國 立 交 通 大 學

電信工程學系

碩 士 論 文

具有嵌入式循環冗餘檢驗特性之錯誤更正技術於混合式自動重傳請求機制下之研究

CRC-embedded Error Correction for Hybrid ARQ

研 究 生：孫 玉 婷

指導教授：王 忠 炫 博士

中華民國九十八年七月

具有嵌入式循環冗餘檢驗特性之錯誤更正技術於混合式自動重傳請求機制下之研究

CRC-embedded Error Correction for Hybrid ARQ

研 究 生：孫玉婷　　Student：Yu-Ting Sun

指導教授：王忠炫 博士　　Advisor：Dr. Chung-Hsuan Wang

國 立 交 通 大 學

電信工程學系碩士班

碩 士 論 文

A Thesis
Submitted to Department of Communication Engineering
College of Electrical and Computer Engineering
National Chiao-Tung University
in Partial Fulfillment of the Requirements
for the Degree of
Master of Science
In
Communication Engineering
July 2009
Hsinchu, Taiwan, Republic of China

中華民國九十八年七月

具有嵌入式循環冗餘檢驗特性之錯誤更正技術於混合式自動重傳請求機制下之研究

CRC-embedded Error Correction for Hybrid ARQ

研究生：孫玉婷　　指導教授：王忠炫 博士

國立交通大學電信工程學系碩士班

摘要

在使用串接碼的通訊系統下，循環冗餘檢驗碼是一種通常被當作內部碼用於錯誤偵測的截短碼而迴旋碼則被視為外部碼。然而，如果吾人不僅想使用循環冗餘檢驗碼的錯誤偵測能力而且也想使用其錯誤更正能力，則最直接增加效能的想法就是利用迴旋碼和循環冗餘檢驗碼的聯合解碼來取代傳統的迴旋解碼。其中一種方式就是結合循環冗餘檢驗碼和迴旋碼的解碼網格。然而，雖然使用循環冗餘檢驗碼和迴旋碼的結合網格作解碼的聯合解碼法可以降低區塊錯誤率，但在實際運用上是不可行的。因此，在利用結合網格做解碼的情況下，吾人提出一種針對循環冗餘檢驗碼修正後的編碼方式以及在從結合網格造出的縮小網格上的一種新解碼系統。模擬結果顯示，不只修正後的循環冗餘檢驗解碼程序且提出的解碼系統都會比傳統的重傳系統得到較低的錯誤率。和表單維特比演算法的比較以及提出的解碼方式也會在此論文中呈現。

CRC-embedded Error Correction for Hybrid ARQ

Student: Yu-Ting Sun Advisor: Chung-Hsuan Wang

Department of Communication Engineering

National Chiao Tung University

Abstract

As we know, a cyclic redundancy code (CRC) is a shortened cyclic code which is commonly used as an inner code for error detection and an convolutional code is considered as an outer code in the communication system employing a serially concatenated code. However, if we utilize not only the error detecting capability but also the error correcting ability of CRC, it is straightforward to improve the performance by employing the joint decoding of convolutional codes and CRC instead of the conventional convolutional decoding. One of methods is to combine the decoding trellis of CRC and convolutional codes. Nevertheless, the joint decoding which applies a combined trellis composed of CRC and convolutional codes was infeasible in practical implementation although the block error rate can be deceased. Accordingly, under the circumstance of utilizing the combined trellis for decoding, we propose a modified encoding method for CRC codewords and a new decoding scheme on the reduced trellis constructed from the combined trellis. Simulation results exhibit that not only the modified CRC encoding process but also the proposed decoding scheme achieves lower error rate than the original retransmission schemes. The comparison between list Viterbi algorithm (LVA) and the proposed decoding scheme will be illustrated in this thesis in addition.

誌 謝

在這兩年的研究所生涯中，首先要特別感謝我的指導教授王忠炫博士。當我在研究上遇到困難及瓶頸時，老師總是適時的給予方向引導我們，在每次的meeting中又耐心的教導和提出指正。我不僅從老師身上學到專業知識，更學到如何嚴謹地思考任何問題及做研究的方法。非常感謝老師這兩年來的諄諄教誨。

再來要感謝的是我的家人：非常感謝你們在我遇到挫折和脆弱的時候能夠完全地支持我，雖然相隔的距離很遠，但你們給予我的力量卻每每都從手機的另一端確確實實地傳了過來。

再來要感謝的是實驗室的學長、同學及學弟妹們：首先感謝實驗室的學長力仁、老菜、小白、一哥，讓我們還是小碩一時可以很快地融洽地融入實驗室。而博班學長大師兄，更總是不厭其煩地和我們討論問題，並且指引了一些研究的方向。實驗室在他的照耀下，讓大家在迷茫的研究裡，總是能探索出解決的方法。和我一起共患難的標和郭胖，感謝你們總是帶領著反應較遲鈍的我一起向前邁進。也謝謝學弟妹泡泡和十三為實驗室帶來了不少的歡笑歡樂。

再來是我的朋友們及其他實驗室的同學們，有你們的陪伴、支持及大力相助，才讓我得以支撐到現在！

感謝所有幫助過我的人，謝謝你們！

Contents

List of Figures

List of Tables

Chapter 1

Introduction

There are various error-control strategies such as forward error correction (FEC) and automatic repeat request ARQ in the communication system. The former is used for one-way communication system and employs error-correcting codes that automatically correct errors detected at the receiver, and the latter uses an error-detecting code and an error-correcting code for two-way communication system which permits retransmission. When the receiver detects errors in the ARQ scheme, a request is sent to the transmitter to repeat the message, and the repeated request continuously to be sent until the message is correctly received. However, these two error-control strategies both have inevitable drawbacks. The drawback of the ARQ is that the throughput falls rapidly with increasing channel error rate. For FEC systems, there are two drawbacks: First, when a received vector is detected in error, it must be decoded and delivered to the user regardless of whether it is correct or not. Second, to obtain high system reliability, a long powerful code which makes decoding difficult to implement is necessarily required. Due to the drawbacks in both ARQ and FEC, the combination of these two error control schemes was proposed as a hybrid ARQ [2]. A hybrid ARQ system is an ARQ system consists of an FEC subsystem which is to reduce the frequency of retransmission by correcting the error patterns that occur most frequently. There are several kinds of hybrid ARQ schemes such as Type-I HARQ, Type-II HARQ, Type-III HARQ and reliability-based Hybrid ARQ (RB-HARQ) [1, 3, 4, 5]. When a less frequent error pattern occurs and is detected by error detecting part, the receiver

request a retransmission by ARQ rather than passing the unreliably decoded message to the user. In this way, the system throughput and reliability can be both increasing due to the combination.

The method of increasing the decoding capability of the error correcting part is one of the significant issues in the hybrid ARQ schemes. The retransmission times can be decreased by the increasing rate of succeeded packages which are decoded by the error correcting part. Furthermore, if the number of retransmission times is restricted by the system, superior error correction can decrease the block error rate under limited retransmission times. We consider about the system of using a convolutional code as an error correction code (an inner code) and a cyclic redundancy code (CRC) as an error detection code (an outer code) in this thesis on account of the general employment of this system in the present transmission system. Therefore, we attempt to design some decoding schemes which can improve the throughput and alleviate the block error rate caused by the convolutional decoder in hybrid ARQ system. There are two directions to approach the goal: the first is to attain L^{th} best paths in 1st transmission instead of one ML path. The second is to enhance the decoding capability of the error correction codes. The idea of first direction is proposed as a list Viterbi decoding algorithm (LVA) [11] by Seshadri in 1994. In the system of using convolutional codes as inner codes and CRC as outer codes, LVA tries to search another acceptable decoded codeword to pass the error detection before next retransmission while the ML codeword failed. Thus, the block error rate will be decreased if the retransmission is prohibited or the number of retransmission times is restricted. Nevertheless, the improvement in performance provided by LVA is confined because the correct codeword will not be found in the searching of the L^{th} best paths in an extremely noisy channel. That is to say, the performance will not be upgraded even the L has been increased to a huge number. Consequently, we should try to enhance the decoding capability of the convolutional decoder itself.

As we know, CRC is a shortened cyclic code which is commonly used for error detection in two-way transmission systems. For the conventional convolutional decoder, it decodes an information sequence which contains CRC bits from received sequence. Then we can

detect errors in the decoded sequence by checking the remainder of the decoded sequence divided by the generator polynomial is zero or not. However, if we utilize not only the error detecting capability but also the error correcting capability of CRC, it is straightforward to improve the performance by employing the joint decoding of convolutional codes and CRC instead of the conventional convolutional decoding. One of methods of correcting message errors using CRC is to combine the parity-check matrix of the CRC and the parity-check matrix for any error correcting code which is a block code and is used in conjunction with the CRC. The method with the combination of varied parity-check matrices was presented by Spencer [12].

One of methods for the joint decoding of convolutional codes and CRC is to combine the decoding trellis of CRC and convolutional codes. Nevertheless, the joint decoding which applies a combined trellis composed of CRC and convolutional codes was infeasible in practical implementation although the block error rate can be deceased. This kind of decoding scheme considers not only the states of convolutional codes but also the states of CRC in the combined trellis. Thus, an excessively high complexity will be approached because of the increasing number of states. In addition, the codewords decoded from the combined trellis are always the CRC codewords which result in the extinguishment of the error detecting capability. Accordingly, under the circumstance of utilizing the combined trellis for decoding, we propose a modified encoding method for CRC codewords and a new decoding scheme on the reduced trellis constructed from the combined trellis. The proposed encoding method is partitioning the original CRC into several smaller CRCs and modifying the encoding procedure for CRC codewords to form a new combined trellis consisting of these small CRCs and convolutional codes. For the proposed decoding scheme, we first derive the reduced trellis of the combined trellis and then obtain a simple procedure to construct sub-trellises in the reduced trellis of the combined trellis for a time-varying decoding. These two modified encoding and decoding methods both elevate the performance without a tremendous amount of computation and restore the error detecting ability of CRC by applying the distinct adjusted trellises of the original combined trellis respectively. Simulation results exhibit not

only the modified CRC encoding process but also the proposed decoding scheme achieves lower error rate than the original retransmission schemes The comparison between LVA and the proposed decoding scheme will be illustrated in this thesis in addition.

The remainder of this thesis is organized as follows. In Chapter 2, an overview of convolutional codes and hybrid ARQ schemes is introduced here. In Chapter 3, we give a review of soft-decision decoding algorithms for convolutional codes such as maximum likelihood decoding, sequential decoding and maximum likelihood soft-decision decoding algorithm. The modified encoding method for CRC and the method of constructing a reduced trellis of the combined trellis are presented in Chapter 4. The formation of the combined trellis and a time-varying decoding algorithm on the combined trellis are also given in this chapter. Simulation results are shown in Chapter 5 for performance verifications. In Chapter 6, we summarize our investigations and present some potential future works for improvements to performance and diminished computation times.

Chapter 2

Overview of Convolutional codes and Hybrid ARQ Schemes

2.1 Convolutional Codes

Differing from block codes, the encoder of convolutional codes contains memory, and the encoder outputs at any given time unit depend not only on the inputs at that time unit but also on some number of previous inputs. A rate $R = k/n$ convolutional encoder with memory order m can be realized as a k-input, n-output liner sequential circuit with inputs which are remaining in the encoder for m time units after entering. Large minimum distances and low error probabilities can be achieved by increasing memory size m distinct from block codes.

