基於低密度奇偶查核碼的分散式訊源編碼機制

研究生：王韋超　　指導教授：張文輝 博士

國立交通大學

電信工程研究所

中文摘要

本論文針對二位元相關訊源，利用低密度奇偶查核碼具體實現 Slepian-Wolf 理論在分散式訊源編碼的應用。訊源相關模型以兩種虛擬通道模型表示，分別是二位元對稱通道與 Gilbert 通道。針對訊源編碼輸出的校驗子經由雜訊通道傳輸的問題，我們提出基於渦輪碼原則推導的疊代訊源通道解碼。我們將考慮兩種通道碼，低密度奇偶查核碼應用在分散式訊源編碼的校驗子生成以達到資料壓縮效果，而迴旋碼則用於提昇壓縮資料對抗通道雜訊的能力。模擬結果顯示基於低密度奇偶查核碼的分散式訊源編碼機制，配合疊代訊源通道解碼演算法，可同時兼顧高壓縮率及強健性能。

關鍵字：低密度奇偶查核碼, 分散式訊源編碼, Gilbert 通道, 疊代訊源通道解碼

A Study of Low-Density Parity Check Code for Distributed Source Coding

Student: Wei-Chao Wang Advisor: Dr. Wen-Whei Chang

Institute of Communications Engineering,

National Chiao Tung University

Hsinchu, Taiwan

Abstract

In this thesis, we study the use of low-density parity check (LDPC) codes for distributed source coding (DSC) of correlated binary sources. The Slepian-Wolf theorem states that there is no less in rate to compress two correlated sources using separate encoding, provided that the decoding is done jointly and the source correlation is available to both the encoder and decoder. Source correlation is modeled by two types of virtual channels: binary symmetric channel (BSC) and Gilbert channel. Also proposed is an iterative source-channel decoding (ISCD) algorithm for dealing with the Slepian-Wolf problem over noisy channel. An outer LDPC code is used to perform DSC, and an inner convolution code is used for enhancing the error protecting capability of the compressed data. Simulation results indicate the combined use of ISCD and LDPC-based DSC can provide error robustness as well as channel efficiency.

Keywords: low-density parity check (LDPC), distributed source coding (DSC), iterative source-channel decoding (ISCD) .

誌謝

光陰匆匆，時光飛逝，轉眼間兩年研究學涯即將結束。這篇誌謝文，起頭比論文難寫，想說的話與感謝之言實在難以單篇涵蓋，無奈規範至上，只好舉杯望月寄情竹湖，不免有念天地之悠悠獨愴然而涕下之感。為了逃避工作，當完兵後幸運地重回交大電信，一路走來相當順遂，一人寢宿舍，二校區往返機車無礙，三生有幸受到張文輝老師教導，每日朝九晚五，生活相當愜意，至此難免離情依依，不捨之心難以言喻。

首先，感謝老師兩年來的指導，在研究與精神上給予相當多的幫助。此外，謝謝陳信宏老師、張寶基老師以及王忠炫老師，口試當天分享許多經驗，讓學生獲益良多。另外感謝 97 同梯的吳兄鴻材，總在困難處提點並熱心參與系壘為系爭光。實驗室同屆成員，步步維盈，與眾不凡的程式能力協助突破了長久研究瓶頸，除此也常關心生活起居，三餐問候並討論生活議題。帥哥鴻竣，熱血敢衝的有為青年，為實驗室帶來許多歡笑活力，兩年生活因為如此有許多想不到的回憶。GY 志賢，同為南台灣出身，直爽不諱的台語常令人憶起家鄉，能與之同寢實在難得。實驗室可愛的學弟妹，怡華嘩啦，平衡了陽盛陰衰的 lab，也分享許多美食給大家，雖然為了生活常不在實驗室，***晉瑋也是***。同儕之間，至中與 James，感謝你們讓羽球團有經費來源，讓寢室常有人在，一人早一人晚，充滿家的感覺。感謝小不點詩倩，讓我認識新竹美食，許多奇異的經驗有著無法忘懷的瘋狂痕跡。謝謝你們，真的！

最後感謝我的家人，有你們的陪伴，讓我能心無旁騖完成學業，往人生下一階段邁進。

目錄

表目錄

圖目錄

第1章　　緒論

1.1　研究動機與方向

Gallager 在 1960 年的博士論文[1]中首次提出的低密度奇偶查核(low-density parity check, LDPC)碼，為一線性區塊通道碼(linear block channel code)，且對於資料傳輸與通道儲存提供了接近容量(near-capacity)的效能。由於 VLSI 實作上的難度，早期研究只有 Zyablov 與 Pinsker[2]，Margulis[3]，和 Tanner[4]。其中 Tanner 針對低密度奇偶查核矩陣建構了雙邊架構圖，包含變數點與查核點以及邊際線，對於之後疊代訊息解碼演算法有深遠影響。近期則有 Spielman[5]與 Mackay[6][7]的研究。此外，具有長區塊的 LDPC 碼透過疊代解碼過程展現出接近容量的效能[8]，也是一重要的研究議題。基於 LDPC 矩陣的建構影響疊代複雜度及其效能的事實，因此 Hu 等研究者[9][10]，提出累進邊際成長演算法(progressive-edge growth, PEG)，主要概念在於建構長周長的 LDPC 矩陣，且建構過程中確保新增的邊際不會影響當前的周長。

被遺忘 40 餘年的 LDPC 之所以能夠重見天日，其實有其歷史催生因素存在。首先，於 1962 年，當時錯誤修正碼的主流用途幾乎都是聲音的傳送，其編碼長度 n 比較短，無法充分發揮 LDPC 碼的優點，其效能也差於早兩年開發的里德所羅門碼(Reed Solomon Code)。如今，時空逆轉，多數應用環境的編碼長度 n 從數百到數萬，尤其是高速資料通訊。例如無線的資料通訊應用，編碼長度 n 約在 200～5,000 個位元之間；若是光纖的中樞網路，其編碼長度 n 就可以高達數萬個位元。過去被視為標準錯誤修正碼的里德所羅門碼，其典型的編碼長度 n 約是在 255。但是，不可否認的，LDPC 碼當編碼長度 n 越龐大時，符號的計算量也跟著大幅度增加，可能是以 n 的平方或是 n 的 3 次方遞增。不過，打破此僵局的 Flarion Technologies 找到減少計算量的方法，讓計算量與 n 成比例，此問題便被克服。

第二個讓 LDPC 碼浴火重生的觸媒就是半導體製程的進步。誠如先前所提，LDPC 碼能夠揮灑的空間是編碼長度 n 較大的場合，也就意味著處理電路的規模也會相對成比例增加。晶片面積也跟著變大，成本自然上揚，而解決此問題的絕佳良方就是 90 奈米的半導體製程。美商朗訊(Lucent)曾針對光通訊用途，採用 0.16 毫米製程試做一個 1Gbps 傳送速度的 LSI，內部光是編碼電路的門閘數量就高達 175 萬個，LSI 的大小為 7.2 毫米。另一家公司 System LSI 也針對 HDD 訊號處理電路用途，採用 0.13 微米製程，試做一個

1Gbps 傳送速度的 LSI，內部電路的門閘數量也是接近 100 萬個，LSI 的大小為 3 毫米。依據 LSI 設計廠商的估算，若是採用 90 奈米設計準則，LSI 大小約可以降到 2 毫米，邁入實用化便來日不遠。

至於分散式訊源編碼(distributed source coding, DSC)的相關研究，則起源於 1970 年代 Slepian 和 Wolf 針對兩相關訊源提出的無失真編碼理論[11]。主要訴求是，兩個有相關性的訊源，可藉由資源共享的合併編碼(joint encoding)模式降低其理論熵值(entropy)。更重要的是，即使在各自獨立編碼(separate encoding)的情況下，仍能以合併解碼(joint decoding)模式取得相同的理論熵值。為了具體實現這個壓縮理論值，2003 年 Pradhan 和 Ramachandran 首次運用了通道編碼理論的碼分級校驗子(syndrome)觀念[12]。他們將兩個訊源所屬的位元序列劃分成定長的區塊，其中一個訊源進行通道編碼處理後傳送其碼組(coset)的校驗子；而另一個完全不壓縮的訊源則視為接收端的邊訊息(side information)。其關鍵在於事先建立一個足以描述兩訊源之間相關性的數學模型，並將其視之為一虛擬的傳輸錯誤通道。而在接收端，則針對邊訊息進行通道解碼處理，最後在校驗子所屬的碼組中找出訊源的最佳預估值。分散式訊源編碼系統的壓縮率及合成品質，取決於兩項關鍵元件：訊源相關模型及校驗子生成機制。

在[13]中，LDPC 碼首次被應用到 Slepian-Wolf 理論基礎，利用低密度查核矩陣進行訊源壓縮並產生其校驗子，且採用對稱性分散式訊源架構進行分析。而 Xiong 等學者，進一步將 LDPC 碼運用到非對稱的 DSC 上[14]。假設通道無傳輸錯誤下，以虛擬的二位元對稱通道(binary symmetric channel, BSC)模擬兩訊源間的相關性，將校驗子訊息納入疊代解碼過程，其效能遠高於渦輪碼的最佳值。Eckford 等學者者提出 LDPC 碼在記憶性通道(Gilbert-Elliott channel)的運用[15]，更能符合實際通道環境的模擬。我們將以此為參考，建構一記憶性虛擬通道(Gilbert channel)，並具體實現基於 LDPC 碼的 DSC 架構，同時附加考慮的狀態點資訊也有助於疊代解碼正確還原訊源。

針對雜訊環境下的 Slepian-Wolf 壓縮，Hu 與 Li 於[16]中提供模擬架構，分開訊源與通道編碼可使設計簡單且易於掌控，合併解碼採用渦輪原則提昇整體效能。至於疊代訊源通道解碼(iterative source-channel decoding, ISCD)，主要探討 Slepian-Wolf 壓縮輸出的校驗子經由雜訊通道傳輸時的挑戰。我們將考慮兩種通道碼，LDPC 碼應用在分散式訊源編碼的校驗子生成，由於校驗子有誤，解碼採用擴展式 LDPC 矩陣進行疊代訊息演算法，而迴旋碼(convolution code)則提昇壓縮資料對抗通道雜訊的能力，解碼以 BCJR 演算法為主，並求出通道額外訊息做為訊源解碼端的軟性輸入信息。透過兩解碼器之間

額外訊息的交換，以提昇整體解碼效能。

1.2 章節概要

第 2 章介紹 LDPC 碼的理論基礎，包含矩陣表示與 Tanner 圖，以及通道解碼所使用的演算法，PEG 建構方式也一併討論。第 3 章介紹 DSC 原理與基於 LDPC 碼的分散式訊源編碼機制，另外分別檢視虛擬通道為 BSC 或 Gilbert 通道的解碼流程。第 4 章則介紹疊代訊源通道解碼，包含額外訊息的運作與 BCJR 演算法，及針對校驗子傳輸錯誤而推導的擴展式 LDPC 碼。第 5 章進行實驗模擬與分析。第 6 章則為結論與未來展望。

