长度 $\leq b$ 的突发，所以若伴随式不为0，则说明在删除位置上放入的码元组合，不是被删除掉的正确码元，故可另用两组长度 $\leq b$ 的码元组合放入删除位置，重新计算伴随式。这过程可一直进行下去，直到伴随式为0为止，此时说明这时放入的两组长度 $\leq b$ 的码元组合，就是被删除掉的正确码元。 ■

定理 9.1.2 是线性码纠突发能力的上限，如果码的纠错能力能达到 R 限，则称该码为 R 意义下的最佳纠突发错误码，简称 **R 最佳码**。我们通常用

$$Z = \frac{2b}{n - k} \quad (9.1.4)$$

来衡量码的最佳程度。显然， Z 越大码的纠突发能力也越强； $Z=1$ 就达到了纠突发能力的上限，也就是码为最佳码。有时也用

$$\sigma = n - k - 2b \quad (9.1.5)$$

来衡量码纠突发能力“差”的程度， $\sigma=0$ 说明码是最佳码， σ 越大说明码纠突发能力程度越低。

下面从伴随式的角度定义最佳码。在 n 长 q 进制码字中，所有可能的长度 $\leq b$ 突发错误图样有 $(n-b+2)(q-1)q^{b-1}$ 个，因此码要能纠正所有 $\leq b$ 的突发错误，则伴随式数目

$$q^{n-k} \geq (n-b+2)(q-1)q^{b-1} \quad (9.1.6)$$

如果是 q 进制循环码，则在码字中，所有可能的长度 $\leq b$ 的突发错误图样有 $n(q-1)q^{b-1}$ 个，因此仅当

$$q^{n-k} \geq 1 + n(q-1)q^{b-1} \quad (9.1.7)$$

或

$$n \leq \frac{q^{(n-k)-b+1} - 1}{q - 1} \quad (9.1.8)$$

成立时, $[n, k]$ 循环码才能纠正所有长度 $\leq b$ 的突发错误。

能使式(9.1.6)或式(9.1.7)、式(9.1.8)中等式成立的码也称为最佳码。显然, 这时的伴随式已充分利用, 所以这类码称伴随式意义下的最佳码(完备码), 简称 S 最佳码, 该限也称为 S 限。因此, 式(9.1.2)是线性码所必须满足的, 而式(9.1.7)是循环码所必须满足的。

如果 $[n, k]$ 最佳循环码的码长 $n = (q^m - 1)/(q - 1)$, 且 $m > (n - k) - b + 1$, 则由 R 限可知, $m \geq b + 1$ 。也就是说仅当 $m \geq b + 1$ 时, 才存在 S 最佳码。下面若无特别说明, 我们均指 R 限意义下的最佳码, 也即 R 最佳码。

§ 9.2 纠单个错误循环码的构造

由上节讨论看到, 构造纠单个突发错误码, 就是致力于构造纠错能力接近或达到 $Z=1$ 的准最佳与最佳码。有关构造纠突发错误码的方法有以下几种: 一是用分析方法构造; 另一是把短的最佳码用交错方法构造成长的最佳码; 再一种方法是用计算机搜索, 寻找最佳码。这一节主要讨论用分析方法构造几类纠突发错误码的方法。

一、纠随机错误循环码的纠突发能力

讨论纠随机错误循环码, 特别是 BCH 码的纠突发能力是很有意义的。定理 9.1.1 告诉我们, 一个 $[n, k]$ 循环码能检测长度 $\leq n - k$ 的所有突发错误。当然一个能纠正 t 个随机错误的循环码, 也必然能纠正长度为 $b \leq t$ 的突发错误。但是, 事实上很多循环码的纠突发能力远超过 t , 因此必须讨论循环码纠突发能力的下限。

定理 9.2.1 有最小距离为 d 的 $[n, k]$ 循环码, 能检测每个长度 $\leq b_i$ ($i = 1, 2, \dots, T$) 的所有 T 个突发, 其中

$$\begin{aligned} T &\leq d - 1 \\ \sum_{i=1}^T b_i &\leq \max\left(\left\lfloor \frac{3d - 2T - 4}{2} \right\rfloor, d - 1\right) \end{aligned} \quad (9.2.1)$$

证明 设 T 个突发图样的多项式表示是 $B(x)$, 其中每个突发的长度是 b_i , $i = 1, 2, \dots, T$ 。若所有突发均为实心(突发图样中无 0 分量), 因而 T 个突发总重量为 $\sum_{i=1}^T b_i$ 。设 $B(x)$ 中有 m 项的系数为 0, 因而

$$w(B(x)) = \sum_{i=1}^T b_i - m$$

$B(x)$ 不能是一个码字, 所以

$$w(B(x)) = \sum_{i=1}^T b_i - m < d \quad (9.2.2)$$

由于码是循环码, 所以 $(1+x)B(x) = B(x) + xB(x)$ 也不能是一个码字, 故

$$w(B(x)(1+x)) \leq 2T + 2m < d \quad (9.2.3)$$

这是由于有 T 个突发, 而 $B(x)$ 与 $xB(x)$ 之间每个突发对应位均相错一位, 所以, $B(x) + xB(x)$ 的重量至多为 $2T + 2m$, 其中 $2T$ 是由于 T 个突发的始末所提供的重量, $2m$ 是 m 个 0 项位置所提供的重量。由式(9.2.3)可得:

$$m < \frac{d}{2} - T$$

或

$$m \leq \frac{d}{2} - T - 1$$

把上式代入式(9.2.2)可得：

$$\begin{aligned}\sum_{i=1}^T b_i - \left(\frac{d}{2} - T - 1\right) &< d \\ \sum_{i=1}^T b_i &< d + \frac{d}{2} - T - 1 = \frac{3d - 2T - 2}{2} \\ \sum_{i=1}^T b_i &\leq \frac{3d - 2T - 2}{2} - 1 = \frac{3d - 2T - 4}{2}\end{aligned}$$

另一方面，若

$$\sum_{i=1}^T b_i \leq d - 1$$

则显然在码的检错能力以内。所以

$$\sum_{i=1}^T b_i \leq \max\left\{\left\lfloor \frac{3d - 2T - 4}{2} \right\rfloor, d - 1\right\}$$

■

定理 9.2.2 最小距离为 d 的 $[n, k]$ 循环码，能同时纠正 p 个突发错误，且每个突发长度为 b_i , $i=1, 2, \dots, p$, 这里

$$\begin{aligned}p &\leq \left\lfloor \frac{d-1}{2} \right\rfloor \\ \sum_{i=1}^p b_i &\leq \max\left\{\left\lfloor \frac{3d - 4p - 4}{4} \right\rfloor, \left\lfloor \frac{d-1}{2} \right\rfloor\right\}\end{aligned}\quad (9.2.4)$$

证明 由定理 9.2.1 可知，该码能检测满足式(9.2.1)的 T 个突发错误。令 $T=2p$ 代入式(9.2.1)，并由推论 9.1.3 可知，该码能纠正总长为

$$\sum_{i=1}^p b_i \leq \left\lfloor \frac{1}{4}(3d - 4p - 4) \right\rfloor$$

的任何 p 个突发。另一方面，若 p 个突发总长不超过 $t=[(d-1)/2]$ ，则码显然能纠正它，因而

$$\sum_{i=1}^p b_i \leq \left\lfloor \frac{d-1}{2} \right\rfloor$$

所以该码的纠突发能力为

$$\sum_{i=1}^p b_i \leq \max\left\{\left\lfloor \frac{3d - 4p - 4}{4} \right\rfloor, \left\lfloor \frac{d-1}{2} \right\rfloor\right\}$$

■

若 $p=1$ ，则式(9.2.4)成为

$$b \leq \max\left\{\left\lfloor \frac{3d - 8}{4} \right\rfloor, \left\lfloor \frac{d-1}{2} \right\rfloor\right\}\quad (9.2.5)$$

这说明任何 $[n, k, d]$ 循环码都具有纠正长度由式(9.2.5)决定的单个突发错误能力。但是，这种能力是很小的，仅当 $d \geq 7$ 时， $[n, k, d]$ 循环码才有可能纠正长度 $\leq [(d-1)/2]+1=t+1$ 的突发错误。但实际上，近几年的研究表明^[3,4,5]，循环码特别是 BCH 码的纠单个突发

错误能力远远超过式(9.2.5)，如表7-1所示。因此，如何比较精确地决定循环码的纠突发能力，特别是纠突发能力的下限，无论从扩大码的应用范围，还是从理论上，都是很有意义的。

定理9.2.3^[3] 对任何一个 $[n, k, d \geq 3]$ 二进制循环码，纠突发能力

$$b \geq \left\lfloor \frac{n - 2k + 1}{2} \right\rfloor \quad (9.2.6)$$

显然，此定理仅适用于码率 $R < 0.5$ 的情况，码率越低则由式(9.2.6)得到的 b 与实际的越接近。对于 $R \geq 0.5$ 的码，此定理并不适用。但此定理所给出的比式(9.2.5)所确定的要准确。

对于绝大部分二进制 $[n, k, d \geq 3]$ BCH码来说，我们可以证明其纠突发能力 b 的上、下限为^[4]

$$d - 2 \leq b \leq \left\lfloor \frac{n - k}{2} \right\rfloor \quad (9.2.7)$$

此限比式(9.2.6)给出的更紧。由于BCH码的最小距离 d 与 $g(x)$ 的根有密切关系，所以式(9.2.7)也给出了码生成多项式 $g(x)$ 的根与 b 的关系。可以根据 b 的要求很快地得到所需要的码，从而也为BCH码扩大了纠错能力和应用范围。与表7-1比较可知：当 n 不很长，码率 R 不在0.5附近，式(9.2.7)给出的下限与码实际所具有的 b 非常一致；仅当 n 较大， R 为0.5时该限才显得比较松，但仍比式(9.2.6)的限要紧。

除了表7-1和表7-2中列出的BCH码的纠突发能力以外，表9-1中还列出了用计算机搜索得到的最佳或接近最佳的、纠单个突发错误循环与准循环码。如果把这些码与交错方法相结合，就能得到码长更长，能纠正更长突发的最佳或准最佳码，因此表中所列的这些码有较大的实际意义。表中， $g(x)$ 列下面的数字代表 $g(x)$ 多项式的二进制系数的八进制数表示，与表7-1同。

二、Fire码

弗尔(Fire)码是最早(1959年)也是最大的一类用分析方法构造出的纠单个突发错误的二进制循环码。由于可以根据不同的要求很方便地设计所需要的码， Z 接近 $2/3$ ，译码也很简单，因此Fire码是一类比较实用的，也是最基本的纠单个突发错误循环码。

定理9.2.4 设 $g_1(x)$ 生成一个纠突发能力为 b 的 $[n_1, k_1]$ 循环码， $p(x)$ 是周期为 a 的多项式，且 $\partial \cdot p(x) \geq b$ ， $(p(x), g_1(x)) = 1$ ，则由 $g(x) = g_1(x)p(x)$ 生成的循环码，码长 $n = n_1a$ ，能纠正长度 $\leq b$ 的所有突发错误。

表9-1 用计算机搜索得到的最佳或准最佳循环码

| σ | (n, k) | b | $R = \frac{k}{n}$ | $g(x)$ |
|----------|----------|-----|-------------------|-------------|
| 0 | [7, 3] | 2 | 0.42 | 35 |
| 0 | [15, 9] | 3 | 0.60 | 171 |
| 0 | [15, 7] | 4 | 0.47 | 721 |
| 0 | [15, 5] | 5 | 0.33 | (31)(23)(7) |
| 0 | [19, 11] | 4 | 0.578 | 625 |

(续表)

| σ | (n, k) | b | $R = \frac{k}{n}$ | $g(x)$ |
|----------|------------|-----|-------------------|----------------|
| 0 | [27, 17] | 5 | 0.63 | 2671 |
| 0 | [30, 17] | 6 | 0.60 | (23)(23)(7)(7) |
| 0 | [34, 22] | 6 | 0.65 | 15173 |
| 0 | [38, 24] | 7 | 0.63 | 53345 |
| 0 | [50, 34] | 8 | 0.68 | 224531 |
| 0 | [56, 38] | 9 | 0.68 | 1505773 |
| 0 | [59, 39] | 10 | 0.66 | 4003351 |
| 0 | [65, 43] | 11 | 0.66 | 37774757 |
| 0 | [72, 48] | 12 | 0.67 | 177772011 |
| 0 | [78, 52] | 13 | 0.67 | 564625255 |
| 0 | [82, 54] | 14 | 0.66 | 3000056271 |
| 1 | [15, 10] | 2 | 0.67 | 65 |
| 1 | [21, 14] | 3 | 0.67 | (127)(3) |
| 1 | [21, 12] | 4 | 0.57 | (17)(15) |
| 1 | [27, 20] | 3 | 0.74 | 311 |
| 1 | [31, 20] | 5 | 0.66 | (75)(45)(3) |
| 1 | [38, 29] | 4 | 0.76 | 1151 |
| 1 | [48, 57] | 5 | 0.77 | 4501 |
| 1 | [63, 50] | 6 | 0.79 | (163)(103)(3) |
| 1 | [67, 54] | 6 | 0.81 | 36365 |
| 1 | [96, 79] | 8 | 0.82 | 501001 |
| 1 | [103, 88] | 7 | 0.85 | 114361 |
| 1 | [111, 88] | 11 | 0.79 | 6005741 |
| 1 | [121, 102] | 9 | 0.84 | 2203717 |
| 1 | [127, 106] | 10 | 0.84 | 12463441 |
| 1 | [131, 106] | 12 | 0.82 | 321327031 |
| 2 | [31, 25] | 2 | 0.81 | 161 |
| 2 | [63, 55] | 3 | 0.87 | 711 |
| 2 | [85, 75] | 4 | 0.88 | 2651 |
| 2 | [105, 93] | 5 | 0.89 | 13321 |
| 2 | [105, 91] | 6 | 0.87 | (127)(31)(23) |
| 2 | [131, 119] | 5 | 0.91 | 15163 |
| 2 | [169, 155] | 6 | 0.92 | 55725 |
| 2 | [200, 184] | 7 | 0.92 | 207375 |
| 2 | [209, 191] | 8 | 0.91 | 1274533 |
| 2 | [248, 228] | 9 | 0.92 | 4076773 |

证明 由推论 9.1.2 可知, 只要证明任何两个长度 $\leq b$ 的突发组合不是由 $g(x)$ 生成的码的一个码组即可。设 $B_1(x)、B_2(x)$ 是两个长度 $\leq b$ 的突发, 如果 $x^i B_1(x)$ 和 $x^j B_2(x)$ 的组合是一个码组, 则

$$\begin{aligned} x^i B_1(x) + x^j B_2(x) &= x^i(B_1(x) + x^{j-i} B_2(x)) = q'_1(x)g(x) \\ &= q'_1(x)g_1(x)p(x) \quad j > i, 0 \leq i, j \leq n-1 \end{aligned}$$

因为 $g(x) \nmid x^i, g_1(x) \mid x^{n_1}-1$, 所以, $g_1(x) \mid x^{n_1\gamma}-1$, 且

$$\begin{aligned} B_1(x) + x^i B_2(x) &= q_1(x)g_1(x)p(x) \quad s = j - i < n \\ x^{n_1\gamma}-1 &= g_1(x)q'_2(x) \quad \gamma = 1, 2, \dots \end{aligned} \quad (9.2.8)$$

因此

$$\begin{aligned} x^i B_2(x) &= x^{s-\gamma n_1} B_2(x) x^{\gamma n_1} \\ &= x^{s-\gamma n_1} B_2(x) (1 + g_1(x)g'_2(x)) \\ &= x^{s-\gamma n_1} B_2(x) + g_1(x)q_2(x) \end{aligned}$$

由式(9.2.8)及上式可得:

$$\begin{aligned} B_1(x) + x^i B_2(x) &= B_1(x) + x^{s-\gamma n_1} B_2(x) + g_1(x)q_2(x) \\ &= q_1(x)g_1(x)p(x) \\ B_1(x) + x^{s-\gamma n_1} B_2(x) &= q_1(x)g_1(x)p(x) - g_1(x)q'_2(x) \\ &= g_1(x)(q_1(x)p(x) - q_2(x)) = g_1(x)q'_2(x) \end{aligned}$$

这里, 若所选的 γ 使 $x^{s-\gamma n_1} B_2(x)$ 的次数小于 n_1 , 则由上两式看出, 两个长度 $\leq b$ 的突发 $B_1(x)$ 和 $B_2(x)$, 是 $g_1(x)$ 生成的 $[n_1, k_1]$ 码中的一个码字, 而这与假设码能纠 $\leq b$ 长的突发相矛盾。因此, 只有使 $s-\gamma n_1=0, s=\gamma n_1$, 且 $B_1(x)=B_2(x)$ 。故

$B_1(x)=B_2(x)=B(x)$, 则式(9.2.8)可写成

$$B(x)(x^{\gamma n_1}+1) = q_1(x)g_1(x)p(x)$$

但由假设 $\partial \circ p(x) \geq b$, 故它除不尽 $B(x)$, 因而 $p(x) \mid x^{\gamma n_1}+1$ 。这就是说, $g_1(x)$ 与 $p(x)$ 均除尽 $x^{\gamma n_1}+1$, 但这是不可能的, 因为已假设 $(g_1(x), p(x))=1$ 。因此只有 $\gamma n_1 > n$ 才有可能。另一方面, $g_1(x)p(x)$ 是除尽 x^n-1 的最小多项式, $n > s = \gamma n_1$, 这与前面所讲的 $\gamma n_1 > n$ 相矛盾, 故

$$x^i B_1(x) \not\equiv x^i B_2(x) \pmod{g_1(x)p(x)} \quad \blacksquare$$

因此, 若选择 $g_1(x)=x^{2b-1}+1$, 生成一个 $[2b-1, 0]$ 码, 则由于它是唯一的一个全为 0 码字, 不可能是两个长度 $\leq b$ 的突发图样的组合, 所以它能纠正所有 $\leq b$ 的单个突发错误, 由此得出 Fire 码。

推论 9.2.1(Fire 码) 由

$$g(x) = (x^{2b-1}+1)p(x) \quad (9.2.9)$$

生成的 $[n, n-2b-m+1]$ 循环码称 Fire 码, 它能纠正码字内长度 $\leq b$ 的所有单个突发错误。码长 $n=\text{LCM}(e, 2b-1)$, e 是 $p(x)$ 的周期, 且 $\partial \circ p(x)=m \geq b$, $(p(x), x^{2b-1}+1)=1$ 。

若取 $m=b$, 则校验元数为 $3b-1$, 因而 Fire 码的

$$Z = \frac{2b}{3b-1}$$

当 b 较大时, Z 接近 $2/3$ 。

在 Fire 码生成多项式 $g(x)$ 中的因子 $x^{2b-1} + 1$, 除不尽任何长度 $\leq b-1$ 的突发, 因此由该因子得到了错误图样。另一因子 $p(x)$ 则与 $x^{2b-1} + 1$ 一起共同决定突发所发生的位置。

推论 9.2.2(广义 Fire 码) 由

$$g(x) = (x^c - 1) \prod_{j=1}^m p_j(x) \quad (9.2.10)$$

生成的 $[n, k]$ 二进制循环码, 能检测所有长度 $\leq d$ 的突发, 或纠正所有长度 $\leq b$ 的突发图样 $B(x)$, 且 $(B(x), \prod_{j=1}^m p_j(x)) = 1$ 。这里

$$\begin{aligned} c &\geq b + d - 1 & b &\leq \sum_{i=1}^m m_i \\ n &= \text{LCM}\{c, e_1, e_2, \dots, e_m\} \\ k &= n - c - \sum_{i=1}^m m_i \end{aligned}$$

e_j 是 $p_j(x)$ 的周期。

由该推论看到, 广义 Fire 码仅只能纠正突发图样 $B(x)$ 与 $\prod_{j=1}^m p_j(x)$ 互素的部分, 不能全部纠正长度 $\leq b$ 的突发图样。但不能纠正部分的图样比例, 可以证明随着 $p_j(x)$ 次数的增加而指数下降, 并且也与译码方法有关。

例 9.1 构造一个 $n=15, b=3$ 的 Fire 码。由推论 9.2.1 可知, 该码的生成多项式

$$g(x) = (x^{2b-1} + 1)p(x) = (x^5 + 1)(x^4 + x + 1)$$

$$n = \text{LCM}(5, 15) = 15, n - k = 9, Z = \frac{6}{9} = \frac{2}{3}$$

该 $[15, 6]$ 码能纠正所有长度 ≤ 3 的突发错误, 或者检测所有长度 $\leq b$ 的突发错误。 ■

事实上, 绝大部分 Fire 码实际所具有的纠突发能力 b_c , 大大超过 Fire 码定义所给出的纠突发能力 b_F 。表 9-2 中列出了用计算机搜索得到的, 某些 Fire 码的 b_F 与 b_c 及其它参数。表中, $p(x)$ 栏下的数字代表 $p(x)$ 多项式的二进制系数的八进制数表示, 如“23”表示“010011”, 说明 $p(x)=1+x+x^4$ 。

表 9-2 用计算机得到的某些 Fire 码的 b_c 与 b_F

| n | k | b_c | b_F | $r/n \%$ | $b_c/r \%$ | c | $p(x)=g(x)/(x^e-1)$ |
|-----|-----|-------|-------|----------|------------|-----|---------------------|
| 15 | 6 | 4 | 3 | 60.0 | 44.4 | 5 | 23 |
| 21 | 8 | 6 | 4 | 61.9 | 46.2 | 7 | 127 |
| 33 | 12 | 9 | 6 | 63.6 | 42.9 | 11 | 3043 |
| 39 | 14 | 12 | 7 | 64.1 | 48.0 | 13 | 13617 |
| 45 | 18 | 12 | 8 | 60.0 | 44.4 | 15 | 10011 |
| 57 | 20 | 16 | 10 | 64.9 | 43.2 | 19 | 1341035 |
| 69 | 24 | 22 | 12 | 65.2 | 48.9 | 23 | 34603145 |
| 75 | 30 | 20 | 13 | 60.0 | 44.4 | 25 | 410001 |
| 87 | 30 | 28 | 15 | 65.5 | 49.1 | 29 | 3706175715 |
| 99 | 36 | 27 | 17 | 63.6 | 42.9 | 33 | 11000100011 |

(续表)

| <i>n</i> | <i>k</i> | <i>b_c</i> | <i>b_F</i> | <i>r/n %</i> | <i>b_c/r %</i> | <i>c</i> | <i>p(x)=g(x)/(x^a-1)</i> |
|----------|----------|----------------------|----------------------|--------------|--------------------------|----------|------------------------------------|
| 75 | 40 | 15 | 8 | 46.7 | 42.9 | 15 | 4100001 |
| 65 | 40 | 10 | 7 | 38.5 | 40.0 | 13 | 10761 |
| 145 | 88 | 24 | 15 | 39.3 | 42.1 | 29 | 3572445367 |
| 155 | 104 | 20 | 16 | 32.9 | 39.2 | 31 | 5521623 |
| 215 | 144 | 28 | 22 | 33.0 | 39.4 | 43 | 3657555473 |
| 105 | 72 | 12 | 11 | 31.4 | 36.4 | 21 | 13321 |
| 165 | 112 | 20 | 17 | 32.1 | 37.7 | 33 | 6130725 |
| 175 | 120 | 20 | 18 | 31.4 | 36.4 | 35 | 4102041 |
| 63 | 48 | 6 | 5 | 23.8 | 40.0 | 9 | 103 |
| 91 | 66 | 11 | 7 | 27.5 | 44.0 | 13 | 15173 |
| 105 | 78 | 12 | 8 | 25.7 | 44.4 | 15 | 13321 |
| 189 | 144 | 18 | 14 | 23.8 | 40.0 | 27 | 1011011 |
| 231 | 168 | 30 | 17 | 27.3 | 47.6 | 33 | 11274767701 |
| 245 | 189 | 21 | 18 | 22.9 | 37.5 | 35 | 10040001 |
| 63 | 50 | 5 | 4 | 20.6 | 38.5 | 7 | 103 |
| 117 | 92 | 9 | 7 | 21.4 | 36.0 | 13 | 10377 |
| 171 | 134 | 15 | 10 | 21.6 | 40.5 | 19 | 1055321 |
| 385 | 300 | 30 | 28 | 22.1 | 35.3 | 55 | 16471647235 |
| 85 | 72 | 5 | 3 | 15.3 | 38.5 | 5 | 567 |
| 143 | 120 | 8 | 7 | 16.1 | 34.8 | 13 | 3777 |
| 165 | 140 | 10 | 8 | 15.2 | 40.0 | 15 | 3043 |
| 187 | 160 | 10 | 9 | 14.4 | 37.0 | 17 | 3777 |
| 275 | 230 | 20 | 13 | 16.4 | 44.4 | 25 | 4345543 |
| 297 | 246 | 18 | 17 | 17.2 | 35.3 | 33 | 1001001 |
| 495 | 410 | 30 | 28 | 17.2 | 35.3 | 55 | 11000100011 |
| 153 | 136 | 6 | 5 | 11.1 | 35.3 | 9 | 763 |
| 273 | 240 | 12 | 11 | 12.1 | 36.4 | 21 | 13077 |
| 459 | 408 | 18 | 14 | 11.1 | 35.3 | 27 | 100011011 |
| 525 | 470 | 20 | 18 | 10.5 | 36.4 | 35 | 4100001 |
| 391 | 363 | 11 | 9 | 7.2 | 39.3 | 17 | 5343 |
| 437 | 407 | 11 | 10 | 6.9 | 36.7 | 19 | 5343 |
| 675 | 628 | 18 | 14 | 7.0 | 38.3 | 27 | 4100001 |
| 725 | 676 | 18 | 15 | 6.8 | 36.7 | 29 | 4102041 |
| 897 | 836 | 22 | 20 | 6.8 | 36.1 | 39 | 34603145 |
| 1127 | 1045 | 33 | 25 | 7.3 | 40.2 | 49 | 107753475213 |

三、S 最佳循环码的充要条件

Fire 码虽然是一类容易构造的纠突发错误循环码，但其理论最佳度 Z 只有 $2/3$ 。下面讨论使式(9.1.7)或式(9.1.8)中等式成立的最佳码，即 S 最佳码的 $g(x)$ 所必须满足的条件。

定理 9.2.5(充要条件)^[6] 若 $g(x)=e(x)p(x)$ 生成二进制 S 最佳码，则当且仅当 $g(x)$ 满足以下条件：

(1) $e(x)$ 是一个 $b-1$ 次的非平方多项式，且不能被 x 除尽。

(2) $p(x)$ 是次数 $m \geq b+1$ 的本原多项式，且 $m \equiv 0 \pmod{m_e}$ ，这里 m_e 是 $e(x)$ 最小分离域 $\text{GF}(2^{m_e})$ 中的参数。

(3) 多项式 $p(x)$ 与 $e(x)$ 一起满足 AES 条件。

这里，AES 条件是：对于所有次数 $\leq b-1$ 的不同的突发错误图样多项式 $B_1(x)$ 、 $B_2(x)$ ，当

$$B_1(x) + x^l B_2(x) \equiv 0 \pmod{e(x)} \quad 0 \leq l < 2^m - 1$$

时

$$B_1(x) + x^l B_2(x) \not\equiv 0 \pmod{p(x)}$$

该定理其实是推论 9.1.2 的推广。其中，AES 条件其实就是推论 9.1.2，也就是两个长度 $\leq b$ 的突发之任意组合，不能是码的一个码字。而条件(1)、(2)是保证最佳码所必须的。

例如， $g(x)=(x+1)p(x)$ ， $p(x)$ 是本原多项式， $e(x)=x+1$ 是一个一次的非平方多项式，且在 $\text{GF}(2)$ 上完全分解， $m_e=1$ 。对所有长度 $b \leq 2$ 的两个突发 $\{1, 1+x\}$ 之任意组合，如果本原多项式 $p(x)$ 的次数 $m=b+1=3$ ，则 $p(x)$ 与 $e(x)$ 一起显然满足 AES 条件，因此，由 $g(x)=(x+1)p(x)$ 生成的码是一个纠正长度 ≤ 2 的 S 最佳码，它最早由艾布拉姆森 (Abramson) 证明了它的纠错能力，所以这类码称为 Abramson 码，它其实就是增余删信汉明码。当然扩展汉明码也是一个能纠正长度 ≤ 2 实发错误的 S 最佳码(请读者证明)。

事实上，纠正 t 个错误的 $[n, k]$ 循环码，若 $g(x)$ 中不含有 $(x+1)$ 因子，则用 $g'(x)=(x+1)g(x)$ 生成的 $[n, k-1]$ 增余删信码，能纠正任何长度 $\leq t+1$ 的突发错误，这类码也称为 Abramson 码。

如果要构造能纠正长度 $b \leq 3$ 的 S 最佳码，则满足条件(1)的 $e(x)=1+x+x^2$ ，它在 $\text{GF}(2^2)$ 域上完全分解，令分解域的 $m_e=2$ ，所以满足条件(2)的 $p(x)$ ，它的次数必须是大于 3 的偶数，即 $m=4, 6, 8, \dots$ ，且 $p(x)$ 是本原多项式。当然，还必须检验 $e(x)、p(x)$ 是否满足 AES 条件，在[6]中证明了对所有 $m \geq 4$ 的偶数次项的、满足 AES 条件的本原多项式是存在的。如 $m=4$ ，则 $p(x)=x^4+x+1$ ； $m=6$ ，则 $p(x)=x^6+x+1$ 等。

若要构造能纠正长度 $b \leq 4$ 的 S 最佳码，则满足条件(1)的 $e(x)$ 有 $1+x^3, 1+x+x^3$ 和 $1+x^2+x^3$ 。这里我们仅考虑用 $e(x)=1+x^3$ 时， $p(x)$ 所必须满足的条件。 $1+x^3$ 在 $\text{GF}(2^2)$ 上完全分解， $m_e=2$ ，所以满足条件(2)的 $p(x)$ ，它的次数 $m \geq 4+1$ ，必须是 ≥ 6 的偶数，但已证明对于 $m < 10$ 的偶数，不存在 $b \leq 4$ 的最佳码，因此仅当 $m \geq 10$ 的偶数，才存在能产生 S 最佳码的 $p(x)$ 。如 $m=10$ ， $p(x)=x^{10}+x^5+x^3+x^2+1$ ； $m=12$ ， $p(x)=x^{12}+x^{11}+x^9+x^8+x^7+x^5+x^2+x+1$ ； $m=14$ ， $p(x)=x^{14}+x^8+x^6+x+1$ 等等。

在文献[6]中还详细分析了纠正长度 $b \leq 5$ 时的 S 最佳码的 $g(x)$ 所应满足的条件，并且更进一步证明了纠正任何长度的最佳码是存在的，并存在无限多个。而在文献[7]中，则把[6]的结果推广到 q 进制码。

这里所讨论的最佳码，并不是 R 最佳码。事实上，只有当 $m=b+1$ 的 $p(x)$ 存在时，由此得到的码，既是 R 最佳码，又是 S 最佳码。如由 $g(x)=(x^2+1)(x^4+x+1)$ 生成的[15, 9]码。而一般情况下，S 最佳码不一定是 R 最佳码，反之亦然。由于 R 限和 S 限均是必要条件，因此对任何线性码必须满足 R 限，对任何循环码则必须同时满足 R 限和 S 限。

四、R 最佳码的一种构造方法

定理 9.2.6 用 $g(x) = 1 + \sum_{i=0}^{m-1} x^{2^i}$ 生成的 $n = 2^m - 1$, $n-k = 2^{m-1}$ 的二进制码是一循环码，它能纠正长度 $b \leq 2^{m-2}$ 的单个突发错误。

证明 若 $b = 2^{m-2}$, 则码长 $n = 4b - 1$, 校验位 $n-k = 2b$ 。首先，容易证明用 $g(x) = 1 + \sum_{i=0}^{m-1} x^{2^i}$ 生成的码是一循环码(作为习题)；其次，我们只要证明任何两个长度 $\leq b$ 的突发， $B_1(x)$ 和 $B_2(x)$ 的任意组合 $x^j B_1(x) + B_2(x)$, 不是一个码字即可。这里， $\partial \circ B_1(x) \leq b - 1$, $\partial \circ B_2(x) \leq b - 1$, $b + 1 \leq j \leq 2b - 1$, 也就是 $2^{m-2} + 1 \leq j \leq 2^{m-1} - 1$ 。这就是说， $x^j B_1(x)$ 的位置至多有 b 个，因为 $g(x) = 1 + x + x^2 + x^4 + \dots + x^{2^{m-1}} = 1 + x + x^2 + \dots + x^b + x^{2b}$, 最高次项 x^{2b} 到第二项 x^b 之间有 $b + 1$ 位，因此当用 $g(x)$ 除 $x^j B_1(x)$ 时，所得的余项，也即伴随式多项式的次数至少为 j , $b + 1 \leq j \leq 2b - 1$ 。此外，余项也必是 $1 + \sum_{i=0}^{2^{m-2}} x^{2^i}$ 加上另一项次数至少为 j 、但不在 $\sum_{i=0}^{2^{m-2}} x^{2^i}$ 的形式中，所以， $x^j B_1(x) + B_2(x)$ 不能被 $g(x)$ 除尽，故不是码字。 ■

例如， $m=4$ ，则得到一个码长 $n=2^4 - 1 = 15$, 校验位 $n-k = 2^{m-1} = 8$ 的[15, 7]码，它的 $g(x) = x^8 + x^4 + x^2 + x + 1$ 能纠正长度 $\leq b = 2^{m-2} = 4$ 的突发错误。

这类码的 $Z = 2b/(n-k) = 2 \cdot 2^{m-2}/2^{m-1} = 1$, 显然是 R 最佳码。也可把这种构造方法从二进制推广到多进制，此时 q 进制码的码长 $n=q^m - 1$, $n-k=q^{m-1}$, 能纠正 $\leq b=q^{m-2}$ 长的突发错误，码的生成多项式 $g(x) = \sum_{i=0}^{m-1} x^{q^i} - a$, $a \in GF(q)$, 且 $a \neq 0$ 。若 $q=2^m$, 则这类码是循环码；否则它们是准循环码。

§ 9.3 纠定段(字节)突发错误码

设某一 $[n, k]$ 码，它的码长 $n=\sigma m$ 是某一整数 m 的倍数。则码多项式可表示如下：

$$C(x) = c_0 + c_1 x + c_2 x^2 + \dots + c_{m-1} x^{m-1} + c_m x^m \\ + \dots + c_{2m-1} x^{2m-1} + \dots + c_{\sigma m-1} x^{\sigma m-1}$$

定义 m 个连续码元 $c_0 c_1 \dots c_{m-1}$ 为一段。一个长度 $\leq \lambda m$ 并局限于连续 λ 段的突发，我们定义为一个定段(字节)突发错误。能纠正这种定段突发错误的码称为纠定段(字节)突发错误

码。长度 $\leq (\lambda-1)m+1$ 的突发至多只影响码组中的 λ 段，因此一个能纠正 λm 长的定段突发错误码，就能够纠正任何长度 $\leq (\lambda-1)m+1$ 的单个突发错误。

纠定段突发错误码有三类，一是伯顿(Burton)码，二是 $GF(q)$ 上的($q>2$)RS码，三是孙子定理(CR)码。

一、Burton 码

Burton 码是 1969 年首先由伯顿提出的一类纠定段突发错误的二进制循环码，它的构造方法完全类似于 Fire 码。

定理 9.3.1 令 $p(x)$ 是一个周期为 e 的 b 次既约多项式，且 $(p(x), x^b - 1) = 1$ ，则由

$$g(x) = (x^b - 1)p(x) \quad (9.3.1)$$

生成的 $[n, n-2b]$ 码称 Burton 码。其码长 $n=eb$ ，信息位 $k=n-2b$ ，能纠正长度 $\leq b$ 的定段突发错误。

证明 由推论 9.1.2 可知，只要证明任何两个长度 $\leq b$ 的定段突发 $B_1(x)$ 和 $B_2(x)$ 的任何组合，不是一个码字即可，可用反证法证明。设 $B_1(x)$ 和 $B_2(x)$ 的组合是由 $g(x)=p(x)(x^b-1)$ 生成的 Burton 码的一个码字 $C(x)$ ，则

$$\begin{aligned} C(x) &= B_1(x) + x^{ib}B_2(x) \quad i = 1, 2, \dots, e-1 \\ &= q_1(x)g(x) = q_1(x)(x^b - 1)p(x) \end{aligned} \quad (9.3.2)$$

因为 $x^b - 1 | (x^{ib} - 1)B_2(x)$ ，故

$$(x^{ib} - 1)B_2(x) = (x^b - 1)q'_2(x)$$

由上两式得

$$B_1(x) + B_2(x) = q_1(x)p(x)(x^b - 1) - (x^b - 1)q'_2(x) = (x^b - 1)q_2(x)$$

因 $\partial \circ B_1(x)$ 和 $\partial \circ B_2(x)$ 均小于 b ，所以要上式成立，只有 $B_1(x) = -B_2(x)$ 才有可能。因而， $B_1(x)$ 与 $B_2(x)$ 的下标可取掉，式(9.3.2)成为

$$B(x)(x^b - 1) = q'_1(x)p(x)(x^b - 1)$$

由于 $\partial \circ B(x) < b$ ， $p(x)$ 是 b 次既约多项式，因此为了使上式成立， $p(x)$ 必须除尽 $x^b - 1$ ，而 $x^b - 1 | x^{ib} - 1$ 。但是 $i < e$ ， $p(x)$ 的周期为 e ，所以， $p(x)(x^b - 1)$ 除尽 $x^{ib} - 1$ 形式的最低次多项式是 $x^e - 1$ ，因而 $i \geq e$ ，这与原假设的 $i < e$ 相矛盾，因而 $C(x)$ 不是一个码字，故 Burton 码能纠正长度 $\leq b$ 的单个定段突发错误。■

若要纠正 λ 段突发错误，则可把 Burton 码进行交错。一般说来，用交错 Burton 码方法构造纠正单个突发错误的码，其最佳程度 Z 比 Fire 码大，这将在交错码一节讨论。

二、RS 码

$GF(q^m)$ 上的能纠正 t 个错误的 $[q^m-1, q^m-1-2t]$ RS 码，显然能纠正 $GF(q^m)$ 上的长度 $\leq t$ 的突发错误，其最佳度 $Z=1$ ，可知是一个 $GF(q^m)$ 上的最佳纠突发错误码。但是，如果每个码元符号用 $GF(q)$ 上的元素表示，则该码能纠正长度 $\leq m$ 的 t 个定段突发错误，或者纠正长度 $\leq mt$ 的单个定段突发错误，或者纠正长度 $\leq (t-1)m+1$ 的任何单个突发错误，或纠正 $\lfloor t/2 \rfloor$ 个长度 $\leq m+1$ 的突发错误。若 RS 码用来纠正 $GF(q)$ 上的任何单个突发错误，则

$$Z = \frac{2(t-1)m+2}{2tm} \quad (9.3.3)$$

当 t 较大时, Z 接近 1。所以, RS 码是一个接近最佳的准最佳纠突发错误码。

$GF(q^m)$ 上的 RS 码除了上述纠错能力之外, 还能纠正 $GF(q)$ 上的某些随机错误与突发错误之组合图样。由于一个长度 $b \leq mt - (m-1)$ 的突发, 至多只影响 $GF(q^m)$ 上的 t 个符号, 因此, 一个符号取自 $GF(q^m)$ 上的纠正 t 个随机错误的 RS 码, 能纠正 $GF(q)$ 上的 p 个突发错误, 且每个突发的长度

$$l \leq \left(\left\lfloor \frac{t}{p} \right\rfloor - 1 \right) m + 1 \quad (9.3.4)$$

从上面讨论看出, RS 码是一类有很强纠错能力的码。

三、孙子定理码(CR 码)

与 RS 码一样, CR 码也是符号在 $GF(q^m)$ 上的, 纠正 t 个随机错误分组码, 因而码纠正突发能力与 RS 码相同。不同的是 CR 码是用孙子定理方法构造出来的。

定理 9.3.2 令 $I(x)$ 表示次数 $\leq k-1$ 的 $GF(q^m)$ 上的多项式, $m_i(x)$ 表示同一域上的 d_i 次多项式, 且 $(m_i(x), m_j(x)) = 1$, $\sum_{i=1}^n d_i > k-1$, $i = 1, 2, \dots, n$, $i \neq j$, 则根据孙子定理, $I(x)$ 可以由以下同余组唯一确定:

$$I(x) \equiv r_i(x) \pmod{m_i(x)} \quad i = 1, 2, \dots, n$$

由此我们可以构造 CR 码。设 $I(x)$ 是符号取自 $GF(q^m)$ 上的, 由 k 个信息符号作为系数的 $k-1$ 次多项式。由于 $x^{q^m-1}-1$ 在 $GF(q^m)$ 域上完全分解, 所以若选择 $x^{q^m-1}-1$ 的所有一次因子作为 $m_i(x)$, 则

$$m_i(x) = (x - \alpha^i) \quad i = 0, 1, \dots, q^m - 2$$

式中, α 是 $GF(q^m)$ 上的本原域元素, $(m_i(x), m_j(x)) = 1$ 。由此, 根据信息多项式 $I(x)$ 和 $m_i(x)$ 可以决定 $n = q^m - 1$ 个余式:

$$I(x) \equiv r_i(x) \pmod{m_i(x)} \quad i = 0, 1, 2, \dots, n-1$$

这些余式的集合 $\{r_{n-1}, r_{n-2}, \dots, r_1, r_0\}$ 就是 $GF(q^m)$ 上一个 CR 码的码字, 有码长 $n = q^m - 1$, 信息位 k , 且是非系统码。

例 9.2 $GF(2^3)$ 中, $x^{2^3-1}-1$ 可完全分解为

$$x^7 - 1 = (x - 1)(x - \alpha)(x - \alpha^2)(x - \alpha^3)(x - \alpha^4)(x - \alpha^5)(x - \alpha^6)$$

α 是 $GF(2^3)$ 中的本原元, 所以, 可以构造一个 $GF(2^3)$ 上的, $n = 2^m - 1 = 2^3 - 1 = 7$, $k = 3$ 的 CR 码。设已给定信息多项式:

$$I(x) = \alpha^4 x^2 + \alpha x + \alpha$$

则:

$$I(x) \equiv r_0(x) = \alpha^4 \pmod{x - 1}$$

$$I(x) \equiv r_1(x) = \alpha^3 \pmod{x - \alpha}$$

$$I(x) \equiv r_2(x) = \alpha^3 \pmod{x - \alpha^2}$$

$$I(x) \equiv r_3(x) = \alpha^5 \pmod{x - \alpha^3}$$

$$I(x) \equiv r_4(x) = \alpha \pmod{x - \alpha^4}$$

$$I(x) \equiv r_5(x) = \alpha^4 \pmod{x - \alpha^5}$$

$$I(x) \equiv r_6(x) = \alpha^5 \pmod{x - \alpha^6}$$

由此得到 $\text{GF}(2^3)$ 上 $[7, 3]$ CR 码的一个码字: $(\alpha^5 \alpha^4 \alpha \alpha^5 \alpha^6 \alpha^3 \alpha^4)$ 。相应的码多项式为

$$C(x) = \alpha^5 x^6 + \alpha^4 x^5 + \alpha x^4 + \alpha^5 x^3 + \alpha^3 x^2 + \alpha^3 x + \alpha^4$$

CR 码可以化成循环码的形式。设:

$$I(x) = I_{k-1} x^{k-1} + \cdots + I_1 x + I_0$$

$$r_i(x) \equiv I(x) \pmod{x - \alpha^i} \quad \alpha^i \in \text{GF}(q^n)$$

若 $x = \alpha^i$, 则 $r_i(\alpha^i) = I(\alpha^i)$ 。

设 $\text{GF}(q^n)$ 上一个 $[n, k]$ 循环码的生成矩阵为

$$\mathbf{G} = \begin{bmatrix} (\alpha^{k-1})^{n-1} & (\alpha^{k-1})^{n-2} & \cdots & \alpha^{k-1} & \alpha^0 \\ (\alpha^{k-2})^{n-1} & (\alpha^{k-2})^{n-2} & \cdots & \alpha^{k-2} & \alpha^0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha^{n-1} & \alpha^{n-2} & \cdots & \alpha & \alpha^0 \\ \alpha^0 & \alpha^0 & \cdots & \alpha^0 & \alpha^0 \end{bmatrix} = \begin{bmatrix} \alpha^{(k-1)(n-1)} & \alpha^{(k-1)(n-2)} & \cdots & \alpha^{(k-1)} & \alpha^0 \\ \alpha^{(k-2)(n-1)} & \alpha^{(k-2)(n-2)} & \cdots & \alpha^{(k-2)} & \alpha^0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha^{(n-1)} & \alpha^{(n-2)} & \cdots & \alpha & \alpha^0 \\ \alpha^0 & \alpha^0 & \cdots & \alpha^0 & \alpha^0 \end{bmatrix} \quad (9.3.5)$$

则信息组 $\mathbf{I}_k = (I_{k-1}, I_{k-2}, \dots, I_1, I_0)$ 与 \mathbf{G} 相乘, 得到码字

$$\begin{aligned} \mathbf{C}_k &= \mathbf{I}_k \mathbf{G} \\ &= (I(x)|_{x=\alpha^{n-1}}, I(x)|_{x=\alpha^{n-2}}, \dots, I(x)|_{x=\alpha}, I(x)|_{x=1}) \\ &= (r_{n-1}, r_{n-2}, \dots, r_1, r_0) \end{aligned}$$

它正好是 CR 码的一个码字。

$[n, k]$ 循环码的生成矩阵 \mathbf{G} 与它的 \mathbf{H} 矩阵互为零空间。因此, 若以式(9.3.5)的 \mathbf{G} 矩阵作为码的校验矩阵, 则该码以 $\alpha^0, \alpha, \dots, \alpha^{k-1}$ 为根, 而这正是 RS 码。因此, 以 $\text{GF}(q^n)$ 中的 $x^{q^n-1} - 1$ 的完全分解式的一次因式为 $m_i(x)$, 所得出的 CR 码就是 $\text{GF}(q^n)$ 中 RS 码的对偶码, 而 RS 码的对偶码仍为 RS 码, 因此该类 CR 码就是 RS 码。构造 CR 码的因子 $m_i(x)$, 除了一次因式外, 也可用更高的因子, 例如所有 $m_i(x)$ 都用两两互素的 $d_i (> 1)$ 次因式, 这样构造的 CR 码, 可以证明其纠错能力比用一次因式构造的 CR 码要弱得多。CR 码除了包含 RS 码作为它的一个子类之外, 还包含很多其它广义码如 Goppa 码、交替码等, 有关方面的内容可参阅有关文献。

下面讨论 CR 码的译码及其纠错能力。所有 $m_i(x)$ 用一次因式构造的 CR 码就是 RS 码, 因而该类 CR 码的纠错能力与译码完全同 RS 码。除此之外, CR 码也还可用其它译码方法译码。

设被传递的 CR 码码字中出现了 t 个错误, 则其余 $n-t$ 个码元(即 $m_i(x)$ 的余式)是正确的。因而, 根据孙子定理, $I(x)$ 可以由接收到的任意 k 个值再次构造出:

$$I(x) \equiv \sum_{i=1}^k M_i(x) M'_i(x) r'_i(x) \pmod{\prod_{j=1}^k m_j(x)}$$

式中, $r'(x)$ 是接收到的余式(即相应的码元), $M_i(x) = \prod_{j=1, j \neq i}^k m_j(x)$, $M'_i(x) M_i(x) \equiv 1 \pmod{m_i(x)}$ 。因此, 可以有 $\binom{n}{k}$ 种方法决定 $I(x)$ 。但由正确的 $n-t$ 个码元中选取的 k 个

余式, 得到的 $I(x)$ 是正确的, 所以有 $\binom{n-t}{k}$ 种不同方法得出的 $I(x)$ 应有相同的值, 并且是正确的 $I(x)$ 。如果 k 个余式中有一个余式是挑了一个错误的余式, 则由此构造出的 $I'(x)$ 就不是原来发送的 $I(x)$ 。而任意 $k-1$ 个正确余式与任一不正确余式决定的 $I'(x)$, 至多只有 $\binom{k-1+t}{k}$ 个, 所以若

$$n - t > k - 1 + t \quad (9.3.6)$$

则得到正确的相同 $I(x)$ 的数目, 超过不正确 $I'(x)$ 的数目。故根据再次构造出 $I(x)$ 、 $I'(x)$ 相同数目的多少, 就能正确求得 $I(x)$ 。因而, 某一码参数若能使式(9.3.6)满足 $n-t \geq k-1+t+1$, $n-k \geq 2t$, $k \leq n-2t$, 或 $t \leq \lfloor (n-k)/2 \rfloor$ 时, 则该码能纠正所有 t 个错误。如例 9.2 中的 [7, 3]CR 码就能纠正任意两个错误, 或纠正长度 ≤ 4 的二进制上的单个突发错误, 以及纠正长度 ≤ 3 的两个定段突发错误。

§ 9.4 交错码与乘积码

实际信道中产生的错误往往是突发错误或突发错误与随机错误并存, 如短波、散射和有线交换等信道中。在这类信道中应用纠随机错误码纠错, 效果显然是不好的, 但是如果首先能把突发错误离散成随机错误, 然后再用纠随机错误的码纠错, 则能取得明显效果。本节将介绍几类常用的这种把突发错误离散成随机错误, 然后纠错的方法。

一、交错码

交错方法是一种很实用而且常用的构造码的方法, 它能把比较长的突发错误或多个突发错误离散成随机错误。用交错方法构造出的码称为交错码。

定义 9.4.1 若把由 $g(x)$ 生成的, 纠 t 个随机错误或纠正 b 长突发错误的 $[n, k]$ 线性分组码, 排成如图 9-1 的码阵, 每行是 $[n, k]$ 码的一个码字共 i 行、 n 列, 则该码阵就定义为 $[ni, ki]$ 交错码的一个码字。称 i 为交错次数或交错度; 称每一行为交错码的行码或子码。并规定传送时, 以列的次序自左至右传输: $(a_{1(n-1)} a_{2(n-1)} \cdots a_{i(n-1)} a_{1(n-2)} \cdots a_{i(n-2)} \cdots a_{10} a_{20} \cdots a_{i0})$ 。

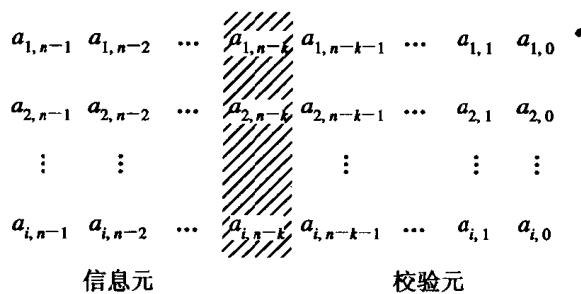


图 9-1 交错码阵

由以上定义可知, 交错码其实就是把 $[n, k]$ 分组码的 i 个码字, 使每个码字相邻码元之间均相隔 i 位。因此, 当接收端收到交错码的码字后, 若仍排成图 9-1 的码阵, 并以行为单位, 按 $[n, k]$ 子码的方式译码, 则就把信道中的错误分散到每个行码中去了。因此, 若行

码能纠正 t 个随机错误或 b 长突发错误，则 $[ni, ki]$ 交错码就能纠正所有长度 $\leq i_t$ 或 $\leq ib$ 的突发错误，如图 9-1 中的阴影线所示。

交错码除了能纠正很长的单个突发错误以外，也能纠正较短的多个突发错误。若 $[n, k]$ 行码能纠正 t 个随机错误，则 $[ni, ki]$ 交错码能纠正 t 个长度 $\leq i$ 的突发错误，或纠正长度 $\leq it$ 的单个突发错误。因此，正如上面所讲的，交错的作用就是把长的突发错误分散到每个行码中去，即把错误离散化，然后由行码再把这些错误进行纠正。因此，交错度越大，则离散度越大。从这个意义上讲，交错方法是一种时间扩散技术，它把信道错误的相关性减小，当交错度足够大时，就可把突发错误离散为随机错误，从而可以用纠随机错误或纠较短突发错误的码作为行码，构成交错码以纠正较长的突发错误或突发和随机错误的组合。并且，交错度足够大的交错码，对信道干扰的统计特性并不敏感，且由于应用的行码一般都能用较简单的方法译码。所有这些优点，使得交错技术在实际信道如短波、散射、有线（特别是交换线路）等有记忆信道的前向纠错系统中，得到了极为广泛的应用。

下面定理说明若行码是循环码，则交错码也是循环码，且它的生成多项式与行码的生成多项式有极为密切的关系。

定理 9.4.1 设由 $g(x)$ 生成的 $[n, k]$ 循环码，能纠正长度 $\leq b$ 的所有突发，则由 $g(x^i)$ 生成的 $[ni, ki]$ 码也是循环码，有纠突发能力为 bi 。

证明 由循环码定义可知， $[n, k]$ 码的生成多项式 $g(x) | x^n - 1$ ， $\partial \circ g(x) = n - k$ 。
 $\partial \circ g(x^i) = i(n - k)$ ，显然， $g(x^i) | x^{ni} - 1$ ，因此 $g(x^i)$ 产生的码也是循环码。

由 $g(x^i)$ 生成的码，它的一致校验多项式 $h(x^i)$ 。由 $h(x^i), x^i h(x^i), x^{2i} h(x^i), \dots, x^{(n-k-1)i} h(x^i)$ 组成了 $(n - k)$ 个相应于 $x^{(n-k)i}, x^{(n-k+1)i}, \dots, x^{(n-1)i}$ 信息位上的一致校验多项式。而 $x^j h(x^i), x^{j+i} h(x^i), x^{j+2i} h(x^i), \dots, x^{j+(n-k-1)i} h(x^i)$ ($0 \leq j \leq i - 1$)，则组成了相应于对 $x^{(n-k)i+j}, x^{(n-k+1)i+j}, \dots, x^{(n-1)i+j}$ 信息位的 $(n - k)$ 个互不相关的一致校验多项式，这意味着上面的这些校验关系所产生的码，可看成是用 $g(x^i)$ 生成的一个如下排列的码字：

| | | | | | | | |
|------------|----------------|----------|------------------|----------------|----------|------------|-----------|
| x^{ni-1} | $x^{(n-1)i-1}$ | \dots | $x^{(n-k+1)i-1}$ | $x^{(n-k)i-1}$ | \dots | x^{2i-1} | x^{i-1} |
| x^{ni-1} | $x^{(n-1)i-2}$ | \dots | $x^{(n-k+1)i-2}$ | $x^{(n-k)i-2}$ | \dots | x^{2i-2} | x^{i-2} |
| \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots | \vdots |
| x^{ni-i} | $x^{(n-1)i-i}$ | \dots | $x^{(n-k+1)i-i}$ | $x^{(n-k)i-i}$ | \dots | x^i | x^0 |
| 信息位 | | | | 校验位 | | | |

显然，上述阵中的每一行就是 $g(x)$ 生成的 $[n, k]$ 码的一个码字，且该阵的结构完全与图 9-1 的码阵完全相同。因而，由 $g(x^i)$ 生成的码就是 $g(x)$ 生成的 $[n, k]$ 码作为行码所组成的 i 次交错码。所以长度 $\leq bi$ 的任何突发错误，至多只影响每行相邻 b 个码元，而这在 $[n, k]$ 码的纠错能力以内，故 $[ni, ki]$ 码能纠正长度 $\leq bi$ 的突发错误。■

若 $[n, k]$ 码是一个缩短循环码，则同理可证交错码也是一个缩短循环码。

若 $[n, k]$ 码的纠突发能力为 b ， $Z = 2b/(n - k)$ ，用该码作行码交错 i 次后，所得的 $[ni, ki]$ 交错码，其纠突发能力为 ib ，最佳度

$$Z = \frac{2ib}{i(n - k)} = \frac{2b}{(n - k)}$$

与 $[n, k]$ 行码相同。所以，为了得到较长的最佳码，只要利用较短的最佳码作为行码进行交错即可，并且其码率 R 也与行码相同。

若用纠定段突发错误的码如 Burton 码、RS 码、CR 码作行码进行交错，且交错码阵中以每一码段为单位进行发送。也就是交错码中的每一行是纠定段突发错误码的一个码组，共有 i 行，传送时发送第一行的第一个码段(m 个码元)，然后发送第二行的第一个码段等，这样得到的交错码称为**码段交错码**。

一个 $[n, n-2b]$ Burton 码，进行 i 次交错后，可得到一个 $[ni, i(n-2b)]$ 码，它能纠正长度 $\leq ib$ 的定段突发错误。由前可知，该交错码也能纠正任何长度 $\leq (i-1)b+1$ 的单个突发错误，因此该交错码的最佳程度

$$Z = \frac{2((i-1)b+1)}{2bi} = 1 - \frac{1}{i} \left(\frac{b-1}{b} \right) \quad (9.4.1)$$

显而易见，当交错度 i 增加时，交错纠定段突发错误的最佳程度也增加，当 $i \rightarrow \infty$ 时 $Z \rightarrow 1$ 。与 Fire 码相比，应用纠定段突发错误码如 Burton 码、RS 以及 CR 码作行码进行交错后，在相同码长和信息位下，其纠错能力要比 Fire 码强，性能更优越。

综上所述，我们可以在表 7-1、表 7-2 和表 9-1、表 9-2 所列的码中，挑选出符合要求的码长较短的最佳或准最佳码，然后应用交错方法，就可得到码长更长的最佳或准最佳码，或者得到纠多个突发错误码。并且这类码的码率 R 仍与原来的行码相同，实现起来也并不复杂，只要在原行码编译码器的基础上，加上一个 ni 位存贮器及相应的控制电路即成。

用上述交错阵的交错方法，交错度是固定的，因此是一种周期交错方法。如果交错度用一伪随机序列控制而变化，则得到的是一种伪随机交错，这种方法能抗击周期性干扰。

二、乘积码

图 9-1 交错码阵中的每一行是 $C_1 = [n_1, k_1, d_1]$ 线性码的一个码字，如果码阵中的每列是另一个线性码 $C_2 = [n_2, k_2, d_2]$ 的一个码字，如图 9-2 所示。这样得到的交错码阵共有 n_2 行 n_1 列，是一个 $[n_1 n_2, k_1 k_2]$ 线性码的码组，它是 C_1 和 C_2 码的直积 $C_1 \otimes C_2$ ，因而称这类交错码为**乘积码**。

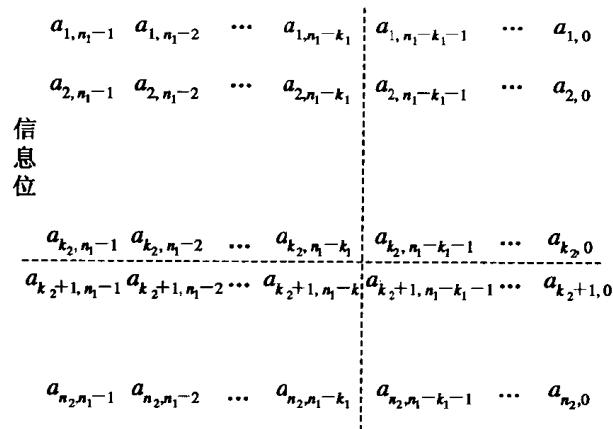


图 9-2 乘积码码阵图

由此可见，交错码仅由一个子码（行码）构成，而乘积码 $C_1 \otimes C_2$ 由两个子码（行码 C_1 和列码 C_2 ）构成。因此，有时称这类乘积码为**二维码**。

乘积码有两种传输数据的方法：一是同交错码的传输方法一样，可以按行的次序逐行传送，或按列自左至右逐列传输；另一是按码阵的对角线次序传送数据，但后者传输方法所得到的码与前者是不一样的。

如果乘积码以行或列的次序逐行或逐列传输，则译码时可先以行按 C_1 码，后按列以 C_2 码进行译码；也可以先以列按 C_2 码译码，后再以行按 C_1 码译码；因此需两级译码。可知乘积码译码器的复杂性完全决定于行码和列码译码器的复杂性，以及 $n_1 n_2$ 存贮器的大小。

若 C_1 码能纠正长度 $\leq b_1$ 的突发错误， C_2 码能纠正长度 $\leq b_2$ 的突发错误，则由交错码的纠突发能力可知， $C_1 \otimes C_2$ 乘积码能纠正长为

$$b \leq \max(b_1 n_2, b_2 n_1) \quad (9.4.2)$$

的突发错误。

如果码 C_1 的最小距离为 d_1 ，码 C_2 的最小距离为 d_2 ，则由图 9-2 的码阵可看出，非全为 0 阵中至少有一行非全为 0 的码字，该行中至少有 d_1 个非 0 码元，因而至少有 d_1 个非全为 0 的列，而这些非全为 0 列，每列至少有 d_2 个非 0 码元，因而码阵中至少有 $d_1 d_2$ 个非 0 码元，由此可知 $C_1 \otimes C_2$ 乘积码有最小距离为 $d_1 d_2$ ，因而，该乘积码能纠正 $\lfloor (d_1 d_2 - 1)/2 \rfloor$ 个随机错误。

若码 C_1 能纠正 $t_1 = \lfloor (d_1 - 1)/2 \rfloor$ 个随机错误，码 C_2 能纠正 $t_2 = \lfloor (d_2 - 1)/2 \rfloor$ 个随机错误，则可以证明乘积码 $C_1 \otimes C_2$ 能纠正长为

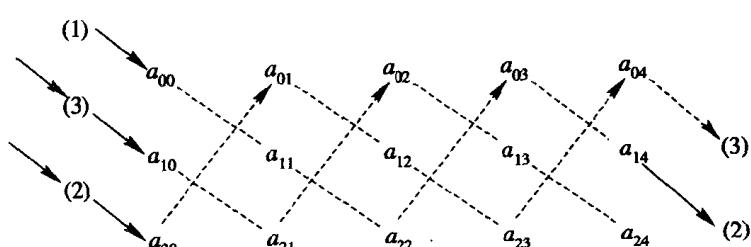
$$b \leq \max(n_1 t_2, n_2 t_1) \quad (9.4.3)$$

的突发错误。

除了上述所列的纠错能力以外，乘积码还能纠正其它大量的随机和突发错误图样，由于乘积码纠错能力很强，因而这类码得到了很大发展，例如可以从二维扩展到由多个子码组成的多维乘积码，以及由一般乘积码扩展到循环乘积码等。

如果乘积码的每个子码（行码 C_1 、列码 C_2 ）都是循环码，并且传输时不是以行或列为单位传输，而是以码阵的对角线次序传送，则得到的是循环乘积码。

例如， C_1 码是一个长为 5 的循环码， C_2 码是一个长为 3 的循环码，则 $C_1 \otimes C_2$ 码的二维码阵及其码元的传输次序为



因而循环乘积码的一个码字为： $a_{00} a_{11} a_{22} a_{03} a_{14} a_{20} a_{01} a_{12} a_{23} a_{04} a_{10} a_{21} a_{02} a_{13} a_{24}$ 。从该例看出，发送时是先以码阵第一行第一列的第一个码元(a_{00})，以对角线的次序传至最后一行的一个码元(a_{22})，然后以该码元所处列的下列的第一个码元(a_{03})开始，以对角线的次序传送至最后一列的一个码元(a_{04})，再以该码元所处的行的下一行的第一个码元(a_{20})开始，以对角线

的次序传送，如此等等，直至发完最后一个码元(a_{24})为止，如阵中虚线箭头所示。

可以证明，若 $C_1 = [n_1, k_1]$ 和 $C_2 = [n_2, k_2]$ 循环码的码长互素 ($n_1, n_2 = 1$)，则按上述规则发送的循环乘积码 $C_1 \otimes C_2$ 的码字也是一个循环码。如果 C_1 和 C_2 循环码的生成多项式分别是 $g_1(x)$ 和 $g_2(x)$ ，则循环乘积码 $C_1 \otimes C_2$ 的生成多项式

$$g(x) = \text{LCM}\{\text{GCD}[(g_1(x^{bn_2}), x^{n_1 n_2} - 1), \text{GCD}[g_2(x^{an_1}), x^{n_1 n_2} - 1]]\} \quad (9.4.4)$$

$$g(x) = \text{GCD}[g_1(x^{bn_2}) g_2(x^{an_1}), x^{n_1 n_2} - 1]$$

式中， a, b 由欧几里德算法确定：

$$an_1 + bn_2 = (n_1, n_2) = 1 \quad (9.4.5)$$

如果从交错码的观点出发，则循环乘积码的生成多项式也可写成

$$g(x) = \text{LCM}[g_1(x^{n_2}), g_2(x^{n_1})] \quad (9.4.6)$$

与多维乘积码一样，若把二维循环乘积码 $C_1 \otimes C_2$ 的每个码字作行，另一循环码 C_3 的码字作列，组成一个码阵，并且以对角线的次序组成码字，就得到了一个三维循环乘积码 $C_1 \otimes C_2 \otimes C_3$ 。因此，可把这种构造二维循环乘积码的方法和生成多项式的表示，推广到更高维的循环乘积码中去，当然要求所有子码的码长必须两两互素。

可以证明如果循环乘积码的一个子码 C_1 是一步大数逻辑可译码，有最小距离为 d_1 。另一个子码 C_2 是 L 步大数可译码，并有最小距离为 d_2 ，则二维循环乘积码 $C_1 \otimes C_2$ 是 L 步大数逻辑可译码，能实现 $d_1 d_2$ 的纠错能力。

三、奇偶校验乘积码

在乘积码中，如果行码和列码都是奇偶校验码，则该乘积码的最小汉明距离仅为 4。若按交错码的方法逐行或逐列传输，则该乘积码只能纠一个随机错误，没有什么纠突发错误能力。但是，如果按码阵的对角线次序传送，则纠突发能力得到很大提高。

以 $A(n_1, n_2, s)$ 表示一个有 n_1 行 n_2 列的乘积码 $C_1 \otimes C_2$ ，传输时以对角线次序传输，每一对角线之间隔 s 位，且 $(n_1, s) = 1$ ， $1 \leq s \leq n_2 - 1$ 。如 $A(5, 11, 3)$ 码的码阵如下所示。

| | | | | | | | | | | |
|----|----|----|----|----|----|----|----|----|----|----|
| 0 | 20 | 40 | 5 | 25 | 45 | 10 | 30 | 50 | 15 | 35 |
| 36 | 1 | 21 | 41 | 6 | 26 | 46 | 11 | 31 | 51 | 16 |
| 17 | 37 | 2 | 22 | 42 | 7 | 27 | 47 | 12 | 32 | 52 |
| 53 | 18 | 38 | 3 | 23 | 43 | 8 | 28 | 48 | 13 | 33 |
| 34 | 54 | 19 | 39 | 4 | 24 | 44 | 9 | 29 | 49 | 14 |

传送时按上述的 0, 1, 2, … 序号发送数据，由该码阵看出，0 至 4 的对角线与 5 至 9 的对角线相隔 $s = 3$ 位。35 至 39 与 40 至 44 对角线也相隔 $s = 3$ 位，这种传送方法与上例中 $n_1 = 3, n_2 = 5$ 的循环乘积码的码阵传送方法相同。

已证明如 $n_2 \geq 2n_1 - 3, s = 1$ ，则 $A(n_1, 2n_1 - 3, 1)$ 码能纠正长度 $\leq n_1 - 1$ 的突发错误^[8]，而 $A(n_1, n_2 \geq 2n_1 + 1, -1)$ 码能纠正长度 $\leq n_1$ 的任何单个突发错误^[9]，这里 $s = -1$ ，意味着 $s = n_2 - 1$ 。但是这两类码的最佳度 Z 约为 $2/3$ ，与 Fire 码差不多。下面的定理给出了一类 Z 接近 $4/5$ 的码。

定理 9.4.2^[10] 令 $n_1 = 4a + b + 2, n_2 = 6a + 2b + 5, a \geq 1, b \geq 0$ ，且 $b \neq 1$ 。则 $A(n_1, n_2, -2)$ 码能纠正长度 $\leq n_1$ 的任何单个突发错误。

满足定理 9.4.2 条件的 $A(n_1, n_2, -2)$ 码的最佳度

$$Z = \frac{8a + 2b + 4}{10a + 3b + 6} \approx 4/5 \quad \text{当 } a \rightarrow \infty$$

如果列码 C_1 是 $[n_1, n_1 - 1, 2]$ 奇偶校验码，行码 C_2 是 $[n_2, k_2, 2t]$ 的偶重量码，则当行、列码的码长 n_1, n_2 满足一定条件，并用对角线方法传输 $C_1 \otimes C_2$ 乘积码的码阵时，该乘积码能纠正 t 个突发错误。

定理 9.4.3^[11] 设 n_1 和 n_2 是整数，且 $n_2 \geq 2tn_1 + 1$ ，则由 $[n_1, n_1 - 1, 2]$ 码和 $[n_2, k_2, 2t]$ 码组成的 $A(n_1, n_2, -1, t)$ 乘积码能纠正 t 个长度 $\leq n_1$ 的突发错误。

这里， $A(n_1, n_2, -1, t)$ 表示乘积码阵由 n_1 行和 n_2 列组成，按 $s = -1$ 的方法对角线传输码字数据，行码的最小距离为 $2t$ 。显然，当 $t = 1$ 时，就归结为前面提到的 $A(n_1, n_2 \geq 2n_1 + 1, -1)$ 码。

定理 9.4.4^[11] 令 n_1 和 n_2 是整数，当且仅当 $n_2 \geq 2t(n_1 - 2) + 1$ 时，则由 $[n_1, n_1 - 1, 2]$ 码和 $[n_2, k_2, 2t]$ 码组成的 $A(n_1, n_2, 1, t)$ 乘积码，能纠正 t 个长度 $\leq n_1 - 1$ 的突发错误。

例如， $t = 2, n_1 = 7, n_2 = 2tn_1 + 1 = 29, s = -1$ 。令行码 C_2 是 $[29, 23, 4]$ 缩短 BCH 码，则 $A(7, 29, -1, 2)$ 码是 $[7 \times 29, 6 \times 23, 8] = [203, 138, 8]$ 码，由定理 9.4.3 知，能纠正两个长度 ≤ 7 的突发错误。如果用 $[29, 19, 5]$ 码作行码，作交错度为 7 的交错码，则 $[29 \times 7, 19 \times 7] = [203, 133]$ 交错码也能纠正两个长度 ≤ 7 的突发，但信息位只有 133，比 $A(7, 29, -1, 2)$ 码要小。

明显看出，用对角线方法传送 $A(n_1, n_2, s)$ 码的数据，当信道产生的突发长度 $\leq n_1$ ，则当收端把收到的数据排成与发端相同的 $n_1 \times n_2$ 码阵时，每行每列中的错误数目不会超过 1 个。因此，可以用很简单的行、列奇偶校验方法纠正该错误。同样，对 $A(n_1, n_2, s, t)$ 码，也有类似情况，译码也很简单。有关这类码纠错能力的详细证明可参阅文献[8-11]。

§ 9.5 组合信道纠错码

交错码、乘积码和 RS 码，都是既能纠正随机错误又能纠正突发错误的码类，所有这些码特别适用于信道干扰很复杂的、组合信道所使用的差错控制系统中。

所谓组合信道，就是信道中出现的错误类型，在某一时刻可能是突发错误也可能是随机错误，但在某一确定时刻只能出现一种类型的错误。要纠正这种类型错误的组合信道纠错码，必须满足以下条件：

- (1) 码必须能纠正所有 $\leq t$ 个随机错误所组成任何错误图样；
- (2) 码必须能纠正长度 $\leq b$ 的任何单个突发错误；
- (3) 任何重量大于 t 而长度为 b' ($t < b' \leq b$) 的单个突发错误所对应的伴随式，与任何重量小于等于 t 、长度大于 b 的突发错误图样所对应的伴随式，均不应相同。

前两个条件是显而易见的，而第三个条件是为了保证译码器能正确区分这两种错误类型所必须的。由此可知，组合信道纠错码的译码器是由两种类型的纠错译码器组成，如图 9-3。当信道出现 $\leq t$ 个随机错误时，则由纠随机错误译码器给出译码结果；若为长度 $\leq b$ 的突发错误，则由纠突发错误的译码器给出信息元。仅当长度 $\leq b$ 且重量 $\leq t$ 的错误图样产生时，两个译码器才同时工作并给出相同结果。由于这种译码器能自动地适应信道错误类型的变化而改变纠错方式，故称为**自适应译码器**，相应的码称为**自适应码**。

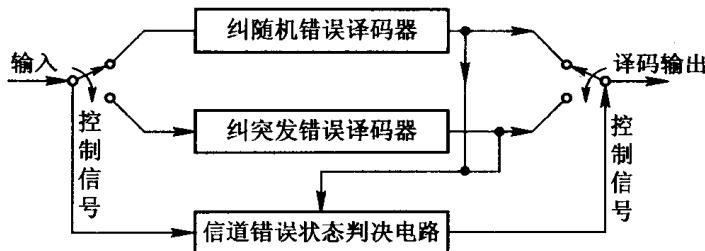


图 9-3 组合信道纠错码译码器

BCH 码不仅是一类很好的纠随机错误码，又是一类接近最佳的纠突发错误码。但在实际使用中，不能同时达到表 7-1 或表 7-2 中所给出的纠错能力。例如，[23, 12]Golay 码，由表 7-2 知，它能纠正 3 个随机错误，也能纠正长度 ≤ 5 的突发错误，但不能同时既纠正 ≤ 3 的随机错误，又纠正长度 ≤ 5 的突发错误。这是由于该码是一个完备码，它不能保证组合信道纠错码所必须满足的第三个条件，所以它不是一个组合信道纠错码。

但是[24, 12]扩展 Golay 码却是一个组合信道纠错码，它能纠正 ≤ 3 个随机错误同时又能纠正长度 ≤ 5 的突发错误，具有很强的纠错能力，并且由于它的译码器并不复杂，特别当用软判决译码时具有更强的纠错能力，因此在实际上得到极为普遍的利用。

近几年来，不少科技工作者致力于分析已有纠随机错误码的纠组合错误能力，并用计算机进行检验。表 9-3 中给出了某些循环码的纠组合信道错误的能力。表中， b_0 表示单用作纠突发错误时，码所能纠正的突发长度；而 B 表示码用作纠组合错误时，码所能纠正的突发长度； $t(t_0)$ 表示码所能纠随机错误的数目，此数目也是纠组合信道错误时所能纠正的。例如，[15, 5] 码， $t(t_0)=3$ ， $B=4$ ，说明它既能纠正 3 个随机错误同时又能纠正长度 ≤ 4 的突发错误；但仅当用来纠突发错误时，却能纠正长度 ≤ 5 的突发错误。 $g(x)$ 列下的数字与表 7-1 中的意义相同。表示 $g(x)$ 多项式的二进制系数的八进制数表示。

表 9-3 BCH 码纠组合信道错误的能力

| n | k | $t(t_0)$ | B | b_0 | $g(x)$ |
|-----|-----|----------|-----|-------|--------------|
| 15 | 5 | 3 | 4 | 5 | 2467 |
| 15 | 3 | 2 | 6 | 6 | 11111 |
| 21 | 12 | 2 | 3 | 4 | 1663 |
| 21 | 9 | 2 | 6 | 6 | 14515 |
| 21 | 7 | 3 | 6 | 7 | 47343 |
| 21 | 6 | 3 | 7 | 7 | 126357 |
| 21 | 4 | 4 | 8 | 8 | 643215 |
| 21 | 3 | 3 | 9 | 9 | 1111111 |
| 25 | 5 | 2 | 10 | 10 | 4102041 |
| 31 | 16 | 3 | 5 | 7 | 107657 |
| 31 | 16 | 3 | 5 | 7 | 161411 |
| 31 | 11 | 5 | 8 | 10 | 5423325 |
| 31 | 11 | 5 | 7 | 10 | 4636261 |
| 31 | 6 | 7 | 10 | 12 | 313365047 |
| 33 | 3 | 5 | 15 | 15 | 111111111111 |

(续表)

| n | k | $t(t_0)$ | B | b_0 | $g(x)$ |
|-----|-----|----------|-----|-------|-------------------|
| 35 | 20 | 2 | 7 | 7 | 147257 |
| 35 | 8 | 3 | 13 | 13 | 1572464475 |
| 35 | 5 | 3 | 15 | 15 | 10204102041 |
| 35 | 4 | 7 | 15 | 15 | 26130542613 |
| 39 | 15 | 4 | 11 | 12 | 153651205 |
| 39 | 13 | 5 | 12 | 13 | 423136633 |
| 39 | 3 | 6 | 18 | 18 | 11111111111111 |
| 45 | 21 | 2 | 6 | 12 | 111010001 |
| 45 | 19 | 3 | 12 | 13 | 754431721 |
| 45 | 15 | 3 | 12 | 15 | 10100110111 |
| 45 | 11 | 4 | 17 | 17 | 262310531217 |
| 45 | 9 | 2 | 18 | 18 | 1001001001001 |
| 45 | 9 | 4 | 18 | 18 | 1710017100171 |
| 45 | 7 | 7 | 17 | 19 | 7210072100721 |
| 45 | 5 | 4 | 20 | 20 | 20410204102041 |
| 45 | 3 | 7 | 21 | 21 | 111111111111111 |
| 49 | 7 | 3 | 21 | 21 | 100402010040201 |
| 63 | 42 | 2 | 10 | 10 | 16723145 |
| 63 | 40 | 2 | 9 | 11 | 61616771 |
| 63 | 36 | 2 | 13 | 13 | 1637365531 |
| 63 | 30 | 3 | 16 | 16 | 104406705613 |
| 63 | 30 | 3 | 16 | 16 | 176052550565 |
| 63 | 28 | 2 | 17 | 17 | 645645000645 |
| 63 | 28 | 3 | 17 | 17 | 572325033113 |
| 63 | 27 | 2 | 18 | 18 | 1100101001101 |
| 63 | 27 | 3 | 18 | 18 | 1620621001621 |
| 63 | 24 | 3 | 19 | 19 | 17126612222055 |
| 63 | 24 | 3 | 19 | 19 | 13673405647577 |
| 63 | 24 | 3 | 19 | 19 | 15470164000521 |
| 63 | 24 | 4 | 19 | 19 | 17505507632405 |
| 63 | 22 | 3 | 20 | 20 | 62777715062715 |
| 63 | 21 | 3 | 18 | 21 | 100111011100011 |
| 63 | 15 | 3 | 24 | 24 | 15151450144514445 |
| 63 | 15 | 4 | 24 | 24 | 17725062100340401 |

* § 9.6 级联码与贾斯特逊(Justesen)码

信道编码定理指出，随着码长 n 的增加，译码错误概率按指数接近于零。因此，为了

使码有效就必须用长码。但是，随着码长的增加，在一个码组中要求纠错的数目相应增加，译码器的复杂性和计算量也相应增加以致难以实现。为了解决性能与设备复杂性的矛盾，1966年福尼提出了级联码概念，把编制长码的过程分几级完成，通常分两级。

从本质上讲，级联码是乘积码的特殊情况，通常也由两个子码组成，不同的是这两个子码取自不同的域并串接而成。级联码不仅有极强的纠突发和纠随机错误的能力，更重要的是利用级联码的构造方法，能达到信道编码定理所给出的码限，也就是能构造出渐近好码(Shannon 码)。1972 年贾斯特逊首先用级联码的构造方法，具体地给出了构造一类渐近好码——Justesen 码(J 码)的方法，当 $n \rightarrow \infty$ 时，其性能能达到信道编码定理所指出的限。正由于级联码好的渐近性能，并且近来又发现用它可以构造一类具有极好性能的密钥分散保管系统^[16]，因此级联码日益引起了人们的极大兴趣。

一、级联码的基本概念

图 9-4 给出了一个应用两级级联码的差错控制系统。由此图看到，级联码由内码 C_i 和外码 C_o 组成。内码 C_i 是 $GF(2)$ 上的一个 $[n, k]$ 码，外码 C_o 是 $GF(2^k)$ 上的 $[N, K]$ 码，并按以下方法编码：

- (1) 把 Kk 个二进制信息元，划分成 K 段，每段有 k 个码元。
- (2) 把每一组有 k 个信息元的段，看成是 $GF(2^k)$ 上的一个符号(元素)。将 K 个符号按外码 C_o 的编码规则编成一个码字，码长为 N ，有 K 个信息符号， $N-K$ 个校验符号，最小距离为 d_o ，码率 $R_o = K/N$ 。

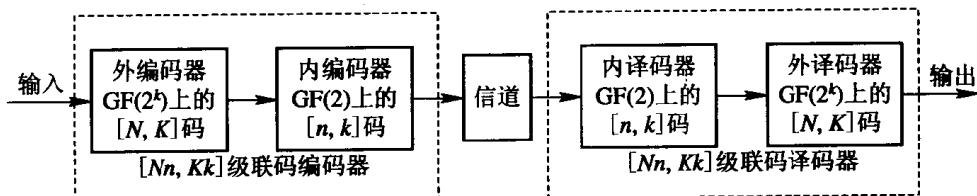


图 9-4 利用级联码的差错控制系统

(3) 再把 C_o 外码的每一个符号，看成是 k 个二进制码元的信息组，输入内码 C_i 编码器，得到内码的一个码字。它的码长为 n ，有 $n-k$ 个校验元，最小距离为 d_i ，码率 $R_i = k/n$ ，由此得到 $[n, k]$ 内码的 N 个码字序列。故总共有 Nn 个二进制码元、 Kk 个信息元，组成了 $[Nn, Kk, d_o, d_i]$ 级联码的码字。

可知两级级联码是一个 $[Nn, Kk, d_o, d_i]$ 二进制线性分组码，码率 $R_c = R_o R_i$ 。级联码的译码采用分级处理：先按内码 C_i 的规则译码，得到 N 个有 k 个码元组成的符号后，再送入外译码器，按外码 C_o 的规则译码，最后送出 K 个信息符号，每个符号有 k 个二进制信息元。所以，最后由外译码器输出的是已经过纠错的 Kk 个信息元。

当然，可以把两级级联的方法推广到多级，但目前仅局限于两级。内码 C_i 一般是一个二进制线性分组码，如 BCH 码。外码 C_o 通常是 $GF(2^k)$ 上的 $[N, K]$ RS 码，它的 $d_o = N-K+1$ ，能纠正 $t = \lfloor (N-K)/2 \rfloor$ 个随机错误，或纠正删除和随机错误的组合。总之，根据不同要求和给定的 N ，取不同的 K 就可得到不同纠错能力的 RS 码。由 RS 码和级联码

构造的关系可知，内外码的参数之间的关系是 $N=2^k-1$ 。若要求 $N < 2^k-1$ ，则可采用缩短码的形式。

当信道产生少量的随机错误时，通过内码 C_i 就可纠正。当产生较长的突发错误或随机错误很多，以至超过 C_i 码的纠错能力，则内译码器产生错译，输出的码字有几个错误。但这仅相当于外码 C_o 的几个符号错误，外码译码器就能较容易地纠正。从这里可以看出，级联码用来纠正组合信道错误是非常有效的。

在级联码的实现中，内码既可以用作纯纠错，也可以用作纠错与检错。一般情况下，级联码用在干扰比较严重的组合信道中，内码中的某些码字内错误很多，往往超过内码的纠错能力。所以，通常内码仅用来纠正少量错误，而大部分能力用来检错，指出错误位置（对外码来说就是删除符号的位置），纠错任务则由外码译码器完成。这样两级译码的结果，使得内、外译码器均较简单：内译码器主要是检错译码器，而外译码器主要是一个纠删译码器。这种译码方法使得译码器的复杂性和计算量，大大低于有相同参数的其它单级分组码。当然，这种只有一个内、外译码器的串行译码方法，达不到级联码最小距离 $d_i d_o$ 所决定的纠错能力，而必须采用多个内、外译码器平行工作，并用软判决的平行译码方法^[12]，才能达到或接近 $d_i d_o$ 所决定的纠错能力。

例 9.3 内码用二进制 $[7, 3, 4]$ 增余删信汉明码，外码用 $GF(2^3)$ 域上的 $[7, 3, 5]$ RS 码，组成一个 $[49, 9, 20]$ 级联码。内码的距离是 4，由两个内译码器并行组成，一个是按纠正 1 个错误同时发现 2 个错误，另一个是按纠正 3 个删除设计的内译码器，外码距离是 5，按纠正 4 个删除错误设计外译码器。

设 9 个信息元是 (111011010) ，按 3 个码元为一组分段，得到 $GF(2^3)$ 上的 3 个元素是： $\alpha^5(111)$ ， $\alpha^3(011)$ 和 $\alpha(010)$ （参看表 4.1）。 $[7, 3]$ RS 码的生成多项式

$$g(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^3)(x - \alpha^4) = x^4 + \alpha^3 x^3 + x^2 + \alpha x + \alpha^3$$

得到 $[7, 3]$ RS 码的一个码字是 $(\alpha^5 \alpha^3 \alpha \alpha^6 \alpha^4 \alpha^2 1)$ ，把这 7 个 $GF(2^3)$ 上的符号，映射成二进制三重，并把每一个三重作为一信息组，送入由 $g(x) = x^4 + x^3 + x^2 + 1$ 确定的内码编码器，得到 7 个内码码字，它们就是 $[49, 9, 20]$ 级联码的一个码组，如下表(a)所示，其中

| 外 码 | | | | | | | | | |
|------------------|------------|----------|------------|------------|------------|---|---|---|---|
| 信息元 | | | 校验元 | | | | | | |
| α^5 | α^3 | α | α^6 | α^4 | α^2 | 1 | 1 | 0 | 0 |
| 内 容 容 容 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | x | 1 |
| | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 |
| | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| | 0 | 1 | 0 | 0 | 1 | 1 | 0 | x | x |
| | 0 | 0 | 1 | 0 | 0 | 1 | 1 | x | 1 |
| | 0 | 1 | 1 | 1 | 0 | 1 | 0 | x | 0 |
| (a) 发的码字 | | | (b) 接收码字 | | | | | | |
| 1 | 0 | x | x | x | x | 0 | 1 | 0 | 0 |
| 1 | 1 | x | x | x | x | 0 | 1 | 1 | 1 |
| 1 | 1 | x | x | x | x | 1 | 1 | 1 | 0 |
| (c) 内译码器输出 | | | | | (d) 外译码器输出 | | | | |

的(b)、(c)和(d)还分别给出了接收码字及内外译码器的译码结果。图中的 x 表示这一位删除，码元下面的一条横线表示这位有错。由此看出级联码有很强的纠错能力。■

从该例看到，级联码的两个码在不同域中，而乘积码的两个码在同一域中都是二进制码或 q 进制码。

除了分组码与分组码级联外，分组码也可以与卷积码级联。这时外码用RS码，内码用卷积码并采用维特比(Viterbi)译码。理论分析和计算机模拟表明，在高斯白噪声信道中，外码用[255, 233]RS码，内码用(2, 1, 6)卷积码并用维特比译码，则大约有7 dB以上的编码增益，故特别适合于卫星通信和宇航通信中。

从以上讨论可以看出，从本质上讲级联码其实是一种直接序列扩频系统，它把每一个外码的码元，用码长为 n_i 的内码码字填充，从而使得系统的带宽扩展，提高了系统的抗干扰能力。

为了提高级联码在突发信道，特别是短波、散射等信道中的性能，可把交错方法与级联码结合应用，内码以按组的方式进行交错，以抗击较长的突发错误。

级联码设备的复杂性主要决定于译码器，可以证明其复杂性并不随码长 $n_c = Nn$ 的增长指数增加，而是随 n_c 以小的幂次增加。在最坏情况下，其复杂性和 n_c^4 成比例；如果不用广义最小距离译码，则与 n_c^2 成比例。因此，这比一级译码时的复杂性随码长指数增加要好得多。

1971年乔勃洛夫(Зяблов)首先讨论了级联码的渐近性，指出一定存在有一类级联码，当 R 保持一定， $n \rightarrow \infty$ 时能做到 $d/n > 0$ ，并给出了以下渐近性能限：

$$\frac{d}{n} \geq \max_{0 \leq r \leq 1} \left\{ \left(1 - \frac{R}{r} \right) H_2^{-1}(1 - r) \right\} \quad (9.6.1)$$

称该限为乔勃洛夫(ZR)限，它是在外码用最大距离可分码时得到的。式中

$$H_2(x) = -x \log_2 x - (1 - x) \log_2(1 - x) \quad (9.6.2)$$

是熵函数， R 是级联码的码率， r 是内码的码率。该限的 d/n 与 R 之间的关系，示于图9-5中，为了比较也给出了V-G限曲线，显然，ZR限比V-G限要差，这说明虽然可用级联码构造渐近好码，但在同样的码率 R 下，它所具有的纠错能力比V-G限所指出的要小得多。事实上，福尼早已指出了这一点，他证明应用级联码的通信系统，其误码率 p 能达到

$$p \leq e^{-NE_c(R)}$$

式中， N 是级联码的码长； R 是级联码的码率； $E_c(R)$ 称为级联指数，当 $R < C$ 时，恒有 $E_c(R) > 0$ 。由此式看出：随着 $n \rightarrow \infty$ ， $p \rightarrow 0$ 。但是由于 $E_c(R)$ 通常小于式(1.4.1)中的误差指数 $E_b(R)$ ，这说明应用级联码后，在同样码率下要达到同样的性能，级联码的码长比不用级联码的Shannon码要长。通常以

$$\eta_c(R) = \frac{E_c(R)}{E_b(R)} \quad (9.6.3)$$

来衡量级联码的效率。如果 $\eta_c(R)$ 为0.1，则说明级联码必须长10倍，才能达到未级联时码的性能，这就是级联码所付出的代价。

正由于级联码的渐近特性很好，也就是说利用级联码的构造方法，能构造出Shannon码，因而自70年代以来，不少学者纷纷致力于用级联方法构造渐近好码。1972年贾斯特

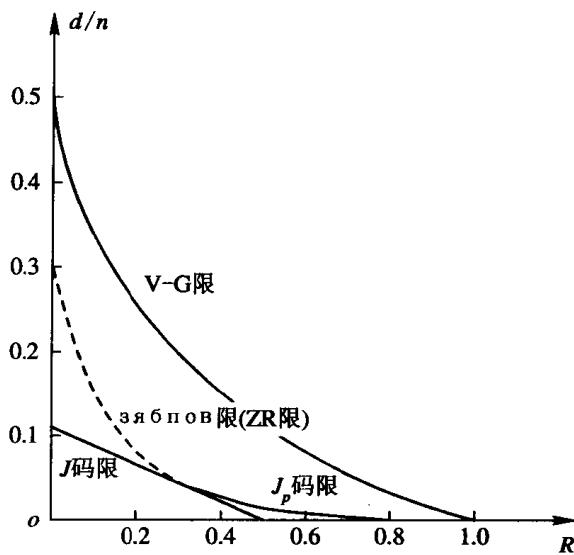


图 9-5 ZR 限、V-G 限与 J 码限之比较

逊终于用级联方法具体地构造了一类好码——J 码，从而首次具体地给出了构造Shannon码的方法。

二、Justesen 码(J 码)