A rate $R = k/n$ convolutional encoder with memory order m is referred to as an (n, k, m) encoder. For an encoder with memory order m, the impulse responses can last at most $m+1$ time units, so the constraint length ν is defined as $\nu = 1 + m$. A binary diagram of a rate $R = 1/2$ nonsystematic feedforward convolutional encoder with memory order $m = 3$ is shown in Figure 2.1.

Besides the decoding algorithm employed, the distance properties of convolutional codes also have a great effect on the performance. The most significant distance measure for convolutional codes is the minimum *free distance* d_{free}. The minimum free distance of a

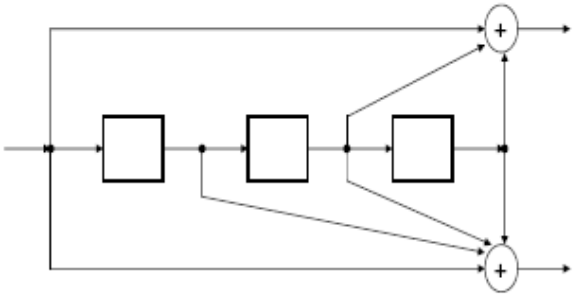

Figure 2.1: A rate $R = 1/2$ binary convolutional encoder with memory order $m = 3$

convolutional code is defined as

$$d_{\text{free}} \triangleq \min_{\boldsymbol{u}', \boldsymbol{u}''} \{d(\boldsymbol{v}', \boldsymbol{v}'') : \ \boldsymbol{u}' \neq \boldsymbol{u}''\}$$

where $\boldsymbol{v}'$ and $\boldsymbol{v}''$ are the codewords corresponding to the information sequences $\boldsymbol{u}'$ and $\boldsymbol{u}''$, respectively.

It is assumed that the codeword v' and v'' have finite length, and starts and ends in the all-zero state S_0; that is, m termination blocks are appended to the information sequences $\boldsymbol{u}'$ and $\boldsymbol{u}'$. Hence, d_{free} is the minimum distance between any two finite-length codewords in the code. Because a convolutional code is a linear code, d_{free} is not only the minimum-weight codeword produced by any finite length non-zero information sequence but also the minimum weight of all finite-length paths in the state diagram that diverge from and remerge with the all-zero state S_0. In order to improve the performance of convolutional decoding, a large free distance d_{free} and a small number of nearest neighbors $A_{d_{free}}$ is demanded in the purpose of a low uncorrected error probability.

Another important distance measure for convolutional codes is the *Column distance function* (CDF). Let

$$[\boldsymbol{v}]_l = (v_0^{(0)} v_0^{(1)} \cdots v_0^{(n-1)}, v_1^{(0)} v_1^{(1)} \cdots v_1^{(n-1)}, \cdots, v_l^{(0)} v_l^{(1)} \cdots v_l^{(n-1)})$$

denote the lth truncation of the codeword $\boldsymbol{v}$, and let

$$[\boldsymbol{u}]_l = (u_0^{(1)} u_0^{(2)} \cdots u_0^{(k)}, u_1^{(1)} u_1^{(2)} \cdots u_1^{(k)}, \cdots, u_l^{(1)} u_l^{(2)} \cdots u_l^{(k)})$$

denote the lth truncation of the information sequence $\boldsymbol{u}$.

The column distance function of order l, d_l, is defined as

$$\begin{aligned} d_l &\triangleq \min_{[\boldsymbol{u}']_l,[\boldsymbol{u}'']_l} \{d([\boldsymbol{v}']_l,[\boldsymbol{v}'']_l) : [\boldsymbol{u}']_0 \neq [\boldsymbol{u}'']_0\} \\ &= \min_{[\boldsymbol{u}]_l} \{w([\boldsymbol{v}]_l) : [\boldsymbol{u}]_0 \neq \mathbf{0}\}. \end{aligned}$$

where $\boldsymbol{v}$ is the codeword corresponding to the information sequence $\boldsymbol{u}$. For Sequential decoding, a good performance requires initial column distance growth rapidly for fast decoding and a low erasure probability. The *distance profile* of a code is defined as its CDF over only the first constraint length, that is, d_0, d_1, $\cdots$, d_v. Since the distance profile determines the initial column distance growth and is easier to compute than the entire CDF, it is often used instead of the CDF as a criterion for selecting codes for using sequential decoding. A code is said to have an *optimum distance profile*(ODP) if its distance profile is *superior* to that of any other code of the same rate and constraint length. A distance profile d_0, d_1, $\cdots$, d_v of a code is considered *superior* to the distance profile d_0, d_1, $\cdots$, d_v of another code of the same constraint length v if for some t, $0 \leq t \leq v$,

$$d_l = \begin{cases} = d'_l, & l = 0, 1, \cdots, t-1 \\ > d'_l & l = t. \end{cases}$$

That is to say, the initial portion of the CDF determines which code has the superior distance profile. Usually, the sequential decoding algorithm requires a rapid initial growth of CDF in order to have a small decoding complexity [1].

2.1.1 Maximum Likelihood Decoding and Sequential Decoding for Convolutional Codes

In communication theory and coding theory, decoding is the process of translating received sequences into codewords of a given code and is often used to recover messages sent over a noisy channel, such as a binary symmetric channel or an additive white Gaussian noise(AWGN) channel. Given a received codeword $\mathbf{r}$, *maximum likelihood decoding* (MLD) selects a codeword $\mathbf{v}$ to maximize the channel transition probability

$$P(\boldsymbol{r}|\boldsymbol{v}).$$

For a DMC,

$$P(\boldsymbol{r}|\boldsymbol{v}) = \prod_i P(r_i|v_i). \tag{2.1}$$

since for a memoryless channel each received symbol depends only on the corresponding transmitted symbol. Because $\log x$ is a monotone increasing function of x, maximizing $P(\boldsymbol{r}|\boldsymbol{v})$ is equivalent to maximizing the *log-likelihood function*,

$$\log P(\boldsymbol{r}|\boldsymbol{v}) = \sum_i \log P(r_i|v_i).$$

For convolutional codes in the AWGN channel, a viterbi decoding algorithm is a ML decoding algorithm for a bitstream that has been encoded using forward error correction based on a Convolutional code.

However, when the noise is light, Viterbi algorithm is not always needed because it performs on the order of 2^v computations while v is the constraint length. Therefore, the sequential decoding is proposed to be a decoding procedure which has the decoding effort adaptable to the noise level and can be used without great resource-consuming even for a code with large constraint length v. In 1963 Fano introduced a version of sequential decoding with Fano metric on code tree[7] and few years later another version of sequential decoding, called the *ZJ* or *stack algorithm*, was discovered independently by Zigangirov and Jelinek [8]. The purpose of a sequential decoding algorithm is to search through the nodes of the code tree in an efficient way instead of exploring the whole trellis. A code tree of a binary convolutional code $(2, 1, 2)$ with generator polynomial $\boldsymbol{g} = [7\ 5]_{octal}$ is illustrated in Figure 2.2.

The paths that have been examined by sequential algorithm after any decoding step are generally of many different length. For the Viterbi algorithm, the paths being compared at any decoding step are all of the same length. The Fano branch metric takes into account the lengths of distinct paths desired to compare and is defined as

$$M(r_l|v_l) = \log_2 \frac{P(r_l|v_l)}{P(r_l)} - R, \tag{2.2}$$

Figure 2.2: The Code tree for the binary (2, 1, 2) convolutional code with information length 5. Input zero tail bits after level 5 to return back to the zero state. A codeword $\boldsymbol{v}$ = (11 01 10 01 00 10 11) is shown above.

where $P(r_l|v_l)$ is a channel transition probability, $P(r_l)$ is a channel output symbol probability, and R is the encoder rate. The Fano path metric for the first $\boldsymbol{t}$ branches of a path $\boldsymbol{v}$:

$$\begin{aligned} M([\boldsymbol{r}|\boldsymbol{v}]_t) &= \sum_{l=0}^{t-1} M(\boldsymbol{r}_l|\boldsymbol{v}_l) \\ &= \sum_{l=0}^{nt-1} M(r_l|v_l) \\ &= \sum_{l=0}^{nt-1} \log_2 P(r_l|v_l) - \sum_{l=0}^{nt-1} \log_2 P(r_l) - ntR \\ &= \underbrace{\sum_{l=0}^{nt-1} \log_2 P(r_l|v_l)}_{ML\ metric} + \underbrace{\sum_{l=0}^{nt-1} [\log_2 \frac{1}{P(r_l)} - R]}_{positive\ bias}. \end{aligned} \tag{2.3}$$

For the sake of avoiding unfairly comparison between paths with different length, the Fano path metric consists of not only ML metric but also a positive bias term to forecast the prospective paths. In this way, the longer path will have more possibility to be explored first and reach the end of the code tree.

In addition to the definition of metric, an ordered list or a stack of previously examined paths of different lengths is required keeping in storage. Each stack entry contains a path along with its metric. The path with the largest metric is placed on top, and the others are listed in order of decreasing metric. The ZJ(Stack) Algorithm is delineated as below:

The ZJ(Stack) Algorithm:

Step 1. *Load stack with origin node.*

Step 2. *Compute metric of successors of top path.*

Step 3. *Delete top path from stack.*

Step 4. *Reorder stack according to metric values.*

Step 5. *If the top path reaches at the end of tree, stop and output the decoded codeword. Otherwise, return to Step 2.*

Advantages:

- The number of computations performed by a sequential decoder and the erasure probability are independent of the encoder constraint length.
- Decoding effort is adaptable to the noise level.
- A sequential decoder considers errors as erasure by "sensing" noisy frames due to the elimination of the overflowed received buffer. A complete decoder (such as Viterbi algorithm) will always decode a frame and deliver to the user even though the decoded codeword is incorrect.

Disadvantages:e

- Some extensive searches can lead to buffers which storing incoming data overflow, causing information bit to be lost or erased.
- Due to the finite stack size, it may cause the path at the bottom of the stack be lost and probably remove the ML path.
- Although the number of computations performed by a sequential decoder is a random variable, a extremely noisy received sequence will require a large amount of computations with a sequential decoding algorithm.

Reordering the stack after each decoding step causes a tremendous amount of complexity, a modified algorithm, *stack-bucket algorithm*, is proposed by Jelinek [8]. It quantizes the path metric storing in the stack in order to decrease the complexity of sorting stack. Figure 2.3 illustrates the quantization of metric in stack. If the received sequence is not excessively noisy and the quantization of metric values is not too coarse, the degradation in performance from stack-bucket algorithm is very slight.

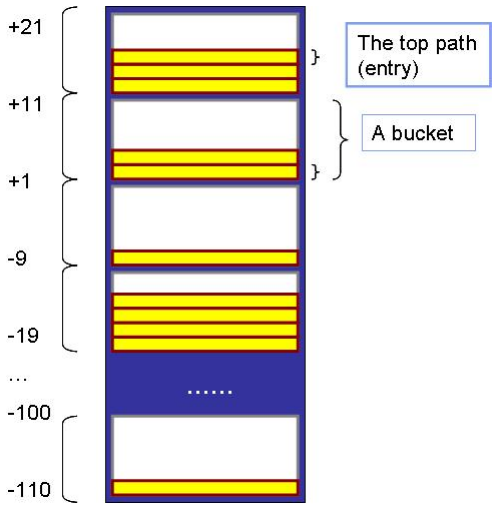

Figure 2.3: The quantized stack of stack-bucket algorithm.