第2章　基於 LDPC 矩陣的通道碼

此章節介紹低密度奇偶查核碼(low density parity check code,LDPC)的理論基礎以及演算流程，首先以奇偶查核矩陣和 Tanner 圖來分析基本架構。接著根據 Tanner 圖的程度分布模式，歸類出兩種不同型態的 LDPC 碼，分別為規則與不規則形態。我們也參考歷年文獻提供不同的矩陣建構方法，包含最初 Gallager 提供的方式與隨機產生法，以及目前常用且證實為最佳的累進邊際成長演算法(Progressive Edge-Growth)。

2.1　LDPC 矩陣

LDPC 碼，為一線性區塊通道碼(linear block channel code)，且對於資料傳輸與通道儲存提供了接近容量(near-capacity)的效能。由 Gallager 在 1960 年的博士論文[1]中首次被提出，卻一直到近十年來才被熱烈討論。期間 Tanner 重新泛論 LDPC 碼並提出對應的圖碼表示架構，現今被通稱為 Tanner 圖(Tanner graphs)。90 年代中期，MacKay，Luby 與 Neal 等學者，進一步探討線性區塊碼(linear block code)有關稀少(sparse)或低密度(low-density)奇偶檢驗矩陣的優點。

由於 LDPC 碼產生容易，例如隨機建構方式，且解碼演算複雜度與碼的區塊長度呈線性關係，再加上其可觀的效能，使得 LDPC 碼有許多不同的應用。研究方向有相關效能的分析，疊代解碼演算的收斂，Slepian-Wolf 壓縮的使用，以及通道編碼與資料壓縮的交互運作。

LDPC 碼可以在非二進位的字元上運作，但我們在此只考慮二進位的 LDPC 碼。由於 LDPC 碼為線性區塊通道碼的一種，因此將其描述成一 k 維度的子空間集合 C，且每一合法碼字在二進位域(binary field)上具有 n 個位元。進一步表示，基底向量 $B=\{g_0,g_1,...,g_{k-1}\}$ 組成碼空間 C，且對於所有合法碼字 c 可以表示成 $c=u_0g_0+u_1g_1+...+u_{k-1}g_{k-1}$，或以矩陣表示為 $c=uG$，其中訊息位元組 $u=[u_0,u_1,...,u_{k-1}]$，G 為 $k*n$ 維度的生成矩陣(generator matrix)，其列向量為 $\{g_i\}$。生成矩陣 G 的零空間 $C^{\perp}$ 為 $(n-k)\times n$ 維度，且由基底向量 $B^{\perp}=\{h_0,h_1,...,h_{n-k-1}\}$ 組成，所以對於碼空間 C 任何一合法碼字 c 皆滿足 $ch_i^T=0$。而整體矩陣可寫成 $cH^T=0$，其中 H 就是我們所提到 $(n-k)*n$ 維度的奇偶查核矩陣(parity check matrix)，其列向量為 $\{h_i\}$。

一個(n,k)的二進位線性區塊通道碼，k個訊息位元透過生成矩陣G產生n個位元的合法碼字c，所以對應碼率為k/n。總共的合法碼字有2^k個，而且任何兩合法碼字相加會得到另一個合法碼字。LDPC code 為二進位線性區塊碼，且其查核矩陣H所擁有位元 1 的個數比例很少，這種現象稱低密度(low-density)或稀疏(sparse)。另外，位元 1 個數的比例(相較於 0)並無一定值來決定低密度與否，只是高效能的查核矩陣，對應的 1 出現比例都很少。而由於 1 出現的密度很低，所以大幅降低解碼的複雜度。

2.2 Tanner 圖示法

Tanner 針對 LDPC 碼提出一有效的圖解表示架構，也就是所稱的雙邊架構圖(bipartite graph)，現今比較常被稱做 Tanner 圖。首先，將低密度查核矩陣的節點(nodes)分成兩種類型，而邊際(edge)只會連接於不同類型的節點上，不允許有同類型節點互連的現象。此兩種類型的節點分別為：

(1) 合法碼字中n個位元對應的節點，又稱位元節點(bit nodes)或變數點(variable nodes)。

(2) $m(\geq n-k)$個查核節點，稱作查核點(check nodes)或作用點(function nodes)。

在此我們統一用$v-nodes$和$f-nodes$分別代表變數點和查核點。

在查核矩陣H中，我們依序連接h_{ji}為 1 的查核點j與變數點i，依循此方法完成所有連線即繪成完整的 Tanner 圖。舉例來說，(n,k)為$(10,5)$的線性區塊碼，且$w_c=2$，$w_r=4$表示成查核矩陣如下：

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

對應的 Tanner 圖如圖 2.1 所示。

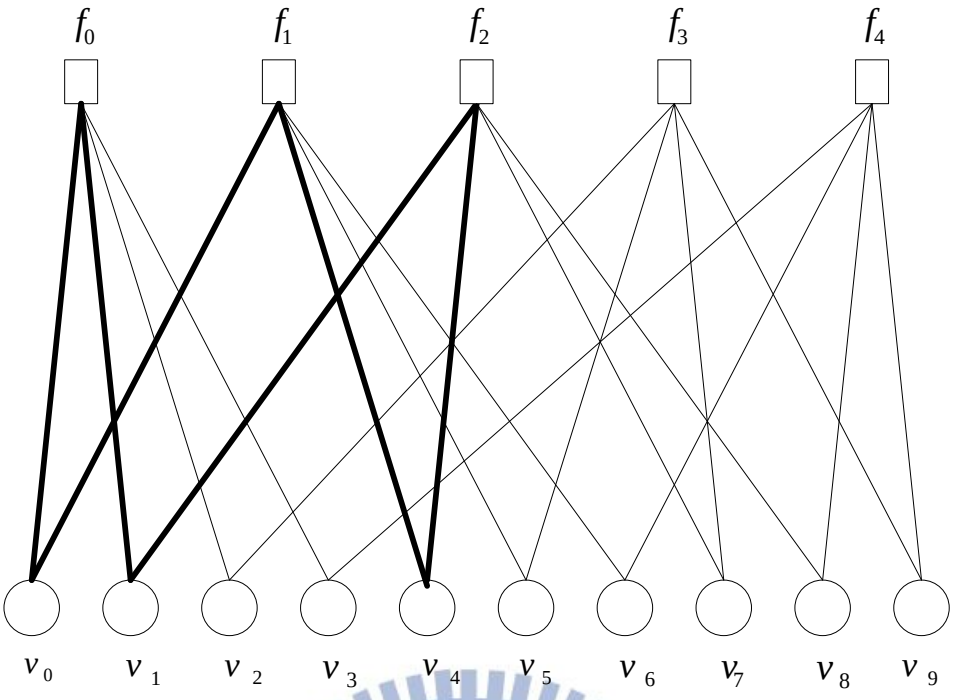


圖 2.1　　Tanner 圖例

從列角度來看，變數點v_0, v_1, v_2和v_3連接到查核點f_0，主要是查核矩陣H中第零列$h_{00}=h_{01}=h_{02}=h_{03}=1$(其餘皆為 0)，接著依序觀察查核點$f_1, f_2, f_3, f_4$所各自對應的列向量，如此即可繪出 Tanner 圖。我們也可以從行角度做起，變數點c_0連接到查核點f_0與f_1，因為矩陣H中第零行$h_{00}=h_{10}=1$。由於是規則的查核矩陣，所以每一個變數點有 2 個邊際連接(degree w_c =2)，每一個查核點有 4 個邊際連接(degree w_r =4)。

在 Tanner 圖上，從任一節點出發且不重複路徑，最後返回同一點所走過的邊際總數或長度l，我們稱為一個循環(cycle)。而在所有循環之中，具有最小長度的，稱為此 Tanner 圖的周長(girth)。顯而易見，最小的循環長度為 4，反應在H矩陣中，位元 1 出現在矩形的四個角落。循環長度大小對於低密度奇偶查核矩陣的疊代訊息解碼有深遠的影響，越短的循環其解碼效果越差，原因在於訊息傳遞解碼演算(message passing algorithm)，短循環增加了節點間訊息的相關性。為了避免循環長度過短(例如$l=4$)，建構一個好的低密度奇偶查核矩陣，是一門重要的研究議題。

規則(regular)的 LDPC 碼，是指一$m*n$維度($m \geq n-k$)的查核矩陣，每一行皆具有w_c個位元 1，每一列皆具有w_r個位元 1。此外，任兩列或兩行在同一個位置上出現 1 的次數不會超過一次，此現象影響低密度奇偶查核矩陣的效能好壞。從另一方面來說，每一個碼字會被查核w_c次；每一組查核方程式會包含w_r個碼字。定義上會給予密度值$\gamma = w_c/m = w_r/n$，所以低密度也可以指$w_c \ll m \& w_r \ll n$，查核矩陣總共有$w_c \cdot n = w_r \cdot m$

個 1，且由於 $m \geq n-k$，碼率 $R = k/n \geq 1-(w_c/w_r)$，所以 $w_c \ll w_r$。習慣上我們以 (w_c, w_r) 表示規則的低密度奇偶查核矩陣碼，例如 $(3,6)$ regular LDPC code。

反之，若一低密度奇偶查核矩陣中，各行或各列 1 出現的個數不為定值，我們稱非規則(irregular)的 LDPC 碼。而不同行與列的重值則以多項式表示為 $\lambda(x)$ 與 $\rho(x)$，稱做程度分布多項式(degree distribution polynomials)。

$$\lambda(x) = \sum_{d=1}^{d_v} \lambda_d x^{d-1}, \tag{2}$$

λ_d 為對應在 Tanner 圖上，所有連接於變數點的邊際數目(總和為 d)百分比，而 d_v 為 $v-nodes$ 邊際數最大值。同理，

$$\rho(x) = \sum_{d=1}^{d_c} \rho_d x^{d-1}, \tag{3}$$

ρ_d 為對應在 Tanner 圖上，所有連接於查核點的邊際數目(總和為 d)百分比，而 d_c 為 $f-nodes$ 邊際數最大值。

2.3 LDPC 矩陣的建構模式

2.3.1 Gallager 法

Gallager 提供一簡單實用的方式，令 k 為正整數且大於 1，H_1 為 $k \times kw_r$ 維度的查核矩陣，其內部每一列有 w_r 個 1，每一行 1 的個數則只有一個。分配方式以列來排序，第 $i(i=1,2,...,k)$ 列將 w_r 個 1 分到第 $(i-1)w_r+1 \sim iw_r$ 行之內。一旦建立好 H_1，剩餘的 $H_2 ... H_{w_c}$ 為 H_1 各自做適當的行置換。查核矩陣 H 因此具有 $kw_c * kw_r$ 維度分布，表示如下：

$$H = \begin{bmatrix} H_1 \\ H_2 \\ \vdots \\ H_{w_c} \end{bmatrix} \tag{4}$$

由 Gallager 法建構的查核矩陣，任兩列或兩行同位置 1 出現的次數絕對不會超過一次，且為規則 (w_c, w_r)，但不保證循環長度一定大於 4。由於矩陣 H 中，位元 1 的總數為 $kw_r w_c$，因此密度值 $\gamma = kw_r w_c / k^2 w_r w_c = 1/k$，所以選定適當大小 k，使得查核矩陣 H 具