J 码是外码用 $GF(2^m)$ 上的 RS 码，内码用某一特殊的二进制码构造成的级联码。

定义 9.6.1 设 R_s 是 $GF(2^m)$ 上的 $[N=2^m-1, K, D=N-K+1]$ RS 码，它的码字 $\mathbf{a}=(a_{N-1}, a_{N-2}, \dots, a_0)$, $a_i \in GF(2^m)$ 。又设 α 是 $GF(2^m)$ 中的一个本原域元素，作以下矢量：

$$\mathbf{C} = (a_{N-1}\alpha^{N-1}a_{N-1}, a_{N-2}\alpha^{N-2}a_{N-2}, \dots, a_i\alpha^ia_i, \dots, a_0\alpha^0a_0) \quad (9.6.4)$$

若矢量 \mathbf{C} 中的每个 a_i 与 $\alpha^i a_i$ 均用二进制 m 重表示，则 \mathbf{C} 矢量就成为 $GF(2)$ 上的长为 $2mN$ 的矢量，所有这些矢量集合就称为 $GF(2)$ 上的一个 J 码。

由以上定义可知，J 码有如下参数：

$$n = 2mN = 2m(2^m - 1) \quad k = mK$$

$$R = \frac{k}{n} = \frac{mK}{2m(2^m - 1)} = \frac{K}{2(2^m - 1)} < \frac{1}{2}$$

因此，由定义 9.6.1 所确定的 J 码的 R 小于 $1/2$ ，当然也可构造码率大于 $1/2$ 的 J 码，这将在后面介绍。

例 9.4 构造一个 $n=2 \times 4 \times (2^4-1)=120$, $k=44$ 的 J 码。

先构造外码，由定义可知，外码是 $GF(2^4)$ 上的 $[15, 11, 5]$ RS 码，设该码的生成多项式

$$\begin{aligned} g(x) &= (x + \alpha)(x + \alpha^2)(x + \alpha^3)(x + \alpha^4) \\ &= x^4 + \alpha^{13}x^3 + \alpha^6x^2 + \alpha^3x + \alpha^{10} \end{aligned}$$

式中， α 是 $GF(2^4)$ 中的本原元，它是 x^4+x+1 的根。

设信息位是： $\alpha \alpha^2 \alpha \alpha^3 \alpha^8 \alpha^3 \alpha^4 \alpha^9 \alpha^8 \alpha^7 \alpha^4$ ，由此得到 RS 码系统码的一个码字 \mathbf{a} 为

$$\begin{aligned}\mathbf{a} &= (\alpha \alpha^2 \alpha \alpha^0 \alpha^8 \alpha^3 \alpha^4 \alpha^9 \alpha^8 \alpha^7 \alpha^4 \alpha^{12} \alpha^4 \alpha^8 \alpha^7) \\ &= (a_{14}, a_{13}, \dots, a_1, a_0)\end{aligned}$$

式中，最后 4 个元素是检验元，由此 \mathbf{a} 可得矢量

$$\mathbf{C} = (a_{14}, \alpha^{14}a_{14}, a_{13}, \alpha^{13}a_{13}, \dots, a_1, \alpha^1a_1, a_0, \alpha^0a_0)$$

为此先求：

$$\begin{aligned}(a_0, \alpha^0a_0) &= (\alpha^7, \alpha^7) = (1101, 1101) \\ (a_1, \alpha^1a_1) &= (\alpha^8, \alpha\alpha^8) = (\alpha^8, \alpha^9) = (1010, 0101) \\ (a_2, \alpha^2a_2) &= (\alpha^4, \alpha^2\alpha^4) = (\alpha^4, \alpha^6) = (1100, 0011) \\ &\vdots \\ (a_{14}, \alpha^{14}a_{14}) &= (\alpha, \alpha^{14}\alpha) = (\alpha, 1) = (0100, 1000)\end{aligned}$$

所以 $(a_i, \alpha^i a_i)$ 就是一个 $[8, 4]$ 内码，而 $\alpha^i a_i$ 就是 a_i 的右移 i 次。由此我们可得到 J 码的一个码矢为：(0100, 1000, …, 1010, 0101, 1101, 1101)。 ■

由该例看出，J 码的外码就是一般的本原 RS 码。但内码 $(a_i, \alpha^i a_i)$ 比较特殊，它是一个循环移位重复码。有以下特点：对任何 $i \neq j$ ，即使 $a_i = a_j$ ，但 $(a_i, \alpha^i a_i) \neq (a_j, \alpha^j a_j)$ 。今后我们用 $(\bar{a}_i, \bar{b}_i) = (a_i, \alpha^i a_i)$ 表示内码。

因此我们可把 J 码的码组 $\bar{\mathbf{C}} = (\bar{\mathbf{a}}, \bar{\mathbf{b}})$ 看成是由两部分组成： $\bar{\mathbf{a}} = (a_0, a_1, \dots, a_{N-1})$ ， $\bar{\mathbf{b}} = (a_0, \alpha a_1, \dots, \alpha^{N-1} a_{N-1})$ 。若写成多项式表示，则：

$$\begin{aligned}\bar{a}(x) &= a_0 + a_1x + a_2x^2 + \dots + a_{N-1}x^{N-1} \\ \bar{b}(x) &= a_0 + \alpha a_1x + \alpha^2 a_2x^2 + \dots + \alpha^{N-1} a_{N-1}x^{N-1} = \bar{a}(\alpha x)\end{aligned}$$

所有 $\bar{\mathbf{a}}$ 组成的集合我们称为 a 码，所有 $\bar{\mathbf{b}}$ 组成的集合称为 b 码，则 a 码的生成多项式

$$g_a(x) = (x + \alpha)(x + \alpha^2) \cdots (x + \alpha^{N-K})$$

而 b 码的生成多项式

$$g_b(x) = g_a(\alpha x) = (x + 1)(x + \alpha) \cdots (x + \alpha^{N-K-1})$$

因而

$$\frac{g_a(x)}{g_b(x)} = \frac{x + \alpha^{N-K}}{x + 1}$$

下面定理给出了 J 码的渐近性能。

定理 9.6.1 对任一给定的码率 R ， $0 < R < 1/2$ ，存在有 $[2mN, mK]$ 二进制 J 码，有 $R = (1/2)K/N$ ，且满足：

$$\frac{d_n}{n} \geq (1 - 2R)(H_2^{-1}(0.5) - o(m)) \approx 0.11(1 - 2R) \quad (9.6.5)$$

式中， $H_2^{-1}(0.5)$ 由式 (9.6.2) 决定， d_n 是 J 码的最小距离， $o(m)$ 随 m 很快趋近于 0。

由该定理可知 J 码的 d_n/n 是 R 的线性函数，它们之间的关系示于图 9-5 中。由该图看到，当 $R = 0.3$ 时，J 码与 ZR 限一致，显然，J 码是一类渐近好码。

上面讨论 J 码的码率都是小于 $1/2$ ，我们也可以构造出码率大于 $1/2$ 的 J 码，称这类码为删除 Justesen 码，用 J_p 表示。

把 J 码的每个码字中分量 (\bar{a}_i, \bar{b}_i) 表示成二进制 m 重以前，若把 b 码的每个分量 $\bar{b}_i = \alpha^i a_i$ 的二进制 m 重表示，删去其最后 s 个 ($0 \leq s \leq m$) 码元成为 $m-s$ 重，则由此得到的二进制码

字集合，就称为 J_p 码，因而 J_p 码有以下参数：

$$n = (2m - s)N \quad k = mK$$

$$R = \frac{mK}{(2m - s)N}$$

式中， s 由内码率 r 所确定，当 $1/2 \leq r < 1$ 选定时，则

$$s = \frac{m(2r - 1)}{r}$$

在此条件下， J_p 码渐近性能下限是

$$\frac{d}{n} \geq \left(1 - \frac{R}{r}\right) H_2^{-1}(1 - r) \quad m \rightarrow \infty \quad (9.6.6)$$

该式对 r 求偏导数并使其为 0，则得到使 d/n 最大时的 r ，此时 r 应满足：

$$R = \frac{r^2}{1 + \log_2[1 - H_2^{-1}(1 - r)]} \quad (9.6.7)$$

由于允许构造内码的码率至少为 $1/2$ 。所以若式(9.6.7)的解使 $r < 1/2$ ，则应取 $r = 1/2$ ，这就由 J_p 码成为 J 码。

J_p 码的渐近性能曲线也示于图 9-5 中，由此图看出 Justesen 码 (J 码和 J_p 码) 在 $R \leq 0.3$ 时与 ZR 限一致，但当 $R > 0.3$ 时，则渐近性能要差。

§ 9.7 纠突发错误码的译码

纠突发错误循环码的译码方法有四类：一般译码、快速译码、最佳译码和纠删译码。其中最佳译码^[13]是为了纠正长度 $\leq n-k$ 突发中的大多数突发图样，这是为了充分利用码的纠错能力而设计的一种译码方法。而快速译码主要用于容错计算机中的纠错。本节仅介绍在实际中最常用到的一般译码器，并简单介绍快速译码器，至于最佳译码方法可参阅 [13, 14]。

一、一般译码器

$[n, k]$ 线性分组码的纠突发能力 $b \leq (n-k)/2$ ，因此可以用捕错译码方法纠正码字中的突发错误。与纠随机错误循环码的捕错译码器一样，纠突发错误循环码的捕错译码器也分两类。不同的是捕错译码器中的检测电路，在这里不是检测伴随式 $S(x)$ 的重量，而是检测可纠正的突发错误图样。这种检测电路比较简单，是一个有 $(n-k-b)$ 个输入端与门，下面讨论这一点。

设接收到的 n 重 $R(x) = C(x) + E(x)$ ， $C(x)$ 为发送的码字， $E(x)$ 为错误图样，可把

$$E(x) = E_I(x) + E_p(x)$$

式中， $E_I(x)$ 和 $E_p(x)$ 分别是产生在信息段与校验段或 x^{n-1} 至 x^{n-k} 位与 x^{n-k-1} 至 x^0 位内的错误图样。

由于可纠正的突发错误图样，仅局限于 $b < n-k$ 个连续码元以内，设突发产生在校验段内，则

$$E(x) = E_p(x)$$

伴随式

$$S(x) \equiv R(x) \equiv E_p(x) \pmod{g(x)}$$

式中, $g(x)$ 是 $[n, k]$ 码的生成多项式。伴随式多项式

$$S(x) = s_{n-k-1}x^{n-k-1} + \cdots + s_1x + s_0$$

因 $E_p(x)$ 的次数小于 $n-k$, 且仅局限在连续 b 个码元内, 设集中在 x^{n-k-1} 至 x^0 码元段的高次位内, 即

$$E_p(x) = e_{n-k-1}x^{n-k-1} + \cdots + e_{n-k-b}x^{n-k-b}$$

则

$$S(x) = s_{n-k-1}x^{n-k-1} + \cdots + s_1x + s_0 = e_{n-k-1}x^{n-k-1} + \cdots + e_{n-k-b}x^{n-k-b}$$

因而, $s_{n-k-(b+1)}, s_{n-k-(b+2)}, \dots, s_1, s_0$ 分量应全为 0。所以, 根据所得伴随式最低次的 $n-k-b$ 个分量是否全为 0, 就可判断出突发错误是否发生在 $n-k$ 校验段以内的最前面连续 b 个码元内。如果突发错误不在 $R(x)$ 的 $x^{n-k-1}, \dots, x^{n-k-b}$ 的连续码元段内, 而在 $x^{n-k-1-i}, \dots, x^{n-k-b-i}$ 的连续码元段内, 则根据伴随式计算电路($g(x)$ 运算电路)自发运算时的性质可知, 只要 $S(x)$ 在伴随式计算电路中自发运算 i 次, 同时 n 级移存器循环移位 i 次, 就可使得错误图样自动地移到校验位的最高次的 b 个码元内, 此时伴随式计算电路中的伴随式

$$S_i(x) \equiv x^i S(x) \pmod{g(x)}$$

即为此时的错误图样。如果突发不在校验码元段内, 而在 $R(x)$ 的任何地方开始, 则由于循环码的循环特性, 最后总可把突发错误图样循环地移到校验码元段的 b 个最高次码元段内, 从而进行纠正。

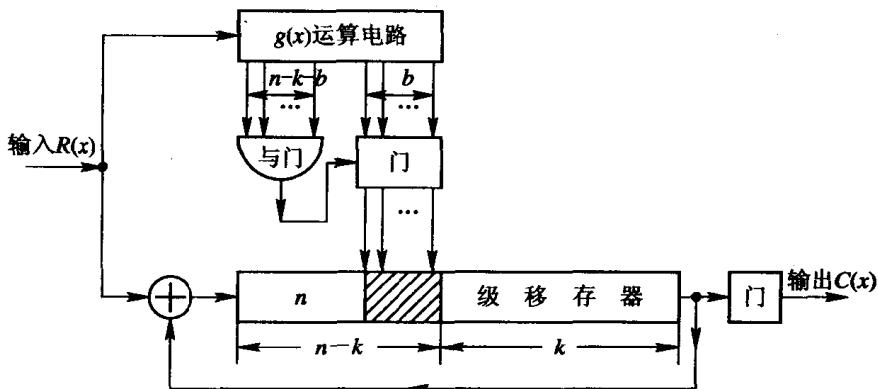


图 9-6 第 I 类纠突发错误译码器

由上可知, 纠突发错误循环码的 I、II 类捕错译码器如图 9-6、图 9-7 所示。图中的阴影线表示突发的错误位置。图 9-6 的第 I 类译码器能纠正首尾相接而总长不大于 b 的突发错误图样。而图 9-7 的第 II 类译码器, 由于其 n 级缓存器的输出不再回到输入端, 故不能纠正这种首尾相接的错误图样。但是, 第 II 类译码器的特点是, 只要收到的 $R(x)$ 从 n 级缓存器输出至用户时, 就对它进行了纠错。因此, 数据串行输出时, 用第 II 类较好, 因为第 I 类译码器若要串行输出, 还得再移动 n 次。这两种译码器, 译一个码组总共需运算移位 $2n$ 次。

有时为了节省器件, 可把 n 级缓存器改为 k 级, 但这时必须加以适当的控制电路, 以控制 k 级缓存器何时移位。此外, 这两类电路只适合于循环码, 若为缩短循环码, 则必须对

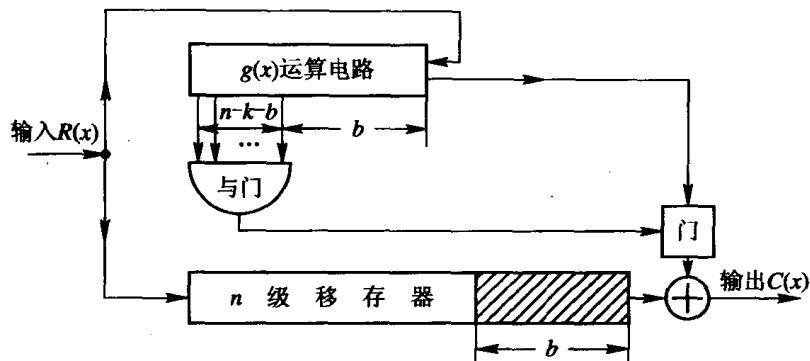


图 9-7 第Ⅰ类纠突发错误译码器

它们进行修正，加上修正项。因此，在应用表 9-2 中所列出的码时，首先必须确定它们是否是循环码；若不是，则必须找出原来的循环码，再找出修正项。

例 9.5 [279, 265] Fire 码，它的生成多项式 $g(x) = (x^9 - 1)(x^5 + x^2 + 1) = x^{14} + x^{11} + x^9 + x^5 + x^2 + 1$ ，能纠正长度 ≤ 5 的突发错误。该码的第Ⅰ类捕错译码器画于图 9-8 中。图中，门 1 与门 2 作为控制用，当接收数据时门 1 开、门 2 关。若错误全部集中在校验位（共 14 位）的前 5 位中，则后 9 位错误图样全为 0。因此，如果伴随式寄存器中后 9 位（即 $S(x)$ 的 x^0 至 x^8 ）全为 0，说明突发错误已捕获到校验位的前 5 位中。这时控制门 1 关门 2 开进行纠错。■

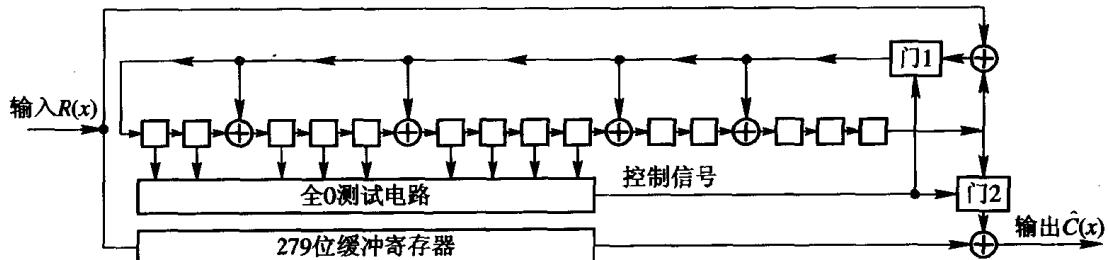


图 9-8 [279, 265] Fire 码译码器

例 9.6 把例 9.5 中的 Fire 码缩短 65 位，成为 [214, 200] 缩短 Fire 码。

每个码字均缩短了 65 位，相当于 $R(x)$ 应乘以 x^{65} ，但因为第Ⅰ类译码电路还需要乘以 $x^{n-k} = x^{14}$ ，所以，总共必须自动乘以 $x^{65+14} = x^{79}$ 。故修正项是

$$x^{79} \equiv x^{13} + x^{11} + x^{10} + x^9 + x^7 + x^4 + x^2 + x + 1 \pmod{g(x)}$$

与 $g(x) = x^{14} + x^{11} + x^9 + x^5 + x^2 + 1$ 比较可知， $x^{11}, x^9, x^2, 1$ 为公共项，由此可得图 9-9 所示的第Ⅱ类捕错译码电路。■

二、快速译码器

从上面两个例子看出，至少需要 $2n$ 次移位，才能完成 Fire 码的一个码字译码。其中前 n 次是接收 $R(x)$ 并计算 $S(x)$ ，后 n 次完成纠错，并且为了使译码连续必须有两个 $g(x)$ 电路交替工作。最近的研究表明^[14]，利用 Fire 码的结构特点，至多只要 $n-k-1$ 次移位就能

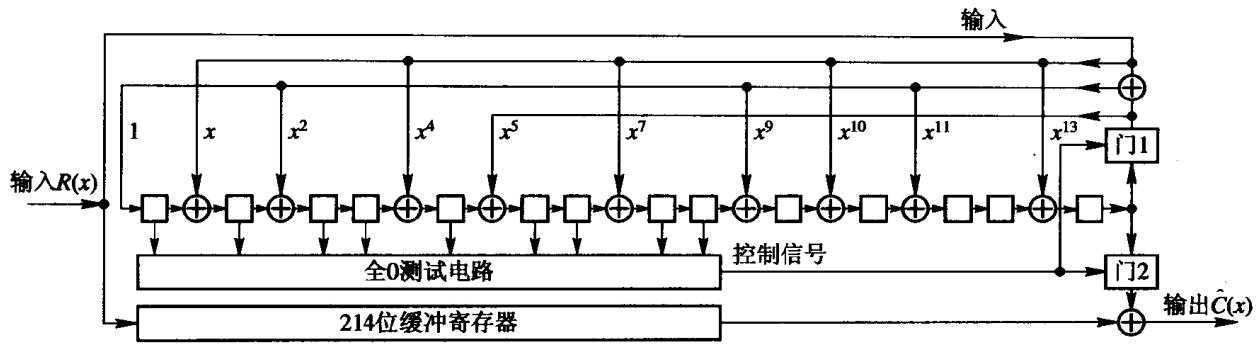


图 9-9 [214, 200] 缩短 Fire 码译码器

完成一个码字的译码。这种快速译码方法，在计算机存贮系统中有着较广的应用前景。下面讨论这种译码方法。

纠突发错误的快速译码，也就是快速地决定突发位置，是基于孙子定理基础上的。为了理解快速译码原理，我们先讨论纠突发错误的第Ⅲ类译码器。

若 $[n, k]$ 纠突发错误循环码的 $g(x) = p_p(x)p_c(x)$, $\partial \circ p_p(x) = m_p$, $\partial \circ p_c(x) = m_c$, $m_p + m_c = n - k$, 且 $m_c > m_p \geq b$, $n = \text{LCM}(c, p)$, c 和 p 分别是 $p_c(x)$ 和 $p_p(x)$ 的周期。对具有这种 $g(x)$ 的纠突发错误循环码来说，除了 I、Ⅱ类译码器外，还有如图 9-10 所示的第Ⅲ类译码器。这类译码器的纠错原理仍是基于捕错译码基础上的。任何长度 $\leq b$ 的突发 $x^i B(x)$ ($0 \leq i \leq n-b$, $\partial \circ B(x) \leq b-1$)，均不能被 $p_p(x)$ 和 $p_c(x)$ 除尽，被它们除后所得的余式，必是一个可纠正的突发错误图样，下面说明这点。

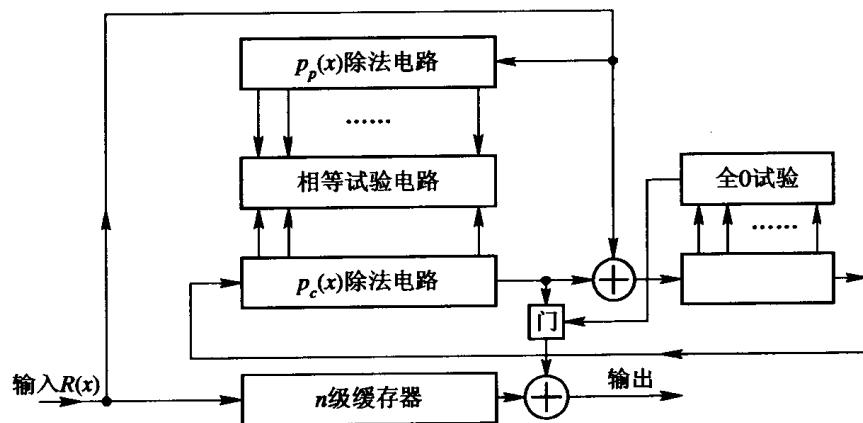


图 9-10 第Ⅲ类纠突发错误译码器

由该图看到， $R(x)$ 是从 $p_p(x)$ 的高次位 (x^{m_p} 次位) 和 $p_c(x)$ 的 x^{m_p} 次位的地方输入，这等效于 $R(x)$ 自动地乘以 x^{2m_p} 后送入 $g(x)$ 电路。并且错误图样是从 $p_c(x)$ 的高次位地方输出，这相当于错误图样自动地乘 $x^{2m_p + (m_c - m_p)} = x^{n-k}$ ，这与第Ⅱ类电路完全等效。所以当 $R(x)$ 全部进入 n 级移存器后，在 $p_p(x)$ 和 $p_c(x)$ 电路中的数据为：

$$S_p(x) \equiv R(x)x^{m_p} \equiv x^{i+m_p}B(x) \pmod{p_p(x)}$$

$$S_c(x) \equiv R(x)x^{m_p} \equiv x^{i+m_p}B(x) \pmod{p_c(x)}$$

式中, $B(x)$ 是 $R(x)$ 中的长度 $\leq b$ 的突发错误图样。当再进行 $n-i-m_p$ 次自发运算后, 这两个电路中的内容分别为:

$$x^{n-i-m_p}S_p(x) \equiv x^{n-(i+m_p)}x^{i+m_p}B(x) \equiv B(x) \pmod{p_p(x)}$$

$$x^{n-i-m_p}S_c(x) \equiv x^{n-(i+m_p)}x^{i+m_p}B(x) \equiv B(x) \pmod{p_c(x)}$$

所以, 当 $n-i-m_p$ 次移位后, 这两个电路中的内容完全相同, 即为 $B(x)$ 。并且这时 n 级移存器中的突发错误图样刚好出现在输出端, 就可进行纠正。对缩短循环码来说, 也仅只要加上适当的修正项, 就可以构造第Ⅲ类译码器。

从这类译码器可以看出, 当第一组 n 次移位计算后, 在 $p_p(x)$ 与 $p_c(x)$ 除法电路中已存贮了突发图样。如果现在不是从 $p_p(x)$ 和 $p_c(x)$ 的 m_p 次位的地方输入 $R(x)$, 而是从 $p_p(x)$ 和 $p_c(x)$ 的输入端(最低次位)送入, 则第一组 n 次移位后, 在两个除法电路中存贮的内容即为:

$$x^iB(x) \equiv S_p(x) \pmod{p_p(x)}$$

$$x^iB(x) \equiv S_c(x) \pmod{p_c(x)} \quad (9.7.1)$$

现在的关键问题是如何很快地决定 i 。由于码长 $n=LCM(c, p)$, 其中 c 和 p 分别是 $p_c(x)$ 和 $p_p(x)$ 的周期, 由此可知, 错误位置 i 必是 c 和 p 的剩余:

$$i = r_p \pmod{p}$$

$$i = r_c \pmod{c}$$

若 $(c, p)=1$, 则根据孙子定理可得

$$i \equiv A_c pr_c + A_p cr_p \pmod{n} \quad (9.7.2)$$

式中, A_c 和 A_p 由欧几里德算法决定

$$A_c p + A_p c = (c, p) = 1$$

由于一个码确定后, A_c 和 A_p 均可事先计算得到, 预先存贮在译码器内, 因此, 由式(9.7.2)看到, 要确定 i 关键是求出 r_c 和 r_p 。

由于 $\partial \cdot B(x) \leq b-1 < \partial \cdot p_p(x) < \partial \cdot p_c(x)$, 因此当第一组 n 次移位后, 即在 $p_c(x)$ 电路中得到了 $x^iB(x)$, 如果再移动 $c-i$ 次, 则式(9.7.1)成为

$$x^{c-i}x^iB(x) \equiv x^cB(x) \equiv B(x) \equiv x^{c-i}s_c(x) \pmod{p_c(x)}$$

所以, 错误图样 $B(x)$ 出现在 $p_c(x)$ 电路中最低次项的 b 位上, 而其余的 $c-b$ 位全为 0。因此, 如果在第一组 n 次移位完毕后, 使 $p_c(x)$ 电路进入自发运算状态, 如果计算 N_c 次后, 在该电路的最高次位出现了 $c-b$ 个 0 状态, 则说明此时在最低次位的内容即为 $B(x)$, 故

$$c - i \equiv N_c$$

$$i \equiv c - N_c \equiv r_c \pmod{c}$$

所以, 由第二组 n 次移位中的第一个 c 次移位内即可得到 r_c 。当 $p_c(x)$ 电路中出现这种情况后, 保持 $r_c(x)$ 中的内容不变, 即 $B(x)$ 仍保持在该电路的最低次 b 级之中, 然后, 我们移动 $p_p(x)$ 中的内容, 使得

$$x^{p-i}x^iB(x) \equiv x^{p-i}S_p(x) \equiv B(x) \pmod{p_p(x)}$$

如果 N_p 次移位后, 在 $p_p(x)$ 最低次的 b 级内容与 $p_c(x)$ 中最低次 b 级内容相同, 则说明 $p_p(x)$ 电路中也得到了突发图样 $B(x)$ 。而从次数上讲这个过程也就是模 p 的过程, 故

$$i \equiv p - N_p \equiv r_p \pmod{p}$$

由上述运算过程中, 可以得出第二组 n 次移位中, 只要移动

$$N = N_c + N_p < c + p \ll cp = n$$

次, 就可确定出突发错误位置和突发错误图样, 特别当 n 很大时, 译码速度可大大加快。如 [693, 676] Fire 码, $g(x) = p_p(x)p_c(x) = (x^6+x+1)(x^{11}+1)$ 。若用一般译码器需计算 693 次才能译完一个码字(不算第一组 n 次移位, 下同); 而用快速译码, 则至多只需计算 $11+63=74$ 次即可译完, 速度几乎快了一个数量级。

可把这种快速译码方法, 推广到 $g(x)$ 由多于两个因子乘积的码中, 如广义 Fire 码。

例 9.7 作由 $g(x) = p_c(x)p_1(x)p_2(x) = (x^{11}+1)(x^3+x+1)(x^4+x+1)$ 生成的二进制 [1155, 1137] 广义 Fire 码的快速译码器。

该码能纠正 $b=3$ 的突发错误和绝大多数长度 $\leqslant 6$ 的突发错误。若用一般译码器需计算 1155 次; 但用快速译码器, 则只需要计算 $11+15=26$ 次。它的快速译码电路如图 9-11 所示。

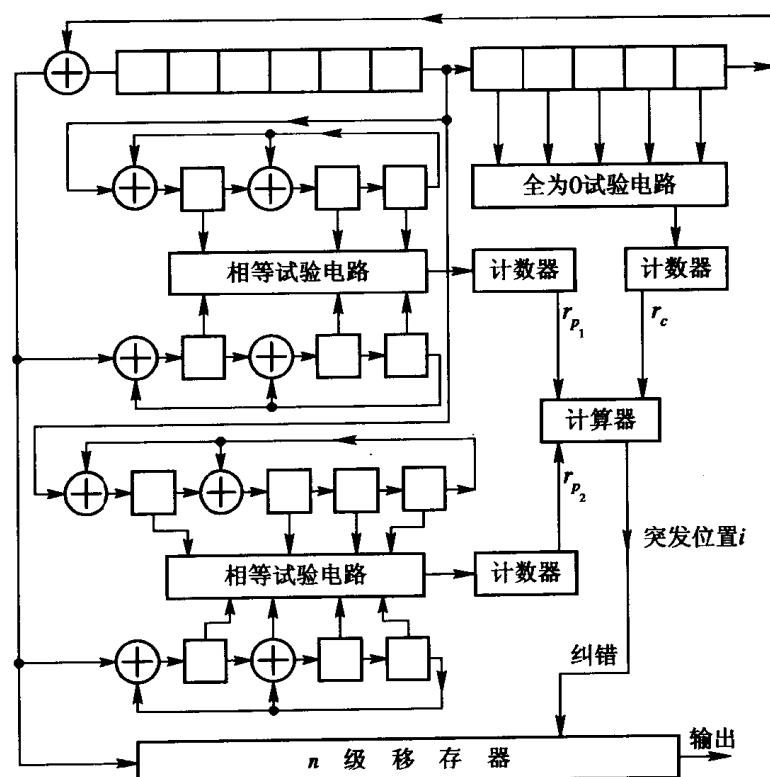


图 9-11 [1155, 1137] 码快速译码器

(1) 收到的 $R(x)$ 同时送入 $p_c(x)=x^{11}+1$, $p_1(x)$, $p_2(x)$ 电路中计算伴随式, 并也送入 n 级缓存器缓存。当第一组 n 次移位完后, 若 3 个计算伴随式电路(除法电路)中的数据均为 0, 则认为接收的 $R(x)$ 无错, 是一个码字。若只有 1 个或 2 个除法电路的数据为 0, 则检测到不可纠正的错误, 若 3 个除法电路中的数据均不为 0, 则认为这是可纠正的错误图样, 译码器开始进行纠错。

(2) 这时突发错误图样已贮存于 $x^{11}+1$ 除法电路中, 使该电路进入自发运算状态, 若

移位 N_c 次后在 x^{11} 至 x^7 的最高次位置上检测到 5 个 0，则说明长度 ≤ 6 的突发 $B(x)$ 已在 $x^{11}+1$ 移存器的最低次位置中，并且求得了 $r_c = 11 - N_c$ 。

(3) 把这个突发图样同时供给 $p_1(x)$ 和 $p_2(x)$ 除法电路，计算伴随式。若分别运算 N_{p_1} 和 N_{p_2} 次后， $p_1(x)$ 和 $p_2(x)$ 的上、下两个除法电路中的内容相等，则停止移位计算，此时得到了：

$$r_{p_1} = e_1 - N_{p_1}$$

$$r_{p_2} = e_2 - N_{p_2}$$

式中， $e_1 = 7$ 是 $p_1(x) = x^3 + x + 1$ 的周期， $e_2 = 15$ 是 $p_2(x) = x^4 + x + 1$ 的周期。由于 11, 7, 15，两两互素，因此可由孙子定理，根据 r_c, r_{p_1}, r_{p_2} 唯一地决定突发位置 i ：

$$\begin{aligned} i &\equiv A_1 A'_1 r_c + A_2 A'_2 r_{p_1} + A_3 A'_3 r_{p_2} \pmod{n} \\ &\equiv 210r_c + 660r_{p_1} + 616r_{p_2} \pmod{1155} \end{aligned}$$

式中

$$A_1 = e_1 e_2 = 105, A_2 = c e_2 = 165, A_3 = c e_1 = 77$$

由 $A_1 A'_1 \equiv 1 \pmod{c=11}$ ，得 $A'_1 = 2$ ；由 $A_2 A'_1 \equiv 1 \pmod{e_1=7}$ ，得 $A'_2 = 4$ ；由 $A_3 A'_2 \equiv 1 \pmod{e_2=15}$ ，得 $A'_3 = 8$ 。

(4) 得到了 i 后，就把贮存在 $x^{11}+1$ 电路中的突发图样送入 n 级移存器的相应位置中，对 $R(x)$ 进行纠正，然后把已纠正的 n 级移存器中的数据并行送至用户，完成了一个码字的纠错。因此至多只需计算 $11+15=26$ 次就完成了译码。■

由上讨论可知，为了使码能用快速译码方法，要求 $g(x) = p_1(x)p_2(x)\cdots p_l(x)$ 中因子的周期必须两两互素。否则，不能直接用上述方法，需要加以适当修正。

习 题

1. 若 $[n, k]$ 循环码要纠正所有长度 $\leq b$ ，同时检测所有长度 $\leq l \geq b$ 的突发，证明校验位 $n-k \geq b+l$ 。
2. 已知 $[n, k, d=2t+1]$ 循环码的生成多项式 $g(x)$ 无 $(x+1)$ 因子，证明用 $g(x) = (x+1)g(x)$ 生成的 $[n, k-1, d=2t+2]$ 码，至少能纠正长度 $\leq t+1$ 的单个突发错误。
3. 构造一个能纠正长度 ≤ 51 ，码长 $n=255$ ，且最佳度 Z 最大的纠突发错误循环码，并画出它的译码器。
4. 构造一个能纠正局限于 4 个码元长的定段突发错误 Burton 码，如果把此码交错 5 次，求出该交错码的纠单个突发的长度，码长 n ，信息位 k 以及最佳度 Z 。
5. 构造一个能纠正长度 ≤ 3 ，既是 R 最佳，又是 S 最佳的纠突发错误循环码，画出它的第 I 和第 II 类译码器。
6. 若把第 5 题中的码交错 5 次，则得到的码是否还是 R 最佳和 S 最佳，为什么？
7. 把表 9-1 中的 $[34, 22]$ 码交错 3 次，并与 4 题中的 Burton 交错码进行比较，试问哪一个码更好？为什么？
8. 证明用 $g(x) = 1 + \sum_{i=0}^{m-1} x^{2^i}$ 生成的二进制码是循环码（提示：从迹的定义出发，证明 $g(x) | x^{2^m-1} - 1$ ）

9. 把第 5 题中的码删去 3 位, 得到一缩短码, 构造它的缩短码的纠突发错误捕错译码器。

10. 符号取自 $GF(2^4)$ 上的 $[15, 10]$ RS 码, 求出它的 $g(x)$, 若把此 RS 码映射成二进制码, 讨论此二进制码的纠错能力。

11. 令 C_1 是由 $g_1(x) = x^2 + x + 1$ 生成的 $[3, 1, 3]$ 二进制循环码, C_2 是由 $g_2(x) = x^4 + x^2 + x + 1$ 生成的 $[7, 3, 4]$ 极长码, 求 $C_1 \otimes C_2$ 乘积码的 $g(x)$ 和 $h(x)$ 及它的最小距离, 讨论它的纠错能力。

12. 证明以 $[n_1, k_1, d_1]$ 作内码, 以 $[n_2, k_2, d_2]$ 作外码所生成的级联码, 最小距离至少为 $d_1 d_2$ 。

13. 用 $g(x) = (x^c - 1)g_0(x)$ 生成的一个缩短循环码有最小距离 $d = 2t + 2$, 并能纠正长度 $\leq b$ 的任何单个突发错误, 证明该码既能纠正任何重量 $\leq t$ 的随机错误图样, 又能同时纠正任何长度 $\leq \min(c, b)$ 的突发图样。(提示: 证明任何重量 $\leq t$ 的错误图样与任何长度 $\leq \min(c, b)$ 的突发图样不在同一陪集中。)

14. 证明码长为 15 的二进制极长码, 既能纠正 $t \leq 3$ 个随机错误, 也能够同时纠正长度 ≤ 5 的突发错误。

15. 构造一个行码和列码都是奇偶校验码的乘积码, 要求该乘积码能纠正长度 ≤ 4 的单个突发错误, 并有最高码率。求出码的参数, 列出码阵的数据传输次序。

参 考 文 献

- [1] W. W. Peterson and E. J. Weldon, Jr., *Error-Correcting Codes*, MIT Press, 1971.
- [2] 王新梅:《纠错码与差错控制》, 人民邮电出版社, 1989.
- [3] H. J. Matt and J. L. Massey, "Determining the burst error-correcting limit of cyclic codes", IEEE Trans. on IT, No. 3, pp. 289—297, 1980.
- [4] Wang Xinmei, "Lower bound on the burst-error-correcting ability of the BCH codes and its relation to the roots of the codes", Proceeding of ISIT, Les Arcs, France, June 21—25, 1982.
- [5] 常盤, 笠原, 滑川, BCH 符号を用いた複合誤ソ訂正に關考察, 信学技報, Vol. 82, No. 263, pp. 35—40, 1983.
- [6] K. A. S. Abdel Ghaffar, R. J. McEliece, et al., "On the existence of optimum cyclic burst correcting codes", IEEE Trans. on IT, No. 6, pp. 768—775, 1986.
- [7] K. A. S. Abdel Ghaffar, "On the existence of optimum cyclic burst correcting codes over $GF(q)$ ", IEEE Trans. on IT, No. 2, pp. 329—332, 1988.
- [8] M. Blaum, et al., "A class of burst error-correcting array codes", IEEE Trans. on IT., No. 6, pp. 336—339, 1986.
- [9] W. Zhang and J. K. Wolf, "A class of binary burst error-correcting quasi-cyclic codes", IEEE Trans. on IT, No. 3, pp. 463—480, 1988.
- [10] M. Blaum, "A family of efficient burst-correcting array codes", IEEE Trans. on IT, No. 3, pp. 671—675, 1990.
- [11] M. Blaum, et al., "Multiple burst-correcting array codes", IEEE Trans. , on IT, No. 5, pp. 1061—1066, 1988.
- [12] A. A. Hassang and W. E. Stark, "On decoding concatenated codes", IEEE Trans. on IT, No. 3, pp. 683—685, 1990.

- [13] R. G. Gallager, Information Theory and Reliable Communication, John Wiley and Sons, New York, 1968.
- [14] W. Adi, "Fast burst error—correction scheme with fire code", IEEE Trans. on Computer No. 7, 1984.
- [15] [美]林舒、科斯特洛:《差错控制编码:基础和应用》(王育民、王新梅译),人民邮电出版社,1986.
- [16] 王新梅,曾开明:“级连码(K, N)门限通信密钥分散保管系统”,《通信学报》,No. 4, pp. 1—9, 1987.

第十章 卷积码基础

卷积码是 1955 年由爱里斯(Elias)提出的，它与以前各章所讨论的分组码不相同。分组码编码时，本组中的 $n - k$ 个校验元仅与本组的 k 个信息元有关，而与其它各组码元无关。分组码译码时，也仅从本码组中的码元内提取有关译码信息，而与其它各组无关。但是，卷积码编码中，本组的 $n_0 - k_0$ 个校验元不仅与本组的 k_0 个信息元有关，而且还与以前各时刻输入至编码器的信息组有关。同样，在卷积码译码过程中，不仅从此时刻收到的码组中提取译码信息，而且还要利用以前或以后各时刻收到的码组中提取有关信息。此外，卷积码中每组的信息位 k_0 和码长 n_0 ，通常也比分组码的 k 和 n 要小。

正由于在卷积码的编码过程中，充分利用了各组之间的相关性，且 k_0 和 n_0 也较小，因此，在与分组码同样的码率 R 和设备复杂性条件下，无论从理论上还是从实际上均已证明卷积码的性能至少不比分组码差，且实现最佳和准最佳译码也较分组码容易。所以，从信道编码定理看，卷积码是一种非常有前途的，能达到信道编码定理所提出的码类。但由于卷积码各组之间相互有关，因此在卷积码的分析过程中，至今仍未找到像分组码那样有效的数学工具，以致性能分析比较困难，从分析上得到的成果也不像分组码那样多，而往往还要借助计算机的搜索来寻找好码。

但由于卷积码各组的 n_0, k_0 均比分组码小，译码似乎比分组码要容易，并且卷积码有三种比较好的译码方法：(1)1963 年由梅西(Massey)提出的门限译码，这是一种利用码代数结构的代数译码，类似于分组码中的大数逻辑译码；(2)1961 年由沃曾克拉夫特(Wozen-craft)提出，1963 年由费诺(Fano)改进的序列译码，这是基于码树图结构上的一种准最佳的概率译码；(3)1967 年由维特比(Viterbi)提出的 Viterbi 算法，这是基于码的网(trellis)图基础上的一种最大似然译码算法，是一种最佳的概率译码方法。

本章先介绍卷积码的基本概念、定义和它们的各种表示方法，以后两章分别介绍用大数逻辑译码方法译码的纠随机与纠突发错误的卷积码和卷积码的概率译码。本章仅讨论二进制卷积码，当然所讨论的结果可推广到 q 进制情况。

§ 10.1 卷积码的基本概念

可以从很多不同观点来讨论卷积码，如同分组码那样，既可以从码的生成矩阵和校验矩阵观点，也可从码的多项式观点讨论，同样也可从矩阵和多项式观点研究卷积码。此外，由于卷积码的特点，也可从树图和网图的观点研究。

图 10-1 给出了一个二进制卷积码的编码器。若每一时间单位输入编码器一

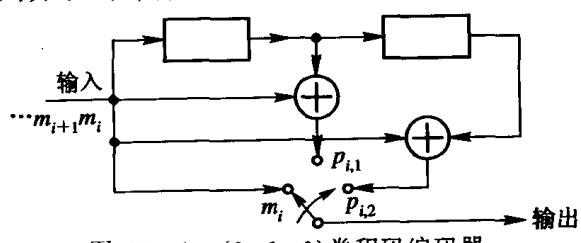


图 10-1 (3, 1, 2) 卷积码编码器

一个新的信息元 m_i , 且存贮器内的数据往右移一位, 则 m_i 一方面直接输出至信道, 另一方面与前两个单位时间送入的信息元 m_{i-1} 、 m_{i-2} 按图中线路所确定的规则进行运算, 得到此时刻的两个校验元 $p_{i,1}$ 、 $p_{i,2}$, 跟随在 m_i 后面组成一个子码 $c_i = (m_i, p_{i,1}, p_{i,2})$ 送入信道。由图可知:

$$\begin{aligned} p_{i,1} &= m_i + m_{i-1} \\ p_{i,2} &= m_i + m_{i-2} \end{aligned} \quad (10.1.1)$$

下一个时间单位输入的信息元为 m_{i+1} 与其相应的两个校验元:

$$\begin{aligned} p_{i+1,1} &= m_{i+1} + m_i \\ p_{i+1,2} &= m_{i+1} + m_{i-1} \end{aligned}$$

组成第二个子码 $c_{i+1} = (m_{i+1}, p_{i+1,1}, p_{i+1,2})$ 送至信道, 如此等等。在每一时间单位, 送至编码器 k_0 (这里为 1)个信息元, 编码器就送出相应的 n_0 (这里为 3)个码元组成一个子码 c_i 送入信道, 在卷积码中, 这 n_0 个码元组成的子码 c_i 有时也称卷积码的一个码段或子组。

由上可知, 第 i 时刻输入至编码器的信息组 m_i 及其相应的码段 c_i , 不仅与前 m 个码段 $c_{i-1}, c_{i-2}, \dots, c_{i-m}$ 中的码元有关, 而且也参与了后 m (这里为 2) 个码段 $c_{i+1}, c_{i+2}, \dots, c_{i+m}$ 中的校验运算, 如图 10-2 中的虚线方框内的关系表示。这正如一条链子, 它的每一环都与前后各环有关, 这样一环扣一环就组成了卷积码的一个码序列。因此, 这种卷积码也称连环码, 以后将看到这种运算在数学上称为卷积运算。

由式(10.1.1)和图 10-1 可知, 用这种卷积码编码器输出的每一子码中的校验元, 是此时刻输入的信息元与前 m (这里为 2)个子码中信息元的模 2 和, 它们是线性关系, 所以由这类编码器编出的卷积码是线性码。今后我们仅讨论线性卷积码, 称 m 为**编码存贮**, 它表示输入信息组在编码器中需存贮的单位时间。称 $m+1=N$ 为**编码约束度**, 说明编码过程中互相约束的码段个数。称 $Nn_0=N_A$ 为**编码约束长度**, 说明编码过程中互相约束的码元个数。在图 10-1 中的卷积码, $m=2$, $N=3$, $Nn_0=6$ 。由此可知, m 或 N 、 N_A 是表示卷积码编码器复杂性的一个重要参数。

同样, 在卷积码译码过程中, 不仅要根据此时刻输入到译码器的子码, 而且还要根据以后很长一段时间如 m_d 段单位时间内, 收到的各子码, 才能译出一个子码信息元, 通常 $m_d \geq m$ 。我们称 $m_d+1=N_d$ 为**译码约束度**, 称 n_0N_d 为**译码约束长度**。它们分别表示译码过程中互相约束的码段或码元个数。总之, 在卷积码的各码段之间, 不论是编码还是译码都不是每段各自处理, 而是与前后 m 或 m_d 段有关。所以, 卷积码通常用 (n_0, k_0, m) 或 (n_0N, k_0N) 表示。 k_0 和 n_0 比分组码的 k 和 n 通常要小, 而它们的比值 $k_0/n_0=R$, 同分组码一样, 称为卷积码的**码率**, 它是衡量卷积码传输信息有效性的一个重要参数。

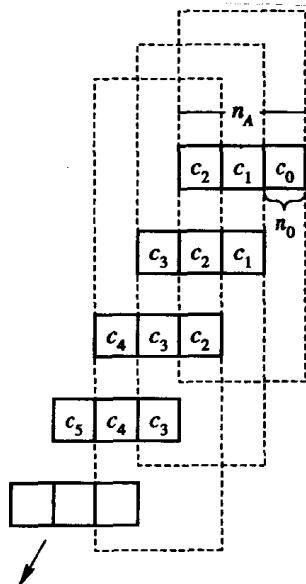


图 10-2 卷积码子码之间约束关系

§ 10.2 卷积码的矩阵和多项式描述

描述卷积编码译码过程的方法很多，如：矩阵方法，多项式方法，码树和格、网图表示方法等。但由于卷积码的表示方法与它所采用的译码方法有密切关系，因此这一节我们仅介绍与代数译码有关的矩阵和多项式表示方法。

一、卷积码的校验矩阵和生成矩阵

卷积码的一般编码器如图 10-3 所示。在某一时刻 i ，输入到编码器的是由 k_0 个信息元组成的信息组 m_i ，相应地输出序列是由 n_0 个码元组成的子码 c_i 。若输入的信息序列 $m_0, m_1, \dots, m_i, \dots$ 是一个半无限序列，则由卷积码编码器输出的序列，也是一个由各子码 $c_0, c_1, \dots, c_i, \dots$ 组成的半无限长序列，称此序列为卷积码的一个码序列或码字。

如同分组码那样，卷积码码字中的每一个子码 c_i ，它的某一段码元（通常在最左边 k_0 个码元）可以是原来的 k_0 个信息元，而其余的是 $n_0 - k_0$ 个校验元，这是系统码的形式，如图 10-1 编码器编出的码。当然，也可以编出非系统码形式的卷积码。但卷积码与分组码不同，在同样的码参数下，系统卷积码与非系统卷积码的抗干扰性能不一定相同。

由分组码理论可知，无论是系统码还是非系统码，只要码的校验矩阵或生成矩阵一旦确定，则码也就完全确定了。也就是说码完全可由它的校验矩阵或生成矩阵来描述。同样，在卷积码中，码也完全可由它的生成矩阵或校验矩阵描述。

仍以图 10-1 的 $(3, 1, 2)$ 系统卷积码为例，说明它的生成矩阵是如何得出的。设编码器的初始状态全为 0，若输入的信息序列 $M = (m_0, m_1, m_2, \dots) = (1\ 0\ 0\ \dots)$ ，则由编码器输出的码序列 $C = (c_0, c_1, c_2, \dots) = (111, 010, 001, 000, \dots)$ 。码序列 C 中第 $m+1=3$ 段以后，后面各段取值均为 0，说明 $m_0=1$ 信息元输入到编码器后，至多只影响后面的 m 段码元，当 $m+1$ 段以后 m_0 已移出编码器，不再对以后各段产生任何影响了。如果信息序列

$$M = (1\ 1\ 1\ 0\ 0\ 0\ \dots) = (1\ 0\ 0\ \dots) + (0\ 1\ 0\ 0\ \dots) + (0\ 0\ 1\ 0\ \dots) + \dots$$

则由编码器输出的码序列

$$\begin{aligned} C &= (111, 010, 001, 000, 000, 000, \dots) + (000, 111, 010, 001, 000, 000, \dots) \\ &\quad + (000, 000, 111, 010, 001, 000, \dots) + \dots \\ &= (111, 101, 100, 011, 001, 000, \dots) \end{aligned}$$

若把上述运算结果写成如下矩阵形式，则

$$\begin{aligned} C &= MG_{\infty} = [1\ 1\ 1\ 0\ 0\ \dots] \begin{bmatrix} 111 & 010 & 001 & 000 & 000 & \dots \\ 000 & 111 & 010 & 001 & 000 & \dots \\ 000 & 000 & 111 & 010 & 001 & \dots \\ \dots & & & & & \end{bmatrix} \\ &= (111, 101, 100, 011, 001, 000, \dots) \end{aligned} \tag{10.2.1}$$

我们称 G_{∞} 为该卷积码的生成矩阵。由此矩阵看出，这是一个半无限矩阵。它有无限多的行

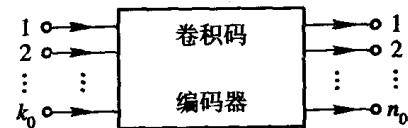


图 10-3 卷积码编码器

和列。但是，仔细观察 G_∞ 可知，它的每一行都是前一行右移 $n_0(3)$ 位的结果，也就是说它完全由第一行决定。若第一行决定了，则该码的 G_∞ 矩阵也就完全决定了。我们把决定 G_∞ 的第一行取出，并表示成

$$g_\infty = [111, 010, 001, 000, 000, \dots] = [g_0, g_1, g_2, 0, 0, \dots] \quad (10.2.2)$$

称 g_∞ 为该码的基本生成矩阵。上式中的 $g_i (i=0, 1, 2)$ 与 0 都是一个 $1 \times 3 (k_0 \times n_0)$ 阶矩阵。

如果我们用迟延算子 D （也可用 x ），表示卷积码编码过程中一个单位时间 (n_0 个码元) 的迟延，则式(10.2.1)中的 G_∞ 矩阵可写成

$$G_\infty = \begin{bmatrix} g_\infty \\ Dg_\infty \\ D^2g_\infty \\ \vdots \end{bmatrix} = \begin{bmatrix} g_0 & g_1 & g_2 & 0 & \cdots \\ 0 & g_0 & g_1 & g_2 & 0 & \cdots \\ 0 & 0 & g_0 & g_1 & g_2 & 0 & \cdots \\ \cdots & & & & & & \end{bmatrix} \quad (10.2.3)$$

g_∞ 完全由前 $m+1=3$ 段的值 g_0, g_1, g_2 决定，从第 $m+2=4$ 段起均为 0 。而每一个 $g_i (i=0, 1, 2)$ 由 $3 (n_0=3)$ 个数字决定，若我们把 g_∞ 中前 $m=3$ 段 g_0, g_1, g_2 中的每一段的第 $j (j=1, 2, 3)$ 个数字，用如下 $3 (n_0=3)$ 个多项式表示：

$$\begin{aligned} g^{(1,1)}D &= 1 \\ g^{(1,2)}D &= 1 + D \\ g^{(1,3)}D &= 1 + D^2 \end{aligned} \quad (10.2.4)$$

则 g_∞ 完全由式(10.2.4)确定。我们称式(10.2.4)中的每一个多项式为该卷积码的子生成元，它的最高次数是 m 次。由此可知，只要码的子生成元确定，则码的 G_∞ 也就确定了。

由子生成元，我们可把码的生成矩阵 G_∞ 写成 D 的函数形式：

$$G(D) = [g^{(1,1)}(D), g^{(1,2)}(D), g^{(1,3)}(D)] = [1, 1 + D, 1 + D^2] \quad (10.2.5)$$

称 $G(D)$ 为 (n_0, k_0, m) 卷积码的生成多项式矩阵或变换函数。

如果把信息序列 $M=(1\ 1\ 1\ 0\ 0\ \dots)$ 也写成 D 的函数，则

$$M(D) = (m_0 + m_1D + m_2D^2 + \dots) = (1 + D + D^2 + 0 + 0 \dots)$$

相应的码序列 C 也可写成 D 的函数：

$$\begin{aligned} C(D) &= (c_0 + c_1D + c_2D^2 + \dots) \\ &= (111) + (101)D + (100)D^2 + (011)D^3 + (001)D^4 + 0 \dots \end{aligned} \quad (10.2.6)$$

称它为卷积码码序列的多项式表示。与分组码码字多项式表示进行比较可知，在卷积码的多项式表示中，每一项的系数是一个 n_0 重矢量（子码），而不是一个数值。正由于 $C(D)$ 中的每一系数 c_i 由 n_0 个分量组成，因而若我们把所有系数（子码）的第 $j (j=1, 2, \dots, n_0)$ 个分量写成多项式 $C^{(j)}(D)$ ，则

$$C^{(j)}(D) = c_{0j} + c_{1j}D + c_{2j}D^2 + \dots \quad j = 1, 2, \dots, n_0$$

在该例中有：

$$C^{(1)}(D) = 1 + D + D^2 + 0 \dots$$

$$C^{(2)}(D) = 1 + D^3 + 0 \dots$$

$$C^{(3)}(D) = 1 + D + D^3 + D^4 + 0 \dots$$

因此，一旦 $C^{(j)}(D) (j=1, 2, \dots, n_0)$ 确定后，则 $C(D)$ 也就完全确定了。所以，编码器的任务就是如何由 $M(D)$ 得到 $C^{(j)}(D)$ 。

由 $C = MG_\infty$, 我们也可以用它们的多项式表示出 $C(D)$ 。

$$\begin{aligned}
 C(D) &= M(D)G(D) \\
 &= M(D)(g^{(1,1)}(D), g^{(1,2)}(D), g^{(1,3)}(D)) \\
 &= (M(D)g^{(1,1)}(D), M(D)g^{(1,2)}(D), M(D)g^{(1,3)}(D)) \\
 &= (M(D) \cdot 1, M(D)(1+D), M(D)(1+D^2)) \\
 &= ((1+D+D^2+0\cdots), (1+D+D^2+0\cdots)(1+D), \\
 &\quad (1+D+D^2+0\cdots)(1+D^2)) \\
 &= (1+D+D^2+0\cdots, 1+D^3+0\cdots, 1+D+D^3+D^4+0\cdots) \\
 &= (C^{(1)}(D), C^{(2)}(D), C^{(3)}(D)) \\
 &= (111) + (101)D + (100)D^2 + (011)D^3 + (001)D^4 + 0\cdots
 \end{aligned}$$

该式与(10.2.6)式完全相同。 $M(D)$ 是卷积码器的输入, $C(D)$ 是输出, 因而卷积码的生成多项式矩阵

$$G(D) = \frac{C(D)}{M(D)}$$

是输出与输入之比, 这正是网络的传输函数, 因此我们也可以从网络理论来讨论卷积码编码器。上述求 $C(D) = M(D)G(D)$ 的运算过程, 就是输入信息序列和子生成元的卷积运算求 $C^{(j)}(D)$ 的过程:

$$\begin{aligned}
 C^{(j)}(D) &= \sum_{i=1}^{k_0} M^{(i)}(D)g^{(i,j)}(D) \\
 &= c_{0j} + c_{1j}D + c_{2j}D^2 + c_{3j}D^3 + \cdots \quad j = 1, 2, \dots, n_0
 \end{aligned} \tag{10.2.7}$$

该例中 $k_0=1$, $n_0=3$, 故

$$C^{(j)}(D) = M(D)g^{(1,j)}(D) \quad j = 1, 2, 3 \tag{10.2.8}$$

这正是卷积码名称的来源。

为了更好地理解卷积码生成矩阵和生成元, 下面我们再举一个 $k_0>1$ 的例子。

图 10-4 画出了信息元并行输入的 $(3, 2, 2)$ 系统卷积码编码器。设编码器初始状态全为 0, 输入的信息序列:

$$\begin{aligned}
 M &= (M^{(1)} + M^{(2)}) \\
 &= (m_0^{(1)}, 0, m_1^{(1)}, 0, m_2^{(1)}, 0, \dots) + (0, m_0^{(2)}, 0, m_1^{(2)}, 0, m_2^{(2)}, \dots) \\
 &= (m_0^{(1)}m_0^{(2)}, m_1^{(1)}m_1^{(2)}, m_2^{(1)}m_2^{(2)}, \dots)
 \end{aligned}$$

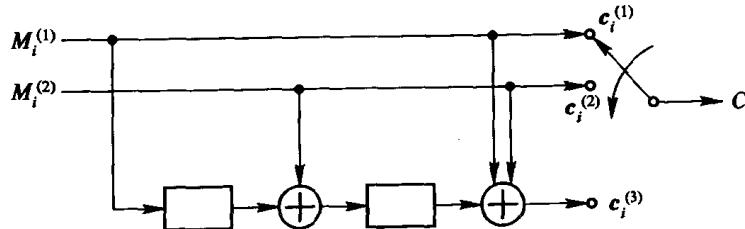


图 10-4 $(3, 2, 2)$ 卷积码编码器

若第一个信息序列

$$\mathbf{M}^{(1)} = (10, 00, 00, \dots)$$

则由图可知相应的输出码序列

$$\mathbf{C}^{(1)} = (\mathbf{c}_0^{(1)}, \mathbf{c}_1^{(1)}, \mathbf{c}_2^{(1)}, \dots) = (101, 000, 001, 000, 000, \dots)$$

第二个信息序列

$$\mathbf{M}^{(2)} = (01, 00, 00, \dots)$$

则相应的输出码序列

$$\mathbf{C}^{(2)} = (\mathbf{c}_0^{(2)}, \mathbf{c}_1^{(2)}, \mathbf{c}_2^{(2)}, \dots) = (011, 001, 000, 000, \dots)$$

因而若 $\mathbf{M} = \mathbf{M}_1^{(1)} + \mathbf{M}_2^{(2)} = (11, 00, 00, \dots)$, 则相应输出的码序列

$$\begin{aligned}\mathbf{C} &= \mathbf{C}^{(1)} + \mathbf{C}^{(2)} = (\mathbf{c}_0^{(1)} + \mathbf{c}_0^{(2)}, \mathbf{c}_1^{(1)} + \mathbf{c}_1^{(2)}, \dots) \\ &= \begin{pmatrix} 101 & 000 & 001 & 000 & 000 & \dots \\ 011 & 001 & 000 & 000 & 000 & \dots \end{pmatrix} \\ &= (110, 001, 001, 000, 000, \dots)\end{aligned}$$

显然, 若输入信息序列

$$\begin{aligned}\mathbf{M} &= (11, 11, 11, 11, \dots) \\ &= (11, 00, 00, \dots) + (00, 11, 00, 00, \dots) \\ &\quad + (00, 00, 11, 00, \dots) + (00, 00, 00, 11, \dots)\end{aligned}$$

则相应输出的码序列便成为

$$\begin{aligned}\mathbf{C} &= \begin{pmatrix} 101 & 000 & 001 & 000 & 000 & 000 & 000 & \dots \\ 011 & 001 & 000 & 000 & 000 & 000 & 000 & \dots \end{pmatrix} \\ &\quad + \begin{pmatrix} 000 & 101 & 000 & 001 & 000 & 000 & 000 & \dots \\ 000 & 011 & 001 & 000 & 000 & 000 & 000 & \dots \end{pmatrix} \\ &\quad + \begin{pmatrix} 000 & 000 & 101 & 000 & 001 & 000 & 000 & \dots \\ 000 & 000 & 011 & 001 & 000 & 000 & 000 & \dots \end{pmatrix} \\ &\quad + \begin{pmatrix} 000 & 000 & 000 & 101 & 000 & 001 & 000 & \dots \\ 000 & 000 & 000 & 011 & 001 & 000 & 000 & \dots \end{pmatrix} + \dots \\ &= (110, 111, 110, \dots)\end{aligned}$$

把上式写成矩阵形式有

$$\mathbf{C} = \mathbf{M}\mathbf{G}_{\infty} = (11, 11, 11, \dots) \begin{bmatrix} 101 & 000 & 001 & 000 & \dots \\ 011 & 001 & 000 & 000 & \dots \\ 101 & 000 & 001 & 000 & \dots \\ 011 & 001 & 000 & 000 & \dots \\ 101 & 000 & 001 & 000 & \dots \\ 011 & 001 & 000 & 000 & \dots \\ \dots & \dots & \dots & \dots & \dots \end{bmatrix}$$

显然, \mathbf{G}_{∞} 完全由最上面两行决定。因而该 \mathbf{G}_{∞} 矩阵的基本生成矩阵

$$\mathbf{g}_{\infty} = \begin{pmatrix} 101 & 000 & 001 & 000 & \dots \\ 011 & 001 & 000 & 000 & \dots \end{pmatrix} = [\mathbf{g}_0, \mathbf{g}_1, \mathbf{g}_2, \mathbf{0}, \dots] \quad (10.2.9)$$

式中, \mathbf{g}_i 是一个 $k_0 \times n_0 = 2 \times 3$ 阶阵, 仔细观察 \mathbf{g}_0 可知, \mathbf{g}_0 的最左边 $k_0 \times k_0 = 2 \times 2$ 阶阵是一个单位方阵, 设用 \mathbf{I}_{k_0} 表示, 而其余的 $k_0 \times (n_0 - k_0) = 2 \times 1$ 阶阵用 \mathbf{P}_0 表示。 $\mathbf{g}_1, \mathbf{g}_2$ 阵

最左边的 $k_0 \times k_0 = 2 \times 2$ 阶阵是全为 $\mathbf{0}$ 阵，而其余的 $k_0 \times (n_0 - k_0) = 2 \times 1$ 阶阵非全为 0 ，设分别用 $\mathbf{p}_1, \mathbf{p}_2$ 表示，则

$$\mathbf{g}_{\infty} = [\mathbf{g}_0, \mathbf{g}_1, \mathbf{g}_2, \mathbf{0}, \dots] = [\mathbf{I}_{k_0} \mathbf{p}_0, \mathbf{0} \mathbf{p}_1, \mathbf{0} \mathbf{p}_2, \mathbf{0}, \dots] \quad (10.2.10)$$

式中，自第 $m+1=3$ 段以后开始，取值均为 0 ，说明某一时刻输入的信息元，至多只影响此时刻和以后 m 个单位时间的子码，而在 $m+1$ 个时间单位以后，信息元已移出编码器，不再对以后各段子码的编码发生影响。而第 0 段中的 \mathbf{I}_{k_0} 是一个 $k_0 \times k_0$ 阶单位方阵，说明码字中每一段前 k_0 位就是输入的信息元，而后 $n_0 - k_0$ 位则由 $\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2$ 决定的校验方程得到。因而，由这种 \mathbf{g}_{∞} 决定的码就是系统码。

把式(10.2.10)中的 \mathbf{g}_{∞} 代入 \mathbf{G}_{∞} 矩阵，则

$$\mathbf{G}_{\infty} = \begin{bmatrix} \mathbf{g}_0 & \mathbf{g}_1 & \mathbf{g}_2 \\ \mathbf{g}_0 & \mathbf{g}_1 & \mathbf{g}_2 \\ \mathbf{g}_0 & \mathbf{g}_1 & \mathbf{g}_2 \\ \dots & & & \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{k_0} \mathbf{p}_0 & \mathbf{0} \mathbf{p}_1 & \mathbf{0} \mathbf{p}_2 \\ \mathbf{I}_{k_0} \mathbf{p}_0 & \mathbf{0} \mathbf{p}_1 & \mathbf{0} \mathbf{p}_2 \\ \mathbf{I}_{k_0} \mathbf{p}_0 & \mathbf{0} \mathbf{p}_1 & \mathbf{0} \mathbf{p}_2 \\ \dots & & & \end{bmatrix} = \begin{bmatrix} \mathbf{g}_{\infty} \\ D \mathbf{g}_{\infty} \\ D^2 \mathbf{g}_{\infty} \\ \vdots \end{bmatrix} \quad (10.2.11)$$

矩阵中所有空白的地方全为 0 。

由式(10.2.9)可知，该例中的 \mathbf{g}_{∞} 由 $k_0 = 2$ 行组成。因而，与此 \mathbf{g}_{∞} 相应的第一行与第二行的子生成元分别为： $\mathbf{g}^{(1,1)}(D) = 1, \mathbf{g}^{(1,2)}(D) = 0, \mathbf{g}^{(1,3)}(D) = 1 + D^2; \mathbf{g}^{(2,1)}(D) = 0, \mathbf{g}^{(2,2)}(D) = 1, \mathbf{g}^{(2,3)}(D) = 1 + D$ 。共有 $k_0 \times n_0 = 6$ 个子生成元。如果我们定义 \mathbf{g}_{∞} 的第一行数据为该码的第一个生成元 $\mathbf{g}(1)$ ，则 (n_0, k_0, m) 码共有 k_0 个生成元，在该例中有 $k_0 = 2$ 个生成元，它们分别是：

$$\begin{aligned} \mathbf{g}(1) &= [101, 000, 001] \\ \mathbf{g}(2) &= [011, 001, 000] \end{aligned}$$

生成元与子生成元的关系为：

$$\begin{aligned} \mathbf{g}(1) &= [\mathbf{g}_1^{(1,1)} \mathbf{g}_1^{(1,2)} \mathbf{g}_1^{(1,3)}, \mathbf{g}_2^{(1,1)} \mathbf{g}_2^{(1,2)} \mathbf{g}_2^{(1,3)}, \mathbf{g}_3^{(1,1)} \mathbf{g}_3^{(1,2)} \mathbf{g}_3^{(1,3)}] \\ \mathbf{g}(2) &= [\mathbf{g}_1^{(2,1)} \mathbf{g}_1^{(2,2)} \mathbf{g}_1^{(2,3)}, \mathbf{g}_2^{(2,1)} \mathbf{g}_2^{(2,2)} \mathbf{g}_2^{(2,3)}, \mathbf{g}_3^{(2,1)} \mathbf{g}_3^{(2,2)} \mathbf{g}_3^{(2,3)}] \end{aligned}$$

式中， $\mathbf{g}_i^{(i,j)}$ ($i=1, 2, \dots, k_0; l, j=1, 2, \dots, n_0$) 是 $\mathbf{g}^{(i,j)}(D)$ 子生成元的 D^l 次项的系数。由此可知，只要码的生成元或子生成元确定了，则码的 \mathbf{G}_{∞} 也就完全确定了。

利用子生成元的多项式表示，我们可把 \mathbf{G}_{∞} 表示为如下的生成多项式矩阵：

$$\mathbf{G}(D) = \begin{bmatrix} \mathbf{g}^{(1,1)}(D) & \mathbf{g}^{(1,2)}(D) & \mathbf{g}^{(1,3)}(D) \\ \mathbf{g}^{(2,1)}(D) & \mathbf{g}^{(2,2)}(D) & \mathbf{g}^{(2,3)}(D) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 + D^2 \\ 0 & 1 & 1 + D \end{bmatrix} \quad (10.2.12)$$

若输入的信息序列 $\mathbf{M} = (11, 11, 11, 00, \dots) = (\mathbf{M}^{(1)}, \mathbf{M}^{(2)})$ ，相应的 $\mathbf{M}(D) = (11) + (11)D + (11)D^2 + \dots = (1 + D + D^2 + \dots, 1 + D + D^2 + \dots)$ ，则由编码器输出的码序列多项式为

$$\begin{aligned} \mathbf{C}(D) &= \mathbf{M}(D)\mathbf{G}(D) = (\mathbf{M}^{(1)}(D), \mathbf{M}^{(2)}(D)) \begin{bmatrix} \mathbf{g}^{(1,1)}(D) & \mathbf{g}^{(1,2)}(D) & \mathbf{g}^{(1,3)}(D) \\ \mathbf{g}^{(2,1)}(D) & \mathbf{g}^{(2,2)}(D) & \mathbf{g}^{(2,3)}(D) \end{bmatrix} \\ &= \left(\sum_{i=1}^2 \mathbf{M}^{(i)}(D) \mathbf{g}^{(i,1)}(D), \sum_{i=1}^2 \mathbf{M}^{(i)}(D) \mathbf{g}^{(i,2)}(D), \sum_{i=1}^2 \mathbf{M}^{(i)}(D) \mathbf{g}^{(i,3)}(D) \right) \\ &= (1 + D + D^2 + 0 + \dots, 1 + D + D^2 + 0 \dots, D + D^4 + 0 \dots) \end{aligned}$$

$$\begin{aligned}
&= (\mathbf{C}^{(1)}(D), \mathbf{C}^{(2)}(D), \mathbf{C}^{(3)}(D)) \\
&= (110) + (111)D + (110)D^2 + (000)D^3 + (001)D^4 + \dots
\end{aligned}$$

与以前结果相同。上式中 $\mathbf{M}^{(i)}(D)$ 是由 \mathbf{M} 中每段第 i (1 或 2)个信息元组成的序列。

通过上面两个例子，我们不难看出 (n_0, k_0, m) 卷积码生成矩阵 \mathbf{G}_∞ 的一般形式为

$$\mathbf{G}_\infty = \left[\begin{array}{cccccc} \mathbf{g}_0 & \mathbf{g}_1 & \cdots & \cdots & \mathbf{g}_m \\ \mathbf{g}_0 & \mathbf{g}_1 & \cdots & \mathbf{g}_{m-1} & \mathbf{g}_m \\ \mathbf{g}_0 & \mathbf{g}_1 & \cdots & \mathbf{g}_{m-2} & \mathbf{g}_{m-1} & \mathbf{g}_m \\ \vdots & \vdots & & & & \\ \mathbf{g}_0 & \mathbf{g}_1 & \cdots & \mathbf{g}_m \\ \cdots & & & & & \infty \end{array} \right] \quad (10.2.13)$$

而基本生成矩阵

$$\mathbf{g}_\infty = [\mathbf{g}_0, \mathbf{g}_1, \mathbf{g}_2, \dots, \mathbf{g}_m, 0, \dots] \quad (10.2.14)$$

式中, $\mathbf{g}_i (i=0, 1, 2, \dots, m)$ 是一个 $k_0 \times n_0$ 阶阵, 而空白的地方全为 0。 \mathbf{g}_∞ 共有 k_0 个生成元

$$\mathbf{g}_\infty = \begin{bmatrix} \mathbf{g}(1) \\ \mathbf{g}(2) \\ \vdots \\ \mathbf{g}(k_0) \end{bmatrix}$$

而每一个生成元 $\mathbf{g}(j)$ 的子生成元为: $\mathbf{g}^{(j, 1)}(D), \mathbf{g}^{(j, 2)}(D), \dots, \mathbf{g}^{(j, n_0)}(D), j = 1, 2, \dots, k_0$ 。因而相应的生成多项式矩阵

$$\mathbf{G}(D) = \begin{bmatrix} \mathbf{g}^{(1, 1)}(D) & \mathbf{g}^{(1, 2)}(D) & \cdots & \mathbf{g}^{(1, n_0)}(D) \\ \mathbf{g}^{(2, 1)}(D) & \mathbf{g}^{(2, 2)}(D) & \cdots & \mathbf{g}^{(2, n_0)}(D) \\ \vdots & \vdots & & \vdots \\ \mathbf{g}^{(k_0, 1)}(D) & \mathbf{g}^{(k_0, 2)}(D) & \cdots & \mathbf{g}^{(k_0, n_0)}(D) \end{bmatrix} \quad (10.2.15)$$

若 (n_0, k_0, m) 是一个系统卷积码，则 $\mathbf{g}_\infty, \mathbf{G}_\infty$ 和 $\mathbf{G}(D)$ 分别为：

$$\mathbf{g}_\infty = [\mathbf{I}_{k_0} \mathbf{p}_0, \mathbf{0} \mathbf{p}_1, \mathbf{0} \mathbf{p}_2, \dots, \mathbf{0} \mathbf{p}_m, \mathbf{0}, \dots]$$

$$\mathbf{G}_\infty = \left[\begin{array}{ccccc} \mathbf{I}_{k_0} \mathbf{p}_0 & \mathbf{0} \mathbf{p}_1 & \mathbf{0} \mathbf{p}_2 & \cdots & \mathbf{0} \mathbf{p}_m \\ \mathbf{I}_{k_0} \mathbf{p}_0 & \mathbf{0} \mathbf{p}_1 & \cdots & \cdots & \mathbf{0} \mathbf{p}_m \\ \vdots & & & & \\ \mathbf{I}_{k_0} \mathbf{p}_0 & \mathbf{0} \mathbf{p}_1 & \cdots & \mathbf{0} \mathbf{p}_m \\ \cdots & & & & \\ & & & & \infty \end{array} \right] \quad (10.2.16)$$

$$\begin{aligned}
\mathbf{G}(D) &= \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & \mathbf{g}^{(1, k_0+1)}(D) & \cdots & \mathbf{g}^{(1, n_0)}(D) \\ 0 & 1 & 0 & \cdots & 0 & \mathbf{g}^{(2, k_0+1)}(D) & \cdots & \mathbf{g}^{(2, n_0)}(D) \\ \vdots & \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 & \mathbf{g}^{(k_0, k_0+1)}(D) & \cdots & \mathbf{g}^{(k_0, n_0)}(D) \end{bmatrix} \\
&= [\mathbf{I}_{k_0} \mathbf{P}(D)]
\end{aligned} \quad (10.2.17)$$

式中, $\mathbf{P}(D)$ 是一个以子生成元 $\mathbf{g}^{(i, j)} (i = 1, 2, \dots, k_0; j = k_0 + 1, \dots, n_0)$ 为元素的

$k_0 \times (n_0 - k_0)$ 阶矩阵, 可知与系统分组码的 \mathbf{G} 有相同的形式。因此, 在系统码情况, $k_0 \times k_0$ 个子生成元是已知的, 即:

$$\begin{aligned}\mathbf{g}^{(1, 1)}(D) &= 1, \mathbf{g}^{(1, 2)}(D) = \mathbf{g}^{(1, 3)}(D) = \cdots = \mathbf{g}^{(1, k_0)}(D) = 0 \\ \mathbf{g}^{(2, 1)}(D) &= 0, \mathbf{g}^{(2, 2)}(D) = 1, \mathbf{g}^{(2, 3)}(D) = \cdots = \mathbf{g}^{(2, k_0)}(D) = 0 \\ &\vdots \\ \mathbf{g}^{(k_0, 1)}(D) &= \mathbf{g}^{(k_0, 2)}(D) = \cdots = \mathbf{g}^{(k_0, k_0-1)}(D) = 0, \mathbf{g}^{(k_0, k_0)}(D) = 1\end{aligned}$$

只有 $k_0 \times (n_0 - k_0)$ 个子生成元需要确定, 一旦这些子生成元确定了, 则码的 \mathbf{G} 以及编码电路也就确定了。

二、卷积码的一致校验矩阵及编码电路

得到了卷积码的生成矩阵 \mathbf{G}_∞ 后, 就不难得出它的校验矩阵 \mathbf{H}_∞ 。卷积码是线性码, 因而也像分组码一样, 生成矩阵 \mathbf{G}_∞ 和校验矩阵 \mathbf{H}_∞ 之间必须满足:

$$\mathbf{G}_\infty \cdot \mathbf{H}_\infty^T = \mathbf{0}$$

即 (n_0, k_0, m) 卷积码的每一个码字, 必须满足由 \mathbf{H}_∞ 矩阵的行所确定的线性方程组, 或者说都在 \mathbf{H}_∞ 矩阵的零空间中。 (n_0, k_0, m) 卷积码校验矩阵的一般形式为

$$\mathbf{H}_\infty = \begin{bmatrix} \mathbf{h}_0 & & & & \\ \mathbf{h}_1 & \mathbf{h}_0 & & & \\ \mathbf{h}_2 & \mathbf{h}_1 & \mathbf{h}_0 & & \\ \vdots & \vdots & & & \\ \mathbf{h}_m & \mathbf{h}_{m-1} & \cdots & \mathbf{h}_0 & \cdots \\ \mathbf{0} & \mathbf{h}_m & \cdots & \mathbf{h}_1 & \cdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \cdots \\ \vdots & \vdots & & \vdots & \end{bmatrix} \quad (10.2.18)$$

式中, $\mathbf{h}_0, \mathbf{h}_1, \dots, \mathbf{h}_m$ 及 $\mathbf{0}$ 都是 $(n_0 - k_0) \times n_0$ 阶矩阵, 其余空白地方全为 0。显然, \mathbf{H}_∞ 和 \mathbf{G}_∞ 一样, 也是一个半无限矩阵。由 $\mathbf{G}_\infty \cdot \mathbf{H}_\infty^T = \mathbf{0}$, 我们就不难求出 \mathbf{H}_∞ 中的每个元素。

如以前例举的 $(3, 2, 2)$ 卷积码, 由式(10.2.9)可知它的生成矩阵

$$\mathbf{G}_\infty = \begin{bmatrix} \mathbf{g}_0 & \mathbf{g}_1 & \mathbf{g}_2 & & \\ & \mathbf{g}_0 & \mathbf{g}_1 & \mathbf{g}_2 & \\ & & \mathbf{g}_0 & \mathbf{g}_1 & \mathbf{g}_2 \\ & & & \cdots & \end{bmatrix} = \begin{bmatrix} 101 & 000 & 001 \\ 011 & 001 & 000 \\ & & 101 & 000 & 001 \\ & & 011 & 001 & 000 \\ & & & 101 & 000 & 001 \\ & & & 011 & 001 & 000 \\ & & & & \cdots & \\ & & & & & \cdots \end{bmatrix}$$

由

$$\mathbf{G}_\infty \cdot \mathbf{H}_\infty^T = \mathbf{0}$$

$$\left[\begin{array}{ccc} 101 & 000 & 001 \\ 011 & 001 & 000 \\ & 101 & 000 & 001 \\ & 011 & 001 & 000 \\ & 101 & 000 & 001 \\ & 011 & 001 & 000 \\ & \cdots & & \\ & \cdots & & \end{array} \right] \left[\begin{array}{ccc} \mathbf{h}_0^T & \mathbf{h}_1^T & \mathbf{h}_2^T \\ \mathbf{h}_0^T & \mathbf{h}_1^T & \mathbf{h}_2^T \\ \mathbf{h}_0^T & \mathbf{h}_1^T & \mathbf{h}_2^T \\ \cdots & & \cdots \end{array} \right] = \mathbf{0}$$

可得到

$$\mathbf{h}_0 = [111], \mathbf{h}_1 = [010], \mathbf{h}_2 = [100]$$

因而该码的校验矩阵

$$\mathbf{H}_{\infty} = \left[\begin{array}{ccc} \mathbf{h}_0 & & \\ \mathbf{h}_1 & \mathbf{h}_0 & \\ \mathbf{h}_2 & \mathbf{h}_1 & \mathbf{h}_0 \\ 0 & \mathbf{h}_2 & \mathbf{h}_1 \\ \vdots & 0 & \mathbf{h}_2 \\ \vdots & 0 & \cdots \end{array} \right] = \left[\begin{array}{cccc} 111 & & & \\ 010 & 111 & & \\ 100 & 010 & 111 & \\ 000 & 100 & 010 & \cdots \\ 000 & 100 & \cdots & \\ 000 & \cdots & & \cdots \end{array} \right]$$

显然, \mathbf{H}_{∞} 矩阵完全由它的第一列元素 $[\mathbf{h}_0, \mathbf{h}_1, \mathbf{h}_2, \dots]^T$ 所决定, 若令第一列元素为 $\mathbf{B}_0^T = [\mathbf{h}_0, \mathbf{h}_1, \mathbf{h}_2, \dots]$, 则 \mathbf{H}_{∞} 矩阵有如下形式:

$$\mathbf{H}_{\infty} = \left[\begin{array}{c|c|c|c|c} \mathbf{B}_0 & & & & n_0 - k_0 \\ \hline & \mathbf{B}_0 & & & \\ \hline & & \mathbf{B}_0 & & \\ \hline & & & \mathbf{B}_0 & \\ \hline n_0 & & & & \ddots \end{array} \right]$$

(n_0, k_0, m) 卷积码若为系统码, 则由系统码的 \mathbf{G}_{∞} 一般形式的式(10.2.16), 根据 $\mathbf{G}_{\infty} \cdot \mathbf{H}_{\infty}^T = \mathbf{0}$, 可得到系统码的一致校验矩阵为

$$\mathbf{H}_{\infty} = \left[\begin{array}{cccccc} -\mathbf{p}_0^T \mathbf{I}_{n_0 - k_0} & & & & & \\ -\mathbf{p}_1^T \mathbf{0} & -\mathbf{p}_0^T \mathbf{I}_{n_0 - k_0} & & & & \\ \vdots & \vdots & & & & \\ -\mathbf{p}_m^T \mathbf{0} & -\mathbf{p}_{m-1}^T \mathbf{0} & \cdots & -\mathbf{p}_0^T \mathbf{I}_{n_0 - k_0} & \cdots & \\ -\mathbf{p}_m^T \mathbf{0} & & & \vdots & & \\ -\mathbf{p}_m^T \mathbf{0} & & & & & \end{array} \right] \quad (10.2.19)$$

我们定义该矩阵的第 $m+1$ 行元素所组成的矩阵

$$\begin{aligned} \mathbf{h}_{\infty} &= [-\mathbf{p}_m^T, -\mathbf{p}_{m-1}^T, \dots, -\mathbf{p}_0^T \mathbf{I}_{n_0 - k_0}, \mathbf{0}, \dots] \\ &= [\mathbf{h}_m, \mathbf{h}_{m-1}, \dots, \mathbf{h}_0, \mathbf{0}, \dots] \end{aligned} \quad (10.2.20)$$

为码的基本校验矩阵。上面两式中 $\mathbf{p}_0^T, \mathbf{p}_1^T, \dots, \mathbf{p}_m^T$ 都是一个 $(n_0 - k_0) \times k_0$ 阶矩阵。它们分别是式(10.2.16) \mathbf{G}_{∞} 矩阵中 $\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_m$ 矩阵的转置矩阵。 $\mathbf{I}_{n_0 - k_0}$ 是一个 $(n_0 - k_0) \times (n_0 - k_0)$

阶单位矩阵， $\mathbf{0}$ 是一个 $(n_0 - k_0) \times (n_0 - k_0)$ 阶全0矩阵，其余空白地方全为0。在上面所举例中， $\mathbf{h}_\infty = [100, 010, 111, \dots]$ 。由以上讨论可知，只要码的 \mathbf{h}_∞ 矩阵确定，码的 \mathbf{H}_∞ 和 \mathbf{G}_∞ 矩阵也就完全确定了。因此，与 \mathbf{g}_∞ 一样， \mathbf{h}_∞ 矩阵在卷积码中也是一个很重要的决定码的矩阵。

类似于码的生成多项式矩阵 $\mathbf{G}(D)$ ，也可以定义码的校验多项式矩阵 $\mathbf{H}(D)$ 为

$$\mathbf{H}(D) = \begin{bmatrix} \mathbf{h}^{(1, 1)}(D) & \mathbf{h}^{(1, 2)}(D) & \cdots & \mathbf{h}^{(1, n_0)}(D) \\ \vdots & \vdots & & \vdots \\ \mathbf{h}^{(n_0 - k_0, 1)}(D) & \mathbf{h}^{(n_0 - k_0, 2)}(D) & \cdots & \mathbf{h}^{(n_0 - k_0, n_0)}(D) \end{bmatrix} \quad (10.2.21)$$

式中， $\mathbf{h}^{(i, j)}(D)$ ($i = 1, 2, \dots, n_0 - k_0$; $j = 1, 2, \dots, n_0$) 为码的子校验元，通常，它也是一个次数小于 m 的多项式。由 $\mathbf{G}(D) \cdot \mathbf{H}(D)^T = \mathbf{0}$ ，我们就不难求得每个子校验元。仍以(3, 2, 2)码为例，由式(10.2.5)知，它的

$$\mathbf{G}(D) = \begin{bmatrix} 1 & 0 & 1 + D^2 \\ 0 & 1 & 1 + D \end{bmatrix} \quad (10.2.22)$$

由此可得码的

$$\mathbf{H}(D) = [1 + D^2, 1 + D, 1] \quad (10.2.23)$$

下面讨论 $\mathbf{G}(D)$ 和 $\mathbf{H}(D)$ 的物理意义，以式(10.2.22)与式(10.2.23)为例说明。由该两式可知 $\mathbf{G}(D)$ 有 3 ($n_0 = 3$)列 2 ($k_0 = 2$)行，它的元素说明卷积码码字中的每一段子码的 n_0 个码元与 k_0 个信息元的关系。如式(10.2.22)中， $\mathbf{G}(D)$ 的第一列是 $[1, 0]^T$ ，说明每个子码的第一个码元是此时直接输入的第一个信息元，而与第二个信息元无关。第二列是 $[0, 1]^T$ ，说明子码的第二个码元是此时刻输入的第二个信息元，而与第一个信息元无关。第三列是 $[1 + D^2, 1 + D]^T$ ，说明子码的第三个码元是：先求出此(0)时刻输入的第一个信息元与前两个时间单位输入的第一个信息元的模2和；再求出此时刻输入的第二个信息元与前一个时间单位输入的模2和；然后，把这两个模2和的结果再进行模2加而决定的。这与该码的基本生成矩阵

$$\mathbf{g}_\infty = \begin{pmatrix} 101 & 000 & 001 & 000 & \cdots \\ 011 & 001 & 000 & 000 & \cdots \end{pmatrix}$$

所表示的物理意义完全一致。可知这个 $\mathbf{G}(D)$ 所确定码字的每一个子码的前二位是输入的不变信息元，后面一位是校验元，因此是系统码。这也说明了为什么式(10.2.17)的 $\mathbf{G}(D)$ 矩阵的左边是一个 $k_0 \times k_0$ 阶单位方阵。

同样，式(10.2.23)的 $\mathbf{H}(D)$ 矩阵中共有 3 ($n_0 = 3$)列， 1 ($n_0 - k_0 = 1$)行，它说明子码中的一个(一般情况下有 $n_0 - k_0$ 个)校验元与前两个(共 k_0 个)信息元的关系。第一列 $1 + D^2$ 说明此时刻的第一个信息元与前两个时间单位的第一个信息元，参加了求此时刻校验元的运算。而第二例的 $1 + D$ ，说明此时刻的第二个信息元与前一个时间单位的第二个信息元参加了求此时刻校验元的运算。因此，式(10.2.23)的 $\mathbf{H}(D)$ 所表示的物理意义与码的基本校验矩阵

$$\mathbf{H}_\infty = [100, 010, 111, 000, \dots]$$

所表示的物理意义完全相同。

由上面的讨论，我们不难得出系统码的校验矩阵多项式 $\mathbf{H}(D)$ 与式(10.2.17)的 $\mathbf{G}(D)$ 之间的关系，有如同系统分组码 \mathbf{H} 与 \mathbf{G} 之间的关系：

$$\begin{aligned}
H(D) &= \begin{bmatrix} h^{(1,1)}(D) & h^{(1,2)}(D) & \cdots & h^{(1,k_0)}(D) & h^{(1,k_0+1)}(D) \\ \vdots & \vdots & & \vdots & \vdots \\ h^{(n_0-k_0,1)}(D) & h^{(n_0-k_0,2)}(D) & \cdots & h^{(n_0-k_0,k_0)}(D) & h^{(n_0-k_0,k_0+1)}(D) \\ \cdots & h^{(1,n_0)}(D) & & \vdots & \\ \cdots & h^{(n_0-k_0,n_0)}(D) & & & \end{bmatrix} \\
&= \begin{bmatrix} -g^{(1,k_0+1)}(D) & -g^{(2,k_0+1)}(D) & \cdots & -g^{(k_0,k_0+1)}(D) & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ -g^{(1,n_0)}(D) & -g^{(2,n_0)}(D) & \cdots & -g^{(k_0,n_0)}(D) & 0 & 0 & \cdots & 1 \end{bmatrix} \\
&= [-p^T(D) I_{n_0-k_0}]
\end{aligned} \tag{10.2.24}$$

即 $H(D)$ 的最左边 k_0 列元素是式(10.2.17) $G(D)$ 阵中最右边 $P(D)$ 阵的转置, 而最右边是一个 $(n_0 - k_0) \times (n_0 - k_0)$ 阶单位方阵, 因而 $H(D)$ 的每一行说明子码中的每一个校验元是如何得到的, 它也就是一个校验元的线性方程组。