2.2 Cyclic Redundancy Codes

CRC is a shorten cyclic code which is commonly used for error detection in two-way transmission system. The characteristic of CRC is to simply detect errors in a long received sequence and form a powerful class of codes suited especially for the detection of burst errors. The basic idea behind CRC is to treat the message as a binary information sequence $u(x)$, and divide it by a generator polynomial $g(x)$ that is known to both the transmitter and the receiver. The remainder $R(x)$ left after dividing $u(x)$ by $g(x)$ constitutes the parity check bits for the given message. The transmitter sends both the message string $u(x)$ and the check bits, and the receiver can then check the data by dividing $u(x)$ by the generator polynomial $g(x)$, and verifying that the remainder is $R(x)$. Generally, we denote a CRC with p parity bits as a CRC-p.

There is a simple encoding procedure for CRC. Let $u(x)$ and $v(x)$ represent the polynomials of a k-bit message and the corresponding n-bit codeword respectively, and let $g(x)$ be the generator polynomial of degree $n-k$. The degree of $u(x)$ and $v(x)$ is equal to $k-1$ and $n-1$. Following the procedure of encoding the systematic cyclic codes, the CRC codeword $v(x)$ can be written as

$$R(x) = u(x) \cdot x^{n-k} \bmod g(x)$$

$$\begin{aligned} v(x) &= u(x) \cdot x^{n-k} + R(x) \\ &= u(x) \cdot x^{n-k} + u(x) \cdot x^{n-k} \bmod g(x) \end{aligned}$$

In this way, we can detect errors in the received sequence by checking the remainder of the received sequence divided by $g(x)$ is zero or not.

However, this method of checking errors is apparently not perfect, because there are a great amount of distinct message strings that give the equivalent remainder when divided by $g(x)$, the received sequence could be decoded into another codeword. In such case, the error would go undetected. Therefore, the selection of generator polynomial is an essential part of implementing the CRC algorithm. The polynomial must be chosen to maximise the error detecting capabilities while minimising overall collision probabilities.

2.3 Conventional Hybrid ARQ Schemes

There are various error-control strategies such as *forward error correction*(FEC) and *automatic repeat request*(ARQ) in the communication system. The former is used for one-way system and employs error-correcting codes that automatically correct errors detected at the receiver, and the latter uses an error-detecting code and an error-correcting code for two-way system which permits retransmission. In an ARQ scheme, when the receiver detects errors, a request is sent to the transmitter to repeat the message, and the repeated request continuously to be sent until the message is correctly received.

However, these two error-control strategies both have inevitable drawbacks. The drawbacks of ARQ systems is that the throughput falls rapidly with increasing channel error rate. For FEC systems, there are two drawbacks: First, when a received vector is detected in error, it must be decoded and delivered to the user regardless of whether it is correct or not. Second, to obtain high system reliability, a long powerful code which makes decoding difficult to implement and expensive is necessarily required.

The drawbacks in both ARQ and FEC can be overcome by properly combining these two error control schemes. Such a combination is referred to as a *hybrid* ARQ[2]. A hybrid ARQ system is an ARQ system consists of an FEC subsystem which is to reduce the frequency

of retransmission by correcting the error patterns that occur most frequently. When a less frequent error pattern occurs and is detected, the receiver request a retransmission by ARQ rather than passing the unreliably decoded message to the user. In this way, the system throughput and reliability can be both increasing due to the combination.

There are several kinds of Hybrid ARQ schemes such as Type-I HARQ, Type-II HARQ, Type-III HARQ, ..., and reliability-based Hybrid ARQ (RB-HARQ)[1, 3, 4, 5].

- Type I hybrid ARQ: it use an (n, k) linear code that is designed to simultaneously correct and detect errors. Error correction and retransmission continue until the received sequence is successfully decoded.

- Type II hybrid ARQ: by using the incremental redundancy(IR) scheme, the parity check digits for error correction are sent to the receiver only when they are needed. Two linear codes are used in this scheme:

  - A high-rate (n, k) code C_0 for error detection only.
  - An invertible half-rate $(2k, k)$ code C_1 for error correction and error detection.

  If the decoding of the combination of information bits and parity bits fails after retransmission, the transmitter retransmits the first information block again and repeats the process.

- Type III hybrid ARQ: combining the advantages of Type I and Type II, Type III hybrid ARQ utilizes the *partial* incremental redundancy scheme and regards Complementary Punctured Convolutional (CPC) codes as an error correction code. The main advantage of using CPC codes is that any complementary sequence sent for a packet that is detected in errors is self-decodable. Differing from Type II HARQ, the decoder needs not to rely on previously received sequences for the same data packet for decoding.

- Reliability-based Hybrid ARQ: based on the estimated bit reliabilities (LLR), the bits that suppose to be retransmitted are adaptively selected at the receiver. On account

of the desired acknowledge of LLR, a Soft-in Soft-out (SISO) decoder (i.e. BCJR, SOVA) is needed with convolutional codes, turbo codes or LDPC codes.

In practice, incorrectly decoded data blocks are often stored at the receiver rather than discarded. Thus, the retransmitted block and the former incorrect blocks should be combined. Although available received sequences are not possible to decode error-free codewords independently, it may happen that the combination of all the previously erroneously received transmissions give us enough information to correctly decode. There are mainly two ways of re-combining in HARQ:

- Chase combining: every retransmission contains the same data and parity bits. One could think of every retransmission adding extra "power" to the received transmission.
- Incremental redundancy: the first transmission has only information bits and few parity bits, incremental redundancy bits are added as the retransmission begins. In other words, every retransmission contains different information than the previous one and the receiver gains knowledge of extra information at every retransmission.

One of applications of the incremental redundancy scheme is the Rate-compatible punctured convolutional codes (RCPC codes) proposed by Hagenauer[6]. Instead of repeating information and parity bits if the retransmission is unsuccessful as in previous type-II HARQ scheme, we transmit additional code bits of a low rate RCPC code until the code is powerful enough to enable decoding.

Chapter 3

CRC-assisted Error Correction Schemes for Convolutional Codes

The method of increasing the decoding capability of the error correcting part is one of significant issues in the Hybrid ARQ schemes. We attempt to design some encoding and decoding schemes which can improve the throughput and decrease the error rate. Instead of obtaining L^{th} best paths before next retransmission, enhancing the decoding capability of the error correction codes is our desired objective.

The proposed decoding algorithm including of the procedure of combining the trellises and the time-varying decoding strategy is presented in this chapter. Several proposed methods of reducing the combined trellis are also provided.

3.1 The Combined Trellis of CRC and Convolutional Codes

As we know, CRC is a shorten cyclic code which is commonly used for error detection in two-way transmission system. If we use not only the error detecting capability but also the error correcting capability of CRC, it is feasible to strengthen the decoding ability of the convolutional decoder by adding the effect of CRC. For the purpose of considering the restriction caused by CRC codeword, we propose a method to combine the decoding trellis of CRC and convolutional codes. At first, consider about a simple case by using a $(2, 1, 2)$ convolutional codes , $\boldsymbol{g} = [101\ 111]_{\text{binary}}$, free distance $= 5$ and CRC generator polynomial

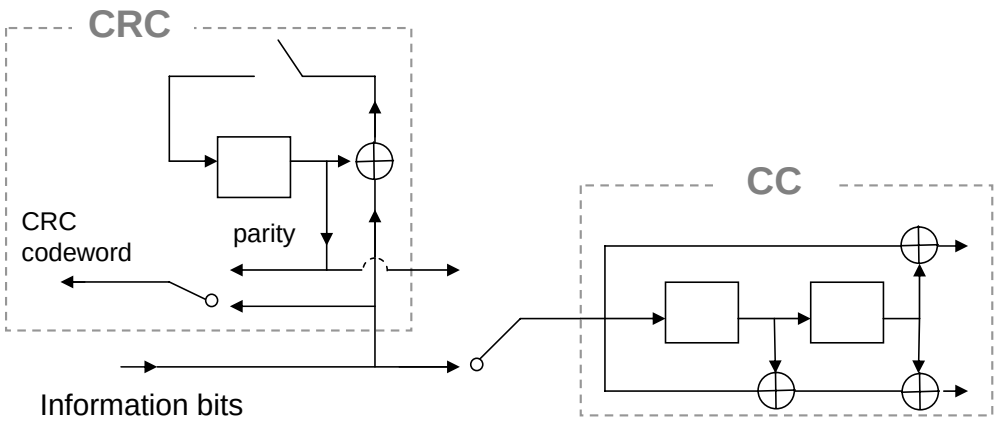

Figure 3.1: A rate $R = 1/2$ binary convolutional encoder with memory order $m = 2$ and a CRC with generator polynomial $g(x) = 1 + x$

$g(x) = 1 + x$. The encoding block diagram can be illustrated in Figure 3.1 and the state diagram of the combined codes is presented in Figure 3.2.

The most left bit in Figure 3.2 is the state transition of CRC and the right two bits are the state transition of convolutional codes (CC). The dotted lines represent that input bit is 0 and the solid lines mean that 1 is the input bit. Its respective input bit/output code bitson each branch is labeled equivalently as the state diagram of the conventional convolutional decoder. The combined trellis is shown in Figure 3.3. Note that each state in the combined trellis is composed of the state of CRC and the state of convolutional codes.