有低密度，即可完成低密度奇偶查核矩陣的建立，式子(5)為一例子。

$$H_{15\times 20}=\left[\begin{array}{c}
\begin{matrix}
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1
\end{matrix}\\
\hline
\begin{matrix}
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0\\
0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&1&0&0&0\\
0&0&1&0&0&0&1&0&0&0&0&0&0&1&0&0&0&1&0&0\\
0&0&0&1&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0\\
0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1
\end{matrix}\\
\hline
\begin{matrix}
1&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&1&0&0\\
0&1&0&0&0&0&1&0&0&0&1&0&0&0&0&1&0&0&0&0\\
0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&1&0\\
0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&1&0&0&0\\
0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1
\end{matrix}
\end{array}\right]\begin{matrix}H_{1_{5\times 4}}\\ \\ \\ H_{2_{5\times 4}}\\ \\ \\ H_{3_{5\times 4}}\end{matrix} \tag{5}$$

另外值得一提也是 LDPC 矩陣的優點，就是當區塊長度n趨近無窮大，其合法碼字間的最短距離$d_{\min}$ (minimum distance)，也會與n成比例關係。

$$\limsup_{n\to\infty}\frac{k}{n}>0\ ,\limsup_{n\to\infty}\frac{d_{\min}}{n}>0 \tag{6}$$

這對於解碼效果有極大的影響，試舉$(n,k,m,d_{\min})=(2,1,6,10)$的迴旋碼為例，當$n$值不斷攀升，$d_{\min}$始終維持定值，一旦錯誤個數使$d_{\min}\geq 2t_{error}+1$不成立，解碼端將無法正確還原訊息。

2.3.2 隨機產生法

除了 Gallager 建議的造碼方式，也有人提出隨機的 LDPC 碼(random LDPC code)。為了符合其碼率為k/n，必須先設計一查核矩陣且具有適當的行重w_c以及相當的列數$(m\geq n-k)$，而為了使列重w_r為定值以及$m=n-k$，須滿足$w_c\times n=w_r\times(n-k)$。因此當$n$能被$n-k$整除，$w_r=w_c n/(n-k)$，對應產生規則的低密度查核矩陣；如果$n$無法被$n-k$除盡，則改寫數學式為$w_c\times n=w_r(n-k)+b=w_r(n-k-b)+b(w_r+1)$，其中$b$為定值，所以在矩陣中會有兩個列重，分別為$w_r$和$w_r+1$，即是非規則的 LDPC 矩陣。對此，假設已找到部分查核矩陣如下：

$$H_{i-1}=\{h_1,h_2,...,h_{i-1}\} \tag{7}$$

隨機產生的h_i必須為$(n-k)$*1維度，且具有w_c的行重。另外，還需經過三個步驟才能將h_i歸入我們所要的矩陣集合裡。

(1) h_i不得與H_{i-1}內元素重複，反之，重新產生比對。

(2) h_i比對所有在H_{i-1}集合內的元素，同一位置 1 出現的次數不得超過一次，反之，重新產生比對。

(3) 將h_i加入集合，如果此暫定集合前b個列重$\le w_r+1$且後$(n-k-b)$個列重$\le w_r$，則h_i為所要的新元素，而新集合定為H_i，反之，重新產生比對。

2.3.3 累進邊際成長演算法

本論文採用的 LDPC 矩陣，是根據 MacKay 發表於研究網站[17]的模擬程式軟體所建構。此軟體主要是根據累進邊際成長演算法(Progressive Edge-Growth,PEG)寫成，過程可參考文獻[9]，在此我們簡介演算內容。

首先定義演算過程中使用的參數。(I,E)代表著 Tanner 圖，$I=I_v\cup I_f$為節點集合，包含變數點群$I_v\in\{v_0,v_1,...,v_{n-1}\}$與查核點群$I_f\in\{f_0,f_1,...,f_{m-1}\}$。$E$為所有邊際的集合，在查核矩陣$H$中，若$h_{ji}=1$，代表對應的邊際$(v_i,f_j)\in E$，且$0\le i\le n-1, 0\le j\le m-1$。另外定義程度分布序列$D_v=\{d_{v_0},d_{v_1},...,d_{v_{n-1}}\}$，其中$d_{v_i}$表示變數點$v_i$所擁有的邊際數或程度，且$d_{v_0}\le d_{v_1}\le...\le d_{v_{n-1}}$；同理，$D_f=\{d_{f_0},d_{f_1},...,d_{f_{m-1}}\}$。而$E_{v_i}^k$為變數點$v_i$的第$k$個邊際，而且$0\le k\le d_{v_i}-1$。最後，按照圖 2.2，定義$N_{v_i}^l$是由變數點$v_i$出發且深度(depth)為$l$的延展式樹狀圖，收集圖上所有查核點並集合統稱為變數點v_i的鄰居。其中樹狀圖的連接方式，一開始從變數點v_i連接所有邊際到查核點上，而被連接的查核點也會有對應邊際連接到其他變數點，依此類推，其他變數點再各自連線延展，就可以完成樹狀圖。至於樹狀圖中未被連接的查核點群，另外定義成$\overline{N}_{v_i}^l$，亦即$N_{v_i}^l\cup\overline{N}_{v_i}^l=I_f$。此外，定義所有通過變數點$v_i$的循環之最小值為局部周長$g_{v_i}$，因此周長為各變數點局部周長的最小值$g=\min\{g_{v_0},g_{v_0},...,g_{v_{n-1}}\}$。

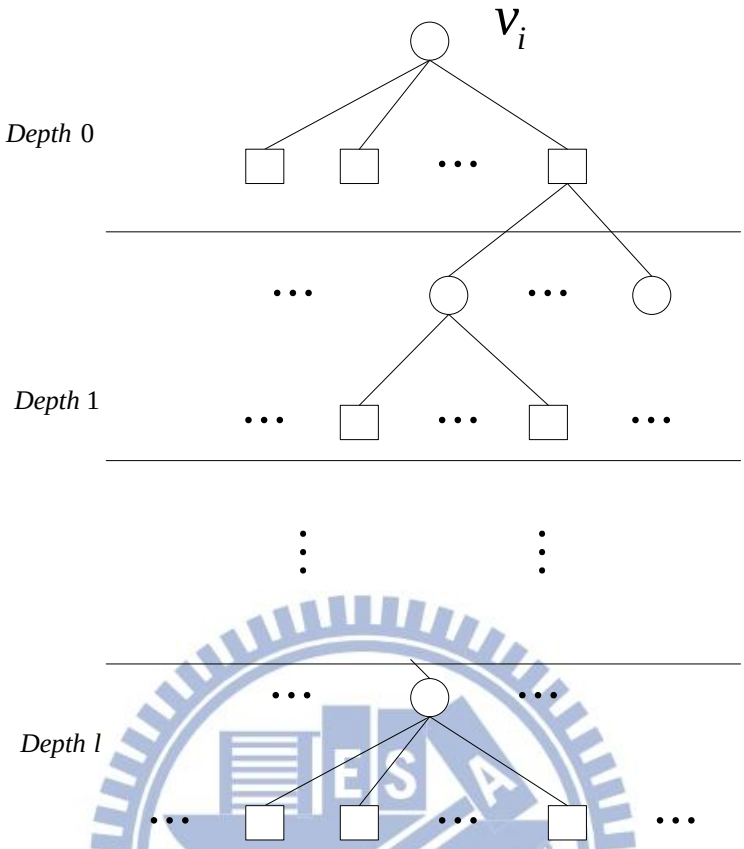


圖 2.2　　變數點的樹狀圖

要直接建立一個具有最大周長的 Tanner 圖，整體上有難度。因此，累進邊際成長演算提供了一個次佳的做法，概念在於讓每個變數點的局部周長為最大值，即使有新的邊際連接於此。換句話說，變數點新增的邊際所產生的循環，並不會影響之前局部周長的最大值。舉例來說，假定已經建立邊際集合 $E_{v_0} \cup E_{v_1} \cup ... \cup E_{v_{i-}}$ ，且暫定周長 $g^t = \min\{g_{v_0}, g_{v_0}, .., g_{v_i}\}$，若再新增邊際群 E_{v_i} 於目前 Tanner 圖上，產生的循環必須不影響到 g^t ，因此，我們必須使局部周長 g_{v_i} 盡可能最大化。

所以，在新增邊際群 E_{v_i} 的過程中，是邊際接著邊際完成，而且每條通過變數點 v_i 的邊際都確保其最短循環維持最大值且不影響周長的變動。每當新增一邊際於變數點 v_i 時，會先將 v_i 做深度 l 的樹狀圖延展，並使得 $\overline{N}_{v_i}^{l} \neq \varnothing$ 且 $\overline{N}_{v_i}^{l+1} = \varnothing$ ，或者 $N_{v_i}^{l}$ 中集合元素總數停止增長且小於 m 。之後將新的邊際接連到 $\overline{N}_{v_i}^{l}$ 裡程度分布最小的查核點，如此可以保證通過此邊際的最短循環至少大於 $2(l+2)$ 。總結演算流程如下：

for $i = 0$ **to** $n-1$ **do**

begin

for $k=0$ **to** $d_{v_i}-1$ **do**

begin

if $k=0$

建立邊際$E_{v_i}^0 \leftarrow \left(f_j, v_i\right)$，其中$E_{v_i}^0$為變數點$v_i$第 0 個邊際接於查核點$f_j$。且此查核點為目前邊際集合$E_{v_0} \bigcup E_{v_1} \bigcup ... \bigcup E_{v_{i-1}}$對應的 Tanner 圖下，查核程度(check degree)最小的查核點。

else

擴展變數點v_i的樹狀圖且深度為l，並使得$\overline{N}_{v_i}^l \neq \varnothing$且$\overline{N}_{v_i}^{l+1} = \varnothing$，或者$N_{v_i}^l$中集合元素總數停止增長且小於$m$。建立$E_{v_i}^k \leftarrow \left(f_j, v_i\right)$，其中$E_{v_i}^k$為變數點$v_i$第$k$個邊際接於查核點$f_j$。而選取的查核點為$\overline{N}_{v_i}^l$集合裡程度分布最小的節點。

end

end

新邊際連接的查核點，是從$\overline{N}_{v_i}^l$中選取的，雖然以程度最小的節點為優先挑選，但是容易遇到多重選擇問題，尤其是起初建立 Tanner 架構圖的過程，同樣程度分布的查核點有許多。所以一般採用兩種方式，一為隨機選取，另一則是按照編號挑選。

2.4 LDPC 通道碼的解碼演算法

LDPC 碼的疊代解碼過程，是基於渦輪原則推導而得。如圖 2.3 所示，渦輪原則的四大重點分別為：

(1) 成員區塊間的連接(Concatenation of component blocks)。在 LDPC 矩陣中，變數點被查核點群查核，而查核點檢驗變數點群的正確性，兩群節點在 Tanner 圖上相互連接。
(2) 交換器(Interleaver)。相接的成員區塊間必須使用交換器打亂其訊息的相關性，而 LDPC 矩陣本身即是隨機建構，可視為一交換器。
(3) 軟性輸入與輸出(soft-input-soft-output,SISO)的解碼過程。
(4) 額外訊息的交換(Exchange of extrinsic information)。