了解 $G(D)$ 和 $H(D)$ 的物理意义, 可以很快地构造出 (n_0, k_0, m) 卷积码编码器。

(n_0, k_0, m) 卷积码编码器分两类: 一类是有 $m(n_0 - k_0)$ 级的串行编码器; 另一类是 mk_0 级的串行和并行编码器。图 10-4 就是 $(3, 2, 2)$ 码的 $m(n_0 - k_0)$ 级编码器, 它有 $m(n_0 - k_0) = 2$ 级迟延单元(移位寄存器), 每一子码的两个信息元直接由两个并行的输入线得到, 而第三个码元, 即校验元则由 $G(D)$ 的第三列所确定的校验关系得到(即由子生成元 $g^{(1,3)}(D) = 1 + D^2$, $g^{(2,3)}(D) = 1 + D$ 确定)。而 $(3, 2, 2)$ 码的 mk_0 级的并行编码器和串行编码器分别如图 10-5 和图 10-6 所示。把此二图与图 10-4 比较可知, 以图 10-4 的 $m(n_0 - k_0)$ 级编码器较简单, 图 10-5 的并行输入的 mk_0 级编码器次之, 而以图 10-6 的串行输入的 mk_0 级编码器最复杂, 且图 10-6 编码器较前两种编码器要多一个单位时间的延迟。这三种类型的编码器均有用, 在实际中, 究竟应采用何种类型电路, 要看实际情况而定。

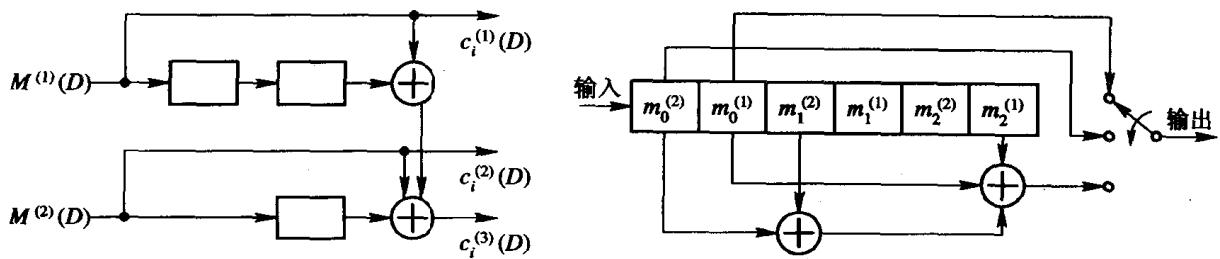


图 10-5 $(3, 2, 2)$ 码的 mk_0 级并行编码器

图 10-6 $(3, 2, 2)$ 码的 mk_0 级串行编码器

例 10.1 求 $(2, 1, 2)$ 码的编码器, 该码的两个子生成元: $g^{(1,1)}(D) = 1 + D + D^2$, $g^{(1,2)}(D) = 1 + D^2$ 。由此可知该码的 $G(D)$ 是

$$G(D) = [1 + D + D^2, 1 + D^2]$$

这是一个非系统码。它的子码中的第一个码元由 $g^{(1,1)}(D)$ 决定, 即由此时刻输入的信息元与前一个和二个单位时间输入信息元的模 2 和, 第二个码元由 $g^{(1,2)}(D)$ 决定, 是此时刻输入的信息元与前二个单位时间输入的信息元的模 2 和。由此可画出如图 10-7 的编码

电路。

非系统码的 $H(D)$ 中的元素 $h^{(i,j)}(D)$, 与 $G(D)$ 中的元素之间, 不存在像系统码那样相互之间有简单关系, 而必须通过 $G(D) \cdot H^T(D) = \mathbf{0}$, 才能得出 $H(D)$ 。

通过上面这些例子, 我们可以看出, 在 $G(D)$ 、 $H(D)$ 以及子生成元 $g^{(i,j)}(D)$ 之间, 存在着密切关系。只要知道了其中之一, 其它两个都可以很快得到。但以子生成元 $g^{(i,j)}(D)$ 最为有用, 知道了它, 就很容易画出编码电路和求出生成矩阵 G_∞ , 并决定出编码存储 $m = \max \partial \circ g^{(i,j)}(D)$ ($i=1, 2, \dots, k_0; j=1, 2, \dots, n_0$)。

三、初始截段(短)码

卷积码的 G_∞ 虽然都是半无限矩阵, 但在任何 $(m+1)$ 段的编码约束度内, 它们所表示的码元之间的约束关系完全相同。此外, 在卷积码的一般代数译码中, 通常也只考虑一个编码约束长度 $N_A = (m+1)n_0$ 内的码序列。不失一般性, 我们只考虑编码器初始状态全为 0 时, 卷积编码器输出码序列的首 N_A 个码元之间的约束关系即可。

定义 10.2.1 编码器初始状态全为 0 时, 编码器输出码序列的首 $m+1$ 段子码所组成的码字, 称为卷积码的初始截段码字, 以

$$C_{00} = c_0 + c_1 D + c_2 D^2 + \dots + c_m D^m$$

表示, 式中 $c_i = (c_{i1}, c_{i2}, \dots, c_{in_0})$ 。而以

$$M_{00} = m_0 + m_1 D + \dots + m_m D^m$$

表示在同一时间段内输入至编码器的信息序列。其中 $m_i = (m_{i1}, m_{i2}, \dots, m_{ik_0})$ 。

在系统码条件下

$$c_{ij} = m_{ij} \quad i = 0, 1, \dots, m; j = 1, 2, \dots, k_0$$

而每段中的后 $n_0 - k_0$ 个校验元 c_{ij} ($j=k_0+1, \dots, n_0; i=0, 1, \dots, m$), 则由信息序列与子生成元的卷积运算得到。

由以上定义可知, 所有初始截段码字的集合, 也就是与输入信息序列 M_{00} 的所有可能取值所对应的码字集合, 构成一个 $((m+1)n_0, (m+1)k_0)$ 线性码, 我们称它为 (n_0, k_0, m) 卷积码的初始截段码(以后简称截段码)。由此可知, 由式(10.2.16)系统码的初始截段码的生成矩阵 G 为

$$G = \begin{bmatrix} I_{k_0} p_0 & 0p_1 & 0p_2 & \cdots & 0p_m \\ & I_{k_0} p_0 & 0p_1 & \cdots & 0p_{m-1} \\ & & \ddots & & \vdots \\ & & & & I_{k_0} p_0 \end{bmatrix} \quad (10.2.25)$$

相应的一致校验矩阵 H 为

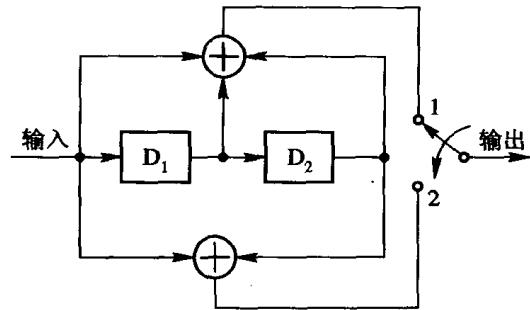


图 10-7 (2, 1, 2) 非系统卷积码编码器

$$\mathbf{H} = \begin{bmatrix} -\mathbf{p}_0^T \mathbf{I}_{n_0-k_0} \\ -\mathbf{p}_1^T \mathbf{0} & -\mathbf{p}_0^T \mathbf{I}_{n_0-k_0} \\ \vdots & \vdots & \ddots \\ -\mathbf{p}_m^T \mathbf{0} & -\mathbf{p}_{m-1}^T \mathbf{0} & -\mathbf{p}_0^T \mathbf{I}_{n_0-k_0} \end{bmatrix} \quad (10.2.26)$$

若前 n_0 列或最后 $n_0 - k_0$ 行确定了，则 \mathbf{H} 也就确定了。而它的后 $n_0 - k_0$ 行就是式(10.2.20)基本校验矩阵前 $m+1$ 段

$$\mathbf{h} = [-\mathbf{p}_m^T \mathbf{0}, -\mathbf{p}_{m-1}^T \mathbf{0}, \dots, -\mathbf{p}_0^T \mathbf{I}_{n_0-k_0}] \quad (10.2.27)$$

我们称它为初始截段码的基本校验矩阵，今后若无特别声明，就称它为码的基本校验矩阵。同理可知，初始截段码的基本生成矩阵为

$$\mathbf{g} = [\mathbf{I}_{k_0} \mathbf{p}_0, \mathbf{0} \mathbf{p}_1, \mathbf{0} \mathbf{p}_2, \dots, \mathbf{0} \mathbf{p}_m] \quad (10.2.28)$$

今后一般情况下我们用 \mathbf{g} 代替 \mathbf{g}_∞ ，称它为码的基本生成矩阵。

式(10.2.26)的 \mathbf{H} 矩阵，可用它的前 n_0 列组成的 \mathbf{B}_0 矩阵

$$\mathbf{B}_0 = \begin{bmatrix} -\mathbf{p}_0^T \mathbf{I}_{n_0-k_0} \\ -\mathbf{p}_1^T \mathbf{0} \\ -\mathbf{p}_m^T \mathbf{0} \end{bmatrix} \quad (10.2.29)$$

简化表示。比较 \mathbf{B}_0 和 \mathbf{H} 矩阵不难看出，以 \mathbf{B}_0 矩阵每次下降 $(n_0 - k_0)$ 行并取掉最后 $(n_0 - k_0)$ 行，就自左至右地给出了 \mathbf{H} 的每 n_0 列，故 \mathbf{H} 可简写成

$$\mathbf{H} = [\mathbf{B}_0, \mathbf{B}_1, \dots, \mathbf{B}_m]$$

式中， \mathbf{B}_i 就是 \mathbf{B}_0 阵下降 $i(n_0 - k_0)$ 行，并取掉最后 $i(n_0 - k_0)$ 行得到的矩阵。

由式(10.2.25)的矩阵看到，卷积码的 $((m+1)n_0, (m+1)k_0)$ 截段码与 $[n = (m+1)n_0, k = (m+1)k_0]$ 线性分组码的差别，仅在于截段码的信息位是不连在一起的，而是间隔地分布在每一段子码内。因此，若把截段码作为线性码看待，则有关线性分组码中的结果对它也同样成立。

由于初始截段码的 \mathbf{G} 和 \mathbf{H} 完全确定了卷积码的校验关系，因此今后在通常代数译码中，就用 \mathbf{G} 和 \mathbf{H} 来代替 \mathbf{G}_∞ 和 \mathbf{H}_∞ 。

四、对偶码与透明码

由 $\mathbf{G}_\infty \cdot \mathbf{H}_\infty^T = \mathbf{0}$ 可知， (n_0, k_0, m) 码的 \mathbf{G}_∞ 矩阵生成的空间和由它的 \mathbf{H}_∞ 矩阵生成的空间互为零空间。因此，若以 $\mathbf{C} = (n_0, k_0, m)$ 码的 \mathbf{H}_∞ 矩阵作为 \mathbf{C}_\perp 码的 \mathbf{G}_∞ 矩阵，而以 \mathbf{C} 码的 \mathbf{G}_∞ 作为 \mathbf{C}_\perp 码的 \mathbf{H}_∞ 矩阵，则 $\mathbf{C}_\perp = (n_0, n_0 - k_0, m)$ 码与 \mathbf{C} 码互为对偶码。

如(3, 2, 2)码，它的 \mathbf{H} 和 \mathbf{G} 矩阵分别为：

$$\mathbf{H} = \begin{bmatrix} 111 \\ 010 & 111 \\ 100 & 010 & 111 \end{bmatrix} \quad \mathbf{G} = \begin{bmatrix} 101 & 000 & 001 \\ 011 & 001 & 000 \\ & 101 & 000 \\ & 011 & 001 \\ & & 101 \\ & & 011 \end{bmatrix}$$

因而它的对偶码(3, 1, 2)码的 \mathbf{H}_\perp 和 \mathbf{G}_\perp 为:

$$\mathbf{H}_\perp = \begin{bmatrix} 110 \\ 101 \\ 000 & 110 \\ 100 & 101 \\ 100 & 000 & 110 \\ 000 & 100 & 101 \end{bmatrix} \quad \mathbf{G}_\perp = \begin{bmatrix} 111 & 001 & 010 \\ & 111 & 001 \\ & & 111 \end{bmatrix}$$

相应的编码器如图 10-8 所示。

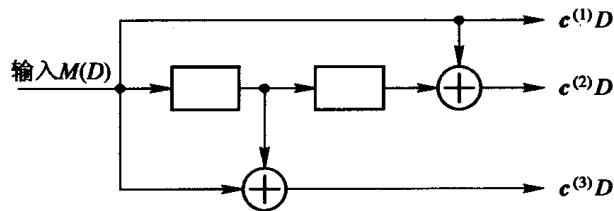


图 10-8 (3, 1, 2) 码编码器

在系统码情况下, (n_0, k_0, m) 码的

$$\mathbf{H} = \begin{bmatrix} -\mathbf{p}_0^T \mathbf{I}_{n_0-k_0} \\ -\mathbf{p}_1^T \mathbf{0} & -\mathbf{p}_0^T \mathbf{I}_{n_0-k_0} \\ \vdots & \vdots & \ddots \\ -\mathbf{p}_m^T \mathbf{0} & -\mathbf{p}_{m-1}^T \mathbf{0} & \cdots & -\mathbf{p}_0^T \mathbf{I}_{n_0-k_0} \end{bmatrix} \quad (10.2.30)$$

$$\mathbf{G} = \begin{bmatrix} \mathbf{I}_{k_0} \mathbf{p}_0 & \mathbf{0} \mathbf{p}_1 & \cdots & \mathbf{0} \mathbf{p}_m \\ & \mathbf{I}_{k_0} \mathbf{p}_0 & \cdots & \mathbf{0} \mathbf{p}_{m-1} \\ \ddots & \vdots & & \\ & & & \mathbf{I}_{k_0} \mathbf{p}_0 \end{bmatrix} \quad (10.2.31)$$

相应的 (n_0, n_0-k_0, m) 对偶码的

$$\mathbf{H}_\perp = \begin{bmatrix} \mathbf{p}_0 \mathbf{I}_{k_0} \\ \mathbf{p}_1 \mathbf{0} & \mathbf{p}_0 \mathbf{I}_{k_0} \\ \vdots & \vdots & \ddots \\ \mathbf{p}_m \mathbf{0} & \mathbf{p}_{m-1} \mathbf{0} & \cdots & \mathbf{p}_0 \mathbf{I}_{k_0} \end{bmatrix} \quad (10.2.32)$$

$$\mathbf{G}_\perp = \begin{bmatrix} \mathbf{I}_{n_0-k_0} (-\mathbf{p}_0^T) & \mathbf{0} (-\mathbf{p}_1^T) & \cdots & \mathbf{0} (-\mathbf{p}_m^T) \\ & \mathbf{I}_{n_0-k_0} (-\mathbf{p}_0^T) & \cdots & \mathbf{0} (-\mathbf{p}_{m-1}^T) \\ \ddots & & & \vdots \\ & & & \mathbf{I}_{n_0-k_0} (-\mathbf{p}_0^T) \end{bmatrix} \quad (10.2.33)$$

在非系统码情况下, 对偶码的 \mathbf{G}_\perp 和 \mathbf{H}_\perp 之间没有如此的简单关系, 而必须根据 $\mathbf{G}_\perp \cdot \mathbf{H}_\perp^T = \mathbf{0}$ 求出 \mathbf{H}_\perp 或 \mathbf{G}_\perp 。

如例 10.1 的(2, 1, 2)非系统码, 它的

$$G = \begin{bmatrix} 11 & 10 & 11 \\ & 11 & 10 \\ & & 11 \end{bmatrix} \quad G(D) = [1 + D + D^2, 1 + D^2]$$

首先我们由 $G(D) \cdot H(D)^T = \mathbf{0}$, 求出 $H(D)$ 。设 $H(D)$ 为

$$H(D) = [h^{(1, 1)}(D), h^{(1, 2)}(D)]$$

式中, $h^{(1, 1)}(D) = 1 + aD + bD^2$, $h^{(1, 2)}(D) = 1 + cD + dD^2$, 代入 $G(D) \cdot H(D)^T = \mathbf{0}$, 得:

$$(1 + D + D^2)(1 + aD + bD^2) + (1 + D^2)(1 + cD + dD^2) = 0$$

$$(a + c + 1)D + (a + b + d)D^2 + (a + b + c)D^3 + (b + d)D^4 = 0$$

所以各项系数必为 0, 故有:

$$a + c + 1 = 0$$

$$a - b + d = 0$$

$$a + b + c = 0$$

$$b + d = 0$$

可解得 $a=0$, $b=c=d=1$, 因而码的校验多项式矩阵

$$H(D) = [1 + D^2, 1 + D + D^2]$$

由此可知, 该非系统码的对偶码(2, 1, 2)码

的生成矩阵:

$$G_{\perp} = \begin{bmatrix} 11 & 01 & 11 \\ & 11 & 01 \\ & & 11 \end{bmatrix}$$

$$G_{\perp}(D) = [1 + D^2, 1 + D + D^2]$$

$$H_{\perp}(D) = [1 + D + D^2, 1 + D^2]$$

该码的编码器如图 10-9 所示。

如果 (n_0, k_0, m) 卷积码的每个子生成元中有奇数项, 则称为**透明码**。透明码的每个码字的补码也是该码的一个码字。

例如(2, 1, 2)码的生成矩阵

$$G(D) = [1, 1 + D + D^2]$$

它的两个生成元有奇数项。若 $M(D) = [1 + D^2 + D^4 + D^6 + \dots]$, 则码序列

$$\begin{aligned} C(D) &= M(D)G(D) = [1 + D^2 + D^4 + \dots][1, 1 + D + D^2] \\ &= (1, 1) + (0, 1)D + (1, 0)D^2 + (0, 1)D^3 + (1, 0)D^4 + (0, 1)D^5 + \dots \end{aligned}$$

它的补码(D^2 以后的码序列)

$$\begin{aligned} \bar{C}(D) &= (0, 0) + (1, 0)D + (0, 1)D^2 + (1, 0)D^3 + (0, 1)D^4 + (1, 0)D^5 + \dots \\ &= \bar{M}(D)G(D) = (D + D^3 + D^5 + \dots)G(D) \end{aligned}$$

也是码的一个码字, 因此该码是透明码。透明码的这个特点在维特比译码中为解决码元相位模糊问题非常有用。

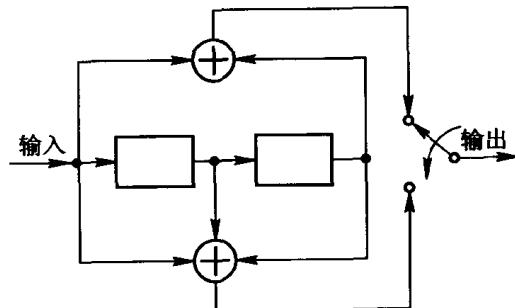


图 10-9 (2, 1, 2)非系统码编码器

§ 10.3 伴随式计算与一般译码

卷积码各码段之间前后均有约束关系，如何充分利用各码段之间的约束关系进行译码，以及由此带来的一些特殊问题，就成为与分组码译码的最主要不同点之一。卷积码的译码可分为代数译码与概率译码两类。这一节我们主要介绍卷积码代数译码的基本概念，为此首先介绍伴随式的定义和实现电路。

一、伴随式计算

与分组码所定义的伴随式相同，设发送的半无限长码序列为 C_∞ ，信道的半无限长错误图样为 E_∞ ，则接收到的半无限长序列 $R_\infty = C_\infty + E_\infty$ 。定义以下半无限长序列：

$$\begin{aligned} S_\infty &= R_\infty \cdot H_\infty^T = (C_\infty + E_\infty) \cdot H_\infty^T = E_\infty \cdot H_\infty^T \\ &= [s_{0,1} \cdots s_{0,n_0-k_0}, s_{1,1} \cdots s_{1,n_0-k_0}, \dots, s_{m,1} \cdots s_{m,n_0-k_0}, \dots] \end{aligned} \quad (10.3.1)$$

为 R_∞ 的伴随式。式中， H_∞^T 是 (n_0, k_0, m) 卷积码的校验矩阵 H_∞ 的转置矩阵。

若错误图样序列 E_∞ 全为 0，则 S_∞ 全为 0，说明 R_∞ 就是 C_∞ ；否则， S_∞ 不全为 0。若我们仅检验接收序列中连续 $m+1$ 段以内的校验关系，则 (n_0, k_0, m) 卷积码的伴随式定义为

$$\begin{aligned} S &= R \cdot H^T = (C + E) \cdot H^T = E \cdot H^T \\ &= (e_{0,1} \cdots e_{0,n_0}, e_{1,1} \cdots e_{1,n_0}, \dots, e_{m,1} \cdots e_{m,n_0}) H^T \\ &= (s_{0,1} \cdots s_{0,n_0-k_0}, s_{1,1} \cdots s_{1,n_0-k_0}, \dots, s_{m,1} \cdots s_{m,n_0-k_0}) \end{aligned} \quad (10.3.2)$$

相应的伴随式多项式定义为

$$\begin{aligned} S(D) &= R(D) + H^T(D) = E(D) \cdot H^T(D) \\ &= (s_{0,1}, \dots, s_{0,n_0-k_0}) + (s_{1,1}, \dots, s_{1,n_0-k_0})D \\ &\quad + \cdots + (s_{m,1}, \dots, s_{m,n_0-k_0})D^m \\ &= S_1(D) + S_2(D) + \cdots + S_{n_0-k_0}(D) \end{aligned} \quad (10.3.3)$$

式中：

$$\begin{aligned} S_1(D) &= s_{0,1} + s_{1,1}D + \cdots + s_{m,1}D^m \\ S_2(D) &= s_{0,2} + s_{1,2}D + \cdots + s_{m,2}D^m \\ &\vdots \\ S_{n_0-k_0}(D) &= s_{0,n_0-k_0} + s_{1,n_0-k_0}D + \cdots + s_{m,n_0-k_0}D^m \end{aligned} \quad (10.3.4)$$

称 $S_i(D)$ 为子伴随式多项式序列。由此可知，伴随式是一个有 $(m+1)(n_0-k_0)$ 个分量的序列，它反映了编码约束长度内有关错误图样的信息。

因此，若译码约束长度等于编码约束长度，则我们仅只要讨论 S 就足够了。若 S 全为 0，说明此时间单位收到子组（第 0 子组）前的第 m 子组内无错，但不能说明第 $m-1$ 子组至第 0 子组内有无错误，因为第 $m-1$ 至第 0 子组还与以后各子组有约束关系。因此，在 t_0 时刻求得的 S ，只能用于纠正其前 m 个时间单位的第 m 子组的错误。因而，卷积码的伴随式计算和译码是以子组或码段为单位进行的，每一个时间单位，仅完成一个子组的译码。

下面讨论伴随式计算电路的实现，以例说明之。

例 10.2 $(2, 1, m)$ 系统卷积码，它的

$$G(D) = [1, g(D)]$$

$$H(D) = [g(D), 1]$$

$$R(D) = C(D) + E(D)$$

$$\begin{aligned} &= (c_{01}c_{02}) + (c_{11}c_{12})D + \cdots + (c_{m1}c_{m2})D^m \\ &\quad + (e_{01}e_{02}) + (e_{11}e_{12})D + \cdots + (e_{m1}e_{m2})D^m \\ &= [(c_{01} + e_{01}) + (c_{11} + e_{11})D + \cdots + (c_{m1} + e_{m1})D^m], \\ &\quad [(c_{02} + e_{02}) + (c_{12} + e_{12})D + \cdots + (c_{m2} + e_{m2})D^m] \\ &= [(C^{(1)}(D) + e^{(1)}(D)), (C^{(2)}(D) + e^{(2)}(D))] = [R^{(1)}(D), R^{(2)}(D)] \end{aligned}$$

式中：

$$R_1(D) = [C^{(1)}(D) + e^{(1)}(D)] = [(c_{01} + e_{01}) + (c_{11} + e_{11})D + \cdots + (c_{m1} + e_{m1})D^m]$$

$$R_2(D) = [C^{(2)}(D) + e^{(2)}(D)] = [(c_{02} + e_{02}) + (c_{12} + e_{12})D + \cdots + (c_{m2} + e_{m2})D^m]$$

$$E(D) = [e^{(1)}(D), e^{(2)}(D)] = [e_{01} + e_{11}D + \cdots + e_{m1}D^m, e_{02} + e_{12}D + \cdots + e_{m2}D^m]$$

所以伴随式多项式

$$\begin{aligned} S(D) &= R(D) \cdot H(D)^T \\ &= [R^{(1)}(D), R^{(2)}(D)] \begin{bmatrix} g(D) \\ 1 \end{bmatrix} = [R^{(1)}(D)g(D) + R^{(2)}(D)] \\ &= (C^{(1)}(D) + e^{(1)}(D))g(D) + (C^{(2)}(D) + e^{(2)}(D)) \\ &= (C^{(1)}(D)g(D) + C^{(2)}(D)) + (e^{(1)}(D)g(D) + e^{(2)}(D)) \\ &= e^{(1)}(D)g(D) + e^{(2)}(D) \\ &= S^{(1)}(D) = s_{01} + s_{11}D + \cdots + s_{m1}D^m \end{aligned}$$

由此可知， $(2, 1, m)$ 码的伴随式是收到码序列的第一个码元序列 $[C^{(1)}(D) + e^{(1)}(D)]$ （信息元序列）通过 $g(D)$ 电路后，与第二个码元序列 $[C^{(2)}(D) + e^{(2)}(D)]$ （校验元序列）相加得到。该码的伴随式计算电路如图 10-10 所示。由该图可知，伴随式计算电路的实质，就是把收到信息元序列 $[C^{(1)}(D) + e^{(1)}(D)]$ 再通过与发送端相同的编码器，得到一个新的校验元，把它与收到的校验元进行比较（模 2 加）。若相同，则得到 S 的一分量为 0；否则为 1。

上面我们通过 $(2, 1, m)$ 系统卷积码，说明伴随式计算电路的结构，其实对任何系统卷积码伴随式的计算，都可用同样的方法实现。图 10-11 就给出了 (n_0, k_0, m) 系统卷积码的伴随式计算电路。

二、代数译码的基本概念

卷积码的译码问题就是从收到的序列 $R(D) = C(D) + E(D)$ 中，确定出发送的码序列 $C(D)$ 或相应的信息序列 $M(D)$ ，这相当于要求译码器首先计算伴随式 $S(D)$ ，然后由 $S(D)$ 确定错误图样 $\hat{E}(D)$ ，最后 $\hat{C}(D) = R(D) - \hat{E}(D)$ 得到了已纠正过的序列 $\hat{C}(D)$ ，它可能与

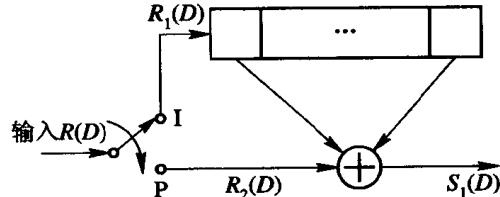


图 10-10 $(2, 1, m)$ 卷积码伴随式计算电路

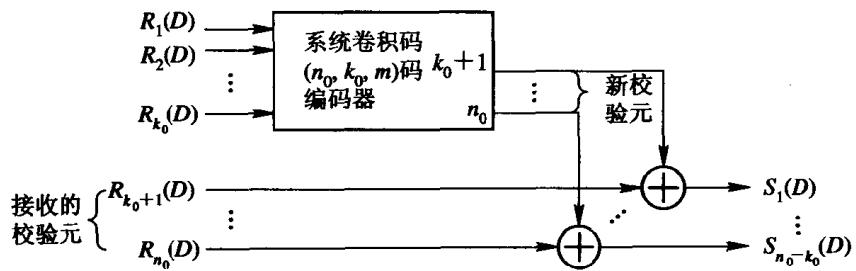


图 10-11 (n_0, k_0, m) 系统卷积码的伴随式计算电路

原发送的序列 $C(D)$ 相同，也可能不同。

若 $\hat{C}(D) \neq C(D)$ ，说明产生了译码错误。

图 10-12 给出了系统卷积码一般代数译码器的框图，它的工作过程大致如下：

(1) 由收到的序列 R 中，分成两个序列，信息序列 M 与校验序列 P 。 M 序列送入信息缓冲寄存器中寄存，同时送入伴随式计算电路(它就是一个与发端相同的编码器)计算新的校验元，并与收到的校验元序列进行比较，得到伴随式寄存在伴随式寄存器中。

(2) 当接收完第 $m+1$ 段子组后，伴

随式寄存器中的伴随式送到错误图样检测电路，以确定此刻前第 m 个时间单位内，收到的子组(第 0 子组)中的信息位内的错误图样 $\hat{e}_{01}\hat{e}_{02}\cdots\hat{e}_{0k_0}$ 。这时第 0 子组的信息元已移至信息元存贮器的最右边，与错误图样 $\hat{e}_{01}\hat{e}_{02}\cdots\hat{e}_{0k_0}$ 相减后就得到了已纠正过的第 0 子组内的信息元 $\hat{m}_{01}\cdots\hat{m}_{0k_0}$ 。若 $\hat{m}_{01}\cdots\hat{m}_{0k_0}$ 与原发送的信息元 $m_{01}\cdots m_{0k_0}$ 相同，则认为译码无错；否则有错。

(3) 由卷积码的编码过程可知，第 0 子组的信息元，对其后 m 段子码中的校验元均有约束关系，因而为了消除第 0 子组中的错误对以后 m 段子组的影响，充分发挥码的纠错能力，就反馈一个信号给伴随式寄存器以修正伴随式，消除第 0 子组内错误对以后子组的影响。

(4) 当第 0 子组的信息元译出之后，第一子组的信息元就可根据修正伴随式后的第一至第 $m+1$ 段码元以同一算法译出，并对伴随式作相应修正。这一过程可一直继续下去，每次接收 n_0 个码元，译出 k_0 个信息元，并对伴随式进行修正。

像这种每次对伴随式进行修正的译码方法，称为反馈译码，其译码约束长度 $n_F = (m+1)n_0$ 。若每次译码时不对伴随式进行修正，则称为定译码。

(5) 若为非系统码，则得到已纠过错的码段 \hat{C}_i 后，再令其通过乘 $G^{-1}(D)$ 的电路得到信息组 \hat{m}_i 。

(n_0, k_0, m) 非系统卷积码的生成矩阵 $G(D)$ 是 $k_0 \times n_0$ 阶矩阵，不是一个满秩阵，因而没有也不能用一般求矩阵逆阵的方法得到右逆阵 $G^{-1}(D)$ 。但在编码理论中可根据

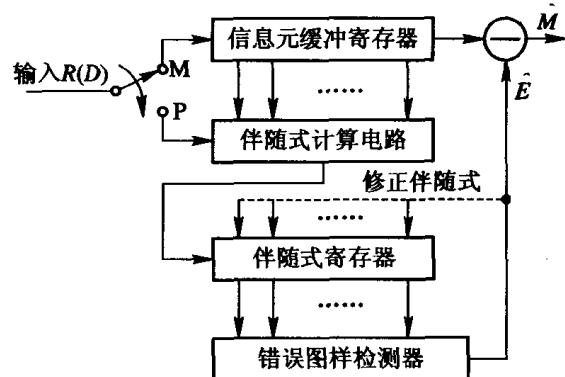


图 10-12 系统卷积码的一般译码器

$$\mathbf{G}(D)\mathbf{G}^{-1}(D) = \mathbf{I} \quad (10.3.5)$$

通过待定系数法求得拟逆阵 $\mathbf{G}^{-1}(D)$ ，这里 \mathbf{I} 是 $k_0 \times k_0$ 阶单位方阵。

例如(2, 1, 2)非系统码的生成矩阵为

$$\mathbf{G}(D) = [1 + D + D^2, 1 + D^2]$$

经计算可得

$$\mathbf{G}^{-1}(D) = \begin{bmatrix} D \\ 1 + D \end{bmatrix}$$

$$\mathbf{G}(D)\mathbf{G}^{-1}(D) = [1 + D + D^2, 1 + D^2] \begin{bmatrix} D \\ 1 + D \end{bmatrix} = 1$$

若 $\mathbf{M} = (1 \ 1 \ 1 \ 0 \ 0 \ \cdots)$ ，则由式(10.2.7)可知：

$$\mathbf{M}(D) = 1 + D + D^2$$

$$\mathbf{C}(D) = \mathbf{M}(D)\mathbf{G}(D) = (1 + D^2 + D^4, 1 + D + D^3 + D^4) = (\mathbf{C}^{(1)}(D)\mathbf{C}^{(2)}(D))$$

把 $\mathbf{C}(D)$ 左乘 $\mathbf{G}^{-1}(D)$ 得

$$\begin{aligned} \mathbf{C}(D)\mathbf{G}^{-1}(D) &= (1 + D^2 + D^4, 1 + D + D^3 + D^4) \begin{bmatrix} D \\ 1 + D \end{bmatrix} \\ &= 1 + D + D^2 = \mathbf{M}(D) \end{aligned}$$

这就恢复了原来的信息序列 $\mathbf{M}(D)$ 。所以，

有时也称 $\mathbf{G}^{-1}(D)$ 为卷积码的译码多项式矩阵。该(2, 1, 2)码的 $\mathbf{G}^{-1}(D)$ 的电路如图 10-13 所示，图中 $\hat{\mathbf{C}}^{(1)}(D)$, $\hat{\mathbf{C}}^{(2)}(D)$ 分别是经译码器纠错后的码序列， $\hat{\mathbf{M}}(D)$ 就是译码器最终输出的信息估值序列。

事实上，由于 $\mathbf{G}(D)$ 并非满秩，所以 $\mathbf{G}^{-1}(D)$ 也并不是唯一的。如该例中的 $\mathbf{G}^{-1}(D)$ 也可以是

$$\mathbf{G}^{-1}(D) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\mathbf{G}(D)\mathbf{G}^{-1}(D) = [1 + D + D^2, 1 + D^2] \begin{bmatrix} 1 \\ 1 \end{bmatrix} = D$$

从而

$$\mathbf{C}(D)\mathbf{G}^{-1}(D) = \mathbf{M}(D)\mathbf{G}(D)\mathbf{G}^{-1}(D) = D\mathbf{M}(D)$$

由此得到的信息序列仅仅迟延一个单位时间，但乘 $\mathbf{G}^{-1}(D)$ 的电路却异常简单，仅仅是一个模 2 加法器而已。

由此可知 $\mathbf{G}^{-1}(D)$ 也可以从以下方程求得：

$$\mathbf{G}(D)\mathbf{G}^{-1}(D) = \mathbf{I}_{k_0} \quad (10.3.6)$$

在(2, 1, m)非系统码中，若满足式(10.3.6)的两个子生成元如果只相差一个 D^i 项 ($i=0, 1, \dots$)，如该例中(2, 1, 2)码的两个子生成元只差一个 D 项，则译码过程中恢复信息元的电路很简单。通常这种子生成元只相差一个 D^i 项的码称为内快检码或似系统码，说明这类码与系统码相似，可以由译码纠错后的序列 $\hat{\mathbf{C}}(D)$ 中很快得到信息元序列 $\hat{\mathbf{M}}(D)$ ，

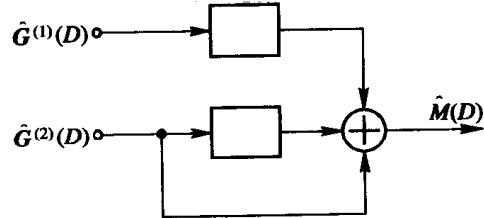


图 10-13 (2, 1, 2)码的乘 $\mathbf{G}^{-1}(D)$ 电路

且延迟仅 i 个单位时间。

对系统码来说，能从 $\hat{C}(D)$ 中直接地很快地得到 $\hat{M}(D)$ ，因而这是一类快检码，译码过程中的第(5)步可以省略，这是系统码相对非系统码的一大优点。

在卷积码的代数译码中，一般就是这两种译码方法。通常情况下反馈译码能充分利用码的纠错能力，在同样的码参数下，它比定译码的纠错能力要强些。但在反馈译码中，存在一个问题，那就是下节介绍的误差传播问题。

§ 10.4 误差传播

反馈译码的优点是在前面各子组已正确译码前提下，能最大限度地利用码的纠错能力来译当前需要译的码段。但是，当前面某一段产生了错误译码后，对伴随式修正的结果，则不仅不能把某段信息位上的错误对以后 m 段上的影响消去，反而叠加一个新的错误，该错误至少还要影响后 m 段上译码的正确性，这种现象称为误差传播。它是卷积码反馈译码中所特有的问题。

本节首先讨论误差传播的一般概念，然后讨论什么情况下会产生误差传播，从而找出不存在误差传播的条件。

一、误差传播的基本概念

卷积码反馈译码中的误差传播分为两类：一类称为**有限误差传播**，也就是一个码段的错误译码，仅对后面有限码段的译码有影响。反之，称为**无限误差传播**，也就是说当某段的译码产生错误以后，即使以后所有各子组的输入全部正确，但仍会引起以后各段码组的错译。下面举例说明这种现象。

例 10.3 (2, 1, 5)二进制卷积码。它的

$$\mathbf{h} = [10, 10, 10, 00, 00, 11]$$

在译码约束长度 $n_F = 12$ 个码元长度内，能纠正两个随机错误。该码的译码器如图 10-14 所示。设发送的全为 0 序列，而收到的 R 是：(00, 01, 00, 01, 01, 00, 00, …)，在前五段

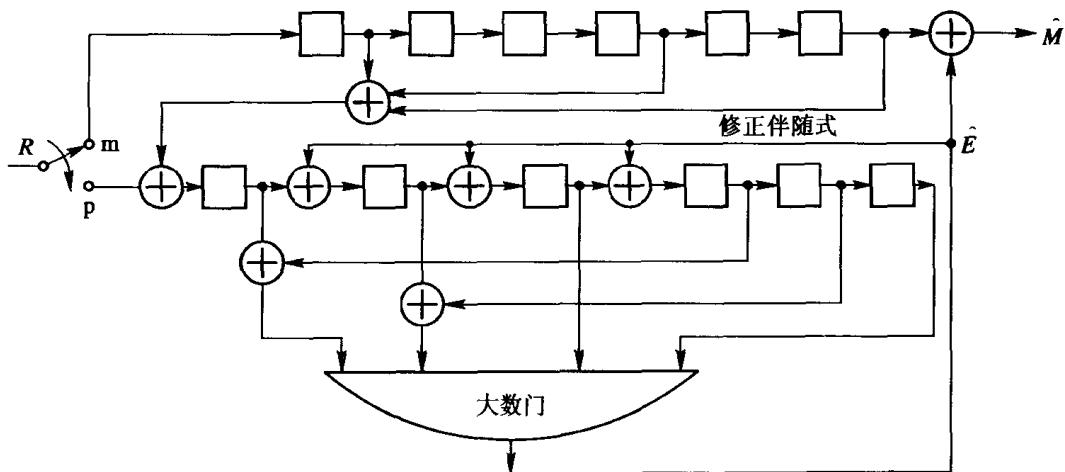


图 10-14 (2, 1, 5) 码反馈译码器

产生了 3 个错，但从第五段以后译码器的输入不再有错误。

当译码器收完第四子组以后，在伴随式寄存器中的存数自左至右为(011010)，再移动一次后变成(001101)，大数门的 3 个输入为 1，因而输出一个“1”对第一子组进行纠错，并反馈一个信号修正伴随式，使伴随式寄存器的存数变为(011010)。再移位一次后，伴随式寄存器内的存数又成为(001101)，结果又对第三子组进行纠错，并反馈给伴随式寄存器一个信号修正伴随式，这样又使得伴随式寄存器的存数成为(011010)。因此，随着译码节拍的不断进行，伴随式寄存器中的存数就处于 $011010 \rightarrow 001101 \rightarrow 011010 \rightarrow 001101 \rightarrow \dots$ 的无限循环之中。从而导致纠错脉冲每隔一个节拍出现一次，使译码器的输出一半有错误。只要信道不再有其它错误，这种误差就一直传播下去。

但如果把修正伴随式的反馈线断开，使反馈译码变成定译码，就不会引起无限误差传播。这并不是说定译码总是优于反馈译码，一旦我们克服了误差传播影响之后，反馈译码就优于定译码了。

那么，如何会造成无限误差传播呢？其根本原因是在反馈译码下，伴随式寄存器实质上是一个非线性反馈移存器，这种移存器的输入全为 0 条件下，当其初始状态为非 0 时，可能会产生周期性的状态循环，从而造成无限误差传播。

无限误差传播又称为**恶性误差传播**，又可分为两类：一类是由于码的子生成元选择不当引起的（即由编码器决定）称为Ⅰ类无限误差传播，并称这类码为**恶性码**，如该例中的(2, 1, 5)码；另一类是由译码器设计不当引起的，称为Ⅱ类无限误差传播。

下面简单讨论防止Ⅰ、Ⅱ类无限误差传播的条件。

二、Ⅰ类无限误差传播的条件

要防止出现Ⅰ类无限误差传播，首先必须讨论存在Ⅰ类无限误差传播的条件是什么。以例说明之。

设一个(2, 1, 2)二进制卷积码，它的子生成多项式为： $\mathbf{g}^{(1,1)}(D) = 1 + D$, $\mathbf{g}^{(1,2)}(D) = 1 + D^2$ 。若输入至编码器的信息序列全为 1， $\mathbf{M}(D) = 1 + D + D^2 + D^3 + \dots = 1/(1 + D)$ ，则相应的编码器输出的码序列为

$$\mathbf{C}(D) = \mathbf{M}(D)\mathbf{G}(D) = (11) + (01)D + (00)D^2 + (00)D^3 + \dots = (1, 1 + D)$$

即

$$\mathbf{C}^{(1)}(D) = 1 \quad \mathbf{C}^{(2)}(D) = 1 + D$$

它仅有 3 个非 0 元素。这就是说由这两个子生成元确定的编码器，将一个有无限多个非 0 元素的输入序列，变换成一个只有有限个非 0 元素的输出序列。如果在信道的传输过程中刚好将该码序列中的 3 个非 0 元素错成 0，则译码器输入序列将是一个全为 0 序列，因而相应的译码器必输出一个全为 0 的信息元估值序列 $\hat{\mathbf{M}}$ ，即 $\hat{\mathbf{M}}(D) = 0$ 。这样，在信道干扰只有使 3 个码元错误情况下，却使得译码器的输出序列有无限多个错误。

若选用码的子生成多项式为 $\mathbf{g}^{(1,1)}(D) = 1 + D$, $\mathbf{g}^{(1,2)}(D) = 1 + D + D^2$ ，则对应于信息序列为 $\mathbf{M}(D) = 1 + D + D^2 + \dots = 1/(1 + D)$ 的码序列为

$$\begin{aligned} \mathbf{C}(D) &= \mathbf{M}(D)\mathbf{G}(D) = (11) + (00)D + (01)D^2 + (01)D^3 + \dots \\ &= (1, 1 + D^2 + D^3 + \dots) \end{aligned}$$

即

$$C^{(1)}(D) = 1 \quad C^{(2)}(D) = \frac{1+D+D^2}{1+D} = D + \frac{1}{1+D}$$

可见码序列中非 0 元素的个数也为无限多个。由

$$G(D)G^{-1}(D) = [1+D, 1+D+D^2] \begin{bmatrix} D \\ 1 \end{bmatrix} = D + D^2 + 1 + D + D^2 = 1$$

注意，这里矩阵 $G(D)$ 的逆与一般满秩矩阵的逆不相同，这里仅是借用逆阵这一名词而已。可知，此码生成多项式的逆生成多项式是

$$G^{-1}(D) = \begin{bmatrix} D \\ 1 \end{bmatrix}$$

若以此逆生成多项式作为码的生成多项式，则由此生成多项式产生的码，称为原码的逆码。若把原码的码序列输入逆码编码器，则

$$\begin{aligned} [C^{(1)}(D), C^{(2)}(D)]G^{-1}(D) &= [C^{(1)}(D), C^{(2)}(D)] \begin{bmatrix} D \\ 1 \end{bmatrix} = DC^{(1)}(D) + C^{(2)}(D) \\ &= D + \frac{1+D+D^2}{1+D} = \frac{1}{1+D} = M(D) \end{aligned}$$

恢复了原来的输入信息序列。可知，逆码编码器就是原码的译码器。由 $G^{-1}(D) = \begin{bmatrix} D \\ 1 \end{bmatrix}$ 知，译码器中仅有一级存贮器，故信息元中的错误只要经过一个单位时间即被遗忘掉，我们称这种逆生成多项式（以下简称逆）为前馈逆。

但是前一例中的

$$G(D) = [1+D, 1+D^2]$$

它的逆为

$$G^{-1}(D) = \begin{bmatrix} \frac{D}{1+D} \\ \frac{1}{1+D} \end{bmatrix}$$

式中，分母项为 $1+D$ ，在电路实现上相当于作除法，即电路中有反馈，称这种逆为反馈逆。这种电路的记忆为无限的（因为 $1/(1+D) = 1+D+D^2+D^3+\dots$ ），因而信道错误的影响长时间不能消除。由此可知，当一个编码器的 $G(D)$ 有一个无反馈逆或有一个前馈逆时，就不会造成 I 类无限误差传播。

下面两个定理给出了 $(n_0, 1, m)$ 卷积码与 (n_0, k_0, m) 卷积码有一个前馈逆的充要条件。

定理 10.4.1^[4] $(n_0, 1, m)$ 卷积码有一个前馈逆的充要条件是：

$$\text{GCD}[g^{(1,1)}(D), g^{(1,2)}(D), \dots, g^{(1,n_0)}(D)] = D^L \quad (10.4.1)$$

式中， $L \geq 0$ ，且存在有一个最小迟延为 L 的逆，其它任何迟延的逆都小于它。

定理中所说的最小迟延为 L 的逆如下定义。若

$$G(D)\tilde{G}^{-1}(D) = I_{k_0}D^L \quad (10.4.2)$$

则称 $\tilde{G}^{-1}(D)$ 是迟延为 L 的逆，这里 I_{k_0} 是 $k_0 \times k_0$ 阶单位方阵。若 $L = 0$ ，则 $\tilde{G}^{-1}(D)$ 就是迟延为 0 的逆，这是一个最小迟延的逆。

如由 $g^{(1,1)}(D) = 1+D$, $g^{(1,2)}(D) = 1+D+D^2$ 组成的 $G(D)$ ，它的

$$\text{GCD}(1+D, 1+D+D^2) = D^0 = 1$$

且

$$G(D)G^{-1}(D) = 1$$

因而满足有前馈逆阵的条件式(10.4.1)，所以该码无 I 类无限误差传播，而由

$$G(D) = [1 + D^2, 1 + D]$$

生成的码，由于

$$\text{GCD}(1 + D, 1 + D^2) = 1 + D \neq D^L$$

不满足式(10.4.1)，故是一个有 I 类无限误差传播的码。

定理 10.4.2^[4] (n_0, k_0, m) 卷积码有前馈逆的充要条件是

$$\text{GCD}\left(\Delta_i, i = 1, 2, \dots, \binom{n_0}{k_0}\right) = D^L \quad (10.4.3)$$

式中， $L \geq 0$ ，且恰有一个迟延为 L 的逆存在(对 $k_0 > 1$ 可能有迟延小于 L 的逆存在)。式中， Δ_i 是 $k_0 \times n_0$ 阶 $G(D)$ 阵中所有可能的 $k_0 \times k_0$ 阶子阵的行列式。

如(3, 2, 2)卷积码的生成矩阵是

$$G(D) = \begin{bmatrix} 1 & 0 & 1 + D^2 \\ 0 & 1 & 1 + D \end{bmatrix}$$

它的所有二阶子阵行列式为：1, $1+D^2$, $1+D$ 。因而

$$\text{GCD}(1, 1 + D^2, 1 + D) = D^0 = 1$$

且

$$G(D)G^{-1}(D) = \begin{bmatrix} 1 & 0 & 1 + D^2 \\ 0 & 1 & 1 + D \end{bmatrix} \begin{bmatrix} D^2 & 1 + D^2 \\ 1 + D & D \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} D^0$$

满足定理 10.4.2 的要求，所以该码是一个无 I 类误差传播码。

事实上，由式(10.2.17)可知，系统码的生成矩阵 $G(D) = [\mathbf{I}_{k_0} \mathbf{P}(D)]$ ，它的 $k_0 \times k_0$ 阶子阵行列式 Δ_i 中一定有一个 1，因而 $\text{GCD}(1, \Delta_{i_1}, \Delta_{i_2}, \dots) = D^0 = 1$ 。所以对系统码来说无 I 类误差传播，这是系统码优于非系统码的另一点。

三、Ⅱ类无限误差传播的条件

Ⅱ类无限误差传播是由于译码器的约束度太短引起的(即使码本身不是恶性的)。因此，要避免Ⅱ类误差传播只要译码约束度 N_d 选得足够大时就可避免。但从译码器的复杂性考虑，希望 N_d 尽可能小，究竟应选多大合适？下面的定理给出了 N_d 的一个上限。

定理 10.4.3 一个 $(n_0, 1, m)$ 非恶性的二进制码，设 ν 是其中子生成多项式中两两互素的最大个数 ($1 < \nu \leq n_0$)，若用译码约束度

$$N_d = \left\lfloor \frac{mn_0}{\nu - 1} \right\rfloor m + m \quad (10.4.4)$$

的最小距离译码器，则就无Ⅱ类无限误差传播。式中， $\lfloor x \rfloor$ 表示 x 的整数部分。

该定理给出了 N_d 的上限，但试验表明，在一般反馈码条件下，若选用 $N_d = m + 1$ ，则对系统码一般就不会引起Ⅱ类误差传播。而对某些非系统的非恶性的码来说，可能会引起Ⅱ类误差传播，这时就必须选用更大的 N_d 。因此，为了避免Ⅱ类无限误差传播， N_d 必须在下列范围内选：

$$m + 1 \leq N_d \leq \left\lfloor \frac{mn_0}{\nu - 1} \right\rfloor m + m \quad (10.4.5)$$

总之，为了克服恶性误差传播，一方面应选用非恶性码，另一方面 N_d 应选得足够大。

四、克服误差传播的其它方法

由以上讨论知，为了防止无限误差传播，选的码必须满足定理 10.4.2 的要求，且译码约束长度要足够长。满足这两个要求的码称为具有**自动恢复特性**的码。这种特性能使按反馈译码的译码器，很快地从错误译码中恢复正常工作，只要在发生错误译码后，在很短的接收码元内不再有信道错误就可以了。除此以外，还有其它方法克服无限误差传播。

方法之一是**周期同步法**。从卷积码的译码过程可知，码序列中的任何一段子码，只和本段及后 m 段发生关系。因此，在译第 l 段时，比第 $l-m$ 段更前的段中的错误，对第 l 段及其以后各段的伴随式计算均没有影响。所以，如果译码错误早于第 $l-m$ 段，且从第 $l-m$ 段到第 l 段上不再有译码错误，则译第 l 段所利用的伴随式就不再有以前错误伴随式的影响了。这就是说，译码错误出现之后，若跟着有 m 段能正确译码，则译码器就恢复到正常状态。误差传播必然中止。这就给我们提供了一种控制误差传播的方法。

在编码时，我们把信息元分段，每隔 L 段 ($k_0 L$ 个) 信息元后，跟着将 $k_0 m$ 个固定不变的(一般是 $k_0 m$ 个零) 码元送入编码器编码。译码时，每当这 $k_0 m$ 个确知码元所对应的码序列到达译码器并被译码器识别后，译码器就自动将其译为 $k_0 m$ 个原先发送的码元序列。这样就保证每隔 L 段必有连续 m 段的正确译码，因而使误差传播限制在 $L+m$ 段内，这 $k_0 m$ 个固定不变的码元序列称为**同步序列**。这种防止误差传播的方法也就称为周期同步法。当然，当 L 段信息元送完后，为了使编译码器恢复到初始的全 0 状态，一般情况下还需再送 mk_0 个全 0 码元，应用这种方法后传信率由原来的 R 下降为 $R_s = (L/L+m)R$ ，式中， $R = k_0/n_0$ 。若 $L \gg m$ ，则 $R_s \approx R$ ，因此传信率牺牲并不大。

如果接收端有反馈信道可利用，则还可采用一种“**数错法**”来控制误差传播。这种方法是，译码时译码器计数连续几个码段长度的纠错脉冲的个数，一旦此数目超过该码在此长度内的纠错能力，译码器便认为译码不正确，或认为信道处于严重干扰情况下，此时译码器立即发出一个反馈信号给发端，要求重发从某一时刻起的码元序列，从而控制误差传播。当然，使用这种方法不仅要求有反馈信道，而且必须附加一些控制电路。

§ 10.5 卷积码的树图描述和距离特性

卷积码的树图表示是一种很形象的表示卷积码编译码过程的方法，特别在卷积码的概率译码中，更经常要用到这种表示方法。此外，卷积码的各种距离量度，也往往与树图发生密切关系，因此卷积码的树图表示是一种非常重要的描述卷积码的方法。本节先介绍卷积码的树图表示，然后再介绍各种距离量度。

一、卷积码的树图描述

若输入 (n_0, k_0, m) 卷积码编码器的信息序列是一个半无限长序列，则由卷积编码器输出的码序列也是一个半无限长的序列，且每一码段前后 m 段之间均有约束关系。描述卷积

码这种特性的除了用它的生成矩阵 G_∞ 和校验矩阵 H_∞ 外，还可用半无限码树图来描述。

如例 10.1 中的 $(2, 1, 2)$ 码，它的生成多项式矩阵和生成矩阵分别为：

$$G(D) = [1 + D + D^2, 1 + D^2]$$

$$G_\infty = \begin{bmatrix} 11 & 10 & 11 \\ & 11 & 10 & 11 \\ & & 11 & 10 & 11 \\ & & & \cdots \\ & & & \cdots \end{bmatrix} = \begin{bmatrix} g_0 & g_1 & g_2 \\ g_0 & g_1 & g_2 \\ g_0 & g_1 & g_2 \\ & \cdots \\ & \cdots \end{bmatrix}$$

若输入编码器的信息序列 $M = (m_0, m_1, \dots) = (1 \ 1 \ 0 \ 1 \ 1 \ \dots)$ ，则由编码器输出的码序列 C 为

$$C = MG_\infty = (11, 01, 01, 00, 01, 01, \dots) = (c_0, c_1, c_2, c_3, \dots)$$

我们可把这个编码过程用如图

10-15 所示的半无限码树图说明。设编码器初始状态为 0，输入信息序列 $M = (m_0, m_1, \dots)$ ，则编码器输出的第一段子码 c_0 仅由 m_0 确定。若 $m_0 = 0$ ，则 $c_0 = (00)$ ，在码树上相应于从第 0 级节点（初始节点）出发走上一分支输出 (00)；若 $m_0 = 1$ ，则在码树上相应走下面分支，输出 $c_0 = (11)$ 。同理，当第二个信息组 m_1 输入时，这时编码器已处在第一阶节点上。若在 a 点，则输入 $m_1 = 0$ 时，走上面分支输出 $c_1 = (00)$ ；若 $m_1 = 1$ ，则走下面分支输出 $c_1 = (11)$ 。若在 b 点，则输入 $m_1 = 0$ 时，走上面分支输出 $c_1 = (10)$ ；若 $m_1 = 1$ ，则走下面分支输出 $c_1 = (01)$ ，此时编码器已处在第二阶节点上。若再输入 m_2 ，则编码器从第二阶节点出发，按输入为 0 走上面分支，输入为 1 走下面分支的规则输出 c_2 。这

样随着信息序列的不断输入编码器，从码树上的一个节点走向下一个节点，并送出相应的子组。因此输入不同的信息序列，编码器就走不同的路径，输出不同的码序列。如输入信息序列 $M = (0 \ 0 \ 0 \ \dots)$ ，则输出码序列为 $(00, 00, 00, \dots)$ 相应于码树上的最上一条支路；若输入的信息序列为 $(1 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ \dots)$ ，则输出的码序列为 $(11, 01, 01, 00, \dots)$ ，相应于码树中用粗线表示的一条路径。对该码序列来说，树图上的这条路径就是它的正确路径，而其它的所有路径都是它的不正确路径。

由上讨论可知，卷积编码器编码过程的实质，就是在输入信息序列的控制下，编码器沿码树通过某一特定路径的过程。显然，译码过程就是根据接收序列和信道干扰的统计特性，译码器在原码树上力图恢复原来编码器所走的路径即寻找正确路径的过程。

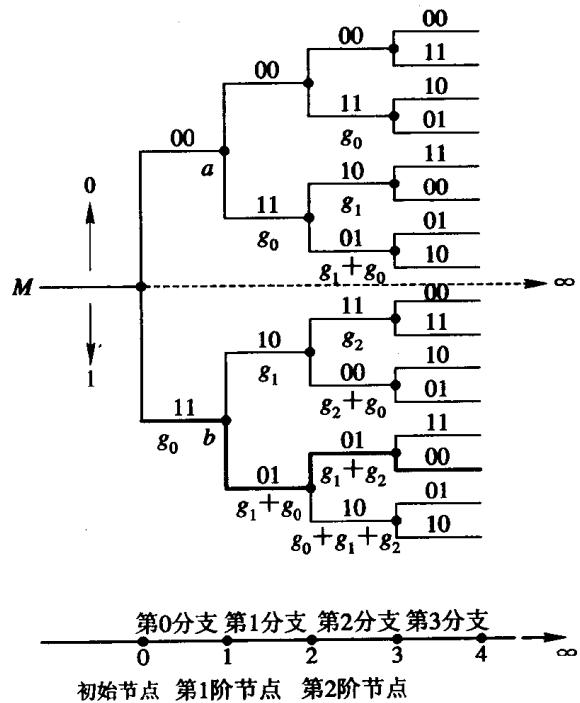


图 10-15 $(2, 1, 2)$ 码码树图

上面所例举的码树图是针对(2, 1, 2)码编码器。对一般的二进制(n_0, k_0, m)编码器来说，每次输入的是 k_0 个信息元，有 2^{k_0} 个可能的信息组，这相应于从码树每一节点上分出的分支数有 2^{k_0} 条，相应于 2^{k_0} 种不同信息组的输入，并且每条都有 n_0 个码元，作为与此相应的输出子码。

卷积编码器的码树图表示，在卷积码的概率译码，如序列译码和Viterbi译码中将经常要用到，以后还将进一步讨论。

所以，编码器输出的码序列就是在输入的信息序列控制下，编码器沿码树所走的某一路径所对应的子码序列。因此，码树上所有可能的路径，就是该卷积码编码器所有可能输出的码序列。若在码树中间划一条直线（如图10-15中的虚线），把码树分成上、下相等的两部分，这相应于把所有码序列划分成大小相等的两个子集 s_0 和 s_1 。显然，在同一子集中它们有相同的第0段子码（第0分支的值），两个不同子集之间都有不同的第0段子码。如图10-15中，若虚线上半部分的子集为 s_0 ，下半部分的子集为 s_1 ，则 s_0 中所有路径（码序列）都有相同的第0分支值 $c_0=00$ ，它相应于输入信息元 $m_0=0$ 时的情况。同样，在 s_1 子集中有相同的第0段分支值 $c_0=11$ ，它相应于 $m_0=1$ 时的情况，如图10-16(a)所示。在 s_0 和 s_1 子集中，又可以把它们划分成大小相等的两个子集： s_{00} 和 s_{01} ， s_{10} 和 s_{11} 。每个子集中的所有路径不仅含有相同的第0分支值 c_0 ，而且还含有相同的第一分支值 c_1 ，如图10-16(b)所示。显然，第二次划分，相应于输入信息元为 m_1 时的情况。当然这种划分可以无限地进行下去。对于一般的(n_0, k_0, m)卷积码来说，也可在它的码树上进行这种子集划分，只不过由于信息组的每次输入有 2^{k_0} 个不同的值，相应于从每一节点出发可走 2^{k_0} 条分支，因而每次划分含有 2^{k_0} 个子集。

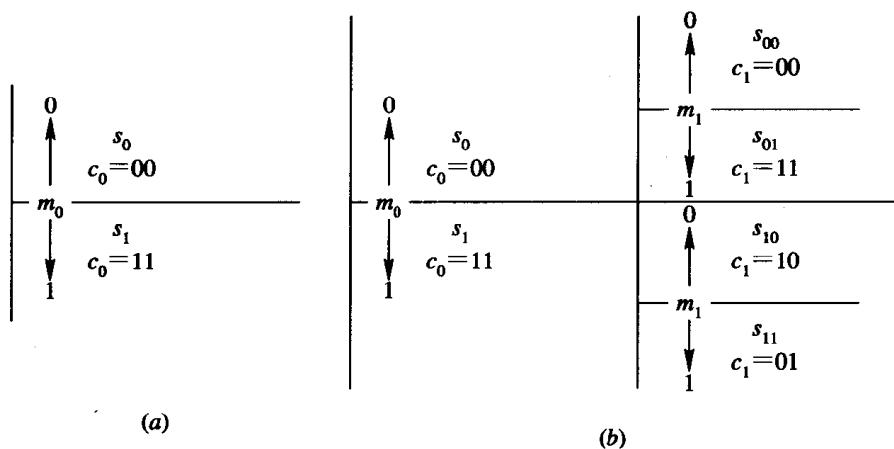


图 10-16 码树中子集的划分

从码树结构上看，初始截短码就是从半无限码树上截取从第0节点到第 $m+1=N$ 节点之间，所有分支所组成的有限码树（或称初始截短码树）。它共有 N 阶节点和 N 段分支，如图10-17就是从图10-15的半无限码树上截取的有限码树。它由三阶节点和三段分支所组成，共有 $2^{k_0(m+1)}=2^3=8$ 个可能的路径，它们就是 $(n_0N, k_0N)=(6, 3)$ 初始截短码的所有8个码序列。对这8个码序列也可以进行子集划分，把第0段分支值相同的归成一个子集，这相应于图10-17中虚线上所有路径划成一个子集 s_0 ，虚线以下的所有路径划分

成另一个子集 s_1 。对于一般的 (n_0N, k_0N) 初始截短码，则可划分成 2^{k_0} 个子集： $s_0, s_1, \dots, s_{2^{k_0}-1}$ 。

当然，从半无限码树上，也可以任意地截取第 0 阶节点或第 $m' + 1$ 阶节点之间的所有支路，构成一个有限码树，有限码树上的所有路径就构成了一个 $(n_0(m' + 1), k_0(m' + 1))$ 线性码的码序列，共有 $2^{k_0(m'+1)}$ 个，且对这些码序列也可进行如上所述的子集划分。

由上节可知，卷积码的译码约束长度 $N_d = (m_d + 1)$ ，则译码器根据收到的 $m_d + 1$ 段子组，计算它的伴随式，然后对此时刻前收到的第 m_d 子组（即第 0 子码）进行译码。从码树上看，这相当于根据接收序列，译码器在码树上推进到 $m_d + 1$ 阶节点时，译码器来确定从第 0 阶节点出发的 2^{k_0} 个分支中，哪一条是最似的分支，或者说在 $s_0, s_1, \dots, s_{2^{k_0}-1}$ 个子集中，确定一个最似的子集。若为 s_0 ，则第 0 段接收序列就译为相应的第 0 组子码 c_{00} (c_{00} 是 s_0 中所有码序列的第 0 段子码) 或相应的信息组。然后，译码器推进到第 $m_d + 2$ 阶节点，译码器译第一个子组等等。

如以 $(2, 1, 2)$ 码为例，选择码的约束长度 $N_d = N = 3$ ，因而只有 $2^{k_0(m'+1)} = 2^3 = 8$ 个码序列，可分成 $2^{k_0} = 2$ 个子集 s_0, s_1 ，每个子集含有 4 个码序列，而可能的接收序列有 $2^{n_0(m'+1)} = 2^6$ 个，把它们划给不同的子集中，如表 10-1 所示，称此表为译第 0 段的译码表。

从译码表看出，如果发送码序列的第 0 分支为 00，则凡是接收序列中的错误图样与最左边一列中的图样一致，一定可使第 0 分支作出正确判断，也就是说如果接收序列落在最左边的 4 列之中，则一定能对第 0 子组作出是 00 的正确译码。因此，称这 4 列中的元素所组成的子集（包括 s_0 子集在内）为译第 0 分支是 00 的正确子集，以 A_0 表示；而另外一个子集（若为 (n_0, k_0, m) 码，则还有 $2^{k_0} - 1$ 个子集）称为不正确子集，以 A_1 表示。反之，若发送码序列的第 0 分支是 11，则称 A_1 是译第 0 分支是 11 的正确子集；而 A_0 为不正确子集。

表 10-1 译第 0 段译码表

| 正确子集 A_0 | | | | 不正确子集 A_1 | | | |
|----------------------------|-----------------------------|-----------------------------|-----------------------------|-------------|--|--|--|
| 序 列 | 000000 000011 001110 001101 | 111011 111000 110101 110110 | 011011 011000 010101 010110 | | | | |
| 其 它 接 收 序 列 | 100000 100011 101110 101101 | 010111 101000 100101 100110 | 101011 101000 100101 100110 | | | | |
| | 010000 010011 011110 011101 | 110011 110000 111101 111110 | 110011 110000 111101 111110 | | | | |
| | 001000 001011 000110 000101 | 111111 111100 110001 110010 | 111111 111100 110001 110010 | | | | |
| | 000100 000111 001010 001001 | 111001 111010 110111 110100 | 111001 111010 110111 110100 | | | | |
| | 000010 000001 001100 001111 | 011111 011100 010001 010010 | 011111 011100 010001 010010 | | | | |
| | 100100 100111 101010 101001 | 011001 011010 010111 010100 | 011001 011010 010111 010100 | | | | |
| | 100010 100001 101100 101110 | 011001 011010 010111 010100 | 011001 011010 010111 010100 | | | | |
| | | | | | | | |
| 译码器 输出 | 0 | 1 | | | | | |

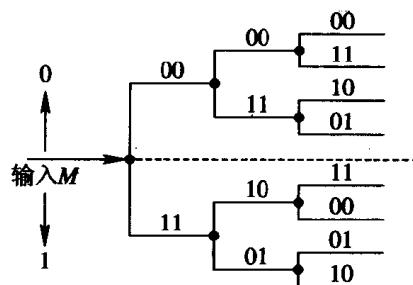


图 10-17 $(2, 1, 2)$ 码初始截短码树

当译码器译完第 0 子组，并译第一子组时，由码树图可知：第一段子码（第一段分支值）有 4 种可能，即 00, 11, 01, 10，因而至少需两个译码表；而译第二段子码及以后各段子码，则需 4 个译码表才能全面说明译码过程。但由于线性卷积码的对称性，只要用一个译码表就能说明问题，而不必列出 4 个。

在编码理论中，可以用无限或半无限树图描述其编、译码过程的码称为树(Tree)码，由上讨论可知卷积码是一类具有有限约束长度、满足式(10.2.7)卷积运算、且子生成元不随时间变化的线性树码。

二、卷积码的距离度量

衡量卷积码的纠错性能是用它的距离特性来描述的。由于卷积码的纠错能力与它所采用的译码方法有很大关系，因此不同的译码方法就有不同的距离量度，这里仅介绍几个常用的距离量度和定义。

定义 10.5.1(汉明距离) (n_0, k_0, m) 卷积码的最小汉明距离 d ，定义为整个码树上不同子集的码字之间距离最小值。

该距离量度适用于各节点有相同距离特性的码，这时 d 就是译码表中不同子集的码字之间的最小距离。如表 10-1 中的(2, 1, 2)码，两个不同码字集合之间的最小距离 $d=3$ 。

在代数译码情况下，若译码约束度 N_d 等于编码约束度 $N=m+1$ ，则采用初始截短码字之间的最小距离 d_{\min} 更为有用。

定义 10.5.2(最小汉明距离) (n_0, k_0, m) 卷积码的最小汉明距离 d_{\min} 定义为不同初始截短码字子集之间距离的最小值

$$d_{\min} = \min_{i \neq j} \{d(u, v) | u \in s_i, v \in s_j, i, j = 1, 2, \dots, 2^{k_0}\} \quad (10.5.1)$$

式中， s_i, s_j 是两个不同的初始截短码字子集， u, v 是子集内的初始截短码字。

这里仅讨论线性卷积码，所以若 u, v 是初始截短码字，则 $u+v=z$ 也是初始截短码字。由于 u 和 v 的第 0 子组不同，因而 z 的第 0 子组不可能全为 0，令 $w(z)$ 表示 z 的重量，则 $d(u, v) = w(z)$ 。由此可知，卷积码的最小距离就等于第 0 子组为非 0 的初始截短码字的最小重量。上例中，(2, 1, 2) 码的最小距离 $d_{\min}=3$ ，它就是 s_1 子集中初始截短码字的最小重量。

最小距离表明了卷积码在连续 $m+1$ 段以内的距离特性，若最小距离为 d_{\min} ，则能在 $m+1$ 连续段内纠正 $t \leq \lfloor (d_{\min}-1)/2 \rfloor$ 个随机错误，下面的定理说明了这点。

定理 10.5.1 设 \mathbf{E}_1 和 \mathbf{E}_2 是两个长为 $m+1$ 段，且第 0 组不同的错误图样。若 \mathbf{E}_1 和 \mathbf{E}_2 的重量 $w(\mathbf{E}_1)$ 和 $w(\mathbf{E}_2)$ ，均小于 $t \leq \lfloor (d_{\min}-1)/2 \rfloor$ ，则 \mathbf{E}_1 和 \mathbf{E}_2 不可能产生相同的伴随式，因而有

$$\mathbf{E}_1 \mathbf{H}^T \neq \mathbf{E}_2 \mathbf{H}^T$$

证明 用反证法证明。若 \mathbf{E}_1 和 \mathbf{E}_2 产生相同的伴随式，则：

$$\mathbf{E}_1 \mathbf{H}^T - \mathbf{E}_2 \mathbf{H}^T = 0$$

$$(\mathbf{E}_1 - \mathbf{E}_2) \mathbf{H}^T = 0$$

$\mathbf{E}_1 - \mathbf{E}_2 = \mathbf{z}_1$ ，它必是第 0 段为非 0 的初始码矢，且 $w(\mathbf{z}_1) = w(\mathbf{E}_1 - \mathbf{E}_2) \leq w(\mathbf{E}_1) - w(\mathbf{E}_2) < d_{\min}$ ，而这与初始截短码字有重量至少为 d_{\min} 相矛盾，所以

$$\mathbf{E}_1 \mathbf{H}^T \neq \mathbf{E}_2 \mathbf{H}^T$$

下面定理说明，若相邻 $m+1$ 段内的错误个数大于 $\lfloor (d_{\min} - 1)/2 \rfloor$ ，则就可以找到某些不能纠正的错误图样。

定理 10.5.2 若 (n_0, k_0, m) 卷积码的最小距离为 d_{\min} ，则在 $m+1$ 段相邻码段内，总可以找到某一错误图样 \mathbf{E}_1 ，它的重量 $w(\mathbf{E}_1) = \lfloor (d_{\min} - 1)/2 \rfloor + 1$ ，不能被码所纠正。

证明 只要证明 \mathbf{E}_1 伴随式与另一重量小于或等于 $\lfloor (d_{\min} - 1)/2 \rfloor$ 错误图样 \mathbf{E}_2 的伴随式相等即可。

设 $\mathbf{u} \in s_i$ 是第 0 段为非 0 的一个初始码字，其重量 $w(\mathbf{u}) = d_{\min}$ 。由假设 \mathbf{E}_1 是已知的，现找一个错误图样 $\mathbf{E}_2 = \mathbf{u} - \mathbf{E}_1$ ，或 $\mathbf{E}_2 + \mathbf{E}_1 = \mathbf{u}$ 。由于 $\mathbf{u} \in s_i$ ，所以 \mathbf{E}_1 与 \mathbf{E}_2 两个错误图样的第 0 段不相同，又因为 \mathbf{u} 是一个初始码字，故

$$\begin{aligned}\mathbf{u} \cdot \mathbf{H}^T &= (\mathbf{E}_1 + \mathbf{E}_2) \cdot \mathbf{H}^T = 0 \\ \mathbf{E}_1 \mathbf{H}^T &= (-\mathbf{E}_2) \mathbf{H}^T\end{aligned}$$

这说明 \mathbf{E}_1 和 $(-\mathbf{E}_2)$ 两个错误图样的伴随式相同，所以其中之一必不能纠正。若 d_{\min} 为奇数，则 $w(-\mathbf{E}_2) = d_{\min} - ((d_{\min} - 1)/2) - 1 = (d_{\min} - 1)/2 < w(\mathbf{E}_1) = (d_{\min} - 1)/2 + 1$ ，说明 $(-\mathbf{E}_2)$ 能纠正， \mathbf{E}_1 不能被纠正。若 d_{\min} 为偶数，则

$$w(\mathbf{E}_1) = w(-\mathbf{E}_2) = \frac{d_{\min} - 1}{2} + 1$$

说明至少其中一个错误图样不能被纠正。

上述两个定理说明，任何相邻 $m+1$ 段以内的错误个数，不多于 $\lfloor (d_{\min} - 1)/2 \rfloor$ 个，则最小距离为 d_{\min} 的 (n_0, k_0, m) 卷积码，总可以把它们纠正。当然，若此卷积码用来检错，则一定可以检测出 $m+1$ 段相邻码段内 $d_{\min} - 1$ 个错误。

在卷积码的概率译码中，一般都是译码约束长度大于编码约束长度。在这种情况下，最有用的距离量度是码的自由距离，为此首先介绍列距离、行距离及其它距离的概念。

定义 10.5.3 (n_0, k_0, m) 卷积码的第 j 阶列距离 d_j ，是 $j+1$ 段长截短码树上定义的最小汉明距离：

$$d_j = \min_{m_0 \neq m'_0} d(\mathbf{M}_j \mathbf{G}_j, \mathbf{M}'_j \mathbf{G}_j) \quad (10.5.2)$$

式中：

$$\begin{aligned}\mathbf{M}_j &= (m_0, m_1, \dots, m_j) \\ \mathbf{M}'_j &= (m'_0, m'_1, \dots, m'_j) \\ \mathbf{G}_j &= \begin{bmatrix} \mathbf{g}_0 & \mathbf{g}_1 & \cdots & \mathbf{g}_j \\ & \mathbf{g}_0 & \cdots & \mathbf{g}_{j-1} \\ & & \vdots & \\ & & & \mathbf{g}_0 \end{bmatrix}\end{aligned}$$

是 \mathbf{G}_∞ 中截取约 $j+1$ 列和 $j+1$ 行的截短矩阵。

如上例中 $(2, 1, 2)$ 码的 2 阶列距离 d_2 为

$$\begin{aligned}d_2 &= \min_{m_0 \neq m'_0} d(\mathbf{M}_2 \mathbf{G}_2, \mathbf{M}'_2 \mathbf{G}_2) \\ \mathbf{G}_2 &= \begin{bmatrix} \mathbf{g}_0 & \mathbf{g}_1 & \mathbf{g}_2 \\ & \mathbf{g}_0 & \mathbf{g}_1 \\ & & \mathbf{g}_0 \end{bmatrix} = \begin{bmatrix} 11 & 10 & 11 \\ & 11 & 10 \\ & & 11 \end{bmatrix}\end{aligned}$$

设 $\mathbf{M}_2 = \begin{pmatrix} 1 & 0 & 1 \end{pmatrix}$, $\mathbf{M}'_2 = \begin{pmatrix} 0 & 1 & 1 \end{pmatrix}$, 则

$$d_2 = \min_{\mathbf{m}_0 \neq \mathbf{m}'_0} \{ d(\mathbf{u}, \mathbf{v}) \mid \mathbf{u} \in s_0, \mathbf{v} \in s_1 \} = d_{\min}$$

就是卷积码的最小汉明距离 d_{\min} 。因此 $m+1$ 阶列距离 $d_{m+1} = d_{\min}$ 。

列距离有如下性质：

$$(1) d_j = \min_{\mathbf{m}_0 \neq 0} w(\mathbf{M}_j \mathbf{G}_j) \quad (10.5.3)$$

即第 j 列距离就是第 0 码段为非 0 的, $j+1$ 段截短码字的最小重量。

$$(2) d_j \leq w(\mathbf{G}_{j1})$$

\mathbf{G}_{j1} 是 \mathbf{G}_j 矩阵的第一行, 因为若令 $\mathbf{m}_0 = 1, \mathbf{m}_1 = \mathbf{m}_2 = \dots = \mathbf{m}_j = 0$, 则

$$d_j = \min_{\mathbf{m}_0 \neq 0} w(\mathbf{M}_j \mathbf{G}_j) = w(\mathbf{M}_j \mathbf{G}_{j1}) = w(\mathbf{G}_{j1}) \quad (10.5.4)$$

$$(3) d_j \leq d_{j+1}$$

说明列距离是非降的。这是因为 $w(\mathbf{G}_{(j+1)1}) \geq w(\mathbf{G}_{j1})$ 之故。

下面定义反馈译码距离与定译码距离。

定义 10.5.4 (n_0, k_0, m) 卷积码的反馈译码距离 d_{FD} 定义为

$$d_{FD} = \min_{\mathbf{m}_0 \neq \mathbf{m}'_0} d(\mathbf{M}_m \mathbf{G}_m, \mathbf{M}'_m \mathbf{G}'_m) \quad (10.5.5)$$

由此可知, d_{FD} 就是 m 阶列距离, 即 $d_{FD} = d_m$ 。

由定理 10.5.1 与定理 10.5.2 可知, 若 (n_0, k_0, m) 的反馈译码距离为 d_{FD} , 则一定能用反馈译码方法, 纠正长为 $(m+1)n_0$ 长码序列中 $(d_{FD}-1)/2$ 个错误。

定义 10.5.5 (n_0, k_0, m) 卷积码的定译码最小距离 d_{DD} 定义为

$$d_{DD} = \min_{\mathbf{m}_0 \neq \mathbf{m}'_0} d(\mathbf{M}_{2m} \mathbf{G}_{2m}, \mathbf{M}'_{2m} \mathbf{G}'_{2m}) \quad (10.5.6)$$

式中:

$$\mathbf{M}_{2m} = (\mathbf{m}_0, \mathbf{m}_1, \dots, \mathbf{m}_{2m})$$

$$\mathbf{M}'_{2m} = (\mathbf{m}'_0, \mathbf{m}'_1, \dots, \mathbf{m}'_{2m})$$

都是 $2m+1$ 段长的信息序列。

由此可知, d_{DD} 就是 $2m$ 阶列距离。由于定译码时对伴随式不进行修正, 因此某一段上的错误图样必然影响其前后 m 段的译码, 所以在定译码情况下, 译码约束度为 $2m+1$ 。而在反馈译码下, 由于对伴随式进行了修正, 消去了该段错误对后 m 段码元的影响, 因此与以后各段码元无关, 故译码约束度为 $m+1$, 这是反馈译码与定译码最本质的不同。

若 (n_0, k_0, m) 卷积码的定译码距离为 d_{DD} , 则由定理 10.5.1 和定理 10.5.2 可知, 一定能用定译码方法, 纠正长为 $N_D n_0 = (2m+1)n_0$ 个码元序列以内 $\lfloor (d_{DD}-1)/2 \rfloor$ 个错误。

从式(10.5.5)与式(10.5.6)不难看出, 若译码约束度相等, 则对任何 (n_0, k_0, m) 卷积码有

$$d_{DD} \leq d_{FD} \quad (10.5.7)$$

这是因为由式(10.5.6)求极小的集合包含了式(10.5.5)求极小的集合。

式(10.5.7)说明, 在同样编码约束长度下, 定译码的纠错能力比反馈译码要差。

定义 10.5.6 (n_0, k_0, m) 卷积码的 j 阶行距离 d_{rj} 定义为

$$d_{rj} = \min_{\mathbf{m}_0 \neq \mathbf{m}'_0} d((\mathbf{M}_j, \mathbf{0}^\infty) \mathbf{G}_\infty, (\mathbf{M}'_j, \mathbf{0}^\infty) \mathbf{G}_\infty) \quad (10.5.8)$$

即卷积码的 j 阶行距离就是由长为 $j+1$ 段的所有信息序列产生的半无限长码序列之间的最小汉明距离。而 j 阶列距离是 $j+1$ 段长截短码树上定义的最小汉明距离。也就是说，列距离是 \mathbf{G}_∞ 矩阵中，前 $j+1$ 段的行组合之间最小的汉明距离。而行距离是 \mathbf{G}_∞ 矩阵中，前 $(j+1)k_0$ 行所组合的码序列之间的最小汉明距离。

行距离有如下性质：

$$(1) d_{rj} = \min_{\mathbf{m}_0 \neq \mathbf{0}} w\{(\mathbf{M}_j, \mathbf{0}^\infty) \mathbf{G}_\infty\} \quad (10.5.9)$$

即 j 阶行距离等于 \mathbf{G}_∞ 矩阵中前 $(j+1)k_0$ 行所组成的码序列中最小的汉明重量。

$$(2) d_{rj} \geq d_{rj+1} \quad (10.5.10)$$

即行距离是非增的。

$$(3) d_{rj} \leq \min_{1 \leq i \leq k_0} w(\mathbf{g}(i)) \quad (10.5.11)$$

这意味着 j 阶行距离，是不大于 k_0 个生成元中重量最轻生成元的重量。

有了行距离与列距离的概念之后，就可以定义卷积码的自由距离。

定义 10.5.7 (n_0, k_0, m) 卷积码的自由距离 d_f 定义为：

$$d_f = d_{r\infty} \quad (10.5.12)$$

或

$$d_f = \min_{\mathbf{m}_0 \neq \mathbf{0}} w(\mathbf{M}_\infty \mathbf{G}_\infty) \quad (10.5.13)$$

也就是说，卷积码的自由距离，是定义在整个码树上的，所有半无限长码序列之间的最小汉明距离或最小汉明重量。但是可以证明，在求 d_f 时，不必计算半无限长码序列之间的汉明距离或者计算 ∞ 阶行距离，而只要在有限段长的截段码树上计算就够了。由式(10.5.5)，式(10.5.11)和式(10.5.13)不难证明 d_f 在下述范围内：

$$d_{FD} \leq d_f \leq \min_{1 \leq i \leq k_0} w(\mathbf{g}(i)) \quad (10.5.14)$$

由此看到，自由距离是定义在整个码树上的，它比 d_{\min} 要大。该例中的 $(2, 1, 2)$ 码的 $d_f = 5$ ，而 $d_{\min} = 3$ ，因此若用译码约束长度很长的概率译码，它能纠正相邻 $m+1$ 段内的两个错误，比一般的代数译码的纠错能力要强。此外，与分组码不同的是，即使有相同的 n_0 、 k_0 、 m ，但系统卷积码与非系统卷积码的距离特性常常不一样，在某些码中非系统码的 d_f 要比系统码大。

* § 10.6 卷积码的状态图表示和码的重量分布

除了用树图表示 (n_0, k_0, m) 卷积码以外，也可用编码器状态图表示卷积码编码器的工作过程，并且可求出卷积码的重量分布。

一、卷积码的状态图

图 10-18 就是例 10.1 中的 $(2, 1, 2)$ 卷积码编码器状态图。编码器寄存器中任一时刻的存数称为编码器的一个状态，以 s_i 表示。该例中，编码贮存 $m=2$ ， $k_0=1$ ，编码器由两级移存器组成。因此，移存器中的存数只有 4 种可能：00, 10, 01 和 11，相应于编码器有 4 个

状态： s_0 , s_1 , s_2 和 s_3 。随着信息序列不断送入，编码器就不断地从一个状态转移到另一状态，并输出相应的码序列。如果把这种状态变化画一流程图，则此图就表征了编码器的工作特征，这种图就称为编码器的状态图，如图 10-18。

由该图看出：若编码器的初始状态处于 s_0 ，输入 1 信息元时，编码器从 s_0 转移到 s_1 状态，并输出子组 11；若输入 0 信息元，则仍停留在 s_0 状态，输出子组 00，如此等等。随着信息元的不断送入，编码器的状态在不断地转移，并输出相应的分支(子组)，组成对应于输入信息序列的一个码序列。图中，实线表示 0 输入，虚线表示 1 输入时的状态转移。

例如，信息组 $M = (1100)$, $M(D) = 1 + D$ ，则由例 10.1 可知，输出的码序列

$$\begin{aligned} C(D) &= M(D)G(D) = (1 + D)[1 + D + D^2, 1 + D^2] \\ &= (11) + (01)D + (01)D^2 + (11)D^3 \end{aligned}$$

相应地 $C = (11, 01, 01, 11)$ 。在状态图中编码器的状态变化为： $s_0 \xrightarrow{11} s_1 \xrightarrow{01} s_3 \xrightarrow{01} s_2 \xrightarrow{11} s_0$ 。

由上例可知，编码器编出的码序列其实就是在信息序列 M 控制下，从状态 s_0 开始再回到 s_0 状态时，状态转移所经过的路径。使这种从 s_0 状态出发回到 s_0 状态时，编码器所经状态后得到的码序列为**结尾码字**。这种结尾码字，要求输完信息序列后，还应再向编码器输入 mk_0 个全 0 信息元，以便使编码器回到全 0 状态 s_0 。

卷积码的自由距离除了用式(10.5.12)和式(10.5.13)定义外，也可用上述的编码器状态图来定义。

定义 10.6.1 卷积码的自由距离 d_f 等于编码器从 s_0 状态出发，又回到该状态时，所有可能非全 0 路径重量的最小值。

如上例中，从 s_0 出发回到 s_0 状态的所有可能非全零路径有很多，但重量最小的一条非全 0 路径是： $s_0 \xrightarrow{11} s_1 \xrightarrow{10} s_2 \xrightarrow{11} s_0$ ，其输出结尾码字的重量为 5，因此该码的 $d_f = 5$ 。

二、生成函数和重量分布

状态图除了可以方便地说明编码器的工作过程，计算码的 d_f 以外，还可以由状态图得到卷积码的重量分布。为此先作如下一些定义。

分支增益 x^i ：是分支中也就是子码中非 0 元素的个数是 i 。如 11 为 x^2 , 10 为 x^1 , 00 为 x^0 等，它说明了每个分支的重量。

路径增益 G ：是路径所经过的所有分支增益的乘积，它说明了相应路径(码字)的重量。如上例中 $s_0 \xrightarrow{11} s_1 \xrightarrow{10} s_2 \xrightarrow{11} s_0$ ，它的分支增益相应地为 x^2 , x^1 , x^2 ，因而路径增益 $G = x^2 x^1 x^2 = x^5$ ，相应的码字重量 $w(C) = 5$ 。

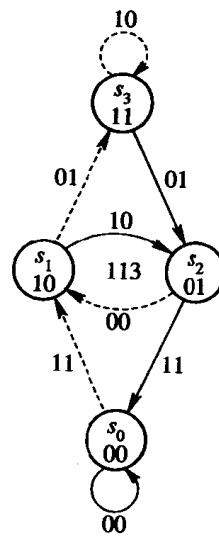


图 10-18 (2,1,2)卷积码编码器状态图

前向路径：从 s_0 回到 s_0 的所有经历的状态中，中间没有经历某状态两次的路径，称为前向路径。如上例中 $s_0 \xrightarrow{11} s_1 \xrightarrow{10} s_2 \xrightarrow{11} s_0$ ($s_0 s_1 s_2 s_0$) 是一条前向路径。用 G_i 表示第 i 条前向路径增益，如该例中前向路径增益 $G_1 = x^5$ 。

环路：从任一状态开始回到该状态的一条闭合路径（不包括前向路径），且中间不经过任何状态两次，如 $s_2 s_1 s_2$ 。

非交环路集合：在环路集合中，没有两个环路有同一状态，如 $s_2 s_1 s_2, s_3 s_3$ 等。

L_1 表示所有环路的集合，如 $\{s_2 s_1 s_2, s_3 s_3, s_1 s_3 s_2 s_1\}$ 。用 L_{1i} 表示第 i 条环路的增益，它是组成环路的各分支增益的乘积，如 $s_2 s_1 s_2$ 的 $L_{11} = x$ 。

L_2 表示所有两个非交环路对的集合，如 $\{s_2 s_1 s_2, s_3 s_3\}$ 就是一对非交环路对。用 L_{2i} 表示第 i 对非交环路增益，它是这对环路增益的积，如该例中 $L_{21} = x x = x^2$ 。

一般用 L_m 表示所有 m 个非交环路组的集合。 L_{mi} 表示第 i 组 m 个非交环路增益的积，它是该组 m 个环路增益的乘积。令

$$\Delta = 1 - \sum_i L_{1i} + \sum_j L_{2j} - \sum_l L_{3l} + \cdots + (-1)^m \sum_k L_{mk} \quad (10.6.1)$$

应用梅塞(Mason)的状态拓扑特性^[6]，可以用以下的生成函数

$$T(x) = \frac{\sum_i G_i \Delta_i}{\Delta} = \sum_{i=0}^{\infty} A_i x^i \quad (10.6.2)$$

来计算码的重量分布。式中， Δ_i 是与第 i 条前向路径不交的图的相应部分的 Δ 值。 A_i 是重量为 i 的码字数，可知它类似于分组码的重量分布多项式。

为了便于计算码的重量分布，把状态图的全 0 状态 s_0 分成两部分，得到编码器的修正状态图。图 10-18 的修正状态图如图 10-19 所示，由该图看到有两条前向路径： $s_0 s_1 s_2 s_0$ 和 $s_0 s_1 s_3 s_2 s_0$ ，它们的前向路径增益分别是 $G_1 = x^5$ 和 $G_2 = x^6$ 。

由该图容易得到

$$L_1: \{s_2 s_1 s_2, s_3 s_3, s_1 s_3 s_2 s_1\}$$

相应的 L_{1i} 分别为 $L_{11} = x$, $L_{12} = x$, $L_{13} = x^2$ 。

$$L_2: \{s_2 s_1 s_2, s_3 s_3\}$$

相应的 $L_{21} = L_{11} L_{12} = x^2$ 。把上述这些值代入式(10.6.1)可得

$$\begin{aligned} \Delta &= 1 - \sum_i L_{1i} + \sum_j L_{2j} \\ &= 1 - (x + x + x^2) + x^2 = 1 - 2x \end{aligned}$$

第一条前向路径为 $s_0 s_1 s_2 s_0$ ，与它不相交的图中的其它环路仅为 $s_3 s_3$ ，所以

$$\Delta_1 = 1 - x$$

第二条前向路径为 $s_0 s_1 s_3 s_2 s_0$ ，所有状态都经历，因此 $\Delta_2 = 1$ 。把上述这些值代入式(10.6.2)可得

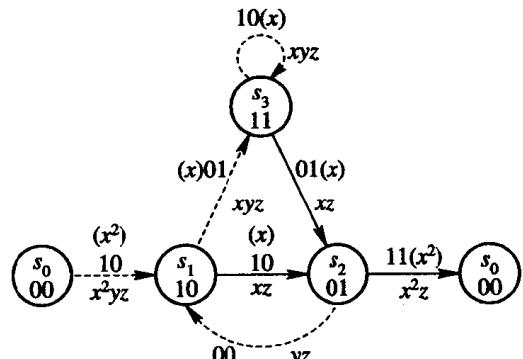


图 10-19 对于图 10-18 的修正状态图

$$\begin{aligned}
T(x) &= \frac{G_1\Delta_1 + G_2\Delta_2}{\Delta} = \frac{x^5(1-x) + x^6}{1-2x} \\
&= \frac{x^5}{1-2x} \\
&= x^5(1+2x+4x^2+\cdots+2^ix^i+\cdots) \\
&= \sum_{i=0}^{\infty} 2^i x^{i+5}
\end{aligned}$$

说明该码有 1 个重量为 5 的码字，有 2 个重量为 6 的码字，有 4 个重量为 7 的码字等。

当然还可以利用修正状态图更细致地描述码的重量分布和距离特性。如重量为 5 的码字有几个码段组成，其中信息序列的重量为多少等。为此可把修正状态图加以标号。如图 10-19 中， $s_0 \rightarrow s_1$ ，有一个输出分支(11)，它的输入信息元是 1，重量 $y^j = y^1$ ，输出分支重量或增益为 x^2 ，因此用 x^2yz 表示。其中 z 表示一个分支， z^i 表示有 i 个分支。由此可得到修正的生成函数：

$$T(x, y, z) = \frac{\sum_i G_i(xyz)\Delta_i(xyz)}{\Delta(xyz)} = \sum_{i=0}^{\infty} A_{ijl}x^i y^j z^l \quad (10.6.3)$$

仍以图 10-19 为例，可得：

$$\Delta(xyz) = 1 - (xyz^2 + xyz + x^2y^2z^3) + x^2y^2z^3 = 1 - xyz(1+z)$$

$$\Delta_1(xyz) = 1 - xyz$$

$$\Delta_2(xyz) = 1$$

$$G_1(xyz) = (x^2yz)(xz)(x^2z) = x^5yz^3$$

$$G_2(xyz) = (x^2yz)(xyz)(xz)(x^2z) = x^6y^2z^4$$

把上述这些值代入式(10.6.3)可得

$$\begin{aligned}
T(xyz) &= \frac{x^5yz^3(1-xyz) + 1(x^6y^2z^4)}{1-xyz(1+z)} = \frac{x^5yz^3}{1-xyz(1+z)} \\
&= x^5yz^3 + x^6y^2z^4(1+z) + x^7y^3z^5(1+z^2) + \cdots \\
&\quad + x^{5+k}y^{1+k}z^{3+k}(1+z)^k + \cdots
\end{aligned} \quad (10.6.4)$$

该生成函数表示重量为 5 的一个码序列由 3 个分支组成，并由重量为 1 的信息序列输入编码器时产生的；重量为 6 的码序列由 4 个分支组成，且信息序列的重量为 2；另一个重量为 6 的码序列由 5 个分支组成，信息序列的重量也为 2，如此等等。可知该码的最小自由距离 d_f 是 5。

由以上讨论可以看到，用码的生成函数计算码的重量分布，当编码贮存 m 超过 4 或 5 时，计算非常复杂而难于应用。但是，作为了解码的代数结构，计算码的性能限时，生成函数不失是一个好的工具。

表 10-2 给出了用计算机搜索得到的某些最佳(在同样的 n_0, k_0 下，有最大的 d_f ，且非恶性的)非系统卷积码的重量分布^[5]。表中， B_i 表示重量为 i 的码字中非 0 信息元的数目，也就是式(10.6.4)中 y 的次数。

表 10-2 部分(n_0 , k_0 , m)最佳非系统卷积码的重量分布

| R | m | 子生成元 | d_f | 重量分布 A_i | | | | 重量分布 B_i | | | |
|-----|-----|-----------------------------|-------|------------|------------|------------|------------|------------|------------|------------|------------|
| | | | | A_{df} | A_{df+1} | A_{df+2} | A_{df+3} | B_{df} | B_{df+1} | B_{df+2} | B_{df+3} |
| 1/2 | 2 | 7,5 | 5 | 1 | 2 | 4 | 8 | 1 | 4 | 12 | 38 |
| | 3 | 15,17 | 6 | 1 | 3 | 5 | 11 | 2 | 7 | 18 | 49 |
| | 4 | 23,35 | 7 | 2 | 3 | 4 | 16 | 4 | 12 | 20 | 72 |
| | 5 | 53,75 | 8 | 1 | 8 | 7 | 12 | 2 | 36 | 32 | 62 |
| | 6 | 133,171 | 10 | 11 | 0 | 38 | 0 | 36 | 0 | 211 | 0 |
| | 7 | 247,371 | 10 | 1 | 6 | 12 | 26 | 2 | 22 | 60 | 148 |
| | 8 | 561,753 | 12 | 11 | 0 | 50 | 0 | 33 | 0 | 281 | 0 |
| | 9 | 1 167,1 545 | 12 | 2 | 8 | 15 | 35 | 14 | 26 | 74 | 257 |
| | 10 | 2 335,3 661 | 14 | 21 | 0 | 74 | 0 | 94 | 0 | 463 | 0 |
| | 11 | 4 355,5 723 | 15 | 16 | 31 | 44 | 129 | 76 | 180 | 374 | 1 142 |
| | 12 | 10 533,17 661 | 16 | 33 | 0 | 111 | 0 | 152 | 0 | 971 | 0 |
| | 13 | 21 675,27 123 | 16 | 4 | 17 | 35 | 76 | 22 | 99 | 218 | 608 |
| 1/3 | 2 | 5,7,7 | 8 | 2 | 0 | 5 | 0 | 3 | 0 | 15 | 0 |
| | 3 | 13,15,17 | 10 | 3 | 0 | 2 | 0 | 6 | 0 | 6 | 0 |
| | 4 | 25,33,37 | 12 | 5 | 0 | 3 | 0 | 12 | 0 | 12 | 0 |
| | 5 | 47,53,75 | 13 | 1 | 3 | 6 | 4 | 1 | 8 | 26 | 20 |
| | 6 | 133,145,175 | 15 | 3 | 5 | 5 | 6 | 11 | 16 | 19 | 28 |
| | 7 | 225,331,367 | 16 | 1 | 0 | 8 | 0 | 1 | 0 | 24 | 0 |
| | 8 | 557,663,711 | 18 | 5 | 0 | 7 | 0 | 11 | 0 | 32 | 0 |
| | 9 | 1117,1 365,1 633 | 20 | 8 | 0 | 18 | 0 | 29 | 0 | 91 | 0 |
| 1/3 | 10 | 2 353,2 671,3 175 | 22 | 14 | 0 | 18 | 0 | 53 | 0 | 92 | 0 |
| | 11 | 4 767,5 723,6 265 | 24 | 21 | 0 | 9 | 0 | 80 | 0 | 58 | 0 |
| | 12 | 10 533,10 675,17 661 | 24 | 10 | 0 | 14 | 0 | 27 | 0 | 74 | 0 |
| | 13 | 21 645,35 661,37 133 | 26 | 12 | 0 | 32 | 0 | 41 | 0 | 165 | 0 |
| | 2 | 5,7,7,7 | 10 | 1 | 1 | 1 | 3 | 2 | 1 | 4 | 9 |
| 1/4 | 3 | 13,15,15,17 | 13 | 2 | 1 | 0 | 3 | 4 | 2 | 0 | 10 |
| | 4 | 25,27,33,37 | 16 | 4 | 0 | 2 | 0 | 8 | 0 | 7 | 0 |
| | 5 | 53,67,71,75 | 18 | 3 | 0 | 5 | 0 | 6 | 0 | 17 | 0 |
| | 6 | 135,135,147,163 | 20 | 10 | 0 | 0 | 0 | 37 | 0 | 0 | 0 |
| | 7 | 235,275,313,357 | 22 | 1 | 4 | 3 | 2 | 2 | 10 | 10 | 8 |
| | 8 | 363,535,733,745 | 24 | 2 | 0 | 6 | 0 | 4 | 0 | 22 | 0 |
| | 9 | 1 117,1 365,1 633,1 653 | 27 | 4 | 4 | 4 | 8 | 10 | 12 | 18 | 44 |
| | 10 | 2 327,2 353,2 671,3 175 | 29 | 5 | 6 | 4 | 6 | 13 | 24 | 18 | 22 |
| | 11 | 4 767,5 723,6 265,7 455 | 32 | 14 | 0 | 10 | 0 | 49 | 0 | 40 | 0 |
| | 12 | 11 145,12 477,15 573,16 727 | 33 | 5 | 5 | 3 | 9 | 19 | 16 | 15 | 46 |
| | 13 | 21 113,23 175,35 527,35 537 | 36 | 19 | 0 | 16 | 0 | 74 | 0 | 80 | 0 |

习 题

1. 已知(2, 1, 3)码的子生成元 $\mathbf{g}^{(1, 1)} = (1101)$, $\mathbf{g}^{(1, 2)} = (1111)$ 。

(1) 求出该码的 $\mathbf{G}(D)$ 、 $\mathbf{H}(D)$ 矩阵, 以及 \mathbf{G}_{∞} 和 \mathbf{H}_{∞} 矩阵。

(2) 画出该码的编码器。

(3) 求出相应于信息序列 $\mathbf{M} = (11001)$ 的码序列。

(4) 此码是否是系统码?