3.2 A time-varying Decoding Algorithm on Combined trellis

After the combined trellis was constructed, we should figure out a method of merging in the end of the combined trellis. The convolutional decoder will force the current state return to zero state by inputting zeros to memory registers in the end of the trellis in the conventional convolutional decoding. Since the CRC encoder is a feedback circuit, we should input a bit stream which is same as the one in memory registers rather than putting bits of all zeros into registers. This force us to separate the merging section of a trellis: the first merging portion is for CRC and the second one is for original convolutional codes. Differing

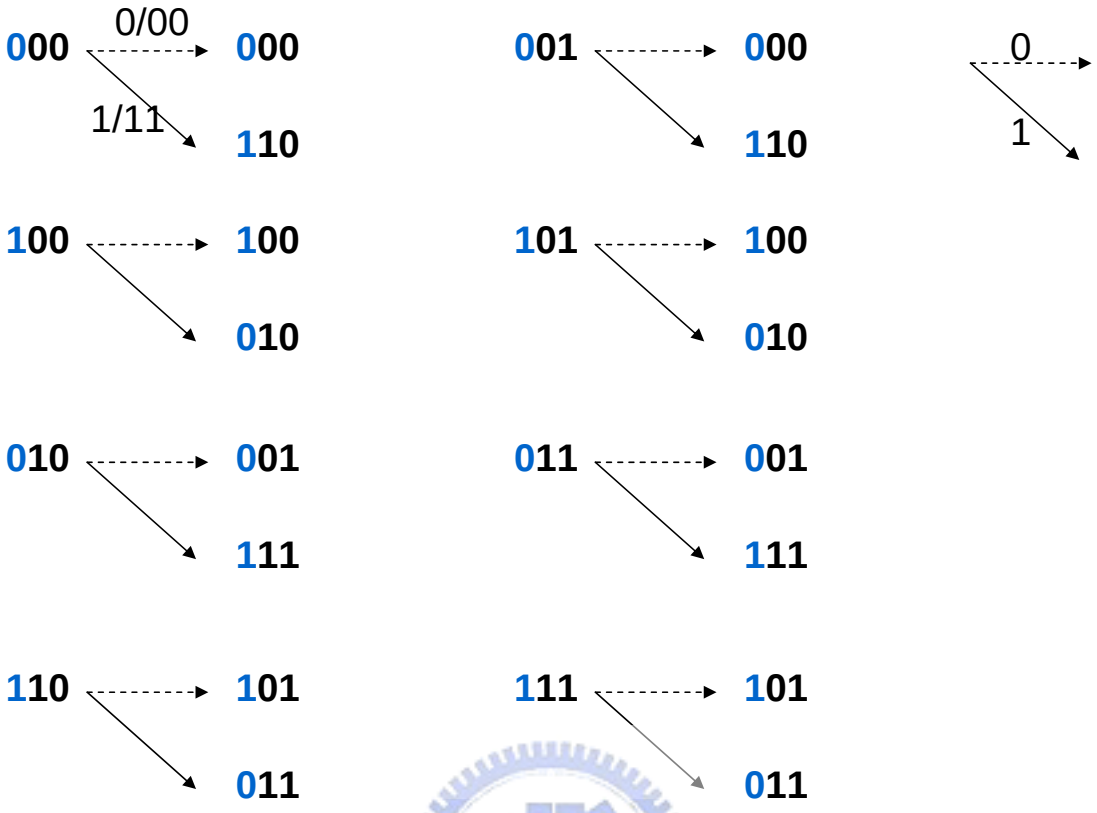

Figure 3.2: The state diagram of combined CRC and convolutional codes

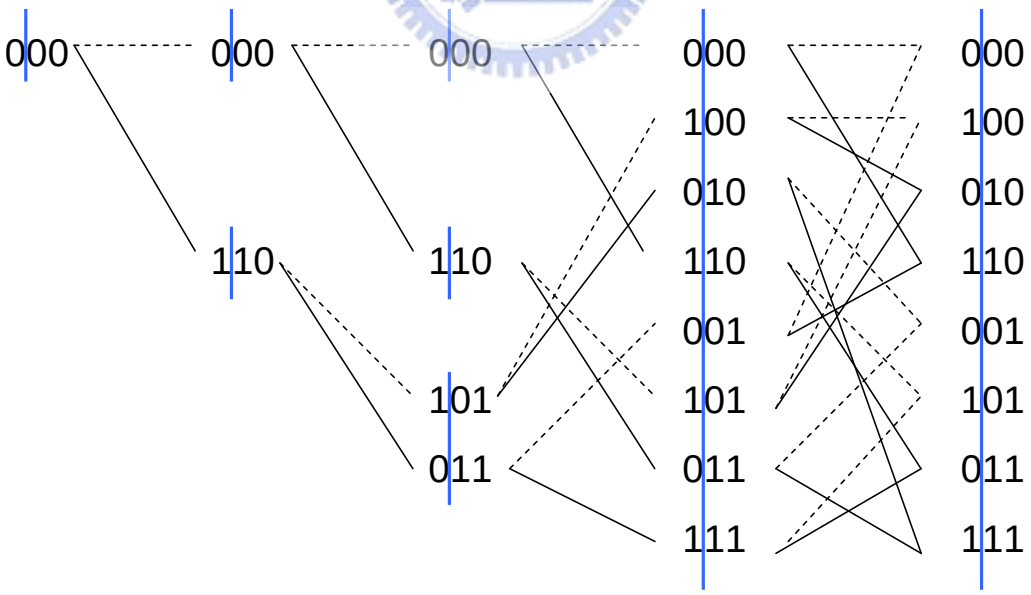

Figure 3.3: The combined trellis of CRC and convolutional codes

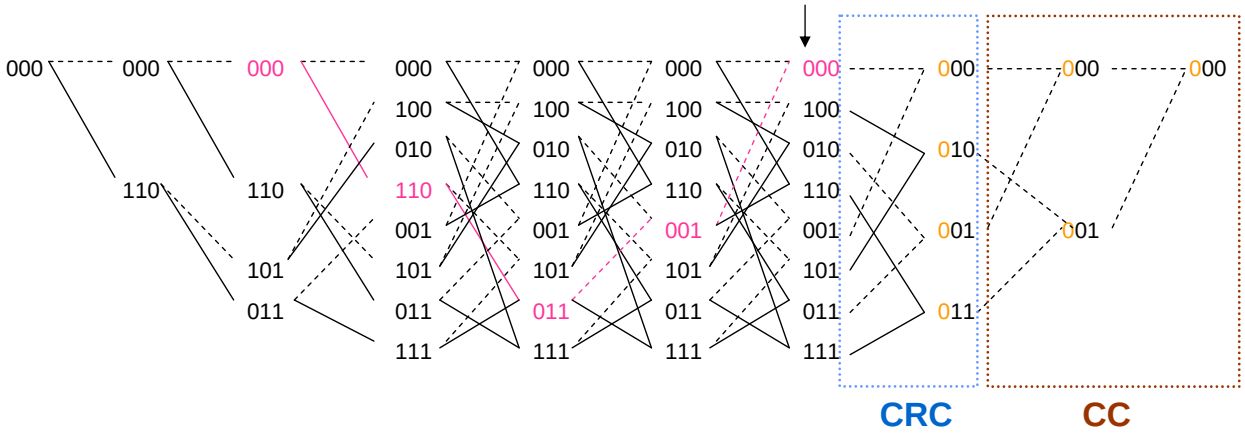

Figure 3.4: The complete combined trellis of CRC and convolutional codes

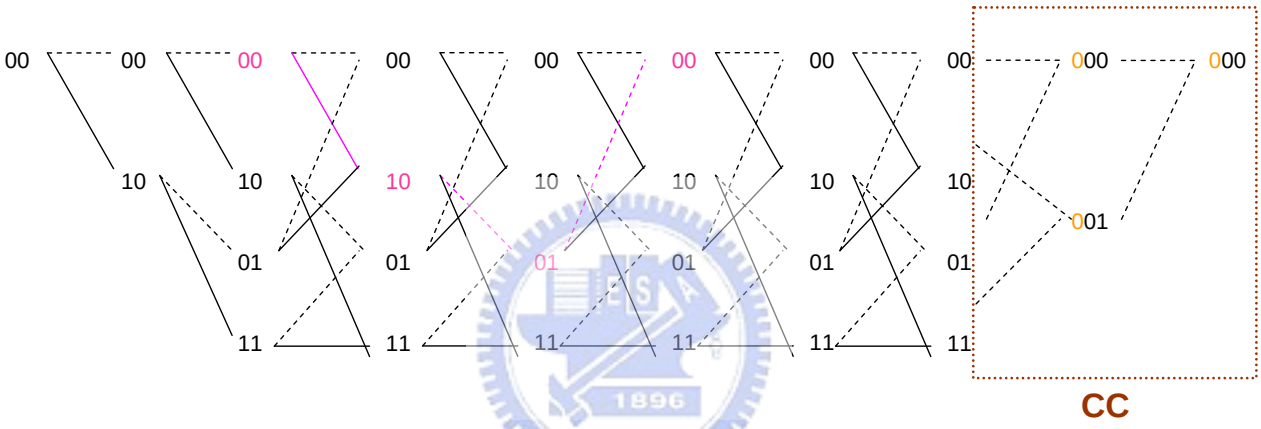

Figure 3.5: The conventional trellis of convolutional codes

from the conventional convolutional decoding trellis, there are three stages in the combined trellis: exploring stage, CRC merging stage and CC merging stage. Figure 3.4 presents a complete combined trellis of a $(2, 1, 2)$ convolutional codes , $\boldsymbol{g} = [101\ 111]_{\text{binary}}$ and CRC generator polynomial $g(x) = 1 + x$.

When the decoding process enters into the CRC merging stage, there will be only one corresponding input sequence for each state individually. Each input sequence is the current state of CRC, so the memory registers of CRC will be all zeros at the end of CRC merging stage. Afterward by putting a sequence of zeros into the memory register of CC, we obtain a codeword of both CRC and CC at the terminal of the combined trellis. It is particular

to mention that the combined trellis upgrades the free distance. The free distance of the combined trellis is six as illustrated in Figure 3.4 and is greater than the free distance of the conventional CC trellis which is only five. Because the additional constraint provided by CRC is added into the codeword searching, the free distance of combined trellis is identical to the one of the concatenated codes of CRC and CC.

3.3 A Modified Encoder of Combined CRC and Convolutional Codes: the Proposed Partition Method for Information Package

Although the combined trellis elevates the performance,there are two drawbacks of the proposed Decoding Algorithm. One is that the codewords found from the combined trellis will always pass the CRC check even the codeword is wrong. Another obstacle is the tremendous complexity caused by the combined states of CRC and convolutional codes. A generator polynomial of CRC with a high degree will make the complexity of the combined trellis extremely high. Consequently, we propose a modified encoder of combined CRC and convolutional codes which partitions the information sequence.

To alleviate the first drawback, we add only part of the CRC restriction into the decoding of combined trellis instead of execute the CRC constraint completely. The first attempt is to use several small CRC rather than the original long CRC to encode the information sequence. So, there are some CRCs for assisting error correcting and one CRC for error detecting and the total length of these CRCs is equivalent to the length of the original CRC. In Figure 3.6, the first codeword is the original CRC codeword with information length 114 and CRC-15. The second one is that we separate CRC-15 into CRC-8 which is the one combined with CC decoding and CRC-7 which is for error detection. And we divide CRC-15 into two CRC-4 for error correction and one CRC-7 for error detection in the third codeword. The encoding procedure for the second codeword is described in the following: we put the information sequence with length 114 into the CRC-8 encoder and afterward put these codeword with length 122 into CRC-7 encoder. Finally, regard the codeword with