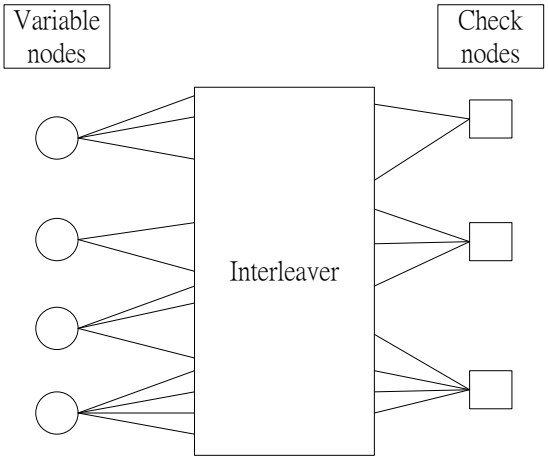


圖 2.3　　LDPC 碼的渦輪架構

訊息傳遞演算法(message passing algorithm)，是利用 Tanner 圖進行其疊代解碼，而每一次疊代會在變數點與查核點之間進行額外訊息(extrinsic information)的交換，傳遞路徑則為對應的邊際(edge)。邊際上的訊息依照傳遞方向可分為兩類，$q_{i,j}(x)$ 與 $r_{j,i}(x)$。訊息從變數點 x_i 出發傳遞到查核點 f_j 定義為 $q_{i,j}(x)$，反之，$r_{j,i}(x)$ 為查核點 f_j 出發到變數點 x_i 的訊息。舉例來說，前半部疊代如圖 2.4 所示，變數點 x_0 處理所有來自查核點(除了 f_2)的輸入訊息 $\{r_{0,0}, r_{1,0}\}$ 以及通道訊息(來自 y_0)，並輸出 $q_{0,2}$ 給查核點 f_2。特別強調的是，變數點 x_0 排除來自查核點 f_2 的輸入訊息，是為了避免查核點 f_2 重複利用已知的訊息，而造成不可預期的解碼效果，所以只有「額外」訊息在路徑上做資料交換。

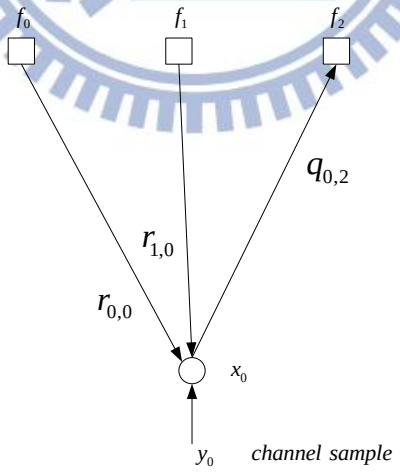


圖 2.4　　從變數點到查核點的前半部疊代

後半部疊代如圖 2.5 所示，查核點 f_0 傳遞輸出訊息 $r_{0,4}$ 給變數點 x_4，是針對 Tanner 圖上所有與之相連變數點(除了 x_4)的輸入訊息 $\{q_{0,0}, q_{1,0}, q_{2,0}\}$，處理後將額外訊息輸出給 x_4。疊代解碼不停重複直到事先給定的次數上限，或滿足某些條件才會終止。例如，

$xH^T = 0$ 常被用於程式迴圈內。

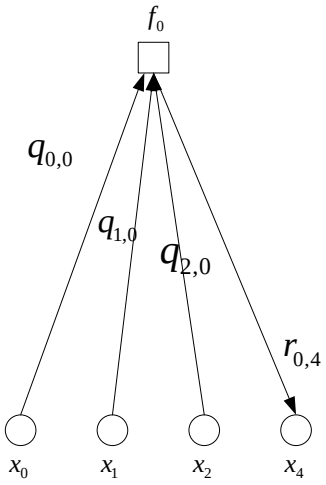


圖 2.5　　從查核點到變數點的後半部疊代

有了訊息疊代概念之後，接著解說訊息 $q_{i,j}(x)$ 與 $r_{j,i}(x)$ 代表的含意。$q_{i,j}(x)$ 指的是給定通道 y_i 以及所有來自查核點(除了 f_j)的額外訊息下，碼字中第 i 個位元值為 0 或 1 的機率，數學表示成 $\Pr(x_i = 0 \text{ or } 1 | \textit{input message})$。而 $r_{j,i}(x)$ 定義為在碼字第 i 個位元值等於 0 或 1 搭配其他相連變數點的額外訊息下，查核方程式 f_j 滿足的機率。數學表示 $\Pr(\textit{check equation } f_j \textit{ is satisfied} | \textit{input message})$。

2.4.1　機率域的加乘演算解碼器(Probability-Domain SPA Decoder)

為了方便推導演算，首先定義相關符號，如下表所示。

V_j = {連接查核點 f_j 的變數點群}

$V_j \setminus i$ = {連接查核點 f_j 的變數點群} \ {變數點 x_i}

C_i = {連接變數點 x_i 的查核點群}

$C_i \setminus j$ = {連接變數點 x_i 的查核點群} \ {查核點 f_j}

$M_v(\sim i)$ = {來自節點 x_i 以外所有變數點的訊息}

$M_c(\sim j)$ = {來自節點 f_j 以外所有查核點的訊息}

$P_i = \Pr(x_i = 1 | y_i)$

S_i = 查核方程式(包含 x_i)被滿足的事件

$q_{ij}(b) = \Pr(x_i = b \mid S_i, y_i, M_c(\sim j))$, 其中b∈{0,1}。

$r_{ji}(b) = \Pr(\text{查核方程式} f_j \text{被滿足} \mid x_i = b, M_v(\sim i))$, 其中b∈{0,1}。

若訊息以機率模式傳遞，透過上述定義表示 $q_{ij}(0)$ 如下：

$$q_{ij}(0) = \Pr(x_i = 0 \mid y_i, S_i, M_c(\sim j)) \tag{8}$$

$$= (1-P_i)\Pr(S_i \mid x_i = 0, y_i, M_c(\sim j)) / \Pr(S_i) \tag{9}$$

$$= K_{ij}(1-P_i)\prod_{j' \in C_i \setminus j} r_{j'i}(0) \tag{10}$$

其中我們使用兩次貝式定理，以及訊息間無相關假設。同理可推，

$$q_{ij}(1) \quad = K_{ij}P_i \prod_{j' \in C_i \setminus j} r_{j'i}(1) \tag{11}$$

其中常數 K_{ij} 的選定，是為了保證 $q_{ij}(0) + q_{ij}(1) = 1$ 成立。

接下來為了推導 $r_{ji}(b)$，我們必須運用到定理 1 的結論。

定理 1. (Gallanger): 假設有 M 個不相關的二進位位元 a_i，且位元為 1 的機率 $\Pr(a_i = 1) = p_i$，則位元串 $\{a_i\}_{i=1}^{M}$ 擁有偶數個 1 的機率為

$$\frac{1}{2} + \frac{1}{2}\prod_{l=i}^{M}(1-2p_i) \tag{12}$$

透過上述結果，我們將 $p_i \leftrightarrow q_{ij}(1)$，重新套用並改寫成下式。

$$r_{ji}(0) = \frac{1}{2} + \frac{1}{2}\prod_{i' \in V_j \setminus i}(1-2q_{i'j}(1)) \tag{13}$$

因此，當 x_i 為 0 時，其他碼字位元 $\{x_{i'} : i' \in V_j \setminus i\}$ 必須總含偶數個 1，以便滿足查核方程式 f_j。另外 $r_{ji}(1) = 1 - r_{ji}(0)$。

訊息傳遞演算的目標是計算碼字各位元的後驗機率(a posteriori probability 或 APP)，所以初始值設定為 $q_{ij}(b) = \Pr(x_i = b \mid y_i)$，換句話說，$q_{ij}(1) = P_i$ 和 $q_{ij}(0) = 1 - P_i$。而 y_i 代表通道實際接收值，以下針對兩種常見通道模型進行分析。

1. 二位元對稱通道(binary symmetric channel,BSC)：$y_i \in \{0,1\}$ 且通道錯誤率 $p = \Pr(y_i = \bar{b} \mid x_i = b)$，假設 $\Pr(x_i = 1) = \Pr(x_i = 0) = 1/2$，APP 初始值設定如下：

$$\Pr\left(x_i = b \mid y_i\right) = \begin{cases} 1-p, \text{當 } y_i = b \\ p, \text{當 } y_i = \bar{b} \end{cases} \tag{14}$$

2. 二位元輸入的白色高斯雜訊通道(binary input-additive white Gaussian noise channel, BI-AWGNC)：$x_i = +1(-1)$，$y_i = x_i + n_i$ 且雜訊 n_i 為 $\left(0, \sigma^2\right)$ 無相關性高斯分布。所以 APP 初始值設定如下：

$$\Pr\left(x_i = x \mid y_i\right) = \left[1 + \exp\left(-2y_i x / \sigma^2\right)\right]^{-1} \tag{15}$$

綜合以上結果，初始值的設定，訊息 $q_{ij}\left(x\right)$ 和 $r_{ji}\left(x\right)$ 的關係，變數點與查核點內部資料的運作，以及疊代終止條件。我們對操作在機率域的訊息疊代解碼(加乘演算)做總結。

(步驟 1)初始化：

根據 LDPC 矩陣畫出對應的 Tanner 圖後，將所有邊際上訊息歸零，並設定碼字各位元的對應初始值 $q_{ij}\left(b\right) = \Pr\left(x_i = b \mid y_i\right)$。例如在 BSC 通道假設下，$q_{ij}\left(1\right) = P_i$ 和 $q_{ij}\left(0\right) = 1 - P_i$。

(步驟 2)更新查核點到變數點的額外訊息：

$$r_{ji}(0) = \frac{1}{2} + \frac{1}{2}\prod_{i' \in V_j \setminus i}(1 - 2q_{i'j}(1)) \tag{16}$$

$$r_{ji}(1) = 1 - r_{ji}(0) \tag{17}$$

(步驟 3)更新變數點到查核點的額外訊息：

$$q_{ij}(0) = K_{ij}\left(1 - P_i\right)\prod_{j' \in C_i \setminus j} r_{j'i}(0) \tag{18}$$

$$q_{ij}(1) = K_{ij}P_i\prod_{j' \in C_i \setminus j} r_{j'i}(1) \tag{19}$$

而 K_{ij} 的選定，是為了保證 $q_{ij}(0) + q_{ij}(1) = 1$ 成立。

(步驟 4)分別對碼字各位元 i 計算後驗機率：

$$Q_i(0) = K_i\left(1 - P_i\right)\prod_{j \in C_i} r_{ji}(0) \tag{20}$$

$$Q_i(1) = K_i P_i\prod_{j \in C_i} r_{ji}(1) \tag{21}$$

而 K_i 的選定，是為了保證 $Q_i(0) + Q_i(1) = 1$ 成立。

(步驟 5)硬性決定解碼位元值，並判定疊代終止與否。

$$X_i = \begin{cases} 0, \ if \ \ Q_i(0) \geq 0.5; \\ 1, \qquad \text{otherwise} \end{cases} \tag{22}$$

假定疊代次數達到最大值或所有查核方程式被滿足，我們會終止程式迴圈。反之，重複步驟 2。

2.4.2 對數域的加乘演算解碼器(Log-Domain SPA Decoder)