2. 已知某一卷积码的编码器如图 10-20 所示。

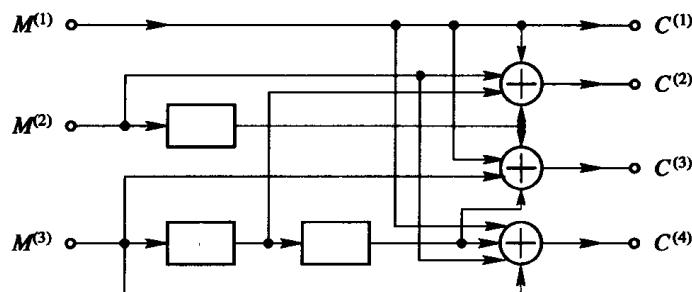


图 10-20 (4, 3, 2)卷积码编码器

(1) 找出该码的子生成元。

(2) 写出该码的 $\mathbf{G}(D)$ 和 $\mathbf{H}(D)$ 。

(3) 求出相应于信息序列 $\mathbf{M} = (110, 011, 101)$ 的编码序列。

3. 已知(3, 2, 1)码的子生成元为: $\mathbf{g}^{(1, 1)}(D) = 1 + D$, $\mathbf{g}^{(1, 2)}(D) = D$, $\mathbf{g}^{(1, 3)}(D) = 1 + D$, $\mathbf{g}^{(2, 1)}(D) = D$, $\mathbf{g}^{(2, 2)}(D) = 1$, $\mathbf{g}^{(2, 3)}(D) = 1$ 。

(1) 画出该码的编码器。

(2) 写出 $\mathbf{G}(D)$ 、 $\mathbf{H}(D)$ 。

(3) 求出 $\mathbf{C}^{(1)}(D)$ 、 $\mathbf{C}^{(2)}(D)$ 和 $\mathbf{C}^{(3)}(D)$, 已知 $\mathbf{M}(D) = [1 + D + D^3, 1 + D^2 + D^3]$, 并写出 $\mathbf{C}(D)$ 。

4. 设(3, 2, 3)系统码的子生成元: $\mathbf{g}^{(1, 3)}(D) = 1 + D^2 + D^3$, $\mathbf{g}^{(2, 3)}(D) = 1 + D + D^3$,

(1) 此码是恶性码吗? 为什么?

(2) 画出该码的编码器和对偶码的编码器。

(3) 画出有 4 个分支长的树图。

(4) 求出此码的最小距离 d_m , 反馈译码距离 d_{FD} 和定译码距离。

(5) 求出此码的自由距离。

5. 已知 $\mathbf{G}(D) = [1 + D + D^2, 1 + D + D^2 + D^3]$,

(1) 求出该码的 $\mathbf{H}(D)$ 和对偶码的 $\mathbf{G}_{\perp}(D)$ 和 $\mathbf{H}_{\perp}(D)$ 。

(2) 此码是恶性码吗?

(3) 此码是快检码吗? 找出有最小迟延前馈逆的矩阵 $\mathbf{G}^{-1}(D)$ 。

(4) 画出由 $\mathbf{C}(D)$ 求 $\mathbf{M}(D)$ 的电路, 且要求级数最少。

(5) 画出该码的状态转移图。

6. 已知(3, 1, 2)码的子生成元是 $\mathbf{g}^{(1, 1)}(D) = 1 + D$, $\mathbf{g}^{(1, 2)}(D) = 1 + D^2$, $\mathbf{g}^{(1, 3)}(D)$

$$= 1 + D + D^2,$$

- (1) 求出该码的 $\mathbf{G}(D)$ 和 $\mathbf{H}(D)$ 。
 - (2) 画出该码的编码电路。
 - (3) 该码是否是恶性码？找出有最小迟延前馈逆的矩阵 $\mathbf{G}^{-1}(D)$ ，它是快检码吗？
7. 求第 6 题码的最小汉明距离 d_m ，反馈译码距离 d_{FD} 和定译码距离及最小自由距离。
8. 画出第 6 题编码器的状态转移图，修正状态转移图，计算它的生成函数 $T(x)$ 和 $T(xyz)$ 。

参 考 文 献

- [1] S. Lin, D. J. Costello, Jr., Error Control Coding—Fundamentals and Applications, 1984. 中译本：王育民、王新梅译，《差错控制编码：基础和应用》，人民邮电出版社，1986.
- [2] 王新梅：《纠错码与差错控制》，人民邮电出版社，1989.
- [3] M. Y. Rhee, Error - Correcting Coding Theory, McGraw - Hill, 1990.
- [4] J. L. Massey and M. K. Sain, “Inverses of linear sequential circuits”, IEEE Trans. on Comput., pp. 330—337, 1968.
- [5] J. Conan, “The weight spectra of some short low - rate convolutional codes”, IEEE Trans. on Commun., No. 9, pp. 1050—1053, 1984.
- [6] S. Mason and H. Zimmermann, Electronic Circuits, Signals, and Systems, Wiley, New York, 1960.

第十一章 纠随机错误与纠突发 错误卷积码

纠随机错误卷积码分为两大类：一类是用代数方法译码的码，如怀纳—阿什(Wyner - Ash)码(WA 码)、自正交码和可正交码等；另一类是用概率方法译码的码。

在一般代数译码中，通常译码约束长度等于编码约束长度。因此若要求(n_0, k_0, m)卷积码在连续的 $m+1$ 段以内，能纠正 $t \leq \lfloor (d-1)/2 \rfloor$ 个随机错误，则由定理 10.5.1 和定理 10.5.2 知，要求码的最小汉明距离 $d \geq 2t+1$ 。这相当于要求初始截段码的 H 阵中任意 $d-1$ 列线性无关，且其中至少一列在第 0 段中选取(即要求码字的第 0 段为非全 0)。这种要求与第三章中定理 3.3.1 所叙述的一致的。根据这一要求，1963 年怀纳—阿什^[1]仿照汉明码的构造方法，构造了一类纠单个错误的 $(2^m, 2^m-1, m)$ WA 码。自此以后，很多学者还根据循环码的构造原理与方法，得到了一批由循环码导出的卷积码^[2]。但是，在实际中用得最为普遍的是利用大数逻辑译码方法译码的自正交码与可正交码，这类码不仅构造容易，码类较多，而且译码器非常简单，并在不长的译码约束长度内(几十位)能提供 $1\sim 2$ dB 的编码增益，因此深受工程技术人员的欢迎，在各种数字通信系统中得到了普遍的应用。

利用概率译码方法译码的卷积码，通常能获得最佳或准最佳译码方法(最大似然译码)所能获得的性能，对这类码要求有大的自由距离 d_f 和其它性质，这通常用计算机搜索寻找得到。

纠突发错误卷积码也可分为两类：一类是纠正固定类型突发和突发长度的 **B₁型**和**B₂型**码；另一类是能根据信道错误情况，自动改变纠错类型的**自适应码**。但无论哪一类码，所用的译码方法，基本上都是利用大数逻辑译码方法。

本章先介绍用代数方法译码的大数逻辑可译码(自正交码与可正交码)，然后介绍纠突发错误卷积码，至于概率译码将在下一章讨论。

§ 11.1 卷积码的大数逻辑译码

大数逻辑译码是门限译码的一种，1963 年由梅西最早提出。1967 年鲁滨逊(Robinson)等利用差集三角构造了一批可用大数逻辑译码的自正交码，用试凑方法构造了一些可正交码^[1]。对卷积码来说，大数逻辑译码是卷积码代数译码中最主要的译码方法。它的译码原理与循环码的大数逻辑译码原理相同。这一节主要讨论系统码形式的自正交码与可正交码的大数逻辑译码方法，下节介绍非系统码的大数逻辑译码方法。

在 (n_0, k_0, m) 系统卷积码的任一码字中，任意连续 $m+1$ 码段中的码元之间都必须满足由初始截短码的校验矩阵

$$\mathbf{H} = \begin{bmatrix} \mathbf{P}_0^T \mathbf{I}_{n_0 - k_0} \\ \vdots \\ \mathbf{P}_m^T \mathbf{0} & \mathbf{P}_{m-1}^T \mathbf{0} & \cdots & \mathbf{P}_0^T \mathbf{I}_{n_0 - k_0} \end{bmatrix}$$

所确定的校验关系。因此，若对第 0 子组的 k_0 个码元位能组成 J 个正交一致校验和式的话，则可用大数逻辑译码方法纠正任意连续 $m+1$ 段内， $t \leq \lfloor J/2 \rfloor$ 个随机错误。可知，大数逻辑译码的译码约束度大都是 $m+1$ ，且一般应用反馈译码。

一、自正交码

先通过下面的例子说明自正交的含义。设有 $(2, 1, 6)$ 系统卷积码，其子生成元为：
 $\mathbf{g}^{(1,1)} = (1000000)$, $\mathbf{g}^{(1,1)}(D) = 1$; $\mathbf{g}^{(1,2)} = (1010011)$, $\mathbf{g}^{(1,2)}(D) = 1 + D^2 + D^5 + D^6$ 。相应的校验矩阵

$$\mathbf{H} = \begin{bmatrix} 11 \\ 00 \quad 11 \\ 10 \quad 00 \quad 11 \\ 00 \quad 10 \quad 00 \quad 11 \\ 00 \quad 00 \quad 10 \quad 00 \quad 11 \\ 10 \quad 00 \quad 00 \quad 10 \quad 00 \quad 11 \\ 10 \quad 10 \quad 00 \quad 00 \quad 10 \quad 00 \quad 11 \end{bmatrix}$$

设错误图样

$$\mathbf{E} = (e_{01}e_{02}, e_{11}e_{12}, e_{21}e_{22}, e_{31}e_{32}, e_{41}e_{42}, e_{51}e_{52}, e_{61}e_{62})$$

则伴随式为

$$\mathbf{S} = \mathbf{E} \cdot \mathbf{H}^T = (s_{01}, s_{11}, s_{21}, s_{31}, s_{41}, s_{51}, s_{61})$$

式中

$$\begin{aligned} s_{01} &= e_{01} + e_{02} \\ s_{11} &= e_{11} + e_{12} \\ s_{21} &= e_{01} + e_{21} + e_{22} \\ s_{31} &= e_{11} + e_{31} + e_{32} \\ s_{41} &= e_{21} + e_{41} + e_{42} \\ s_{51} &= e_{01} + e_{31} + e_{51} + e_{52} \\ s_{61} &= e_{01} + e_{11} + e_{41} + e_{51} + e_{62} \end{aligned} \tag{11.1.1}$$

由此 7 个方程看出，在 s_{01}, s_{21}, s_{51} 和 s_{61} 四个方程中，除了 e_{01} 位以外，其它码元位至多只出现一次，从而组成了 4 个对 e_{01} 码元位正交的一致校验和式。所以， e_{01} 码元位上的错误值，完全可由 s_{01}, s_{21}, s_{51} 和 s_{61} 的值确定，而它们的值则完全由 \mathbf{H} 矩阵中第 0、2、5、6 行的校验关系决定。

定义 11.1.1 任一个 (n_0, k_0, m) 系统卷积码，若能由 \mathbf{H} 矩阵中的 J_i 行（用 \mathbf{S} 中的 J_i 个分量表示），直接（不经线性组合）组成对 e_{0i} ($i = 1, 2, \dots, k_0$)，若为非系统码 $i = 1, 2, \dots, n_0$)。码元位正交的 J_i 个正交校验和式，则称此码为自正交系统卷积码；若码的最小距离

$d_{FD}=J+1$, 则称为完备自正交码。

由定义可知, 自正交码能纠正译码约束长度 $(m+1)n_0$ 个码元内, 任意 $i \leq \lfloor J/2 \rfloor$ 个错误, J 是 J_1, J_2, \dots, J_{k_0} (或 J_{n_0})中的最小者。

上例中的 $J=J_1=4$, 而 $d_{FD}=5=J+1$, 所以是完备自正交码, 能用反馈译码纠正14个连续码元内的2个随机错误。从式(11.1.1)看到, 若一个错误在 e_{01} 位上, 另一个错误在任意其它位上, 则 s_{01}, s_{21}, s_{51} 和 s_{61} 伴随式分量中至少有3个1; 反之, 若2个错误均不在 e_{01} 位上, 则此4个伴随式分量中至多有2个1。所以根据这4个伴随式分量中1的多少, 判断 e_{01} 码元位是否有错。

一般而言, 自正交码的 $k_0=1$ 或 $n_0-k_0=1$, 并且若 $(n_0, 1, m)$ 是自正交码的话, 那么它的 (n_0, n_0-1, m) 对偶码也是自正交码。如上例中的对偶码也是一个 $(2, 1, 6)$ 码, 就是它自身, 这是一个自对偶码。事实上, 所有码率 $R=1/2$ 的自正交码都是自对偶的。

为了使用方便, 表11-1~表11-4中列出了 n_0 取2、3、4和5时自正交系统码的参数^[1](至于更高码率和更长约束度的自正交码表可参阅[1,3])。表中, m 是编码存储, R 是码率, t_{ML} 是用反馈译码所能纠正的错误数目。子生成元列下面的数字, 表示子生成元多项式中系数取1的次数。例如表11-1中, $t_{ML}=1, m=1$, 子生成元列下面的数字是 $(0, 1)$, 表示一个子生成元为 $\mathbf{g}^{(1,2)}(D)=1+D$ 。

表 11-1 $n_0=2$ 自正交系统卷积码

| $R=1/2$ | | $\mathbf{g}^{(1,2)}$ |
|----------|-----|---|
| t_{ML} | m | |
| 1 | 1 | {0,1} |
| 2 | 6 | {0,2,5,6} |
| 3 | 17 | {0,2,7,13,16,17} |
| 4 | 35 | {0,7,10,16,18,30,31,35} |
| 5 | 55 | {0,2,14,21,29,32,45,49,54,55} |
| 6 | 85 | {0,2,6,24,29,40,43,55,68,75,76,85} |
| 7 | 127 | {0,5,28,38,41,49,50,68,75,92,107,121,123,127} |
| 8 | 179 | {0,6,19,40,58,67,78,83,109,132,133,162,165,169,177,179} |
| 9 | 216 | {0,2,10,22,53,56,82,83,89,98,130,148,153,167,188,192,205,216} |
| 10 | 283 | {0,24,30,43,55,71,75,89,104,125,127,162,167,189,206,215,272,275,282,283} |
| 11 | 358 | {0,3,16,45,50,51,65,104,125,142,182,206,210,218,228,237,289,300,326,333,356,358} |
| 12 | 425 | {0,22,41,57,72,93,99,139,147,153,197,200,214,253,263,265,276,283,308,367,368,372,396,425} |

表 11-2 $n_0=3$ 自正交系统卷积码

| $R=1/3$ | t_{ML} | m | $\mathbf{g}^{(1,2)}$ | $\mathbf{g}^{(1,3)}$ |
|---------|----------|-----|---|---|
| $R=2/3$ | t_{ML} | m | $\mathbf{g}^{(1,3)}$ | $\mathbf{g}^{(2,3)}$ |
| 1 | 2 | 2 | {0,1} | {0,2} |
| 2 | 4 | 13 | {0,8,9,12} | {0,6,11,13} |
| 3 | 6 | 40 | {0,2,6,24,29,40} | {0,3,15,28,35,36} |
| 4 | 8 | 86 | {0,1,27,30,61,73,81,83} | {0,18,23,37,58,62,75,86} |
| 5 | 10 | 130 | {0,1,6,25,32,72,100,108,120,130} | {0,23,39,57,60,74,101,103,112,116} |
| 6 | 12 | 195 | {0,17,46,50,52,66,88,125,150, 165,168,195} | {0,26,34,47,57,58,112,121, 140,181,188,193} |
| 7 | 14 | 288 | {0,2,7,42,45,117,163,185,195, 216,229,246,255,279} | {0,8,12,27,28,64,113,131,154, 160,208,219,233,288} |

表 11-3 $n_0=4$ 自正交系统卷积码

| $R=1/4$ | t_{ML} | m | $\mathbf{g}^{(1,2)}$ | $\mathbf{g}^{(1,3)}$ | $\mathbf{g}^{(1,4)}$ |
|---------|----------|-----|---|---------------------------------------|---------------------------------------|
| $R=3/4$ | t_{ML} | m | $\mathbf{g}^{(1,4)}$ | $\mathbf{g}^{(2,4)}$ | $\mathbf{g}^{(3,4)}$ |
| 1 | 3 | 3 | {0,1} | {0,2} | {0,3} |
| 2 | 6 | 19 | {0,3,15,19,} | {0,8,17,18} | {0,6,11,13} |
| 3 | 9 | 67 | {0,5,15,34,35,42} | {0,31,33,44,47,56} | {0,17,21,43,49,67} |
| 4 | 12 | 129 | {0,9,33,37,38,97,122, 129} | {0,11,13,23,62,76,79, 123} | {0,19,35,50,71,77, 117,125} |
| 5 | 15 | 202 | {0,7,27,76,113,137, 155,156,170,202} | {0,8,38,48,59,82,111, 146,150,152} | {0,12,25,26,76,81,98, 107,143,197} |

表 11-4 $n_0=5$ 自正交系统卷积码

| $R=1/5$ | t_{ML} | m | $\mathbf{g}^{(1,2)}$ | $\mathbf{g}^{(1,3)}$ | $\mathbf{g}^{(1,4)}$ | $\mathbf{g}^{(1,5)}$ |
|---------|----------|-----|--------------------------------|-----------------------------------|--------------------------------|-------------------------------|
| $R=4/5$ | t_{ML} | m | $\mathbf{g}^{(1,5)}$ | $\mathbf{g}^{(2,5)}$ | $\mathbf{g}^{(3,5)}$ | $\mathbf{g}^{(4,5)}$ |
| 1 | 4 | 4 | {0,1} | {0,2} | {0,3} | {0,4} |
| 2 | 8 | 26 | {0,16,20,21} | {0,2,10,25} | {0,14,17,26} | {0,11,18,24} |
| 3 | 12 | 78 | {0,5,26,51,55,69} | {0,6,7,41,60,72} | {0,8,11,24,44,78} | {0,10,32,47,49,77} |
| 4 | 16 | 178 | {0,19,59,68,85, 88,103,141} | {0,39,87,117,138, 148,154,162} | {0,2,13,25,96, 118,168,172} | {0,7,65,70,97, 98,144,178} |

自正交码的大数逻辑译码器与译码过程，有点类似于分组码的大数逻辑译码器与译码过程，下面举例说明。

例 11.1 构造一个(3, 1, 2)和(3, 2, 2)码的自正交系统卷积码大数逻辑译码器。

由表 11-2 知, 对于 $n_0=3, m=2, R=1/3, k_0=1$ 的 $(3, 1, 2)$ 码来说有两个校验元; 而对 $R=2/3, k_0=2$ 的 $(3, 2, 2)$ 码来说有两个信息元。表中生成元列下面的数字是 $(0, 1)$ 和 $(0, 2)$, 表示对 $(3, 1, 2)$ 码来说, 它的子生成元是: $\mathbf{g}^{(1,2)}(D)=1+D, \mathbf{g}^{(1,3)}(D)=1+D^2$; 对 $(3, 2, 2)$ 码来说, 其子生成元是: $\mathbf{g}^{(1,3)}(D)=1+D, \mathbf{g}^{(2,3)}(D)=1+D^2$ 。由此不难得得到 $(3, 1, 2)$ 码的

$$\mathbf{G}(D) = [1, 1+D, 1+D^2] \quad \mathbf{H}(D) = \begin{bmatrix} 1+D & 1 & 0 \\ 1+D^2 & 0 & 1 \end{bmatrix}$$

$(3, 2, 2)$ 码的

$$\mathbf{G}(D) = \begin{bmatrix} 1 & 0 & 1+D \\ 0 & 1 & 1+D^2 \end{bmatrix} \quad \mathbf{H}(D) = [1+D, 1+D^2, 1]$$

显然它们互为对偶码。

由 $\mathbf{H}(D)$ 可以很快得到 $(3, 1, 2)$ 码和它的对偶码 $(3, 2, 2)$ 码的校验矩阵 \mathbf{H} , 它们分别是:

$$\mathbf{H}_1 = \left[\begin{array}{ccc|c} 110 & & & 0^1 \\ 101 & & & 0^2 \\ \hline 100 & 110 & & 1^1 \\ 000 & 101 & & 1^2 \\ \hline 000 & 100 & 110 & 2^1 \\ 100 & 000 & 101 & 2^2 \end{array} \right]$$

$$\mathbf{H}_2 = \left[\begin{array}{ccc|c} 111 & & & 0 \\ 100 & 111 & & 1 \\ 010 & 100 & 111 & 2 \end{array} \right]$$

由 \mathbf{H}_1 可组成 $(3, 1, 2)$ 码的以下 4 个对 e_{01} 码元位正交的校验和式:

$$\begin{cases} s_{01} = e_{01} + e_{02} \\ s_{02} = e_{01} + e_{03} \\ s_{11} = e_{01} + e_{11} + e_{12} \\ s_{22} = e_{01} + e_{21} + e_{23} \end{cases}$$

该码的 $d=J+1=5$, 能纠正连续 $(m+1)n_0=9$ 个码元内的两个错误。

由它的对偶码 $(3, 2, 2)$ 码的 \mathbf{H}_2 矩阵, 可得到两个对 e_{01} 和两个对 e_{02} 码元位正交的一致校验和式:

$$\begin{cases} s_0 = e_{01} + e_{02} + e_{03} \\ s_1 = e_{01} + e_{11} + e_{12} + e_{13} \\ s_2 = e_{01} + e_{02} + e_{03} \\ s_3 = e_{02} + e_{11} + e_{12} + e_{21} + e_{22} + e_{23} \end{cases}$$

由此可知，该码能纠正连续 9 个码元内的一个错误。这两个码的大数逻辑译码器分别如图 11-1 和图 11-2 所示。

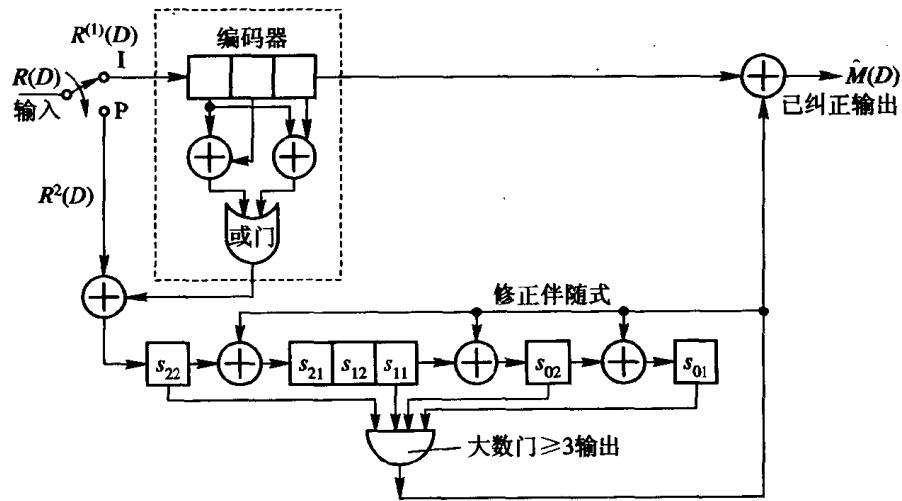


图 11-1 (3, 1, 2) 码大数逻辑译码器

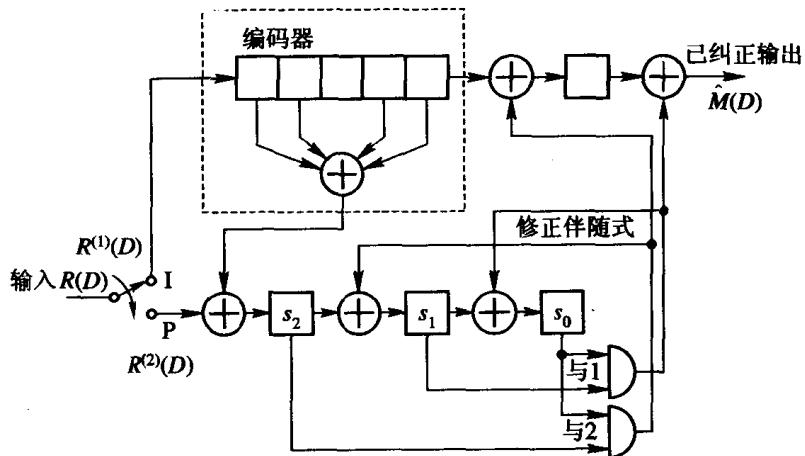


图 11-2 (3, 2, 2) 码大数逻辑译码器

这两个译码器的工作过程大致如下：把接收到的 $R(D)$ 中的每一段信息元送入编码器中求出校验元，与其后面收到的校验元模 2 加。若两者一致，则输出的伴随式分量 s_i 为 0；否则为 1。把加得的值送入伴随式寄存器中寄存。当接收完 3 个码段以后就开始对第 0 码段纠错，若此时大数逻辑门（或图 11-1 中的与 1、与 2 门）的输出为 1，则说明第 0 码段的信息元有错。这时正好第 0 子组的信息元移至编码器的输出端，从而把它们纠正。同时，纠错信号也反馈至伴随式寄存器修正伴随式，以消去此错误对伴随式的影响。如果大数判决门没有输出，则说明第 0 子组的信息元没有错误，这时从编码器中直接把信息元输出。这样，译码器每接收一个码段，就对此时刻前 m 个时刻输入的码段译码。所以，这类反馈大数逻辑译码器的译码约束度等于编码约束度，为 $m+1$ 。

由于伴随式寄存器中的数字大于一半为 1 时，大数逻辑门才有信号输出，所以每次对

伴随式修正后，总使得伴随式重量减轻。可知这类自正交码的译码电路并不会引起误差传播，因此它的纠错能力虽没有可正交码高，但实际中仍有应用。

二、可正交码

可正交码与自正交码相比，其纠错效率高，而实现编译码器的复杂性并不增加。但可正交码没有一般的构造方法，基本上只能用计算机搜索方法寻找，且仅局限于 $k_0=1$, $R=1/2, 1/3$ 等情况。

可正交码与自正交码都可以用大数逻辑方法译码，但它们之间也有差别，主要在于自正交码的 J 个正交校验一致和式，是由 \mathbf{H} 矩阵的行直接得到，而可正交码的 J 个正交一致校验和式，是由 \mathbf{H} 矩阵的行经过线性组合后得到的。

如一个 $(2, 1, 5)$ 系统可正交码，它的一个子生成元为：

$$\begin{aligned} g^{(1,2)} &= (100111) \quad g^{(1,2)}(D) = 1 + D^3 + D^4 + D^5 \\ G(D) &= [1, 1 + D^3 + D^4 + D^5] \end{aligned}$$

由此可知码的校验矩阵

$$\mathbf{H} = \left[\begin{array}{cccccc} 11 & & & & & & \\ 00 & 11 & & & & & \\ 00 & 00 & 11 & & & & \\ 10 & 00 & 00 & 11 & & & \\ 10 & 10 & 00 & 00 & 11 & & \\ 10 & 10 & 10 & 00 & 00 & 11 \end{array} \right]$$

设错误图样 $\mathbf{E} = (e_{01}e_{02}, e_{11}e_{12}, e_{21}e_{22}, e_{31}e_{32}, e_{41}e_{42}, e_{51}e_{52})$ ，则相应的伴随式序列是

$$\mathbf{s} = \mathbf{E} \cdot \mathbf{H}^T = (s_0, s_1, s_2, s_3, s_4, s_5)$$

式中：

$$\begin{aligned} s_0 &= e_{01} + e_{02} \\ s_1 &= \quad \quad \quad + e_{11} + e_{12} \\ s_2 &= \quad \quad \quad + e_{21} + e_{22} \\ s_3 &= e_{01} \quad \quad \quad \quad \quad + e_{31} + e_{32} \\ s_4 &= e_{01} \quad + e_{11} \quad \quad \quad \quad \quad + e_{41} + e_{42} \\ s_5 &= e_{01} \quad + e_{11} \quad + e_{21} \quad \quad \quad \quad \quad + e_{51} + e_{52} \end{aligned}$$

若取：

$$\begin{aligned} A_1 &= s_0 = e_{01} + e_{02} \\ A_2 &= s_3 = e_{01} + e_{31} + e_{32} \\ A_3 &= s_4 = e_{01} + e_{11} + e_{41} + e_{42} \\ A_4 &= s_1 + s_5 = e_{01} + e_{12} + e_{21} + e_{51} + e_{52} \end{aligned}$$

则 A_1, A_2, A_3 和 A_4 组成了 4 个对 e_{01} 码元位正交的正交一致校验和式，它们分别是 \mathbf{H} 矩阵第 0、3、4 行及第 1 和第 5 行的线性相加（模 2 和）得到的。

定义 11.1.2 任一个 (n_0, k_0, m) 卷积码，若可用 \mathbf{H} 矩阵的某些行的线性组合，组成 J 个或更多个对每个 e_{oi} ($0 \leq i \leq k_0$, 若为非系统码则 $0 \leq i \leq n_0$) 码元位正交的正交一致校验和式，则称这类码为可正交码。若码的最小距离 $d = J+1$ ，则称为完备可正交码。

得到了正交校验和式 A_1 、 A_2 、 A_3 和 A_4 后，可用大数逻辑译码估计 e_{01} 码元位的错误值，从而纠正相继 $(m+1)n_0=12$ 个码元内的两个随机错误。一般而言，可正交码能够用反馈大数逻辑译码，纠正连续 $(m+1)$ 码段内的 $\lfloor J/2 \rfloor$ 个错误。

表 11-5~表 11-7 中列出了某些系统码形式的可正交码。表中的“正交化规则”一列中的数字，指明正交一致校验和式是由 H 矩阵的哪几行线性组合得到的，下面举例说明。

表 11-5 $n_0=2$ 可正交系统卷积码

| $R=1/2$ | | $g^{(1,2)}$ | 正交化规则 |
|----------|-----|--|--|
| t_{ML} | m | | |
| 2 | 5 | {0,3,4,5} | $(0^2)(3^2)(4^2)(1^25^2)$ |
| 3 | 11 | {0,6,7,9,10,11} | $(0^2)(6^2)(7^2)(9^2)(1^23^210^2)(4^28^211^2)$ |
| 4 | 21 | {0,11,13,16,17,19,20,21} | $(0^2)(11^2)(13^2)(16^2)(17^2)(2^23^26^219^2)$ $(4^214^220^2)(1^25^28^215^221^2)$ |
| 5 | 35 | {0,18,19,27,28,29,30,32,33,35} | $(0^2)(18^2)(19^2)(27^2)(1^29^228^2)(10^220^229^2)$ $(11^230^231^2)(13^221^223^232^2)(14^233^234^2)$ $(2^23^216^224^226^235^2)$ |
| 6 | 51 | {0,26,27,39,40,41,42,44,45,47, 48,51} | $(0^2)(26^2)(27^2)(30^2)(1^213^240^2)(14^228^241^2)$ $(15^242^243^2)(17^229^231^244^2)(18^245^246^2)$ $(2^23^220^232^234^247^2)(21^235^248^249^250^2)$ $(24^230^233^236^238^251^2)$ |

表 11-6 $n_0=3$ 可正交系统卷积码

| $R=1/3$ | | $g^{(1,2)}$ | $g^{(1,3)}$ | 正交化规则 |
|----------|-----|---|-----------------|---|
| t_{ML} | m | | | |
| 3 | 4 | {0,1} | {0,2,3,4} | $(0^2)(0^3)(1^2)(2^3)(1^33^3)(2^24^3)$ |
| 4 | 7 | {0,1,7} | {0,2,3,4,6} | $(0^2)(0^3)(1^2)(2^3)(1^33^3)(2^24^3)(7^2)(3^25^26^26^3)$ |
| 5 | 0* | {0,1,9} | {0,1,2,3,5,8,9} | $(0^2)(0^3)(1^2)(2^22^3)(9^2)(3^34^3)(3^25^25^3)(1^34^26^26^3)$ $(8^28^3)(7^39^310^3)$ |
| 6 | 17* | {0,4,5,6,7,9, 12,13,16} | {0,1,14,15,16} | $(0^2)(0^3)(1^21^3)(4^2)(5^2)(2^36^2)(14^3)(7^210^211^211^3)$ $(3^35^39^2)(6^38^312^2)(3^316^317^3)(4^310^312^316^2)$ |
| 7 | 22 | {0,4,5,6,7,9, 12,13,16,19, 20,21} | {0,1,20,22} | $(0^2)(0^3)(1^21^3)(4^2)(5^2)(2^36^2)(7^210^211^211^3)$ $(3^35^39^2)(19^320^3)(22^3)(6^38^312^2)(4^310^312^316^2)$ $(3^27^313^315^319^2)(9^313^214^318^220^221^221^3)$ |
| 8 | 35 | {0,4,5,6,7,9, 12,16,17,30,31} | {0,1,22,25,35} | $(0^2)(0^3)(1^21^3)(4^2)(5^2)(2^36^2)(22^3)$ $(7^210^211^211^3)(3^225^3)(3^35^39^2)(6^38^312^2)$ $(7^314^217^218^218^3)(9^316^219^220^220^3)(14^315^335^3)$ $(12^321^328^231^232^2)(10^313^319^326^329^330^2)$ |

表 11-7 $n_0=5$ 可正交系统卷积码

| R=1/5 | | $\mathbf{g}^{(1,2)}$ | $\mathbf{g}^{(1,3)}$ | $\mathbf{g}^{(1,4)}$ | $\mathbf{g}^{(1,5)}$ | 正交化规则 |
|----------|-----|-------------------------------------|----------------------|----------------------|----------------------|---|
| t_{ML} | m | | | | | |
| 3 | 1 | {0,1} | {0,1} | {0} | {0} | $(0^2)(0^3)(0^4)(0^5)(1^21^4)(1^31^5)$ |
| 4 | 2 | {0,1,2} | {0,1} | {0,2} | {0} | $(0^2)(0^3)(0^4)(0^5)(1^21^4)(1^31^5)$ $(2^22^3)(2^42^5)$ |
| 5 | 3 | {0,1,2,3} | {0,1} | {0,2} | {0,3} | $(0^2)(0^3)(0^4)(0^5)(1^21^4)(1^31^5)$ $(2^22^3)(2^42^5)(3^5)(3^23^3)$ |
| 6 | 5 | {0,1,2,3,4} | {0,1} | {0,2,5} | {0,3,5} | $(0^2)(0^3)(0^4)(0^5)(1^21^4)(1^31^5)$ $(2^22^3)(2^42^5)(3^5)(3^23^3)(3^44^24^4)$ $(5^5)(4^35^35^4)$ |
| 7 | 6 | {0,1,2,3,4} | {0,1} | {0,2,5,6} | {0,3,5} | $(0^2)(0^3)(0^4)(0^5)(1^21^4)(1^31^5)(2^2$ $2^3)(2^42^5)(3^5)(3^23^3)(3^44^24^4)(5^5)$ $(4^35^35^4)(4^56^4)$ |
| 8 | 8 | {0,1,2,3,4} | {0,1,8} | {0,2,5,6,7} | {0,3,5} | $(0^2)(0^3)(0^4)(0^5)(1^21^4)(1^31^5)2^2$ $2^3)(2^42^5)(3^5)(3^23^3)(3^44^24^4)(5^5)$ $(4^35^35^4)(4^56^4)(8^3)(5^26^37^27^4)$ |
| 9 | 10 | {0,1,2,3,5, 6,8,10} | {0,3,5,6,8} | {0,1} | {0,2,10} | $(0^2)(0^3)(0^4)(0^5)(1^2)(2^22^4)(3^3)$ $(1^41^5)(2^32^5)(3^24^2)(3^45^25^4)(9^410^2$ $10^3)(5^3)(3^56^3)(10^5)(1^34^46^26^4)$ $(7^27^48^29^2)(4^35^57^38^38^4)(6^59^512^3)$ |
| 10 | 12 | {0,1,2,3,5, 6,8,10} | {0,3,5,6,8} | {0,1,10} | {0,2,10,12} | $(0^2)(0^3)(0^4)(0^5)(1^2)(2^22^4)(3^3)$ $(1^41^5)(2^32^5)(3^24^2)(3^45^25^4)(9^410^2$ $10^3)(5^3)(3^56^3)(10^5)(1^34^46^26^4)$ $(7^27^48^29^2)(4^35^57^38^38^4)(6^59^512^3)$ $(10^411^512^412^5)$ |
| 11 | 15 | {0,1,2,3,5, 6,8,10,11, 13,14} | {0,3,5,6,8} | {0,1,10} | {0,2,10,12, 15} | $(0^2)(0^3)(0^4)(0^5)(1^2)(2^22^4)(3^3)$ $(1^41^5)(2^32^5)(3^24^2)(3^45^25^4)(9^410^2$ $10^3)(5^3)(3^56^3)(10^5)(1^34^46^26^4)$ $(7^27^48^29^2)(4^35^57^38^38^4)(6^59^512^3)$ $(10^411^512^412^5)(4^513^414^214^3)$ $(13^314^415^415^5)$ |

例 11.2 表 11-6 中的(3, 1, 4)码, $d=7$, 子生成元列中的数字是(0, 1)和(0, 2, 3, 4), 表示两个子生成元多项式中系数取 1 的次数, 可知生成元为:

$$\begin{aligned}\mathbf{g}^{(1,2)}(D) &= 1 + D & \mathbf{g}^{(1,3)}(D) &= 1 + D^2 + D^3 + D^4 \\ \mathbf{G}(D) &= [1, 1 + D, 1 + D^2 + D^3 + D^4]\end{aligned}$$

相应的校验矩阵为:

$$\mathbf{H}(D) = \begin{bmatrix} 1 + D & 1 & 0 \\ 1 + D^2 + D^3 + D^4 & 0 & 1 \end{bmatrix}$$

| | | | | | | 行号码 |
|-------|-----|-----|-----|-----|-----|-------|
| | 110 | | | | | 0^2 |
| | 101 | | | | | 0^3 |
| | 100 | 110 | | | | 1^2 |
| | 000 | 101 | | | | 1^3 |
| $H =$ | 000 | 100 | 110 | | | 2^2 |
| | 100 | 000 | 101 | | | 2^3 |
| | 000 | 000 | 100 | 110 | | 3^2 |
| | 100 | 100 | 000 | 101 | | 3^3 |
| | 000 | 000 | 000 | 100 | 110 | 4^2 |
| | 100 | 100 | 100 | 000 | 101 | 4^3 |

表中“正交化规则”列中的数字是(0^2)、(0^3)、(1^2)、(2^3)、(1^33^3)和(2^24^3)，说明由 H 矩阵的第 0^2 、 0^3 、 1^2 、 2^3 行，及 1^3 和 3^3 行的线性组合、 2^2 和 4^3 行的线性结合，组成了 6 个正交一致校验和式：

$$\begin{aligned}
 A_1 &= s_{01} = e_{01} + e_{02} \\
 A_2 &= s_{02} = e_{01} + e_{03} \\
 A_3 &= s_{11} = e_{01} + e_{11} + e_{12} \\
 A_4 &= s_{22} = e_{01} + e_{21} + e_{23} \\
 A_5 &= s_{12} + s_{32} = e_{01} + e_{13} + e_{31} + e_{33} \\
 A_6 &= s_{21} + s_{42} = e_{01} + e_{22} + e_{41} + e_{43}
 \end{aligned}$$

可知该码能用反馈大数逻辑译码、纠正连续 5 个码段中的 3 个随机错误。此码的对偶码是(3, 2, 4)码，它并不是可正交码，这一点正好与自正交码相反。

此(3, 1, 4)可正交系统卷积码的反馈大数逻辑译码器如图 11-3 所示，它的工作过程类似于图 11-1 和图 11-2 的自正交码译码器，这里不再重复。■

上面讨论的反馈大数逻辑译码的约束长度等于编码约束长度，为 $n_0(m+1)$ 。虽然也可用定译码的大数逻辑译码方法译码，但是对可正交码来说，在定译码约束长度 $(2m+1)n_0$ 个码元内，所得到的正交校验和式的数目，与反馈译码时得到的不一定相等。如以前列举的(2, 1, 5)可正交码，若用定译码时只能在 $n_0(2m+1)=22$ 个码元内构造两个正交校验和式，仅能在连续 22 个码元内纠正一个错误。由此可见，对可正交码来说反馈译码比定译码要好得多。但是反馈译码可能会带来误差传播，必须仔细选择码的子生成元。已证明在 BSC 信道下，短约束长度可正交码的性能和相应 BCH 码的性能差不多，但译码实现上却比 BCH 码要简单得多。

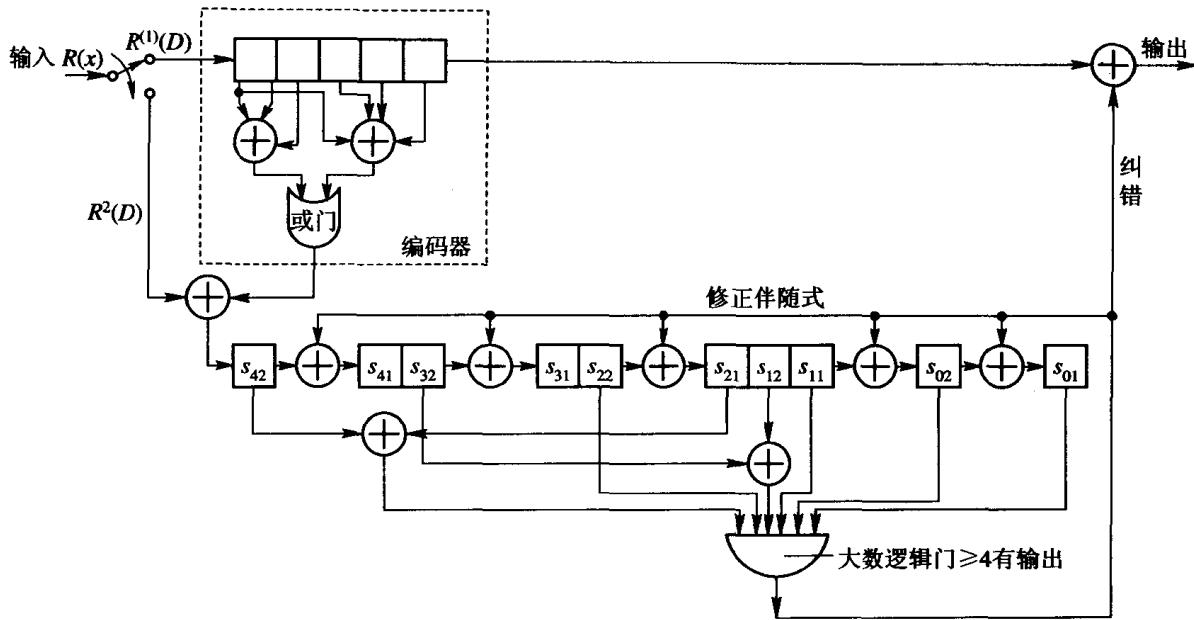


图 11-3 (3, 1, 4) 可正交系统卷积码译码器

§ 11.2 非系统卷积码的大数逻辑译码

对于非系统码来说，利用 G 矩阵行的线性变换可以转换成系统码，且有相同的最小汉明距离，但不能保证有相同的自由距离。所以，在一个译码约束长度内（通常等于编码约束长度 $(m+1)n_0$ ）利用大数逻辑方法译码时的纠错能力与系统码相同。这表明当选择好码的参数后，如果利用大数逻辑方法译码，则只要考虑系统码而不必考虑非系统码。但是，对某些非系统码来说，可以通过增加译码约束度的方法，提高大数逻辑译码时的纠错能力^[4,5]。

例如(2, 1, 3)非系统卷积码，它的子生成元为：

$$\begin{aligned} g^{(1,1)} &= (1111) & g^{(1,1)}(D) &= 1 + D + D^2 + D^3 \\ g^{(1,2)} &= (1011) & g^{(1,2)}(D) &= 1 + D^2 + D^3 \end{aligned}$$

可知生成矩阵和校验矩阵分别是：

$$G(D) = [1 + D + D^2 + D^3, 1 + D^2 + D^3]$$

$$H(D) = [1 + D^2 + D^3, 1 + D + D^2 + D^3]$$

相应的初始截短码的校验矩阵为

$$H = \begin{bmatrix} 11 & & & \\ 01 & 11 & & \\ 11 & 01 & 11 & \\ 11 & 11 & 01 & 11 \end{bmatrix}$$

错误图样 $E = (e_{01}e_{02}, e_{11}e_{12}, e_{21}e_{22}, e_{31}e_{32})$ ，由 $S = E \cdot H^T$ 得伴随式 $S = (s_0, s_1, s_2, s_3)$ ，式中：

$$\begin{aligned}
s_0 &= e_{01} + e_{02} \\
s_1 &= e_{02} + e_{11} + e_{12} \\
s_2 &= e_{01} + e_{02} + e_{12} + e_{21} + e_{22} \\
s_3 &= e_{01} + e_{02} + e_{11} + e_{12} + e_{22} + e_{31} + e_{32}
\end{aligned}$$

由此 4 个伴随式分量可得到以下的正交校验和式：

$$\begin{aligned}
A_0 &= s_0 = e_{01} + e_{02} \\
A_1 &= s_0 + s_1 = e_{01} + e_{11} + e_{12} \\
A_2 &= s_1 + s_3 = e_{01} + e_{22} + e_{31} + e_{32}
\end{aligned}$$

它们组成了对 e_{01} 码元位正交的 3 个校验和式，而

$$\begin{aligned}
B_0 &= s_0 = e_{01} + e_{02} \\
B_1 &= s_1 = e_{02} + e_{11} + e_{12} \\
B_2 &= s_0 + s_1 + s_3 = e_{02} + e_{22} + e_{31} + e_{32}
\end{aligned}$$

组成了对 e_{02} 码元位正交的 3 个校验和式。

从上面两组正交方程可知，在连续 $(m+1)n_0 = 8$ 个码元内，该码能用反馈大数逻辑译码方法纠正一个错误，在某些特定的码元位上可纠两个或两个以上错误。该码的 $d = J+1 = 4$ ，因此在编码约束长度 $(m+1)n_0 = 8$ 个连续码元内，上述纠错能力已达到了极限。

如果延长由 \mathbf{H} 矩阵所决定的约束长度，例如从 8 位增至 12 位，则由 \mathbf{H}_∞ 中截取 12 段长可得初始截短码的校验矩阵为

$$\mathbf{H}_{12} = \left[\begin{array}{ccccccccc} 11 & & & & & & & & \\ 01 & 11 & & & & & & & \\ 11 & 01 & 11 & & & & & & \\ 11 & 11 & 01 & 11 & & & & & \\ & 11 & 11 & 01 & 11 & & & & \\ & & 11 & 11 & 01 & 11 & & & \\ & & & 11 & 11 & 01 & & & \\ & & & & 11 & 11 & 01 & & \\ & & & & & 11 & 11 & & \\ & & & & & & 11 & & \\ & & & & & & & 11 & \end{array} \right]$$

错误图样 $\mathbf{E} = (e_{01}e_{02}, e_{11}e_{12}, e_{21}e_{22}, e_{31}e_{32}, e_{41}e_{42}, e_{51}e_{52})$ ，由 $\mathbf{S} = \mathbf{E} \cdot \mathbf{H}_{12}^T$ 可得以下两组正交校验和式：

$$\begin{aligned}
A_0 &= s_0 = e_{01} + e_{02} \\
A_1 &= s_0 + s_1 = e_{01} + e_{11} + e_{12} \\
A_2 &= s_1 + s_3 = e_{01} + e_{22} + e_{31} + e_{32} \\
A_3 &= s_1 + s_3 + s_5 = e_{01} + e_{21} + e_{42} + e_{51} + e_{52} \\
B_0 &= s_0 = e_{01} + e_{02} \\
B_1 &= s_1 = e_{02} + e_{11} + e_{12} \\
B_2 &= s_0 + s_1 + s_3 = e_{02} + e_{22} + e_{31} + e_{32} \\
B_3 &= s_0 + s_1 + s_3 + s_5 = e_{02} + e_{21} + e_{42} + e_{51} + e_{52}
\end{aligned}$$

它们分别组成了对 e_{01} 和 e_{02} 码元位正交的 4 个正交校验和式，所以可在连续 12 个码元内纠正两个随机错误。

该码是非系统码，译码器纠错后得到的 $\hat{C}(D)$ 序列还必须经过 $G^{-1}(D)$ 变换，以得信息估值序列 $\hat{M}(D) = \hat{C}(D)G^{-1}(D)$ 。

由 $G(D)G^{-1}(D) = D^k I_k$ ，可求得译码多项式矩阵为

$$G^{-1}(D) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

可以验证

$$\begin{aligned} G(D)G^{-1}(D) &= [1 + D + D^2 + D^3, 1 + D^2 + D^3] \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\ &= [1 + D + D^2 + D^3 + 1 + D^2 + D^3] \\ &= D \end{aligned}$$

可知该码是一个似快检码。由此可组成如图 11-4 所示的译码器，这是一个反馈大数逻辑译码器。图中，最右边的一个模 2 加法器完成 $G^{-1}(D)$ 运算。

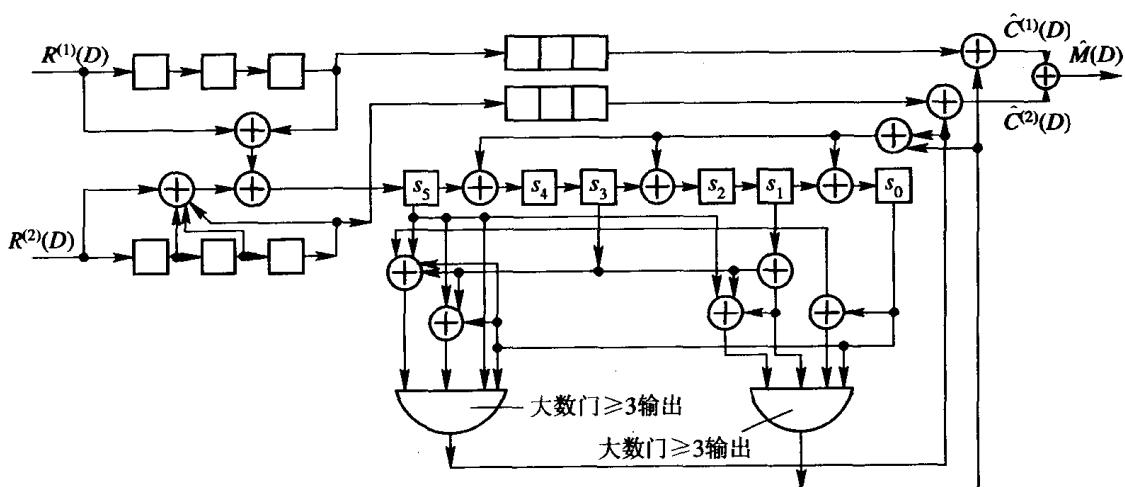


图 11-4 (2, 1, 3) 非系统卷积码反馈大数逻辑译码器

以前曾谈到对非系统码而言，由 $G(D)$ 求得的 $H(D)$ 并不是唯一的，因此可以挑选不同的校验矩阵 $H(D)$ ，以便能组成更多的正交校验和式，提高码的纠错能力^[5]。

§ 11.3 纠突发错误卷积码的基本概念

纠突发错误卷积码分为两类： B_1 型码和 B_2 型码。这两类码的唯一区别在于 B_2 型码的纠突发错误能力是以子码(码段)为单位考虑的(类似于分组码中的纠定段突发错误码)，而 B_1 型码是以码元为单位，因此 B_2 型码仅是 B_1 型的特殊情况。从实用观点来看 B_1 型码更为有用，但从分析与构造码的角度考虑， B_2 型码更为容易。但这两类码有密切的关系， B_1 型码可以很容易地当作 B_2 型码来使用，反之亦然。

如果 (n_0, k_0, m) 码的纠 B_1 型突发错误的能力为 b_1 (以码元为单位)，则它的纠 B_2 型突发错误(突发错误从一码段开始结束于另一码段)的能力为

$$b_2 \leq n_0 \lfloor b_1/n_0 \rfloor \quad (11.3.1)$$

反之，若码有 B_2 型纠突发能力 $b_2 = \lambda n_0$ ，则其 B_1 型纠突发能力为

$$b_1 \leq (\lambda - 1)n_0 + 1 \quad (11.3.2)$$

必须注意，无论哪类码，它们的纠错能力与保障区间和译码方法这两个因素有密切关系。

定义 11.3.1 连续两个突发错误之间的无误区间定义为**保障区间**。

对不同型码，即使在同样的译码方法下也需要有不同的保障区间。若译码约束长度是 $n_0(m+1)$ ，则 B_1 型码所要求的保障区间长度 g_1 为

$$n_0(m+1) - 1 \leq g_1 \leq n_0(m+1) \quad (11.3.3)$$

B_1 型码所要求的保障区间长度 g_2 为

$$g_2 \geq n_0 m \quad (11.3.4)$$

除了 B_1 型和 B_2 型码之外，还有另一类纠突发错误卷积码，它们虽然不能纠正长度 $\leq b$ 的所有突发错误，但能纠正其中绝大部分错误，即能以很小的译码错误概率纠正长度 $\leq b$ 的突发错误，这类码被称为**有误纠突发错误码**。而 B_1 型和 B_2 型码能纠正长度 $\leq b_1$ 和 b_2 的全部突发错误，所以这两类码也称为**无误纠突发错误码**。今后除非特别声明，所讨论的纠突发错误卷积码均指后者。

下面首先讨论纠突发错误卷积码，纠突发能力 b （若无特别说明， b 通常指 b_1 ）与 H 矩阵之间的关系。

定理 11.3.1 任何一个 (n_0, k_0, m) 卷积码，有纠突发能力为 b 的充要条件是：在校验矩阵 H 中，由任意相邻 b 列为一组的，互不相交的两组，它们列的任意线性组合不能为 0，且其中一组至少有一列在 H 矩阵中的首 n_0 列中选取。

该定理说明任何不相交的两个长度 $\leq b$ 的突发，且其中一个突发从第 0 码段开始，则它们共同组成的错误图样，与 H 矩阵相乘所得之伴随式不能为 0。也就是说要求不同的突发错误图样，有不同的伴随式，只有如此才能保证译码器能正确区分两个不同的突发，否则译码器不能正确判决，引起译码错误，因而该定理的正确性是显而易见的。

定理 11.3.2 对任何一个 $R > 0$ 的有限记忆（存贮）的二进制线性码，纠突发能力 b ，保障区间 g 和码率 R 之间，必须满足以下关系：

$$\frac{g}{b} \geq \frac{1+R}{1-R} \quad (11.3.5)$$

该定理的证明可参阅有关文献^[4]。对所有线性码，无论是分组码还是卷积码上式都必须满足。例如，对纠 b 长突发错误的 $[n, k]$ 线性分组码来说， $g = n - b$ ，代入上式得

$$\frac{n-b}{b} \geq \frac{1+R}{1-R} = \frac{n+k}{n-k}$$

求得

$$n - k \geq 2b$$

$$b \leq \frac{n-k}{2}$$

这与式(9.1.2)相同。

对于有误纠突发错误码， g 、 b 与 R 之间必须满足

$$\frac{g}{b} \geq \frac{R}{1-R} \quad (11.3.6)$$

能使式(11.3.5)或式(11.3.6)等号成立的码，称为**最佳纠突发错误码**，简称**最佳码**。

比较该两式可知，在同样的 R 、 b 下，有误纠突发错误卷积码要求的保障区间比无误纠突发错误卷积码要小得多。如 $R=0.5$ 时，有误纠突发错误卷积码要求的 $g \geq b$ ，仅仅是无误纠突发错误卷积码所要求的三分之一。因此，在突发错误比较频繁的信道中，应用有误纠突发错误卷积码能取得更好的效果。

目前，寻找和构造最佳或接近最佳纠突发错误卷积码有以下一些方法：

- (1) 应用交错技术，由约束长度短的最佳码得到约束长度长的最佳码；
- (2) 利用分析方法构造纠正单个突发错误的最佳码，如岩垂码、BPM 码^[1]和扩散卷积码等。
- (3) 确定突发位置然后予以纠正，这就是纠突发删除码，这类码就是有误纠突发错误卷积码，如加拉格尔(Gallager)码等^[1,4]。

§ 11.4 交 错 码

交错码既可用来纠随机错误又可用来纠突发错误，因此特别适合于组合信道的纠错系统。用卷积码构造的交错码，在基本思想、构造方法以及性能分析上均与分组码的交错码相同。不同的是用卷积码交错时有两种交错方法：按码元交错和按子码(码段)交错。

卷积码的码段或码元交错，是把 (n_0, k_0, m) 卷积码的 i 个码序列 C_1, C_2, \dots, C_i 作为行码构成如下码阵：

$$\begin{aligned} C_1: & c_{01}^1 c_{02}^1 \cdots c_{0n_0}^1, c_{11}^1 c_{12}^1 \cdots c_{1n_0}^1, \dots, c_{m1}^1 c_{m2}^1 \cdots c_{mn_0}^1, \dots \\ C_2: & c_{01}^2 c_{02}^2 \cdots c_{0n_0}^2, c_{11}^2 c_{12}^2 \cdots c_{1n_0}^2, \dots, c_{m1}^2 c_{m2}^2 \cdots c_{mn_0}^2, \dots \\ & \vdots \\ C_i: & c_{01}^i c_{02}^i \cdots c_{0n_0}^i, c_{11}^i c_{12}^i \cdots c_{1n_0}^i, \dots, c_{m1}^i c_{m2}^i \cdots c_{mn_0}^i, \dots \end{aligned}$$

传输时以列的次序，按以下方式传输：

$$C_B: c_0^1 c_{02}^1 \cdots c_{0n_0}^1 c_1^1 c_{12}^1 \cdots c_{1n_0}^1 c_m^1 c_{m2}^1 \cdots c_{mn_0}^1 \cdots$$

式中， $c_j^k = (c_{j1}^k c_{j2}^k \cdots c_{jn_0}^k)$ ($k=1, 2, \dots, i$; $j=0, 1, 2, \dots, m, \dots$) 是第 k 个码字序列的第 j 码段，可知 C_B 序列就是 i 度码段交错卷积码 $(n_0, k_0, i(m+1)-1)$ 码的一个码序列。

由上表示可知，若 (n_0, k_0, m) 卷积码的子生成元是 $\mathbf{g}^{(l,j)}(D)$ ($l=1, 2, \dots, k_0$; $j=1, 2, \dots, n_0$)，则 i 度码段交错码的子生成元是 $\mathbf{g}_i^{(l,j)}(D^i)$ ，也就是把原码的每个码段之间的约束关系拉长 $(i-1)$ 倍。

若以列的次序按如下方式传输：

$$C_s: c_{01}^1 c_{02}^2 \cdots c_{01}^i c_{02}^1 c_{02}^2 \cdots c_{02}^i c_{0n_0}^1 c_{0n_0}^2 \cdots c_{0n_0}^i \cdots c_{11}^1 c_{12}^2 \cdots c_{11}^i c_{12}^1 c_{12}^2 \cdots c_{12}^i \cdots c_{mn_0}^1 \cdots$$

则 C_s 序列就是 i 度码元交错卷积码 (in_0, ik_0, m) 码的一个码字序列。可知子生成元的个数从 $k_0 n_0$ 个增加到 $ik_0 n_0$ 个，其中 $k_0 n_0$ 个是原来的子生成元，而新增加的 $(i-1)k_0 n_0$ 个子生成元是 0。

无论是码段还是码元交错，都称 i 是交错卷积码的交错度。

例 11.3 (3, 2, 8) 系统卷积码的两个子生成元是： $\mathbf{g}^{(1,3)}(D) = D^3 + D^8$ ， $\mathbf{g}^{(2,3)}(D) = D + D^7$ 。它的 B₁ 型纠突发能力 $b_1 = n_0 = 3$ ，要求保障区间 $g_1 = (m+1)n_0 - 1 = 26$ 。按二度交错构成以下交错阵。设：

$$C_1: (100, 000, 000, 001, 000, 000, 000, 000, 001, \dots)$$

$$C_2: (010, 001, 000, 000, 000, 000, 000, 001, 000, \dots)$$

若按码元交错，则得到(6, 4, 8)二度码元交错卷积码的一个码字为

$$C_s: (100100, 000001, 000000, 000010, 000000, \dots, 000010, \dots)$$

若按码段交错，则得到(3, 2, 17)二度码段交错卷积码的一个码字为

$$C_B: (100, 010, 000, 001, 000, 000, 001, 000, \dots, 001, 000, \dots)$$

显然，该码的两个子生成元： $\mathbf{g}^{(1,3)}(D) = D^6 + D^{16}$, $\mathbf{g}^{(2,3)}(D) = D^2 + D^{14}$ 。 ■

交错卷积码的译码方法与分组交错码完全一样，把接收到的码序列 C_s 或 C_B ，仍排成与发端相同的码阵，然后按行码的译码规则逐行译码。由此可知，对码段交错卷积码来说，若行码能在约束长度 $(m+1)n_0$ 个码元内，纠正长度 $b \leq \lambda n_0$ 的突发错误，则在交错码约束长度 $i n_0 (m+1)$ 个码元内，长度 $\leq i b$ 的任何突发错误，不论在 i 度交错码序列 C_B 中何处开始，对每一行码的码字序列的影响不会超过 b 位相邻码元。当按行码译码规则分别对每行译码时就能纠正这些错误。若行码能在约束长度内纠正 $\leq t$ 个随机错误，则在交错码的约束长度内， $\mu = \lfloor t/n_0 \rfloor$ 个长度 $\leq i n_0$ 的突发也能得到纠正。如上例中用(3, 2, 8)码构造的二度码段交错卷积码，能在 C_B 的 $i n_0 (m+1) = 54$ 个连续码元中，纠正长度 ≤ 6 的任何单个突发错误，要求的保障区间

$$g_{is} = i(m+1)n_0 - i = 52 \quad (11.4.1)$$

对码元交错卷积码来说，若行码能在约束长度 $(m+1)n_0$ 个相邻码元内纠正长度 $\leq b$ (不必是 n_0 的倍数) 的突发错误，则 i 度码元交错码能在 C_s 码序列中的连续 $(m+1)n_0 i$ 个码元内，纠正长度 $\leq i b$ 的任何单个突发错误。如上例中的(6, 4, 8)二度码元交错码，能在连续 54 个码元内纠正长度 ≤ 6 的任何单个突发错误，要求的保障区间

$$g_{is} \leq i(m+1)n_0 - 1 \leq 53 \quad (11.4.2)$$

同样，若行码能纠正 $\leq t$ 个随机错误，则交错码能纠正 t 个长度 $\leq i$ 的突发错误的任意组合。由此看出，码元交错卷积码可以更灵活地同时对付突发和随机错误。

交错卷积码的编译码器，仅仅是把行码的编译码器中移存器的每一级重复 $i-1$ 次即成，因而码元交错与码段交错的实现复杂度差不多。

* § 11.5 岩垂(Iwadare)码

岩垂码是 $(n_0, n_0-1, m)B_1$ 型纠突发错误卷积码，它能用比较简单的方法译码，且 n_0 较小时接近最佳码，因此是一类比较实用的纠突发错误卷积码。

岩垂码分为两类，除了 $n_0=2$ 和 3 之外，第一类比第二类码所需的保障区间要小，对大的 n_0 和小的 λ (≥ 1 的整数)而言，实现编译码所需的移位寄存器的级数也比第二类要少。但是，对大的 λ ，且 $\lambda \gg n_0$ 时，第二类比第一类码更为经济。这两类码在构造方法上没有本质不同，我们这里仅介绍第一类岩垂码。

系统形式的 (n_0, n_0-1, m) 岩垂码，它的 n_0-1 个子生成元为

$$\mathbf{g}^{(i, n_0)}(D) = D^{a(i)} + D^{b(i)} \quad i = 1, 2, \dots, n_0 - 1 \quad (11.5.1)$$

这里：

$$\begin{aligned} a(i) &= (\lambda + 1)(n_0 - i) - 1 \\ b(i) &= (\lambda + 1)(2n_0 - i) + i - 3 \end{aligned} \quad (11.5.2)$$

$\lambda \geq 1$ 的整数。当 $i=1$ 时, 式(11.5.2)中的 $b(1)$ 达到最大, 此时

$$b(1) = (\lambda + 1)(2n_0 - 1) - 2 = m \quad (11.5.3)$$

可知, 该码的编码约束度 $m=b(1)$ 。该码能纠正 $b_1=\lambda n_0$ 长的 B_1 型突发错误, 所需的保障区间由式(11.3.3)知为

$$g_1 = n_0(m + 1) - 1 = n_0(\lambda + 1)(2n_0 - 1) - n_0 - 1 \quad (11.5.4)$$

由式(10.2.29)可知, H 矩阵的首 n_0 列组成了 B_0 阵, 一当 B_0 阵已定, 则 H 矩阵也就确定了。由式(11.5.1)可知, 岩垂码 B_0 阵的首 n_0-1 列中的每列只有两个 1, 它的位置是 $a(i)$ 和 $b(i)$, 第 n_0 列中只有一个 1 处在第 0 行(首行)。由此 B_0 矩阵所构成的 H 矩阵, 与长度 $\leq \lambda n_0$ 的任何突发图样相乘所得之伴随式均不相同, 且不为 0, 满足定理 11.3.1 所提的要求, 故能纠正任何长度 $\leq \lambda n_0$ 的单个突发错误。下面通过具体例子说明该码的编译码方法。

设 $\lambda=1$, $n_0=3$, $k_0=2$ 。由式(11.5.3)得 $m=8$, 得到一个 $(3, 2, 8)$ 岩垂码, 能纠正 $b_1=\lambda n_0=3$ 个码元长的任何单个突发错误。由式(11.5.1)和式(11.5.2)可知, 该码的两个子生成元是: $g^{(1,3)}(D)=D^3+D^8$, $g^{(2,3)}(D)=D+D^7$ 。所需的保障区间, 由式(11.5.4)可知为 $g_1=26$ 。

由 $g^{(1,3)}(D)$ 和 $g^{(2,3)}(D)$ 不难能得到该 $(3, 2, 8)$ 岩垂码的 H 矩阵是

$$H = \left[\begin{array}{ccccccccc|c} 001 & & & & & & & & & \\ 010 & 001 & & & & & & & & \\ 000 & 010 & 001 & & & & & & & \\ 100 & & 010 & 001 & & & & & & \\ \hline 000 & 100 & 010 & 001 & & & & & & \\ 000 & & 100 & 010 & 001 & & & & & \\ 000 & & & 100 & 010 & 001 & & & & \\ 010 & & & & 100 & 010 & 001 & & & \\ \hline 100 & 010 & 000 & 000 & 000 & 100 & 000 & 010 & 001 & \\ \hline 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & \end{array} \right] \quad (11.5.5)$$

该矩阵的首 $n_0(3)$ 列组成了 B_0 阵, 显然, B_0 阵的前二列只有两个 1, 分别处在第 3、第 8 行和第 1、第 7 行, 也就是由 $a(i)$ 和 $b(i)$ 决定的行, 而第 3 列只有一个 1, 处在第 0 行。

$n_0=3$ 位长的 B_1 型突发图样分 3 种情况, 如下所示:

$$E_1 = (e_{01}e_{02}e_{03}, 00 \cdots 0),$$

$$E_2 = (0e_{02}e_{03}, e_{11}0 \cdots 0),$$

$$E_3 = (00e_{03}, e_{11}e_{12}0, \cdots 0)$$

相应的伴随式分别为:

$$\begin{aligned} S_{1\infty} &= E_1 \cdot H_\infty^T = (e_{03}e_{02}0e_{01}000e_{02}e_{01}00 \cdots 0) \\ &= (s_0^1, s_1^1, s_2^1, s_3^1, s_4^1, s_5^1, s_6^1, s_7^1, s_8^1, \cdots) \end{aligned} \quad (11.5.6)$$

$$\begin{aligned} S_{2\infty} &= E_2 \cdot H_\infty^T = (e_{03}e_{02}00e_{11}00e_{02}0e_{11}0 \cdots 0) \\ &= (s_0^2, s_1^2, s_2^2, s_3^2, s_4^2, s_5^2, s_6^2, s_7^2, s_8^2, s_9^2, \cdots) \end{aligned} \quad (11.5.7)$$

$$\begin{aligned} S_{\infty} &= E_3 \cdot H_{\infty}^T = (e_{03} 0 e_{12} 0 e_{11} 0 0 0 e_{12} e_{11} 0 \cdots 0) \\ &= (s_0^3, s_1^3, s_2^3, s_3^3, s_4^3, s_5^3, s_6^3, s_7^3, s_8^3, s_9^3, \dots) \end{aligned} \quad (11.5.8)$$

事实上,由式(10.3.3), $S(D)=E(D) \cdot H(D)^T$,也可以很快得到上面的伴随式,而不必求助于 H_{∞} 矩阵。

由式(11.5.6)可知, (s_3^1, s_8^1) 组成了对 e_{01} 码元位的两个正交校验和, (s_1^1, s_7^1) 组成了对 e_{02} 码元位的两个正交校验和。但是,对于 E_2 错误图样来说,虽然 (s_1^2, s_7^2) 能组成对 e_{02} 码元位正交的两个校验和式,但 (s_3^2, s_8^2) 却不能组成对 e_{11} 码元位正交的校验和,同理对于 E_3 错误图样来说,也不能从 (s_1^3, s_7^3) 和 (s_3^3, s_8^3) 组成对 e_{12} 和 e_{11} 的两个正交校验和。因此,用直接的一次大数逻辑译码方法不能纠正 B_1 型的长为3的单个突发图样。为此必须进行分段(级)大数逻辑译码。

由式(11.5.6)~式(11.5.8)看出,每一信息位 e_{ji} $(j=0, 1, i=1, 2)$ 在 S_{∞} 中均出现两次,且相隔 $d(i)$ 位,称 $d(i)$ 为再现间隔。由式(11.5.1)和 H 矩阵中 B_0 阵的结构形式可知, $d(i)$ 是 $b(i)$ 与 $a(i)$ 之差,即

$$d(i) = b(i) - a(i) - 1 = (\lambda + 1)n_0 + (i - 3) \quad (11.5.9)$$

在该例中 $d(1)=4$, e_{01} 在 S_{∞} 中相隔4位。 $d(2)=5$, e_{02} 在 S_{∞} 中相隔5位。利用这种 e_j 在 S_{∞} 中的不同的再现间隔,可以对 (e_{01}, e_{02}) 或 (e_{02}, e_{11}) 或 (e_{12}, e_{11}) 进行分段大数逻辑译码纠错。

由式(11.5.5)的 H 矩阵中虚线所示的矩阵内可得到:

$$\begin{aligned} s_2 &= s'_1 = e_{12} + e_{23} \\ s_8 &= s'_7 = e_{12} + e_{51} + e_{72} + e_{83} \end{aligned} \quad (11.5.10)$$

组成了对第一码段第二个信息位 e_{12} 的两个正交校验和式。用该式的大数判决结果对该码元进行纠错后,该子组在译码器中进入第0码段位置,由于第二个信息元已纠错,错误对该信息元的影响已消除,所以由 H 矩阵的第四行和第九行可得:

$$\begin{aligned} s_3 &= e_{01} + e_{22} + e_{33} \\ s_8 &= e_{01} + e_{51} + e_{72} + e_{83} \end{aligned} \quad (11.5.11)$$

组成了对第0码段第一个信息位 e_{01} 的两个正交校验和式,利用大数准则可对该信息位进行纠错。因此用这种二次分段大数逻辑译码后,就完成了对 e_{02} 和 e_{01} 的纠错。一般而言, n_0-1 个信息位中的错误,可用这种 (n_0-1) 次的分段大数逻辑译码方法译码。

该 $(3, 2, 8)$ 岩垂码的译码器如图11-5所示。其译码过程大致如下:

(1) 把输入 $R(D)$ 中每一段的前两个信息元送入编码器求得新的校验元,与后面输入的校验元进行比较,得到相应的伴随式分量,送入伴随式寄存器;

(2) 若与门 A_2 输出1,说明第一码段的第二个信息元有误,对它进行纠正,同时修正伴随式以消去此错误的影响;

(3) 若与门 A_1 输出1,说明第0码段的第一个信息元发生了错误,对它进行纠错,并修正伴随式。

可知用这种分段大数逻辑译码方法能纠正约束长度内任意的长度 $\leq n_0 = 3$ 的突发错误。

仔细观察该译码器,我们可以发现与门 A_2 两个输入端来自 s_2 、 s_8 ,它们之间相隔6个伴随式分量,这就是 e_{02} 在伴随式 S_{∞} 中的再现间隔加一;同理可知,与门 A_1 的两个输入端来自 s_3 、 s_8 ,它们之间的间隔就是 S_{∞} 中 e_{01} 的再现间隔加一。所以可以根据再现间隔,直接

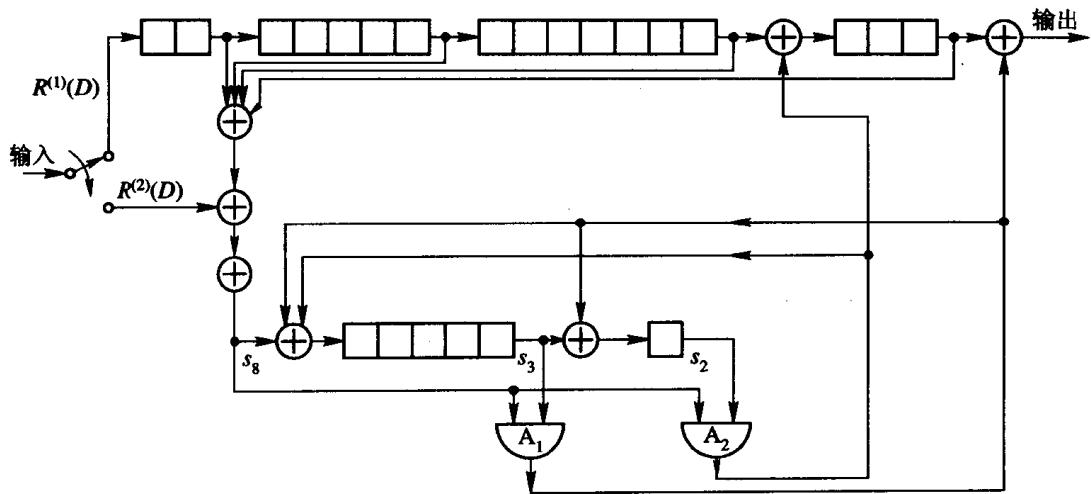


图 11-5 (3, 2, 8) 岩垂码译码器 $b_1=3$

得到译码器中大数门的组成，而不必求助于 \mathbf{H} 矩阵。

当 $\lambda=1$ 时，岩垂码的保障区间与纠突发能力之比为

$$\frac{g}{b} = \frac{(m+1)n_0 - 1}{b} = (4n_0 - 3) - \frac{1}{n_0} \quad (11.5.12)$$

而对于 $R=(n_0-1)/n_0$ 的最佳 B_1 型码来说，由式(11.3.5)可知

$$\frac{g}{b} = \frac{1+R}{1-R} = 2n_0 - 1 \quad (11.5.13)$$

对于 $\lambda>1$ 的岩垂码来说，它的译码器与 $\lambda=1$ 时的稍有不同，下面通过具体例子说明这一点。

例 11.4 $\lambda=2, n_0=3, k_0=2$ ，由式(11.5.3)知 $m=13$ ，得到一个(3, 2, 13)岩垂码，能纠正 $b_1 \leq \lambda n_0 = 6$ 位长的 B_1 型突发错误。由式(11.5.1)得到两个子生成元分别是：
 $\mathbf{g}^{(1,3)}(D)=D^5+D^{13}$, $\mathbf{g}^{(2,3)}(D)=D^2+D^{11}$ 。所需保障区间 $g=n_0(m+1)-1=41$ 。

由子生成元不难得到码的生成矩阵和校验矩阵分别为：

$$\mathbf{G}(D) = \begin{bmatrix} 1 & 0 & D^5 + D^{13} \\ 0 & 1 & D^2 + D^{11} \end{bmatrix}$$

$$\mathbf{H}(D) = [D^5 + D^{13}, D^2 + D^{11}, 1]$$

设长度 $b_1 \leq 6$ 的突发图样 $E_1 = (e_{01}e_{02}e_{03}, e_{11}e_{12}e_{13}, 0 \cdots 0)$ ，相应的 $\mathbf{E}_1(D) = (e_{01} + e_{11}D, e_{02} + e_{12}D, e_{03} + e_{13}D)$ 。由式(10.3.3)不难得到伴随式多项式

$$\begin{aligned} S_1(D) = \mathbf{E}_1(D) \cdot \mathbf{H}(D)^T = & e_{03} + e_{13}D + e_{02}D^2 + e_{12}D^3 \\ & + e_{01}D^5 + e_{11}D^6 + e_{02}D^{11} + e_{12}D^{12} + e_{01}D^{13} + e_{11}D^{14} \end{aligned}$$

相应的伴随式

$$\begin{aligned} S_{1\infty} = & (e_{03}e_{13}e_{02}e_{12}0e_{01}e_{11}0000e_{02}e_{12}e_{01}e_{11}0\cdots) \\ = & (s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9, s_{10}, s_{11}, s_{12}, s_{13}, s_{14}, \cdots) \end{aligned} \quad (11.5.14)$$

由此可知, e_{j1} ($j=0, 1$) 的再现间隔 $d(1)$ 由式(11.5.9)可知是 7, e_{j2} ($j=0, 1$) 的 $d(2)=8$ 。利用 e_{j1} 与 e_{j2} 再现间隔的不同, 可得到图 11-6 所示的译码器。仔细观察该图可知, 除了图 11-5 的编码器是串行, 图 11-6 是并行之外, 图 11-6 与图 11-5 的最大不同点是与门 A_1 的输入端有 3 个, 其中一个是由伴随式寄存器最后一级的输出并通过反相器 N 输入的。由式(11.5.14)可知, 若 $e_{02}=1$, 则由 $s_2=s_{11}=e_{02}=1$, 与门 A_2 的两个输入端均为 1, A_2 输出 1 对 e_{02} 进行纠错, 如果这时 $e_{12}=1$, 则与门 A_1 的两个输入端 $s_3=e_{12}=1$ 与 $s_{11}=e_{02}=1$, 引起与门 A_1 输出, 从而引起错纠。因此, 为了在纠正 e_{02}, e_{12} 期间, 不使 A_1 有输出引起错纠, 伴随式寄存器最右级的输出禁止 A_1 的输出。一般而言, 对 $\lambda>1$ 的岩垂码, 它的 n_0-1 个与门中 $A_1, A_2, \dots, A_{n_0-2}$ 与门都必须有一个从伴随式寄存器最右端经反相后的输入, 以禁止错纠。 ■

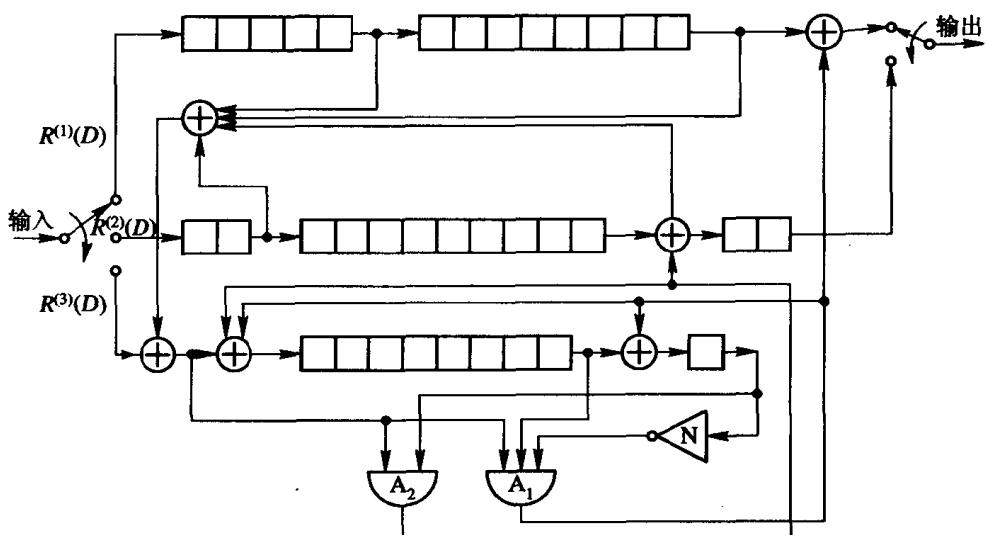


图 11-6 (3, 2, 13) 岩垂码译码器 $b_1=6$

§ 11.6 扩散卷积码

扩散卷积码的基本思想与交错码相同, 都是把子码之间的检验关系拉长, 以抗击长突发错误的影响, 所有的码是自正交码或可正交码, 并应用反馈大数逻辑方法译码。

一、自正交扩散卷积码

先以 $R=1/2$ 的 $(2, 1, 9)$ 系统卷积码为例, 说明码的纠错能力以及编码过程。该码的一个子生成元是

$$g^{(1,2)}(D) = 1 + D^3 + D^7 + D^9$$

相应的 H 矩阵是

$$\mathbf{H} = \begin{bmatrix} 11 \\ 00 & 11 \\ 00 & 00 & 11 \\ 10 & 00 & 00 & 11 \\ 00 & 10 & 00 & 00 & 11 \\ 00 & 00 & 10 & 00 & 00 & 11 \\ 00 & 00 & 00 & 10 & 00 & 00 & 11 \\ 10 & 00 & 00 & 00 & 10 & 00 & 00 & 11 \\ 00 & 10 & 00 & 00 & 00 & 10 & 00 & 00 & 11 \\ 10 & 00 & 10 & 00 & 00 & 00 & 10 & 00 & 00 & 11 \end{bmatrix}$$