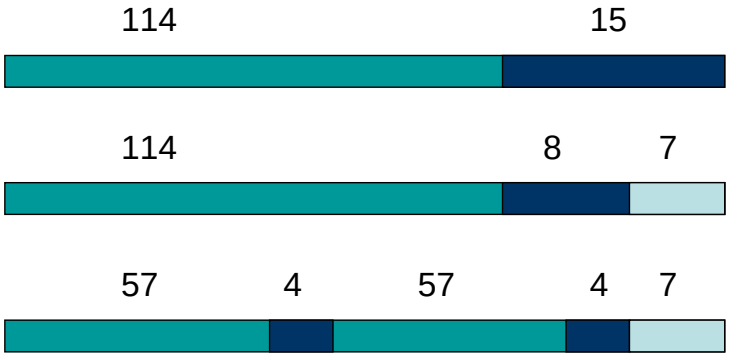

Figure 3.6: The CRC codewords with a modified CRC encoder

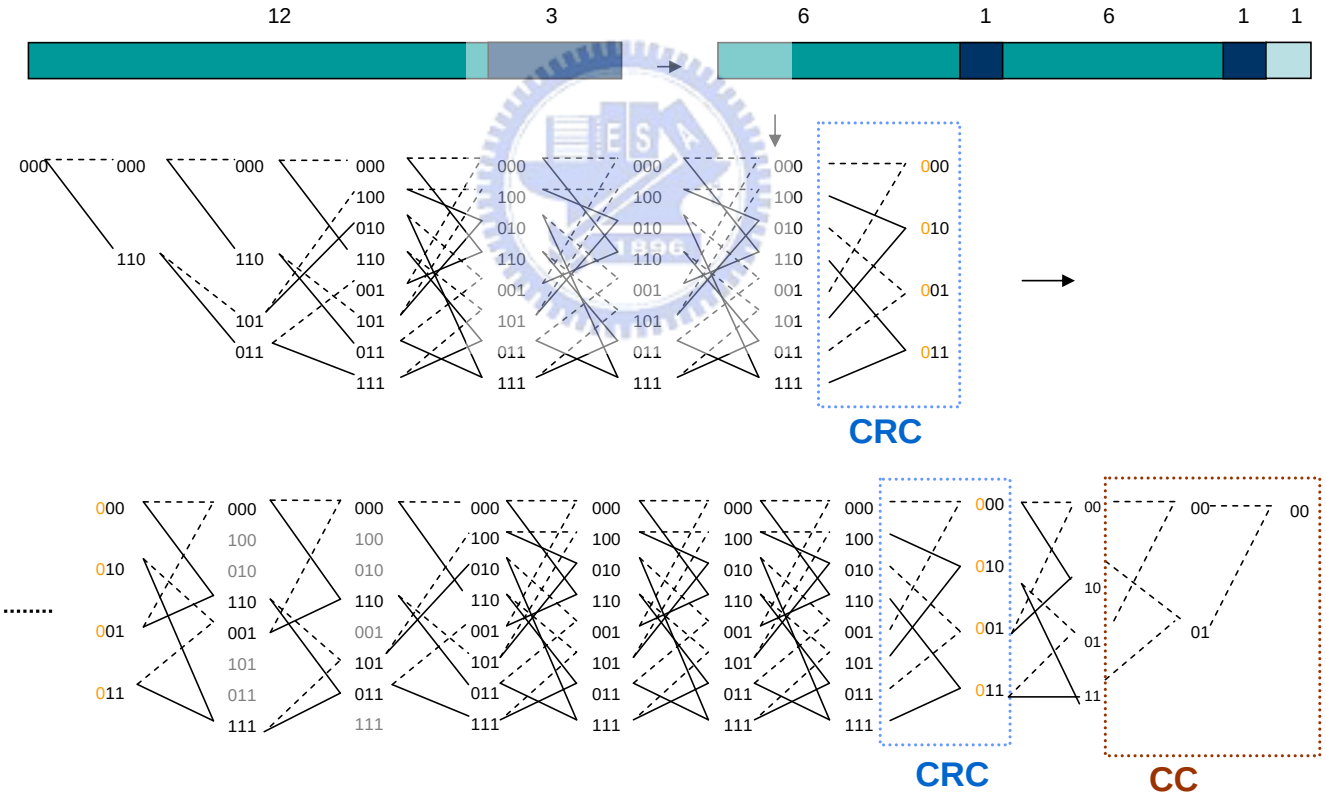

Figure 3.7: The realization of the decoding of combined trellis with a modified CRC encoder

length 129 as the input of the convolutional encoder. Hence, during the decoding we will reach a CRC codeword of CRC-8 after the CRC merge stage of the decoding of combined trellis and further explore the trellis to acquire the last 7 bits for detection. There is a simple example for comprehending the decoding process of the case of the second codeword in Figure 3.7. We partition CRC-3 into two CRC-1 for error correction and one CRC-1 for detection and use a $(2, 1, 2)$ convolutional codes, $\boldsymbol{g} = [101\ 111]_{\text{binary}}$, a generator polynomial $g(x) = 1 + x$ for CRC-1 and the length of CRC codeword (which is also the input sequence for CC encoder) is 15. Due to the varied CRC encoding scheme, there are two CRC merging stage at the position of the 7th bit and the 14th bit separately. Then we explore the trellis of CC without CRC at the 15th bit. The first 7 bits of the decoded sequence is a codeword of CRC-1, the 8th bit to the 15th bit is also a codeword of CRC-1 and the last 1 bit is for detecting the former 14 bits are correct or not. In this way, the shortcoming that the codewords found from the combined trellis will always pass the CRC check can be surely eradicated by this modified encoding scheme. However, we should notice the length of the information and partitioned CRCs when applying the partition method for the original CRC. If the length of the partitioned CRC for error detection is not long enough to detect errors in the received sequence, the undetected error rate will increase seriously. For example, if we partition a CRC-8 which concatenated with an information sequence of length 200 into one CRC-4 for error correction and one CRC-4 for error detection, CRC-4 is not powerful enough to detect errors an information sequence with a such long length. Therefore, the error detecting capability of the partitioned CRC should be evaluated conscientiously before partitioning.

Considering about the second drawback caused by original combined trellis, we can also alleviate the tremendous complexity by the modified encoder of combined CRC and convolutional codes. Return back to the example in 3.6, the decoding complexity of the original CRC-15 codeword in the combined trellis consisting of CRC-15 and CC is $2^{15+\text{CCmemory}} * 114$ which is greater than the complexity of the second codeword which is only $2^{8+\text{CCmemory}} * 114$ by using the modified combined trellis. Nevertheless, this method stipulate a new encoder

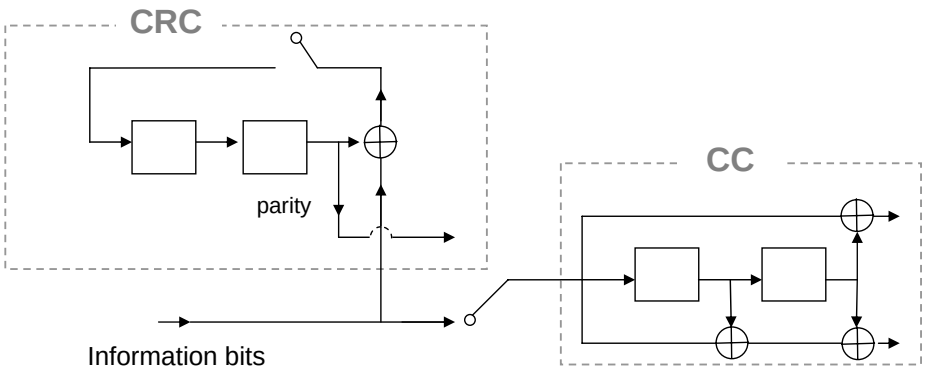

Figure 3.8: A rate $R = 1/2$ binary convolutional encoder with memory order $m = 2$ and a CRC with generator polynomial $g(x) = 1 + x^2$

which is not adequate to the modern transmission system. Our aspiration is to eliminate drawbacks of the combined trellis without altering the primary CRC encoding and improve the performance in the meanwhile.

3.4 Method of Forming the Reduced Trellis of Combined CRC and Convolutional Codes

On the condition of preserving the modern CRC encoding scheme, we should eliminate some pathes in the combined trellis to decrease the decoding complexity. To obtain a reduced version of the combined trellis, we attempt to partition the combined trellis into several sub-trellises first. Now consider a simple circumstance which uses a $(2, 1, 2)$ convolutional codes, $\boldsymbol{g} = [101\ 111]_{\text{binary}}$, a generator polynomial $g(x) = 1 + x^2$ for CRC-2. The encoding circuit and the combined trellis are shown in Figure 3.8 and Figure 3.9 separately.

The first observation is that the CRC state "00" and "01" will be the same CRC state '0' by ignoring one right bit of CRC state in the upper portion of the combined trellis and is illustrated in Figure 3.10. Thus, there are two pairs of paths which are drawn as green lines and blue lines for one CRC state. The second observation is that there are also two pairs of paths for specifical one combined state. In Figure 3.11, two pairs of paths enter into the state "010". Finally, the reduced trellis of combined trellis can be attained by ignoring one

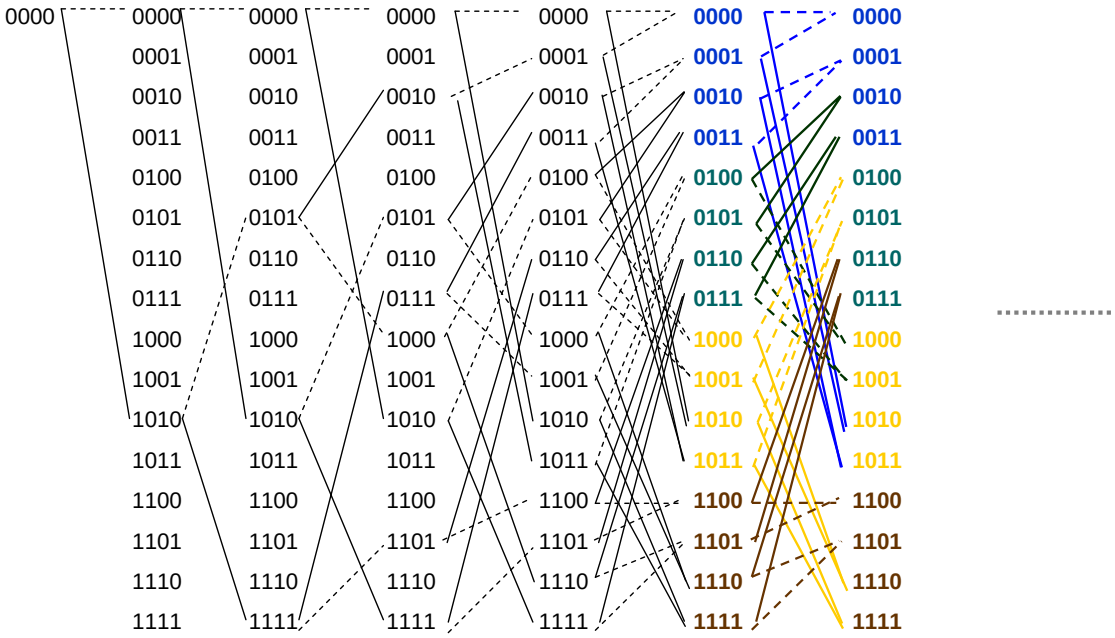

Figure 3.9: The combined trellis of CRC-2 and a $(2, 1, 2)$, $m = 2$ convolutional code. The left two bits are the state of CRC and the right two bits are the state of CC