就 Viterbi 與 BCJR 演算法而言，更常使用對數域的訊息傳遞過程，原因在於機率域相關的乘法運算容易造成數值不穩定,而且實作上對數域的加法運算相對簡單。所以，在對數域操作的加乘演算法，是我們訊息傳遞過程的主要方式。

首先重新定義 LLR(log-likelihood ratio)，以便後續的推導公式：

$$L(x_i) = \log\left(\frac{\Pr(x_i = 0 \mid y_i)}{\Pr(x_i = 1 \mid y_i)}\right) \tag{23}$$

$$L(r_{ji}) = \log\left(\frac{r_{ji}(0)}{r_{ji}(1)}\right) \tag{24}$$

$$L(q_{ij}) = \log\left(\frac{q_{ij}(0)}{q_{ij}(1)}\right) \tag{25}$$

$$L(Q_i) = \log\left(\frac{Q_i(0)}{Q_i(1)}\right) \tag{26}$$

因此針對不同通道錯誤，對數域後驗機率初始值個別表示如下：

$$L\left(q_{ij}\right) = L\left(x_i\right) = (-1)^{y_i} \log\left(\frac{1-p}{p}\right) \qquad (BSC) \tag{27}$$

$$L\left(q_{ij}\right) = L\left(x_i\right) = 2y_i / \sigma^2 \qquad (BI-AWGNC) \tag{28}$$

重新檢視式子(13)，得到 $1-2r_{ji}(1) = \prod_{i' \in V_j \setminus i} (1-2q_{i'j}(1))$ 的結果，再搭配定理 2 如下：

定理 2: 假定 p_0 與 p_1 為正數且相加為 1，則兩數有相對關係如下式。

$$\tanh\left[\frac{1}{2}\log(p_0 / p_1)\right] = p_0 - p_1 = 1 - 2p_1 \tag{29}$$

因此，得到最後表示式(30)。

$$\tanh(\frac{1}{2}L(r_{ji})) = \prod_{i' \in V_j \backslash i} \tanh(\frac{1}{2}L(q_{i'j})) \tag{30}$$

雖然成功轉換機率表示式到對數域，但仍存在相乘關係與複雜 hyperbolic 方程式。對此，我們進一步將 $L(q_{ij})$ 分解成對應的大小與正負值。

$$L(q_{ij}) = \alpha_{ij}\beta_{ij} \tag{31}$$

$$\alpha_{ij} = sign\left[L(q_{ij})\right] \tag{32}$$

$$\beta_{ij} = \left|L(q_{ij})\right| \tag{33}$$

因此，式子(30)繼續改寫如下方流程。

$$\tanh(\frac{1}{2}L(r_{ji})) = \prod_{i' \in V_j \backslash i} \alpha_{i'j} \bullet \prod_{i' \in V_j \backslash i} \tanh(\frac{1}{2}\beta_{i'j}) \tag{34}$$

$$L(r_{ji}) = \prod_{i'} \alpha_{i'j} \bullet 2\tanh^{-1}(\prod_{i'} \tanh(\frac{1}{2}\beta_{i'j})) \tag{35}$$

$$= \prod_{i'} \alpha_{i'j} \bullet 2\tanh^{-1}\log^{-1}\log(\prod_{i'} \tanh(\frac{1}{2}\beta_{i'j})) \tag{36}$$

$$= \prod_{i'} \alpha_{i'j} \bullet 2\tanh^{-1}\log^{-1}\sum_{i'} \log(\tanh(\frac{1}{2}\beta_{i'j})) \tag{37}$$

$$= \prod_{i' \in V_j \backslash i} \alpha_{i'j} \bullet \phi(\sum_{i' \in V_j \backslash i} \phi(\beta_{i'j})) \tag{38}$$

其中 $\phi(x) = -\log\left[\tanh(x/2)\right] = \log\left(\frac{e^x+1}{e^x-1}\right)$ 且 $\phi^{-1}(x) = \phi(x)$。

如此一來，原本相乘的部分變為相加，大幅減低運算量及花費成本。在取得 $L\left(r_{ji}\right)$ 的表示式之後，$L\left(q_{ij}\right)$ 只要將機率域下 $q_{ij}\left(0\right)$ 與 $q_{ij}\left(1\right)$ 相除並取對數即可。

$$L(q_{ij}) = L(x_i) + \sum_{j' \in C_i \backslash j} L(r_{j'i}) \tag{39}$$

而 $L\left(Q_i\right)$ 如法炮製。

綜合以上結果，我們對於對數域的訊息疊代解碼(加乘演算)做總結如下：

(步驟 1)初始化：

根據 LDPC 矩陣畫出對應的 Tanner 圖後，所有邊際上訊息歸零，並設定碼字各位元的對應初始值(LLR)，例如 BSC 通道下，$L\left(q_{ij}\right) = L\left(x_i\right) = (-1)^{y_i}\log\left(\frac{1-p}{p}\right)$。

(步驟 2)更新查核點到變數點的額外訊息：

$$L\left(r_{ji}\right)=\prod_{i'\in V_j\setminus i}\alpha_{i'j}\bullet\phi\left(\sum_{i'\in V_j\setminus i}\phi(\beta_{i'j})\right) \tag{40}$$

其中，$\alpha_{ij}=sign\left[L(q_{ij})\right]$ 且 $\beta_{ij}=\left|L(q_{ij})\right|$ 另外 $\phi(x)=-\log\left[\tanh(x/2)\right]=\log\left(\frac{e^x+1}{e^x-1}\right)$。

(步驟 3)更新變數點到查核點的額外訊息：

$$L(q_{ij})=L(x_i)+\sum_{j'\in C_i\setminus j}L(r_{j'i}) \tag{41}$$

(步驟 4)分別對碼字各位元 i 計算對數上的後驗值：

$$L(Q_i)=L(x_i)+\sum_{j\in C_i}L(r_{ji}) \tag{42}$$

(步驟 5)硬性決定解碼位元值，並判定疊代終止與否。

$$X_i=\begin{cases}0, & if\ L(Q_i)\geq 0;\\ 1, & \text{otherwise}\end{cases} \tag{43}$$

假定疊代次數達到最大值或所有查核方程式被滿足，我們會終止程式迴圈。反之，重複步驟 2。

第3章　基於 LDPC 矩陣的分散式訊源編碼

此章節介紹分散式訊源編碼理論(distributed source coding, DSC)，並展現 LDPC 矩陣在 Slepian-Wolf 理論的具體實現。透過加乘演算法(Sum-product algorithm)與訊息傳遞過程(Message passing processing)的推導，聚焦在邊訊息(side information)的不對稱訊源壓縮。基於 LDPC 碼的分散式編碼與解碼機制，將在內文做詳細的解說。

3.1　分散式訊源編碼理論

有別於標準化壓縮技術的複雜編碼器及簡易解碼器，分散式訊源編碼架構是將運算量從編碼端轉移到解碼端，其低複雜度的編碼演算法非常適合應用在無線感測網路(wireless sensor networks)，以滿足其低功率與即時製作的設計需求。分散式訊源編碼的相關研究，起源於1970年代Slepian和Wolf針對兩相關訊源提出的無失真編碼理論[11]。其主要訴求是，兩個有相關性的訊源，可藉由資源共享的合併編碼模式降低其理論熵值，更重要的是即使在各自獨立編碼的情況下，仍能以合併解碼模式取得相同的理論熵值，如圖 3.1所示。

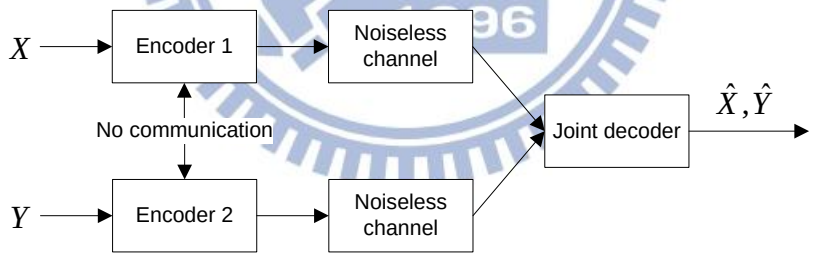


圖 3.1　相關訊源的無失真編碼流程

假設 $H(X)$ 與 $H(Y)$ 分別為X與Y兩個訊源的熵(entropy)值，根據Shannon的編碼理論，針對兩者合併編碼所需的最小碼率為其合併熵值 $H(X,Y)$，而 $H(X,Y) \leq H(X)+H(Y)$。令人驚訝的是，分散式訊源編碼理論證明，即使在編碼端分別針對X與Y進行獨立編碼(separate encoding)，只要解碼端採用合併解碼(joint decoding)模式，最小碼率仍然可以逼近H(X,Y)。更明確的說，若Y是以熵值 $H(Y)$ 進行編碼，則X的理論熵值可由 $H(X)$ 減少為 $H(X|Y)=H(X,Y)-H(Y)$。依此理論設計的無失真分散式訊源編碼器稱為Slepian-Wolf編碼器，其碼率範圍如圖 3.2所示。

Slepian-Wolf編碼器的方法歸為兩類，分別是對稱性與非對稱性架構。操作在C點為對稱性，而A，B兩點為非對稱性架構，也是我們據以實作分散式訊源編碼的基礎。

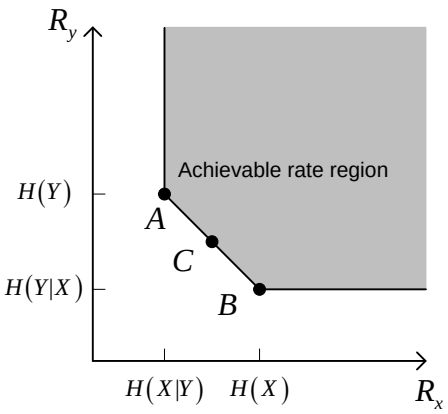


圖 3.2　　分散式訊原編碼之碼率範圍

3.2　無記憶性通道假設的 Slepian-Wolf 壓縮

利用 LDPC 碼壓縮具有相關性的二位元訊源。我們實作的分散式訊源編碼採用[18]介紹的非對稱性架構，如圖 3.3 所示，

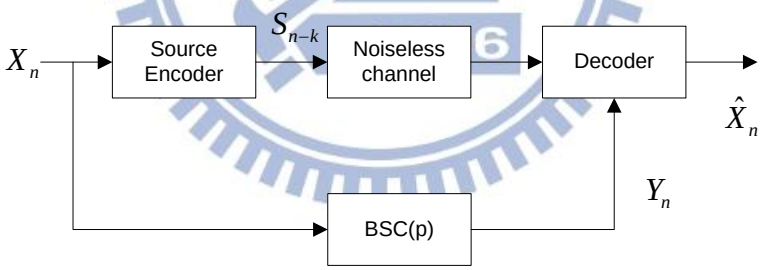


圖 3.3　　BSC 假設的分散式訊原編碼架構

假設兩均勻訊源(uniform source) $X_n=[x_1,x_2,...,x_n]$ 與 $Y_n=[y_1,y_2,...,y_n]$，互為獨立且同分布的變異數群，彼此間的關聯性定義為一無記憶性錯誤(memoryless error)的二位元對稱通道，且 $\Pr[x_i\neq y_i]=p<0.5$。Y 以理論熵值 $H(Y)$ 壓縮且無失真接收於解碼端，X 則是盡可能壓縮到 $R_x\geq H(X|Y)=H(p)=[-p\log_2 p-(1-p)\log_2(1-p)]$。