错误图样

$$\mathbf{E} = (e_{01}e_{02}, e_{11}e_{12}, \dots, e_{91}e_{92})$$

伴随式

$$\mathbf{S} = \mathbf{E} \cdot \mathbf{H}^T = (s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9)$$

式中：

$$\begin{aligned} s_0 &= e_{01} + e_{02} \\ s_3 &= e_{01} + e_{31} + e_{32} \\ s_7 &= e_{01} + e_{41} + e_{71} + e_{72} \\ s_9 &= e_{01} + e_{21} + e_{61} + e_{91} + e_{92} \end{aligned}$$

组成了 4 个对 e_{01} 码元位正交的正交一致校验和式，能在译码约束长度 $(m+1)n_0 = 20$ 个码元内，用反馈大数逻辑译码方法纠正两个随机错误。

该码也能纠正长度为 4 个码元的突发错误。若突发在 e_{01} 码元位上开始，则 $s_0 + s_3 + s_7 + s_9$ 之值 ≥ 3 （只有 s_0 可能取 0，此时的值等于 3），所以可以纠正 e_{01} 位上的错误。若突发在 e_{01} 位以外的其它地方开始，则至多只影响 4 个正交校验和式中的 2 个， $s_0 + s_3 + s_7 + s_9 \leq 2$ ，不会使译码器的大数判决门输出而引起错纠。所以，该码的纠突发能力和要求的保障区间分别是：

$$b = 4$$

$$g = n_0(m+1) - 1 = 19$$

定义 11.6.1 一个自正交或可正交 (n_0, k_0, m) 系统卷积码（当然不限于系统码），若能组成 $J=2t$ 个对第 0 段的信息元 e_{0i} ($i=1, 2, \dots, k_0$) 码元位正交的一致校验和式，且这个正交一致校验和式有如下特点：

(1) 长为 l 的突发错误从第 0 段开始，若 $e_{0i}=1$ ，则至少应使 $t+1$ 个正交校验和取值为 1；

(2) 突发错误如果不从第 0 段开始， $e_{0i}=0$ ，则 l 长突发错误对 J 个正交校验和式的影响不能多于 t 个。

则称此卷积码为 l 度扩散卷积码，它能纠正 t 个随机错误或纠正长为 l 的突发错误。这里的 l 称为扩散度或扩散系数。

上例中的 $(2, 1, 9)$ 码， $l=4$, $J=4$ ，称此码是 4 度扩散卷积码。

为了便于应用,表 11-8 中列出了自正交扩散卷积码的参数。表中的 t 表示纠随机错误的数目, λ 表示扩散程度, 它与扩散系数 l 的关系是 $l = \lambda n_0$, m 是编码存贮, $\mathbf{g}^{(i,j)}$ 是子生成元, λ_{\min} 表示所要构造的扩散码的正交化规则中所要求的最小 λ 值。

表 11-8 自正交扩散系统卷积码

| (a) $R=1/2$ 码 | | λ_{\min} | $\mathbf{g}^{(1,2)}$ | | | |
|---------------|----------------|------------------|---|----------------------|---|----------------------|
| t | m | | | | | |
| 1 | 3λ | 1 | $\{0, \lambda, 3\lambda\}$ | | | |
| 2 | $4\lambda+1$ | 2 | $\{0, \lambda+1, 3\lambda+1, 4\lambda+1\}$ | | | |
| 3 | $5\lambda+4$ | 4 | $\{0, 1, \lambda+3, 3\lambda+3, 4\lambda+3, 5\lambda+4\}$ | | | |
| 4 | $6\lambda+10$ | 8 | $\{0, 1, 3, \lambda+7, 3\lambda+7, 4\lambda+7, 5\lambda+8, 6\lambda+10\}$ | | | |
| 5 | $7\lambda+19$ | 13 | $\{0, 1, 4, 6, \lambda+12, 3\lambda+12, 4\lambda+12, 5\lambda+13, 6\lambda+15, 7\lambda+19\}$ | | | |
| (b) $R=2/3$ 码 | | λ_{\min} | $\mathbf{g}^{(1,3)}$ | $\mathbf{g}^{(2,3)}$ | | |
| t | m | | | | | |
| 2 | $8\lambda+3$ | 3 | $\{0, \lambda, 4\lambda, 8\lambda+3\}$ | | | |
| 3 | $10\lambda+10$ | 7 | $\{0, 1, \lambda+2, 4\lambda+4, 8\lambda+5, 10\lambda+10\}$ | | | |
| 4 | $12\lambda+26$ | 12 | $\{0, 1, 4, \lambda+7, 4\lambda+11, 8\lambda+12, 9\lambda+13, 11\lambda+24\}$ | | | |
| (c) $R=3/4$ 码 | | λ_{\min} | $\mathbf{g}^{(1,4)}$ | $\mathbf{g}^{(2,4)}$ | $\mathbf{g}^{(3,4)}$ | |
| t | m | | | | | |
| 2 | $12\lambda+5$ | 6 | $\{0, \lambda, 6\lambda, 9\lambda-2\}$ | | $\{0, 4\lambda, 7\lambda, 8\lambda-1\}$ | |
| 3 | $15\lambda+12$ | 9 | $\{0, 1, 2\lambda+3, 11\lambda+6, 12\lambda+7, 15\lambda+12\}$ | | | |
| (d) $R=4/5$ 码 | | λ_{\min} | $\mathbf{g}^{(1,5)}$ | $\mathbf{g}^{(2,5)}$ | $\mathbf{g}^{(3,5)}$ | $\mathbf{g}^{(4,5)}$ |
| t | m | | | | | |
| 2 | $16\lambda+8$ | 7 | $\{0, \lambda, 10\lambda+3, 12\lambda+4\}$ | | $\{0, 2\lambda, 13\lambda+5, 14\lambda+7\}$ | |

如果要构造 $R=1/2$, $t=2$ 的扩散卷积码, 查表 11-8 中第二行, 子生成元列中的数字是 $(0, \lambda+1, 3\lambda+1, 4\lambda+1)$, $\lambda_{\min}=2$, 所以子生成元是 $(0, 3, 7, 9)$ 。这 4 个数字不仅表示子生成元多项式中系数取 1 的次数是 D^0, D^3, D^7 和 D^9 , 即子生成元为

$$\mathbf{g}^{(1,2)}(D) = 1 + D^3 + D^7 + D^9$$

而且也表示对 k_0 (这里为 1) 个信息元中每一个信息位 e_{0i} ($i=1, 2, \dots, k_0$) 的正交校验和式是由哪些 s_i 组成。这里 $(0, 3, 7, 9)$ 表示对 e_{01} 信息位的 4 个正交校验和式由 s_0, s_3, s_7 和 s_9 组成的。可知此码就是上面例举的 $(2, 1, 9)$ 自正交系统扩散卷积码。此码的扩散度 $l = \lambda n_0 = 4$, 当然也可选取更大的 λ 值使纠突发错误能力增加, 由此可知, λ 起着类似于码段交错卷积码中的交错度 i 的作用。

二、可正交扩散卷积码

除了用自正交码构造扩散卷积码外，也可利用可正交码构造扩散卷积码。

如 $R=1/2$ 的 $(2, 1, 7)$ 可正交系统卷积码，它的子生成元为

$$g^{(1,2)}(D) = 1 + D^2 + D^4 + D^7$$

相应的校验矩阵为

$$\mathbf{H} = \begin{bmatrix} 11 \\ 00 & 11 \\ 10 & 00 & 11 \\ 00 & 10 & 00 & 11 \\ 10 & 00 & 10 & 00 & 11 \\ 00 & 10 & 00 & 10 & 00 & 11 \\ 00 & 00 & 10 & 00 & 10 & 00 & 11 \\ 10 & 00 & 00 & 10 & 00 & 10 & 00 & 11 \end{bmatrix}$$

错误图样为

$$\mathbf{E} = (e_{01}e_{02}, e_{11}e_{12}, \dots, e_{71}e_{72})$$

相应的 4 个正交一致校验和式是：

$$\begin{aligned} s_0 &= e_{01} + e_{02} \\ s_2 &= e_{01} + e_{21} + e_{22} \\ s_4 + s_6 &= e_{01} + e_{42} + e_{61} + e_{62} \\ s_7 &= e_{01} + e_{31} + e_{51} + e_{71} + e_{72} \end{aligned}$$

不难验证，这 4 个正交一致校验和式满足定义 11.6.1 所提出的 $l=4$ 度扩散卷积码的要求，能纠正长为 $l=4$ 的突发错误或 $t=\lfloor J/2 \rfloor=2$ 个随机错误，所需的保障区间 $g_2=15$ ，比同类型自正交扩散卷积码的保障区间要小。一般，用可正交码构造的扩散码与用自正交码构造的扩散码相比，在同样码率和 l, t 下所需的保障区间要小，但在反馈大数逻辑译码时可能会引起误差传播。

表 11-9 中列出了 $R=1/2$ 的可正交扩散系统卷积码，表中各符号的意义与表 11-8 相同，扩散度 $l=\lambda n_0$ 。

如要构造 $t=2$ 的可正交扩散码，查表 11-9 中第一行， $\lambda_{\min}=2$ ，所以得到一个 $(2, 1, 7)$ 可正交码，它的一个子生成元为 $g^{(1,2)}(D)=1+D^2+D^4+D^7$ 。正交化规则列中的数字表示可正交校验和式是由 S 的哪些分量组成。如该码是 $(2\lambda+3\lambda)=(4+6)$ ，表示一个可正交校验和式由 $s_{2\lambda}+s_{3\lambda}=s_4+s_6$ 组成。其它三个是由 S 的分量直接得出，一般是 $g^{(1,2)}$ 列中的前两个数字，如该码 $g^{(1,2)}$ 中的前两个数字是 $0, \lambda$ ，表示两个正交校验和式是 s_0 和 $s_\lambda(s_2)$ ，另一个是由 $s_{3\lambda+1}(s_7)$ 得到，可知该码就是刚才列举的 $(2, 1, 7)$ 可正交扩散卷积码。

又例如构造一个能纠正 $t=3$ 个随机错误的可正交扩散卷积码。查 11-9 表中第二行， $\lambda_{\min}=5$ ，得到一个 $(2, 1, 27)$ 码，它的一个子生成元为 $g^{(1,2)}(D)=1+D+D^6+D^{17}+D^{24}+D^{27}$ ， $J=2t=6$ ，需六个正交校验和式，其中 2 个是 s_0, s_1 ，其余 4 个由表“正交化规则”中的数字 $(3\lambda+9, 2\lambda+3), (3\lambda+12, \lambda+4)$ 得到，分别是 $s_{3\lambda}+s_9=s_{15}+s_9, s_{10}+s_3, s_{15}+s_{12}$ 和 s_5+s_4 。该码是 $l=n_0\lambda=10$ 度可正交扩散卷积码，能用反馈大数逻辑译码方法，在一个译码约束

长度 $(m+1)n_0=56$ 个码元内纠正3个随机错误或长为10的突发错误，所需的保障区间是

$$g = (m + 1)n_0 - 1 \quad (11.6.1)$$

其值等于55。

表 11-9 $R=1/2, n_0=2$ 的可正交扩散系统卷积码

| t | m | λ_{\min} | $\mathbf{g}^{(1,2)}$ | 正交化规则 |
|-----|----------------|------------------|--|---|
| 2 | $3\lambda+1$ | 2 | {0, $\lambda, 2\lambda, 3\lambda+1$ } | $(2\lambda+3\lambda)$ |
| 3 | $3\lambda+12$ | 5 | {0, 1, $\lambda+1, 2\lambda+7, 3\lambda+9, 3\lambda+12$ } | $(3\lambda+9, 2\lambda+3)(3\lambda+12, \lambda+4)$ |
| 4 | $3\lambda+37$ | 17 | {0, 2, 3, $\lambda+3, 2\lambda+18, 2\lambda+23,$ $3\lambda+27, 3\lambda+37$ } | $(2\lambda+23, \lambda+8)$ $(3\lambda+27, 2\lambda+12, 2\lambda+7)$ $(3\lambda+37, \lambda+13)$ |
| 5 | $3\lambda+88$ | 44 | {0, 3, 4, 5, $\lambda+5, 2\lambda+40, 2\lambda+54,$ $3\lambda+60, 3\lambda+67, 3\lambda+88$ } | $(5, 1)(2\lambda+54, \lambda+19)(2\lambda+67,$ $\lambda+12)$ $(3\lambda+60, 2\lambda+11, 2\lambda+25)$ $(3\lambda+88, \lambda+33, \lambda+26)$ |
| 6 | $3\lambda+127$ | 120 | {0, 2, 3, 7, 8, $\lambda+8, 2\lambda+88,$ $2\lambda+118, 2\lambda+138, 3\lambda+147,$ $3\lambda+157, 3\lambda+217$ } | $(8, 4, 5, 6)(2\lambda+118, \lambda+38)$ $(2\lambda+138, \lambda+58, \lambda+28)$ $(3\lambda+147, 2\lambda+17, 2\lambda+67, 2\lambda+37)$ $(3\lambda+157, \lambda+18)$ $(3\lambda+217, \lambda+68, \lambda+78)$ |
| 7 | $3\lambda+374$ | 223 | {0, 6, 7, 9, 10, 11, $, \lambda+11,$ $2\lambda+141, 2\lambda+154, 2\lambda+245,$ $3\lambda+257, 3\lambda+296, 3\lambda+322,$ $3\lambda+374$ } | $(10, 3, 1)(11, 8, 4)(2\lambda+154, \lambda+24)$ $(2\lambda+245, \lambda+115, \lambda+102)$ $(3\lambda+257, 2\lambda+23, 2\lambda+127,$ $\lambda+114)$ $(3\lambda+296, \lambda+50)$ $(3\lambda+322, \lambda+76, \lambda+37)$ $(3\lambda+374, \lambda+89, \lambda+63)$ |

* § 11.7 加拉格尔(Gallager)码

Gallager 码(以下简称 Ga 码)是一种自适应码，它能适应信道错误情况的变化以纠正随机错误或突发错误。但是这种码没有一般的编码理论，仅只是一种编码方法，可用自正交码或可正交码组合得到。

它的 $R=1/2$ 码的编码器如图 11-7 所示。编码器由 $b+x+k$ (x 是与译码器的检测错误性能有关的参数) 级移存器组成，分两部分。左边 k 级组成了一个自正交码或可正交码编

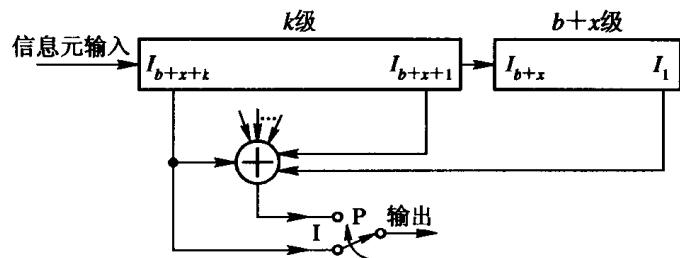


图 11-7 Ga 码编码器

码器，此码能纠正 t 个随机错误并能发现大量的其它错误，且 $b+x+k$ 级移存器中的信息元 I_1 也参加校验，所以编码器输出的第 $b+x+k$ 个校验元是 $P_{b+x+k} = I_{b+x+k} + \dots + I_{b+x+1} + I_1$ ，信息元与校验元交替传输。由此看出，该 Ga 码的编码约束长度是 $b+x+k$ ，这里 $b \gg k+x$ 。

该 Ga 码的译码器如图 11-8，它由编码器和一个 $b+k+x$ 级的伴随式移存器及其它控制电路组成。

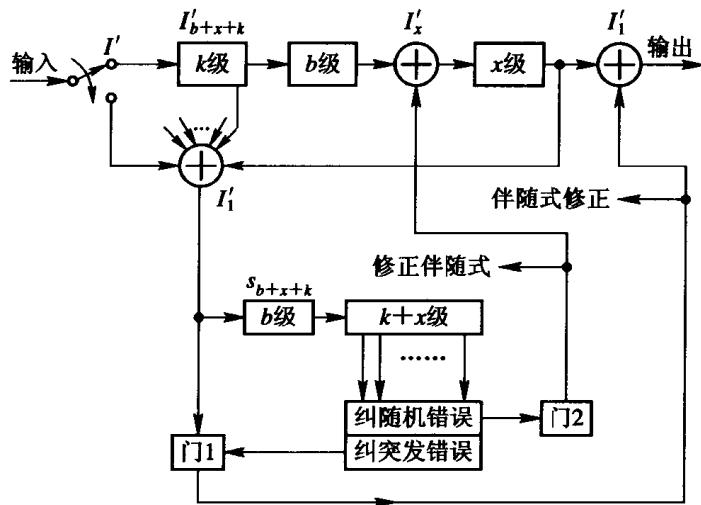


图 11-8 Ga 码译码器

译码器有两个工作状态：随机型与突发型。工作过程大致如下：

首先置译码器在随机型下工作，此时门 1 关闭，门 2 打开，译码器按一般的反馈大数逻辑译码方法纠正随机错误；若错误数目不超过 t ，则对 I'_{b+x+k} 信息元纠错。如果某一瞬间的错误超过了码的纠随机错误能力，但此错误图样能被译码器发现，这时译码器转入突发型下工作，门 1 打开，门 2 关闭，译码器在突发型下对 I'_{b+x+k} 码元纠错。由该译码器看出，伴随式分量 s_{b+x+k} 是

$$s_{b+x+k} = P'_{b+x+k} + I'_{b+x+k} + \dots + I'_{b+x+1} + I'_{b+x+k} \quad (11.7.1)$$

式中， P' 、 I' 分别表示译码器收到的校验元与信息元。若 $I'_{b+x+k} \neq I_{b+x+k}$ ，而：

$$P'_{b+x+k} = P_{b+x+k}$$

$$I'_{b+x+k} = I_{b+x+k}$$

⋮

$$I'_{b+x+1} = I_{b+x+1}$$

则

$$s_{b+x+k} = 1$$

若 $I'_{b+x+1} = I_{b+x+1}$, 则 $s_{b+x+k} = 0$ 。所以可由 s_{b+x+k} 的值确定 I'_{b+x+1} 有无错误。因此, 一旦发现突发错误以后, 只要突发在信息位中的错误位数不大于连续 b 位(总长 $\leq 2b$), 则由于此时突发已通过译码器最左边有抽头(也就是所应用的自正交码或可正交码的译码器)的 k 位, 也就不会再影响伴随式的计算。所以, 若此时刻以后输入到译码器的数据保持一段(与已发生的突发长度相当)时间无误, 就可根据式(11.7.1)求得的 s_{b+x+k} 的值对每个 I'_{b+x+1} 码元纠错。

当译码器在突发型下工作时, 不断检验伴随式寄存器右边 $k+x$ 级, 若某一时刻开始均为 0, 说明突发已全部通过译码器, 此时译码器就关闭门 1 打开门 2, 又转入随机型下工作。

由上述讨论可知, 这类译码器要求在突发错误后面的无误保障区间长度

$$g_G = 2b + 2(k + x) \quad (11.7.2)$$

位, 但由于 $b \gg k + x$, 故一般认为所要求的保障区间是 $2b$ 。

译码器中 x 级移存器的作用是使译码器从随机型转入突发型工作时有一缓冲时间, 以保证译码器能以最大的概率发现突发错误。因为若突发在 I'_{b+x+1} 码元位及以后的地方产生, 这时译码器可能还没有发现, 但只要突发还没有到达 I'_{b+x+1} 的地方, 则该突发还有机会被纠正。故 x 取值越大, 译码器不能发现突发错误的概率越小。由此可知, 这类码不能全部纠正长度 $\leq 2b$ 的突发错误, 因为有些突发图样可能不能被译码器发现。但如果所选用的自正交码或可正交码较好, 检测标准订得比较合适, 则不能检测的突发错误数目很小以致可以忽略。

从上面介绍可以看出 Ga 码是一类有误纠突发错误卷积码, 它的保障区间 g 与纠突发能力 b 之比是

$$\frac{g}{b} = \frac{2b + 2(k + x)}{2b} \approx \frac{2b}{2b} = 1$$

这里选用 $R=1/2$ 的码构造 Ga 码, 因此由式(11.3.6)可知, 对于有误纠突发错误卷积码的

$$\frac{g}{b} = \frac{R}{1-R} = 1$$

说明 Ga 码是一类接近最佳的有误纠突发错误卷积码。

习 题

1. 已知 $(3, 2, 13)$ 系统卷积码的两个子生成元是 $\mathbf{g}^{(1,2)}(D) = 1 + D^8 + D^9 + D^{12}$, $\mathbf{g}^{(2,3)}(D) = 1 + D^6 + D^{11} + D^{13}$ 。

- (1) 写出生成矩阵 \mathbf{G} 和 \mathbf{H} ;
- (2) 组成对 e_{01} 和 e_{02} 码元位正交的一致校验和方程;
- (3) 画出该码的编码电路。

2. 已知 $(2, 1, 5)$ 系统卷积码的子生成元是 $\mathbf{g}^{(1,2)}(D) = 1 + D^2 + D^4 + D^5$ 。

- (1) 写出码的 \mathbf{H} , 并找出对 e_{01} 码元位正交的校验和式;

- (2) 画出该码的编码电路和大数逻辑译码电路。
3. (2, 1, 5) 系统卷积码的子生成元是 $\mathbf{g}^{(1,2)} = (110101)$ 。
- (1) 此码是自正交码还是可正交码？它的最小汉明距离 $d_m = ?$
 - (2) 构造此码的大数逻辑译码器。
4. 已知(3, 1, 4)系统卷积码的子生成元是 $\mathbf{g}^{(1,2)}(D) = 1 + D$, $\mathbf{g}^{(1,3)}(D) = 1 + D^2 + D^3 + D^4$ 。
- (1) 写出码的 \mathbf{H} ;
 - (2) 找出对 e_{01} 码元位正交的正交校验和式；
 - (3) 画出该码的译码电路。
5. 把第 2 题的码按码段交错 3 次。
- (1) 写出交错码的 $\mathbf{G}(D)$ 和 $\mathbf{H}(D)$ ；
 - (2) 该码能纠正多长的单个突发错误？
 - (3) 画出该码的编码电路。
6. 构造 $\lambda = 1$, $n_0 = 4$, $k_0 = 3$ 的岩垂码。
- (1) 求出该码的 $\mathbf{G}(D)$ 和 $\mathbf{H}(D)$, 该码能纠正多长的突发错误？
 - (2) 画出该码的译码电路。
7. 构造能纠正 $b_1 \leqslant 8$ 的 B_1 型突发错误的岩垂码, $n_0 = 4$ 。
- (1) 找出该码的 $\mathbf{G}(D)$ 和 $\mathbf{H}(D)$, 计算保障区间 g 。
 - (2) 画出该码的译码电路, 叙述译码过程。
8. 一个(2, 1, m)系统卷积码的生成元 $\mathbf{g}^{(1,1)} = (1010010001)$ 。
- (1) 该码是自正交码吗？它纠随机错误能力有多大？
 - (2) 该码是扩散码吗？它纠突发错误的能力有多大？
 - (3) 构造该码的译码器。

参 考 文 献

- [1] [美]林舒、科斯特洛著：《差错控制编码：基础和应用》，王育民、王新梅译，人民邮电出版社，1986.
- [2] J. Justesen, "New Convolutional code constructions and a class of asymptotically good time-varying Codes", IEEE Trans. on IT, No 2, PP. 220—225, 1973.
- [3] M. Y. Rhee. , Error—Correcting Coding Theory, McGraw—Hill, 1990.
- [4] R. G. Gallager. , Information Theory and Reliable Communication, John Wiley and Sons, 1968.
- [5] 王新梅：《纠错码与差错控制》，人民邮电出版社，1989.

第十二章 卷积码的概率译码

前一章介绍的卷积码译码是基于码的代数结构基础上的代数译码。本章讨论的概率译码方法不仅基于码的代数结构基础上，而且还利用了信道的统计特性，因而能充分发挥卷积码的特点，使译码错误概率达到很小。

卷积码的概率译码最早始于 1961 年由乌曾格来夫(Wozencraft)提出的序列译码，这是第一个提出的实用的卷积码的概率译码方法，1963 年费诺(Fano)对序列译码进行改进，提出了 Fano 算法，从而推动了序列译码的实际应用。1967 年维特比(Viterbi)提出了另一种概率译码算法——Viterbi(以下简称 VB)算法，它是一种最大似然译码算法。在码的约束度较小时，它比序列译码算法效率更高、速度更快，译码器也较简单。因而自 VB 算法提出以来，无论在理论上还是在实践上都得到了极其迅速的发展，并广泛地应用于各种数传系统，特别是卫星通信系统中。本章首先介绍 VB 译码算法的工作原理和性能；然后介绍序列译码；最后介绍这几年得到极其迅速发展，并广泛应用于实际的调制与卷积码 VB 译码相结合的格形(网格)码调制技术(简称 TCM)。

§ 12.1 Viterbi(VB)译码算法的基本原理和实现

一、VB 译码算法的基本原理

Viterbi(VB)译码算法是一种最大似然译码算法。最大似然译码算法的基本概念已在第一章中详细介绍过，这里不再讨论。

卷积码的代数译码中，用 H_∞ 和 G_∞ 矩阵表示卷积码是很方便的。而在 VB 译码中，则利用时间状态图来描述卷积码的编码过程更为方便。我们仍以例 10.1 中的(2, 1, 2)卷积码为例说明。

(2, 1, 2)码的 $G(D) = [1 + D + D^2, 1 + D^2]$ 。它的编码器及相应状态图分别如图 10-9 和图 10-18 所示。虽然状态图能表示卷积编码器在不同输入的信息序列下，编码器各状态之间的转移关系，但并不能表示出编码器状态转移与时间的关系。为了表示这种状态与时间的关系，和码树图相似可以用篱笆或网格(Trellis)图来表示，如图 12-1 所示。此图是 $L=5$ 时，该(2, 1, 2)码的状态转移时间关系图，它由节点和分支组成，共有 $L+m+1$ 个时间单位(节点)，以 0 至 $L+m$ 予以标号。若编码器从 $s_0(00)$ 状态开始，并且结束于 s_0 状态，则最先的 $m=2$ 个时间单位(0, 1)，相当于编码器由 s_0 状态出发往各个状态行进，而最后 $m=2$ 个时间单位(6, 7)，相当于编码器由各状态返回到 s_0 状态。因而，在开始和最后 m 个时间单位，编码器不可能处于任意状态中，而只能处在某些特定状态(如 s_0, s_1)中之一，仅仅从第 m (2)至第 L (5)时间单位，编码器可以处于任何状态之中(即 4 个状态 s_0, s_1, s_2, s_3 中之任一个)。

正如第 10 章中所提到的，编码器从全为 0 的 s_0 状态出发，最后又回到 s_0 状态时所输出的码序列，称为结尾卷积码序列。因此，当送完 L 段信息序列后，还必须向编码器再送入 m 段全 0 序列，以迫使编码器回到 s_0 状态。

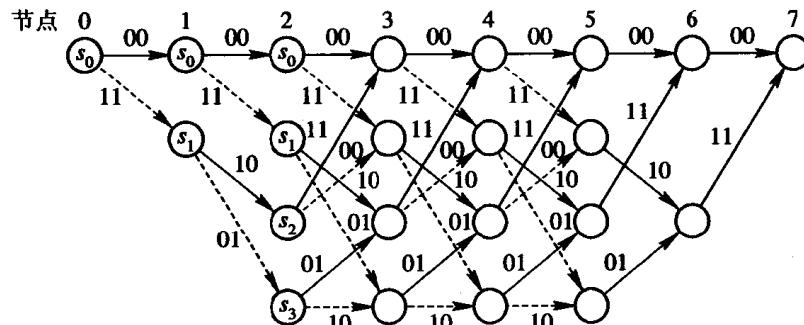


图 12-1 (2, 1, 2) 码 $L=5$ 时的篱笆图

篱笆图中每一状态有两个输入和两个输出分支。在某一时间单位(节点) i , 离开每一状态的虚线分支(下面分支), 表示输入编码器中的信息子组 $m_i = m_i = 1$; 而实线分支(上面分支)表示此时刻输入至编码器的信息子组 $m_i = m_i = 0$; 每一分支上的 $2(n_0)$ 个数字, 表示第 i 时刻编码器输出的子组 $c_i = (c_i^{(1)}, c_i^{(2)})$, 因而篱笆图中的每一条路径都对应于不同输入的信息序列。由于所有可能输入的信息序列共有 $2^{k_0 L}$ 个, 因而篱笆图中可能有的路径也有 $2^{k_0 L}$ 条, 相应于 $2^{k_0 L}$ 个长为 $N_c = (L+m)n_0$ 的不同码序列。

例如，输入至图中编码器的信息序列 $M = (1011100)$ ，则由编码器输出的码序列 $C = (11, 10, 00, 01, 10, 01, 11)$ ，它相应于篱笆图中用粗线表示的一条路径。

一般情况下, (n_0, k_0, m) 卷积码编码器共有 $2^{k_0 m}$ 个状态, 若输入的信息序列长度是 $Lk_0 + mk_0$ (后 mk_0 个码元全为 0), 则进入和离开每一状态的各有 2^{k_0} 条分支, 在篱笆图上有 $2^{k_0 L}$ 条不同的路径, 相应于编码器输出的 $2^{k_0 L}$ 个码序列。

编码器送出的码序列 C , 经过离散无记忆信道(DMC)传输后送入译码器的是序列 $R=C+E$, E 是信道错误序列。译码器根据接收序列 R , 按最大似然译码准则力图找出编码器在篱笆图上所走过的路径, 这个过程就是译码器计算、寻找最大似然函数

$$\max \log_b P(\mathbf{R} | \mathbf{C}_j) \quad j = 1, 2, \dots, 2^{k_0 L}$$

的过程，或者说译码器计算、寻找有最大“度量”的路径过程，即寻找

$$\max_j M(\mathcal{R}|\mathcal{C}_j) \quad j = 1, 2, \dots, 2^{k_0} \quad (12.1.1)$$

的过程。式中, $M(R|C_j) = \log_b P(R|C_j)$ 是 C_j 的自然函数也称为 C_j 的路径度量。

在 § 1.3 节中已证明, 对 BSC 信道而言, 计算和寻找有最大度量的路径, 等价于寻找与 R 有最小汉明距离的路径, 即寻找

$$\min_{\mathcal{R}} d(\mathcal{R}, C_j) \quad j = 1, 2, \dots, 2^{k_0 L} \quad (12.1.2)$$

对二进制输入 Q 进制输出的 DMC 信道而言，就是寻找与 R 有最小软距离的路径，此时的度量就是软判决距离：

$$\min_{\mathcal{R}_s} d_s(\mathcal{R}_s, \mathcal{C}_{js}) \quad j = 1, 2, \dots, 2^{k_0 L} \quad (12.1.3)$$

式中, \mathbf{R}_i 与 \mathbf{C}_j 是接收序列 \mathbf{R} 与 \mathbf{C}_j 序列的 Q 进制表示。

但是, 用上述这些方法译码是难以实现的。例如 $L=50$, $n_0=3$, $k_0=2$, 则共有 $2^{k_0 L}=2^{100}>10^{30}$ 个码序列(或篱笆图上的路径); 若 $m=5$, 则 $(L+m)=55$ 。如果在一秒钟内送出这 $k_0 L=100$ 个信息元, 则信息传输率只有 100 bit/s, 这是很低的。但即使是在如此低的信息速率下, 也要求译码器在一秒钟内计算、比较 10^{30} 个似然函数(或汉明距离、软距离), 这相当于要求译码器计算每一似然函数的时间小于 10^{-30} s, 这是根本无法实现的。更何况通常情况下 L 不是几十, 而是成百上千, 因此, 必须寻找新的最大似然译码算法。

VB 算法正是在解决上述困难中所引入的一种最大似然译码算法。它并不是在篱笆图上一次比较所有可能的 $2^{k_0 L}$ 条路径(序列), 而是接收一段, 计算、比较一段, 选择一段最可能的码段(分支), 从而达到整个码序列是一个有最大似然函数的序列。现把 VB 译码算法的步骤简述如下:

(1) 从某一时间单位 $j=m$ 开始, 对进入每一状态的所有长为 j 段分支的部分路径, 计算部分路径度量。对每一状态, 挑选并存贮一条有最大度量的部分路径及其部分度量值, 称此部分路径为留选(幸存)路径。

(2) j 增加 1, 把此时刻进入每一状态的所有分支度量, 和同这些分支相连的前一时刻的留选路径的度量相加, 得到了此时刻进入每一状态的留选路径, 加以存贮并删去其它所有路径, 因此留选路径延长了一个分支。

(3) 若 $j < L+m$, 则重复以上各步, 否则停止, 译码器得到了有最大路径度量的路径。

由时间单位 m 直至 L , 篱笆图中 $2^{k_0 m}$ 个状态中的每一个有一条留选路径, 共有 $2^{k_0 m}$ 条。但在 L 时间单位(节点)后, 篱笆图上的状态数目减少, 留选路径也相应减少。最后到第 $L+m$ 单位时间, 篱笆图归到全为 0 的状态 s_0 , 因此仅剩下一条留选路径。这条路径就是要找的具有最大似然函数的路径, 也就是译码器输出的估值码序列 $\hat{\mathbf{C}}$ 。由此可知, 在篱笆图上用 VB 译码算法得到的路径一定是一条最大似然路径, 因而这种译码方法是最佳的。

定理 12.1.1 在 VB 译码算法中, 留选路径 $\hat{\mathbf{C}}$ 是有最大似然函数的路径, 即

$$M(\mathbf{R}|\hat{\mathbf{C}}) \geq M(\mathbf{R}|\mathbf{C}) \quad \text{对所有的 } \mathbf{C} \neq \hat{\mathbf{C}}$$

这里 $M(\mathbf{R}|\mathbf{C})$ 是 \mathbf{C} 码序列(或路径)的度量, 它等于

$$\begin{aligned} M(\mathbf{R}|\mathbf{C}) &= \log_b P(\mathbf{R}|\mathbf{C}) = \sum_{i=0}^{L+m-1} M(\mathbf{R}_i|\mathbf{c}_i) \\ &= \sum_{i=0}^{L+m-1} \log_b P(\mathbf{R}_i|\mathbf{c}_i) \\ &= \sum_{i=0}^{L+m-1} \sum_{j=1}^{n_0} M(r_{ij}|\mathbf{c}_{ij}) \\ &= \sum_{i=0}^{L+m-1} \sum_{j=1}^{n_0} \log_b (r_{ij}|\mathbf{c}_{ij}) \end{aligned} \quad (12.1.4)$$

式中, $\log_b P(\mathbf{R}|\mathbf{C})$ 是 $\mathbf{C}=(\mathbf{c}_0, \mathbf{c}_1, \dots) = (c_{01}, c_{02}, \dots, c_{0n_0}, c_{11}, c_{12}, \dots, c_{1n_0}, \dots)$ 序列的似然函数。 $M(\mathbf{R}_i|\mathbf{c}_i)$ 是 \mathbf{C} 序列中第 i 个分支的度量, 它等于第 i 个分支的似然函数 $\log_b P(\mathbf{R}_i|\mathbf{c}_i)$ 。 $M(r_{ij}|\mathbf{c}_{ij})$ 是第 i 个分支中第 j 个码元的度量, 它等于第 j 个码元的似然函数。

证明 如图 12-2 所示。设有最大似然函数的路径 a , 在某一时刻 j , 进入某一状态如 s_3 时被删除掉了, 这说明留选路径 b 的度量值超过了 a 的度量值。所以, 最大似然路径的其

余部分(j 时间以后之路径)与此时刻的留选路径 b 相加,则得到的该路径总度量值超过最大似然路径的总度量值。但这与最大似然路径有最大度量值的定义相矛盾,因而在VB算法中,最大似然路径不会被删掉,所以,最后得到的一定是最似然路径。

由以上定理可知,在篱笆图上用VB译码算法寻找的路径,是一个最大似然路径,因而这种译码方法是最佳的。

例 12.1 输入至图10.9编码器的信息序列 $M=(1011100)$,由编码器输出的码序列 $C=(11, 10, 00, 01, 10, 01, 11)$,通过BSC送入译码器的序列 $R=(10, 10, 00, 01, 11, 01, 11)$ 有两个错误,求利用VB译码算法时译码器输出的估值信息序列 \hat{M} 和码序列 \hat{C} 。

基于图12-1的篱笆图,VB译码器译接收到序列 R 的过程示于图12-3中。图中画出了各时刻进入每一状态的留选路径及其度量值 d (最小汉明距离),以及与此相应的译码器估计的信息序列 \hat{M} 。当 $L+m=7$ 个时刻以后,4条留选路径只剩一条,它就是译码器输出的估值信息序列 $\hat{M}=(11, 10, 00, 01, 10, 01, 11)$,相应的估值信息序列 $\hat{M}=(1011100)$, R 中的两个错误得到了纠正。

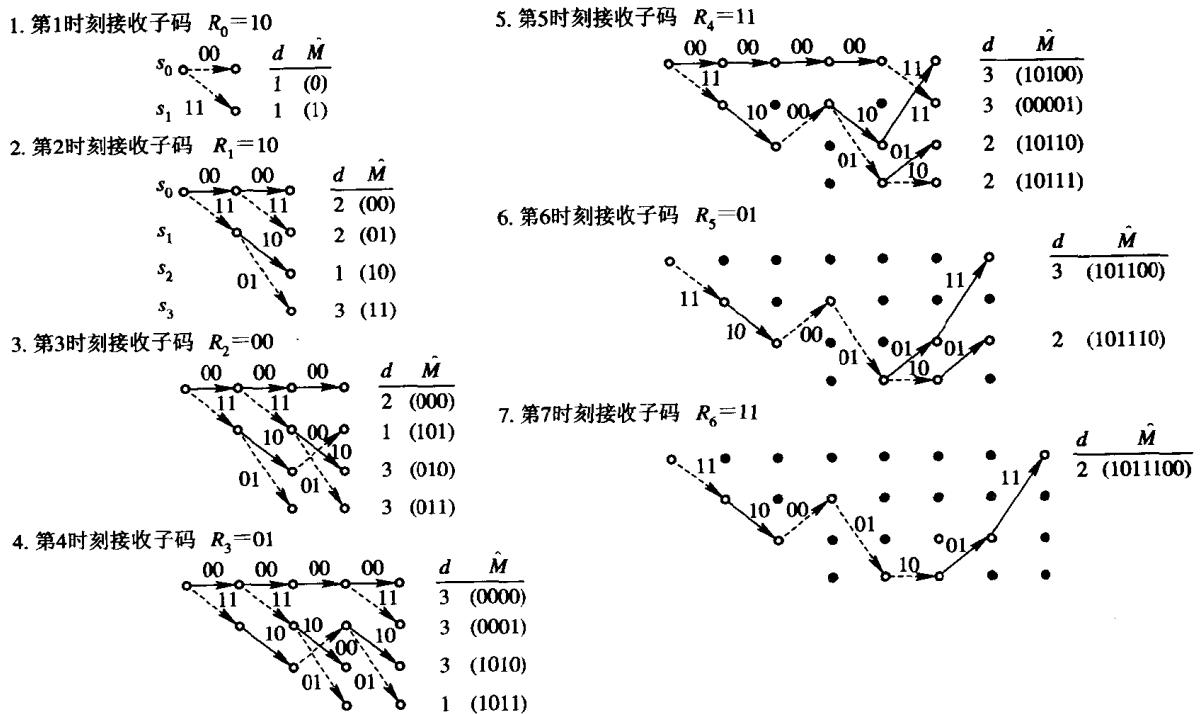


图 12-3 用 VB 译码算法译接收到序列 $R=(10, 10, 00, 01, 11, 01, 11)$ 时
在篱笆图上的译码过程

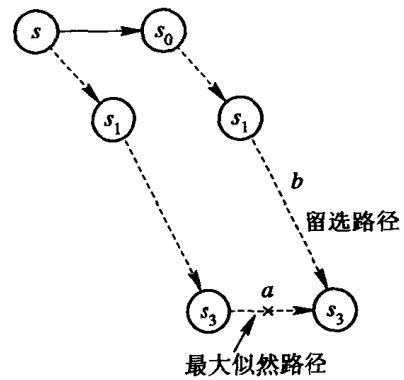


图 12-2 最大似然路径的删除

由图看出，在某一时刻，如 $j=3$ 时，进入 s_0 状态的留选路径的确定过程可叙述如下。进入 s_0 状态的有两条路径：一条是由(00)分支加上与此分支相连的前一时刻(第 2 时刻)的留选路径 $\hat{c}_{01}=(00, 00)$ 连接组成的路径 $(\hat{c}_{01}, 00)=(00, 00, 00)$, $d(\mathbf{R}_2, 00)=d(00, 00)=0$, 因而 $d(\hat{c}_{01}, 00, \mathbf{R}_0 \mathbf{R}_1 \mathbf{R}_2)=d(\hat{c}_{01}, \mathbf{R}_0 \mathbf{R}_1)+d(\mathbf{R}_2, 00)=2+0=2$, 所以该路径的度量值 d 是 2；另一条路径是由(11)分支加上与此分支相连的前一时刻(第 2 时刻)的留选路径 $\hat{c}'_{01}=(11, 10)$ 连接组成的路径 $(\hat{c}'_{01}, 11)=(11, 10, 11)$, 它的度量值 $d=d(\hat{c}'_{01}, 11, \mathbf{R}_0 \mathbf{R}_1 \mathbf{R}_2)=d(\hat{c}'_{01}, \mathbf{R}_0 \mathbf{R}_1)+d(\mathbf{R}_2, 11)=1+2=3$ 。根据最小汉明距离准则可得在第 3 时刻 s_0 的留选路径是 $\hat{c}_{012}=(00, 00, 00)$, 它的度量值 $d=2$ 。

在其它时刻及进入其余状态的留选路径的选择与此完全相同。若某一时刻进入某一状态的两条路径有相同的度量，如第 4 时刻，进入 s_2 状态的两条路径 $(11, 10, 00, 10)$ 和 $(00, 11, 01, 01)$ ，它们的度量值 d 均为 3，故可任选一条作为 s_2 状态的留选路径，在图中选择 $(11, 10, 00, 10)$ 。这种任意选的结果，并不会影响最后结果的正确性。■

能用有限状态组成的篱笆图描述其编、译码过程的码称为网格或篱笆码，也称为 Trellis 码，它是树码的一个大类。因此，卷积码是一类满足线性叠加原理、子生成元不随时间变化的线性网格码。

二、Viterbi 译码器框图

下面介绍一种硬判决(利用汉明距离 d 作为度量)的 Viterbi 译码器。由前面所述的译码过程不难明白，Viterbi 译码器应有以下特点：

(1) (n_0, k_0, m) 卷积码编码器共有 $2^{k_0 m}$ 个状态，因而 VB 译码器必须有同样的 $2^{k_0 m}$ 个状态发生器，并且对每一状态必须有一个路径寄存器以存贮路径或其信息序列 \hat{M} ，以及一个存贮路径度量值的存贮器。所以 VB 译码器的复杂性随 $2^{k_0 m}$ 指数增加。为了不致使译码器太复杂、成本太高，一般要求选用码的编码存贮 $m \leq 10$ 。

(2) 每个路径(或信息序列)存贮器存贮路径的长度是 $n_0 L$ (或 $k_0 L$)。这里， L 是需要存贮的码序列的总长度。若 L 很大，则译码器的存贮量太大而难以实用。由图 12-3 看到，当译完第 5 级节点之后，每个状态留选路径的前几个分支已完全重合到一起，这就告诉我们每个路径存贮器不必存贮 L 很大的码序列(或信息序列 \hat{M})，而只要存贮 $\tau \ll L$ 段子码即可。也就是当译码器接收并处理完第 τ 个码段后，译码器中的路径寄存器已全部存满，当译码器开始处理第 $\tau+1$ 个码段时，它必须对所有路径寄存器中的第一段信息元作出判决并输出。像这种不等处理完所有 L 段码序列，译码器就开始进行判决和输出的译码方法称为 Viterbi 译码的截尾译码，译码器称为截尾译码器。显然，截尾译码器的复杂性比非截尾的要大大减少，但其性能可能稍差，可是如果 τ 选择足够大，则对译码器输出的译码错误概率影响很小。一般情况下选

$$\tau = (5 \sim 10)m \quad (12.1.5)$$

就足够了。

下面的问题是：译码器如何判决并输出所有路径寄存器中的第一段信息元？这有以下几个方法：

(1) 任选一条留选路径的寄存器，把它的第一段信息元作为译码器输出。

(2) 把所有 $2^{k_0 m}$ 个路径寄存器的第一段信息元取出，按大数准则输出第一段信息元。

(3) 在 2^{k_0m} 个路径寄存器中，挑选一个具有最大路径度量的路径，以它的路径寄存器的第一段信息元作为译码器的输出。

(4) 对路径的度量值定出一个门限，当某一路径的门限超过此值(若为最小距离准则，则小于门限)，就输出此路径的第一段信息元。

在上述4种方法中可任选一种。当译码器判决并输出第一个码段的信息元后，译码器就把第 $\tau+1$ 码段的信息元存入路径寄存器的最末级。当译码器接收第 $\tau+2$ 码段时，译码器就判决并输出第二码段(它已处在路径寄存器的最前级)，如此等等。由此可知，这种译码方法的译码器所引入的译码迟延(或译码存贮)为 τ 个码段的时间。

图12-4给出了(2, 1, 2)卷积码VB截尾译码器的框图。可知一个硬判决($Q=2$)的VB译码器必须有以下几部分：

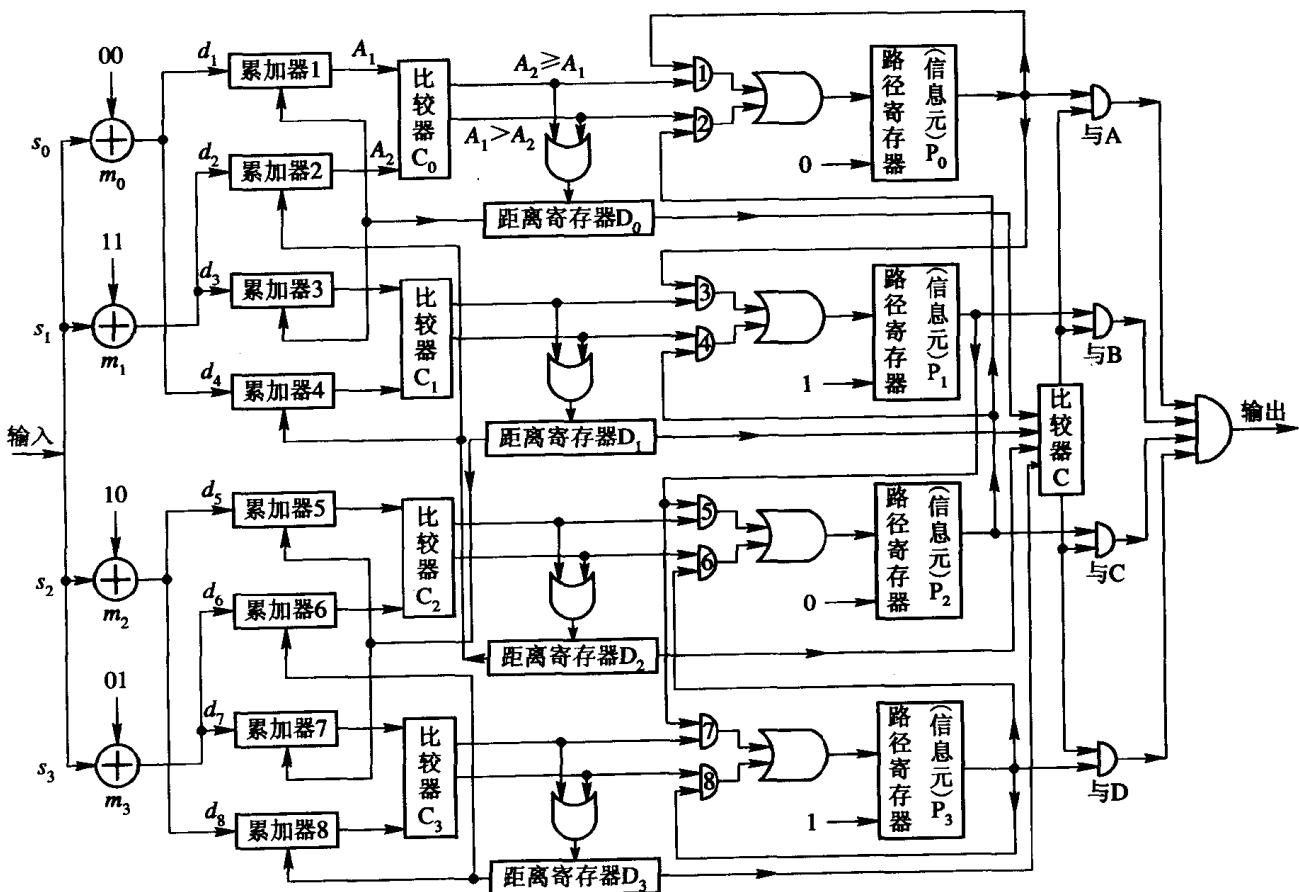


图 12-4 (2, 1, 2) 码硬判决 VB 平行译码器

- (1) 状态发生器：产生 2^{k_0m} 个状态和分支值。本例中产生 $2^{k_0m}=2^2=4$ 个状态。
- (2) 累加器：进行距离(或度量值)累加。
- (3) 比较器：比较各条路径的距离大小。
- (4) 距离寄存器(度量值寄存器)：存贮路径的距离(或度量值)。
- (5) 路径寄存器(或信息序列寄存器)：存贮留选路径，它的长度为 $n_0\tau=n_0(5 \sim 10)m$ 级或 $k_0\tau$ 级。
- (6) 判决器：决定输出信息元的值。
- (7) 其它控制电路等。

能完成(2)、(3)功能以及选择留选路径功能的电路称为 ACS 电路，显然 ACS 电路是平行 VB 译码器的最关键电路，共需要 $2^{k_0}m$ 个。

译码器从上至下由完全相同的 4 部分电路组成(所以称这种译码器为平行 VB 译码器)，它们的作用是分别求 4 个状态的 4 条留选路径。现以最上面一路(求 s_0 状态的留选路径)为例说明译码器的工作过程。

(1) 状态发生器(图中没有画出)产生各分支值:(00), (11), (10)和(01)，每一单位时间分别送入相应的模 2 加法器，如(00)送入 m_0 模 2 加法器。

(2) 在 m_0 模 2 加法器，每一单位时间(一个码段长)译码器把接收到的码段与状态产生器送出的分支值(00)相比较，得到分支距离 d_1 送入累加器 1，与以前的留选路径距离(由距离寄存器 D_0 送来)进行累加，得到 A_1 送入比较器 C_0 。

(3) 在模 2 加法器 m_1 ，译码器接收的码段与状态产生器产生的分支(11)相比较得到 d_2 ，它在累加器 2 与由距离寄存器 D_2 送出的状态 s_2 的留选路径的距离进行累加，得到 A_2 送入比较器 C_0 。

(4) 在比较器 C_0 中， A_1 与 A_2 进行比较，若 $A_2 \geq A_1$ ，把 A_1 送入 D_0 并消去 D_0 中原来的值，同时控制与门 1 送入一个“0”信息元给路径寄存器 P_0 。若 $A_1 > A_2$ ，则 A_2 送入 D_0 代替 D_0 中原来的值，同时控制与门 2，把 s_1 状态的留选路径寄存器 P_1 中的序列全部送入 P_0 ，代替 P_0 中原来的序列，并送入一个“0”附加在该序列的后面。

(5) 在比较器 C 比较 4 个距离寄存器中的值，若 D_0 中的值最小，则比较器 C 送出信号控制与门 A 打开，送出路径寄存器 P_0 中第一级的信息元，通过或门送给用户。

由以上译码过程看到，这种平行译码器是按码段逐段译码，同时计算所有 $2^{k_0}m = 4$ 个状态的留选路径，因而译码速度可以做得很髙，可达几十兆至几百兆比特每秒(Mbit/s)并便于用大规模集成电路做成芯片。目前在国际市场上已有几十 Mbit/s 的 VB 译码器芯片出售，且价格只有几百美元，而速率为 256 Kbit/s 的芯片价格只有约 90 美元。所以这种平行 VB 译码器特别适用于高速、大容量数传通信系统，如卫星通信中。

此外，这种译码器的同步比较简单，只要求分支(节点)同步即可，并且对接收机的自动增益控制(AGC)和信号相位的抖动并不敏感。正是由于这些特点，使得 VB 译码器得到了日益广泛的应用。

三、软判决 VB 译码

图 12-4 给出的是硬判决 VB 译码器框图，适用于 BSC。为了充分利用信道输出信号的信息，提高译码可靠性，往往把信道输出的信号进行 $Q(>2)$ 电平量化，然后再输入 VB 译码器。能适应于这种 Q 进制输入的 VB 译码器称为软判决 VB 译码器，也就是适用于 DMC 的译码器。

正如式(12.1.3)所表明的，软判决 VB 译码器就是寻找与接收序列 R 有最小软判决距离的路径。因此，如果用最小软判决距离代替汉明距离作为选择幸存路径和译码器输出的准则，则软判决 VB 译码器的结构和译码过程，完全与硬判决的相同，只要在 R 与 C 中用 Q 进制的值代替二进制值即可，这里不再重复。

正如 § 6.5 中所指出的，在二进制输入 Q 进制输出的 DMC 中，码元的量化电平值 $j (0 \leq j \leq Q-1)$ ，近似地反映了码元似然函数的值。如果我们在计算码元的似然函数时，不

用这种近似的电平值，而用较为精确的计算方法，则得到了以下的软判决译码准则和码元度量。

由定理12.1.1中可知，码元(或比特)度量

$$M(r_{ij}|c_{ij}) = \log_b P(r_{ij}|c_{ij})$$

从实用考虑，应用整数比用对数更加方便。用以下式子代替上式：

$$M(r_{ij}|c_{ij}) = \log_b P(r_{ij}|c_{ij}) = a_2(\log_b P(r_{ij}|c_{ij}) + a_1) \quad (12.1.6)$$

式中， a_1 是任意实数， a_2 是某一正实数。可知寻找有最大似然函数的路径就是计算

$$\max_j M(\mathbf{R}|\mathbf{C}_d) = \max_j \sum_{i=0}^{N_c-1} \log_b P(r_{ij}|c_{ij}) = \max_j \sum_{i=0}^{N_c-1} a_2(\log_b P(r_{ij}|c_{ij}) + a_1) \quad (12.1.7)$$

式中， $N_c=n_0(L+m)-1$ 。

可知用比特修正度量 $a_2(\log_b P(r_{ij}|c_{ij}) + a_1)$ 代替比特度量 $\log_b P(r_{ij}|c_{ij})$ ，并不会影响 VB 译码器的性能。若 a_1 是选择得使最小的比特度量值是 0，而 a_2 选择得使所有比特度量值近似地用整数表示，则对一个具体的 DMC，选择不同的 a_2 ，就得到不同的整数度量的集合。对比特度量进行这种整数化修正，对 VB 译码器性能的影响极小。下面通过一例说明这种软判决 VB 译码的过程。

一个二进制输入， $Q=4$ 进制输出的 DMC

如图 12-5 所示。表 12-1(a) 中给出了以 10 为底的该信道的对数比特度量。若选 $a_1=1$, $a_2=17.3$ ，则得到相应的整数度量表如表 12-1(b) 所示，可知这是一种非均匀的四电平量化。

设送入该 DMC 的码序列

$$\mathbf{C} = (11, 10, 00, 01, 10, 01, 11)$$

输出并送入 VB 译码器的序列

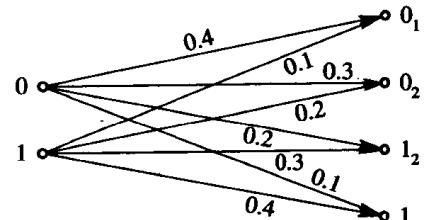


图 12-5 二进制输入： $Q=4$ 进制输出的 DMC

$$\mathbf{R} = (1_2 0_1, 1_1 0_1, 0_1 0_2, 0_1 1_1, 1_1 1_2, 0_2 1_1, 1_2 1_2)$$

表 12-1 图 12-5 信道的比特度量表

| $c_i \backslash r_i$ | 0_1 | 0_2 | 1_2 | 1_1 | $r_i \backslash c_i$ | 0_1 | 0_2 | 1_2 | 1_1 |
|----------------------|-------|-------|-------|-------|----------------------|-------|-------|-------|-------|
| 0 | -0.4 | -0.52 | -0.7 | -1 | 0 | 10 | 8 | 5 | 0 |
| 1 | -1 | -0.7 | -0.52 | -0.4 | 1 | 0 | 5 | 8 | 10 |

(a)

(b)

在篱笆图上利用 VB 译码算法译码输出判决序列的过程如图 12-6。 $j=7$ 个单位时间后，译码器对 4 条留选路径按最大似然函数值(度量值)进行判决，得到译码器输出的估值序列： $\hat{\mathbf{C}}=(11, 10, 00, 01, 10, 01, 11)$ ，相应的信息元估值序列 $\hat{\mathbf{M}}=(1011100)$ ，这条路径有度量值 115。如果接收序列与发送序列完全一致，则最大可能的度量值是 140，所以 $\hat{\mathbf{M}}$ 与最大可能的度量值相差 35。

如果把该例中的 DMC 化成 BSC，则相当于把 DMC 输出的 $0_1, 0_2$ 和 $1_1, 1_2$ 分别划分成 BSC 输出的 0 和 1，故此 BSC 的转移概率是 0.3。例 12.1 中给出了与该例有相同输入和输出序列情况下，硬判决 VB 译码器输出的信息估值序列 $\hat{\mathbf{M}}=(1011100)$ ，这条路径的度量值仅

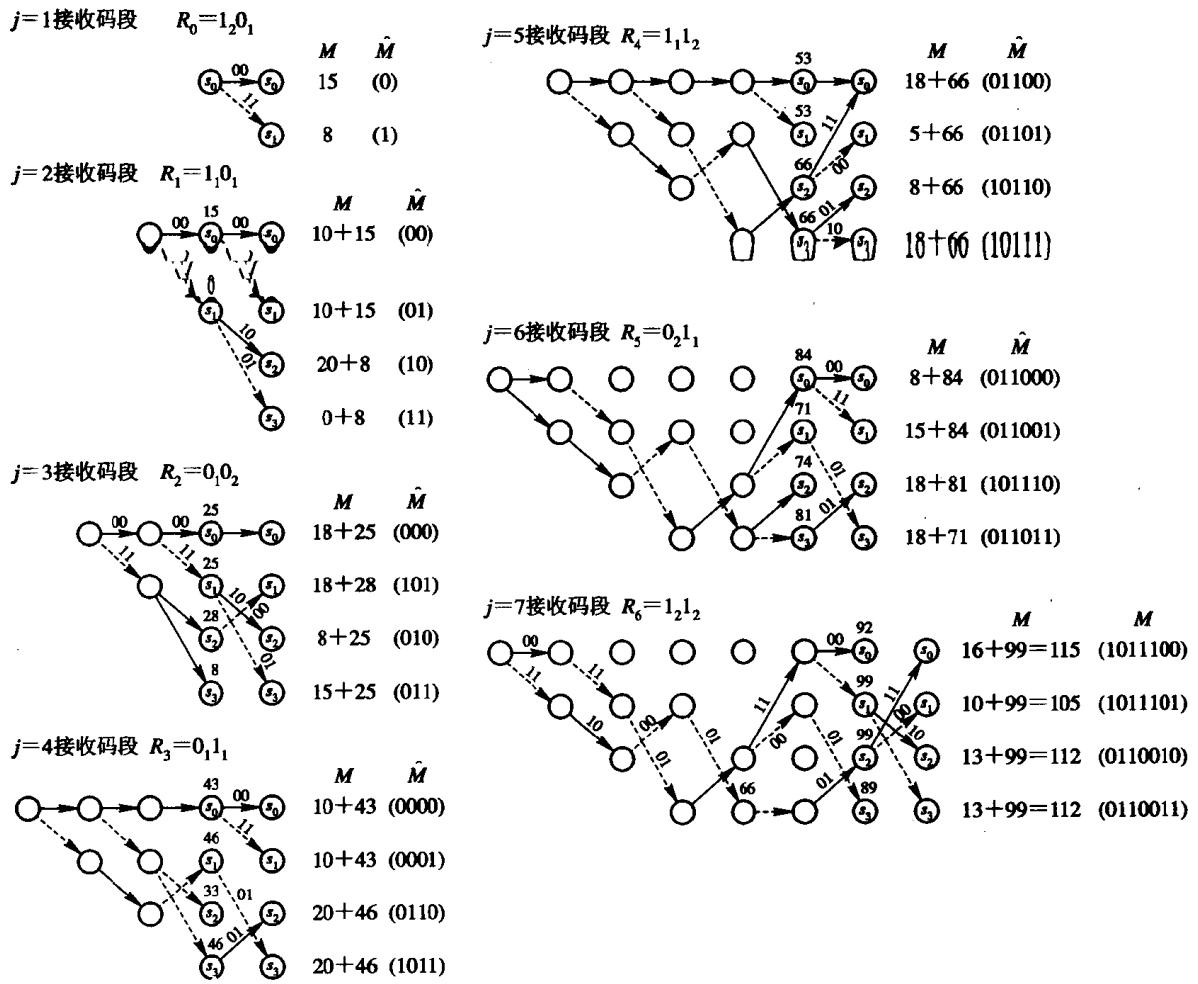


图 12-6 软判决 VB 译码接收序列

$R = (1_2 0_1, 1_1 0_1, 0_1 0_2, 0_1 1_1, 1_1 1_2, 0_2 1_1, 1_2 1_2)$ 的过程

为 105，比软判决 VB 译码器所判决的序列 115 要小 10。虽然这两个译码器最终输出的信息估值序列 \hat{M} 均相同，但由于软判决 VB 译码器的输出序列比硬判决输出的有更高的度量值，因此软判决 VB 译码器的性能更好。

由于软判决 VB 译码器的结构并不比硬判决的复杂，但性能却要好 2~3 dB，因此目前实用中的 VB 译码器几乎都用软判决，并且一般采用八电平均量化，其性能基本上达到了最大似然译码的性能。

从以上讨论可以看出，全并行 VB 译码器的复杂性随 2^{k_m} 增长。为了减少译码器的复杂性和运算量，很多作者提出了许多实现 VB 译码的简化译码算法。例如，只用 $2^l < 2^{k_m}$ 个 ACS 电路的部分并行时分复用译码器，当然其译码速度比全并行的要慢；又如，还提出了为避免不断改写路径寄存器内数据的寻迹法和分段译码^[3, 4]等。

为了便于用超大规模集成电路实现，提高芯片的集成度，1987 年有人提出了能减少译码器门器件数目，减小功耗的少状态转移(SST)译码算法^[5]。这种算法是把接收到的码序列 $R(D)$ 经过 $G(D)$ 的逆 $G^{-1}(D)$ 进行预译码，得到 $\tilde{M}(D)$ ，然后再通过编码器编码得到

$\tilde{R}(D)$, 并与接收的 $R(D)$ 比较得到一个序列 $I(D)$, 把它送入 VB 译码器译码。由于 $I(D)$ 中绝大部分是 0 数据, 因此使 VB 译码器绝大部分时间处于全 0 的 s_0 状态, 从而减少了状态转移。

1990 年有人提出了一种新的变换译码算法^[6]。这种算法能并行处理动态格搜索, 并把它归结为一个简单的矩阵运算, 从而使译码过程中的存贮器和译码过器中的复杂的数据交换大为减少, 导致译码器复杂性的减低。除了上述这些改进方法以外, 还有其它一些方法, 由于篇幅所限, 这里不再介绍, 有兴趣的读者请参阅有关文献^[7]。

§ 12.2 Viterbi 译码算法的性能

本节将首先介绍 BSC 信道中, VB 译码器输出的误码率, 然后讨论 DMC 情况下, VB 译码算法的性能。由于性能分析牵涉到码的重量分布和生成函数等概念, 因此这里仅给出主要结论而不作分析。最后, 将列出适用于 VB 译码算法的各类码及其参数表。

一、BSC 情况下 VB 译码算法的性能

详细分析 VB 译码器输出的误码率是很困难的, 下面仅给出译码错误概率的上限。

对某一给定的 (n_0, k_0, m) 卷积码, 它的 VB 译码器在 j 时刻前均正确译码, j 时刻产生首次错误译码, 称产生这种事件的概率为首次错误译码概率 P_{fej} 。若 BSC 的信道转移概率为 P , 则

$$P_{\text{fej}} < \sum_{d=d_f}^{\infty} A_d P_d \quad (12.2.1)$$

式中, A_d 是给定卷积码中的重量为 d 的码序列数, 由式(10.6.2)知, 它由码的生成函数

$$T(x) = \sum_{d=d_f}^{\infty} A_d x^d \quad (12.2.3)$$

决定。式(12.2.1)中的 d_f 是码的自由距离, 而

$$P_d = \begin{cases} \sum_{i=(d+1)/2}^d \binom{d}{i} p^i (1-p)^{d-i} & d \text{ 奇数} \\ \frac{1}{2} \binom{d}{d/2} p^{d/2} (1-p)^{d/2} + \sum_{i=d/2+1}^d \binom{d}{i} p^i (1-p)^{d-i} & d \text{ 偶数} \end{cases} \quad (12.2.3)$$

可以证明, 对任何 d , 上式可化成

$$P_d < 2^d p^{d/2} (1-p)^{d/2} \quad (12.2.4)$$

把它代入式(12.2.1)可得

$$P_{\text{fej}} < \sum_{d=d_f}^{\infty} A_d (2 \sqrt{p(1-p)})^d$$

该式对任何时刻都正确, 因而在任何时刻, 译码器产生首次错误事件概率 $P_{\text{fe}} = P_{\text{fej}}$, 即

$$P_{\text{fe}} < \sum_{d=d_f}^{\infty} A_d (2 \sqrt{p(1-p)})^d \quad (12.2.5)$$

与式(12.2.2)比较可知:

$$P_{\text{fe}} < T(x) \Big|_{x=2 \sqrt{p(1-p)}}$$

得到了译码器首次错误事件概率 P_{fe} , 就可得到在任何时刻, 译码器产生错误事件的概率 P_E 为

$$P_E \leq P_{fe} < \sum_{d=d_f}^{\infty} A_d P_d < T(x) \Big|_{x=2\sqrt{p(1-p)}} \quad (12.2.6)$$

若 p 很小, 则 P_E 主要由和式的第一项决定, 故

$$P_E \approx A_{d_f} (2\sqrt{p(1-p)})^{d_f} \approx A_{d_f} 2^{d_f} p^{d_f/2} \quad (12.2.7)$$

例如对于图 10-9 中所给出的(2, 1, 2)码, $d_f=5$, $A_{d_f}=1$, 若 BSC 的转移概率 $p=10^{-2}$, 则 VB 译码器产生错误译码事件的概率

$$P_E \approx 2^5 p^{5/2} = 3.2 \times 10^{-4}$$

有了 P_E 后, 可以求得 VB 译码器输出的信息元误码率(以下简称误码率) P_{ME} 。可以证明当信道的转移概率 p 很小时^[1], VB 译码器输出的误码率

$$P_{ME} \approx \frac{1}{k_0} B_{d_f} (2\sqrt{p(1-p)})^{d_f} \approx \frac{1}{k_0} B_{d_f} 2^{d_f} p^{d_f/2} \quad (12.2.8)$$

式中, B_{d_f} 是 (n_0, k_0, m) 卷积码码字中, 所有重量为 d_f 码字(路径)的非零信息位的总数。

如以上例举的(2, 1, 2)码, $d_f=5$, $B_{d_f}=1$, 当 $p=10^{-2}$, 则 VB 译码器输出的误码率

$$P_{ME} \approx 2^5 p^{5/2} = 3.2 \times 10^{-4}$$

与错误译码事件概率相同。这说明 p 很小时, 最可能的错误译码事件是重量为 5 的路径被错译, 引起一个信息元的错误。一般说来, 每产生一次错误译码事件, 便产生一个信息元的译码错误。

二、加性高斯白噪声信道(AWGN)中 VB 译码器输出的误码率

下面讨论一个比较实际的信道中, VB 译码器的性能。设输入信道的信号是二相移相键控(BPSK), 信道中的噪声是加性高斯白噪声, 信道的输出量化成二进制。某些微波和卫星信道就可看成是这类信道。这种信道的误码率 p 由数字通信理论可知为

$$p = \frac{1}{2\pi} \int_{\frac{2E}{N_0}}^{\infty} e^{-\frac{x^2}{2}} dx \approx \frac{1}{2} e^{-\frac{E_s}{N_0}}$$

式中, E_s 是信道中传输的每一符号的能量, N_0 是单边噪声功率谱密度。

由式(12.2.8)可知, 对一个有自由距离为 d_f 的卷积码, 在这种信道中 VB 译码器输出的误码率是

$$P_{Me} \approx \frac{1}{k_0} B_{d_f} 2^{d_f} p^{d_f/2} = \frac{1}{k_0} B_{d_f} 2^{d_f/2} e^{-(d_f/2)(E_s/N_0)} \quad (12.2.9)$$

这里要求 p 很小, 也就是要求 E_s/N_0 很大。设 $E_b=E_s/R$ 定义为每个信息比特的能量, 因为 $1/R$ 是传输每一信息 bit 所需的符号数, 则式(12.2.9)可写成

$$P_{Me} \approx \frac{1}{k_0} B_{d_f} 2^{d_f/2} e^{-(Rd_f/2)(E_b/N_0)} \quad (12.2.10)$$

这里同样要求信噪比 E_b/N_0 很大。

另一方面, 若不应用编码, $R=1$ 。BSC 信道的转移概率 p 就是信道输出的误码率, 因而

$$P_{Me} = p \approx \frac{1}{2} e^{-E_b/N_0} \text{(无编码)} \quad (12.2.11)$$

比较式(12.2.10)和式(12.2.11)可以看到, 对固定的 E_b/N_0 , 有编码的 P_{Me} 中 e 指数项比无编码时的要大($Rd_f/2$)因子。由于在大 E_b/N_0 时误码率主要由 e 指数项决定, 因而我们定义

$$\gamma = 10 \lg(Rd_f/2) \text{(dB)}$$

称为渐近编码增益(或纯编码增益)。它说明使用硬判决 VB 译码比不使用编码, 在同样的信息传输速率和输出误码率下, 功率要省 γ dB。应当指出, 当信噪比 E_b/N_0 减小时, 编码增益 γ 也减小。这是由于当 E_b/N_0 减小时, 有可能使得 R 大于信道容量 C , 从而不能保证用编码进行可靠通信, 因此有可能不用编码的系统反而更好。

若 AWGN 信道输出不采用二电平量化, 而是模拟信号输出(相当于 Q 为无限时的无限电平量化情况), 则 VB 译码器输出的误码率可以证明为

$$P_{ME} \approx \frac{1}{k_0} B_{df} (e^{-R(E_b/N_0)})^{df} = \frac{1}{k_0} B_{df} e^{-(Rd_f)(E_b/N_0)} \quad (12.2.12)$$

这就是无限量化时软判决 VB 译码器输出的误码率。与式(12.2.10)比较可知, 式(12.2.12)中的指数项要比式(12.2.10)中的少一个因子(1/2), 这说明 AWGN 信道比 BSC 信道, 在使用 VB 译码器情况下, 有 3 dB 的功率增益。这就是应用无限量化(模拟信号输出)软判决 VB 译码器比用硬判决(二电平量化)译码器, 能得到的量大软判决增益。当然, 使用这种模拟信号输入的 VB 译码器的结构, 比硬判决译码器要复杂得多。

若 AWGN 信道的输入是二进制信号序列, 输出是 Q 电平量化序列(即 DMC 情况), 则 Q 进制输入的软判决 VB 译码器输出的误码率, 在大的 E_b/N_0 下为

$$P_{ME} \approx \frac{1}{k_0} B_{df} \left(\sum_{j=1}^Q \sqrt{p(j|0)p(j|1)} \right)^{df} \quad (12.2.13)$$

式中, $p(j|0)$ 和 $p(j|1)$ 分别是 0 错成 j 和 1

错成 j 的概率。在硬判决情况下, 式(12.2.13)就成为式(12.2.8)。比较式(12.2.13)与式(12.2.8), 可得到在不同量化电平 Q 下, 软判决 VB 译码器所得的软判决增益(或称量化增益)。

图 12-7 给出了用计算机模拟不同量化电平时, VB 译码器的输出误码率与 E_b/N_0 之关系曲线。这些曲线说明, 八电平量化比二电平(硬判决)量化能得到 2 dB 的软判决增益, 而大于八电平量化后, 软判决增益增加很慢。因此, 一般实用上均采用八电平或十六电平量化, 这时译码器也不太复杂, 且有 2~3 dB 的软判决增益。

正由于软判决 VB 译码器比硬判决译码有 2~3 dB 的增益, 且译码器的结构并不比硬判决复杂多少, 因此一般用户均愿意采用

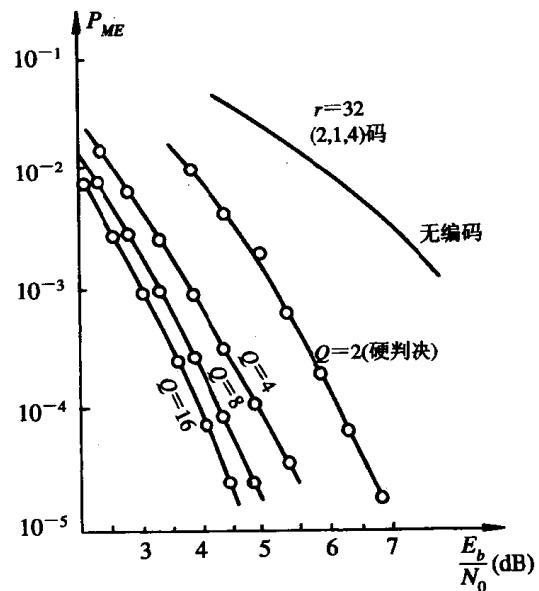


图 12-7 不同 Q 时用计算机模拟 VB 译码器的性能曲线

软判决译码，而很少应用硬判决，特别是在卫星通信中更是如此。现在软判决的 VB 译码技术，几乎已成为标准技术而广泛应用于卫星通信和其它通信系统中。

§ 12.3 适用于 VB 译码算法的码和剩余码

由式(12.2.8)、式(12.2.10)、式(12.2.12)和式(12.2.13)可知，无论是软判决还是硬判决 VB 译码器，它的输出误码率 P_{ME} 主要由所用码的自由距离 d_f 决定，随着 d_f 的增加 P_{ME} 指数下降，当然也与 A_{d_f} 与 B_{d_f} 有关。这说明适用于 VB 算法的码必须有尽可能大的自由距离，其次要求有小的 A_{d_f} 和 B_{d_f} 。由于 VB 译码器的复杂性随编码贮存 m (或约束度)指数增长，因此要求选用的码其 m 不能太大。当然，所有这些码必须无恶性误差传播。

一、适用于 VB 译码的码

用代数方法构造有大自由距离、小 A_{d_f} 和 B_{d_f} ，无恶性误差传播以及约束度不大的卷积码是很困难的，因此往往借助于计算机搜索具有上述性能的好码。表 12-2(a)、(b)、(c)、(d)、(e) 给出了码率分别为 $1/2, 1/3, 1/4, 2/3, 3/4$ 时，用计算机搜索得到的，具有上述性能、适用于 VB 译码算法的好卷积码。码的子生成元用八进制表示，如 $(2, 1, 5)$ 码，它的子生成元 $\mathbf{g}^{(1, 1)}$ 栏下的数字为“65”， $\mathbf{g}^{(1, 2)}$ 栏下的数字为“57”，则

$$\mathbf{g}^{(1, 1)} = (110, 101)$$

$$\mathbf{g}^{(1, 2)} = (101, 111)$$

因而

$$G(D) = [1 + D + D^3 + D^5, 1 + D^2 + D^3 + D^4 + D^5]$$

表 12-2(a) 具有最大 d_f 的 $R=1/2$ 码

| m | $\mathbf{g}^{(1, 1)}$ | $\mathbf{g}^{(1, 2)}$ | d_f | $\gamma(\text{dB})$ | τ_{\min} |
|-----|-----------------------|-----------------------|-------|---------------------|---------------|
| 2 | 5 | 7 | 5 | 0.97 | 8 |
| 3 | 64 | 74 | 6 | 1.76 | 10 |
| 4 | 46 | 72 | 7 | 2.43 | 15 |
| 5 | 65 | 57 | 8 | 3.01 | 19 |
| 5 | 73 | 61 | 8 | 3.01 | 19 |
| 6 | 554 | 744 | 10 | 3.98 | 27 |
| 6 | 171 | 133 | 10 | 3.98 | 27 |
| 7 | 712 | 476 | 10 | 3.98 | 28 |
| 8 | 561 | 753 | 12 | 4.77 | 33 |
| 9 | 4734 | 6624 | 12 | 4.77 | |
| 10 | 4672 | 7542 | 14 | 5.44 | |
| 11 | 4335 | 5723 | 15 | 5.74 | |
| 12 | 42554 | 77304 | 16 | 6.02 | |
| 13 | 43572 | 56246 | 16 | 6.02 | |
| 14 | 56721 | 61713 | 18 | 6.53 | |
| 15 | 447254 | 627324 | 19 | 6.77 | |
| 16 | 716502 | 514576 | 20 | 6.99 | |

τ_{\min} 为截尾译码器中路径寄存器最短的级数

表 12-2(b) 具有最大 d_f 的 $R=1/3$ 码

| m | $\mathbf{g}^{(1, 1)}$ | $\mathbf{g}^{(1, 2)}$ | $\mathbf{g}^{(1, 3)}$ | d_f | $\gamma(\text{dB})$ |
|-----|-----------------------|-----------------------|-----------------------|-------|---------------------|
| 2 | 5 | 7 | 7 | 8 | 1.25 |
| 3 | 54 | 64 | 74 | 10 | 2.21 |
| 4 | 52 | 66 | 76 | 12 | 3.01 |
| 5 | 47 | 53 | 75 | 13 | 3.35 |
| 6 | 554 | 624 | 764 | 15 | 3.98 |
| 7 | 452 | 662 | 756 | 16 | 4.26 |
| 8 | 557 | 663 | 711 | 18 | 4.77 |
| 9 | 4474 | 5724 | 7151 | 20 | 5.22 |
| 10 | 4726 | 5562 | 6372 | 22 | 5.64 |
| 11 | 4767 | 5723 | 6265 | 24 | 6.02 |
| 12 | 42554 | 43364 | 77304 | 24 | 6.02 |
| 13 | 43512 | 73542 | 76266 | 26 | 6.36 |

表 12-2(c) 具有最大 d_f 的 $R=1/4$ 码

| m | $\mathbf{g}^{(1, 1)}$ | $\mathbf{g}^{(1, 2)}$ | $\mathbf{g}^{(1, 3)}$ | $\mathbf{g}^{(1, 4)}$ | d_f | $\gamma(\text{dB})$ |
|-----|-----------------------|-----------------------|-----------------------|-----------------------|-------|---------------------|
| 2 | 5 | 7 | 7 | 7 | 10 | 0.97 |
| 3 | 54 | 64 | 64 | 74 | 13 | 2.11 |
| 4 | 52 | 56 | 66 | 76 | 16 | 3.01 |
| 5 | 53 | 67 | 71 | 75 | 18 | 3.52 |
| 6 | 564 | 564 | 634 | 714 | 20 | 3.98 |
| 7 | 472 | 572 | 626 | 736 | 22 | 4.39 |
| 8 | 463 | 535 | 733 | 745 | 24 | 4.77 |
| 9 | 4474 | 5724 | 7154 | 7254 | 27 | 5.28 |
| 10 | 4656 | 4726 | 5562 | 6372 | 29 | 5.59 |
| 11 | 4767 | 5723 | 6265 | 7455 | 32 | 6.02 |
| 12 | 44624 | 52374 | 66754 | 73534 | 33 | 6.15 |
| 13 | 42226 | 46372 | 73256 | 73276 | 36 | 6.53 |

表 12-2(d) 具有最大 d_f 的 $R=2/3$ 码

| m | K^* | 子生成元 | | | d_f | $\gamma(\text{dB})$ |
|-----|-------|--------------------------------------|--------------------------------------|--------------------------------------|-------|---------------------|
| 1 | 2 | $g^{(1, 1)} = 6$ $g^{(2, 1)} = 2$ | $g^{(1, 2)} = 2$ $g^{(2, 2)} = 4$ | $g^{(1, 3)} = 6$ $g^{(2, 3)} = 4$ | 3 | 0 |
| 2 | 3 | 4 1 | 2 4 | 6 7 | 4 | 1.25 |
| 2 | 4 | 7 2 | 1 5 | 4 7 | 5 | 2.21 |
| 3 | 5 | 60 14 | 30 40 | 70 74 | 6 | 3.01 |
| 3 | 6 | 64 30 | 30 64 | 64 74 | 7 | 3.68 |
| 4 | 7 | 60 16 | 34 46 | 54 74 | 8 | 4.26 |
| 4 | 8 | 64 26 | 12 66 | 52 44 | 8 | 4.26 |
| 5 | 9 | 52 05 | 06 70 | 74 53 | 9 | 4.77 |
| 5 | 10 | 63 32 | 15 65 | 46 61 | 10 | 5.22 |

* K 为编码器的移位寄存器总级数, 下同。

表 12-2(e) 具有最大 d_f 的 $R=3/4$ 码

| m | K | 子生成元 | | | | d_f | $\gamma(\text{dB})$ |
|-----|-----|--|--|--|--|-------|---------------------|
| 1 | 3 | $g^{(1, 1)} = 4$ $g^{(2, 1)} = 0$ $g^{(3, 1)} = 0$ | $g^{(1, 2)} = 4$ $g^{(2, 2)} = 6$ $g^{(3, 2)} = 2$ | $g^{(1, 3)} = 4$ $g^{(2, 3)} = 2$ $g^{(3, 3)} = 5$ | $g^{(1, 4)} = 4$ $g^{(2, 4)} = 4$ $g^{(3, 4)} = 5$ | 4 | 1.76 |
| 2 | 5 | 6 1 0 | 2 6 2 | 2 0 5 | 6 7 5 | 5 | 2.73 |
| 2 | 6 | 6 3 2 | 1 4 3 | 0 1 7 | 7 6 4 | 6 | 3.52 |
| 3 | 8 | 70 14 04 | 3 50 10 | 20 00 74 | 40 54 40 | 7 | 4.19 |
| 3 | 9 | 40 04 34 | 14 64 00 | 34 20 60 | 60 70 64 | 8 | 4.77 |

表中也列出了对 BSC, 也就是硬判决情况下每个码的渐近编码增益 γ 。如果用模拟信号译码或软判决译码, 则还可以再得到 2~3 dB 的软判决增益。例如(2, 1, 5)码, $\gamma=3.01$ dB, 若用模拟信号 VB 算法译码, 则编码增益总共有 6.01 dB, 若用八电平的软判决 VB 译

码，则可得总的编码增益为5.01 dB。

表 12-3 给出了有较长编码贮存，具有最大自由距离，且无恶性误差传播的 $R=1/2$ 互补卷积码。 $R=1/2$ 互补码的两个子生成元 $\mathbf{g}^{(1,1)}(D)$, $\mathbf{g}^{(1,2)}(D)$ 有如下关系：

$$\mathbf{g}^{(1,1)}(D) + \mathbf{g}^{(1,2)}(D) = D + D^2 + D^3 + \cdots + D^{m-1} \quad (12.31)$$

表 12-3 有最大自由距离 $R=1/2$ 的互补码

| m | $\mathbf{g}^{(1,1)}$ | d_f |
|-----|----------------------|-------|
| 2 | 5 | 5 |
| 3 | 54 | 6 |
| 4 | 62 | 7 |
| 5 | 61 | 8 |
| 6 | 504 | 9 |
| 7 | 422 | 10 |
| 8 | 503 | 11 |
| 9 | 4324 | 12 |
| 10 | 5032 | 13 |
| 11 | 5121 | 14 |
| 12 | 50214 | 15 |
| 13 | 51042 | 16 |
| 14 | 51303 | 17 |
| 15 | 503214 | 18 |
| 16 | 715022 | 18 |
| 17 | 425551 | 20 |
| 18 | 6044204 | 20 |
| 19 | 5671412 | 20 |
| 20 | 5011303 | 22 |
| 21 | 44236204 | 22 |
| 22 | 45236046 | 24 |
| 23 | 51202215 | 24 |

且 $\mathbf{g}_0^{(1,1)} = \mathbf{g}_0^{(1,2)} = \mathbf{g}_m^{(1,1)} = \mathbf{g}_m^{(1,2)} = 1$ 。如表中 $m=3$ 的码，由表可知 $\mathbf{g}^{(1,1)}$ 栏下的数字是 54。

$$\mathbf{g}^{(1,1)}(D) = 1 + D^2 + D^3$$

因而

$$\mathbf{g}^{(1,2)}(D) = 1 + D + D^3$$

所以

$$\mathbf{G}(D) = [1 + D^2 + D^3, 1 + D + D^3]$$

当 $m \leq 13$ 时，表 12-3 中所列出的互补码是非常接近最佳的，但是对更大 m 的码，如何构造好的互补码并没有解决。

必须注意，表 12-2、表 12-3 中所列出的码都是非系统码。这是由于在同样的 m 和 R 下，非系统码可以做到比系统码有更大的自由距离。当然系统码也具有一些非系统码所没有的优点：(1) 系统码编码中模 2 加法器的输入线比非系统码少；(2) 由码组中提取信息元

时，系统码能直接提取，而非系统码需要逆变换线路。由 § 10.3 可知，具有(2)优点的码称为快检码。码具有“快检”特性，在某些情况下是很重要的，例如译码器临时出故障而不能译码，或者信道干扰非常小，以致不需要译码时，就可直接从接收序列中很方便地取出信息元。

因此构造一类既具有很大自由距离，又具有快检特性的非系统码，在 VB 译码算法中是很需要的。1971年梅西和考斯特罗构造了一类具有类似于系统码快检能力、 $R=1/2$ 的非系统码，称为**内快检码**。这类码的两个子生成元 $\mathbf{g}^{(1, 1)}(D)$ 、 $\mathbf{g}^{(1, 2)}(D)$ 之间有如下关系：

$$\mathbf{g}^{(1, 1)}(D) + \mathbf{g}^{(1, 2)}(D) = D \quad (12.3.2)$$

且 $\mathbf{g}_0^{(1, 1)} = \mathbf{g}_0^{(1, 2)} = \mathbf{g}_m^{(1, 1)} = \mathbf{g}_m^{(1, 2)} = 1$ 。也就是说两个子生成元序列仅相差一位，用定理 10.4.1 检验可知，这类码显然是非恶性的。这类码的前馈逆是

$$\mathbf{G}^{-1}(D) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

因为

$$\mathbf{G}(D)\mathbf{G}^{-1}(D) = [\mathbf{g}^{(1, 1)}(D), D + \mathbf{g}^{(1, 1)}(D)] \begin{bmatrix} 1 \\ 1 \end{bmatrix} = D$$

这类码中每一码字的信息序列 $\mathbf{M}(D)$ ，可直接由码字序列 $\mathbf{C}(D)$ 迟延一个单位时间得到，其逆运算方程是

$$\begin{aligned} \mathbf{C}(D)\mathbf{G}^{-1}(D) &= \mathbf{C}(D) \begin{bmatrix} 1 \\ 1 \end{bmatrix} = (\mathbf{C}^{(1)}(D), \mathbf{C}^{(2)}(D)) \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\ &= \mathbf{C}^{(1)}(D) + \mathbf{C}^{(2)}(D) = \mathbf{M}(D)\mathbf{G}(D)\mathbf{G}^{-1}(D) = D\mathbf{M}(D) \end{aligned}$$

因此，由码字序列 $\mathbf{C}(D)$ 可很快地得到信息序列 $\mathbf{M}(D)$ 。

若码字序列 $\mathbf{C}(D)$ 在传输过程中，由于信道干扰而产生错误的概率是 p ，则由 $\mathbf{C}(D)$ 中恢复信息序列 $\mathbf{M}(D)$ 所引起的错误概率为 $2p$ 。因为在信息序列中的某一位错误如 m_i ，可以由 $c_{i+1}^{(1)}$ 或 $c_{i+1}^{(2)}$ 位中的错误引起。所以这类内快检码有错误概率放大因子为 2。

对一个 $R=1/2$ 的系统码来说，由于

$$\mathbf{G}(D) = [1, \mathbf{g}^{(1, 2)}(D)]$$

所以

$$\mathbf{G}^{-1}(D) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

因此

$$\mathbf{C}(D)\mathbf{G}^{-1}(D) = \mathbf{C}(D) \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \mathbf{C}^{(1)}(D) = \mathbf{M}(D)\mathbf{G}(D)\mathbf{G}^{-1}(D) = \mathbf{M}(D)$$

所以若原码字 $\mathbf{C}(D)$ 产生错误概率为 p ，则得到的信息序列错误概率至多也为 p ，因此系统码的错误概率放大因子是 1。

对任何一个非恶性的具有快检能力的 $R=1/2$ 非系统码来说，错误放大因子等于 2，是能做到的最小的可能值，正如 § 10.3 中所定义的，这是一类错误放大因子为 2 具有快检性能的**似系统码**。表 12-4 给出 $R=1/2$ 具有最大自由距离、有内快检特性的似系统码。这些码的自由距离，与有相同码参数的其余的最好非系统码的自由距离相比较要小一些，但由于它们具有快检特性，因而在实用中仍很吸引人。

表 12-4 有最大自由距, $R=1/2$ 的具有内快检特性的码

| m | $\mathbf{g}^{(1, 1)}$ | d_f | m | $\mathbf{g}^{(1, 1)}$ | d_f |
|-----|-----------------------|-------|-----|-----------------------|-------|
| 2 | 5 | 5 | 13 | 56406 | 14 |
| 3 | 54 | 6 | 14 | 42651 | 15 |
| 4 | 46 | 7 | 15 | 523034 | 16 |
| 5 | 55 | 8 | 16 | 511542 | 16 |
| 6 | 454 | 9 | 17 | 522415 | 17 |
| 7 | 542 | 9 | 18 | — | — |
| 8 | 551 | 10 | 19 | 4521722 | 18 |
| 9 | 4704 | 11 | 20 | 5404155 | 18 |
| 10 | 5522 | 12 | 21 | 45477404 | 19 |
| 11 | 5503 | 13 | 22 | 50756602 | 19 |
| 12 | 56414 | 14 | 23 | 53615441 | 20 |