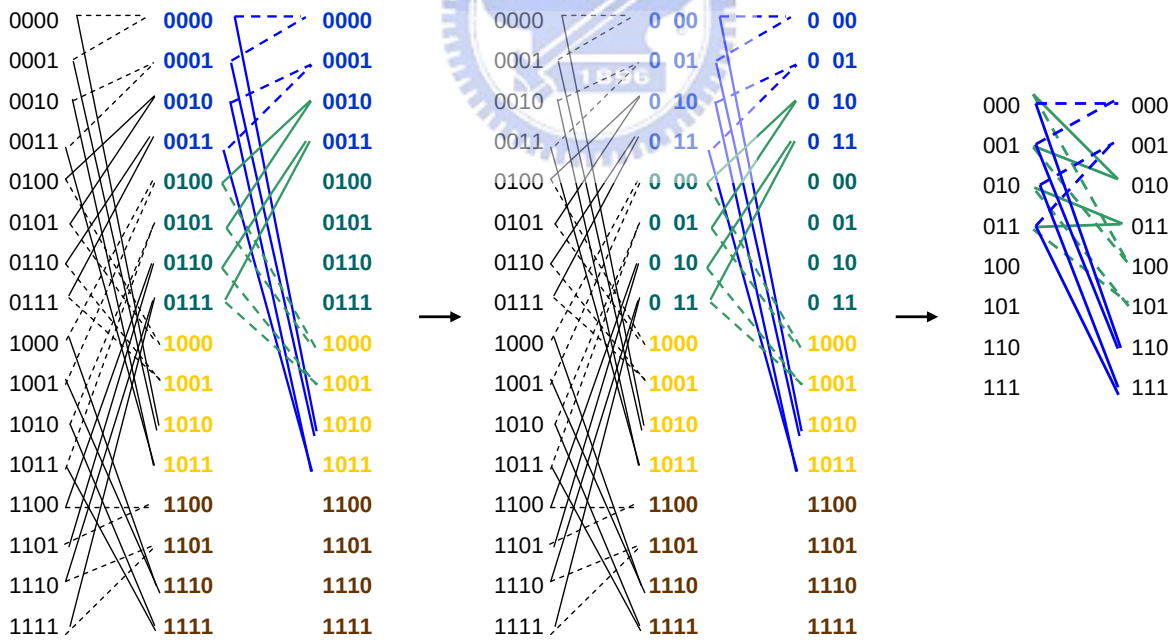

Figure 3.10: Ignore one right bit of CRC state in the upper portion of the combined trellis for Observation 1

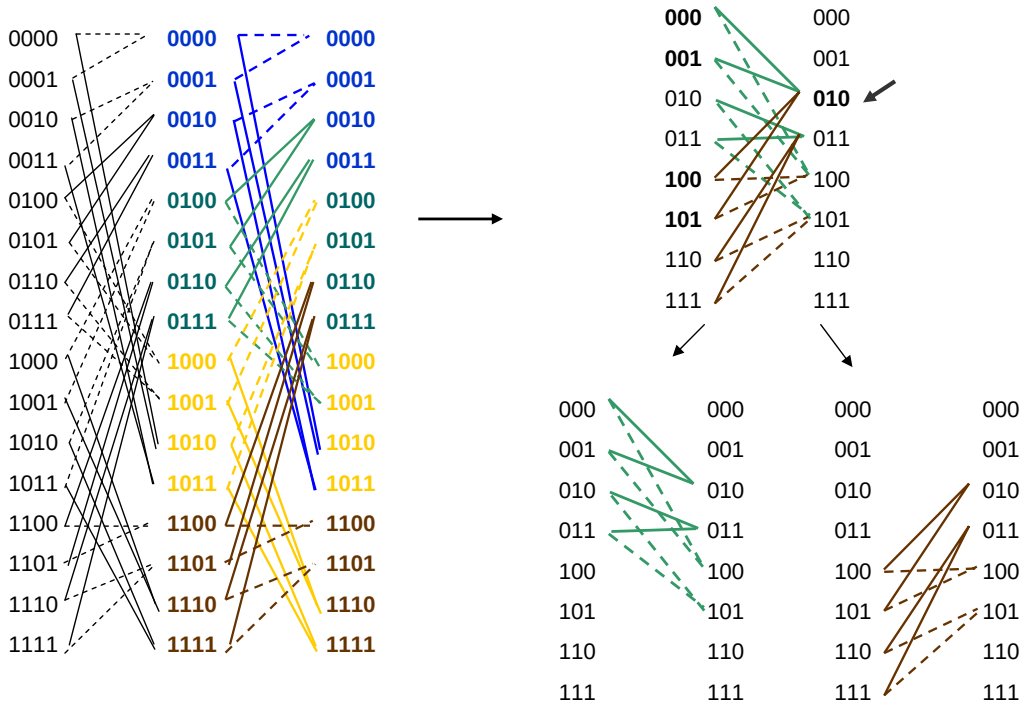

Figure 3.11: Ignore one right bit of CRC state in the upper portion of the combined trellis for Observation 2

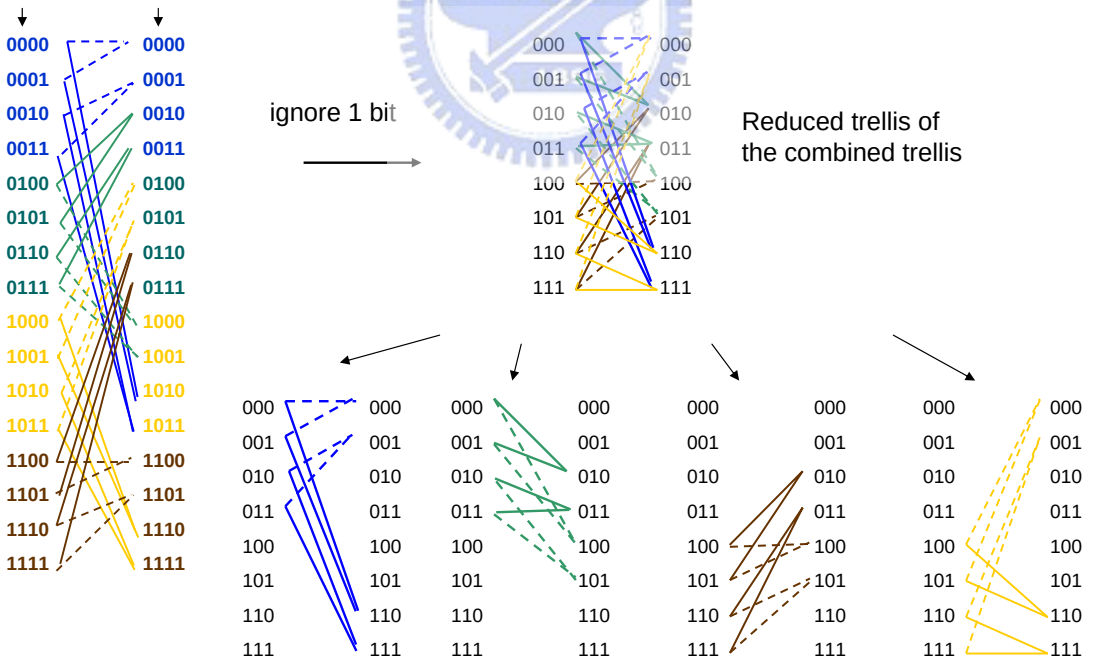

Figure 3.12: The reduced trellis of the combined trellis and the corresponding elements of the reduced trellis

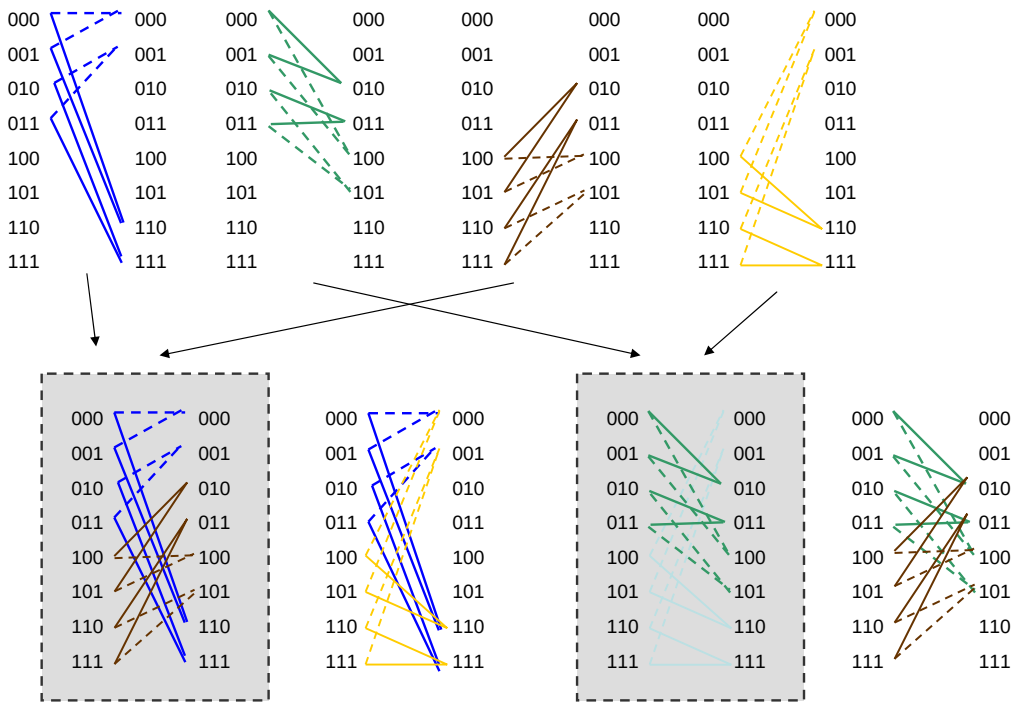

Figure 3.13: Combine the corresponding elements of the reduced trellis to several sub-trellises

state bit of CRC. Nevertheless, the generation of two choices when inputting one kind of information bit is the incoming obstacle caused by the reduced trellis. For instance, there are two directions to stride from the state "000" by inputting '0' where the first direction is to "000" and the second is to "100".

To enable the reduced trellis of the combined trellis for practical utilization, now we attempt to partition the reduced trellis into several sub-trellises. First, we separate the reduced trellis into four elements in 3.12 and combine two of them randomly to obtain four kinds of sub-trellises. Note that there are only two usable sub-trellises of this reduced trellis because the remains are not trellises of full states. In this way, we can force the combined trellis into two sub-trellises. If we select the first left sub-trellis in 3.13 for decoding, decoding on the combined trellis of CRC-1 and CC is equivalent to decoding on that sub-trellis in 3.13. Specifically, one of sub-trellises of the reduced trellis of the combined trellis for CRC-2 is identical to the combined trellis for CRC-1. Hence, it is reasonable to utilize the decoding

of concatenated CRC-N and CC by using the combined trellis for CRC-(N-j) instead of the combined trellis for CRC-N where j is the number of ignored CRC register bits and the connection of the CRC-(N-j) register are same as the connection of left-rest bits of CRC-N. Additional proofs for the equivalence between the combined trellis for CRC-(N-j) and the sub-trellis of the reduced trellis of the combined trellis for CRC-N will be derived in Appendix.

We are now prepared to decode by using the reduced version of the combined trellis which requires much fewer complexity than using the original combined trellis. Consider about the example of a $(2, 1, 6)$ convolutional codes, $\boldsymbol{g} = [163\ 135]_{\text{octal}}$, a generator polynomial $g(x) = 1 + x^2 + x^4 + x^6 + x^7 + x^8$ for CRC-8. If we ignore four bits of the register of CRC-8, that is to say, we use the combined trellis of a specific CRC-4 which is the left half part of the CRC-8 rather than the combined trellis of CRC-8 for decoding. Eventually, we identify a decoding scheme which is executed on the combined trellis and truly decrease the complexity without modifying the original CRC encoding process. However, the loss of performance caused by employing the sub-trellis of the combined trellis is inevitable on account of the elimination of paths in the initial combined trellis. Simulation results will exhibit substantial improvements on the conventional decoding schemes in Chapter 5.