編碼流程：

編碼過程相當直觀，我們的訊源編碼器使用 LDPC 矩陣進行壓縮，訊源 X 與查核矩陣 H 相乘得到對應的校驗子(syndrome)，表示為 $s=xH^T$。因此，n 個位元的訊息透過編

碼器壓縮成$n-k$的校驗子，換句話說，在 Tanner 圖上，壓縮過程可視為連接於同一個查核點的變數點群作二進位加法，其複雜度與碼字長度成線性關係。

解碼流程：

解碼端透過$n-k$無失真的校驗子與n長度的邊訊息 Y，還原出訊源 X。原理與通道編碼一樣，邊訊息 Y 可視為訊源 X 經過二位元對稱通道(BSC)後的結果，假定錯誤率為p。所以在 Tanner 圖上，來自通道錯誤 Y 的初始 LLR 可表示為，

$$L(x_i)=\log\left(\frac{\Pr(x_i=0\mid y_i)}{\Pr(x_i=1\mid y_i)}\right)=(-1)^{y_i}\log\left(\frac{1-p}{p}\right) \tag{44}$$

另外，訊源 X 本身並不一定是 LDPC 碼的合法碼字，所以校驗子不全為 0，因此原先訊息傳遞演算法必須做修改，也就是額外考慮校驗子訊息並給予查核點，整體概念以圖 3.4 表示。

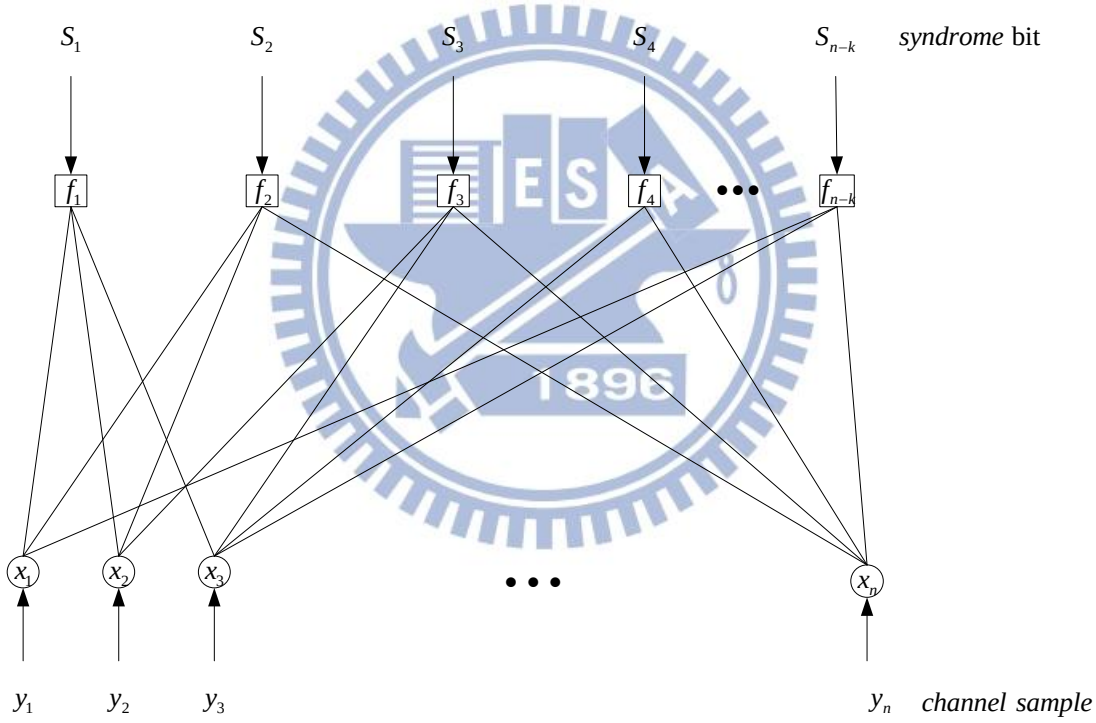


圖 3.4　　BSC 通道的 Tanner 圖

解壓縮演算過程，以邊訊息的初始 LLR 和壓縮後無失真的校驗子為輸入，執行 LDPC 碼的疊代訊息解碼，採用對數域的加乘演算過程，並修正步驟二的(40)式為：

$$\tanh\left(\frac{1}{2}L(r_{ji})\right)=\left(1-2S_j\right)\prod_{i'\in V_j\setminus i}\tanh\left(\frac{1}{2}L(q_{i'j})\right) \tag{45}$$

$$L(r_{ji})=2\tanh^{-1}\left(\left(1-2S_j\right)\prod_{i'\in V_j\setminus i}\tanh\left(\frac{1}{2}L(q_{i'j})\right)\right) \tag{46}$$

查核點將校驗子訊息與 tanh 規則結合在一起，這也是唯一與之前通道解碼的不同處。

3.3 記憶性通道假設的 Slepian-Wolf 壓縮

在 3.2 章節的非對稱性分散式訊源編碼架構裡，邊訊息(side information)Y 為 X 通過二位元對稱性虛擬通道後的結果，也就是說虛擬通道錯誤本身並無記憶性。但在實際應用上，相關訊號間的虛擬錯誤彼此間統計上顯示有相當的關連性[19][20]。因此我們針對有限狀態的馬可夫通道，根據通道記憶特性提出改良的分散式訊源編碼器。我們採用最常見的 Gilbert 通道，並且運用於分散式訊源編碼的設計。

所謂的 Gilbert 通道，就是馬可夫鏈裡有兩個通道環境，稱為好狀態(good state)與差狀態(bad state)，而每一種狀態內就是二位元對稱通道(BSC)。令好狀態的交叉錯誤率為 0，差狀態的交叉錯誤率為 P_e^b。另外，狀態之間有轉移機率，P_{gb} 表示從好狀態轉移到差狀態的轉移機率；P_{bg} 為差狀態到好狀態的轉移機率。整體模型如圖 3.5 所示。

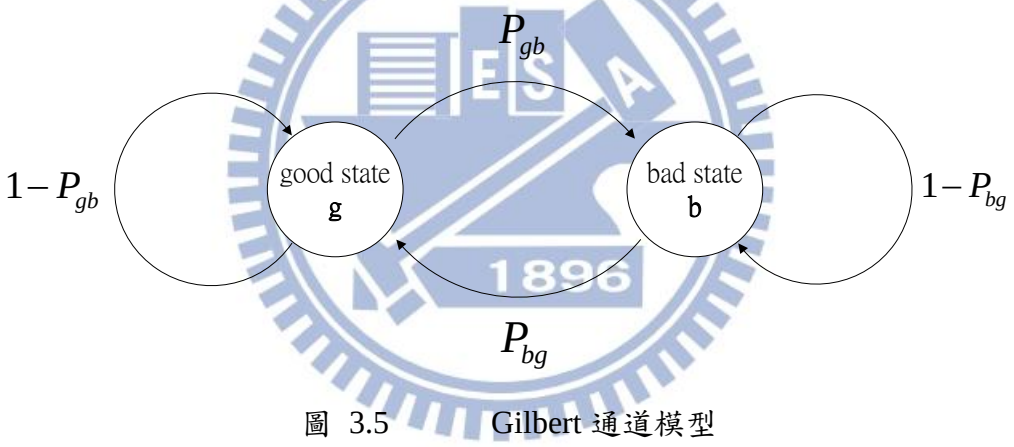


圖 3.5 Gilbert 通道模型

因此，在非對稱性分散式訊源編碼架構圖 3.6 中，邊訊息 Y 定義為 $Y = X \oplus Z$，$\oplus$ 表示二位元加法運算。令訊源 $X \in \{0,1\}^n$ 為虛擬通道輸入源，$Z \in \{0,1\}^n$ 為雜訊序列，而通道輸出 $Y \in \{0,1\}^n$。雜訊 Z 由 Gilbert 通道模擬產生，其中模型參數 $\{P_{gb}, P_{bg}, P_e^b\}$ 之數值範圍分別為 $0 \le P_e^b \le 0.5$ (non-inverting channels)，$0 \le \{P_{gb}, P_{bg}\} \le 0.5$ (non-deterministic，non-oscillatory channels)。$\sigma \in \{g,b\}^n$ 代表狀態序列組且由狀態轉移機率決定，依每個狀態下的二位元對稱通道(BSC)產生雜訊，好狀態不發生錯誤，差狀態則有 P_e^b 的錯誤率。透過狀態轉移機率，可以求得穩態機率(steady-state probability)，$P_b = P_{gb} / (P_{bg} + P_{gb})$ 為差狀態的穩態機率；$P_g = P_{bg} / (P_{bg} + P_{gb})$ 則是好狀態的穩態機率。最後定義 Gilbert 通道的平均錯誤率，$\overline{P}_e = P_e^b P_b = (P_e^b P_{gb}) / (P_{bg} + P_{gb})$。

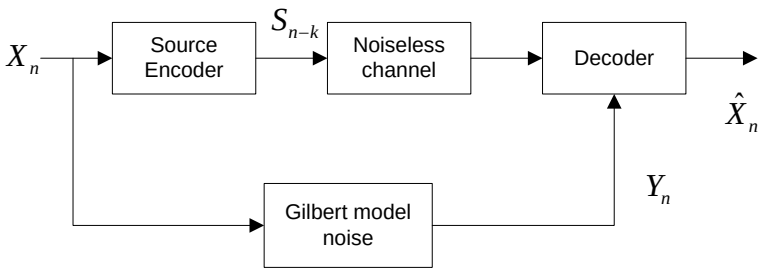


圖 3.6　　Gilbert 虛擬通道的分散式訊源編碼架構

編碼流程：

與 3.2 章節一樣，將訊源 X 與 LDPC 矩陣相乘作壓縮，求得校驗子(syndrome)，$s = xH^T$，符合分散式訊源編碼端低複雜度的要求。

解碼流程：

假設解碼端已知 Gilbert 通道的模型參數，並且成功接受邊訊息 Y 與無失真的校驗子 S_{n-k}，即可進行「修正」的疊代訊息解碼。修正的目的，主要是針對具有記憶性通道錯誤作調整，雜訊是由狀態序列對應條件下所產生，狀態間彼此有相關性，直接影響到雜訊的表現，所以解碼端採用的後驗機率必須考慮到通道狀態序列。因此綜合考慮邊訊息 Y 與所有訊源 X 以及一連串通道狀態的後驗機率最大化表示如下：

$$\max_{x,\sigma} M(y,x,\sigma) = \Pr(y \mid x,\sigma)\cdot\Pr(\sigma)\cdot\Pr(x) \tag{47}$$

$$= \prod_{i=1}^{n}\Pr(y_i \mid x_i,\sigma_i)\cdot\Pr(x_i)\cdot\Pr(\sigma_i)\prod_{i=1}^{n-1}\Pr(\sigma_{i+1} \mid \sigma_i) \tag{48}$$

在變數點求出解碼字元 0 與 1 的機率 $\Pr(x_i)$，而 $\Pr(\sigma)$ 為狀態點(state nodes)反映好狀態與差狀態的機率。訊息傳遞解碼操作是在原始 Tanner 圖上(包含變數點與查核點)，外加新的節點，稱為狀態點，如圖 3.7 所示。