正如 § 10.2 中所指出的, 如果子生成元的重量均为奇数, 也就是编码器中每个模 2 加法器的输入数目是奇数, 则称此类码为**透明码**, 透明码的特点是一个码字的补码仍是一个码字。因此, 透明码允许对码字的补码像码字本身一样正确译码。所以, 只需要在编码器前加一差分编码, 以及在 VB 译码器后加一差分译码, 利用透明卷积码就能消除 180° 的相位误差, 表 12-2 中的一些码是透明码, 特别适用于 VB 译码。图 12-8 给出了有差分编译码的 VB 译码框图。

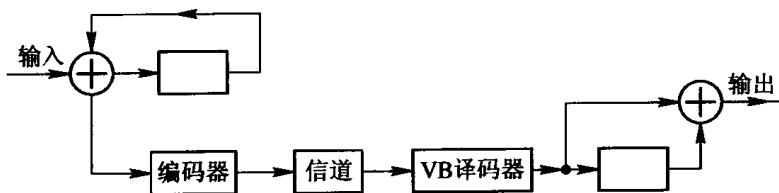


图 12-8 有差分编译码的 VB 译码框图

目前在国际卫星通信和其它通信系统中, 使用 VB 译码的标准卷积码是表 12-2(a)中的 $(2, 1, 6)$ 码, 它的两个子生成元 $(\mathbf{g}^{(1, 1)}, \mathbf{g}^{(1, 2)}) = (171, 133)$, 相应的生成矩阵 $\mathbf{G}(D) = [1 + D + D^2 + D^3 + D^6, 1 + D^2 + D^3 + D^5 + D^6]$, 这是一个透明码, 且 $d_f = 10$, $B_{d_f} = 1$, 由式(12.2.12)可知, 这是一个能使 P_{ME} 达到最小、且能克服相位误差(带差分编译码器)的最佳码。

二、卷积码的增信删余(Punctured)

在 VB 译码中, 由于译码器的复杂性随 $2^{k_0 m}$ 指数增长, 因此一般均采用 $k_0 = 1$ 的 $(2, 1, m)$ 码, 码率较低。为了提高码率, 并且又不致使译码器的复杂性增加, 因此常常把 $(2, 1, m)$ 码进行增信删余(Punctured)。增信删余的基本原理与分组码大致相同, 对 $(2, 1, m)$ 码的码字中某些特指位置的码元予以删除, 在收端译码时, 再用特定的码元在这些位置上填充, 然后输入 $(2, 1, m)$ 码的 VB 译码器译码。当然, 由于码率的提高, 码的自由距离和纠错能力比 $(2, 1, m)$ 码要低。

用以下删除方法^[9],便可以从 $1/n_0$ 码率的卷积码,产生出 $(n_0 - 1)/n_0$ 码率的增信剩余码。

(1) 把信息元以 $n_0 - 1$ 位分组,并输入 $(2, 1, m)$ 编码器,从而把 $(2, 1, m)$ 编码器输出的码字看成是 $(2(n_0 - 1), n_0 - 1, m)$ 卷积码的码字。

(2) 按以下所示的位置,在码字中的每 $2(n_0 - 1)$ 个码元中,删除特定位置的 $n_0 - 2$ 个码元,并发送剩下的 n_0 个码元,从而得到了码率为 $(n_0 - 1)/n_0$ 的码字。下表中的 x 表示这位被删除。

$$\begin{array}{l} \text{输入编码器的信息元: } (m_1, m_2, \dots, m_{n_0-1}, \dots) \\ \text{编码器输出的码字: } \begin{bmatrix} c_{11}, & c_{21}, & \cdots, & c_{n_0-11}, & \cdots \\ c_{12}, & c_{22}, & \cdots, & c_{n_0-12}, & \cdots \end{bmatrix} \\ \text{删除的码元位: } \begin{bmatrix} c_{11}, & x, & \cdots, & x, & c_{n_0-11}, & \cdots \\ c_{12}, & c_{22}, & \cdots, & c_{n_0-22}, & x, & \cdots \end{bmatrix} \end{array}$$

发送的 $(n_0, n_0 - 1, m)$ 码的一个码字: $(c_{11}, c_{12}, c_{22}, \dots, c_{n_0-22}, c_{n_0-11}, \dots)$

接收端收到码字 $(c_{11}, c_{12}, c_{22}, \dots, c_{n_0-22}, c_{n_0-11}, \dots)$ 以后,在发端的删除位置上填充特定的虚假码元,然后送入 $(2, 1, m)$ 码的VB译码器,并且译码器译码时禁止对这些虚假码元作量度计算,从而使这类增信剩余码的译码就可基于与普通的 $(2, 1, m)$ 码的VB译码方法进行。图12-9给出了从 $(2, 1, 6)$ 码产生的 $(n_0, n_0 - 1, 6)$ 增信剩余卷积编码器和VB译码器框图,图12-9和表12-5中删除比特布局图框内的“1”表示这位不删除,“0”表示这一位删除。应当注意,在收端需要有每个子码(n_0 码元)的同步功能。

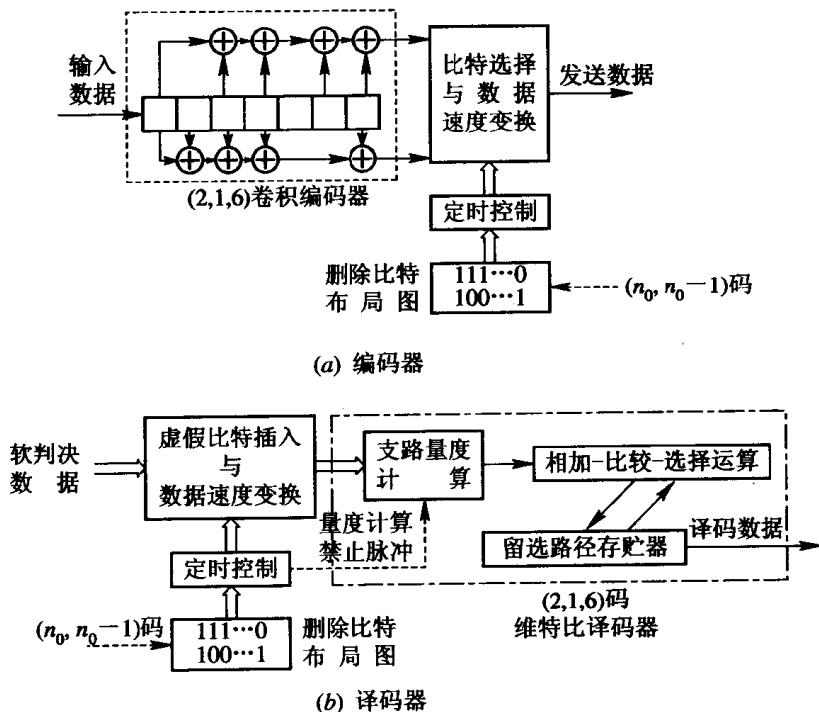


图 12-9 $(n_0, n_0 - 1, m)$ 增信剩余卷积编码器/VB 译码器框图

表 12-5 从 $(2, 1, m)$ 码产生的最佳增信删余码
的删除码元位置布局表

| $\frac{m}{\text{码率}}$ | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----------------------|--------------------------------------|--------------------------------|--------------------------------|--------------------------------|--------------------------------|--------------------------------|--------------------------------|
| $1/2$ | $1g^{(1,1)}:(5)$ $1g^{(1,2)}:(7)$ | 1 (15) 1 (17) | 1 (23) 1 (35) | 1 (53) 1 (75) | 1 (133) 1 (171) | 1 (247) 1 (371) | 1 (561) 1 (753) |
| | | | | | | | |
| $2/3$ | 10 11 | 11 10 | 11 10 | 10 11 | 11 10 | 10 11 | 11 10 |
| | | | | | | | |
| $3/4$ | 101 110 | 110 101 | 101 110 | 100 111 | 110 101 | 110 101 | 111 100 |
| | | | | | | | |
| $4/5$ | 1011 1100 | 1011 1100 | 1010 1101 | 1000 1111 | 1111 1000 | 1010 1101 | 1101 1010 |
| | | | | | | | |
| $5/6$ | 10111 11000 | 10100 11011 | 10111 11000 | 10000 11111 | 11010 10101 | 11100 10011 | 10110 11001 |
| | | | | | | | |
| $6/7$ | 101111 110000 | 100011 111100 | 101010 110101 | 110110 101001 | 111010 100101 | 101001 110110 | 110110 101001 |
| | | | | | | | |
| $7/8$ | 1011111 1100000 | 1000010 1111101 | 1010011 1101100 | 1011101 1100010 | 1111010 1000101 | 1010100 1101011 | 1101011 1010100 |
| | | | | | | | |
| $8/9$ | 10111111 11000000 | 10000011 11111100 | 10100011 11011100 | 11100010 10011101 | 11110100 10001011 | 10110110 11001001 | 11100000 10011111 |
| | | | | | | | |
| $9/10$ | 101111111 110000000 | 101000000 110111111 | 111110011 100001100 | 100001111 111110000 | 111101110 100010001 | 101100110 110011001 | 111000101 100111010 |
| | | | | | | | |
| $10/11$ | 1011111111 1100000000 | 1000000011 1111111100 | 1000000101 1111111010 | 1001110100 1110001011 | 1110110111 1001001000 | 1001000011 1110111100 | 1000101100 1111010011 |
| | | | | | | | |
| $11/12$ | 10111111111 11000000000 | 10000000010 11111111101 | 10101101101 11010010010 | 10001110100 11110001011 | 11110111110 10001000001 | 10110000110 11001111001 | 11000010001 10111101110 |
| | | | | | | | |
| $12/13$ | 101111111111 110000000000 | 100000000011 111111111100 | 101101111011 11001000100 | 110100110110 101011001001 | 111111110101 100000001010 | 100100001100 111011110011 | 110000011010 101111100101 |
| | | | | | | | |
| $13/14$ | 1011111111111 1100000000000 | 1010000000000 1101111111111 | 1101101101111 1001001001000 | 1100011000100 1011100111011 | 1101000001111 1010111100000 | 1010100100000 1101011011111 | 1000000100001 1011111011110 |
| | | | | | | | |

(a)

| 码率 | m | 2 | | 3 | | 4 | | 5 | | 6 | | 7 | | 8 | |
|-------|-----|---------|----------------------------|-------|----------------------------|-------|----------------------------|-------|----------------------------|-------|----------------------------|-------|----------------------------|-------|----------------------------|
| | | d_f | $B_{d_f}(\tilde{B}_{d_f})$ | d_f | $B_{d_f}(\tilde{B}_{d_f})$ | d_f | $B_{d_f}(\tilde{B}_{d_f})$ | d_f | $B_{d_f}(\tilde{B}_{d_f})$ | d_f | $B_{d_f}(\tilde{B}_{d_f})$ | d_f | $B_{d_f}(\tilde{B}_{d_f})$ | d_f | $B_{d_f}(\tilde{B}_{d_f})$ |
| 1/2 | 5 | 1(1) | | 6 | 2(2) | 7 | 4(4) | 8 | 2(2) | 10 | 36(36) | 10 | 2(2) | 12 | 33(33) |
| 2/3 | 3 | 1(0.5) | | 4 | 10(5) | 4 | 1(0.5) | 6 | 96(48) | 6 | 3(1.5) | 7 | 47(23.5) | 7 | 11(5.5) |
| 3/4 | 3 | 15(5) | | 4 | 124(41.3) | 3 | 1(0.3) | 4 | 3(1) | 5 | 42(14) | 6 | 239(79.7) | 6 | 52(17.3) |
| 4/5 | 2 | 1(0.3) | | 3 | 14(3.5) | 3 | 11(28) | 4 | 40(10) | 4 | 12(3) | 5 | 168(42) | 5 | 31(7.8) |
| 5/6 | 2 | 2(0.4) | | 3 | 63(12.6) | 3 | 20(4) | 4 | 100(20) | 4 | 92(18.4) | 4 | 7(1.4) | 5 | 168(33.6) |
| 6/7 | 2 | 5(0.8) | | 2 | 2(0.3) | 3 | 69(11.5) | 3 | 25(4.2) | 3 | 5(0.8) | 4 | 85(14.2) | 4 | 9(1.5) |
| 7/8 | 2 | 8(1.1) | | 2 | 4(0.6) | 3 | 49(7) | 3 | 60(8.6) | 3 | 9*(1.3) | 4 | 258(36.9) | 4 | 70(10) |
| 8/9 | 2 | 14(1.8) | | 2 | 6(0.8) | 3 | 293(36.6) | 3 | 72(9) | 3 | 13(1.6) | 3 | 3(0.4) | 4 | 124(15.5) |
| 9/10 | 2 | 20(2.2) | | 2 | 8(0.9) | 2 | 1(0.1) | 3 | 143(15.9) | 3 | 29(3.2) | 4 | 1189(132.1) | 4 | 300(33.3) |
| 10/11 | 2 | 30(3) | | 2 | 14(1.4) | 2 | 3(0.3) | 3 | 201(20.1) | 3 | 52(5.2) | 3 | 18(1.8) | 4 | 556(55.6) |
| 11/12 | 2 | 40(3.6) | | 2 | 20(1.8) | 2 | 6*(0.5) | 3 | 311(28.3) | 3 | 66(6) | 3 | 14(1.3) | 4 | 1899(172.6) |
| 12/13 | 2 | 55(4.6) | | 2 | 26(2.2) | 2 | 7(0.6) | 3 | 514(42.8) | 3 | 83(6.9) | 3 | 37(3.1) | 4 | 2038(169.8) |
| 13/14 | 2 | 70(5.4) | | 2 | 32(2.5) | 2 | 10(0.8) | 3 | 692(53.2) | 3 | 215(16.5) | 3 | 52(4) | 4 | 3424(263.4) |

(b)

增信删除码的自由距离 d_f 和 A_{d_f} 的值与删除码元位置有极大关系。为了使删除码的 d_f 最大, A_{d_f} 最小, 已利用计算机检验了删除码元位置的全部可能组合, 对于码率从 2/3 至 7/8 的各种删除码, 已找出了不但有最大 d_f 而且有最小 A_{d_f} 的删除码元位置布局图样。

表 12-5(a)、(b) 中列出了 $(2, 1, m)$ 卷积码的删除码元布局图和最小自由距离 d_f 和 B_{d_f} 、 \tilde{B}_{d_f} 的值, 这里 $\tilde{B}_{d_f}=1/k_0$ 。从此表显而易见, 删除码的 d_f 随码率的增加而减小, 对于 $(8, 7, 6)$ 码的 d_f 仍有 3。这意味着即使对于这些高码率码, VB 译码也能改善信道误码率。利用这种删除方法, 可以使 1/2 码率的 VB 译码器能工作在 2/3、3/4、4/5 等码率, 目前市面上所售的 VB 译码器芯片, 几乎都是这种多码率芯片, 以适应更广的应用范围。有关增信删除码的进一步讨论请参阅有关文献^{[8]~[10]}。

在编码过程中, 如果应用几个不同码率的删除比特布局图交替工作, 则得到的删除卷积码的码率和纠错能力交替变化, 从而得到一类有不等保护能力的卷积码。有关这类 UEP 卷积码的编译码过程和性能, 请参阅 [10], [11]。

三、VB 译码算法的计算机模拟结果

利用计算机模拟 VB 译码算法, 在各种不同情况和使用不同码时的性能, 无论在实际

设计译码器，还是在检验理论正确性方面都是很重要的方法。除了图 12-7 以外，在图 12-10 中我们给出了一组由计算机模拟 VB 译码的性能曲线。

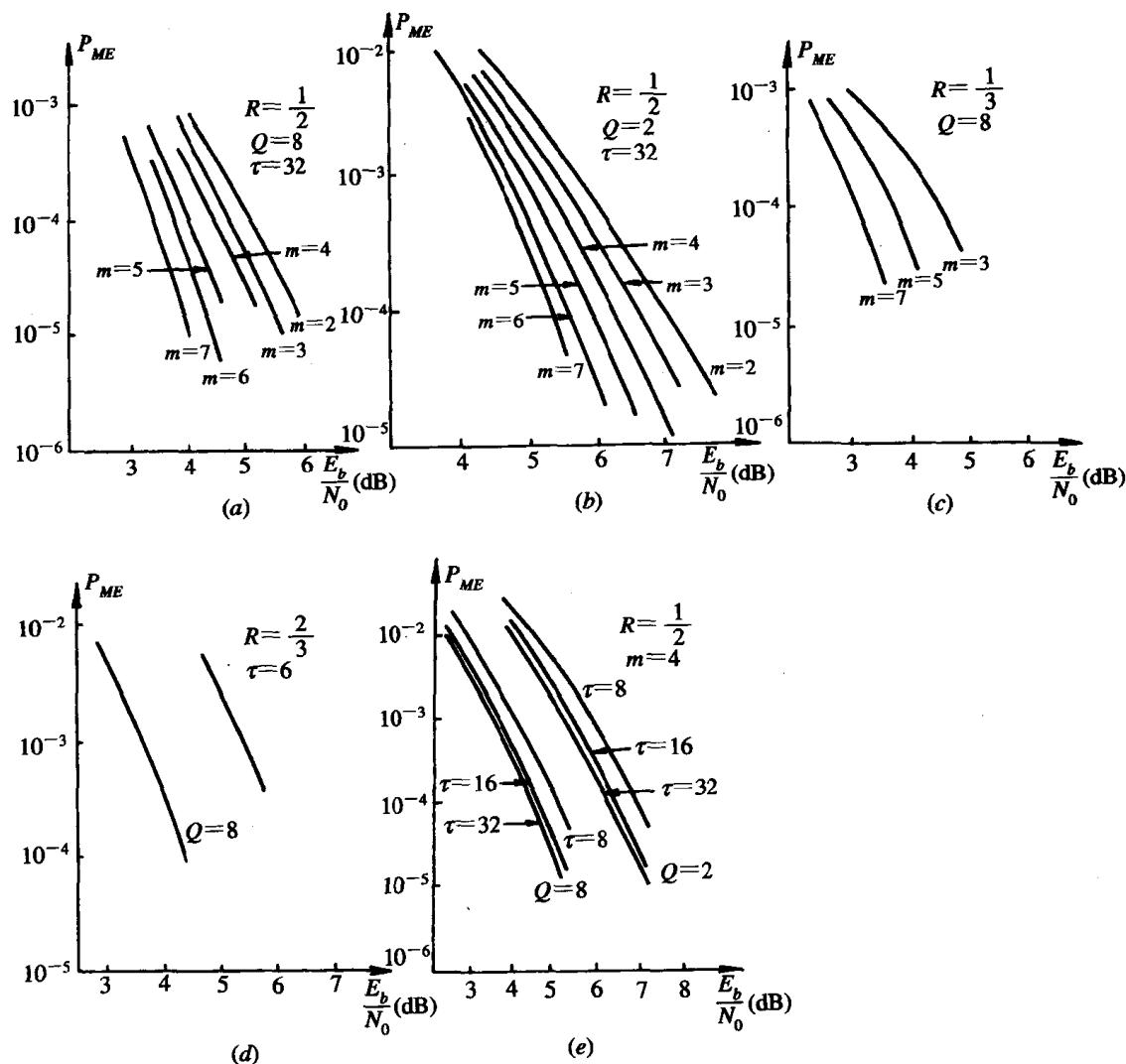


图 12-10 维特比译码算法的模拟性能曲线

图 12-10(a)给出了使用 $R=1/2$, 编码存贮 $2 \sim 7$ 时 6 个不同码、 $Q=8$ 软判决 VB 译码器的输出误码率 P_{ME} 与 E_b/N_0 的关系曲线。这些码在 $Q=2$ (硬判决)时的关系曲线示于图 12-10(b)中。在这两种情况下, 译码器的路径存贮长度 $\tau=32$ 。比较这两组曲线可知, $Q=8$ 比 $Q=2$ 时有 2 dB 的软判决增益, 在不同 Q 时的软判决增益已示于图 12-7 中。由图 12-7 的曲线可以看出, 在误码率为 10^{-4} 时, 应用 $Q=2$ 的硬判决 VB 译码, 比没有用编码能得到 3 dB 的纯编码增益, 而 $Q=8$ 时能得到总共 5 dB 的纯编码增益。

图 12-10(e)中给出了使用同样的码, 但用不同的路径寄存器长度 $\tau=8, 16, 32$ 时, $Q=2$ 和 8 情况下的性能曲线。由这些曲线看到, 路径寄存器长度 $\tau=8=2m_k$ 时的性能比 $\tau=32=8m_k$ 时要差 0.7 dB, 但是 $\tau=16=4m_k$ 的性能与 $\tau=32$ 时的性能几乎差不多。

图 12-10(c)中给出了 3 个 $R=1/3$, $m=3, 5, 7$ 码在 $Q=8$ 时的性能曲线。与图 12-10(a)的曲线相比较可知, 这些码比相应的 $R=1/2$ 的码, 在性能上要好 0.5 dB, 这是由于在

同样的编码存储 m 下, $R=1/3$ 的码比 $R=1/2$ 的码有更大的自由距离。图 12-10(d)给出了 $R=2/3$ 码, $\tau=mk_0=6$, $Q=2$ 和 8 时的性能曲线, 与相应的 $R=1/2$, $m=6$ 的码的性能曲线图 12-10(a)(b)相比, $R=2/3$ 的码要差 0.5 dB。

从上面这些曲线中得出的结论, 与以前理论分析的结果是一致的。由以上讨论我们可以把 VB 译码算法归结如下: VB 译码算法是一种最大似然译码算法, 由于它的译码器不能选用 m 太大的码, 码的 d_f 不能很大, 所以 VB 译码器的输出误码率不能做得很低, 一般只能达到 $10^{-5} \sim 10^{-6}$ 量级。又由于软判决 VB 译码器比硬判决译码器在复杂性上相差不多, 且易于实现, 因而一般均采用 $Q=8$ 或 16 电平量化的软判决译码器, 此时若用 $R=1/2$ 的码, 则大约可得到 5 dB 的纯编码增益。因而 VB 译码算法适用于误码率要求不高且功率受限的信道, 如卫星和微波中继信道中。

§ 12.4 序列译码——Fano 译码算法

VB 译码器的复杂性随 mk_0 指数增长, 故不能适用于 m 太大的码, 从而限制了 VB 译码器输出误码率不能做得太低。另一方面不论信道干扰情况如何, 即使译码器接收到的序列完全正确, 译码器译每一分支的计算量始终是不变的, 所以当信道干扰很小时, 译码器的平均计算量仍比较大。如果一个系统要求有很低的误码率, 且译码器的计算量能随信道干扰情况而变化, 从而使平均计算量很小, 那么就不能用 VB 译码算法, 而必须应用序列译码算法。

序列译码是由沃曾克拉夫特最先提出的一种实用的概率译码算法。1963年费诺对这种序列译码算法进行了修正, 称为 Fano 算法(以下简称 FA 算法)。1966年扎岗奇诺夫(загончинов)和1969年杰利内克(Jelinek)各自提出了另一种序列译码算法——叠式存贮(堆栈)译码算法(称 Stack 算法或 ZJ 算法), 它比 FA 算法译码速度更快, 但需大容量的缓存器。下面首先讨论 FA 译码算法, 下一节再介绍叠式存贮译码算法(以下简称 ST 算法)。

一、码的树图表示和费诺(FA)度量

在 VB 译码算法中应用篱笆图, 能方便地说明其译码算法过程, 而在序列译码中则用 § 10.5 介绍的码树图能更好地描述。

设输入一个 (n_0, k_0, m) 编码器信息序列的长度为 $Lk_0 + mk_0$, 其中最后 mk_0 个码元是全为 0 序列, 则编码器输出的码序列长为 $N = n_0(L + m)$, 它相应于码树图中的一条特定的路径, 该码树图共有 $(L + m)$ 级分支和 $L + m + 1$ 阶节点组成。由 $G(D) = [1 + D + D^2, 1 + D^2]$ 生成的 $(2, 1, 2)$ 码, 若信息序列长为 $Lk_0 + mk_0 = 5 + 2$, 则相应的码树图如图 12-11 所示, 它共有 $L + m = 7$ 级分支, $L + m + 1 = 8$ 阶节点(或时间)。码树图上最左边的一个节点相应于编码器的初始状态 s_0 , 而最右边的 $2^{k_0 L} = 2^5 = 32$ 个节点称为终端节点, 它们也相应于编码器的初始状态 s_0 。

码树图中前 L 阶节点的每一个, 有 $2^{k_0} = 2$ 个分支, 上一分支代表输入的信息元 $m_i = 0$, 下一分支代表输入的信息元 $m_i = 1$, 而分支上的数值, 代表在该信息元输入下, 编码器输出的、由 $n_0 = 2$ 个码元组成的码段 c_i 。所有这些分支组成了码树中 $2^{k_0 L} = 32$ 条不同的路径, 代表 $(2, 1, 2)$ 码的 32 个不同的码序列。在 $L = 5$ 阶节点后, 离开每一节点的只有一个分支,

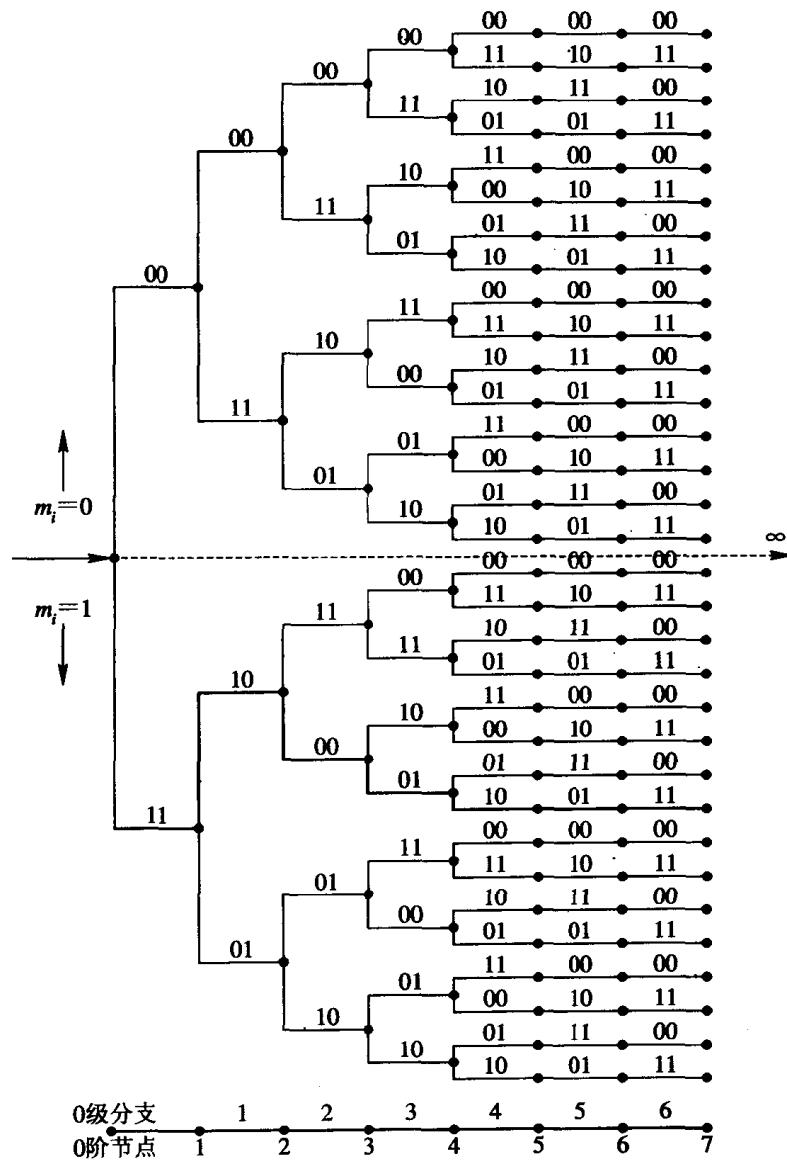


图 12-11 $L=5$ 的 $(2, 1, 2)$ 码树图

代表输入的信息元 m_{L+i} ($i = 1, 2, \dots, m$) 为 0, 它相当于从此时刻后编码器逐步回到全为 0 的初态。我们称 L 级节点后的分支为尾枝。

例如，输入(2, 1, 2)码编码器的信息序列 $m = (1011100)$ ，则输出的码序列为 $C = (11, 10, 00, 01, 10, 01, 11)$ ，它相应于码树图上用粗线所示的一条路径。

把图12-11的码树图与图12-1的篱笆图相比较可知，它们都是用不同的路径表示(2, 1, 2)码的不同码序列，所用的方法也完全相同，不同的仅只是表示形式。

在 VB 译码器中，译码器的基本任务就是根据接收的序列 R ，按最大似然译码准则在篱笆图上寻找有最大度量值的路径。同样，在序列译码中，也是根据接收的序列 R ，译码器力图寻求编码器在树图上所走的那条路径(即正确的路径)。

对一个二进制输入，Q 进制输出的 DMC，在 VB 译码算法中是用式(12.1.4)所表示的

方法求路径的度量，在所有被比较的路径有相同长度下，用式(12.1.4)所计算的度量进行比较是最佳的。但是，与 VB 译码算法不同，在序列译码中，译码时每一时间所比较的路径长度不一定相同。对这些路径，若仍用这种最大似然函数作为度量，就可能引起不正确的结果。

例如(2, 1, 2)码的一个码序列被送往 BSC，接收端收到的序列

$$\mathbf{R} = (10, 10, 00, 01, 11, 01, 00) = (\mathbf{R}_0, \mathbf{R}_1, \mathbf{R}_2, \mathbf{R}_3, \mathbf{R}_4, \mathbf{R}_5, \mathbf{R}_6)$$

对于 BSC，用 \mathbf{R} 与 \mathbf{C} 之间的汉明距离 $d(\mathbf{R}, \mathbf{C})$ 作为 VB 算法中路径 \mathbf{C} 的度量，若 $d(\mathbf{R}, \mathbf{C}_j)$ 是所有被比较的路径度量值中最小的，则 \mathbf{C}_j 被判成是最可能发送的序列。现在考虑两个不同长度的路径，比较它们的度量值。设两个截尾码是：

$$\mathbf{C}_5 = (11, 10, 00, 01, 10, 01)$$

$$\mathbf{C}_0 = (11)$$

则部分路径度量值为：

$$d(\mathbf{R}_0 \dots \mathbf{R}_5, \mathbf{C}_5) = 2$$

$$d(\mathbf{R}_0, \mathbf{C}_0) = 1$$

若仍按最小汉明距离译码准则，则说明 \mathbf{C}_0 是这两个路径中的最好一条路径。然而，从直观感到 \mathbf{C}_5 比 \mathbf{C}_0 更可能是最大似然路径中的一部分，因为由 \mathbf{C}_0 开始还需要再比较 12 个码元的度量值，而从 \mathbf{C}_5 开始仅只需要再比较 2 个码元的度量值就能完成一个码序列的译码。也就是说由 \mathbf{C}_0 开始的路径，与接收序列 \mathbf{R} 的总距离很可能超过由 \mathbf{C}_5 开始的路径与 \mathbf{R} 的总距离。

因此，在序列译码中必须对路径的度量作适当的调整，以适应不同长度路径比较的情况。对一个二进制输入，Q 进制输出的 DMC，设由 (n_0, k_0, m) 编码器送入该信道的序列是

$$\mathbf{C} = (c_0, c_1, \dots, c_{L-1})$$

由 DMC 输出的序列

$$\mathbf{R} = (\mathbf{R}_0, \mathbf{R}_1, \dots, \mathbf{R}_{L-1})$$

在最大似然译码算法中，译码器求

$$\max [P(\mathbf{R}|\mathbf{C})]$$

在 DMC 条件下

$$P(\mathbf{R}|\mathbf{C}) = \prod_{i=0}^{L-1} P(\mathbf{R}_i|c_i) = \prod_{i=0}^{n_0 L - 1} P(r_i|c_i)$$

式中， r_i 、 c_i 分别是接收序列与发送码序列中的第 i 个码元。由贝叶斯公式

$$P(\mathbf{C}|\mathbf{R}) = \frac{P(\mathbf{C})P(\mathbf{R}|\mathbf{C})}{P(\mathbf{R})}$$

式中

$$P(\mathbf{R}) = \prod_{i=0}^{L-1} P(\mathbf{R}_i) \prod_{i=0}^{n_0 L - 1} P(r_i)$$

$P(\mathbf{C})$ 是发送码序列 \mathbf{C} 的先验概率，若为等概发送，则

$$P(\mathbf{C}) = 2^{-k_0 L} = 2^{-n_0 R_c L}$$

式中， $R_c = k_0/n_0$ 是码率。把上面这些式子代入贝叶斯公式得

$$P(\mathbf{C}|\mathbf{R}) = \frac{2^{-n_0 R_c L} \prod_{i=0}^{n_0 L-1} P(r_i|c_i)}{\prod_{i=0}^{n_0 L-1} P(r_i)} = 2^{-n_0 R_c L} \prod_{i=0}^{n_0 L-1} \frac{P(r_i|c_i)}{P(r_i)}$$

两边取对数

$$\log_2 P(\mathbf{C}|\mathbf{R}) = -n_0 L R_c + \sum_{i=0}^{n_0 L-1} \log_2 \frac{P(r_i|c_i)}{P(r_i)} = \sum_{i=0}^{n_0 L-1} (\log_2 \frac{P(r_i|c_i)}{P(r_i)} - R_c) \quad (12.4.1)$$

式中, $P(r_i|c_i)$ 是信道的转移概率, $P(r_i)$ 是信道输出符号的概率。在发送码序列等概情况下, 最大似然函数译码就是最大后验概率译码, 也就是求

$$\max (\log_2 P(\mathbf{C}|\mathbf{R}))$$

由式(12.4.1)知, 就是求最大的

$$\sum_{i=0}^{n_0 L-1} \left(\log_2 \frac{P(r_i|c_i)}{P(r_i)} - R_c \right)$$

因此若我们定义

$$M_F(r_i|c_i) = \log_2 \frac{P(r_i|c_i)}{P(r_i)} - R_c \quad (12.4.2)$$

作为不同长度路径比较时的比特度量, 则此度量一定是最好的, 称此度量为**比特 Fano (FA) 度量**。

路径 \mathbf{C} 的首 l 个分支组成的部分路径 FA 度量是

$$M_F(\mathbf{R}|\mathbf{C})_l = \sum_{i=0}^{l-1} M_F(\mathbf{R}_i|\mathbf{c}_i) = \sum_{i=0}^{n_0 l-1} M_F(r_i|c_i) \quad (12.4.3)$$

这里, $M_F(\mathbf{R}_i|\mathbf{c}_i)$ 是第 i 个分支的分支 FA 度量, 它等于

$$\begin{aligned} M_F(\mathbf{R}_i|\mathbf{c}_i) &= \sum_{j=1}^{n_0} M_F(r_{ij}|c_{ij}) \\ &= \sum_{j=1}^{n_0} \left[\log_2 \frac{P(r_{ij}|c_{ij})}{P(r_{ij})} - R_c \right] \\ &= \log_2 \frac{P(\mathbf{R}_i|\mathbf{c}_i)}{P(\mathbf{R}_i)} - n_0 R_c \end{aligned}$$

式中 r_{ij} 和 c_{ij} 分别是第 i 个分支的第 j 个码元的接收码元和发送码元。比较式(12.4.2)与式(12.4.3)可知:

$$M_F(\mathbf{R}|\mathbf{C})_l = \sum_{i=0}^{n_0 l-1} \log_2 P(r_i|c_i) - \sum_{i=0}^{n_0 l-1} \log_2 P(r_i) - n_0 l R_c \quad (12.4.4)$$

若二进制输入、Q 进制输出的 DMC, 信道中的转移概率是

$$P(j|0) = P(Q-1-j|1) \quad j = 0, 1, \dots, Q-1$$

则我们说该 DMC 是对称信道。若输入该信道的符号 0、1 是等概情况下, 则信道输出符号也是等概, 即

$$P(r_i) = \frac{1}{Q} \quad i = 0, 1, 2, \dots, Q-1$$

把它代入式(12.4.4)得

$$M(\mathbf{R}|\mathbf{C})_l = \sum_{i=0}^{n_0 l-1} \log_2 P(r_i|c_i) - n_0 l \log_2 \frac{1}{Q} - n_0 l R_c$$

$$= \sum_{i=0}^{n_0 l - 1} \log_2 P(r_i | c_i) + n_0 l (\log_2 Q - R_c) \quad (12.4.5)$$

该式的第一项就是 VB 译码算法中的度量表示式(12.1.4)，而第二项随路径长度 $n_0 l$ 线性增加，且是一个正的值，称此值为基。由此式可知，比较长的路径有一个比较大的基，反映出它更接近于码树的终端节点，因此更可能是最大似然路径的部分路径。

例如，上例中的以下两个截尾码：

$$\mathbf{C}_5 = (11, 10, 00, 01, 10, 01)$$

$$\mathbf{C}_6 = (11)$$

它们的部分路径度量值按式(12.4.5)计算是：

$$M(\mathbf{R}|\mathbf{C})_5 = 10 \log_2(1-p) + 2 \log_2 p + 12(\log_2 2 - \frac{1}{2})$$

$$= 10 \log_2(1-p) + 2 \log_2 p + 6$$

$$M(\mathbf{R}|\mathbf{C})_6 = \log_2(1-p) + \log_2 p + 2(\log_2 2 - \frac{1}{2})$$

$$= \log_2(1-p) + \log_2 p + 1$$

若 $p=0.1$ ，则

$$M(\mathbf{R}|\mathbf{C})_5 = -2.26 > -2.49 = M(\mathbf{R}|\mathbf{C})_6$$

说明 \mathbf{C}_5 比 \mathbf{C}_6 是更好的路径。这说明 FA 度量反映出了不同长度时度量的差别。

一般情况下，对于一个有转移概率为 p 的 BSC，则 FA 度量

$$M_F(r_i | c_i) = \begin{cases} \log_2 2p - R_c & r_i \neq c_i \\ \log_2 2(1-p) - R_c & r_i = c_i \end{cases}$$

若 $R_c=1/2$, $p=0.1$ ，则

$$M_F(r_i | c_i) = \begin{cases} -2.82 & r_i \neq c_i \\ 0.35 & r_i = c_i \end{cases}$$

我们列出 FA 度量表如表 12-6(a) 所示。在实际上为了易于工程实现，我们希望用整数表示度量，所以可把表 12-6(a) 的度量，用尺寸因子 $1/0.35$ 乘，并选取最接近的整数，则得到如表 12-6(b) 所示的整数度量表。

表 12-6 有 $p=0.1$ 的 BSC, $R=1/2$ 码时 FA 度量表

| $c_i \backslash r_i$ | 0 | 1 |
|----------------------|-------|-------|
| 0 | 0.35 | -2.82 |
| 1 | -2.82 | 0.35 |

(a)

| $c_i \backslash r_i$ | 0 | 1 |
|----------------------|----|----|
| 0 | 1 | -8 |
| 1 | -8 | 1 |

(b)

二、FA 算法

在讨论 FA 算法以前，先看以下例子。设一个 $(3, 1, 2)$ 卷积码，它的码树图如图 12-12 所示。发送的码序列是一个全为 0 序列，通过 BSC 传至译码器。译码器接收的序列 $\mathbf{R}_1 = (001, 010, 000, 100, 000, 000, \dots)$ ， \mathbf{R}_1 中的每一子组不多于一个错误。如果译码器收到第 0 子组 001 后，与从初始节点分出的 $2^{k_0}=2$ 条分支按最大似然函数值(在这里按

最小汉明距离)进行比较,由于 $d(001, 000) < d(001, 111)$,因而译码器就从初始节点c通过000分支推进到第一阶节点a,并译出与此分支相应的第0子组信息元0。在a节点,译码器用同样方法,把收到的第一子组010与从a点分出的两条分支进行比较,推进到节点b,译出相应的第一子组信息元。我们称这种译码方法为译码约束度为1的逐分支译码法。根据此方法,由接收到的 \mathbf{R}_1 ,译码器沿正确路径由一个节点逐步推向下一个节点,最后译出正确的码序列。但是,如果信道的干扰在某一瞬间变得很大,则这种逐分支译码法就不能正确译码。例如,若发送的仍是全为0序列,译码器收到的序列 $\mathbf{R}_2 = (000, 000, 011, 000, 000, \dots)$,根据逐分支译码法,译码器沿正确路径一直走至第二阶节点b后,就一直走在不正确路径中,我们称译码器开始从不正确路径行走的节点为分离点,如b点。

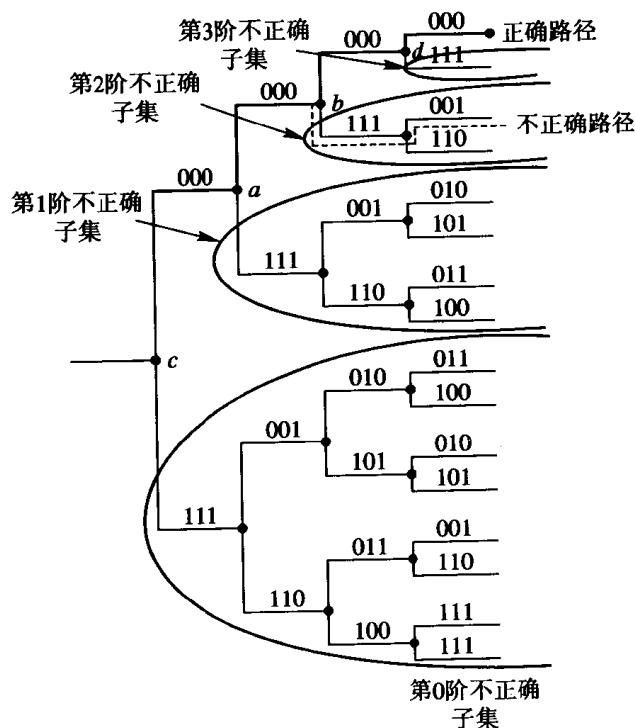


图 12-12 (3, 1, 2) 码树图

由上面的例中看到,虽然 \mathbf{R}_2 中的总错误比 \mathbf{R}_1 还少,但这种译码约束度为1的逐分支译码方法,由于没有充分利用卷积编码器所提供的纠错能力,不能保证正确译码。为了保证译码错误概率尽可能小,必须对这种逐分支译码法进行改进,并补充一些新的译码规则。我们称这些译码规则为译码算法,一个好的译码算法,必须满足以下几点要求:

- (1) 译码器能以很大的概率发现它走在不正确路径之中;
- (2) 当译码器发现它走在不正确路径时,能以很大的概率回到正确路径上;
- (3) 进行译码时,译码计算量不能很大,且设备要尽可能简单。

满足上述要求的译码算法有很多,如FA和ZJ算法等,这里仅介绍FA算法。

首先必须讨论译码器根据什么准则识别它是否走在正确路径之中。设 Δd_j 表示接收序列 \mathbf{R} 的第 j 段子组与某一码序列 C 的第 j 段子码之间的汉明距离, d_l 表示 \mathbf{R} 与 C 的 l 段长码序列之间总的汉明距离,即

$$d_l = d(\mathbf{R}_l, \mathbf{C}_l) = \sum_{j=0}^{l-1} \Delta d_j$$

下面我们观察当译码器沿码树逐分支进行时, d_l 的变化规律。

如果译码器沿正确路径前进, 则 d_l 是发送的码序列 \mathbf{C} 与接收序列 \mathbf{R} 在 l 段长路径上的总距离。由大数定律可知, 在误码率为 p_e 的 BSC 中, 当 l 足够大时, \mathbf{R} 中的错误个数即为 $d(\mathbf{R}_l, \mathbf{C}_l)$, 近似为 $n_0 l p_e$, 所以 d_l 随 l 增长的斜率为 $n_0 p_e$, 译码器输出的判决路径 $\hat{\mathbf{C}}_l$ 的距离轨迹围绕斜率为 $p_e n_0$ 的直线摆动, 如图 12-13 中的曲线所示。当译码器沿错误路径前进时, d_l 表示 \mathbf{R} 与任一非发送的码序列在 l 段上的距离, 这是两个完全无关序列之间的距离, 故 d_l 将在斜率为 $n_0/2$ 的直线附近摆动(参看图 12-13)。由于 $p_e \ll 1/2$, 所以就可利用 d_l 斜率的这种差别, 作为译码器是否走在正确路径上的判决标准, 或者说抛弃错误路径的准则。

在最简单情况下, 可以规定一条斜率为 $p n_0$ 的直线 k_l (参看图 12-13), $p_e < p < 1/2$ 。若 $d_l > k_l$, 就可认为译码器行走在错误路径中, 这时译码器就返回到最近的节点, 试走另一条未走过的分支。若 $d_l \leq k_l$, 则认为译码器走在正确路径上, 这时可继续前进到下一个节点。为了避免在起始节点上由于突发错误造成干扰的影响, 我们选取的 k_l 在 $l=0$ 处不为 0。

我们也可用

$$\lambda_l = p n_0 l - d(\mathbf{C}_l, \mathbf{R}_l) \quad p_e < p < \frac{1}{2} \quad (12.4.6)$$

代替 k_l 作为判别准则。称 λ_l 为 \mathbf{C}_l 与 \mathbf{R}_l 之间的斜距离。如果译码器沿正确路径前进, 则是一个正的值, 且随着 l 的增加而增加。若译码器走在不正确路径, 则 λ_l 是一个负值, 且随 l 的增加而减少。对于正确路径和不正确路径的一个典型 λ_l 状态图如图 12-14 所示。

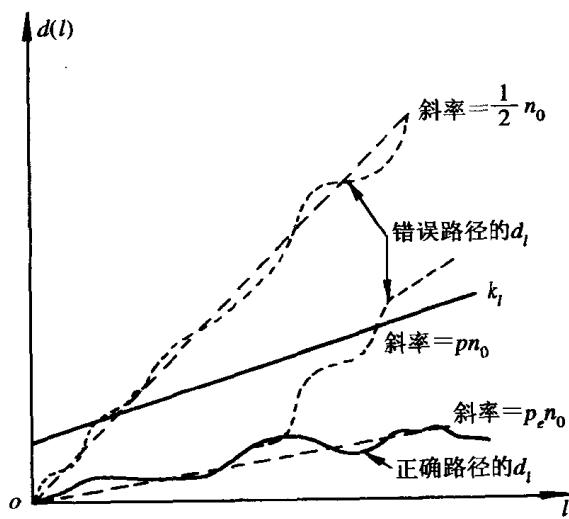


图 12-13 正确的路径与错误路径的 d_l 变化曲线

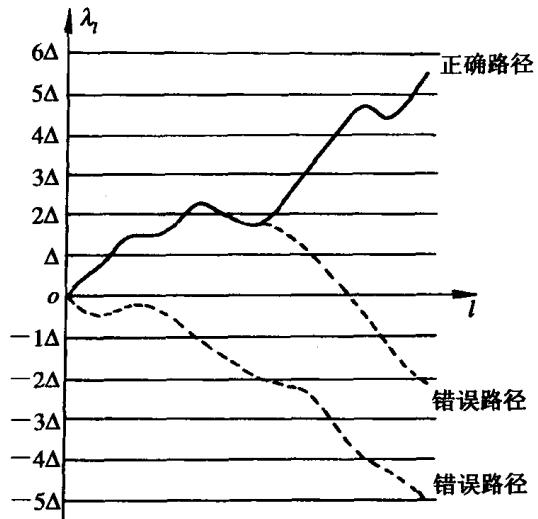


图 12-14 斜距离 λ_l 典型状态图

所以在译码过程中, 当译码器沿码树由一节点向下一节点前进时, 就计算一次 λ_l 。当 λ_l 随 l 增加而增加, 则就可判断译码器在正确路径中; 否则, 就认为在错误路径中, 这时译码器就抛弃这条错误路径, 返回寻找分离点和正确路径。

按照这种方法译码的优点是一旦抛弃了一条错误路径, 就抛弃了以此路径端点出发的

所有后续路径。例如，在第 l 阶节点抛弃了一条错误路径，则就抛弃了由此节点出发的所有后面的 2^{l-1} 路径。因此，这种译码方法的计算量较小，特别在小干扰情况下，这种序列译码方法仅用少量的计算，就可获得正确的译码。也就是说，这种译码方法能尽早地排除错误路径，减少搜索量，从而使平均计算次数减少。

另一个问题是随着译码器在码树上前进， λ 不断增加，如果在某一分离点译码器开始走向错误路径，这时 λ 的值开始减少，但仍可能是正的。所以，若仍以 λ 是否为正或负来判断，就不能及早发现译码器走在错误路径之中。因此，必须对码树的每一节点给定一个规定的 λ 值，称它为门限 T_l 。仅当 $\lambda \geq T_l$ 时，译码器才认为处于正确路径之中，从而由此节点向下一节点前进；否则，就返回搜索分离点。为了便于工程上的实现，通常规定相邻两节点之间的门限差为 $k\Delta$ ，即 $T_{l+1} = T_l + k\Delta$ ，其中 k 是整数， Δ 是设计参数。而初始节点的门限通常规定为零。由于在译码过程中，门限在不断改变之中，所以通常称门限 T_l 为活动门限。

在 FA 算法中，利用部分路径 FA 度量 $M_F(\mathbf{R}|\mathbf{C})_l$ 代替 λ 作为判别准则

$$M_F(\mathbf{R}|\mathbf{C})_l = \sum_{i=0}^{l-1} M_F(\mathbf{R}_i|\mathbf{C})_i = \sum_{i=0}^{n_0 l - 1} \log_2 \frac{P(r_i|c_i)}{P(r_i)} - n_0 l R_c$$

该式等式右边第一项和式表示接收序列和发送码序列之间的互信息量，当译码器沿着正确路径前进时，它随 l 增加而增加；第二项是为确定 l 段消息序列所必需的信息量。当 l 足够大时，第一项总可以大于第二项，即 $M_F(\mathbf{R}|\mathbf{C})_l \geq 0$ 。反之，若译码器走在错误路径之中，由于错误路径与接收序列统计无关，它们之间的互信息量很小，因而第一项随 l 的增加而减小，最后必小于第二项，所以 $M_F(\mathbf{R}|\mathbf{C})_l < 0$ 。因此，可以利用 $M_F(\mathbf{R}|\mathbf{C})_l$ 的这个特点识别译码器是否走在正确路径之中。

FA 算法的一个完整流程图示于图 12-15 中。由此我们可得出 FA 译码算法的译码过程大致如下：

(1) 译码器从码树的初始节点开始译码，每次移动一个分支。该初始节点的门限 $T = 0$ ，FA 度量值 $M_F(\mathbf{R}|\mathbf{C})_0 = 0$ （今后简称度量，用 M 表示）；

(2) 设译码器已沿某一路径走到第 l 阶节点 a ，此点的门限为 T_l （参看图 12-16）。译码器由此节点朝前看，计算由 a 节点分出的 2^{k_0} 个分支的分支度量，并挑选一个有最大分支 FA 度量值的分支（称它为最佳分支），设 M_{Fb} 是由 a 节点出发的经最佳分支的部分路径 FA 度量。若 $M_{Fb} \geq T_l$ ，则译码器通过此分支走到第 $(l+1)$ 阶节点 b （称它为最佳节点）。若 b 节点是译码器第一次到达，则提高门限，成为 $T_{l+1} = T_l + k\Delta$ ，但是新门限 T_{l+1} 不能超过目前最大的部分路径度量 M_{Fb} ，称 Δk 为门限增量，若 b 节点是以前到达过的，则门限保持不变仍为 T_l 。由 b 节点译码器试图再次向前看，前进到第 $(l+2)$ 阶节点。

(3) 若 $M_{Fb} < T_l$ ，则译码器向后看。设 M_{Bc} 是返回到第 $(l-1)$ 阶节点 c 的 FA 度量。若 $M_{Bc} \geq T_l$ ，则译码器返回到节点 c （译码器在上次前进时是由此节点 c 到达节点 a 的）。若译码器是经过由 c 点出发的 2^{k_0} 个分支中最坏的分支（即有最小分支 FA 度量值）回到节点 c 的，则译码器由节点 c 再次向后看。若不是，则译码器由 c 节点经过次最佳分支朝前看第 l 阶节点 d ，若 $M_{Fd} \geq T_l$ ，则译码器前进到第 l 阶节点 d ，并且译码器再试图由节点 d 朝前看。若 $M_{Bc} < T_l$ ， $M_{Fd} < T_l$ ，译码器既不能前进也不能后退，此时门限 T_l 减低 $k\Delta$ ，故节点 a 的门限变为 $T_l^{(1)} = T_l - k\Delta$ ，用此降低的门限 $T_l^{(1)}$ ，译码器再次朝前看节点 b 。

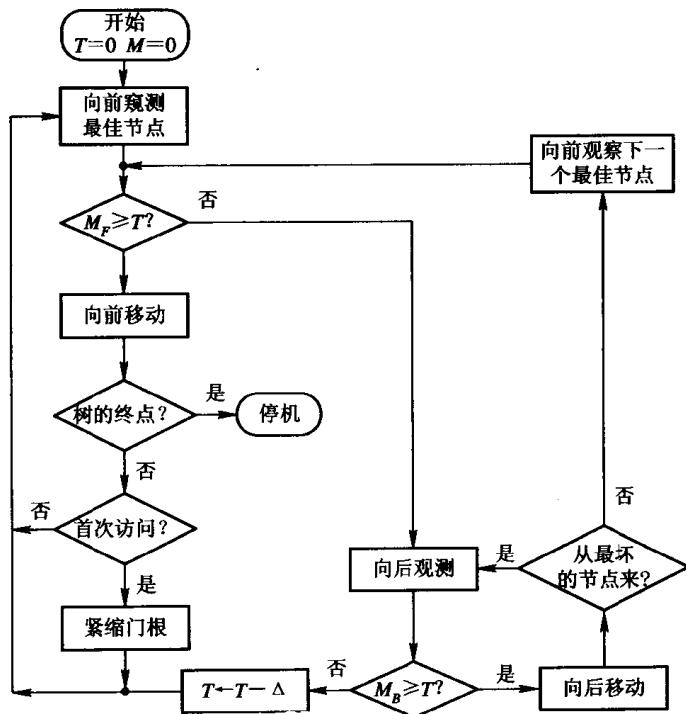


图 12-15 费诺算法流程图

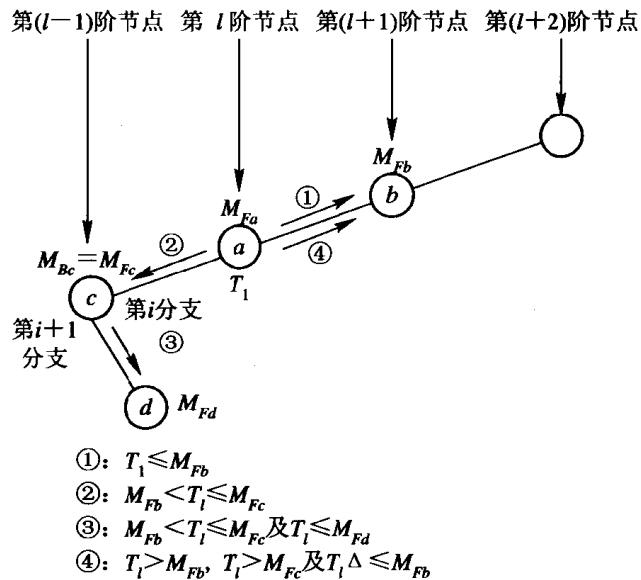


图 12-16 译码器从节点 a 的移动图

(4) 若译码器总是由初始节点向后看，则我们假定前一节点有度量值为 $-\infty$ ，在这种情况下限总是要减低 $k\Delta$ 。若两个分支有相等的分支度量，则可以任选一个分支前进，而并不会影响最后结果的正确性。

例 12.2 以(2, 1, 2)码为例。编码器输出的码序列 $C = (11, 10, 00, 01, 10, 01, 11)$, 通过 $p=0.1$ 的 BSC, 译码器收到的序列: $R = (10, 10, 00, 01, 11, 01, 11)$ 。现用 FA 算法求译码器的译码过程, 使用图 12-11 的码树和表 12-6(b) 的 FA 度量表。

利用 FA 译码算法, 译该接收序列的过程列于表 12-7 中。表中“看”栏下的符号“LFB”表示“朝前看到最佳节点”, “LNFB”表示“朝前看到次最佳节点”。表中的“ x ”代表初始节点, T 代表门限, M_F 和 M_B 分别代表向前和向后看的节点的部分路径 FA 度量。

表 12-7 用 FA 译码算法译接序列 $R=(10, 10, 00, 01, 11, 01, 11)$ 的过程, $k\Delta=7$

| 节拍 | 看 | M_F | M_B | 节点 | 总度量 | T |
|----|------|-------|-----------|---------|-----|-----|
| 0 | | | | x | 0 | 0 |
| 1 | LFB | -7 | $-\infty$ | x | | -7 |
| 2 | LFB | -7 | | 1 | -7 | -7 |
| 3 | LFB | -5 | | 10 | -5 | -7 |
| 4 | LFB | -3 | | 101 | -3 | -7 |
| 5 | LFB | -1 | | 1011 | -1 | -7 |
| 6 | LFB | -8 | -3 | 101 | -8 | -7 |
| 7 | LNFB | -19 | -5 | 10 | -5 | -7 |
| 8 | LNFB | -21 | -7 | 1 | -7 | -7 |
| 9 | LNFB | -23 | 0 | x | 0 | -7 |
| 10 | LNFB | -7 | | 0 | -7 | -7 |
| 11 | LFB | -14 | $-\infty$ | x | | -14 |
| 12 | LFB | -7 | | 1 | -7 | -14 |
| 13 | LFB | -5 | | 10 | -5 | -14 |
| 14 | LFB | -3 | | 101 | -3 | -14 |
| 15 | LFB | -1 | | 1011 | -1 | -14 |
| 16 | LFB | -8 | | 10111 | -8 | -14 |
| 17 | LFB | -6 | | 101110 | -6 | -7 |
| 18 | LFB | -22 | -8 | 10111 | -8 | -14 |
| 19 | LFB | -6 | | 101110 | -6 | -14 |
| 20 | LFB | -22 | -1 | 1011100 | -6 | 停止 |

由此表可知, 译这一帧接收序列需 20 步才能完成, 其最后得到的结果与 VB 译码算法所得的结果相同。一般情况下, Fano 译码算法也是一种近似最大似然译码算法。 ■

通过以上例子, 可以看出 FA 译码算法有如下特点:

(1) 译码器译每一帧序列的计算次数, 随序列中错误的多少, 也就是随着信道干扰的大小而变化。

(2) 计算次数与 $k\Delta$ 密切有关, 若 $k\Delta$ 小, 则计算次数增加(如该例中取 $k\Delta=1$, 则由 0 至第一步的门限减低至 -7 需计算 7 次才能达到); 反之, 则减少。但是若门限增量 $k\Delta$ 取得很大, 则译码器不易发现错误路径, 影响译码器性能。这是因为在某一节点的门限必须

小于最大似然路径的度量值，如果 $k\Delta$ 太大，则虽然 T 低于最大似然路径的度量，但同时也可能有好几条其它路径的度量也低于 T ，因而有可能其它路径比最大似然路径更早地被译码器判决为发送序列。并且若 $k\Delta$ 太大，也同样会使计算量增加，因为门限减低太多时，就需要搜索更多的路径。经验表明：若应用非整数度量，则 $k\Delta$ 选用 2 至 8 之间较好；若采用整数度量，则 $k\Delta$ 选用 2 至 8，并乘以整数尺寸因子。如上例中度量值的尺寸因子是 $1/0.35$ ，说明 $k\Delta$ 选用 5.70 至 20.8 较好。在我们所例举的情况下， Δ 选用 7 至 21 较好。

(3) 译码器需要一个输入缓冲器，以存贮输入的接收序列，以备译码器搜索分离点时存贮后面输入的序列，以及提供以前收到的接收序列。但若信道干扰很大时，译码器搜索时间很长，这时就可能引起缓存器溢出。由于序列译码的计算量与编码存贮 m 无关，故能选用 d_f 很大的码，从而使译码错误概率可以很小。因此，在 FA 算法中，译码错误概率不是主要的问题，而缓存器溢出却是一个主要问题。

* § 12.5 序列译码——ST 译码算法

无论是费诺算法还是 ST 算法，序列译码的中心思想是根据 FA 度量，译码器在码树上寻找正确路径时，力求尽早地排除所有非最大似然路径，从而使译码器的平均计算量减少。

与 FA 算法不同，在 ST 译码算法的译码器中，有一个大容量的列表式(叠式)存贮器(下面简称 SS)，用来存贮被比较的不同长度路径及其度量值。在译码过程中的每一步，有最大度量值的路径及其度量存入 SS 中的最上一行(第一行)，并以度量值递减的次序，把路径及其度量依次存入 SS 中。当进行下一步译码时，译码器计算处在 SS 第一行路径后的、 2^k 个后续分支的度量值，并与存在第一行中的度量值进行相加，得到 2^k 个新度量值，把这些度量值与 SS 中其它行的度量值进行比较，重新在 SS 中按照度量值的大小进行排列。有最大度量值的路径及其度量放在第一行以代替原来存放的内容，其余按度量值递减的次序排列。当译到某一步后，若 SS 中最上一行存贮的路径已处在码树的终点节点(第 $L+m+1$ 阶节点)，则译码完毕，译码器输出 SS 中第一行的路径作为判决路径，送给用户，由此可见这条路径是最大可能有最大似然函数的路径。我们可把上述这种 ST 译码算法过程，归结为如下步骤：

- (1) 从码树图的起始节点(0阶节点)开始译码，SS 的内容全为 0。
- (2) 计算存在 SS 第一行中路径后续各分支的度量值，并与原来存贮的度量值相加。
- (3) 删去 SS 第一行中原来存贮的内容。
- (4) 挑选一条有最大度量值的路径(若有一条以上，则任选一条)存入 SS 的第一行中，并按度量值递减的次序，重新安排 SS 中存贮的次序和内容。
- (5) 若 SS 第一行内的路径已处在码树的终端节点，则译码器停止；否则回到步骤(2)。
- (6) 当译码算法完毕时，译码器把 SS 中第一行存贮器的内容送给用户。

一个完整的上述算法的流程图，示于图 12-17 中。

例 12.3 仍以 $(2, 1, 2)$ 码为例，设编码器输入的信息序列 $M = (1011100)$ ，编码器输出的码序列 $C = (11, 10, 00, 01, 10, 01, 11)$ ，通过 $p=0.1$ 的 BSC 信道，译码器收到序列与例 12.1 同为 $R = (10, 10, 00, 01, 11, 01, 11)$ 。现用 ST 译码算法求译码器译码过程，仍

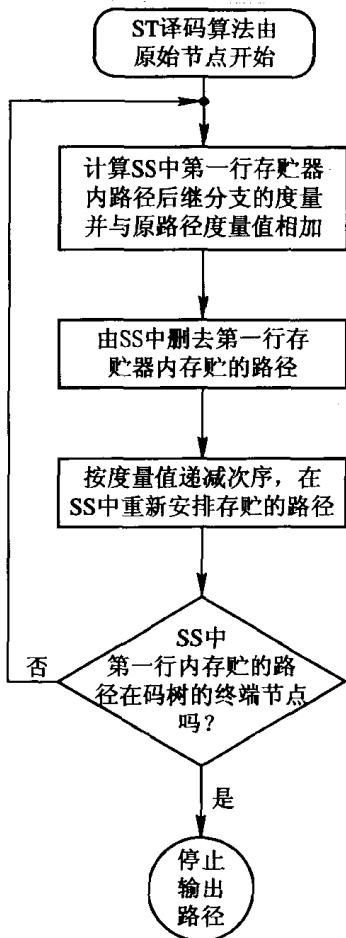


图 12-17 ST 译码算法流程图

使用图 12-11所示的码树和表 12-6(b)的 FA 度量表。

译码步骤和译码器在码树图上的行走过程，以及 ST 存贮器中的内容如图 12-18 所示。图中，树图分支旁边圆括号内的数字为该路径的度量值 M_i ，树图右边的表为 SS 存贮中按度量值递减次序存贮的路径(用信息估值序列 \hat{M} 表示)及其度量值。由图可见第九步以后译码器得到输出的估值序列 $\hat{C} = (11, 10, 00, 01, 10, 01, 11)$ ，相应的信息估值序列 $\hat{M} = (1011100)$ ，此结果与例 12.1 中用 VB 译码器得到的结果相同，也与用 FA 算法的结果相同。这说明用 ST 译码算法得到的输出序列，通常是码树上有最大似然函数的路径。

通过以上例子可看出 ST 译码算法有如下特点：

(1) 与 FA 算法同，译码器译每一帧(L 段)接收序列的计算次数，随序列中错误个数(也即信道干扰的大小)的多少而变化。若信道干扰情况严重错误个数增加，则计算次数增加；反之则减少。与 FA 算法一样，译码器的平均计算次数与编码存贮 m 无关，因而在 ST 译码中可选用大 m 也就是有大自由距离的码，使译码错误概率可以做得很小。

(2) 在树图的尾枝以前译码中，每译一步，就要有 $2^{k_0} - 1$ (对 (n_0, k_0, m) 卷积码来说)组数据(\hat{M} 与 M_i)往列表式存贮器中寄存，因而 SS 的容量要足够大。并且每译一次，SS 中的存贮次序按照 M_i 递减的次序重新排列。

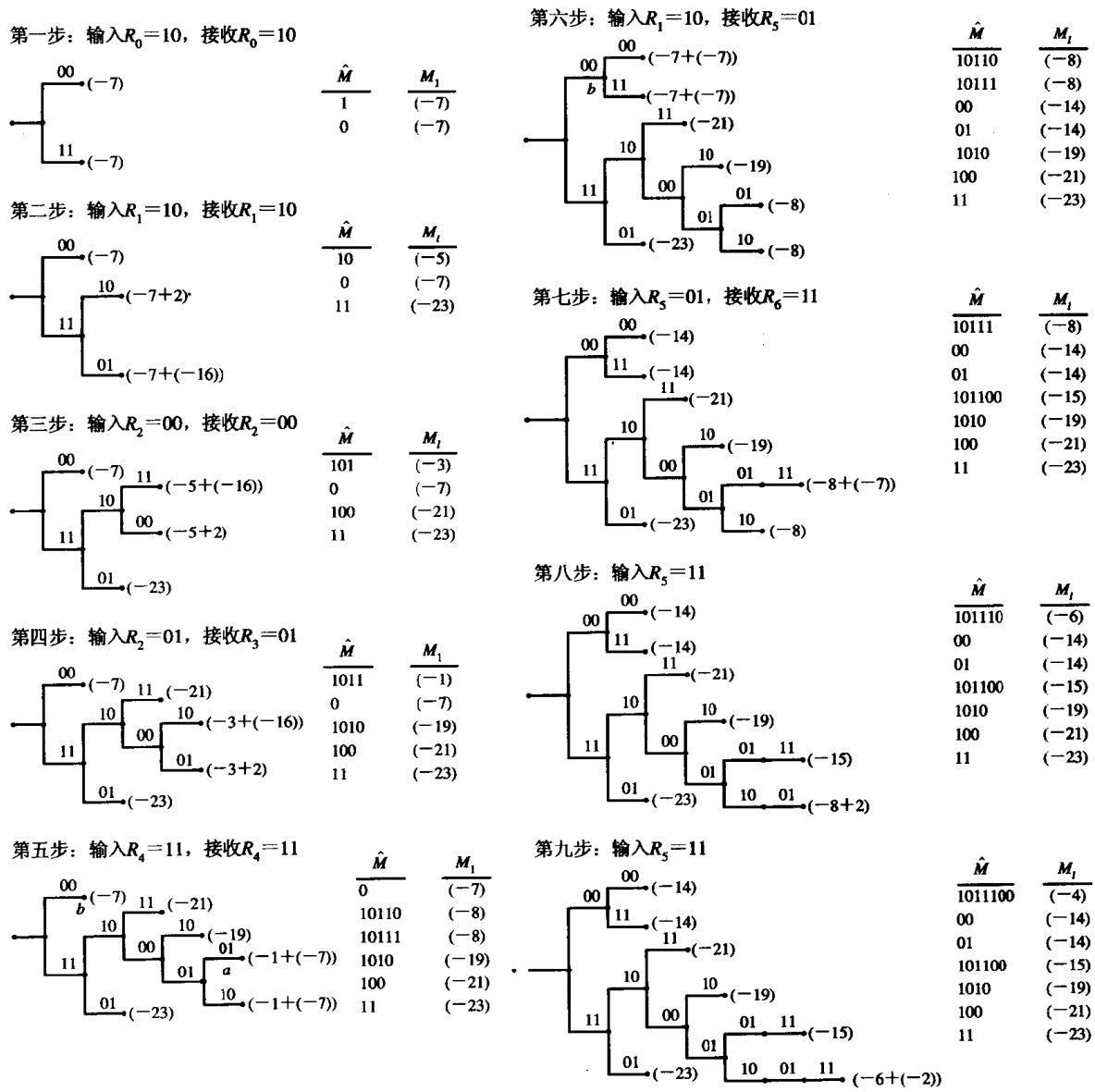


图 12-18 译 $R=(10, 10, 00, 01, 11, 01, 11)$ 的 ST 译码过程图

(3) 与 FA 算法同, 也必须有一个输入缓存器, 存贮输入给译码器的接收序列 R , 以便在译码过程中, 译码器由一个节点跳向另一个节点时(如例12.3中的第六步由 a 节点跳向 b 节点), 使用以前接收的码段, 及存贮此时刻以后接收的码段。一般说来, 该存贮器的容量需 $2n_0L$ 个码元的大小, 也就是说需两帧数据的容量。但是, 即使如此大的容量, 如果信道干扰很严重, 译码器搜索时间很长, 就可能引起缓存器溢出, 从而损失信息。对序列译码来说, 无论是 FA 还是 ST 译码, 所选用码的 d_f 很大, 因而译码错误概率不是主要的, 而缓存器的溢出却是一个主要的问题。当产生缓存器溢出时, 译码器有两种处理方法: 一是通过译码器删除此帧消息并经反馈信道(如有的话)要求发端重发这一帧消息; 二是向用户送出一告警信号指示这一帧消息有误。像这种处理方法, 比译码错误所引起的后果要好得多。而 VB 译码, 由于是完备译码, 因此不可能采用这种删除译码方法。

在 FA 译码算法中，输入缓存器会产生溢出，同样列表式存贮器(SS)也会产生溢出。为了解决这个问题，译码中当要产生溢出时就把 SS 中最低端的数据(也就是有最小 M_i 的路径)抛弃，这样处理结果，就使得这条被抛弃的路径永远再不会被译码器利用了，因而可能影响整个译码器的性能，但是若 SS 的容量足够大(例如1 000组)，则对性能的影响可忽略不计。

(4) 与 FA 算法相比，ST 算法的速度更快，但 FA 算法不需要 SS 存贮器。此外，在 FA 算法中，搜索分离点是在码树上逐个进行的，也就是按照节点的阶数由高至低逐个搜索，而不像 ST 算法中，搜索分离点是跳动的，即可以从高阶节点很快跳到另一阶节点。由于 FA 算法不需要 SS 存贮器，且译码速度也比较高，因此在实用中 FA 算法比 ST 算法用得较多。

为了节省 SS 的容量，以及减少每步译码时重新安排 SS 中次序的时间，杰利内克提出了几种修正 ST 算法，有关这些算法的工作原理，可参阅有关文献^[1]。

* § 12.6 序列译码的性能

分析序列译码的性能比分析 VB 译码更困难，因此下面仅介绍结果，而不作任何分析与证明。衡量序列译码性能的指标主要有：译码计算量和计算分布，译码器的不可检测错误概率(即译码器的输出误码率)，以及输入缓存器的溢出概率(也称删除概率)。下面所介绍的结果都是针对 FA 算法而言，但也适应于其它序列译码算法，如 ST 算法等。

在码树中与正确路径在第 j ($0 \leq j \leq L-1$) 阶节点分离的所有不正确路径集合，称为第 j 阶不正确子集，如图 12-12 中所示，这里， L 表示一帧消息序列的长度(分支为单位)。设 c_j 表示完成第 j 阶不正确子集搜索所需的计算次数，则对所有卷积码字集合而言， c_j 的平均概率分布也就是完成第 j 阶正确路径译码所需计算次数大于 n 次的概率^[1]

$$P_r(c_j \geq n) \approx A_n^{-\rho} \quad (12.6.1)$$

式中， A 是一个与译码方式有关的常数。 $0 < \rho < \infty$ ，它与码率 R 的关系如下：

$$R = \frac{E_0(\rho)}{\rho} \quad 0 < R < C \quad (12.6.2)$$

式中， C 是信道容量(bit/s)， $E_0(\rho)$ 称为加拉格尔(Gallager)函数。对于二进制对称输入的 DMC 而言

$$E_0(\rho) = \rho - \log_2 \frac{1}{2} \sum_j [p(j|0)^{\frac{1}{1+\rho}} + p(j|1)^{\frac{1}{1+\rho}}]^{1+\rho} \quad (12.6.3)$$

式中， $p(j|i)$ 是信道的转移概率。对于信道转移概率为 p 的 BSC 而言

$$E_0(\rho) = \rho - \log_2 [p^{\frac{1}{1+\rho}} + (1-p)^{\frac{1}{1+\rho}}]^{1+\rho} \quad (12.6.4)$$

由式(12.6.1)可知，序列译码的计算概率分布服从帕莱多(Pareto)分布，称 ρ 为 Pareto 指数。若码率 R 已知，则 ρ 可通过式(12.6.2)~式(12.6.4)计算得到。若 $\rho=1$ ，则由上述这些式子可得

$$R = E_0(1) = 1 - \log_2 \frac{1}{2} \sum_j [\sqrt{p(j|0)} + \sqrt{p(j|1)}]^2$$

对于 BSC

$$R = E_0(1) = 1 - \log_2[1 + 2\sqrt{p(1-p)}]$$

由式(12.6.2)可知, $R \rightarrow 0$ 时 $\rho \rightarrow \infty$, 而 $R \rightarrow C$ 时 $\rho \rightarrow 0$, 即 R 与 ρ 成反比。对序列译码来说, $E_0(1)$ 具有特别重要的意义, 它与计算概率分布密切相关。可以证明, 当 $\rho \leq 1$ 时, 序列译码器译每一分支的平均计算次数无限。因此, 为了使平均计算次数为有限, 必须使 $\rho > 1$, 这等价于码率

$$R = \frac{E_0(\rho)}{\rho} < E_0(1) = R_{\text{comp}} \quad (12.6.5)$$

称 $R_{\text{comp}} = E_0(1)$ 为计算截止速率。

例如, 对一个转移概率 $p = 0.045$ 的 BSC, 它的 $R_{\text{comp}} = 1/2$, 若应用 $R = 1/2$ 的码, 则要求信道的误码率低于 0.045, 才能保证 $R < R_{\text{comp}}$; 否则序列译码器的计算量要趋近 ∞ , 从而无法实现。这就是 R_{comp} 称为计算截止速率的意义。

序列译码计算分布是 Pareto 分布, 还说明了另一个问题。可以证明如果 $1 < \rho < 2$, 则译每一分支的平均计算次数虽然为有限, 但方差却是无限的。这说明序列译码器每一分支的计算量经常远大于平均计算次数, 说明序列译码不可能有效地处理突发噪声, 也说明若不采用其它措施(如交错)序列译码方法就不适用于突发信道。

序列译码器在 DMC 中, 译每一分支所需的平均计算次数 \bar{c} 可以由计算 c_i 的一阶矩得到, 也可以由其它分析方法得到为

$$\bar{c} \leq (q^{k_0} + 1) \frac{2^\Delta}{2^{\Delta/2} - 1} \cdot \frac{1}{1 - 2^{-(n_0/2)(R_{\text{comp}} - U)}} \cdot \frac{1}{1 - 2^{-(n_0/2)(R_{\text{comp}} + U - 2R)}} \quad (12.6.6)$$

这里要求路径度量偏置量

$$U < R_{\text{comp}}$$

编码码率

$$R < \frac{U}{2} + \frac{R_{\text{comp}}}{2}$$

(12.6.6) 式中, Δ 是门限增量。

序列译码的不可检测错误概率(输出误码率)

$$p_{ed} \leq 2^{-(m+1)n_0 E(R)} \quad R \geq R_{\text{comp}} < C \quad (12.6.7)$$

$$p_{ed} \leq A_1 2^{-(m+1)n_0(U/2 + R_{\text{comp}}/2)} \quad R < R_{\text{comp}} \quad (12.6.8)$$

对 ST 译码算法来说

$$p_{ed} \leq A(R) 2^{-(m+1)n_0 R_{\text{comp}}} \quad R < R_{\text{comp}} \quad (12.6.9)$$

上述这些式子中, $E(R)$ 就是式(1.4.1)中的误差指数 $E_c(R)$, 它是 R 的正实函数。 A_1 是由 n_0 、 U 、 R_{comp} 决定的常数, $A(R)$ 是 R 的一个有限增函数。

由上面 3 个式子说明, 在高码率情况下 ($R \geq R_{\text{comp}}$), 序列译码的输出误码率性能等于最大似然译码的性能。而当 $R < R_{\text{comp}}$ 时, 序列译码的性能接近于最大似然译码的性能。但由于 $R \geq R_{\text{comp}}$ 时, 序列译码是不能实现的, 因此序列译码在输出误码率的性能方面要稍次于维特比译码。但由式(12.6.1)、式(12.6.2)以及式(12.6.8)和式(12.6.9)可知, 序列译码的计算分布和平均计算次数均与 m 无关, 而 p_{ed} 却随 m 的增加而指数下降, 因此序列译码可采用大 m 的码(而 VB 译码的 m 不能太大)从而使序列译码的输出误码率可大大低于

VB译码时的输出误码率。

由式(12.6.1)可以很容易得到序列译码译 L 长分支为一帧的序列时, 输入缓存器溢出的概率(删除概率)

$$p_{Be} \leq LA_2(\mu B)^{-\rho} \quad (12.6.10)$$

式中, ρ 必须满足式(12.6.2), A_2 是与译码方式有关的常数, 经验证明它在 1 至 10 之间。 μ 是译码器的速度因子, 也就是接收一个子组时间内, 译码器完成 μ 个分支的计算, B 是输入缓存器的容量(以分支为单位)。由该式说明 p_{Be} 与码的 m 无关, 但通常比较大。例如若 $L=1000$, $A_2=3$, $\mu=10$, $B=10^4$, BSC 的转移概率 $p=0.08$, $R_{comp} < 3/8$, $R=1/3 < 3/8$, $\rho=1.31$ 。代入式(12.6.10)可得缓存器溢出概率 $p_{Be}=0.85 \times 10^{-3}$, 这说明序列译码器的缓存器溢出概率是很大的。如何减小 p_{Be} , 或者甚至消除缓存器溢出, 就成为序列译码中一个主要考虑的问题。

消除缓存器溢出的方法之一, 就是限制返回搜索层数技术。这种方法是当译码器由深入码树的最远节点, 开始返回搜索分离点时, 往后搜索节点的阶数(分支层数)不允许超过预先规定的最大值 J 。每当译码器向前进到一个新一阶节点时, 由此节点往后数至译码器所走路径上的第 $j \leq J$ 个分支所代表的 k_0 个信息元, 就由译码器输出至用户。每当返回搜索的分支层数超过最大限制数 J 时, 译码就不再搜索, 开始重新同步, 由译码器所在的节点跳到目前收到的分支上, 并且把以前接收到的相应分支所代表的信息元, 作为译码器输出送至用户。这种方法当然可能会引入译码错误, 但这是消除缓存器溢出所必须化的代价。

适当调整序列译码器的参数, 就可使计算量、 p_{ed} 、 p_{Be} 之间进行折衷。例如, 减低门限增量 Δ , 将使平均计算次数和 p_{Be} 增加, 但使 p_{ed} 减少。另一方面增加缓存器的容量 B , 将减少 p_{Be} , 但却使 p_{ed} 和计算次数增加。此外, 应用不同的度量标准, 也会影响序列译码的性能。

上面介绍的序列译码性能公式, 都是用随机码的计算方法得到的, 它们表示所有码集合的平均性能。下面介绍近几年以来, 由切维莱特(Chevillat)和科斯特洛(Costello)针对具体的 (n_0, k_0, m) 卷积码, 分析得到的序列译码性能公式, 这些公式建立了码的参数与性能之间的直接关系。

对转移概率为 p 的 BSC, 若 (n_0, k_0, m) 码的列距离函数为 d_i , 则计算分布

$$p_r(c_j \geq n) < \sigma N_d e^{-\mu d_i + \varphi} \quad (12.6.11)$$

式中, σ 、 μ 和 φ 仅是 p 和码率 R 的函数, 而

$$l = \lfloor \log_2 k_0 n \rfloor$$

N_d 表示有重量为 d_i 、长度为 $l+1$ 个分支的码序列的数目, 且码率

$$R < 1 + 2p \log_2 p + (1 - 2p) \log_2 (1 - p) = R_{max}$$

而序列译码器的输出错误概率

$$p_{ed} < \eta w_f e^{-\omega d_f} \quad (12.6.12)$$

式中, w_f 是重量等于最小自由距离的码组数目; 而 ω 与列距离增长速度有关, 当超过某一下限时为正值; η 与 ω 和信道有关。

式(12.6.11)说明为了使序列译码的 $p_r(c_j \geq n)$ 随 n 迅速下降, 也就是计算速度加快, 缓存器溢出概率减小, 就必须使用列距离特性好的码, 也就是列距离增长速度快的码。而

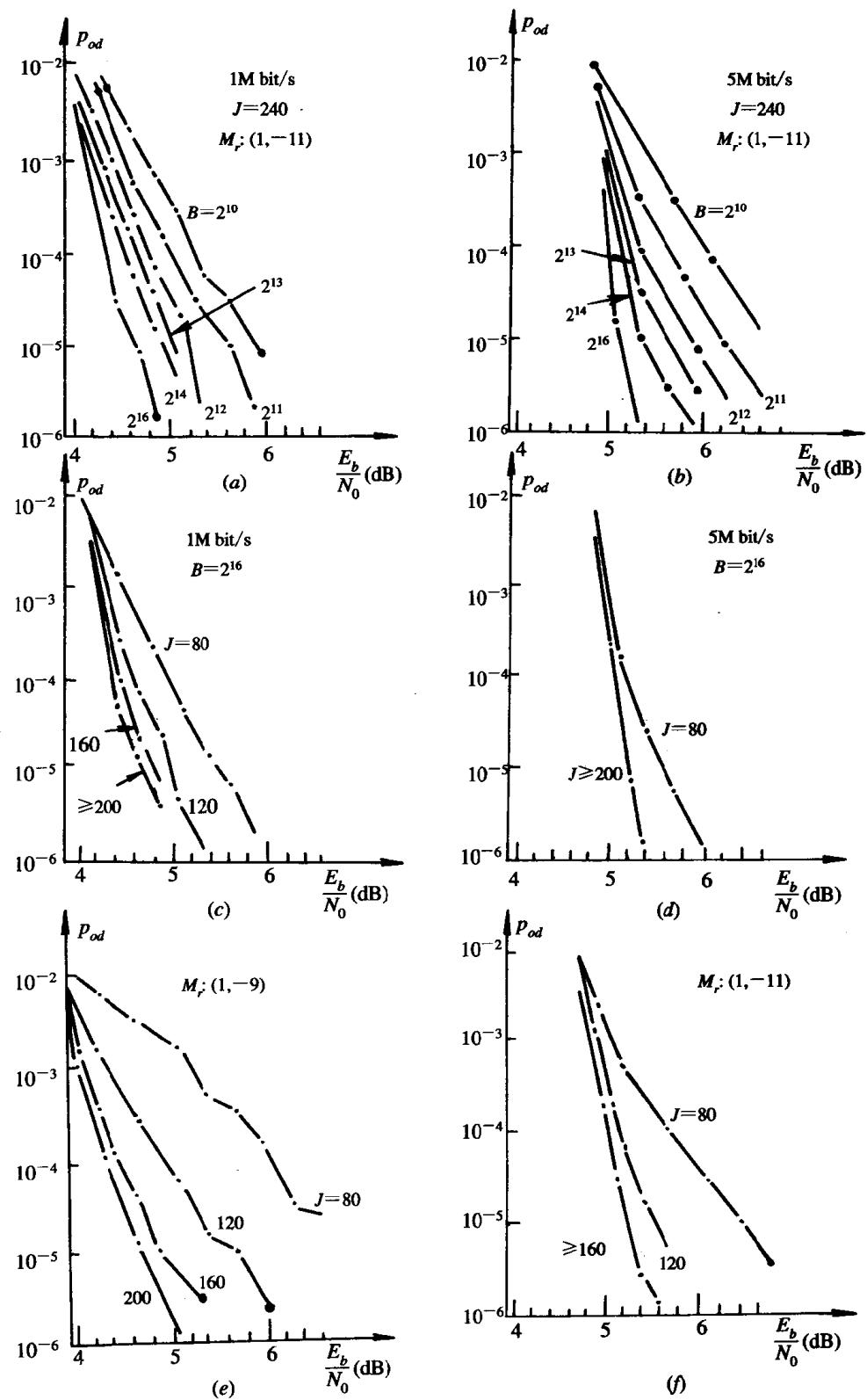


图 12-19 $(2, 1, 47)$ 卷积码应用序列译码的计算机模拟结果

式(12.6.12)说明,为了使 p_{ed} 减小,必须应用列距离特性好、并且有大自由距离的码。因此,式(12.6.11)和式(12.6.12)为我们选择适用序列译码方法的码提供了依据。

为了更清楚地了解序列译码器各参数与性能之间的关系,图 12-19 中列出了用计算机模拟(2, 1, 47)码在不同参数下, p_{ed} 与输入信噪比 E_b/N_0 之间的关系曲线。图(a)和(b)是在数据速率分别为 1 Mbit/s 和 5 Mbit/s 时,不同缓存器容量下的性能曲线。可以看出低速率的比高速率的要好 0.8 dB。图(c)和(d)是在数据速率分别为 1 Mbit/s 和 5 Mbit/s 时,不同最大返回限制层数 J 时的性能曲线。可以看出当 J 足够大时,低速率的比高速率的要好 0.6 dB 左右;但当 J 很小时,则改善不多。图(e)和(f)分别表示了用 M_F 为(1, -9)和 M_F 为(1, -11)比特 FA 度量时的性能曲线;在大 J 情况下,这二者的性能差不多;但 J 小时,则相差 1.1 dB。这说明应用(1, -9)的比特 FA 度量,在不正确路径被排除以前要进行更多次搜索,因而在搜索时返回层数更易超过 J ,从而使误码率增加。

下面我们比较一下序列译码和维特比译码的性能。由图 12-10(b)可知,当 m 由 2 至 7 时,应用 $R=1/2$ 最佳码,要求输出误码率 $p_e = 10^{-4}$,硬判决 VB 译码器要求 E_b/N_0 大约是 5.4~7 dB。这时译码器每接收一个分支,要完成 2^{mk_0} 次计算,这意味着译码器的速度因子是 4~128(非平行译码),且译码器必须有 2^{mk_0} 个存储容量为 32 个 bit 的存储器。而由图 12-19(a)看到,若应用 $R=1/2$ 的(2, 1, 47)码,当缓存器容量为 2^{10} 到 2^{16} 分支时,要得到 10^{-4} 的输出误码率,序列译码器要求 4.4~5.3 dB 的 E_b/N_0 ,比 VB 译码器改善了 1.0~1.7 dB,这时序列译码器的速度因子是 13.3,可与 VB 译码器相比较。但是应当指出,我们这种比较是很粗略的,还没有考虑这两种译码器所需要的其它控制电路。此外,如果考虑软判决译码,则似乎应用 VB 译码更为有利。

* § 12.7 适用于序列译码的码

由以上讨论可知,适用于序列译码的码不仅要求有大的自由距离 d_f ,而且要求列距离特性要好,也就是列距离的增加要快。

所谓码的列距离特性(CDP),定义为码的 0 至 m 阶列距离序列 d_0, d_1, \dots, d_m (参看定义 10.5.3)。我们说在相同 m 下,对某一个 $l(0 \leq l \leq m)$,一个码的距离序列 d_0, d_1, \dots, d_m 的 CDP,优于另一个码的距离序列 d'_0, d'_1, \dots, d'_m 的 CDP,是指

$$d_i \begin{cases} = d'_i & i = 0, 1, \dots, l-1 \\ > d'_i & i = l \end{cases}$$

换一句话说,一个码的 CDP 的初始阶的列距离,决定了码的列距离特性的好坏。

如果一个码的 CDP,优于任何有相同的 m 的其它码的 CDP,则称该码是一个最佳列距离特性码(ODP)。具有最大自由距离的 ODP 码就是序列译码所要求的码。

表 12-8 和表 12-9 中分别列出了 $R=1/2, 1/3, 2/3$,系统和非系统 ODP 码的参数及 d_f 。由这些表中看出,在同样 R 和 m 下,具有最佳列距离特性的非系统码的 d_f 比系统码要大得多。但由于系统码有快检特性,因此在某些场合下仍可能有些应用。由于序列译码的计算特性与 m 无关,所以可选用大 m 的码,从而弥补 d_f 不大的缺点,而在 VB 译码中就不可能选用大 m 的码,因而序列译码的输出误码率就可比 VB 译码低得多。

表 12-8 (a) 有ODP的 $R=1/2$ 的系统码

| m | $g^{(1, 2)}$ | d_f | m | $g^{(1, 2)}$ | d_f |
|-----|--------------|-------|-----|--------------|-----------|
| 1 | 6 | 3 | 19 | 7144616 | 12 |
| 2 | 7 | 4 | 20 | 7144761 | 12 |
| 3 | 64 | 4 | 21 | 67114544 | 12 |
| 4 | 72 | 5 | 22 | 71446166 | 14 |
| 5 | 73 | 6 | 23 | 67114543 | 14 |
| 6 | 734 | 6 | 24 | 714461654 | 15 |
| 7 | 714 | 6 | 25 | 671145536 | 15 |
| 8 | 715 | 7 | 26 | 671151433 | 16 |
| 9 | 7154 | 8 | 27 | 7144760524 | 16 |
| 10 | 7152 | 8 | 28 | 6711454306 | 16 |
| 11 | 7153 | 9 | 29 | 7144760535 | 18 |
| 12 | 67114 | 9 | 30 | 71446162654 | ≥ 16 |
| 13 | 67114 | 9 | 31 | 67114543066 | 18 |
| 14 | 67115 | 10 | 32 | 71447605247 | ≥ 18 |
| 15 | 714474 | 10 | 33 | 714461626554 | ≥ 18 |
| 16 | 671166 | 12 | 34 | 714461625306 | ≥ 18 |
| 17 | 671166 | 12 | 35 | 714461626555 | ≥ 19 |
| 18 | 6711454 | 12 | | | |