Note that the code space of the codewords decoded from the sub-trellis of the reduced trellis constructed by the combined trellis contains the code space of the codewords decoded from the combined trellis. And the code space of the codewords decoded from the original CC trellis contains these two code spaces. So, the code decoded form the sub-trellis of the reduced trellis constructed by the combined trellis is a supercode of the code of the combined trellis and a subcode of the original code decoded from the trellis of convolutional code only.

3.5 Applications on List Decoding

The list Viterbi decoding algorithm (LVA) is a concept of searching for the L^{th} paths instead of one ML path for the sake of obtaining an acceptable path before the next retransmission.

It is verified in Chapter 5 that blending the LVA into our proposed decoding algorithm can achieve greater promotion than purely performing the LVA. In addition, Simulations of the proposed decoding schemes and several established decoding algorithms in retransmission system with Chase combining are also illustrated in Chapter 5.

Chapter 4

Simulation Results

In this thesis, we first utilize a $(2, 1, 6)$ convolutional codes, $\boldsymbol{g} = [163\ 135]_{\text{octal}}$ and a generator polynomial $g(x) = 1+x^2+x^4+x^6+x^7+x^8$ for CRC-8. Here, we remain the original CRC encoding scheme in this simulation. If we ignore four bits of the register of CRC-8, namely, we use the combined trellis of CRC-4 rather than the combined trellis of CRC-8 for decoding. Since the combined trellis of the specifical CRC-4 is one of the sub-trellis of the reduced trellis of the combined trellis of CRC-8, it is reasonable to employ the combined trellis of CRC-4 directly for decoding. Noted that the generator polynomial of the CRC-4 is $g(x) = 1+x^2+x^4$ which is the left half part of the genertor polynomial of the CRC-8 we use. It is explicit that the proposed algorithm is 0.7dB superior to e conventional convolutional decoding which only employs CC for decoding in Figure 4.1. The improvement on bit error rate is 0.6dB greater than the conventional one as illustrated in Figure 4.2. The combined CRC-8 ML is the line which is the ML performance of the concatenated code comprised of CRC and CC. Now consider about the complexity of the proposed algorithm. In Table 4.1, the proposed algorithm has ten times the number of computation times of the conventional decoding algorithm in low SNR (1-1.5dB). Nevertheless, the number of computation times of these two schemes are similar in medium SNR (3-4dB) and the proposed algorithm is even greater than the conventional one in high SNR (>4dB) due to the CRC merging stage in the proposed algorithm. More specifically, the CRC merging stage in the end of the reduced version of the combined trellis can extract some unneeded steps because of the elimination

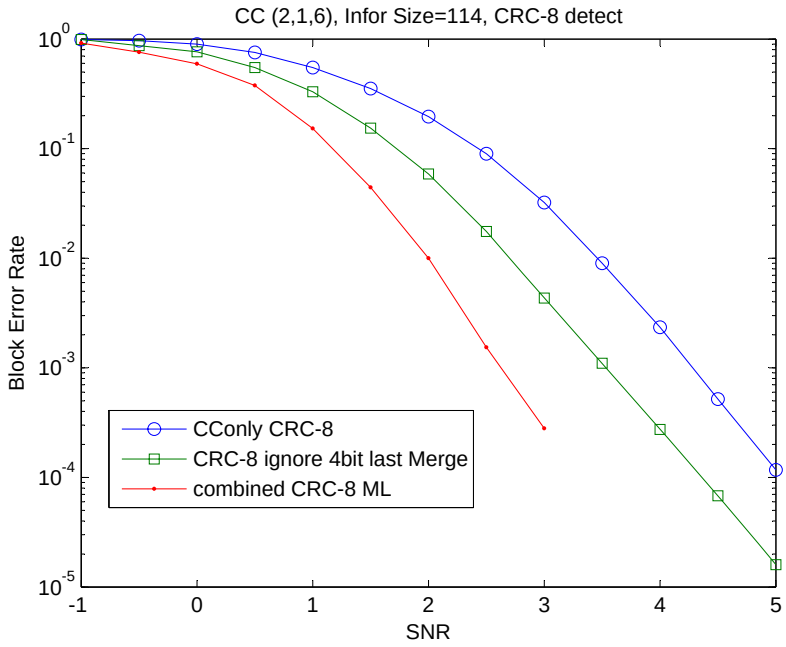

Figure 4.1: CC (2,1,6), information size=114, CRC-8 detect

of paths caused by CRC constrain. Therefore, the number of computation times can be decreased in the latter portion of the combined trellis when the noise is slight.

The comparison between the former idea of partition the CRC bits and the proposed decoding algorithm is shown in Figure 4.6. We utilize a $(2, 1, 6)$ convolutional codes, $\boldsymbol{g} = [163\ 135]_{\text{octal}}$ and a generator polynomial $g(x) = 1+x^3+x^4+x^7+x^8+x^{10}+x^{14}+x^{15}$ for CRC-15. In Figure 4.3, there are three CRC encoding schemes that we apply in this comparison: the first is the conventional CRC encoding for CRC-15. In the second, we partition CRC-15 into the error correction section which is CRC-8 and the error detection section which is CRC-7. Partition CRC-15 into two CRC-4 for error correction and one CRC-7 for error detection in the third encoding scheme. The particular description for the second and third methods is exhibited in Chapter 4. Note that the generator polynomial of CRC-7 and CRC-8 is $g(x) = 1 + x^3 + x^7$ and $g(x) = 1 + x^2 + x^4 + x^6 + x^7 + x^8$ respectively. As illustrated in Figure 4.6, the performance of the proposed algorithm of ignoring 7 bits for CRC-15 is similar to the second scheme in 4.3. However, the proposed algorithm preserves the original CRC encoding scheme but still improves the performance. In addition, on account of using

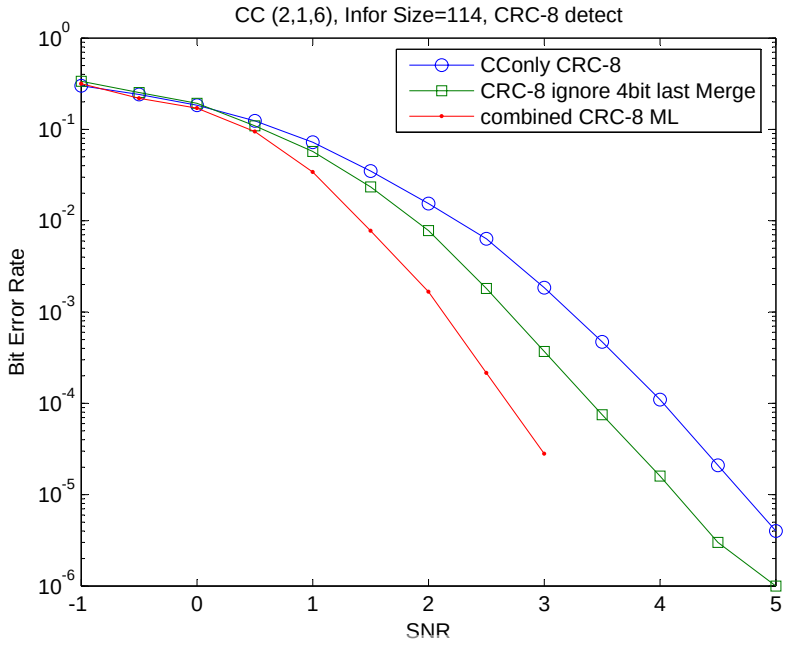

Figure 4.2: CC (2,1,6), information size=114, CRC-8 detect

	CRC-8+CC ignore 4bits		**CC only**	
SNR	Average	Maximum	Average	Maximum
1	151222	223044	12702	14974
1.5	116265	222900	11789	14974
2	75132	222614	10691	14974
2.5	40269	223068	9020	14974
3	17216	210186	7108	14974
3.5	6641	182972	5007	14974
4	2262	202466	3068	14974
4.5	933	189084	1652	14910
5	478	112218	844	13762

Table 4.1: complexity

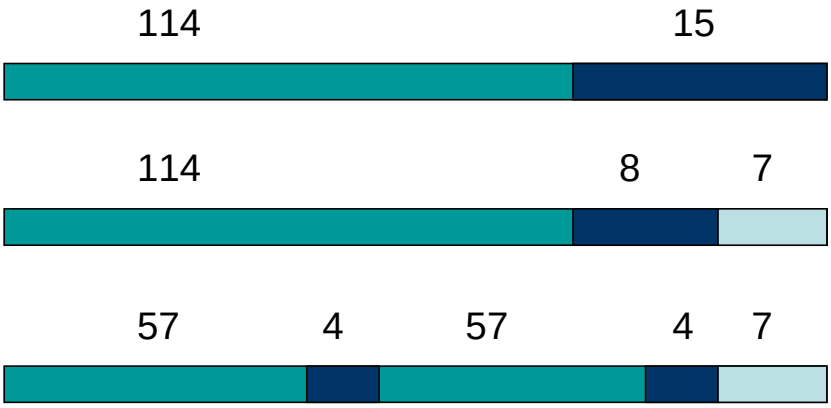

Figure 4.3: Three distinct encoding schemes for CRC codeword

the identical register of CRC-4 to form the combined trellis for the proposed algorithm and the partitioned combined trellis for the second scheme in 4.3, the complexity of the second scheme is slightly fewer than the the complexity of the proposed algorithm because there are two merging stage in the trellis for second scheme. Also, the second scheme approach a superior error rate to the proposed decoding algorithm which uses unaltered encoding scheme. We conclude that the performance can be upgraded by change the CRC encoding scheme.

Now we compare the LVA to our proposed decoding algorithm in Figure 4.5. The block error rate of the proposed decoding algorithm is equivalent to the error rate of the LVA with L=10 in SNR 0-2.5dB and is between the error rate of LVA with L=2 and L=10 in SNR >3dB.

In the retransmission system which permitting once retransmission and employing Chase combining, the proposed decoding algorithm is still superior to the conventional decoding algorithm about 0.7-0.8dB. Note that the codeword decoded from the combined trellis of CRC-8 and CC will always pass the CRC check, so the combined CRC-8 ML has no simulations for retransmission.

The throughput of the proposed decoding algorithm is similar to the throughput of the LVA with L=10 and is greater than the conventional decoding algorithm.