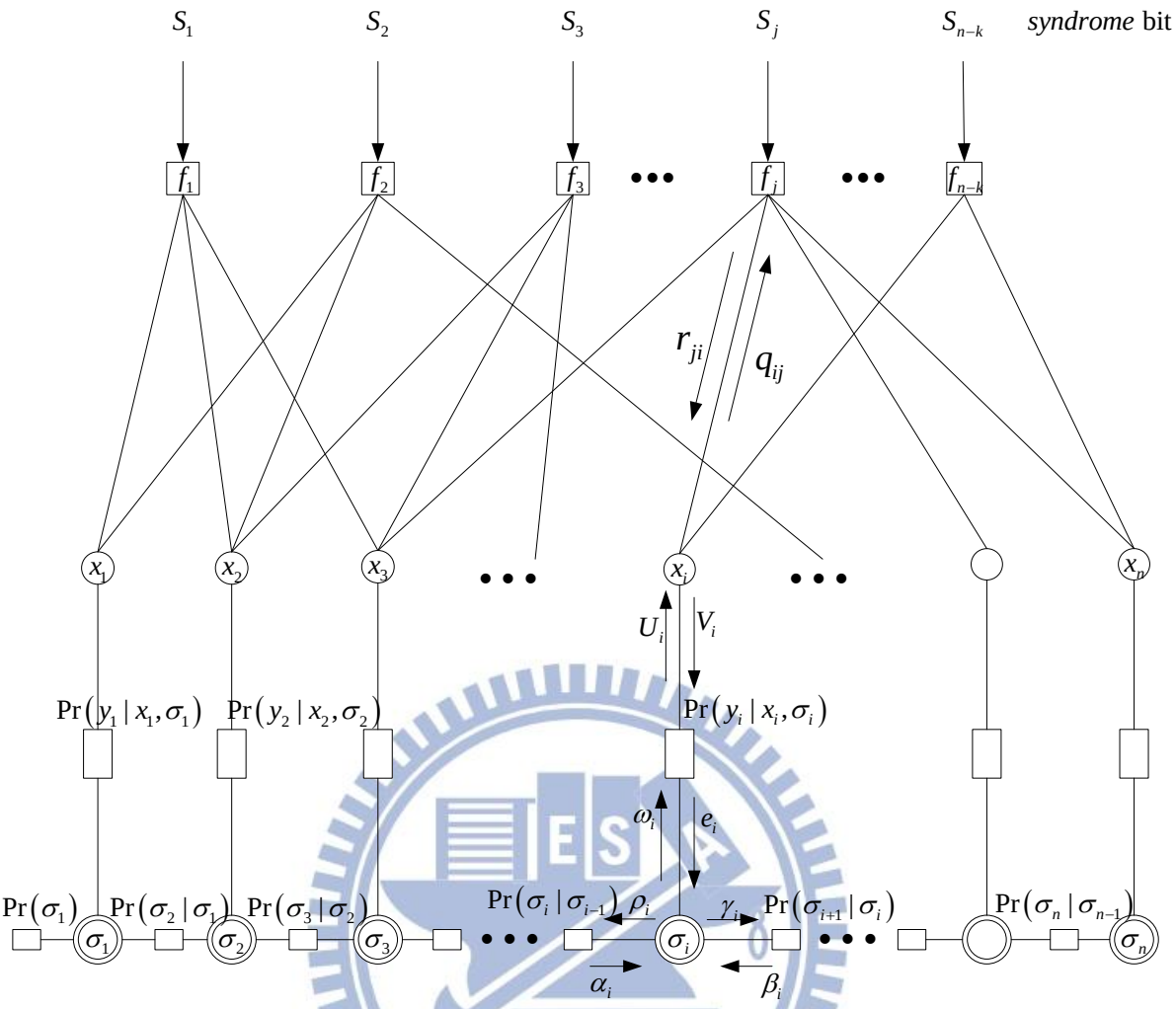


圖 3.7 Gilbert 通道的 Tanner 圖

整個架構可分為上半部的 LDPC 碼本身，與下半部的通道狀態。上半部疊代訊息過程與之前無記憶性錯誤的解碼相似，唯一不同是每個碼字位元的事前 LLR U_n 必須在每次疊代後作更新動作。同時，每個碼字疊代後的邊訊息 V_n 會帶動下半部通道狀態的資訊交流。

在此定義針對記憶性通道解碼演算所需要的相關符號。

$V_j = \{$連接查核點 f_j 的變數點群$\}$

$V_j \setminus i = \{$連接查核點 f_j 的變數點群$\} \setminus \{$變數點 $x_i\}$

$C_i = \{$連接變數點 x_i 的查核點群$\}$

$C_i \setminus j = \{$連接變數點 x_i 的查核點群$\} \setminus \{$查核點 $f_j\}$

$q_{ij}, r_{ji} = \{$於變數點 x_i 與查核點 f_j 之間流動的LLR訊息$\}$

$V_i, U_i = \{$交流於變數點 x_i 與對應相連通道的LLR訊息$\}$

$e_i, \omega_i = \{$交流於通道狀態σ_i與碼字之間的可能性訊息$\}$

$\alpha_i, \rho_i = \{$交流於通道狀態σ_i 與σ_{i-1}之間的可能性訊息$\}$

$\beta_i, \gamma_i = \{$交流於通道狀態σ_i 與σ_{i+1}之間的可能性訊息$\}$

$g = \{$好狀態good state$\}$ & $b = \{$差狀態bad state$\}$

解碼過程如下：

初始化：

在新 Tanner 架構圖上(包含狀態點) 所有的邊際訊息歸零，並對所有碼字設定初始訊息，$U_i = (-1)^{y_i} \ln\left[\left(1-\overline{P}_e\right)/\overline{P}_e\right]$ 且 $q_{ij} = U_i$ 對於 $i \in V_j$。另外，設定所有狀態點初始值為各自穩態機率，$\alpha_i(b) = \beta_i(b) = P_b, \alpha_i(g) = \beta_i(g) = P_g$。

疊代過程：[分兩大區塊進行]

查核矩陣碼疊代過程上半部：

與之前一樣，採用對數域下的加乘演算，查核點與變數點之間額外訊息交換，由於是分散式訊源編碼，查核點會另將校驗子訊息考慮進去。每次做完一次疊代，更新解碼後字元的機率，並將訊息往下傳送給下半部的疊代解碼端(狀態點部分)。

$$V_i = \sum_{j \in C_i} r_{ji} \tag{49}$$

$$v_i(0) = e^{V_i}/\left(1+e^{V_i}\right) \tag{50}$$

$$v_i(1) = 1/\left(1+e^{V_i}\right) \tag{51}$$

通道狀態疊代過程下半部：

1. 變數點到狀態點：

首先，變數點得知解碼字元 0 與 1 的機率，搭配對應接收值 y，並以後驗機率方式求得好狀態與差狀態的情況，過程如下：

對於所有 $i \in \{1,2,...,n\}$，令 $y_i^* = 1 - y_i$。

$$e_i(b) = \Pr(y_i \mid x_i = 0, \sigma_i = b) \cdot v_i(0) + \Pr(y_i \mid x_i = 1, \sigma_i = b) \cdot v_i(1)$$

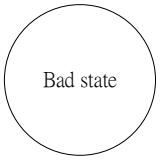
$$= P_e^b v_i(y_i^*) + \left(1 - P_e^b\right) v_i(y_i) \tag{52}$$

$$e_i(g) = \Pr(y_i \mid x_i = 0, \sigma_i = g) \cdot v_i(0) + \Pr(y_i \mid x_i = 1, \sigma_i = g) \cdot v_i(1)$$

$$= P_e^g v_i\left(y_i^*\right) + \left(1 - P_e^g\right) v_i\left(y_i\right)$$

$$= v_i\left(y_i\right), \quad \because P_e^g = 0 \tag{53}$$

式子(52)與(53)反應了 $\max_{x,\sigma} M\left(y,x,\sigma\right) = pr\left(y \mid x,\sigma\right) \cdot \Pr\left(\sigma\right) \cdot \Pr\left(x\right)$ 中的部分關係。



received bit y *coded bit x*

$$0 \xrightarrow[\text{no error}]{} 0 \Rightarrow \left(1 - P_e^b\right) v_i\left(0\right) \oplus = e_i\left(b\right)$$

$$0 \xrightarrow[\text{error}]{} 1 \Rightarrow P_e^b v_i\left(1\right)$$

2. 狀態點之間：

由於馬可夫鏈(Markov chain)通道記憶特性，我們將 $\Pr\left(\sigma\right)$ 拆成 $\Pr\left(\sigma_1\right)\prod_{i=1}^{n-1}\Pr\left(\sigma_{i+1} \mid \sigma_i\right)$。

並定義出前向訊息(forward message) $\left\{\alpha_i, \gamma_i\right\}$ 與後向訊息(backward message) $\left\{\beta_i, \rho_i\right\}$。

整體而言，狀態點之間，利用來自變數點訊息 e_i，此刻狀態機率以及狀態轉移機率不斷去更新狀態最新資訊，其過程又可細分為三個部分。

A. 當狀態點接收到來自變數點的邊訊息 e_i 後，隨即更新前向訊息 γ_i 對應的狀態機率，並提供給下一個狀態點進行後續處理。

$$\gamma_i\left(b\right) = C \cdot \alpha_i(b) \cdot e_i\left(b\right), \gamma_i\left(g\right) = C \cdot \alpha_i(g) \cdot e_i\left(g\right) \tag{54}$$

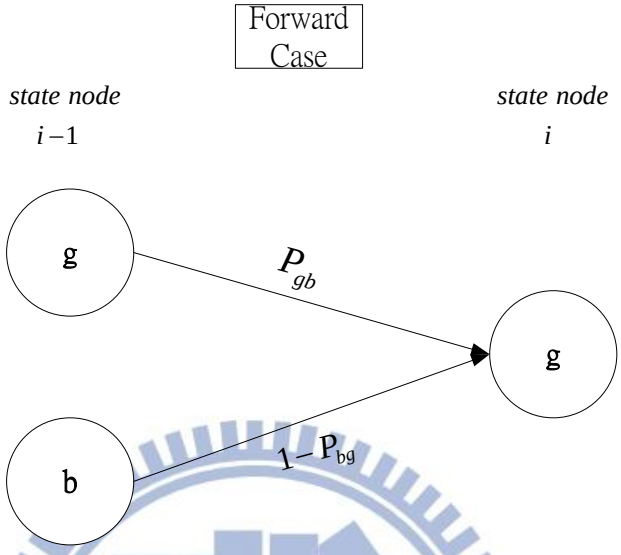
同理，也更新後向訊息 ρ_i，並傳遞給上一個狀態點。

$$\rho_i\left(b\right) = C \cdot \beta_i\left(b\right) \cdot e_i\left(b\right), \rho_i\left(g\right) = C \cdot \beta_i\left(g\right) \cdot e_i\left(g\right) \tag{55}$$