表 12-8(b) 有ODP的 $R=1/3$ 系统码

| m | $g^{(1, 2)}$ | $g^{(1, 3)}$ | d_f |
|-----|--------------|--------------|-------|
| 1 | 6 | 6 | 5 |
| 2 | 5 | 7 | 6 |
| 3 | 64 | 74 | 8 |
| 4 | 56 | 72 | 9 |
| 5 | 57 | 73 | 10 |
| 6 | 564 | 754 | 12 |
| 7 | 626 | 736 | 12 |
| 8 | 531 | 676 | 13 |
| 9 | 5314 | 6764 | 15 |
| 10 | 5312 | 6766 | 16 |
| 11 | 5312 | 6766 | 16 |
| 12 | 65304 | 71274 | 17 |
| 13 | 65306 | 71276 | 18 |
| 14 | 65305 | 71273 | 19 |
| 15 | 653764 | 712614 | 20 |
| 16 | 514112 | 732374 | 20 |
| 17 | 653761 | 712611 | 22 |
| 18 | 6530574 | 7127304 | 24 |
| 19 | 5141132 | 7323756 | 24 |
| 20 | 6530547 | 7127375 | 26 |
| 21 | 65376164 | 71261060 | 26 |
| 22 | 51445036 | 73251266 | 26 |
| 23 | 65305477 | 71273753 | 28 |

表 12-8(c) 有ODP的 $R=2/3$ 系统码

| m | K | $g^{(1, 3)}$ | $g^{(2, 3)}$ | d_f |
|-----|-----|--------------|--------------|-------|
| 1 | 1 | 4 | 6 | 2 |
| 2 | 2 | 5 | 7 | 3 |
| 3 | 3 | 54 | 64 | 4 |
| 4 | 4 | 56 | 62 | 4 |
| 5 | 5 | 57 | 63 | 5 |
| 6 | 6 | 554 | 704 | 5 |
| 7 | 7 | 664 | 742 | 6 |
| 8 | 8 | 665 | 743 | 6 |
| 9 | 9 | 5734 | 6370 | 6 |
| 10 | 10 | 5736 | 6322 | 7 |
| 11 | 11 | 5736 | 6323 | 8 |
| 12 | 12 | 66414 | 74334 | 8 |
| 13 | 13 | 57372 | 63226 | 8 |
| 14 | 14 | 57371 | 63225 | 8 |
| 15 | 15 | 664150 | 743314 | 8 |
| 16 | 16 | 664072 | 743346 | 10 |
| 17 | 17 | 573713 | 632255 | 10 |
| 18 | 18 | 604344 | 7431024 | 10 |
| 19 | 19 | 5514632 | 7023726 | 10 |
| 20 | 20 | 5514633 | 7023725 | 11 |
| 21 | 21 | 57361424 | 63235074 | 12 |
| 22 | 22 | 66415416 | 74311464 | 11 |
| 23 | 23 | 66415417 | 74311465 | 12 |

表 12-9(a) 有 ODP 的 $R=1/2$ 非系统码

| m | $\mathbf{g}^{(1, 1)}$ | $\mathbf{g}^{(1, 2)}$ | d_f |
|-----|-----------------------|-----------------------|-------|
| 1 | 6 | 4 | 3 |
| 2 | 7 | 5 | 5 |
| 3 | 74 | 54 | 6 |
| 4 | 62 | 56 | 7 |
| 5 | 75 | 55 | 8 |
| 6 | 634 | 564 | 10 |
| 7 | 626 | 572 | 10 |
| 8 | 751 | 557 | 12 |
| 9 | 7664 | 5714 | 12 |
| 10 | 7512 | 5562 | 14 |
| 11 | 6643 | 5175 | 14 |
| 12 | 63374 | 47244 | 15 |
| 13 | 45332 | 77136 | 16 |
| 14 | 65231 | 43677 | 17 |
| 15 | 517604 | 664124 | 18 |
| 16 | 717066 | 522702 | 19 |
| 17 | 506477 | 673711 | 20 |
| 18 | 5633664 | 7746714 | 21 |
| 19 | 4305226 | 6574374 | 22 |
| 20 | 6567413 | 5322305 | 22 |
| 21 | 67520654 | 50371444 | 24 |
| 22 | 67132702 | 50516146 | 24 |
| 23 | 55346125 | 75744143 | 25 |
| 31 | 21262405517 | 34217103047 | 28 |
| 31 | 35565573735 | 25565573735 | 23 |
| 31 | 36021201035 | 26021201035 | 21 |
| 31* | 25507054231 | 32724125333 | 29 |
| 31* | 27316266205 | 37703355527 | 30 |
| 31* | 26071365245 | 30357630067 | 30 |
| 31* | 20354742145 | 34466773157 | 30 |
| 31* | 20112717321 | 33506736023 | 30 |
| 31* | 20362647525 | 34440602437 | 29 |
| 31* | 27231545461 | 31651552361 | 29 |

表 12-9(b) 有 ODP 的 $R=1/3$ 非系统码

| m | $\mathbf{g}^{(1, 1)}$ | $\mathbf{g}^{(1, 2)}$ | $\mathbf{g}^{(1, 3)}$ | d_f |
|-----|-----------------------|-----------------------|-----------------------|-------|
| 1 | 4 | 6 | 6 | 5 |
| 2 | 5 | 7 | 7 | 8 |
| 3 | 54 | 64 | 74 | 10 |
| 4 | 52 | 66 | 76 | 12 |
| 5 | 47 | 53 | 75 | 13 |
| 7 | 516 | 552 | 656 | 16 |
| 31* | 25507054231 | 32724125333 | 36077207743 | 49 |
| 31* | 27316266205 | 37703355527 | 21604062735 | 48 |
| 31* | 26071365245 | 30357630067 | 34202713753 | 47 |
| 31* | 20354742145 | 34466773157 | 31342741175 | 47 |
| 31* | 20112717321 | 33506736023 | 27727174131 | 47 |
| 31* | 20362647525 | 34440602437 | 31362035615 | 48 |
| 31* | 27231545461 | 31651552361 | 35536444531 | 48 |

表 12-9(c) 有 ODP 的 $R=2/3$ 非系统码

| m | K | 生 成 序 列 | | | d_f |
|-----|-----|--|--|--|-------|
| 2 | 3 | $\mathbf{g}^{(1, 1)} = 6$ $\mathbf{g}^{(2, 1)} = 1$ | $\mathbf{g}^{(1, 2)} = 2$ $\mathbf{g}^{(2, 2)} = 4$ | $\mathbf{g}^{(1, 3)} = 4$ $\mathbf{g}^{(2, 3)} = 7$ | 4 |
| 2 | 4 | 6 1 | 3 5 | 7 5 | 5 |
| 3 | 5 | 60 34 | 30 74 | 70 40 | 6 |
| 3 | 6 | 50 24 | 24 70 | 54 54 | 6 |
| 4 | 7 | 54 00 | 30 46 | 64 66 | 7 |
| 4 | 8 | 64 26 | 12 66 | 52 44 | 8 |
| 5 | 9 | 54 25 | 16 71 | 66 60 | 8 |
| 5 | 10 | 53 36 | 23 53 | 51 67 | 9 |
| 6 | 11 | 710 320 | 260 404 | 670 714 | 10 |
| 8 | 12 | 740 367 | 260 414 | 520 515 | 10 |
| 8 | 13 | 710 140 | 260 545 | 670 533 | 11 |

(续表)

| m | K | 生 成 序 列 | | | d_f |
|-----|-----|---------|-------|-------|-------|
| 7 | 14 | 676 | 046 | 704 | 12 |
| | | 256 | 470 | 442 | |
| 8 | 15 | 722 | 054 | 642 | 12 |
| | | 302 | 457 | 435 | |
| 9 | 16 | 7640 | 2460 | 7560 | 12 |
| | | 0724 | 5164 | 4260 | |
| 9 | 17 | 5330 | 3250 | 5340 | 13 |
| | | 0600 | 7650 | 5434 | |
| 9 | 18 | 6734 | 1734 | 4330 | 14 |
| | | 1574 | 5140 | 7014 | |
| 10 | 19 | 5044 | 3570 | 4734 | 14 |
| | | 1024 | 5712 | 5622 | |
| 10 | 20 | 7030 | 3452 | 7566 | 14 |
| | | 0012 | 6756 | 5100 | |
| 11 | 21 | 6562 | 2316 | 4160 | 15 |
| | | 0431 | 4454 | 7225 | |
| 12 | 22 | 57720 | 12140 | 63260 | 16 |
| | | 15244 | 70044 | 47730 | |
| 12 | 23 | 51630 | 25240 | 42050 | 16 |
| | | 05460 | 61234 | 44334 | |

表中八进制数字所代表的子生成元表示方法，与表 12-2 的表示方法相同。表中有 * 的码是由我国编码工作者近年来用计算机搜索得到的“多准则兼优非系统码”。在同样 R 和 m 下，这类码不仅有更大的自由距离，且列距离特性非常好，还具有最大的反列距离，因此在序列译码中使用时能降低识别 180° 相位模糊度所需的计算量。故这类码特别适用于与 PSK、MSK 等调制器解调器相连的序列译码编译码器。

§ 12.8 调制与卷积码的结合(TCM 技术)

近代通信系统中，调制解调器与纠错码编译码器是两个主要的组成部分。它们也是提高通信系统信息传输速率、降低误码率的两个关键设备。为了满足目前对带宽越来越严格的要求，提高单位频带内传输信息的速率，目前主要沿两个方向发展：

(1) 研制频带利用率较高的调制方式。如高传输信息速率的多电平调幅和多进制调幅调相；特别是有较好频谱特性的连续相位调制(CPF)，如软调频(TFM)、最小频移键控调制(MSK)、广义 MSK(GMSK)、双正交相移调制(QPSK)以及互相关相移调制(XPSK)等。连续相移调制的主要优点是信号本身所占的频带较窄，带外辐射很低，因而产生的邻道干扰也很小。

(2) 提高原有通信线路，特别是电话线路传输数据的速率。发展适合在这些带宽有限(通常为 3 kHz)信道中高速传输信息的调制方式，如正交调幅(QAM)和多电平调制等。

数字通信系统中，传输信息的有效性和可靠性是一对矛盾。一般说来，传输信息的速率增加时，系统产生的误码率要增加。如何解决由于提高传输信息速率带来的误码率增加，是通信系统设计和实践中一个很有意义的研究方向。

1974年梅西根据山农信息理论最早证明了将编码与调制作为一个整体考虑时的最佳设计，就可大大改善系统的性能。昂格尔博克(Ungerbock)^[13]、今井秀樹^[12]等在70年代后期进行了这方面的研究，并于1982年提出了利用码率为 $n/(n+1)$ 的格状(Trellis)码(卷积码)，并将每一码段映射为有 2^{n+1} 个调制信号集中的一一个信号，在收端信号解调器后经反射变换为卷积码的码序列，并送入VB译码器译码的这样一种调制与编码相结合的方法^[13, 14]。在不增加带宽和相同的信息速率下可获得3~6 dB的功率增益。由于调制信号和卷积码都可看成是网络码，因此这种体制就称为格码或网格码调制(Trellis Coded Modulation)，简称TCM。

自TCM问世以来，它得到了广泛的重视，无论在理论研究和实际应用中都进展很快。1984年魏(L. F. Wei)提出的克服相位模糊的旋转不变码^[15]已被作为国际电报电话咨询委员会(CCITT)建议。利用TCM的9.6 kbit/s和14.4 kbit/s的高速解调器已进入市场。在理论研究上，近几年来从格与陪集码的观点提出了许多调制与卷积码、调制与分组码^[16, 19](BCM)相结合的各种方式，提出了多维TCM编码，使编码增益进一步提高^[17]；还研究了在非高斯信道中TCM的设计问题^[18]，以及采用非对称信号星座的TCM设计等问题。由于篇幅所限，本节不可能详细介绍以上这些内容，而仅仅讨论TCM的基本原理以及在实际中应用得较多的卷积码与正交调幅(QAM)和连续相位调制(CPM)相结合的方式。

一、一般概念与系统模型

自从数字通信问世以来，如何利用现有的、大量使用的电话信道高速传输数据，已成为一个很有实际意义的课题。在一个话音频带的电话信道中，根据测试其振幅相位平坦处大约只有2 400 Hz带宽。对于这种频带有限的电话信道，其主要特点之一是信噪比可以较高，典型的值是28 dB。根据山农信道容量公式(1.4.2)可知：

$$C = W \log_2 \left(1 + \frac{P_s}{W N_0} \right) \text{ (bit/s)}$$

当 $P_s/(WN_0) = S/N = 28$ dB, $W = 2400$ Hz时， C 大约为22.3 kbit/s。但是，事实上往往达不到如此高的信息传输速率，目前最多只能做到19.2 kbit/s。因此，如何在给定的信噪比和信道条件下，尽可能多和可靠地传递信息，是频带受限电话信道中存在的主要问题。应采用什么样的调制制度，才能达到理论上的22.3 kbit/s的传输信息速率，是一个令人感兴趣的课题。

一般数字已调信号为

$$s(t) = A \cos (2\pi f_0 t + \varphi)$$

式中， A 是信号的振幅， f_0 是信号的载频， φ 是相位。为了不扩展频带，要求 f_0 不变，因此只能通过改变 A 和 φ 提高传输信息的速率。在电话信道中，传输数据的有利条件是信噪比较高，可以用信噪比换取传输速率的提高。其办法就是应用多电平调幅(MAM)代替二进制调制，使每一信号携带信息的速率从 $1 \text{ bit}/T$ 提高到 $m \text{ bit}/T$ 。如从二电平调幅变成四电平调幅，每一信号携带信息的速率从 $1 \text{ bit}/T$ 提高到 $2 \text{ bit}/T$ 。这里， T 是信道传送信号的时

间宽度。在高斯白噪声信道中，此时若仍要求误码率不变，则发射信号的平均功率要求增加 7 dB。

增加传信率的另一办法是改变相位，利用多进制相移调制代替二进制。如以四进制相移(QPSK)代替二进制，信息传输速率可以提高一倍，但要求发射信号的平均功率仅增加 3 dB，比调幅的少。

表 12-10 给出了加性高斯白噪声信道(AWGN)中，在误码率相同情况下，多进制调幅(MAM)、调相(PSK)和 QAM 调制方式，随着传信率的提高其平均功率增加的情况，表中 R_T 是每信道符号传送信息的比特数目。

表 12-10 各种调制方式的传信率与平均功率的比较

| R_T (bit/T) | 2 | 4 | 8 | 16 | 32 |
|---------------|---|---|------|------|------|
| 平均功率(dB) | | | | | |
| 调制方式 | | | | | |
| MAM | 0 | 7 | 13.2 | 19.3 | |
| PSK | 0 | 3 | 8.3 | 14.3 | 22.2 |
| QAM | 0 | 3 | 7.4 | 10.0 | 13.0 |

可见，电话信道中使用的调制方式，其信号空间已从简单的一维空间(如图 12-20(a))，发展到二维空间，从调幅发展到调幅调相结合(AM-PM 或 QAM)的调制方式，如图 12-20(b)所示。虽然早在 60 年代初就提出了二维信号星座，但是到底选用何种星座才能使系统的误码率最低？这问题一直到 1974 年才解决。这时证明了在 AWGN 信道中，蜂窝形(等六边形)是最佳的信号星座。但是，随着传信率的提高，每一信道符号传输信息的比特数相应增加，即使采用最佳信号星座，信号点之间的最小间隔仍不断缩小。如果发送

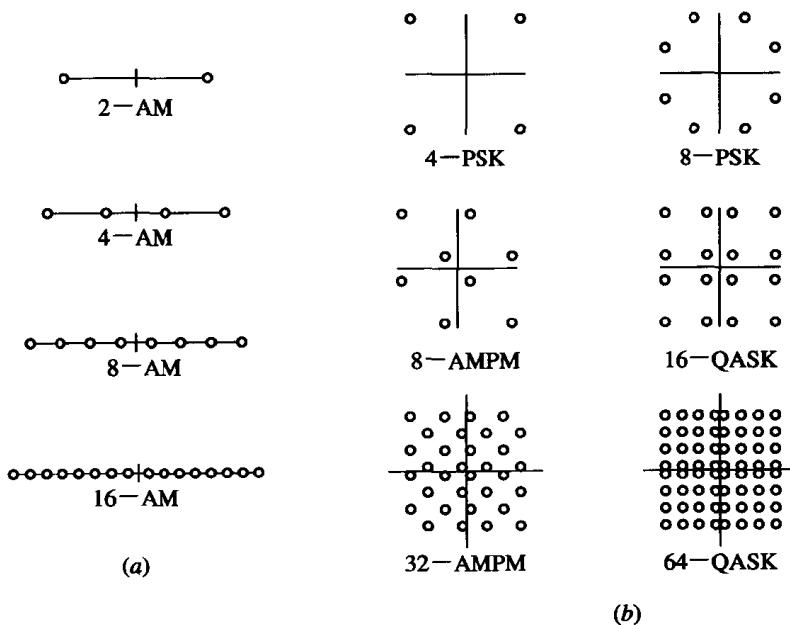


图 12-20 信号空间的星座表示

信号的平均功率不增加的话，系统的性能很快下降。为了在不增加信号的平均功率条件下，改善整个系统的误码率性能，在调制前对信号进行纠错编码，以增加信号点之间的距离(欧几里德距离)，增强抗干扰能力、提高系统的性能。

自 70 年代以来在这方面进行了许多工作，并取得了不少结果。如对四相相移键控、四电平调幅(4 AM)以及四进制调幅调相(4APK)系统，采用 $R=2/3$ 、有最大自由距离的卷积码，在调制前对信息进行编码，采用最佳信号星座，则进行编码后所得的编码增益，对4 PSK 和4 APK 可达2 dB 以上，对4 AM 可达1.5 dB 以上。

其实广义调制本身就蕴涵着纠错码技术，把调制与编码相结合作为一整体考虑，是当前通信系统设计中的必然趋势。纠错码可以用卷积码、分组码，但一般倾向于采用卷积码。这是由于用卷积码或分组码，获得的性能改善差不多，但卷积码比分组码更易于实现。

在具体讨论调制与编码如何结合以前，首先考虑图 12-21 所示的调制与编码作为一整体考虑的系统模型。

从信源输出的是二进制随机序列 $U = (u_0, u_1, \dots)$ ，经过码率为 R 的 (n_0, k_0, m) 卷积编码器，编码器相应的输出是二进制码序列 $V = (v_0, v_1, \dots)$ 。这里 $u_i = (u_i^1, u_i^2, \dots, u_i^{k_0})$ 是信息元输至编码器的信息组， $v_i = (v_i^1, v_i^2, \dots, v_i^{k_0})$ 是卷积编码器输出的子码或子组。图中的电平映射部分，是把二进制序列映射成后面调制器所需要的多电平序列 $\alpha = (\alpha_1, \alpha_2, \dots)$ 。可以有各种不同的映射，如用自然的二进制映射、格雷码映射等，映射方法的不同，对系统的性能有不同影响。

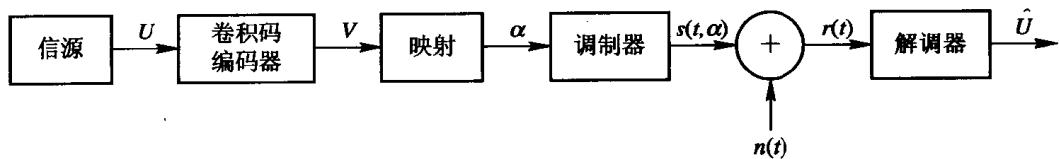


图 12-21 系统模型

系统中的最后一级是调制器，调制器输出的信号为 $s(t, \alpha)$ 。假定信号通过 AWGN 信道，则收端收到的信号是

$$r(t) = s(t, \alpha) + n(t)$$

式中， $n(t)$ 是均值为 0、单边功率谱密度为 N_0 的高斯白噪声。若收端采用最大似然相干检测(MLSE)，则解调器输出的错误概率为

$$P(\epsilon) = \frac{1}{S} \sum_{i=v}^S P(\epsilon | s_i) \leqslant \frac{1}{S} \sum_{i=0}^{S-1} \sum_{\substack{j=0 \\ i \neq j}}^{S-1} Q\left(\sqrt{d_{ij}^2 \frac{E_b}{N_0}}\right) \quad (12.8.1)$$

式中， S 是发送端输出的信号总数，也就是信号星座中的信号点数目， d_{ij} 是信号空间中 i 和 j 信号点之间的欧几里德距离(欧氏距离)，也就是信号星座中信号点之间的几何距离； E_b/N_0 是信噪比； $Q(x)$ 定义为

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{t^2}{2}} dt \quad (12.8.2)$$

当信噪比较大时，式(12.8.1)近似为

$$P(\epsilon) \approx CQ\left(\sqrt{d_{f\min}^2 \frac{E_b}{N_0}}\right) \quad (12.8.3)$$

式中, C 是与 E_b/N_0 无关的常数, $d_{f \min}^2$ 称为归一化自由欧氏距离。两个信号序列 α, β 之间的 $d_{f \min}^2$ 定义为

$$d_{f \min}^2 = \min_{\text{所有 } U_\alpha, U_\beta} \frac{1}{2E_b} \int_0^\infty [s(t, \alpha) - s(t, \beta)]^2 dt \quad (12.8.4)$$

式中, U_α 和 U_β 分别是输入纠错编码器的不同信息序列:

$$U_\alpha = (\dots, u_{\alpha 0}, u_{\alpha 1}, u_{\alpha 2}, \dots)$$

$$U_\beta = (\dots, u_{\beta 0}, u_{\beta 1}, u_{\beta 2}, \dots)$$

从式(12.8.3)看出, 系统的误码率决定于信号序列之间的自由欧氏距离 $d_{f \min}^2$, 而编码的作用就是使 $d_{f \min}^2$ 增加, 从而改善误码率。由以前讨论的纠错码理论可知, 通常用汉明意义上的汉明距离描述分组码或卷积码的抗干扰性能。但是, 当调制与编码作为一整体考虑后, 汉明意义上最大自由距离 d_f 的最佳卷积码, 不一定产生欧氏距离意义上的最佳卷积码, 也就是说有最大 d_f 的卷积码, 当它与调制结合后, 不一定有最大自由欧氏距离。因此, 如何针对不同的调制方式和映射规则, 寻找有最大 $d_{f \min}^2$ 的卷积码, 是编码与调制相结合中的一个最关键的问题。由于用分析方法很难寻找, 目前大都是用计算机搜索方法寻找。

为了更好地理解信号序列之间的欧氏距离, 下面先介绍信号篱笆图中路径重合的概念。正如 § 12.1 中所介绍的, 任何一个 (n_0, k_0, m) 卷积编码器, 都可以用它的篱笆图上的一条路径表示编码器的输出码序列。同样, 从调制器输出的信号序列, 也可以用它的信号篱笆图上的一条路径描述(因此, 从此意义上说, 由调制器输出的信号序列也是一类 Trellis 码)。图 12-22(a) 中, 画出了利用 $(2, 1, 1)$ 系统码卷积编码器与 2PSK(二进制移相调制)

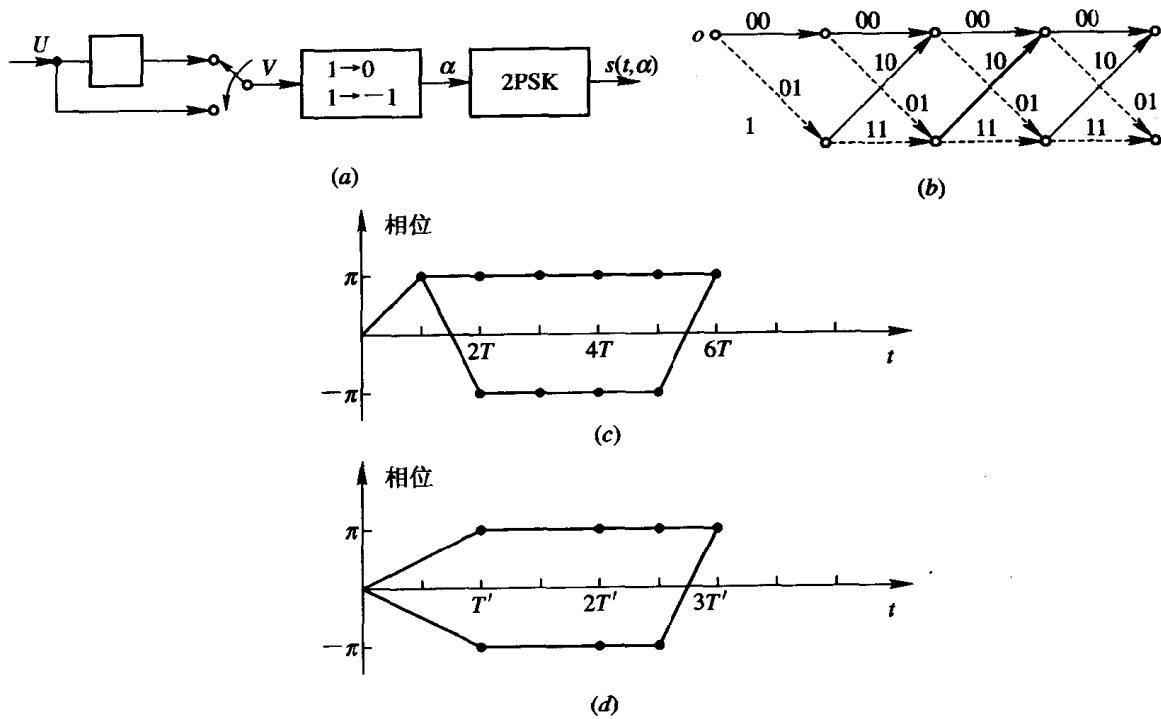


图 12-22 $(2, 1, 1)$ 码与 2PSK 相结合的框图和篱笆图

相结合的框图；而在图(b)和(c)中，分别画出了编码器的篱笆图和调制器的信号篱笆图。图(b)中的粗线相应于输入信息序列 $u_a = (000)$ 和 $u_b = (110)$ 时，对应的卷积编码器输出的码序列 $v_a = (00, 00, 00)$ 和 $v_b = (01, 11, 10)$ 。图(c)中的两条折线，就是 v_a 、 v_b 相应的信号路径，从编码器的篱笆图看到，相应于 u_a 和 u_b 的两条路径，在第三个编码时间单位（每一单位为两个信道码元时间）的 0 状态重合，在信号篱笆图上也相应于在第六个信道码元时间单位，在 π 状态两条信号路径重合（图中， T 为一个信道码元时间）。由此两个篱笆图看到，两条信号路径之间的欧几里德距离，与信号序列从开始到重合时路径中的分支数目有关（此例中为 6 个分支，相应于编码器篱笆图中的 3 个分支），称不重合的分支数为跨度，该例为 3。如果不用编码，则 $u_a = (000)$ 和 $u_b = (110)$ 在信号篱笆图上的两条路径如图中的(d)所示。可见，没有编码时两个信号序列之间的欧氏距离仅为 2，而利用编码后却增加到 4。所以，编码的作用，是使信号篱笆图中信号序列之间的欧几里德距离增加。TCM 设计的一个主要目标就是寻找与各种调制方式相对应的卷积码，当卷积码的每个分支与信号点映射后，使得每条信号路径之间有最大的欧几里德距离。

二、二、四进制调制的无记忆已调信号中使用的卷积码

已调信号的电平数不超过 4、且不存在记忆的调制方式有双正交移相(QPSK)、2PSK 和 ASK(振幅键控)等。由于这类调制方式的已调信号中不存在记忆，描述编码器输出序列的篱笆图与描述调制器输出信号路径的信号篱笆图结构基本类似。如果两个编码序列在篱笆图中的某一时刻重合，则相应的两条信号路径在信号篱笆图中也在同一时刻重合(参看图 12-22 的(b)和(c))。

对于 2PSK 和 ASK 来说，卷积码的汉明距离越大，则欧几里德距离越大，这是由于两个信号点与 0、1 两个码元之间完全一一对应。对于 QPSK，如果电平映射采用如图 12-23 所示的格雷码映射，则汉明距离与欧氏距离之间也有一一对应关系。因此，这类调制方式加上卷积码

以后，其信号序列间的欧氏距离与码序列间的汉明距离是一致的，只要应用 § 12.5 中所介绍的有最大 d_f 的非恶性的卷积码，并用软判决 Viterbi 译码，就能保证系统的总性能达到最佳。

三、 2^m ($m \geq 3$) 进制调制的无记忆已调信号中所用的卷积码

属于这一类的调制方式有 8PSK、16QAM 等。与前一类调制方式类似，当两个码序列在篱笆图上的某一时刻发生重合时，相应的两条信号路径在信号篱笆图上也在同一时刻重合。但由于在一个信道码元期间内，已调信号间可能的欧氏距离数，多于编码器输出序列中可能的汉明距离数，两者之间无法一一对应。如一个系统用 $(3, 2, m)$ 卷积编码器，后接一个 8 PSK 调制器组成。每输入二比特信息组时，编码器输出三个码元组成的码段，它们共有 8 种组合：(000, 001, 010, 011, 100, 101, 110, 111)。经某种映射规则，这 8 种组合

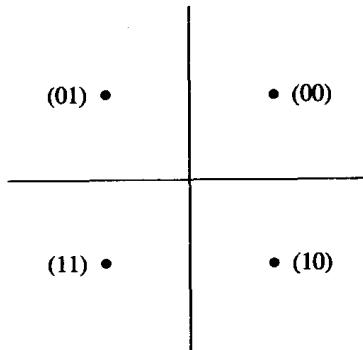


图 12-23 QPSK 的格雷码映射关系

中的每一种分别唯一地与8PSK信号空间中的每一信号点对应，如图 12-24 所示。不难看出，不同信号点之间的欧氏距离有 4 种： Δ_0 、 Δ_1 、 Δ_2 和 Δ_3 ，而 3 个码元组成的 8 个子码之间的汉明距离只有 3 种： $d_0 = 1$ ， $d_1 = 2$ ， $d_3 = 3$ 。因此，无论采用什么样的映射规则，总会出现这样一种情况，两个汉明距离较大的子码所对应的信号点之间的欧氏距离，比两个汉明距离较小的子码所对应的信号点之间的欧氏距离还要小。所以，不能保证汉明意义上的最佳卷积码，使已调信号之间的最小欧氏距离也最大。对大于八进制的 2^n 进制 QAM 调制方式，也存在类似的情况。

1982 年昂格尔博克对这种类型的系统进行了深入研究^[13]，提出了用“子集划分”方法，进行信息元与发射信号之间的变换，并且用计算机搜索了一批用这种“子集划分”方法得到的有最大欧氏距离的码，这类码称为 UB 码。

UB 码应用 $(n+1, n, m)$ 卷积码， n 比特的信息组进入编码器后，得到 $n+1$ 个码元组成的子码（或分支），且每一子码与信号星座中的一个信号点相对应，因此信号空间（星座）共有 2^{n+1} 个点。为了保证发送信号序列之间的欧氏距离最大，昂格尔博克将发送信号空间的 2^{n+1} 个点划分为若干子集，子集中信号点之间的最小欧氏距离随着划分次数的增加而加大： $\Delta_0 < \Delta_1 < \Delta_2 < \dots$ 。图 12-25 中给出了 8PSK 信号空间的划分情况。首先将 8 个信号点划分成 2 个子集： B_0 和 B_1 。每个子集中各含 4 个信号点，同一子集中的信号点之间的欧氏距离是 $\Delta_1 = \sqrt{2} = 1.414 > \Delta_0 = \sqrt{2 - \sqrt{2}} = 0.765$ 。再把 2 个子集中的每一个再

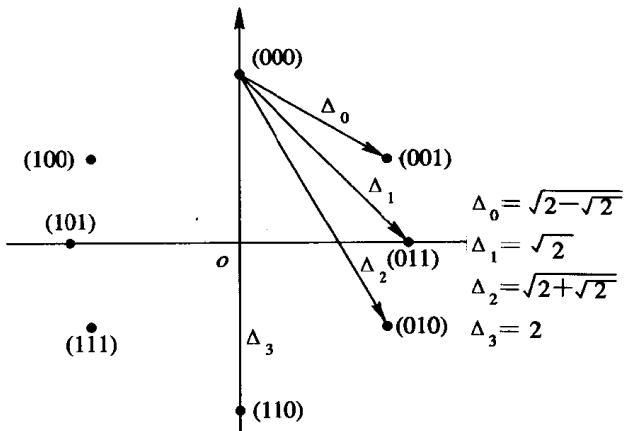


图 12-24 $(3, 2, m)$ 码的子码与 8PSK 信号空间中信号点的对应图

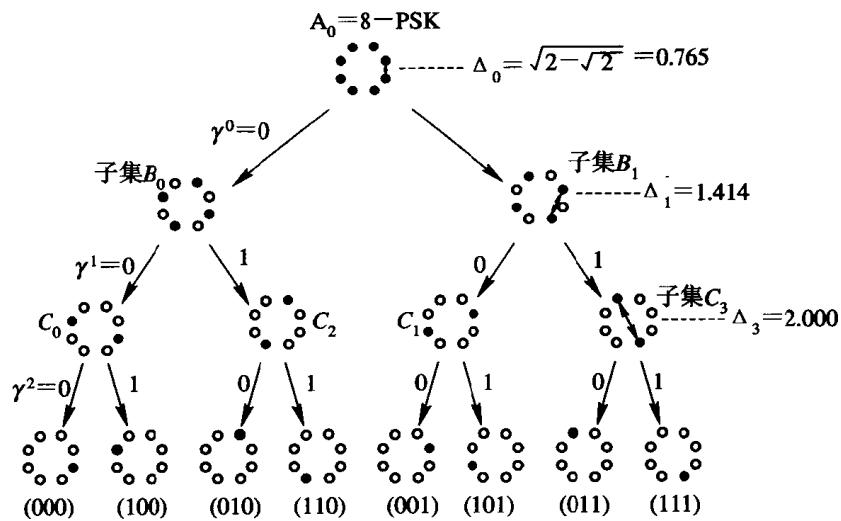


图 12-25 8PSK 信号空间划分成子集的情况

划分为 2 个子集，故共有 4 个子集： C_0 、 C_1 、 C_2 和 C_3 ，其中 $(C_0 \cup C_2) = B_0$ ， $(C_1 \cup C_3) = B_1$ 。4 个子集中的每一个各含有 2 个信号点，它们之间的欧氏距离是 $\Delta_3 = 2 > \Delta_1 > \Delta_0$ 。

得到了信号点的子集划分后，当卷积码编码器给定时，剩下的问题是如何使 2^{n+1} 个信号点与编码器输出的 2^{n+1} 个子码对应，才能进行恰当的映射，使已调信号之间的欧氏距离最大。图 12-26 给出了用 $(3, 2, 1)$ 码、 $(3, 2, 2)$ 码和 $(3, 2, 3)$ 码与 8 PSK 调制器相结合的情况。右边图中的左边的八进制数字，表示编码器的状态，而图中的数字表示分支值。其规则如下：

(1) 重合分支(子码)的输出信号点取自同一 C_i 子集。

(2) 进入某一状态的所有分支的输出信号取自 C_0 、 C_1 子集或 C_2 、 C_3 子集。

(3) 从某一状态出发的所有分支输出信号取自同一 B_i 子集。也就是说重合分支的信号点间应获得最大距离，而进入某一状态分支间的信号点之间和从某一状态出发的分支间的信号点之间获得次最大距离。

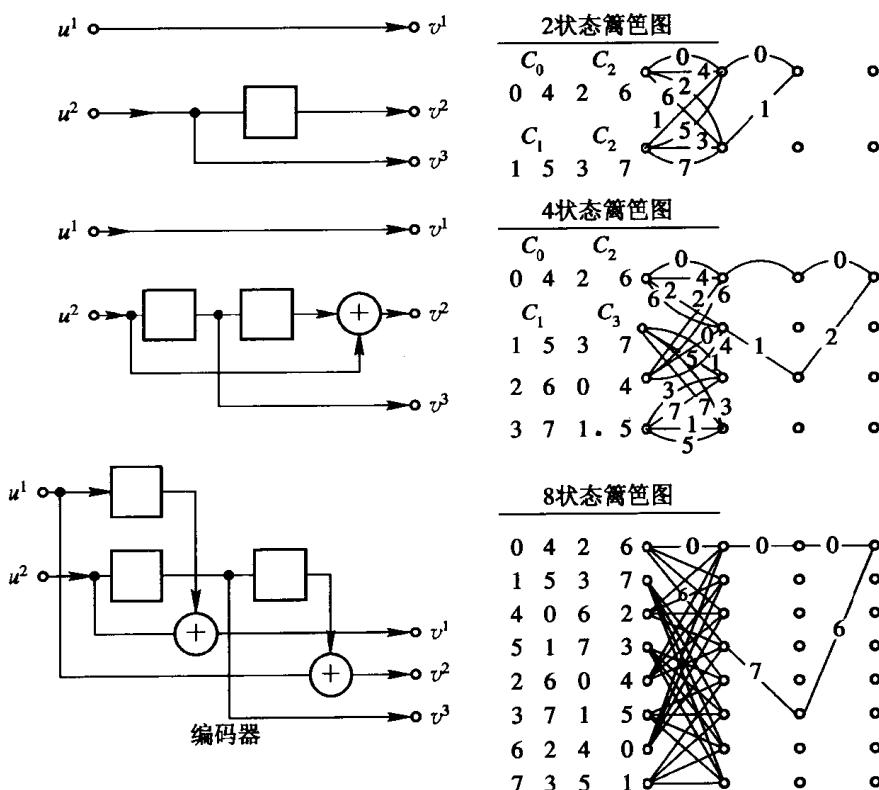


图 12-26 与 8PSK 相结合的编码器及其篱笆图

现以 $(3, 2, 1)$ 码为例说明。该码的编码器有两个状态：0 和 1。当编码器处于 0 状态时，输入的信息组若是 00 或 10，则编码器仍处于 0 状态，相应的输出是 000(0)或 100(4)；若输入的信息组是 01 或 11，则编码器从 0 状态转到 1 状态，输出的分支值是 010(2)或 110(6)。同理可知，编码器处在 1 状态时，若输入的信息组是 00 或 10，则从 1 状态转到 0 状态，相应的输出分支值是 001(1)或 101(5)；若输入的信息组是 11 或 01，则仍处在 1 状态，相应的输出分支值是 111(7)或 011(3)。如果输入的信息序列 $u_a = (00, 00)$ ，则

编码器输出的码序列是 $v_a = (000, 000) = (0, 0)$ (等式右边中的 0 是八进制数)。若另一信息组序列 $u_b = (01, 00)$ ，则编码器输出的码序列 $v_b = (010, 001) = (2, 1)$ 。由图 12-26 可知，此两个码序列在第 2 时刻重合，因而此二序列映射后得到的信号序列 $s(t, \alpha)$ 与 $s(t, \beta)$ ，也在第 2 时刻重合，它们之间的欧氏距离 $d_{f, \min}^2 = d_{f, \min}^2(C_0, C_1) + d_{f, \min}^2(B_0, B_1) = \Delta_1^2 + \Delta_0^2 = 2.586$ 。

如果信道是 AWGN，且解调器应用最大似然相干检测器解调，则八进制或 2^m ($m \geq 3$) 进制 PSK 调制方式的输出误码率为

$$P(e) = N(d_{f, \min})Q(d_{f, \min}/2\sigma) \quad (12.8.5)$$

相应的编码增益是

$$G = 10 \log \frac{(d_{f, \min}^2/P)_{\text{编码}}}{(d_{f, \min}^2/P)_{\text{未编码}}} \quad (\text{dB}) \quad (12.8.6)$$

式中， $N(d_{f, \min})$ 是最大自由欧氏距离为 $d_{f, \min}$ 的错误事件产生的数目， $Q(x)$ 由式 (12.8.2) 确定， σ^2 是噪声的方差， P 是平均信号功率。因此采用 $(3, 2, 1)$ 卷积码后，用 $d_{f, \min} = \sqrt{2.586} = 1.608$ 代入式 (12.8.5) 和式 (12.8.6)，就可得到系统输出的误码率，以及应用与没有应用编码时、相对每信道符号仍传送 2bit 的、4PSK 而言所获得的编码增益为 1.1dB。表 12-11 给出了利用不同编码贮存 m 的 $(3, 2, m)$ 卷积码并与调制相结合的 8PSK 系统，相对未应用编码的 4PSK 时所获得的编码增益。

表 12-11 编码与调制相结合的 8PSK 相对未编码的 4PSK 的编码增益

| 卷积码 | 状态数 | 编码增益(dB) |
|-------------|-----|----------|
| $(3, 2, 2)$ | 4 | 3.0 |
| $(3, 2, 3)$ | 8 | 3.6 |
| $(3, 2, 4)$ | 16 | 4.1 |
| $(3, 2, 5)$ | 32 | 4.6 |
| $(3, 2, 6)$ | 64 | 5.0 |
| $(3, 2, 7)$ | 128 | 5.4 |
| $(3, 2, 8)$ | 256 | 5.7 |

表 12-12 给出了用计算机搜索得到的与 8PSK 相结合的 $(3, 2, m)$ UB 码。表中的 $H^0(D)$ 、 $H^1(D)$ 、 $H^2(D)$ 分别是子校验多项式。如 $m=2$ 的 $(3, 2, 2)$ 码，它的 $H^0(D) = 5$ ， $H^1(D) = 2$ ， $H^2(D) = 0$ ，表示校验矩阵。

表 12-12 与 8PSK 调制相结合的最佳 $(3, 2, m)$ UB 卷积码

| m | $H^0(D)$ | $H^1(D)$ | $H^2(D)$ | $d_{f, \min}^2/\Delta_1^2$ | 相对 4PSK 的 $G(\text{dB})$ |
|-----|----------|----------|----------|----------------------------|--------------------------|
| 2 | 5 | 2 | 0 | 2.000 | 3.0 |
| 3 | 11 | 02 | 04 | 2.293 | 3.6 |
| 4 | 23 | 04 | 16 | 2.586 | 4.1 |
| 5 | 45 | 16 | 34 | 2.879 | 4.6 |
| 6 | 105 | 016 | 074 | 3.000 | 4.8 |
| 7 | 203 | 014 | 016 | 3.172 | 5.0 |
| 8 | 405 | 250 | 176 | 3.465 | 5.4 |
| 9 | 1 007 | 0 164 | 0 260 | 3.758 | 5.7 |

$$\mathbf{H}(D) = [\mathbf{H}^2(D), \mathbf{H}^1(D), \mathbf{H}^0(D)] = [0, 2, 5] = [0, D, 1 + D^2]$$

式中, $\mathbf{H}^i(D)$ 下的数字, 代表子校验多项式系数的八进制数表示。由 $\mathbf{G}(D) \cdot \mathbf{H}(D)^T = 0$, 可得此码的 $\mathbf{G}(D)$ 为

$$\mathbf{G}(D) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 + D^2 & D \end{bmatrix}$$

它就是图 12-26 中的(3, 2, 2)编码器。

必须注意 UB 卷积码一定是非恶性的, 表中给出的是非系统码形式, 由它们也可化成系统码形式, 此时的生成矩阵为

$$\mathbf{G}(D) = \begin{bmatrix} 1 & 0 & \mathbf{H}^2(D)\mathbf{H}^0(D)^{-1} \\ 0 & 1 & \mathbf{H}^1(D)\mathbf{H}^0(D)^{-1} \end{bmatrix}$$

所以(3, 2, 2)码的系统码形式的生成矩阵为

$$\mathbf{G}(D) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & \frac{D}{1 + D^2} \end{bmatrix}$$

与此相应的(3, 2, 2)码的系统码编码器如图 12-27 所示。

表 12-13 给出了与 8AMPM 相结合的(3, 2, m)UB 卷积码, 表中符号的意义与表 12-12 中的相同。

表 12-13 与 8AMPM 相结合的最佳 UB(3, 2, m)卷积码

| m | $\mathbf{H}^0(D)$ | $\mathbf{H}^1(D)$ | $\mathbf{H}^2(D)$ | $d_{f, \min}^2 / \Delta_1^2$ | 相对 4PSK 的 编码增益(dB) |
|-----|-------------------|-------------------|-------------------|------------------------------|-----------------------|
| 2 | 5 | 2 | 0 | 2.0 | 2.0 |
| 3 | 11 | 02 | 04 | 2.5 | 3.0 |
| 4 | 23 | 04 | 16 | 3.0 | 3.8 |
| 5 | 41 | 06 | 10 | 3.0 | 3.8 |
| 6 | 101 | 016 | 064 | 3.5 | 4.5 |
| 7 | 203 | 014 | 042 | 4.0 | 5.1 |
| 8 | 401 | 056 | 354 | 4.0 | 5.1 |
| 9 | 1 001 | 0 346 | 0 510 | 4.5 | 5.6 |

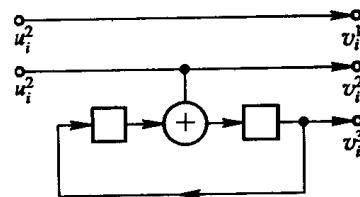


图 12-27 (3, 2, 2)码系统码编码器

四、恒包络连续相位调制中使用的卷积码

连续相位频移键控(CPFSK)、最小移频键控(MSK)、软调频(TFM)等均属恒包络连续相位调制方式。这类调制方式的已调信号的一般表示式是

$$s(t, \alpha) = \sqrt{\frac{2E}{T}} \cos(2\pi f_0 t + \varphi(t, \alpha) + \varphi_0) \quad (12.8.7)$$

这里携带信息的是相位

$$\varphi(t, \alpha) = 2\pi h \sum_{i=-\infty}^{\infty} \alpha_i q(t - iT) \quad (12.8.8)$$

式中, E 表示信道符号的能量, T 表示信道符号的时间宽度, f_0 是载频, φ_0 为初相, h 是调制指数, a_i 是由映射器输出至调制器的码元(可参看图 12-21), 它可以是二进制也可以是多进制电平。相位响应 $q(x)$ 定义为

$$q(x) = \int_{-\infty}^x g(\tau) d\tau$$

式中, $g(\tau)$ 称为频率脉冲响应, 它的持续时间是有限长度 LT , 即 $t < 0$ 和 $t > LT$ 时 $g(\tau) = 0$, 并进一步假定 $g(LT) = 1/2$, 可知这种已调信号是有记忆的。对 CPFSK

$$g(t) = t/T \quad 0 \leq t \leq T$$

在上述条件下, 如果 h 取值是有理数, 则每一信道符号的结束时刻到达的可能状态数目是有限的, 因而完全可以由信号篱笆图上的一条路径来描述输出的信号。不过与前两种情况不同, 在这种情况下, 描述编码器输出码序列的篱笆图与描述调制器输出的信号篱笆图, 在结构上并不一样, 信号篱笆图上的状态数目比编码器篱笆图上的状态数要多。由于相位的连续变化, 以及码之间的相关(记忆)性, 使编码器输出的两个序列在篱笆图上的某一时刻重合时, 对应的调制器输出的两条信号路径在信号篱笆图上并不一定重合。这是与前两种情况的主要不同点之一。如在图 12-22 中用 CPFSK 代替图中的 2PSK 方框, 编码器不变, 输入的信息序列仍为 $u_a = (000)$ 和 $u_b = (110)$, 编码器输出的序列也为 $v_a = (00, 00, 00)$ 和 $v_b = (01, 11, 10)$, 电平转换器(映射器)输出的序列是 $\alpha = (-1-1, -1-1, -1-1)$ 和 $\beta = (-11, 11, 1-1)$, 则它们在编码器篱笆图上的路径与在信号篱笆图上的路径分别如图 12-28 中的(a)、(b)所示。可知在 $t=6T$ 的编码信道码元时刻, 编码器篱笆图上的两条路径重合, 但在信号篱笆图上对应的两条路径并不重合。这正是这类有记忆已调信号与无记忆已调信号的主要不同点之一, 也是这类系统中, 汉明意义上的最佳卷积码并不能保证欧氏意义上的最佳, 也不能保证系统在总性能上最佳的原因。对于这类调制制度, 也只能靠计算机搜索得到, 在给定调制指数 h 下能使 $d_{f, \min}^2$ 最大的最佳卷积码。而且, 对于不同的调制方式, 一般而言它们的最佳卷积码是不同的, 必须针对不同的调制方式, 逐个搜索最佳码。

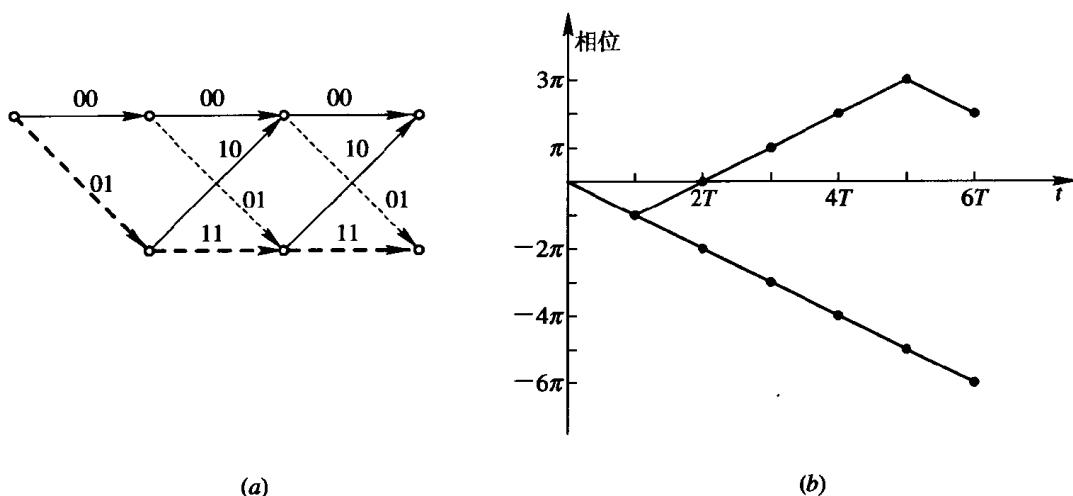


图 12-28 编码器篱笆图与 CPFSK 的信号篱笆图

针对 CPFSK 调制方式，目前已搜索到的最佳卷积码，仅局限于 $R = 1/n_0$ 的 $(n_0, 1, m)$ 卷积码，且必须满足以下两个条件：

- (1) 卷积码必须是非恶性的；
- (2) 子生成元 $\mathbf{g}^{(i,j)}(D) = g_{ij}^0 + g_{ij}^1 D + g_{ij}^2 D^2 + \dots + g_{ij}^m D^m$ ($i = 1, 2, \dots, k_0$; $j = 1, 2, \dots, n_0$)，对某一个 j 有 $g_{ij}^m = 1$, $g_{ij}^0 = 1$ 。

适用于 $M = 2^m$ 进制 CPFSK 的最佳卷积码，与其后面连接的映射器的映射方式共有 $M!$ 种，但只有其中的 $M!/2$ 种映射方式得到的信号之间的欧氏距离可能不相同，因此必须进一步研究码在不同映射方式下的欧氏距离。

若 $M = 2$ ，则仅只有一种映射方式： $0 \rightarrow -1$, $1 \rightarrow 1$ ，称为 B_1 映射方式。至于 $1 \rightarrow -1$, $0 \rightarrow 1$ 的映射方式，与 B_1 映射方式有相同结果。对于 $M = 4$ ，则有 4 种映射方式是我们感兴趣的，分为 Q1 与 Q2 两类，如表 12-14 所示。

表 12-14 四电平映射规则

| 映射器输入 | 映 射 器 输 出 | | | | | | | | | | | |
|-------|-----------|-----|-----|-----|-----|-----|----|-----|-----|-----|-----|-----|
| | Q1 | Q11 | Q12 | Q13 | Q14 | Q15 | Q2 | Q21 | Q22 | Q23 | Q24 | Q25 |
| 00 | -3 | -3 | -3 | -3 | -3 | -3 | -1 | -1 | -1 | -1 | -1 | -1 |
| 01 | -1 | -1 | 1 | 1 | 3 | 3 | -3 | -3 | 1 | 1 | 3 | 3 |
| 10 | 1 | 3 | 3 | -1 | -1 | 1 | 1 | 3 | 3 | -3 | -3 | 1 |
| 11 | 3 | 1 | -1 | 3 | 1 | -1 | 3 | 1 | -3 | 3 | 1 | -3 |

表 12-15(a)、(b) 给出了与二进制 CPFSK 调制方式相结合的、不同调制指数 h 下的最佳 $(2, 1, m)$ 卷积码的参数以及与 h 有关的最小自由欧氏距离 $d_{f \min}^2(h)$ ^[17]。“子生成元, N ”列中圆括号内的数字是编码器子生成元多项式系数的八进制数表示，花括号内的另一个数字，代表达到 $d_{f \min}^2(h)$ 所需的信道符号数(单位时间)。如 $m=1$, $h=0.05$ 时，查(a)表中“子生成元, N ”列的数字是 $\{(1, 2), 6\}$ ，说明该码的两个子生成元 $\mathbf{g}^{(1, 1)}(D)$ 、 $\mathbf{g}^{(1, 2)}(D)$ 的多项式系数是 001、010，所以 $\mathbf{g}^{(1, 1)}(D) = 1$, $\mathbf{g}^{(1, 2)}(D) = D$, $N = 6$ ，说明 6 个信道符号后的信号序列之间的最小欧氏距离达到表中给定的值 $d_{f \min}^2(h) = 0.129$ 。

表 12-16(a)、(b)、(c) 给出了与四进制 CPFSK 调制方式相结合的、不同调制指数 h 下的最佳 $(2, 1, m)$ 卷积码的子生成元、映射规则和达到 $d_{f \min}^2(h)$ 时所需的信道符号数。如 $h=0.05$ ，查(a)表的“子生成元, 映射, N ”列的数字是 $\{(1, 2), Q1, 4\}$ ，表示码的子生成元 $\mathbf{g}^{(1, 1)}(D) = 1$, $\mathbf{g}^{(1, 2)}(D) = D$ ，利用表 12-14 中的 Q1 映射规则，达到最大自由欧氏距离 $d_{f \min}^2(h) = 0.1935$ 时、所需的信道符号数是 4。至于与八进制至三十二进制 CPFSK 调制方式相结合的最佳卷积码，则请参阅有关文献^[18]。

图 12-29 给出了应用这些最佳卷积码的 CPFSK 相对没有用编码时和相对 MSK 所获得的编码增益。图中的横坐标 $2B_c T_b = 2BT_b/R$ ，这里 $2BT_b$ 是未用编码的 M 进制 CPFSK 所需的归一化双边带宽， $R = k_0/n_0$ 是码率。所以横坐标是应用编码后相对未应用编码时的带宽扩展因子。由图可知，编码与调制相结合后，对不同的 h 和 m 大约能获得 1~6 dB 的增益。

必须指出，在编码与调制相结合的各系统中，卷积码译码时都使用软判决维特比译码。否则，编码增益不可能达到上述各表中所列出的数值。

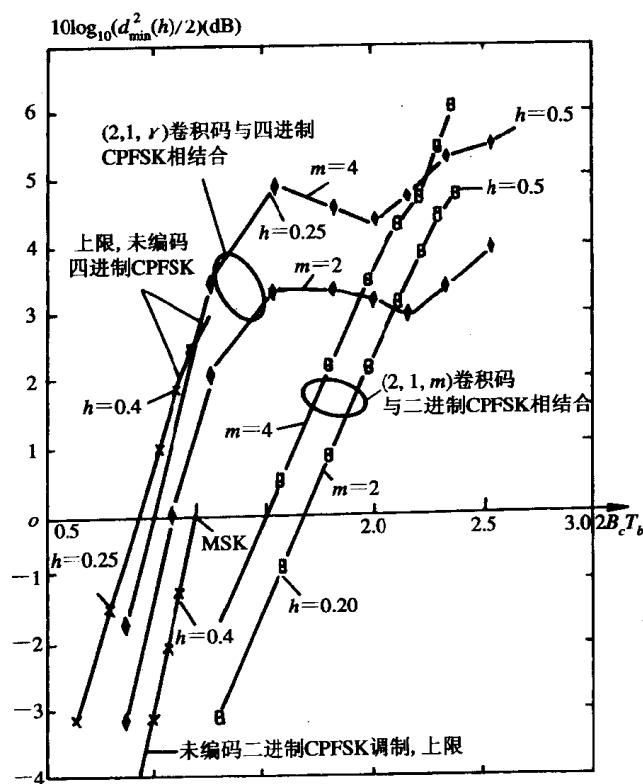


图 12-29 已编码 CPFSK 相对未编码 MSK 的增益

表 12-15 与二进制 CPFSK 相结合的最佳 $(2, 1, m)$ 卷积码及其 $d_{f\min}^2(h)$
(a)

| h | $m=1$ | | $m=2$ | |
|------|--------------|------------------|--------------|------------------|
| | {子生成元, N} | $d_{f\min}^2(h)$ | {子生成元, N} | $d_{f\min}^2(h)$ |
| 0.05 | {(1, 2), 6} | 0.1290 | {(4, 3), 22} | 0.1144 |
| 0.10 | {(1, 2), 6} | 0.4863 | {(4, 3), 22} | 0.4490 |
| 0.15 | {(1, 2), 6} | 0.9909 | {(4, 3), 22} | 0.9786 |
| 0.20 | {(1, 2), 6} | 1.532 | {(4, 3), 22} | 1.664 |
| 0.25 | {(1, 2), 6} | 2.000 | {(4, 3), 22} | 2.454 |
| 0.30 | {(1, 2), 8} | 2.312 | {(4, 3), 22} | 3.291 |
| 0.35 | {(2, 1), 8} | 2.852 | {(4, 3), 26} | 4.116 |
| 0.40 | {(2, 1), 10} | 3.341 | {(4, 3), 22} | 4.874 |
| 0.45 | {(2, 1), 12} | 3.732 | {(4, 3), 24} | 5.514 |
| 0.50 | {(2, 3), 12} | 4.000 | {(4, 3), 20} | 6.000 |
| 0.55 | {(2, 1), 18} | 4.130 | {(4, 3), 20} | 5.592 |
| 0.60 | {(2, 1), 10} | 4.121 | {(4, 3), 20} | 5.366 |
| 0.65 | {(2, 1), 10} | 3.984 | {(4, 3), 54} | 4.943 |
| 0.70 | {(2, 1), 10} | 3.741 | {(4, 3), 16} | 4.484 |
| 0.75 | {(2, 1), 8} | 3.000 | {(4, 3), 14} | 4.000 |
| 0.75 | {(2, 3), 8} | 3.000 | | |
| 0.80 | {(2, 1), 8} | 3.069 | {(4, 5), 14} | 3.652 |
| 0.85 | {(2, 1), 8} | 2.715 | {(5, 7), 14} | 3.536 |
| 0.90 | {(2, 1), 8} | 2.399 | {(5, 7), 16} | 3.368 |
| 0.95 | {(2, 1), 16} | 2.152 | {(5, 7), 30} | 3.175 |
| 1.00 | {(2, 3), 4} | 1.500 | {(5, 7), 12} | 2.500 |
| 1.00 | {(1, 3), 6} | 1.500 | | |

(b)

| h | m=3 | | m=4 | |
|------|--------------------------------|---------------------|--------------------------------|---------------------|
| | {子生成元, N} | $d_{f_{\min}}^2(h)$ | {子生成元, N} | $d_{f_{\min}}^2(h)$ |
| 0.05 | {(2, 11), 26}, {(12, 7), 32} | 0.1225 | {(31, 17), 34}, {(32, 17), 36} | 0.1553 |
| 0.10 | {(2, 11), 26}, {(12, 7), 32} | 0.4800 | {(31, 17), 34}, {(32, 17), 36} | 0.6090 |
| 0.15 | {(2, 11), 26}, {(12, 7), 32} | 1.043 | {(31, 17), 34}, {(32, 17), 36} | 1.326 |
| 0.20 | {(2, 11), 26}, {(12, 7), 32} | 1.766 | {(31, 17), 34}, {(32, 17), 36} | 2.252 |
| 0.25 | {(2, 11), 26}, {(12, 7), 32} | 2.590 | {(31, 17), 34}, {(32, 17), 36} | 3.317 |
| 0.30 | {(2, 11), 26}, {(12, 7), 32} | 3.450 | {(31, 17), 34}, {(32, 17), 36} | 4.441 |
| 0.35 | {(2, 11), 28}, {(12, 7), 32} | 4.278 | {(22, 13), 54}, {(34, 17), 78} | 5.380 |
| 0.40 | {(12, 7), 30} | 5.012 | {(22, 13), 32} | 5.905 |
| 0.45 | {(12, 7), 30} | 5.599 | {(30, 13), 36} | 6.950 |
| 0.50 | {(1, 14), 22}, {(1, 17), 22} | 6.000 | {(22, 13), 32}, {(22, 15), 34} | 8.000 |
| 0.50 | {(2, 11), 22}, {(2, 17), 22} | 6.000 | {(22, 31), 32}, {(30, 13), 34} | 8.000 |
| 0.50 | {(3, 15), 20}, {(4, 17), 22} | 6.000 | {(31, 17), 30}, {(32, 17), 34} | 8.000 |
| 0.50 | {(6, 13), 24}, {(10, 3), 22} | 6.000 | {(34, 17), 32} | 8.000 |
| 0.50 | {(10, 5), 22}, {(12, 7), 26} | 6.000 | | |
| 0.50 | {(12, 15), 26}, {(14, 7), 20} | 6.000 | | |
| 0.50 | {(14, 13), 24} | 6.000 | | |
| 0.55 | {(10, 3), 24} | 5.923 | {(32, 17), 32} | 7.422 |
| 0.60 | {(10, 3), 24} | 5.930 | {(22, 15), 28} | 6.344 |
| 0.65 | {(10, 3), 22} | 5.572 | | ... |
| 0.70 | {(10, 3), 20} | 5.051 | {(24, 7), 26} | 6.174 |
| 0.75 | {(14, 7), 20} | 5.000 | {(24, 7), 24}, {(31, 17), 22} | 5.712 |
| 0.75 | | | {(20, 35), 24}, {(32, 17), 24} | 5.712 |
| 0.75 | | | {(24, 37), 28}, {(30, 13), 26} | 5.712 |
| 0.75 | | | {(34, 27), 26} | 5.712 |
| 0.80 | {(13, 15), 18} | 4.663 | {(22, 31), 24} | 5.295 |
| 0.85 | {(13, 15), 20} | 4.606 | | ... |
| 0.90 | {(13, 15), 22} | 4.416 | {(23, 35), 26} | 4.679 |
| 0.95 | {(13, 15), 30} | 4.207 | | ... |
| 1.00 | {(7, 13), 20}, {(7, 15), 22} | 3.000 | | ... |
| 1.00 | {(13, 7), 20}, {(13, 15), 20} | 3.000 | | |
| 1.00 | {(13, 17), 16}, {(15, 7), 22} | 3.000 | | |
| 1.00 | {(15, 13), 20}, {(15, 17), 18} | 3.000 | | |

表 12-16 与四进制 CPFSK 相结合的最佳 $(2, 1, m)$ 卷积码及其 $d_{f\min}^2$

(a)

| h | $m=1$ | | $m=2$ | | $d_{f\min}^2(h)$ |
|------|----------------------|------------------|--|------------------|------------------|
| | {子生成元, 映射, N} | $d_{f\min}^2(h)$ | {子生成元, 映射, N} | $d_{f\min}^2(h)$ | |
| 0.05 | $\{(1, 2), Q1, 4\}$ | 0.1935 | $\{(7, 2), Q1, 13\}, \{(7, 5), Q2, 13\}$ | 0.2587 | |
| 0.10 | $\{(1, 2), Q1, 4\}$ | 0.7295 | $\{(7, 2), Q1, 13\}, \{(7, 5), Q2, 13\}$ | 0.9826 | |
| 0.15 | $\{(1, 2), Q1, 4\}$ | 1.486 | $\{(7, 2), Q1, 13\}, \{(7, 5), Q2, 13\}$ | 2.029 | |
| 0.20 | $\{(1, 2), Q1, 5\}$ | 2.298 | $\{(7, 2), Q1, 12\}, \{(7, 5), Q2, 12\}$ | 3.202 | |
| 0.25 | $\{(1, 2), Q1, 6\}$ | 3.000 | $\{(7, 2), Q1, 10\}, \{(7, 5), Q2, 10\}$ | 4.302 | |
| 0.30 | $\{(1, 2), Q1, 11\}$ | 3.468 | $\{(7, 2), Q1, 14\}, \{(7, 5), Q2, 14\}$ | 4.312 | |
| 0.35 | $\{(1, 2), Q1, 36\}$ | 3.389 | $\{(3, 4), Q1, 40\}$ | 4.164 | |
| 0.40 | $\{(1, 2), Q1, 5\}$ | 3.126 | $\{(1, 4), Q1, 6\}, \{(1, 5), Q2, 7\}$ | 3.911 | |
| 0.45 | $\{(1, 2), Q1, 7\}$ | 3.312 | $\{(1, 7), Q1, 8\}, \{(1, 6), Q2, 8\}$ | 4.296 | |
| 0.50 | $\{(1, 2), Q1, 3\}$ | 3.000 | $\{(1, 7), Q1, 7\}, \{(1, 6), Q2, 7\}$ | 5.000 | |
| 0.50 | $\{(3, 1), Q1, 3\}$ | 3.000 | $\{(3, 7), Q1, 7\}$ | 5.000 | |
| 0.50 | $\{(1, 3), Q2, 3\}$ | 3.000 | | | |
| 0.55 | $\{(1, 3), Q2, 4\}$ | 2.977 | $\{(1, 7), Q1, 7\}, \{(1, 6), Q2, 7\}$ | 4.346 | |
| 0.55 | | | $\{(3, 7), Q1, 7\}$ | 4.346 | |
| 0.60 | $\{(1, 3), Q2, 4\}$ | 2.998 | $\{(1, 4), Q1, 6\}, \{(1, 5), Q2, 6\}$ | 4.060 | |

(b)

| h | $m=3$ {子生成元, 映射, N} | $d_{f\min}^2(h)$ |
|------|--|------------------|
| 0.05 | $\{(13, 2), Q1, 20\}, \{(13, 11), Q2, 20\}, \{(13, 4), Q1, 26\}, \{(13, 17), Q2, 26\}, \{(13, 2), Q1, 20\}, \{(13, 11), Q2, 20\}, \{(13, 4), Q1, 26\}, \{(13, 17), Q2, 26\}, \{(13, 2), Q1, 20\}, \{(13, 11), Q2, 20\}, \{(13, 4), Q1, 23\}, \{(13, 17), Q2, 23\}$ | 0.3075 |
| 0.10 | $\{(13, 4), Q1, 23\}, \{(13, 17), Q2, 23\}$ | 1.171 |
| 0.15 | | 2.430 |
| 0.20 | | 3.863 |
| 0.25 | $\{(13, 4), Q1, 22\}, \{(13, 17), Q2, 22\}$ | 5.241 |
| 0.30 | $\{(3, 10), Q1, 20\}$ | 5.080 |
| 0.35 | $\{(3, 10), Q1, 26\}$ | 5.006 |
| 0.40 | $\{(3, 10), Q1, 13\}$ | 4.602 |
| 0.45 | $\{(1, 15), Q1, 12\}, \{(1, 14), Q2, 12\}$ | 5.514 |
| 0.50 | $\{(1, 15), Q1, 9\}, \{(1, 14), Q2, 9\}, \{(2, 15), Q1, 10\}, \{(2, 17), Q2, 10\}$ | 6.000 |
| 0.50 | $\{(3, 13), Q1, 9\}, \{(3, 15), Q1, 9\}, \{(3, 16), Q1, 8\}, \{(7, 13), Q1, 9\}$ | 6.000 |
| 0.50 | $\{(7, 14), Q2, 9\}, \{(13, 7), Q1, 9\}, \{(13, 14), Q2, 9\}, \{(17, 13), Q1, 9\}$ | 6.000 |
| 0.55 | $\{(1, 15), Q1, 11\}, \{(1, 14), Q2, 11\}, \{(2, 15), Q1, 10\}, \{(2, 17), Q2, 10\}$ | 5.611 |
| 0.55 | $\{(3, 13), Q1, 11\}$ | 5.611 |
| 0.60 | $\{(3, 10), Q1, 9\}$ | 4.751 |

(c)

| h | $m=4$ {子生成元, 映射, N } | $d_{f_{\min}}^2(h)$ |
|--------|------------------------|---------------------|
| 0.10* | {(23, 10), Q1, 27} | 1.362 |
| 0.20* | {(23, 10), Q1, 24} | 4.427 |
| 0.25 | {(23, 10), Q1, 24} | 6.151 |
| 0.25 | {(23, 33), Q2, 24} | 6.151 |
| 0.30* | {(7, 20), Q1, 19} | 5.702 |
| 0.35** | {(4, 23), Q1, 30} | 5.380 |
| 0.40* | {(4, 23), Q1, 15} | 5.881 |
| 0.45* | {(4, 23), Q1, 14} | 6.760 |
| 0.50** | {(4, 23), Q1, 13} | 7.000 |
| 0.55** | {(4, 23), Q1, 13} | 6.696 |
| 0.60* | {(4, 23), Q1, 11} | 5.761 |

综上所述, 根据不同调制方式及信道情况, 有两种类型的编码与调制结合的方式:

- (1) 适用于带宽有限而功率不太受限的信道, 如电话信道中使用的卷积码。在这种类型信道中利用功率优势, 应用多电平调幅或 QAM 调制, 采用 UB 码和 Viterbi 译码, 使系统从 2^n 电平调制增加到 2^{n+1} 电平调制, 在不增加信道带宽条件下, 保持整个系统的传信率不变, 并使误码率性能得到改善。这种系统是以信号的平均功率换取信道带宽的不扩展。(2) 适用于在功率受限而带宽不太受限信道中使用的卷积码, 如与连续相位调制方式相结合的各种 $(n+1, n, m)$ 卷积码。在这种情况下, 每信道符号传送信息比特数不变, 但信道带宽却相应增加, 以增加信道带宽和编译码设备的复杂性来换取整个系统性能的改善。

习 题

- 已知 $(3, 1, 2)$ 码的 $G(D) = [1 + D^2, 1 + D + D^2, 1 + D + D^2]$,
 - 对长为 $L=4$ 的信息序列画出篱笆图;
 - 求与信道序列 $M=(101100)$ 相应的码字;
 - 用硬判决 VB 译码器译接接收序列 $R=(111, 111, 000, 100, 000, 111)$ 。
- 第 1 题中的码字通过二进制输入、八电平均匀量化输出的 DMC 后, 得到的接收序列 $R=(764, 565, 032, 530, 311, 477)$, 利用最小软距离译码准则, 应用软判决 VB 译码器, 译该接收序列。
- 找出当 $C_1 = 1, C_2 = 10$ 时图 12-5 DMC 的整数度量表。应用这个整数度量表, 用 VB 译码器译接接收序列: $R=(1_11_11_2, 1_21_11_2, 0_10_20_2, 1_20_20_1, 0_20_10_1, 1_21_11_1)$ 。
- 若用表 12-1(b) 的整数度量表, 求用 VB 译码器译第 3 题中的接收序列, 并与第 3 题和第 2 题的结果进行比较。
- 考虑一个二进制输入、八进制输出的 DMC, 有转移概率 $P(r_i | c_i)$ 如下:

| $c_i \backslash r_i$ | 0_1 | 0_2 | 0_3 | 0_4 | 1_4 | 1_3 | 1_2 | 1_1 |
|----------------------|-------|-------|-------|-------|-------|-------|-------|-------|
| 0 | 0.434 | 0.917 | 0.167 | 0.111 | 0.058 | 0.023 | 0.008 | 0.002 |
| 1 | 0.002 | 0.008 | 0.023 | 0.058 | 0.111 | 0.167 | 0.197 | 0.434 |

求出该信道的 bit 度量表和整数度量表。

6. 一个生成矩阵

$$G(D) = [1 + D^2 + D^3, 1 + D + D^2 + D^3]$$

的(2, 1, 3)码,

(1) 画出 $L=4$ 的篱笆图;

(2) 设一个码组在题 5 中所述的 DMC 传输, 它的接收序列: $R = (1_2 1_1, 1_2 0_1, 0_3 0_1, 0_1 0_3, 1_2 0_2, 0_3 1_1, 0_3 0_2)$ 。试用 VB 译码器求出此码序列。

7. 考虑习题 6 的(2, 1, 3)码, 画出 $L=4$ 的码树图。设计一个转移概率 $p=0.045$ 的 BSC:

(1) 求出该信道的 FA 比特度量表和整数度量表;

(2) 利用 FA 算法译接序列: $R=(11, 00, 11, 00, 01, 10, 11)$ 。

8. 计算第 5 题中信道的 FA 比特度量表和整数度量表。

9. 利用 ST 算法译第 7 题中的接收序列。

(1) 利用第 7 题的 FA 整数度量表;

(2) 利用第 8 题的 FA 整数度量表。

10. 对第 5 题中的 DMC 信道计算 R_{comp} 。

参 考 文 献

- [1] [美]林舒、科斯特洛著:《差错控制编码:基础和应用》,(王育民、王新梅译),人民邮电出版社,1986。
- [2] 王新梅:《纠错码与差错控制》,人民邮电出版社,1989。
- [3] 程隆信、郑辉:软判决 Viterbi 分段译码及其性能模拟的研究,《电信技术研究》, No. 1, 1984。
- [4] 郑辉:卷积码的简化 Viterbi 译码算法,《电信技术研究》, No. 11, pp. 24—30, 1986。
- [5] T. Ishitani et al., "A Scarce-state-transition Viterbi decoder VLSI for bit error Correction", IEEE Journal of Solid-State Circuits, SC-22, No. 11, 1987.
- [6] Kuei-Ann Wen et al., "A new transform algorithm for Viterbi decoding" IEEE Trans. on Commun. Vol. 38, No. 6, 1990.
- [7] Y. Yasuda et al., "Optimum soft decision for Viterbi decoding", 5th International Conferenceon Digital Satellite Communications, pp. 251—258, 1981.
- [8] 今井秀樹:《符号理論》,電子情報通信学会編,1990。
- [9] J. B. Cain et al., "Punctured convolutional codes of rate $(n-1)/n$ and simplified maximum likelihood decoding", IEEE Trans. on IT, No. 1, pp. 97—100, 1979.
- [10] J. Hagenauer, "Rate compatible punctured convolutional codes," ICC' 87 , pp. 29. 1. 1—29. 1. 5, Seattle, U. S. A. , 1987.
- [11] J. Hagenauer et al., "The performance of rate—compatible punctured convolutional codes for digital mobile radio", IEEE Trans. on commun. , No. 7, pp. 969—980, 1990.
- [12] H. Ima, S. Hirakawa, "A new multilevel coding method using error—corrcting codes," IEEE Trans. on IT, Vol IT-23, No. 3, pp. 371—377, 1977.
- [13] G. Ungerboeck, "Channel coding with mutilevel / phase signals", IEEE Trans. on IT, No. 1, pp. 55—67, 1982.
- [14] G. Ungerboeck, "Trellis-coded modulation with redundant signal sets: Part I, Introduction; Part II, State of the art", IEEE Commun. Magazine, Vol. 25, No. 2, pp. 5—21, 1987.

- [15] L. F. Wei, "Rotationally m-variant convolutional channel coding with expanded signal space, Part I: 180°", IEEE J. Seleted. Areas Commun., Vol. SAC-2, pp. 659-671, 1984.
- [16] 高田豊雄等, ブロック位相偏移変調符号の線形構造と誤り特的解析, 信学志[A], Vol. J73-A, No. 2, pp. 314-321, 1990.
- [17] L. F. Wei, "Trellis-coded modulation with multidimensional constellations", IEEE Trans. on IT, Vol. IT-33, No. 4, pp. 483-501, 1987.
- [18] D. Divsalar, M. Simon, "The design of trellis coded MPSK for fading channels: performance criterin", IEEE Trans. on commun. Vol. 36, No. 9, 1988.
- [19] G. D. Forney, "Cost codes—Part I , Part II ", IEEE Trans. on IT, No. 5, pp.1123-1187, 1988.

第十三章 Turbo 码

Shannon 理论证明，随机码是好码，但是它的译码却太复杂。因此，多少年来随机编码理论一直是作为分析与证明编码定理的主要方法，而如何在构造码上发挥作用却并未引起人们的足够重视。直到 1993 年，Turbo 码的发现，才较好地解决了这一问题，为 Shannon 随机码理论的应用研究奠定了基础。

在本章中，首先，介绍 Turbo 码的提出与构成原理，介绍迭代反馈译码算法（包括 AWGN 信道与 Rayleigh 衰落信道下的译码）；然后，针对 Turbo 码编译码特性，对几个问题进行了说明；最后，介绍 Turbo 码在 3GPP 中的具体应用。

§ 13.1 Turbo 码的提出

Turbo 码，又称并行级联卷积码（PCCC），是由 C. Berrou 等在 ICC'93 会议上提出的。它巧妙地将卷积码和随机交织器结合在一起，实现了随机编码的思想；同时，采用软输出迭代译码来逼近最大似然译码。模拟结果表明，如果采用大小为 65535 的随机交织器，并且进行 18 次迭代，则在 $E_b/N_0 \geq 0.7$ dB 时，码率为 1/2 的 Turbo 码在 AWGN 信道上的误比特率（BER） $\leq 10^{-5}$ ，达到了近 Shannon 限的性能（1/2 码率的 Shannon 限是 0 dB）。因此，这一超乎寻常的优异性能，立即引起信息与编码理论界的轰动。图 13-1 中给出了 Turbo 码及其它编码方案的性能比较，从中可以看出 Turbo 编码方案的优越性。

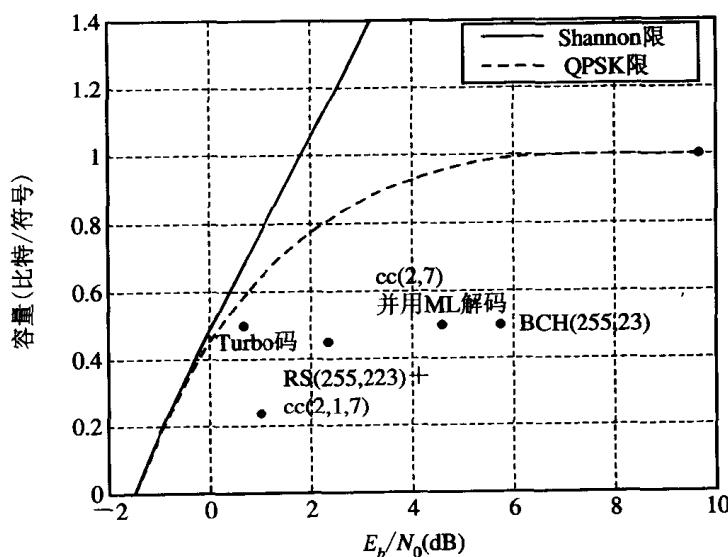


图 13-1 AWGN 信道中的码率与 Shannon 限

由于 Turbo 码的上述优异性能并不是从理论研究的角度给出的，而仅是计算机仿真的

结果。因此，Turbo 码的理论基础还不完善。后来经过不少人的重复性研究与理论分析，发现 Turbo 码的性能确实是非常优异的。因此，turbo 码的发现，标志着信道编码理论与技术的研究进入了一个崭新的阶段，它结束了长期将信道截止速率作为实际容量限的历史。

需要说明的是，由于原 Turbo 编译码方案申请了专利，因此在有关 Turbo 码的第一篇文章中，作者没有给出如何进行迭代译码的实现细节，只是从原理上加以说明。此后，P. Robertson 对此进行了探讨，对译码器的工作原理进行了详细说明。人们依此进行了大量的模拟研究。

Turbo 码的提出，更新了编码理论研究中的一些概念和方法。现在人们更喜欢基于概率的软判决译码方法，而不是早期基于代数的构造与译码方法，而且人们对编码方案的比较方法也发生了变化，从以前的相互比较过渡到现在的均与 Shannon 限进行比较。同时，也使编码理论家变成了实验科学家。

关于 Turbo 码的发展历程，C. Berrou 等在文献^[4]中给出了详细的说明。因为 C. Berrou 主要从事的是通信集成电路的研究，所以他们将 SOVA 译码器看作是“信噪比放大器”，从而将电子放大器中的反馈技术应用于串行级联的软输出译码器，并且为了使两个译码器工作于相同的时钟，以简化时钟电路设计，就提出了并行级联方式，这导致了 Turbo 码的发明。

尽管目前对 Turbo 码的作用机制尚不十分清楚，对迭代译码算法的性能还缺乏有效的理论解释，但它无疑为最终达到 Shannon 信道容量开辟了一条新的途径，其原理思想在相关研究领域中具有广阔的应用前景。目前，Turbo 码被看作是 1982 年 TCM 技术问世以来，信道编码理论与技术研究上所取得的最伟大的技术成就，具有里程碑的意义。

§ 13.2 Turbo 码编码器的组成

Turbo 码编码器是由两个反馈的系统卷积码编码器通过一个随机交织器并行连接而成的，编码后的校验位经过删余阵，从而产生不同码率的码字，见图 13-2。

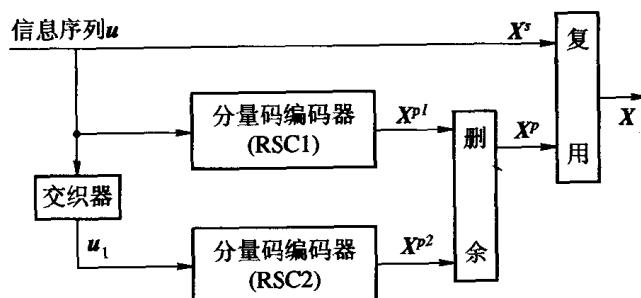


图 13-2 Turbo 码编码器结构框图

图 13-2 所示的是典型的 Turbo 码编码器结构框图，信息序列 $u = \{u_1, u_2, \dots, u_N\}$ 经过一个 N 位交织器，形成一个新序列 $u' = \{u'_1, u'_2, \dots, u'_N\}$ （长度与内容没变，但比特位置经过重新排列）。 u 与 u' 分别传送到两个分量码编码器（RSC 1 与 RSC 2）。一般情况下，这两个分量码编码器结构相同，生成序列 X^{p1} 与 X^{p2} 。为了提高码率，序列 X^{p1} 与 X^{p2} 需要经过

删余器，采用删余(puncturing)技术从这两个校验序列中周期地删除一些校验位，形成校验位序列 X^p 。 X^p 与未编码序列 X^e 经过复用调制后，生成了 Turbo 码序列 X 。例如，假定图 13-2 中两个分量编码器的码率均是 $1/2$ ，为了得到 $1/2$ 码率的 Turbo 码，可以采用这样的删余矩阵： $P = [1 \ 0, 0 \ 1]$ ，即删去来自 RSC 1 的校验序列 X^{p1} 的偶数位置比特与来自 RSC 2 的校验序列 X^{p2} 的奇数位置比特。

例 13.1 一个码率为 $1/3$ 的 Turbo 码编码器的组成框图如图 13-3 所示。

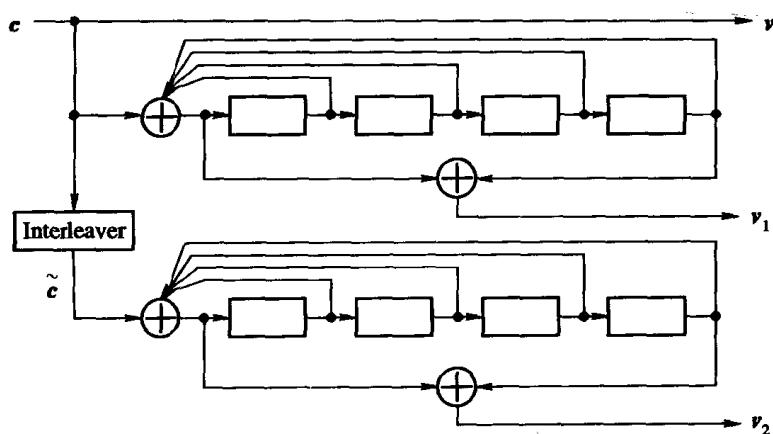


图 13-3 一个码率为 $1/3$ 的 Turbo 码编码器

图 13-3 所示的是基于 $(2,1,4)$ RSC(递归卷积系统码)的 Turbo 码编码器。分量码是码率为 $1/2$ 的寄存器级数为 4 的 $(2,1,4)$ RSC 码。其生成矩阵为

$$G(D) = [1, \frac{1+D^4}{1+D+D^2+D^3+D^4}] \quad (13.2.1)$$

我们假设输入序列为

$$c = (1011001) \quad (13.2.2)$$

则第一个分量码的输出序列为

$$\begin{aligned} v_0 &= (1011001) \\ v_1 &= (1110001) \end{aligned} \quad (13.2.3)$$

假设经过交织器后信息序列为

$$\tilde{c} = (1101010) \quad (13.2.4)$$

第二个分量码编码器所输出的校验位序列为

$$v_2 = (1000000) \quad (13.2.5)$$

则 Turbo 码序列为

$$v = (111, 010, 110, 100, 000, 000, 110) \quad (13.2.6)$$

§ 13.3 Turbo 码的译码

一、Turbo 码的迭代译码原理

由于 Turbo 码是由两个或多个分量码经过不同交织后对同一信息序列进行编码，对任

何单个传统编码，通常在译码器的最后得到硬判决译码比特，然而 Turbo 码译码算法不应局限于在译码器中通过的是硬判决信息。为了更好的利用译码器之间的信息，译码算法所用的应当是软判决信息而不是硬判决。一个由两个分量码构成 Turbo 码的译码器是由两个与分量码对应的译码单元和交织器与解交织器组成的，将一个译码单元的软输出信息作为下一个译码单元的输入；为了获得更好的译码性能，将此过程迭代数次。这就是 Turbo 码译码器的基本的工作原理。

二、Turbo 码译码器的组成

Turbo 码译码器的基本结构如图 13-4 所示。它由两个软输入软输出(SISO)译码器 DEC 1 和 DEC 2 串行级联组成，交织器与编码器中所使用的交织器相同。译码器 DEC 1 对分量码 RSC1 进行最佳译码，产生关于信息序列 u 中每一比特的似然信息，并将其中的“新信息”经过交织送给 DEC 2，译码器 DEC 2 将此信息作为先验信息，对分量码 RSC2 进行最佳译码，产生关于交织后的信息序列中每一比特的似然比信息，然后将其中的“外信息”经过解交织送给 DEC 1，进行下一次译码。这样，经过多次迭代，DEC 1 或 DEC 2 的外信息趋于稳定，似然比渐近值逼近于对整个码的最大似然译码，然后对此似然比进行硬判决，即可得到信息序列 u 的每一比特的最佳估值序列 \hat{u} 。

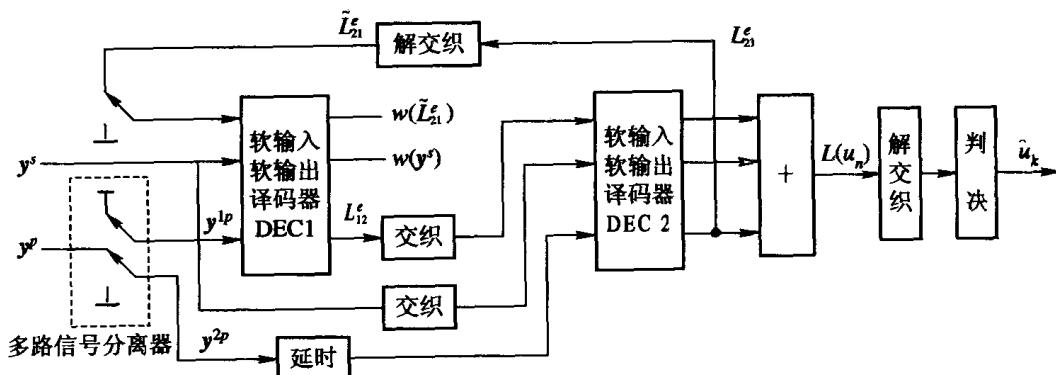


图 13-4 Turbo 码译码器的结构

假定 Turbo 码译码器的接收序列为 $y = (y^s, y^p)$ ，冗余信息 y^p 经解复用后，分别送给 DEC 1 和 DEC 2。于是，两个软输出译码器的输入序列分别为：

$$\text{dec } 1: y_1 = (y^s, y^{1p})$$

$$\text{dec } 2: y_2 = (y^s, y^{2p})$$

为了使译码后的比特错误概率最小，根据最大后验概率译码准则，Turbo 译码器的最佳译码策略是，根据接收序列 y 计算后验概率(APP) $P(u_k) = P(u_k | y_1, y_2)$ 。显然，这对于稍微长一点的码计算复杂度太高。在 Turbo 码的译码方案中，巧妙地采用了一种次优译码规则，将 y_1 和 y_2 分开考虑，由两个分量码译码器分别计算后验概率 $P(u_k | y_1, L_1^e)$ 和 $P(u_k | y_2, L_2^e)$ ，然后通过 DEC 1 和 DEC 2 之间的多次迭代，使它们收敛于 MAP 译码的 $P(u_k | y_1, y_2)$ ，从而达到近 Shannon 限的性能。这里， L_1^e 和 L_2^e 为附加信息。其中， L_1^e 由 DEC 2 提供，在 DEC 1 中用作先验信息； L_2^e 由 DEC 1 提供，在 DEC 2 中用作先验信息。

关于 $P(u_k | y_1, L'_1)$ 和 $P(u_k | y_2, L'_2)$ 的求解, 目前已有多种方法, 它们构成了 Turbo 码的不同译码算法。下面将以 BCJR 的前向-后向 MAP 软输出算法为例来讨论 Turbo 码的译码。

三、分量码的最大后验概率译码(MAP 算法)

考虑图 13-5 所示的软输入软输出(SISO)译码器, 它能为每一译码比特提供对数似然比输出。

在图 13-5 中, MAP 译码器的输入序列为 $y = y^N = (y_1, y_2, \dots, y_k, \dots, y_N)$ 。其中, $y_k = (y_k^s, y_k^p)$ 。 $L'(u_k)$ 是关于 u_k 的先验信息, $L(u_k)$ 是关于 u_k 的对数似然比。它们的定义如下:

$$L'(u_k) \equiv \ln \frac{P(u_k = 1)}{P(u_k = 0)} \quad (13.3.1)$$

$$L(u_k) \equiv \ln \frac{P(u_k = 1 | y_1^N)}{P(u_k = 0 | y_1^N)} \quad (13.3.2)$$