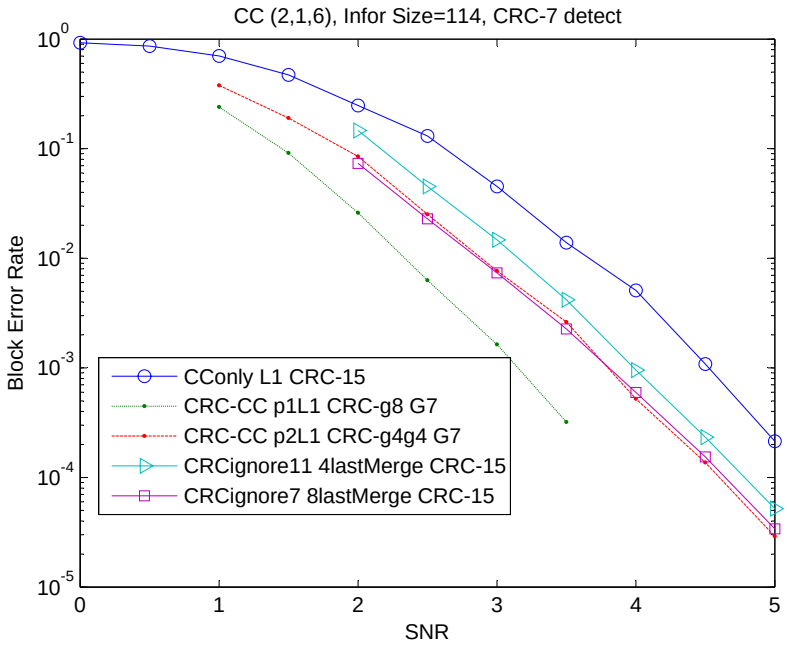

Figure 4.4: CC (2,1,6), information size=114, CRC-15 detect

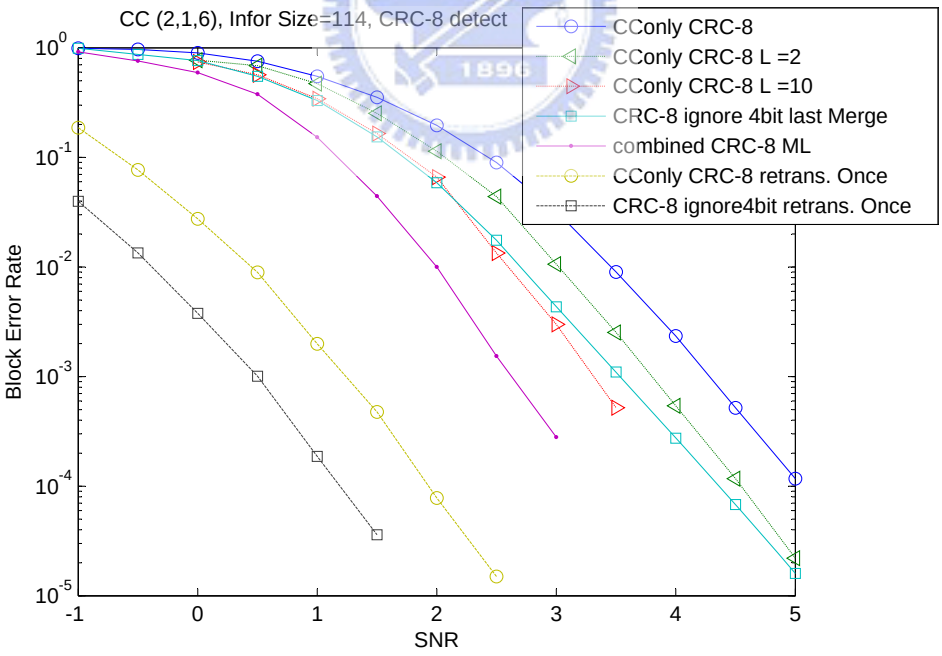

Figure 4.5: CC (2,1,6), information size=114, CRC-8 detect

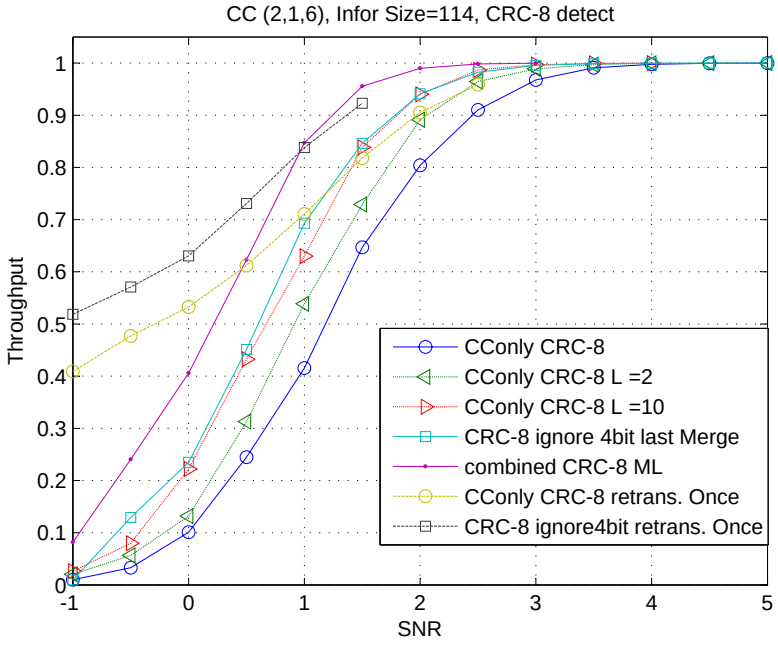

Figure 4.6: Throughput: CC (2,1,6), information size=114, CRC-15 detect

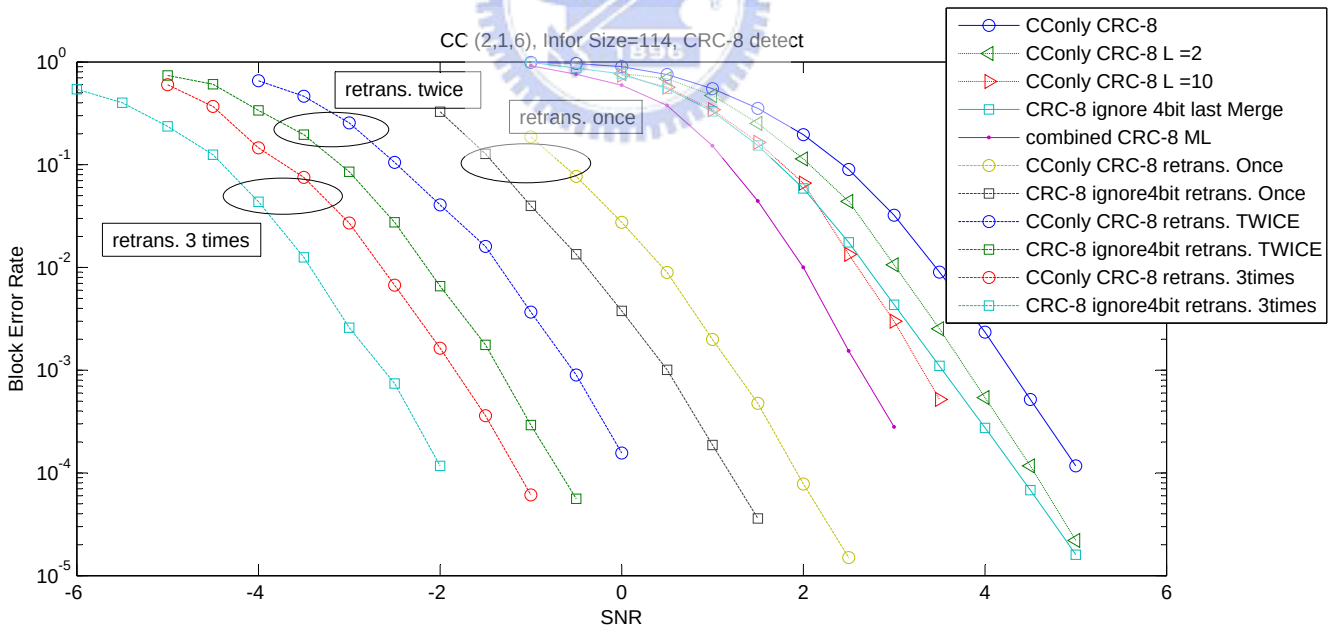

Figure 4.7: CC (2,1,6), information size=114, CRC-8 detect

Chapter 5

Conclusions

The conventional decoding scheme in hybrid ARQ utilizes only convolutional codes to decode the received sequence and considers CRC as just an error detecting code. However, the proposed decoding algorithm combine the error correcting ability of CRC with the convolutional decoding to alleviate the error rate caused by convolutional decoder. Due to the effect of the CRC constrain which added in the combined trellis, the codeword decoded from the proposed decoding algorithm has more opportunities to pass the error detection.

There is a decoding algorithm, list Viterbi algorithm, which can also decrease the error rate generated from the CC decoder by attaining L^{th} best paths in 1st transmission instead of one ML path. The LVA tries to search another acceptable decoded codeword to pass the error detection before next retransmission while the ML codeword failed. The combination of the LVA and our proposed algorithm can achieve substantial improvement in the performance. Instead of employ the combined trellis of a CRC with a large degree, we can use the reduced version of the combined trellis and the LVA to upgrade the performance without tremendous computation times.

Appendix A

Proof of the Reduced Trellis

We try to prove the Lemma.

lemma 1 *The reduced trellis of CRC-N contains the combined trellis of CRC-(N-J) , where j is the number of ignored CRC register bits and the connection of CRC-(N-J) registers are same as the connection of left-rest bits of CRC-N.*

Bibliography

[1] S. Lin and D. J. Costello, *Error control coding: Fundamentals and Applications, second edition.* Englewood Cliffs, NJ: Prentice-Hall, 2004.

[2] K. Brayer, "Error Control Techniques Using Binary Symbol Burst Codes," *IEEE Trans. Commun.*, vol. COM-16, pp. 199-214, Apr. 1968.

[3] D. M. Mandelbaum, "Adaptive-Feedback Coding Scheme Using Incremental Redundancy," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 388-389, May 1974.

[4] S. Kallel, "Complementary punctured convolutional (CPC) codes and their applications," *IEEE Trans. Commun.*, vol. 43, no. 6, Jun. 1995.

[5] J. M. Shea, "Reliability-based hybrid ARQ," *IEEE Electron. Lett.*, vol. 38, pp. 644-645, Jun. 2002.

[6] J. Hagenauer, "Rate-compatible punctured convolutional codes (RCPC codes) and their applications," *IEEE Trans. Commun.*, vol. 36, pp. 389-400, Apr. 1988.

[7] R. Fano, "A heuristic discussion of probabilistic decoding," *IEEE Trans. Inform. Theory*, vol. IT-9, no. 2, pp. 64-73, Apr. 1963.

[8] R. Fano, "A Fast Sequential Decoding Algorithm Using a Stack," *IBMJ. Res. Dev.*, vol. 13, pp. 675-685, Nov. 1969.

[9] Y. S. Han, P.-N. Chen, and H.-B. Wu, "A maximum-likelihood soft-decision sequential decoding algorithm for binary convolutional codes," *IEEE Trans. Commun.*, vol. 50, no. 2, pp. 173-178, Feb. 2002.

[10] Shin-Lin Shieh, "Near Maximum-Likelihood Sequential-Search Decoding Algorithms for Binary Convolutional Codes" PhD Dissertation, Hsinchu, Taiwan, R.O.C.: National Chiao Tung University, Hsinchu, Taiwan, R.O.C., 2008.

[11] N. Seshadri and C.-E. W. Sundberg, "List Viterbi decoding algorithms with applications," *IEEE Trans. Commun.*, vol. 42, no. 2/3/4, pp. 311-323, Feb./Mar./Apr. 1994.

[12] Q. Spencer,"Method of correcting message errors using cyclic redundancy checks," U.S. Patent 11 836 381, Aug. 9, 2007.