图 13-5 软输入软输出译码器框图

假定发送端 RSC 编码器的存储级数为 v , 约束长度为 K , 编码器在 k 时刻的状态为 $S_k = (a_k, a_{k-1}, \dots, a_{k-v+1})$, 编码输出序列为 $x = (x^s, x^p)$ 。传输信道模型如图 13-6 所示。

从图 13-6 可知:

$$y_k^s = a_k^s c_k^s + n_k^s = a_k^s (2x_k^s - 1) \sqrt{E_s} + n_k^s \quad (13.3.3)$$

$$y_k^p = a_k^p c_k^p + n_k^p = a_k^p (2x_k^p - 1) \sqrt{E_p} + n_k^p \quad (13.3.4)$$

式中, a_k^s 和 a_k^p 为信道衰落因子, 对于 AWGN 信道, $a_k^s = a_k^p = 1$; n_k^s 和 n_k^p 是两个独立同分布的高斯噪声样值, 它们的均值为 0, 方差 $\sigma^2 = N_0/2$ 。

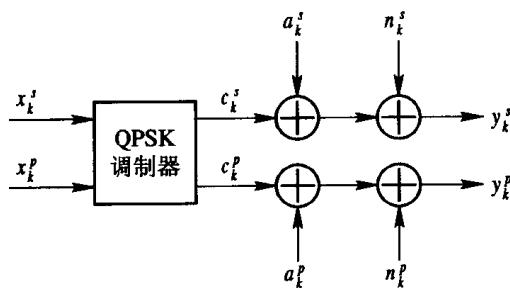


图 13-6 信道模型

MAP 译码器的任务就是求解式(13.3.3), 然后按照下列规则进行判决:

$$\hat{u}_k = \begin{cases} 1 & L(u_k) \geq 0 \\ 0 & L(u_k) < 0 \end{cases} \quad (13.3.5)$$

下面利用 BCJR 算法对式(13.3.2)的计算方法进行推导。

根据 Bayes 规则, 式(13.3.2)可以写为

$$\begin{aligned}
L(u_k) &= \ln \frac{P(u_k = 1, y_1^N) / p(y_1^N)}{P(u_k = 0, y_1^N) / p(y_1^N)} \\
&= \ln \frac{\sum_{\substack{(s', s) \\ u_k=1}} p(S_{k-1} = s', S_k = s, y_1^N) / p(y_1^N)}{\sum_{\substack{(s', s) \\ u_k=0}} p(S_{k-1} = s', S_k = s, y_1^N) / p(y_1^N)} \quad (13.3.6)
\end{aligned}$$

式中，求和是对所有由 $u_k=1$ （或 $u_k=0$ ）引起的 $S_{k-1} \rightarrow S_k$ 的状态转移进行的。根据 BCJR 算法^[12]， $p(S_{k-1}=s', S_k=s, y_1^N)$ 可以按下式计算：

$$p(s', s, y_1^N) = p(s', y_1^{k-1}) \cdot p(s, y_k | s') \cdot p(y_{k+1}^N | s) = \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s) \quad (13.3.7)$$

式中：

$\alpha_k(s) \equiv p(S_k=s, y_1^k)$ 为前向递推；

$\beta_k(s) \equiv p(y_{k+1}^N | S_k=s)$ 为后向递推；

$\gamma_k(s', s) \equiv p(S_k=s, y_k | S_{k-1}=s')$ 为 s' 和 s 之间的分支转移概率。

下面求 $\alpha_k(s)$, $\beta_k(s)$ 和 $\gamma_k(s', s)$ 。由定义得

$$\begin{aligned}
\alpha_k(s) &= \sum_{s'} p(S_k=s, S_{k-1}=s', y_1^k) \\
&= \sum_{s'} p(S_{k-1}=s', y_1^{k-1}) \cdot p(S_k=s, y_k | S_{k-1}=s', y_1^{k-1}) \quad (13.3.8)
\end{aligned}$$

考虑到 RSC 编码器等效于一个马尔可夫源，在状态 S_{k-1} 已知时，在 $k-1$ 时刻以后发生的事件与以前输入无关。因此，从式(13.3.8)可得前向递推公式

$$\alpha_k(s) = \sum_{s'} \alpha_{k-1}(s') \cdot p(S_k=s, y_k | S_{k-1}=s') = \sum_{s'} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \quad (13.3.9)$$

同样， $\beta_k(s)$ 可按下式反向递推得到

$$\begin{aligned}
\beta_{k-1}(s') &= \sum_s p(S_k=s, y_k^N | S_{k-1}=s') \\
&= \sum_s p(y_{k+1}^N | S_k=s) \cdot p(S_k=s, y_k | S_{k-1}=s') \\
&= \sum_s \beta_k(s) \cdot \gamma_k(s', s) \quad (13.3.10)
\end{aligned}$$

至于分支转移概率 $\gamma_k(s', s)$ ，从其定义可得

$$\gamma_k(s', s) = p(S_k=s | S_{k-1}=s') \cdot p(y_k | S_k=s, S_{k-1}=s') = P(u_k) \cdot p(y_k | u_k) \quad (13.3.11)$$

式中， $P(u_k)$ 是 u_k 的先验概率， $p(y_k | u_k)$ 由信道转移概率决定。

考虑到式(13.3.11)是从连续随机变量的概率密度计算得到的， $\gamma_k(s', s)$ 的值可能大于 1，这会使得式(13.3.9)和式(13.3.10)产生溢出，导致整个算法不稳定。因此，有必要对 $\alpha_k(s)$ 、 $\beta_k(s)$ 进行归一化处理。

令

$$\tilde{\alpha}_k(s) = \frac{\alpha_k(s)}{p(y_1^k)}$$

$$\tilde{\beta}_k(s) = \frac{\beta_k(s)}{p(y_{k+1}^N | y_1^k)} \quad (13.3.12)$$

因为 $p(y_1^k) = \sum_s p(S_k = s, y_1^k)$, 所以

$$\tilde{\alpha}_k(s) = \frac{\alpha_k(s)}{\sum_s \alpha_k(s)} \quad (13.3.13)$$

将式(13.3.9)代入上式, 并且分子分母同除以 $p(y_1^k)$, 得到

$$\tilde{\alpha}_k(s) = \frac{\sum_{s'} \alpha_{k-1}(s') \gamma_k(s', s) / p(y_1^{k-1})}{\sum_s \sum_{s'} \alpha_{k-1}(s') \gamma_k(s', s) / p(y_1^{k-1})} = \frac{\sum_{s'} \tilde{\alpha}_{k-1}(s') \gamma_k(s', s)}{\sum_s \sum_{s'} \tilde{\alpha}_{k-1}(s') \gamma_k(s', s)} \quad (13.3.14)$$

对于 $\tilde{\beta}_k(s)$, 考虑到 $p(y_k^N | y_1^k) = p(y_{k+1}^N | y_1^k) p(y_1^k) / p(y_1^{k-1})$, 于是有

$$\begin{aligned} \tilde{\beta}_{k-1}(s') &= \frac{\beta_{k-1}(s')}{p(y_k^N | y_1^{k-1})} = \frac{\sum_s \beta_k(s) \gamma_k(s', s)}{p(y_{k+1}^N | y_1^k) p(y_1^k) / p(y_1^{k-1})} \\ &= \frac{\sum_s \beta_k(s) \gamma_k(s', s) / p(y_{k+1}^N | y_1^k)}{\sum_s \alpha_k(s) / p(y_1^{k-1})} \\ &= \frac{\sum_s \tilde{\beta}_k(s) \gamma_k(s', s)}{\sum_s \sum_{s'} \alpha_{k-1}(s') \gamma_k(s', s) / p(y_1^{k-1})} \\ &= \frac{\sum_s \tilde{\beta}_k(s) \gamma_k(s', s)}{\sum_s \sum_{s'} \tilde{\alpha}_{k-1}(s') \gamma_k(s', s)} \end{aligned} \quad (13.3.15)$$

合并式(13.3.7)和式(13.3.12)得

$$\begin{aligned} p(s', s, y_1^N) &= \tilde{\alpha}_{k-1}(s') p(y_1^{k-1}) \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s) p(y_{k+1}^N | y_1^k) \\ &= \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s) \cdot p(y_1^{k-1}) p(y_1^N) / p(y_1^k) \\ &= \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s) \cdot p(y_1^N) / p(y_k | y_1^{k-1}) \end{aligned} \quad (13.3.16)$$

将上式代入式(13.3.6), 并且分子分母同乘以因子 $p(y_k | y_1^{k-1})$, 便得最终计算公式

$$L(u_k) = \ln \frac{\sum_{\substack{(s=s) \\ u_k=1}} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s)}{\sum_{\substack{(s,s) \\ u_k=0}} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s)} \quad (13.3.17)$$

这样, 就完成了分量码的 MAP 译码算法的推导。 $\tilde{\alpha}_k(s)$ 和 $\tilde{\beta}_k(s)$ 的递推运算示意于图 13-7。其中, $\tilde{\alpha}_k(s)$ 的初始条件为(假定 RSC 编码器的初始状态为零状态):

$$\begin{aligned} \tilde{\alpha}_0(0) &= 1 \\ \tilde{\alpha}_0(s \neq 0) &= 0 \end{aligned} \quad (13.3.18)$$

如果编码器在每帧编码完成之后通过结尾(termination)处理也回到零状态, 那么 $\tilde{\beta}_k(s)$ 递推的初始条件为:

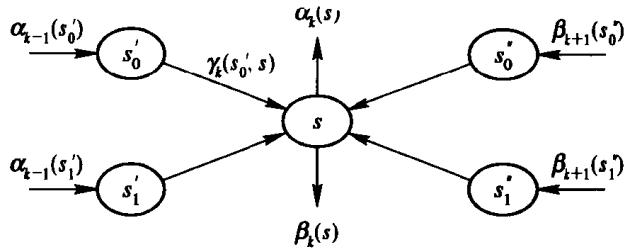


图 13-7 $\tilde{\alpha}_k(s)$ 和 $\tilde{\beta}_k(s)$ 的递推示意图

$$\begin{aligned}\tilde{\beta}_N(0) &= 1 \\ \tilde{\beta}_N(s \neq 0) &= 0\end{aligned}\quad (13.3.19)$$

否则，应设定为

$$\tilde{\beta}_N(s) = 1/2^v \quad \forall s \quad (13.3.20)$$

利用 Bayes 规则，从式(13.3.2)可以看出

$$L(u_k) = \ln \frac{p(y_1^N | u_k = 1)}{p(y_1^N | u_k = 0)} + \ln \frac{P(u_k = 1)}{P(u_k = 0)} = \ln \frac{p(y_1^N | u_k = 1)}{p(y_1^N | u_k = 0)} + L'(u_k) \quad (13.3.21)$$

式中， $L'(u_k)$ 是关于 u_k 的先验信息。在以往的译码方案中，通常认为先验等概，因而 $L'(u_k) = 0$ 。而在迭代译码方案中， $L'(u_k)$ 是前一级译码器作为外信息给出的。为了能使迭代继续进行，当前译码器应从式(13.3.21)的第一项中提取出新的外信息并且提供给下一级译码器，作为下一级译码器接收的先验信息。式(13.3.1)可以写为

$$L'(u_k) = \ln \frac{P(u_k = 1)}{P(u_k = 0)} = \ln \frac{P(u_k = 1)}{1 - P(u_k = 1)} \quad (13.3.22)$$

从上式，可得：

$$P(u_k) = A_k \exp[u_k L'(u_k)/2] \quad (13.3.23)$$

式中， $A_k = \frac{1}{1 + \exp[L'(u_k)]}$ ，为常量。

对于 $p(y_k | u_k)$ ，根据 $y_k = (y_k^i, y_k^p)$, $x_k = (x_k^i, x_k^p) = (u_k, x_k^p)$ ，可得

$$\begin{aligned}p(y_k | u_k) &\propto \exp\left[-\frac{(y_k^i - u_k)^2}{2\sigma^2} - \frac{(y_k^p - x_k^p)^2}{2\sigma^2}\right] \\ &= \exp\left[-\frac{y_k^{i^2} + u_k^2 + y_k^{p^2} + x_k^{p^2}}{2\sigma^2}\right] \cdot \exp\left[\frac{u_k y_k^i + x_k^p y_k^p}{\sigma^2}\right] \\ &= B_k \exp\left[\frac{u_k y_k^i + x_k^p y_k^p}{\sigma^2}\right]\end{aligned}$$

结合式(13.3.11)，可得

$$\gamma_k(s', s) \propto A_k B_k \exp[u_k L'(u_k)/2] \cdot \exp\left[\frac{u_k y_k^i + x_k^p y_k^p}{\sigma^2}\right] \quad (13.3.24)$$

若定义 $\gamma_e^k(s', s) = \exp[\frac{1}{2} L_c y_k^p x_k^p]$ ；定义信道可靠性值 $L_c \equiv 4aE_s/N_0$ ；对于 AWGN 信道上的 QPSK 传输， $L_c = 4/N_0$ ；而 $\sigma^2 = N_0/2$ 。于是，式(13.3.24)可以写为

$$\begin{aligned}
\gamma_k(s', s) &\propto \exp\left[\frac{1}{2}u_k(L^e(u_k) + L_c y_k^s) + \frac{1}{2}L_c y_k^e x_k^p\right] \\
&= \exp\left[\frac{1}{2}u_k(L^e(u_k) + L_c y_k^s)\right] \cdot \gamma_k^e(s', s)
\end{aligned} \tag{13.3.25}$$

结合(13.3.17)与式(13.3.25)，可得

$$\begin{aligned}
L(u_k) &= \ln \left(\frac{\sum_{s+} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k^e(s', s) \cdot \tilde{\beta}_k(s) \cdot C_k}{\sum_{s-} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k^e(s', s) \cdot \tilde{\beta}_k(s) \cdot C_k} \right) \\
&= L_c y_k^s + L^e(u_k) + \left[\frac{\sum_{s+} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k^e(s', s) \cdot \tilde{\beta}_k(s)}{\sum_{s-} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k^e(s', s) \cdot \tilde{\beta}_k(s)} \right]
\end{aligned} \tag{13.3.26}$$

式中，第一项叫做信道值；第二项代表的是前一个译码器为第二个译码器所提供的关于 u_k 的先验信息；第三项代表的是可送给后续译码器的外部信息。对于图 13-3 所示的 Turbo 译码器，如果分量码译码器 DEC 1 和 DEC 2 均采用上述 MAP 译码算法，则它们在第 i 次迭代的软输出分别为：

$$\text{DEC 1: } L_1^{(i)}(u_k) = L_c y_k^s + [L_{21}^e(u_k)]^{(i-1)} + [L_{12}^e(u_k)]^{(i)} \tag{13.3.27}$$

$$\text{DEC 2: } L_2^{(i)}(u_{I_k}) = L_c y_{I_k}^s + [L_{12}^e(u_{I_k})]^{(i)} + [L_{21}^e(u_{I_k})]^{(i)} \tag{13.3.28}$$

式中， $L_{21}^e(u_k)$ 是前一次迭代中 DEC 2 给出的外信息 $L_{21}^e(u_{I_k})$ 经解交织后的信息，在本次迭代中被 DEC 1 用作先验信息； $L_{12}^e(u_k)$ 是 DEC 1 新产生的外信息，即式(13.3.26)中的第三项； $L_{12}^e(u_{I_k})$ 为经交织的从 DEC 1 到 DEC 2 的外信息。整个迭代中软信息的转移过程为：

DEC 1 → DEC 2 → DEC 1 → DEC 2 → ...

图 13-8 是 Turbo 码采用标准 MAP 译码算法所得到的性能图(AWGN 信道，3GPP 交织器，迭代次数为 6 次，分量码 $g=(13, 15)_8$)，从图中可以看出不同的交织器大小对码字性能的影响。

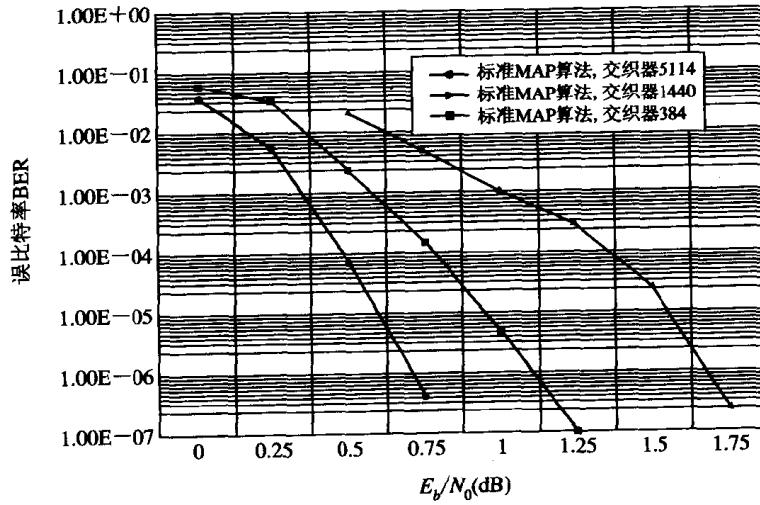


图 13-8 Turbo 码 MAP 译码算法性能曲线

MAP 算法的引入使组成 Turbo 码的两个编码器均可采用性能优异的卷积码，同时采用了反馈译码的结构，实现了软输入/软输出，递推迭代译码，使编译码过程实现了伪随机

化，并简化了最大似然译码算法，使其性能达到了逼近 Shannon 限。但 MAP 算法存在几个难以克服的缺点：

- (1) 需要在接收到整个比特序列后才能做出译码判决,译码延迟很大。
 - (2) 计算时既要有前向迭代又要有后向迭代。
 - (3) 与接收一组序列(交织器大小)呈正比的存储量等。

为了克服 MAP 算法的缺点,一方面根据 MAP 算法进行简化;一方面寻找新的在性能上与 MAP 算法相差不太大的译码算法。常见的译码算法有以下两类。

MAP 算法及其改进算法：分为标准 BCJR 算法；对数域的 LOG - MAP 算法及 MAX - LOG - MAP 算法；减少状态搜索的 M - BCJR 和 T - BCJR 算法；滑动窗 BCJR 算法和只有前向递归的 OSA 算法。它们均可用于 Turbo 码的迭代译码。

另一类是 SOVA 算法及其改进算法，其运算量为标准 Viterbi 算法的两倍，运算量低于 MAP 算法，但其译码增益比 MAP 算法要小。

四、SOVA 译码算法

在工程应用中,由于 SOVA 算法的运算量较小,而且可以采用滑窗法,从而可以大大减小时延,因此更适合于工程应用。下面介绍 SOVA 算法。

SOVA 算法的全称是：软输出 Viterbi 算法(Soft-Output Viterbi Algorithm)。Viterbi 算法(以下简称 VA)在通信接收器中已经成为一个工具，在解调、译码、均衡中都有广泛的应用。在 QAM(正交振幅调制)和 CPM(连续相位调制)等系统中，VA 的级联得到了应用。在这些系统中，Viterbi 接收器代替了传统的调制机制。然而在 Viterbi 级联系统中存在着以下缺点：内 VA 调制会产生突发错误，而外 VA 调制又非常敏感；内 VA 产生硬判决限制了外 VA 运用接收软判决的能力。在实际应用中，这两个缺点越发明显。于是引入了 SOVA 算法，它采用软(硬)判决计算度量值，且输出相应判决比特和可靠性信息。

SOVA 算法是 Hagenauer 于 1989 年提出的。它是 Viterbi 算法的改进类型。Viterbi 算法是一种最大似然译码算法。通俗地来说，它的译码过程是在接收序列 \mathbf{R} 的控制下，在码的篱笆图上走编码器走过的路径。

图 13-9 所示的是(2,1,2)码($G(D) = [1+D+D^2, 1+D^2]$)的篱笆图。它表示了编码器状态转移与时间的关系。VA 就是在篱笆图上找寻一条具有最大“度量”路径。下面我们将来分析 SOVA 算法。

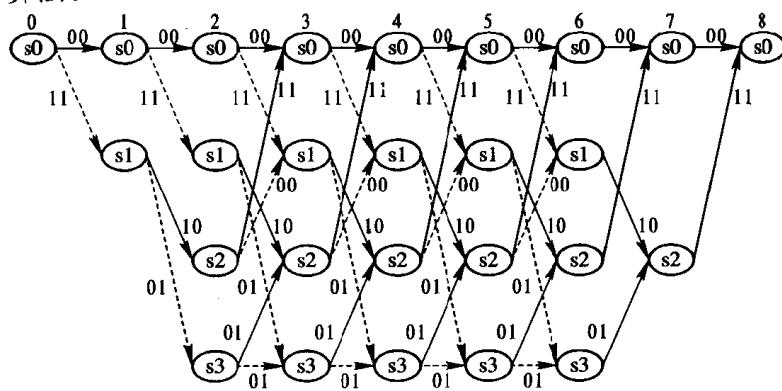


图 13-9 (2,1,2) 码 $L=6$ 时的篱笆图

图 13-10 表示的是一个 SOVA 算法的例子。为了简化起见，我们考虑网格图（篱笆图）上每一个节点有两个分支，状态数为 2^v , v 为编码器寄存器个数。它以 δ 为时延进行一比特判决， δ 足够大，使得 2^v 个幸存路径以足够大的概率汇聚于一点。如图 13-9 所示，在 k 时刻，对于状态 s_k , Viterbi 算法选择一条幸存路径，这是通过计算路径最小距离度量(或最大相关度量)而得到的。同时，状态 s_k 还对应着一条待选路径。对于幸存路径，将其度量标为 M_1 ，相应的，待选路径的度量我们标为 M_2 。于是幸存路径选错的概率为

$$p_{sk} = \frac{e^{-M_2}}{e^{-M_1} + e^{-M_2}} = \frac{1}{1 + e^{M_2 - M_1}} = \frac{1}{1 + e^\Delta} \quad (13.3.29)$$

式中， $\Delta = M_2 - M_1 \geq 0$ 。 p_{sk} 代表的则是传输不可信度。于是在 e 个路径 1(幸存路径)与路径 2(待选路径)的信息比特不等的位置处，其错误概率为 p_{sk} 。我们可以用下式表示：

$$\hat{p}_j \leftarrow \hat{p}_j (1 - p_{sk}) + (1 - \hat{p}_j) p_{sk} \quad j = j_1, \dots, j_e (u_j^{(1)} \neq u_j^{(2)}) \quad (13.3.30)$$

式中， \hat{p}_j 表示的是已存储的路径 1 的错误概率。则对数似然比可以写为

$$\hat{L}_j = \lg \frac{1 - \hat{p}_j}{\hat{p}_j} \quad 0 \leq \hat{L}_j < \infty \quad (13.3.31)$$

结合式(13.3.29)、式(13.3.30)和式(13.3.31)，可以得到

$$\hat{L}_j \leftarrow f(\hat{L}_j, \Delta) = \frac{1}{\alpha} \ln \frac{1 + e^{(\alpha L_j + \Delta)}}{e^\Delta + e^{\alpha L_j}} \quad (13.3.32)$$

式中， α 的引入是为了防止信噪比的增加而产生溢出。 $\alpha = 4d_{\text{free}} E_s / N_0$ 。上式可近似写为

$$f(\hat{L}_j, \Delta) = \min(\hat{L}_j, \Delta/\alpha) \quad (13.3.33)$$

于是 SOVA 算法可以分为以下几个步骤完成：

- (1) 计算路径度量与度量差；
- (2) 更新可靠性度量；
- (3) 减去内信息，得到下一步所需的外信息值。

以上几步完成后，将所得到的外信息值带入下一个 SOVA 译码器中，进行下一步迭代，即可完成 SOVA 算法在 Turbo 码中译码的应用。SOVA 算法存在两种比较重要的形式：一种是 HR-SOVA，另一种是 BR-SOVA。两种的区别在于可靠性度量的更新方法与原则。

$$u_j^i \neq u_j^c \Rightarrow L_j^i = \min(L_j^i, \Delta) \quad (13.3.34)$$

$$\begin{cases} u_j^i \neq u_j^c \Rightarrow L_j^i = \min(L_j^i, \Delta) \\ u_j^i = u_j^c \Rightarrow L_j^i = \min(L_j^i, \Delta + L_j^c) \end{cases} \quad (13.3.35)$$

式(13.3.34)代表的是 HR-SOVA 的置换规则，式(13.3.35)代表的是 BR-SOVA 的置换规则。可以证明，BR-SOVA 与对数域上的 MAP 算法的简化算法 MAX-LOG-MAP 算法是等价的。

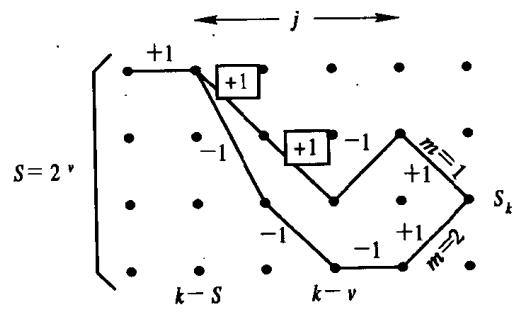


图 13-10 SOVA 算法的一个例子

图 13-11 表示的是 Turbo 码采用 SOVA 译码算法的性能曲线(AWGN 信道, 3GPP 交织器 $N=384$, 迭代次数 6 次, 分量码 $g=(13,15)_8$)。从图中可以看出复杂度减小的代价是性能的降低。

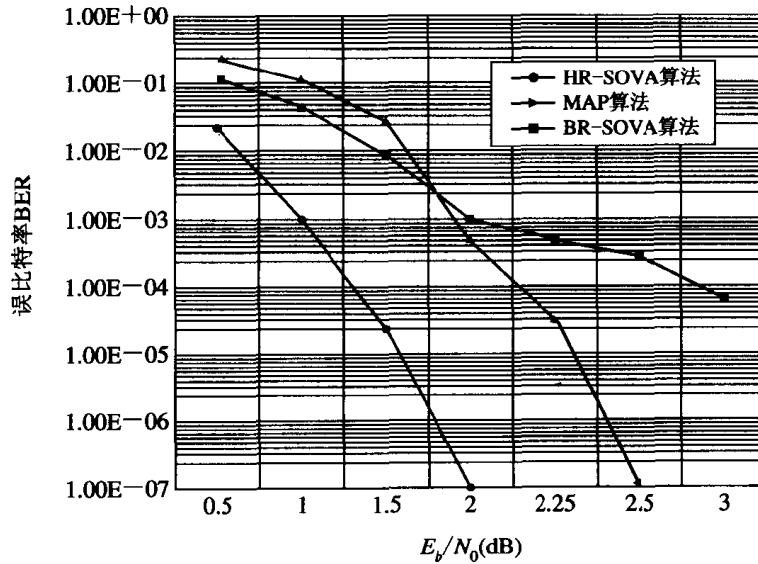


图 13-11 Turbo 码采用 SOVA 译码算法的性能曲线

五、Rayleigh 衰落信道下 Turbo 码的译码

我们详细给出了 AWGN 信道上 Turbo 码的 MAP 译码算法。对于 Rayleigh 慢衰落信道, MAP 译码算法的修正主要是与信道条件有关的分支转移概率 $\gamma_k(s', s)$ 。对于充分交织的衰落信道, 信道可看作是无记忆的。如果接收机已知信道状态信息(CSI) a_k , 则 $\gamma_k(s', s)$ 应修改为

$$\begin{aligned}\gamma_k(s', s) &= p(S_k = s, y_k | S_{k-1} = s', a_k) \\ &= p(S_k = s | S_{k-1} = s') \cdot p(y_k | S_k = s, S_{k-1} = s', a_k) \\ &= P(u_k) \cdot p(y_k^s | u_k, a_k^s) \cdot p(y_k^p | x_k^p, a_k^p)\end{aligned}\quad (13.3.36)$$

式中, $P(u_k)$ 是 u_k 的先验概率, $p(y_k^s | u_k, a_k^s)$ 和 $p(y_k^p | x_k^p, a_k^p)$ 服从高斯概率分布。因此, 有

$$\begin{aligned}&p(y_k^s | u_k, a_k^s) \cdot p(y_k^p | x_k^p, a_k^p) \\ &\propto \exp\left\{-\frac{1}{2\sigma^2}[y_k^s - a_k^s(2x_k^s - 1)]^2\right\} \cdot \exp\left\{-\frac{1}{2\sigma^2}[y_k^p - a_k^p(2x_k^p - 1)]^2\right\} \\ &= B_k \exp\left\{\frac{2(a_k^s y_k^s + a_k^p y_k^p)}{\sigma^2}\right\}\end{aligned}\quad (13.3.37)$$

B_k 为常量。于是

$$\gamma_k(s', s) = K \exp\{u_k L^e(u_k) + L_c a_k^s y_k^s + L_c a_k^p y_k^p\} \quad (13.3.38)$$

对于充分交织的衰落信道, 如果接收机未知信道状态信息, 则分支转移概率 $\gamma_k(s', s)$ 为

$$\gamma_k(s', s) = P(u_k) \cdot p(y_k^s | u_k) \cdot p(y_k^p | x_k^p) \quad (13.3.39)$$

式中, $p(y_k^s|u_k)$ 和 $p(y_k^p|x_k^p)$ 分别是从 $p(y_k^s|u_k, a_k^s)$ 和 $p(y_k^p|x_k^p, a_k^p)$ 在 Rayleigh 衰落幅度 a_k 上取统计平均获得。即

$$\begin{aligned} p(y_k^s|u_k) &= \int_0^\infty p_A(a) p(y_k^s|u_k, a) da \\ &= \int_0^\infty 2ae^{-a^2} \left[\frac{1}{\sqrt{N_0\pi}} e^{-(y_k^s - a(2u_k - 1))^2/N_0} \right] da \end{aligned} \quad (13.3.40)$$

上式计算非常复杂, 一种简化计算方法是: 假定 $p(y_k^s|u_k)$ 是高斯分布, 则有

$$p(y_k^s|u_k) \propto \exp\left(\frac{E_A(a)(2u_k - 1)y_k^s}{N_0}\right) \quad (13.3.41)$$

在一般仿真情况下, 假定 Rayleigh 衰落的平均能量为 1, 于是 $E_A(a)=0.8862$ 。 $p(y_k^p|x_k^p)$ 的计算与此类似。所以, 最后得

$$\gamma_k(s, s) = K \exp(u_k L'(u_k) + L_c E_A(a) y_k^s x_k^s + L_c E_A(a) y_k^p x_k^p) \quad (13.3.42)$$

对于相关的 Rayleigh 衰落信道, 我们可以在 Turbo 编码器之后附加一个信道交织器, 在接收端通过解交织将 Rayleigh 衰落近似为不相关的, 然后即可按照上述充分交织信道的译码算法进行仿真。相关衰落信道的编码器组成框图如图 13-12 所示。

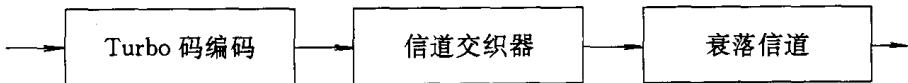


图 13-12 相关衰落信道的信道编码器组成框图

因此, 我们可以统一用式(13.3.38)表示 Turbo 码的 MAP 译码, 并且有

- 未知信道状态信息: $a_k^s = a_k^p = 0.8862$
- AWGN 信道: $a_k^s = a_k^p = 1$

§ 13.4 Turbo 码的分量码、交织器与性能限

本节介绍 Turbo 码中分量码以及交织器对性能的影响, 并讨论了 Turbo 码的平均性能限。

一、分量码

Turbo 码是建立在一种特殊的系统卷积码——递归系统卷积码(RSC)基础之上的, 它以两个 RSC 码作为它的分量码, 因此分量码的选取对 Turbo 码的性能有重要的影响。

1. 分量码的选择

从差错控制编码的有关文献中可知, 非系统卷积码(NSC)的 BER 性能在高信噪比时比约束长度相同的非递归系统码要好, 而在低信噪比时情况却正好相反。递归系统卷积(RSC)码综合了 NSC 码和系统码的特性, 虽然它与 NSC 码具有相同的 trellis 结构和自由距离, 但是在高码率($R_c \geq 2/3$)的情况下, 对任何信噪比, 它的性能均比等效的 NSC 码要好。

因此, 在 Turbo 码中采用 RSC 码作为分量码。一个生成多项式为(37, 21)的十六状态

RSC 编码器结构如图 13-13 所示。

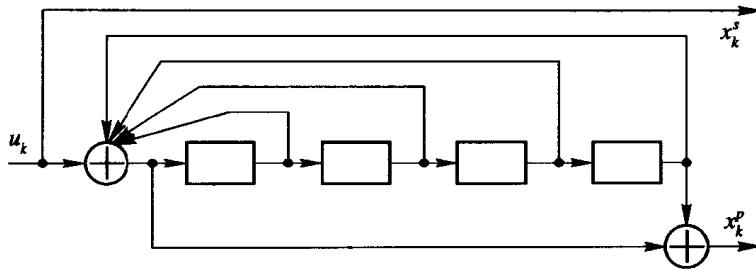


图 13-13 递归系统卷积码编码器 $(g_1, g_2) = (37, 21)$

用 RSC 码构成的 Turbo 码的码率 R 为

$$\frac{1}{R} = \frac{1}{R_1} + \frac{1}{R_2} - 1 \quad (13.4.1)$$

R_1, R_2 为构成 Turbo 码的分量码的码率，在经删除后，分量码 RSC 1 与 RSC 2 的码率 R_1, R_2 可以不同。

表 13-1 给出了十六状态 Turbo 码分量码的两种配置方式。

Turbo 码的最大似然译码性能分析指出，Turbo 码在高信噪比下的性能主要由它的自由距离所决定。因为 Turbo 码的自由距离主要由重量为 2 的输入信息序列所产生的码字间的最小距离所决定，用本原多项式作为反馈连接多项式的分量编码器所产生的码字的最小重量为最大，因此当 Turbo 码交织器的大小给定后，如果分量码的反馈连接多项式采用本原多项式，则 Turbo 码的自由距离会增加，从而 Turbo 码在高信噪比下的“错误平层(error floor)”会降低。错误平层效应指的是在中高信噪比情况下，误码曲线变平。也就是说，即使是再增大信噪比，误码率也降不下来。(一般的系统，比如说是 BPSK 的误码曲线，误码率随着信噪比的增大是单调下降的。)具体结果如图 13-14 所示。

表 13-1 十六状态 Turbo 码的两种配置

| 说 明 | 分量码 RSC 1 $G = (g_0, g_1)_8$ | 分量码 RSC 2 $G = (g_0, g_1)_8$ |
|--------------------|---------------------------------|---------------------------------|
| NP16-NP16(非本原-非本原) | (37, 21) | (37, 21) |
| P16-P16(本原-本原) | (23, 35) | (23, 35) |

在图 13-14 中 Turbo 码的码率为 $1/2$ ，迭代次数为 6 次， N 为交织长度。由图可见，在低信噪比区域，使用非本原反馈多项式的 Turbo 码的性能要优于采用本原多项式的 Turbo 码；而在高信噪比区域，却正好相反，本原 Turbo 码具有很低的错误平层。因此，为了兼顾 Turbo 码在两个区域的性能，一个分量码可以采用本原分量码，另一个为非本原分量码这种非对称编码器结构。因此可以把表 13-1 中分量码 RSC 2 列中上、下两行对调，可以获得较好性能。

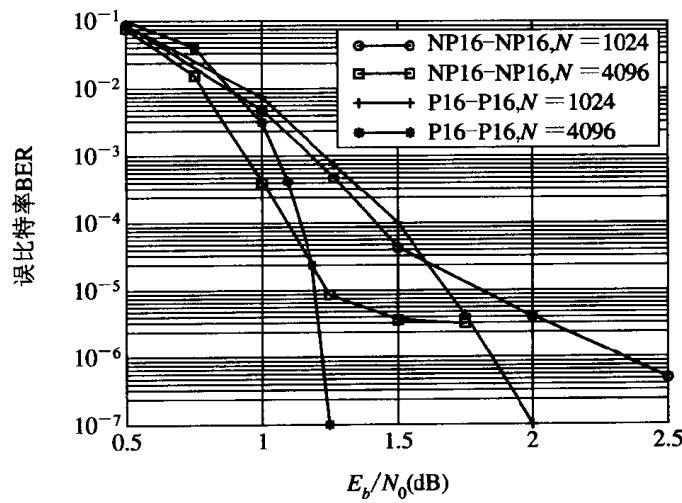


图 13-14 采用本原反馈多项式与非本原反馈多项式的 Turbo 码性能的比较

2. 归零处理对 Turbo 码性能的影响

在 MAP 译码算法中，前向递推 $\alpha_k(s)$ 与反向递推 $\beta_k(s)$ 的初始值一般是根据分量编码器的初始状态和终止状态进行初始化的。对于普通的非系统卷积编码器，很容易将其初始状态和终止状态置为一已知状态，比如零状态。而对于 Turbo 编码器，由于采用了递归结构，其分量编码器需要额外的结尾处理(trellis termination)才能达到终止于零状态，而且由于交织器的存在，将两个编码器同时归零就更为困难。为此，对 $\alpha_k(s)$ 和 $\beta_k(s)$ 的初始化一般采用表 13-2 所示的几种方法。

表 13-2 $\alpha_k(s)$ 和 $\beta_k(s)$ 的初始化方法

| | 译码器 DEC 1 的初始化条件 | 译码器 DEC 2 的初始化条件 |
|-----------------------|--|--|
| RSC 1 和 RSC 2 均不归零 | $\alpha_0(0)=1, \alpha_0(s \neq 0)=0$ $\beta_N(s)=1/M, M=2^v$ | $\alpha_0(0)=1, \alpha_0(s \neq 0)=0$ $\beta_N(s)=1/M, M=2^v$ |
| RSC 1 归零 RSC 2 不归零 | $\alpha_0(0)=1, \alpha_0(s \neq 0)=0$ $\beta_N(0)=1, \beta_N(s \neq 0)=0$ | $\alpha_0(0)=1, \alpha_0(s \neq 0)=0$ $\beta_N(s)=1/M, M=2^v$ |
| RSC 1 和 RSC 2 均归零 | $\alpha_0(0)=1, \alpha_0(s \neq 0)=0$ $\beta_N(0)=1, \beta_N(s \neq 0)=0$ | $\alpha_0(0)=1, \alpha_0(s \neq 0)=0$ $\beta_N(0)=1, \beta_N(s \neq 0)=0$ |

编码器的归零处理对 Turbo 码性能的影响如图 13-15 所示，仿真中 Turbo 码的码率为 1/2，分量码为十六状态，生成多项式 $(g_1, g_2) = (37, 21)$ ，迭代次数为 6 次，使用 BCJR 译码算法。由图可见，当交织器的长度小于 4 096 比特时，分量编码器终了状态的归零处理对 Turbo 码的性能略有改善，并且随着交织长度的增加，这种改善逐渐减小。当交织长度大于 4 096 比特时，分量码的归零处理所带来的性能改善已可以忽略。由此我们得出，当交织长度大于 4 096 比特时，Turbo 码不必进行归零，这样也有利于码率的匹配。

一种将两个分量编码器均归零的解决方案如图 13-16 所示。

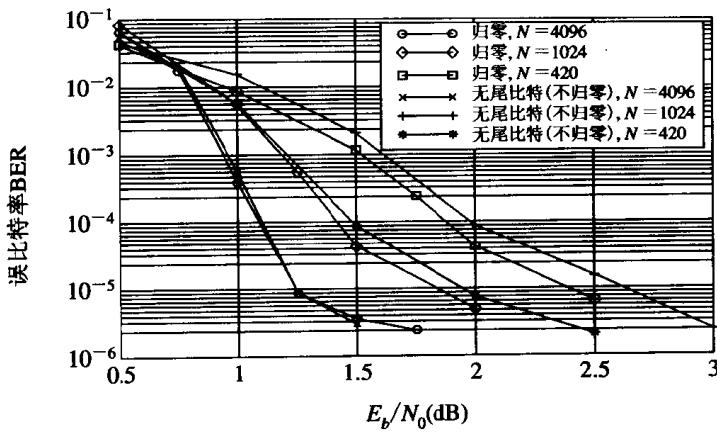


图 13-15 归零处理对 Turbo 码性能的影响

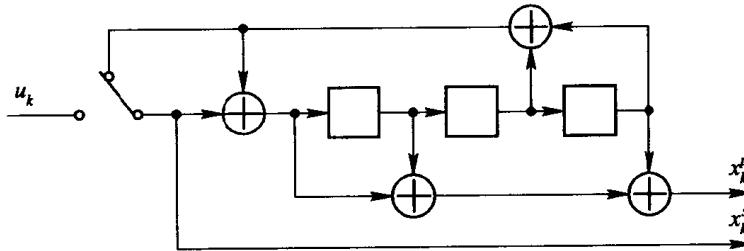


图 13-16 RSC 编码器的一种归零方法 $G=(13,15)$

二、交织器

在 Turbo 码的生成中，交织器扮演着重要的角色。交织器虽然仅仅是在 RSC 2 编码器之前将信息序列中的 N 个比特的位置进行随机置换，但它却起着关键的作用，在很大程度上影响着 Turbo 码的性能。通过随机交织，使得编码序列在长为 $2N$ 或 $3N$ (不使用删余) 比特的范围内具有记忆性，从而由简单的短码得到了近似长码。当交织器充分大时，Turbo 码就具有近似于随机长码的特性。所以交织器的设计是 Turbo 码设计中的一个重要方面。不同交织器对 Turbo 码性能有着不同的影响。

1. 交织器的描述方法

交织器是一个单输入单输出设备，它的输入与输出符号序列有相同的字符集，只是各符号在输入与输出序列中的排列顺序不同。即它是整数 Z 上的置换：

$$\pi: Z \rightarrow Z \quad (13.4.2)$$

如果交织器在第 i 时刻的输出为 $\pi(i)$ ，对于周期为 T 的交织器，它满足下列方程：

$$\pi(i) - T = \pi(i - T) \quad \forall i \quad (13.4.3)$$

并且可以采用下列集合描述：

$$\begin{pmatrix} 0 & 1 & \cdots & T-1 \\ \pi_0 & \pi_1 & \cdots & \pi_{T-1} \end{pmatrix} \quad (13.4.4)$$

目前，Turbo 码交织器有多种设计方法和具体实现形式，常用的有伪随机 S 交织器、

分组交织器(Block Interleaver)和 BG 非均匀交织器等。其中，分组交织器的一种结构如图 13-17 所示。

由图可见，经过这种交织器的置换，信息序列中的首尾比特位置在交织前后保持不变。当分量编码器不归零时，从 MAP 或 SOVA 等算法的译码原理可知，分量译码器对一帧数据中的最后几比特译码的可信度较低，这样如果原帧数据中的最后几比特交织后仍处于帧数据的尾部，则整个 Turbo 码性能的提高就受到了限制。我们称此为尾效应(Tail Effect)。为了避免这个问题，在交织器的设计中应该将原帧数据中的最后几比特置换到 RSC 2 的输入序列的前部(非尾部位置)。当分量编码器均归零时，则不存在尾效应。

关于伪随机交织器，目前所公认的性能较好的是伪随机 s 交织器，即每一个随机产生的置换位置 $\pi(i)$ 均与它前边的 s 个值： $\pi(i-1), \pi(i-2), \dots, \pi(i-s)$ 进行比较。如果距离 $|\pi(i)-\pi(i-j)| < s, j=1, 2, \dots, s$ ，则 $\pi(i)$ 被拒绝，必须重新产生。在本部分中的仿真数据除特别说明外，均采用的是 s 交织器。

不同交织器长度时的 s 交织器与分组交织器比较结果如图 13-18 和图 13-19 所示。其中，Turbo 码的码率为 $1/2$ ，两个分量编码器均归零。

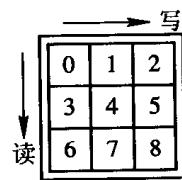


图 13-17 行写列读的分组交织器

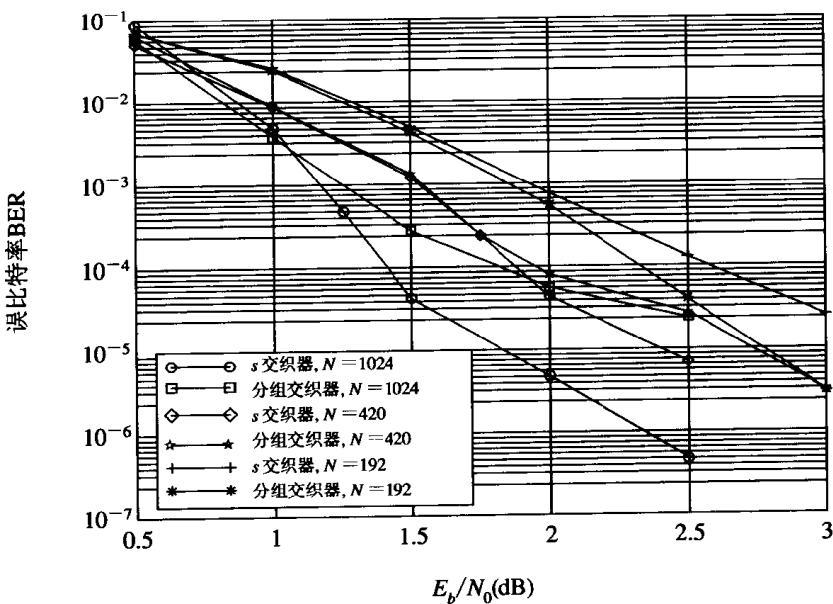


图 13-18 分组交织器与随机交织器的性能比较(十六状态 NP16 - NP16)

从图 13-18 可以看出，对于十六状态的 Turbo 码，当交织长度大于 420 比特时， s 随机交织器的性能要好于同样大小的分组交织器；而当交织长度小于 192 比特时，分组交织器的性能要好于 s 交织器。同时图 13-19 表明，同样的交织长度 192 比特，对于四状态的 Turbo 码， s 随机交织器的性能仍优于分组交织器，直至交织长度低至 80 比特时，分组交

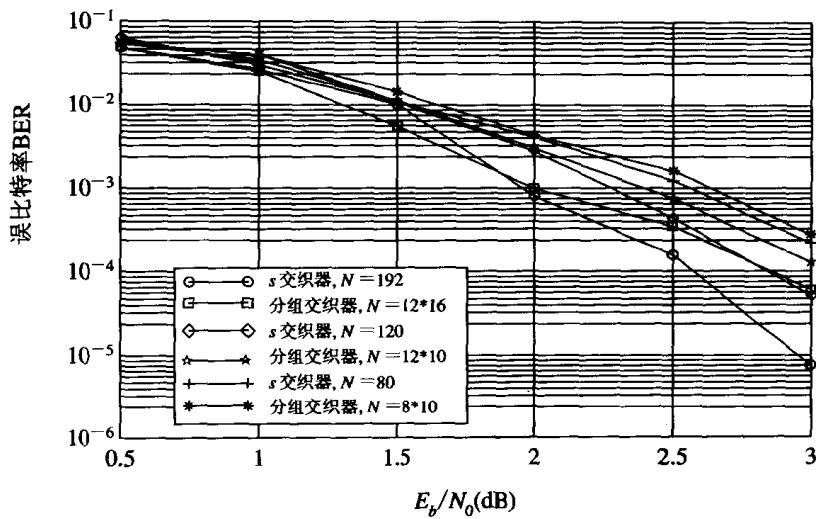


图 13-19 分组交织器与随机交织器的性能比较(四状态 P4 - P4)

织与随机交织的性能才基本一致。由此可以看出， s 交织器与分组交织器性能的差异与分量码有关，即其性能分界点取决于分量码的约束长度。为什么分组交织器与随机交织器的性能会有差别？这可以从下面的“有效码字长度”的概念加以说明。

2. 有效码长的概念

在传统的信道编码中，所使用的交织器一直是分组交织器或卷积交织器，其目的主要是抗信道突发错误，即将信道或级联码内码译码器产生的突发错误随机化，把由于受到噪声干扰而导致具有相关性的数据恢复成相互独立的输入数据。而在 Turbo 码中，目前认为两个 RSC 编码器之间的交织器除了具有上述功能外，还具有一个十分重要的核心作用——改变码的重量分布，使得 Turbo 码的编码输出序列中，重量很轻的码字和重量很重的码字都尽可能少，即使“重量谱窄化(thin)”，从而控制编码序列的距离特性，使 Turbo 码的整体纠错性能达到用户所要求的误码率。例如，假定在图 13-2 中，输入信息序列 u 导致 RSC 1 编码器输出轻重量的码字，那么由于交织器的存在，将 RSC 2 编码器的输入序列中的比特分布模式适当改变，就避免了在 RSC 2 编码器的输出端再产生轻重量的码字，从而提高了 Turbo 码的整体性能。由此可见，交织器的引入改变了输入序列的距离特性。以上这些都是基于传统的码字之间的最小汉明距离来分析 Turbo 码的性能与交织器的作用的，这在一定程度上可以解释 Turbo 码的性能，但我们认为这并没有充分反映交织器的作用。在 Turbo 码中，码字之间的最小汉明距离究竟起到多少作用还需要进一步研究。

还存在一种如下的逆序交织器： $\pi(i) = T - i$ ，即

$$\begin{pmatrix} 0 & 1 & \cdots & T-2 & T-1 \\ T-1 & T-2 & \cdots & 1 & 0 \end{pmatrix} \quad (13.4.5)$$

从上述讨论的改变码的重量分布来看，这个交织器应该也起到了一定的改变重量分布的作用，Turbo 的性能应该不错，但是仿真结果表明，使用该交织器的 Turbo 码性能并不比不交织的 Turbo 码(即单一分量码)的性能好。为此，让我们从另外一个角度来解释交织器。如果分量码的约束长度为 K ，则从卷积码的编码理论与 MAP 译码原理来看，对每一信息比特而言，它与其前边约 L (大于 6 k)个比特和后边约 L 个比特的相关性较大，这一

窗口内的接收数据对当前信息比特的译码起着主要作用，并且随着与当前译码比特相距位置的增大其作用逐渐减弱，这也是滑窗 MAP 译码算法的基础。对于简单的逆序交织器，与当前信息比特 i 最相关的数据在两个分量译码器的译码窗口内的排列次序都相同，因此其效果等同于一个分量编码器。

对于普通分组交织器，参考图 13-20(a)，在网格图上译码时，在第一个分量码的译码器中，对当前译码比特 i 有较大影响的码字比特集中在横坐标上大小为 $2L$ 的窗口内；在第二个分量码的译码器中，对当前译码比特 i 有较大影响的码字比特集中在纵坐标上大小为 $2L$ 的窗口内。这样通过迭代译码，比特 i 的可信度就与周围 $(2L \times 2L) = 4L^2$ 个比特有关。而这 $4L^2$ 个比特中的每一比特又都与一个相应的 $2L \times 2L$ 译码窗口相联系。对于分组交织器，这些窗口大部分重叠在一起，虽然整个 Turbo 码的码字很长，但是对当前译码比特有贡献的码字比特数却限制在 $(4L^2 + L \times 2L + L \times 2L) = 8L^2$ 内（参见图 13-20(b)）。而在随机交织器中，通过随机置换，比特 i 的 $2L \times 2L$ 译码窗口内可能包含距离比特 i 较远的信息比特（如第 8000 个位置），这样与每一比特相对应的译码窗口可能不会完全重叠，从而与当前译码比特有紧密关系的码字比特数就会远大于 $8L^2$ 。交织器越大，这种效果越明显。所以当交织阵较大时，随机交织器要优于分组交织器。

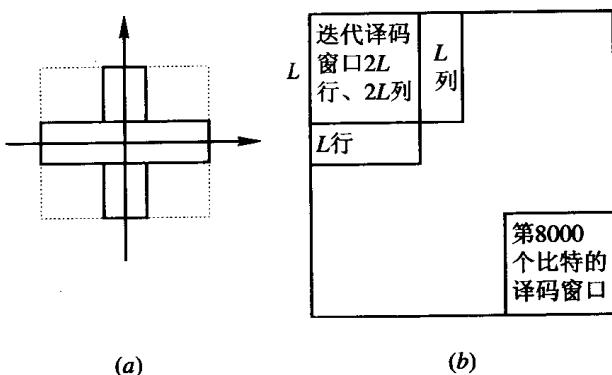


图 13-20 与比特 i 相关的信息位数

基于上述分析，在 Turbo 码中，**交织器的作用主要是使信息比特之间尽可能具有长的关联，从而构成长码**。考虑到最小汉明距离的概念诞生于硬判决译码，而采用最小欧氏距离是基于软判决译码算法的所用度量，所以现在有一个问题是基于 MAP 译码算法，如何定义一种适合于 Turbo 码的有效距离或长度。这在目前还未找到很好的解决方法。这里我们初步定义在长为 N 的信息序列中，采用迭代译码时，与每一译码比特有密切联系的不同信息比特数目为“有效码字长度”。

3. 交织器设计准则

综上所述，提出以下交织器设计准则：

- 最大程度地置乱原数据排列顺序，避免置换前相距较近的数据在置换后仍相距较近，特别要避免置换前相邻数据在置换后再次相邻。
- 尽可能避免与同一信息位直接相关的两个分量编码器中的校验位均被删除。
- 对于不归零的编码器，交织器设计时要避免出现“尾效应”图案。

- 在满足上述要求的交织器中再选择一个好的交织器，使码字之间的最小距离(或自由距离) d_{\min} 尽可能大，而重量为 d_{\min} 的码字数要尽可能少，以改善 Turbo 码在高信噪比时的性能。

此外，从图 13-2 可以看出，由于交织器的存在，Turbo 编码是面向帧数据的，即以帧或块(Block)为单位进行编译码。因此，在设计交织器时，应考虑具体应用系统的数据帧的大小，使交织深度在满足时延要求的前提下，与数据帧大小相一致，或是数据帧长度的整数倍。

4. 删余器

删余的目的是为了得到一定码率的码字。Shannon 在他的信道编码定理中很明确地指出了码率与信道容量的问题，我们的目的是在于尽量提高码率而达到小错误概率的传输。一般情况下，删余之后 Turbo 码的码率为 $1/2, 1/3$ 。相对应的删余阵为 $P = [1 \ 0, 0 \ 1]$ 和 $P = [1, 1]$ 。

三、Turbo 码的平均性能限(界)

1. 联合限(界)

对一种编码方案进行性能分析、评价，一般有两种方法：计算机仿真与标准联合限(界)技术。在低信噪比情况下，由于误码率较大，计算机仿真所需时间较短，而且在信噪比低于信道截止速率 R_0 时标准联合限(Union Bound)发散，因此通常采用计算机仿真的方法。而在高信噪比时，计算机仿真误差较大，一般采用联合限(界)技术来得到码的平均性能上限(界)。对于 Turbo 码的性能分析，也遵循这两种方法。

联合限技术就是基于概率论中 $P(\cup E_i) \leq \sum_i P(E_i)$ 这一论断，来得到采用最大似然(ML)译码时的平均译码错误概率上限(界)：

$$P_e \leq \sum_{d=d_{\min}}^N A_d P_2(d) \quad (13.4.6)$$

式中， A_d 为与码字集合中距离为 d 的码字数，因为是线性码，所以 A_d 也就是汉明重量为 d 的码字总数； $P_2(d)$ 是两个码字之间的错误概率，即成对错误概率。

从式(13.4.6)可见，为了得到 Turbo 码的性能限(界)，首先需要计算出所考虑信道下码字的成对错误概率，与 Turbo 码的重量谱。

2. 成对错误概率(Pairwise Error Probability)

为了得到衰落信道上的成对错误概率，可以考虑二进制编码的 M 元信号传输问题。假定采用 BPSK 调制，则相应于第 i 个 n 维码矢 $c_i = [c_{i1} \ c_{i2} \cdots \ c_{in}]$ 的发送信号可以等效为一个 n 维矢量 $s_i = [s_{i1} \ s_{i2} \cdots \ s_{in}]$ ， $i=1, 2, \dots, M$ 。其中：

$$s_{ik} = \begin{cases} \sqrt{E_s} & \text{当 } c_{ik} = 1 \\ -\sqrt{E_s} & \text{当 } c_{ik} = 0 \end{cases} \quad (13.4.7)$$

从通信理论可知，对于 AWGN 信道，当发送信号为 s_i ，而接收机却判决为 s_j 的判决错误概率为

$$P_2(s_i \rightarrow s_j) = Q\left(\sqrt{\frac{d_{ij}^2}{2N_0}}\right) \quad (13.4.8)$$

式中：

$$d_{ij}^2 = |\mathbf{s}_i - \mathbf{s}_j|^2 \quad (13.4.9)$$

如果 \mathbf{s}_i 和 \mathbf{s}_j 有 d 个分量不同(即 $d_H(\mathbf{c}_i, \mathbf{c}_j) = d$)，则有

$$d_{ij}^2 = \sum_{k=1}^n (s_{ik} - s_{jk})^2 = d(2\sqrt{E_s})^2 = 4dE_s \quad (13.4.10)$$

将上式代入(13.4.8)，得

$$P_2(s_i \rightarrow s_j) = Q\left(\sqrt{\frac{2dE_s}{N_0}}\right) \quad (13.4.11)$$

对于充分交织的 Rayleigh 衰落信道，如果接收机具有信道状态信息，则有

$$d_{ij}^2 = \sum_{k=1}^n [a_{ik}(s_{ik} - s_{jk})]^2 = 4E_s \sum_{k=1}^d a_k^2 \quad (13.4.12)$$

于是我们得到条件成对错误概率

$$P_2(s_i \rightarrow s_j | \mathbf{a}) = Q\left(\sqrt{\frac{2E_s}{N_0} \sum_{k=1}^d a_k^2}\right) \quad (13.4.13)$$

在 \mathbf{a} 上取集合平均得

$$\begin{aligned} P_2(s_i \rightarrow s_j) &= E_a \{P_2(s_i \rightarrow s_j | \mathbf{a})\} \\ &= \int_{a_1}^{\infty} \cdots \int_{a_d}^{\infty} p(a_1, a_2, \dots, a_d) Q\left(\sqrt{\frac{2E_s}{N_0} \sum_{k=1}^d a_k^2}\right) da_1 \cdots da_d \end{aligned} \quad (13.4.14)$$

并且有

$$p(a_1, a_2, \dots, a_d) = \prod_{k=1}^d p(a_k) \quad (13.4.15)$$

为方便起见，我们以后记为 $P_2(s_i \rightarrow s_j)$ 为 $P_2(d)$ 。

Q 函数一般没有闭形式的解，但可以利用它的一些性质来简化 $P_2(d)$ 的计算或得到一些上限(界)。

从性质 1：

$$Q(x) = \frac{1}{\pi} \int_0^{\pi/2} e^{-x^2/(2\sin^2\theta)} d\theta \quad (13.4.16)$$

得到：

$$\text{AWGN 信道: } P_2(d) = \frac{1}{\pi} \int_0^{\pi/2} e^{-(2dE_s/N_0)/(2\sin^2\theta)} d\theta \quad (13.4.17)$$

$$\text{衰落信道: } P_2(d) = \frac{1}{\pi} \int_0^{\pi/2} \left(\frac{\sin^2\theta}{E_s/N_0 + \sin^2\theta} \right)^d d\theta \quad (13.4.18)$$

从性质 2：

$$Q(x) \leq \frac{1}{2} e^{-x^2/2} \quad x \geq 0 \quad (13.4.19)$$

得到：

$$\text{AWGN 信道: } P_2(d) \leq \frac{1}{2} e^{-dE_s/N_0} \quad (13.4.20)$$

$$\text{衰落信道: } Q\left(\sqrt{\frac{2E_s}{N_0} \sum_{k=1}^d a_k^2}\right) \leq \frac{1}{2} e^{-(E_s/N_0) \sum_{k=1}^d a_k^2}$$

$$P_2(d) \leq \frac{1}{2} \left(1 + \frac{E_s}{N_0} \right)^{-d} \quad (13.4.21)$$

3. 重量枚举函数

为了得到 Turbo 码的重量分布，必须求出分量码的重量分布。从纠错码理论知道，这可以通过重量枚举函数来得到。因为一个卷积码归零后可以等效于一个分组码，因此下面以分组码为例来定义重量枚举函数。

对一个 $[N, K]$ 线性分组码 C ，它的重量枚举函数(WEF)定义为

$$A(X) = \sum_{i=0}^N A_i X^i \quad (13.4.22)$$

式中， A_i 是码字集合中汉明重量为 i 的码字数。

对于系统编码器，定义它的输入冗余重量枚举函数(IRWEF)为

$$A(W, Z) = \sum_{w=0}^K \sum_{z=0}^{N-K} A_{w,z} W^w Z^z \quad (13.4.23)$$

式中， $A_{w,z}$ 是码字集合中信息位重量为 w 、校验位重量为 z 的码字数。显然，有下列等式成立：

$$A_i = \sum_{w+z=i} A_{w,z} = \sum_{w=0}^i A_{w,i-w} \quad (13.4.24)$$

为分析方便起见，根据信息位的重量 w 将码字分成 K 组，并对每组定义条件重量枚举函数(CIRWEF)为：

$$A_w(Z) = \sum_{z=0}^{N-K} A_{w,z} Z^z \quad (13.4.25)$$

从式(13.4.23)和式(13.4.25)可以看出，输入冗余重量枚举函数和条件重量枚举函数有下述关系：

$$A(W, Z) = \sum_{w=0}^K W^w A_w(Z) \quad (13.4.26)$$

$$A_w(Z) = \frac{1}{w!} \cdot \left. \frac{\partial^w A(W, Z)}{\partial W^w} \right|_{W=0} \quad (13.4.27)$$

例 13.2 (7, 4) 汉明码的重量枚举函数。一个(7, 4)系统汉明码的生成矩阵如下：

$$G(D) = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \quad (13.4.28)$$

其重量枚举函数为

$$A(X) = 1 + 7X^3 + 7X^4 + X^7 \quad (13.4.29)$$

输入输出重量枚举函数为

$$A(W, X) = 1 + 3WX^3 + WX^4 + 3W^2X^3 + 3W^2X^4 + W^3X^3 + 3W^3X^4 + W^4X^7 \quad (13.4.30)$$

将码字重量分为信息位重量和校验位重量，可获得其输入冗余重量枚举函数

$$A(W, Z) = 1 + W(3Z^2 + Z^3) + W^2(3Z + 3Z^2) + W^3(1 + 3Z) + W^4Z^3 \quad (13.4.31)$$

条件重量枚举函数为：

$$\begin{aligned}
A_0(Z) &= 1 \\
A_1(Z) &= 3Z^2 + Z^3 \\
A_2(Z) &= 3Z + 3Z^2 \\
A_3(Z) &= 1 + 3Z \\
A_4(Z) &= Z^3
\end{aligned}$$

在此基础上，我们需要得到两个分量码级联后的重量枚举函数，以便对 Turbo 码进行性能分析。对于一个特定的交织器，这很困难，因为第二个分量码的校验位不仅与 Turbo 码编码器的输入信息序列有关，而且还依赖于交织阵。但是我们可以在所有交织器上取一个平均性能。为此，S. Benedetto 等引入了均匀交织器的概念，其定义如下：

一个长度为 K 的均匀交织器是一种概率设备(Probabilistic Device)，它将重量为 w 的输入序列等概地置换为 $\binom{K}{w}$ 种不同输出序列之一。

由此出发，便容易得到 Turbo 码的重量枚举函数。令 $A_w^{C_1}(Z)$ 和 $A_w^{C_2}(Z)$ 分别是分量码编码器 RSC 1 和 RSC 2 的条件重量枚举函数，则 Turbo 码整体的条件重量枚举函数为

$$A_w^{C_P}(Z) = \frac{A_w^{C_1}(Z) \cdot A_w^{C_2}(Z)}{\binom{K}{w}} \quad (13.4.32)$$

现在，就可以通过联合界技术求出 Turbo 码的平均性能上界。

4. 联合界(Union Bound)

考虑一个 $[N, K]$ 线性码的 ML 译码的平均译码错误概率上界。应用联合界技术，得到在发送特定码字 c 时的译码错误概率为

$$P(e|c) \leq \sum_{c' \neq c} P_2(c \rightarrow c') = \sum_{d=d_{\min}}^N A_{d|c} P_2(d) \quad (13.4.33)$$

式中， $A_{d|c}$ 是与码字 c 的距离为 d 的码字数， d_{\min} 为码字的最小汉明距离；对于卷积码，有 $d_{\min} = d_f$ 。因此，平均码字错误概率为

$$P_e = P(e) = \sum_c P(c) P(e|c) \leq \sum_{d=d_{\min}}^N A_d P_2(d) \quad (13.4.34)$$

式中， A_d 是码字集合中距离为 d 的码字数。因为是线性码，所以 A_d 也就是汉明重量为 d 的码字总数。

对于 $[N, K]$ 系统码，设码的信息位重量为 w ，校验位重量为 z ，则平均比特错误概率为

$$P_b \leq \sum_{w=1}^N \frac{\delta_w}{K} A_w P_2(w) = \sum_{w=1}^K \frac{w}{K} \sum_{z=0}^{N-K} A_{w,z} P_2(w+z) \quad (13.4.35)$$

式中， δ_w 是重量为 w 的码字中信息位的平均重量，并有 $d=w+z$ 。

由式(13.4.32)和式(13.4.35)即可得到 Turbo 码在所有交织器上的平均性能限(界)。

对于 AWGN 信道，将式(13.4.20)代入(13.4.35)，得

$$P_b \leq \frac{1}{2} \sum_{w=1}^K \frac{w}{K} e^{-wE_s/N_0} \sum_{z=0}^{N-K} A_{w,z} e^{-zE_s/N_0} = \frac{1}{2} \sum_{w=1}^K \frac{w}{K} W^w A_w(Z) \Big|_{W=Z=e^{-E_s/N_0}} \quad (13.4.36)$$

对于 Rayleigh 衰落信道, 将式(13.4.21)代入式(13.4.35), 得

$$\begin{aligned} P_b &\leq \frac{1}{2} \sum_{w=1}^K \frac{w}{K} (1 + E_s/N_0)^{-w} \sum_{z=0}^{N-K} A_{w,z} (1 + E_s/N_0)^{-z} \\ &= \frac{1}{2} \sum_{w=1}^K \frac{w}{K} W^w A_w(Z) \Big|_{W=z=1+E_s/N_0} \end{aligned} \quad (13.4.37)$$

5. 性能仿真

例 13.3 Turbo 码平均译码错误概率上限(界):

一个 $1/3$ 码率, 分量码生成矩阵为 $G(D) = \left[1, \frac{1+D^2}{1+D+D^2} \right]$, 交织器大小为 500, 其平均译码错误概率上限(界)结果如图 13-21 所示。

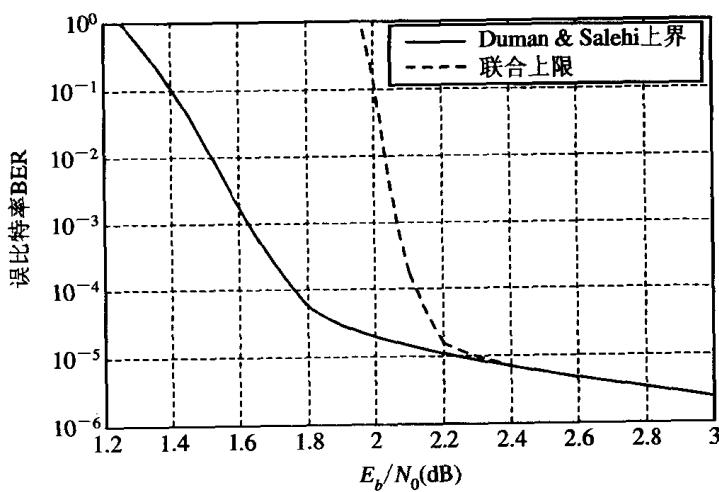


图 13-21 Turbo 码的平均译码错误概率上限(界)^①

§ 13.5 Turbo 码在实际通信系统(3GPP)中的应用

Turbo 码就目前而言, 从出现到发展已经有了 7 年多的历程。许多科学家都在研究 Turbo 码的理论依据, 取得了不少成果。Turbo 码已经有了很大的发展。在各方面也都走向了实际应用阶段。在 Turbo 码的应用研究中, Turbo 码已被美国空间数据系统顾问委员会作为深空通信的标准, 同时它也被确定为第三代移动通信系统(IMT-2000)的信道编码方案之一。其中, 具有代表性的 3GPP 的 WCDMA、CDMA2000 和我国的 TD-SCDMA 三个标准中的信道编码方案都使用了 Turbo 码, 用于高速率、高质量的通信业务, 第三代移动通信标准的实施为 Turbo 码的研究提供了重要的应用背景; 与 Turbo 码相结合的 TCM 技术也在实际中有了很大的应用。同时, 迭代译码的思想已作为“Turbo 原理”而广泛用于编码、调制、信号检测等领域。

第三代移动通信系统(也称 IMT-2000)是使用 2 000 MHz 左右频段、提供业务速率高达 2 000 kb/s、计划在 2000 年左右试运行的全球移动通信系统, 与第二代系统相比应有下

^① Duman & Salehi 上界是由 T. Duman 和 M. Salehi 所求出的, 其具体情况可参考文献[10]。

面特点：

- 系统的国际性，提供全球无缝覆盖和漫游，世界范围设计的高度一致性。
- 业务的多样性，提供话音、数据和多媒体业务，车载通信速率为 144 kb/s，步行通信速率为 384 kb/s，室内通信速率为 2 Mb/s。
- 高质量的业务，满足通信质量能达到与固定网相比拟的高质量业务要求。
- 高度的灵活性，按需分配带宽，支持大范围、可变速率的信息传送。
- 频谱利用率高、通信容量大。
- 袖珍、多频、多模、通用移动终端。
- 满足通信个人化的要求。
- 系统初始配置能充分利用第二代系统设备和设施，随后实现平滑升级。
- 低的费用，包括设备和服务两方面的费用较低。

在第三代移动通信标准(3GPP)中，传输信道主要采用了三种信道编码方式：无信道编码，卷积码和 Turbo 码。其具体应用范围及参数见表 13-3。

表 13-3 3GPP 中信道编码参数

| 信道类型 | 编码方式 | 码率 |
|--------------|---------|---------------|
| 广播信道(BCH) | 卷积码 | 1/2 |
| 无线寻呼信道(PCH) | | |
| 前向接入信道(FACH) | | |
| 随机接入信道(RACH) | | |
| 专用信道(DCH) | | 1/3, 1/2 或无编码 |
| 专用信道(DCH) | Turbo 码 | 1/3 或无编码 |

在 3GPP 系统中，对于 BER 要求在 10^{-3} 至 10^{-6} 的接收系统，采用 Turbo 码。其具体分量码为八状态反馈卷积码。传输函数为：

$$G(D) = \left[1, \frac{n(D)}{d(D)} \right]$$

式中：

$$d(D) = 1 + D^2 + D^3$$

$$n(D) = 1 + D + D^3$$

其编码器结构如图 13-22 所示。其中移位寄存器的初始值为 0。对于码率为 1/3 的 Turbo 码，不存在多余部分，其输出序列为 $X(0), Y(0), Y'(0), X(1), Y(1), Y'(1), \dots$ 。

Turbo 码的归零：当一组信息位(交织器大小)经过编码器编码完后，编码器通过移位寄存器的反馈信息进行网格图归零。前三个比特用于第一个分量码编码器的归零，而第二个分量码编码器未被归零，后三个比特用于第二个分量码编码器的归零，则所传输的编码后归零信息为：

$$\begin{aligned} & X(t) \ Y(t) \ X(t+1) \ Y(t+1) \ X(t+2) \ Y(t+2) \\ & X'(t) \ Y'(t) \ X'(t+1) \ Y'(t+1) \ X'(t+2) \ Y'(t+2) \end{aligned}$$

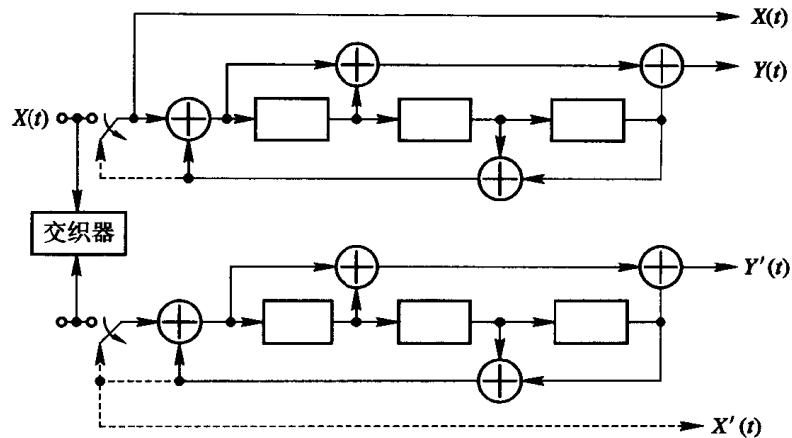


图 13-22 八状态 Turbo 码编码器结构(虚线仅在归零过程有效)

3G 中 Turbo 码交织器：图 13-23 表示的是 3GPP 中八状态 Turbo 码编码器，图中很清晰地描述出了 Turbo 码编码器中内交织器的组成。内交织器包括两个部分：先期交织器(Mother Interleaver)和修剪器(Pruning)。对于任意给定的序列长度 K ，先在 134 个先期交织器中选择出一个合适大小的交织器($(K+l)$ bit)生成，经过交织后，修剪掉 l 比特。这就是 3GPP 中 Turbo 码内交织器的组成。

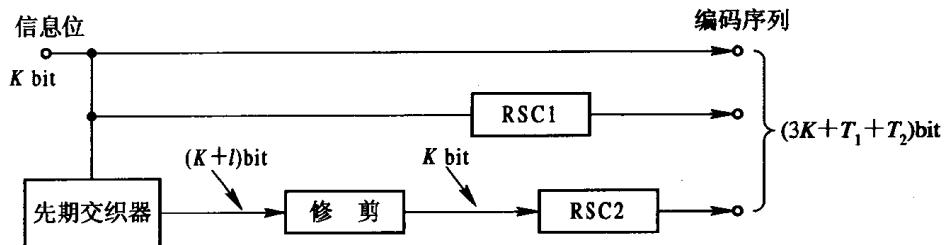


图 13-23 八状态 Turbo 编码器

先期交织器的生成(3G 交织器)：经过仿真结果证明，Turbo 码编码所用的 3G 交织器是非常适合移动通信 Turbo 码编码的交织器。由于移动通信的特殊性，传输信息速率一定时，接收端对每一帧都必须已知该帧在发送端的交织模式；否则，将给移动信道带来附加传输要求，对每帧都要求传送该帧的交织模式，以便接收端能正常进行译码。基于这样的条件，移动通信系统中不能使用每帧交织模式变化的随机交织器，而必须采用交织模式固定的分组交织器。(但普通的分组交织器是无法达到移动通信系统所要求的性能的。)3G 交织也正是同时考虑交织模式固定和交织位置随机而设计出来的。在输入序列长度处于 320 到 5114 比特的情况下，其设计算法中的符号规定如下：

- x_k 输入信息比特；
- K 交织器的交织长度；
- R 交织矩阵的行数；
- C 交织矩阵的列数；
- P 一个质数；

- v 由查表得到的一个基本数；
 $S(i)$ 交织矩阵行内交织的基本序列；
 q_i 一个符合条件的最小质整数；
 r_j 用于置换的质整数；
 $T(j)$ 行间交织模式序列；
 $U_{,}(i)$ 行内交织模式序列；
 i 交织矩阵的行数；
 j 交织矩阵的列数；
 k 输入序列的序号。

设计算法如下：

(1) 由输入信息序列长度来决定交织矩阵的行数 R

$$R = \begin{cases} 10 & 481 \leq K \leq 530 \\ 20 & K < 481 \text{ 或 } K > 530 \end{cases}$$

(2) 选择交织矩阵的列数 C

如果 $481 \leq K \leq 530$, 那么矩阵的列 $C=p$ 且 $p=53$ 。否则, 就得先找到满足下面条件的最小质数 p :

$$p + 1 - \frac{K}{p} \geq 0$$

找到 p 后, 然后来决定交织矩阵的列数 C :

如果 $p - \frac{K}{R} \geq 0$ 且 $p - 1 - \frac{K}{R} \geq 0$, 那么 $C = p - 1$;

如果 $p - \frac{K}{R} \geq 0$ 且 $p - 1 - \frac{K}{R} \leq 0$, 那么 $C = p$;

如果 $p - \frac{K}{R} \leq 0$, 那么 $C = p + 1$ 。

(3) 将输入序列 x_k 按行写入 $R \times C$ 的交织矩阵, 在信息比特写入交织矩阵之后, 按下面的算法将交织矩阵进行行内和行间置换:

① 根据查表得到本原根 v ;

② 构造行内置换序列 $s(i)$:

$$s(i) = [v \times s(i-1)] \bmod p \quad i = 1, 2, \dots, p-2 \text{ 且 } s(0) = 1$$

③ 令 $q_0=1$, 选择满足下面条件的连续质数集 $\{q_j\}$ ($j=1, 2, \dots, R-1$):

- $\text{g.c.d}(q_j, p-1)=1$, g.c.d 为计算最大公约数函数;

- $q_j > 6$;

- $q_j > q_{j-1}$ 。

(4) 按下面的方法置换序列 $\{q_j\}$ 得到 $\{r_j\}$:

$$r_{T(j)} = q_j \quad j = 0, 1, \dots, R-1$$

$T(j)$ 是行间置换的模式, 原来的第 j 行的置换后行位置为 $T(j)$, 它依赖于输入比特交织长度 K 。 $T(j)$ 定义如下:

$$T(j) = \begin{cases} P_A & 320 \leq K \leq 480 \\ P_C & 481 \leq K \leq 530 \\ P_A & 531 \leq K \leq 2280 \\ P_B & 2281 \leq K \leq 2480 \\ P_A & 2481 \leq K \leq 3160 \\ P_B & 3161 \leq K \leq 3210 \\ P_A & 3211 \leq K \leq 5114 \end{cases}$$

其中, P_A , P_B , P_C 分别定义如下:

$$P_A: \{19, 9, 14, 4, 0, 2, 5, 7, 12, 18, 10, 8, 13, 17, 3, 1, 16, 6, 15, 11\} \quad \text{对 } R = 20$$

$$P_B: \{19, 9, 14, 4, 0, 2, 5, 7, 12, 18, 16, 13, 17, 15, 3, 1, 6, 11, 8, 10\} \quad \text{对 } R = 20$$

$$P_C: \{9, 8, 7, 6, 5, 4, 3, 2, 1, 0\} \quad \text{对 } R = 10$$

(5) 对第 j 行 ($j=0, 1, 2, \dots, R-1$) 进行行内交织:

① 若 $C=p$, 则有

$$U_j(i) = s([i \times r_j] \bmod (p-1)) \quad i = 0, 1, 2, \dots, p-2 \text{ 且 } U_j(p-1) = 0$$

② 若 $C=p+1$, 则有

$$U_j(i) = s([i \times r_j] \bmod (p-1))$$

$$i = 0, 1, 2, \dots, p-2 \text{ 且 } U_j(p-1) = 0, U_j(p) = p$$

若 $K=R \times C$, 则将 $U_{R-1}(p)$ 与 $U_{R-1}(0)$ 置换。

③ 若 $C=p-1$, 则有

$$U_j(i) = s([i \times r_j] \bmod (p-1)) - 1 \quad i = 0, 1, 2, \dots, p-2$$

其中, $U_j(i)$ 是行内交织模式序列, 它表示第 j 行经置换后第 i 个输出比特的输入位置。

最后, 先按照 $U_j(i)$ 进行行内交织, 然后按 $T(j)$ 进行行间交织, 交织完了以后, 然后按列从交织矩阵中读出, 得到经交织以后的信息序列。(注: 由于信息序列的长度并不能保证其刚好写入交织矩阵中, 因此为了达到二者的匹配, 在信息序列后相应的补零: $l=C \times R-K$, 先前交织完成后, 经过修剪(pruning), 即完成交织。)

习 题

1. 已知一十六状态的 RSC 码的 $G(D)=\left[1, \frac{1+D+D^3+D^4}{1+D^4}\right]$,

(1) 画出此 RSC 编码器;

(2) 对长为 $L=8$ 的信息序列画出篱笆图;

(3) 求出信息序列 $M=(11100101)$ 相应的码字;

(4) 求出对上信息序列编码后归零所需的尾比特。

2. 一信息序列长为 490 bit, 经过一个 3G 交织器后, 原序列第 19 位信息处于交织后序列的第几位? (可采用计算机模拟编程求得。)

3. 假设一个八状态二进制 Turbo 码交织器大小为 M , 若译码深度为交织器大小, 分别计算一个 SISO 译码器采用 BCJR 译码算法和 SOVA 译码算法所需要的存储单元。(假设一个数值占用一个存储单元。)

附录 A Turbo 码主要译码算法的比较

| 算法名称 | 主要计算公式 | 性能比较与特点 |
|---------|--|---|
| MAP | $L(u_k) = \ln \frac{P_r\{u_k=1 R_1^N\}}{P_r\{u_k=0 R_1^N\}}$ <p>若 $L(u_k) \geq 0 \quad \hat{u}_k = 1$; 反之, $\hat{u}_k = 0$。</p> | 基于比特的译码算法 译码精度高 译码迟延大; 存储量大 |
| BCJR | $L(u_k) = \ln \frac{\sum_{\substack{(s', s) \\ u_k=1}} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s)}{\sum_{\substack{(s', s) \\ u_k=0}} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s)}$ | 基于比特的译码算法 译码精度高 译码迟延大, 计算时需要前、后向迭代, 计算量大; 存储量大 |
| SW-BCJR | $L(u_k) = \ln \frac{\sum_{\substack{(s', s) \\ u_k=1}} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s)}{\sum_{\substack{(s', s) \\ u_k=0}} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s)}$ <p>在计算的过程中引入了滑窗</p> | BCJR 的改进算法 适应于系统的实时应用 译码迟延大, 计算时需要前、后向迭代, 计算量大; 存储量大 |
| M-BCJR | $L(u_k) = \ln \frac{\sum_{\substack{(s', s) \\ u_k=1}} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s)}{\sum_{\substack{(s', s) \\ u_k=0}} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s)}$ <p>在计算过程中, 仅考虑 k 时刻所有的状态中, 状态度量值大小处于前 M 个的状态。</p> | BCJR 的改进算法 计算量较 BCJR 算法有一定的减少 M 的取值要求大于总状态数的 $2/3$, 性能较 BCJR 有恶化 |
| T-BCJR | $L(u_k) = \ln \frac{\sum_{\substack{(s', s) \\ u_k=1}} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s)}{\sum_{\substack{(s', s) \\ u_k=0}} \tilde{\alpha}_{k-1}(s') \cdot \gamma_k(s', s) \cdot \tilde{\beta}_k(s)}$ <p>在计算过程中, 仅考虑 k 时刻所有的状态中, 状态度量值大于所给定门限值的状态。</p> | BCJR 的改进算法 计算量减少 性能较 BCJR 有恶化; 门限的取值要求严格, 低信噪比下合适的门限在高信噪比下显示出不稳定性 |
| LOG-MAP | $L(u_k) = \ln \sum_{s^+} \alpha_{k-1}(s') \cdot \gamma_k(s', s') \cdot \beta_k(s)$ $- \ln \sum_{s^-} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s)$ $= \max_{s^+}^* [A_{k-1}(s') + R_k(s', s) + B_k(s)]$ $- \max_{s^-}^* [A_{k-1}(s') + R_k(s', s) + B_k(s)]$ $\max^*(x, y) = \ln(e^x + e^y) = \max(x, y) + f_c(x - y)$ $f_c(x - y) = \ln(1 + e^{- x - y })$ | BCJR 的对数域算法 计算量与计算级别降低, 复杂度降低, 适合于硬件实现。译码性能与 BCJR 相近 具有一定的译码迟延 |

| 算法名称 | 主要计算公式 | 性能比较与特点 |
|-----------------|---|---|
| MAX - LOG - MAP | $\max^*(x, y) = \ln(e^x + e^y) \approx \max(x, y)$ | BCJR 对数域算法的近似算法 复杂度更低，存储量更小，适合硬件实现。译码性能略有恶化（较 LOG - MAP 恶化约 0.5 dB） |
| OSA | $P(u_{k-D} = i S_k = s, y_j^k) = \frac{\sum_i \sum_j P(u_{k-D} = i S_{k-1} = s', R_1^{k-1}) \cdot \tilde{\alpha}_{k-1}(s') \cdot \gamma_k^i(s', s)}{\sum_i \sum_j \tilde{\alpha}_{k-1}(s') \cdot \gamma_k^i(s', s)}$ $P(u_{k-D} = i S_{k-D} = s, R_1^{k-D}) = \frac{\sum_j \tilde{\alpha}_{k-D-1}(s') \cdot \gamma_{k-D}^i(s', s)}{\sum_i \sum_j \tilde{\alpha}_{k-D-1}(s') \cdot \gamma_{k-D}^i(s', s)}$ | 适用于连续的数据流的译码算法，适合于实时译码，不需后向递推，译码迟延与存储需求小 译码性能随译码深度（迟延）的增加逼近于 BCJR 算法 计算复杂度较高 |
| HR - SOVA | $u_j^i \neq u_j^e \Rightarrow L_j^i = \min(L_j^i, \Delta)$ | 基于序列的译码算法 适用于连续数据流的实时译码，译码迟延与计算复杂度低，适合于硬件实现 译码性能较 BCJR 算法有较大差异，恶化量大于 1 dB。 |
| BR - SOVA | $\begin{cases} u_j^i \neq u_j^e \Rightarrow L_j^i = \min(L_j^i, \Delta) \\ u_j^i = u_j^e \Rightarrow L_j^i = \min(L_j^i, \Delta + L_j^e) \end{cases}$ | HR - SOVA 的修正算法 适用于连续数据流的实时译码，译码迟延与计算复杂度低，适合于硬件实现 译码性能较 HR - SOVA 有一定的改善，较 BCJR 算法，恶化量约为 0.5 dB |

参 考 文 献

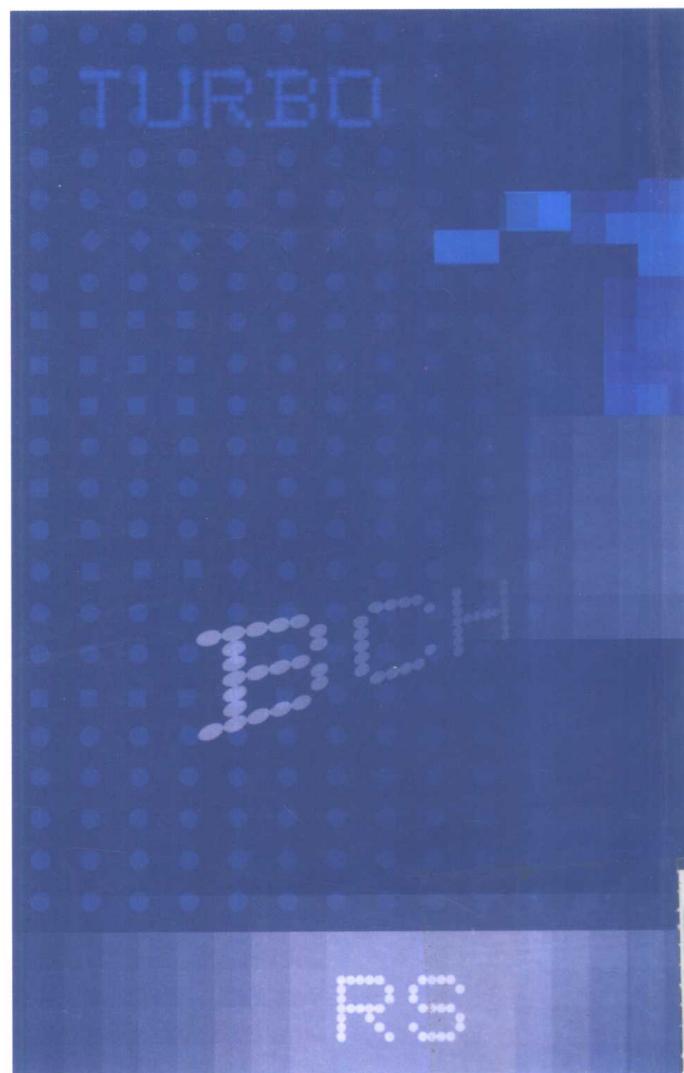
- [1] Chris Heegard and Stephen B. Wicker, "Turbo Coding", Kluwer Academic Publishers, 1999.
- [2] 白宝明：“Turbo 码理论及其应用的研究”，西安电子科技大学博士论文，1999
- [3] C. Berrou, A. Glavieux, and P. Thitimajshima, Near Shannon Limit error-correcting coding and decoding: Turbo-codes, in Proc. ICC'93, pp. 1064—1070, May 1993.
- [4] Claude Berrou and Alain Glavieux, Reflections on the Prize Paper: "Near optimum error-correcting cod-

- ing and decoding: turbo codes”, IEEE information theory society newsletter, Vol. 48, No. 2, June 1998.
- [5] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate, IEEE Trans. Inform. Theory, Vol. IT-20, pp. 248–287, Mar. 1974.
- [6] S. Benedetto and G. Montorsi, “Unveiling turbo code: Some result on parallel concatenated coding schemes”, IEEE Trans. Inform. Theory, Vol. 42, No. 2, pp. 409–428, Mar. 1996.
- [7] S. Benedetto and G. Montorsi, “Design of parallel concatenated convolutional codes”, IEEE Trans. Commun., Vol. 44, No. 5, pp. 591–600, May. 1996.
- [8] J. Hagenauer and P. Hoeher, “A Viterbi Algorithm with Soft—Decision Outputs and its Applications”, in Proc. Globecom’89, pp. 1680–1686, Nov. 1989.
- [9] TS 25. 212 V2. 2. 0(1999–09), 3rd Generation Partnership Project (3GPP), available at <http://www.3gpp.org>.
- [10] T. M. Duman and M. Salehi, “New Performance Bounds for Turbo Codes”, IEEE Trans. Commun., Vol. 46, No. 6, pp. 717–723, Jun. 1998.



XDUP 0330

封面设计:年代设计有限公司



ISBN 7-5606-0163-4

9 787560 601632 >

ISBN 7-5606-0163-4/TN·0059 (课)

定价: 35.00元