B. 兩相鄰狀態點有著關連性，所以 A 步驟的 γ_i 與 ρ_i 以及狀態轉移機率被用來改變 α_i 和 β_i，進一步求得第 i 個狀態點的機率值。

$$\begin{aligned} \alpha_i\left(b\right) &= C \cdot \left(P_{gb}\gamma_{i-1}\left(g\right) + \left(1 - P_{bg}\right)\gamma_{i-1}\left(b\right)\right) \\ \alpha_i\left(g\right) &= C \cdot \left(P_{bg}\gamma_{i-1}\left(b\right) + \left(1 - P_{gb}\right)\gamma_{i-1}\left(g\right)\right) \end{aligned} \tag{56}$$

$$\begin{aligned}\beta_i(b) &= C\cdot\left(P_{bg}\rho_{i+1}(g)+\left(1-P_{bg}\right)\rho_{i+1}(b)\right)\\ \beta_i(g) &= C\cdot\left(P_{gb}\rho_{i+1}(b)+\left(1-P_{gb}\right)\rho_{i+1}(g)\right)\end{aligned} \tag{57}$$



C. 將步驟 B 所得 α_i 和 β_i 相乘即可完成狀態點之間訊息交流過程，也就是更新完成好狀態與差狀態各自的機率。

$$\begin{cases}\omega_i(b) = C\cdot\alpha_i(b)\cdot\beta_i(b)\\ \omega_i(g) = C\cdot\alpha_i(g)\cdot\beta_i(g)\end{cases} \tag{58}$$

3. 狀態點到變數點：

狀態點將更新後的狀態機率向上傳遞給變數點，並依此重新計算平均錯誤率，更新初始 LLR。

$U_i = (-1)^{y_i}\ln\left[\left(1-\overline{P}_{e,i}\right)\Big/\overline{P}_{e,i}\right]$，其中 $\overline{P}_{e,i} = P_e^b\cdot\omega_i(b)$。

因此，變數點給查核點的額外訊息 $q_{ij} = U_i + \sum_{j'\in C_i\setminus j} r_{j'i}$，如此不斷進行疊代演算，直到條件滿足，或達到次數上限。再根據 $U_i + \sum_{j\in C_i} r_{ji} > 0$ 判定第 i 碼字位元為 0，反之為 1。

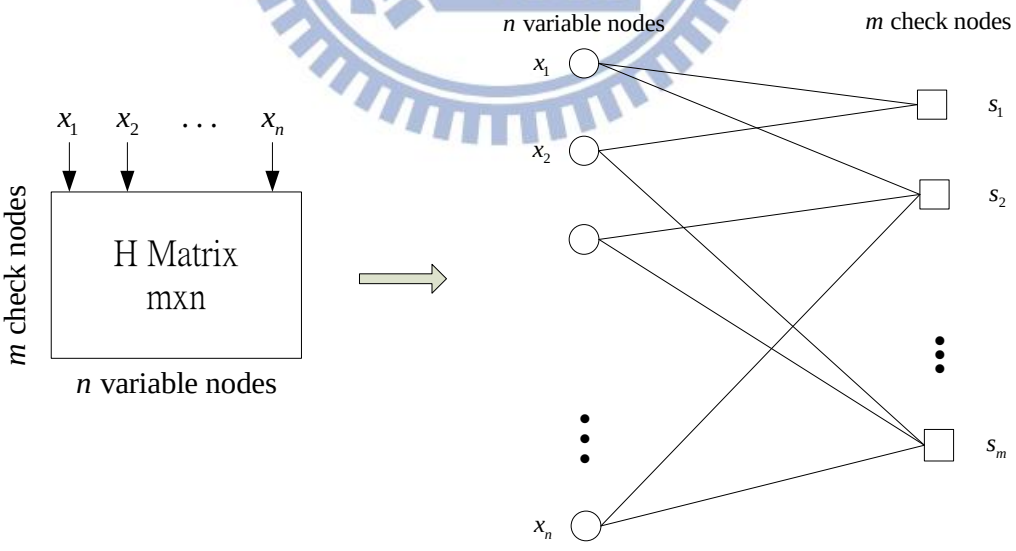
第4章　雜訊通道下 Slepian-Wolf 壓縮

本章節將會探討，Slepian-Wolf 壓縮輸出的校驗子經由雜訊通道傳輸時的挑戰。我們將考慮兩種通道碼，LDPC 碼應用在分散式訊源編碼的校驗子生成，而迴旋碼(convolution code)則提昇壓縮資料對抗通道雜訊的能力。因此，軟性疊代式合併訊源通道解碼(soft iterative joint source-channel decoder)值得深入探討，主要在於訊源與通道編碼分開，可使設計簡單化且效能性更高，而合併疊代解碼能有效運用串連架構潛在的能力。首先，我們介紹 LDPC 碼在校驗子發生錯誤時，所採取的改良解碼方法。接著介紹迴旋碼架構以及 BCJR 演算法，最後敘述合併訊源通道解碼中額外訊息的運用與產生。

4.1　擴展式的 LDPC 碼

如之前所提到的，分散式訊源編碼的壓縮，是根據通道編碼理論中將長的序列(碼字)劃分到短的字串(校驗子)。在 Tanner 圖上，校驗子為相連於同一查核點的變數點群作二進位加法的結果。如圖 4.1 所示，

$$s_j = \sum_{i \in C_j} \oplus x_i,\ j = 1, 2, \cdots, m. \tag{59}$$



Matrix representation　　*Tanner* representation

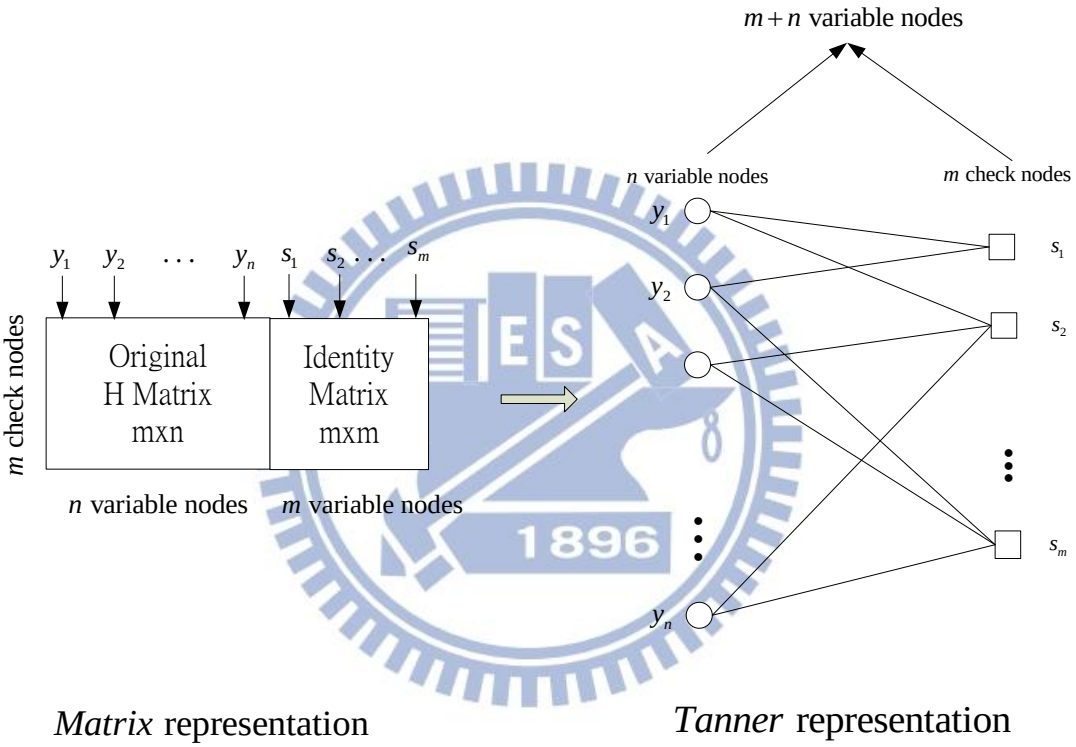
圖 4.1　LDPC 矩陣示意圖

經由移項可改寫成為

$$s_j \oplus \sum_{i \in C_j} \oplus x_i = 0,\ j = 1, 2, \cdots, m. \tag{60}$$

換句話說，假設新增 m 個額外變數點用來傳遞校驗子訊息，搭配原本 n 個變數點後，會滿足對應連接的查核方程式。同等的情況下，如果我們建造一矩陣，左半邊為原矩陣 H，右半邊加入一 $m \times m$ 的單位矩陣，如圖 4.2 所示。此新矩陣稱為擴展式低密度奇偶查核矩陣 H_{ext} (extended LDPC matrix)，且 $[X, S_x]$ 為此查核矩陣的合法碼字。

$$H_{ext} \times [X, S_x] = 0. \tag{61}$$



Matrix representation　　*Tanner* representation

圖 4.2　　擴展式 LDPC 矩陣示意圖

因此，受到雜訊干擾的碼字例如 $[Y, S_x]$，透過擴展式的 LDPC 碼解碼，還原出對應合法碼字 $[X, S_x]$，即可找到訊源 X。訊息疊代解碼方式如前面章節，直接套用對數域的加乘演算法，不過對於新增加的變數點群，校驗子的初始 LLR 來自通道解碼(迴旋碼)後的額外訊息。

4.2　迴旋碼與其通道解碼演算法

4.2.1 迴旋碼(Convolution Code)

在講述迴旋碼(convolution code)之前，我們先從通道編碼概念說起。資料傳輸過程，一定會經過通道，不論有線或無線傳送，因此我們的資料很容易受到雜訊干擾，以致無法在接收端收到完整訊息。為了保護重要資料，對抗通道雜訊，才有了通道編碼的想法。其概念在於傳送端針對資料位元，以特定的規則產生與資料相關的冗餘(redundancy)位元，並將其整合後傳送。由於加了冗餘位元，相當於多一層保護機制，儘管傳輸過程有資料毀損，接收端還是可以利用冗餘位元的相關特性還原資料。舉例來說，(3,1)的重複碼(repetition code)，資料位元 1 編碼後的碼字為(1,1,1)，即使傳送中發生單一位元的錯誤，例如：(1,0,1)或(1,1,0)，解碼端還是可判定資料位元為 1。

傳統的錯誤更正碼(error correcting code)區分為線性區塊碼與迴旋碼兩種，主要差別在於碼字序列與輸入訊息有無記憶性。前者以區塊方式編輯資料成碼字序列，且碼字間不具時間關聯性。至於後者的遞迴系統式迴旋碼(recursive systematic convolution code, RSC)，為本論文採用的通道編碼方式。迴旋碼是在 1955 年由 Elias 提出，至今依舊廣泛使用。迴旋碼具有記憶性，指的是輸出碼字不僅與目前輸入資料有關，也受到之前輸入資料的影響。常以 (n,k,m) 表示碼率 k/n 的迴旋碼，k 個輸入資料位元對應 n 個輸出碼字位元，而 m 為位移暫存器的級數，又稱記憶級數(memory order)，通常級數越高的迴旋碼會有較高的錯誤更正能力。

迴旋編碼器可視為一個有線狀態機(finite-state machine)，其輸入與輸出可用狀態圖(state diagram)表示，而狀態的數量決定於暫存器的級數 m。假設有一(2,1,2)的迴旋碼，四種狀態對應的狀態圖如圖 4.3 所示。另外，依不同時間展開的狀態轉移過程稱之為柵狀圖(trellis diagram)，有助於迴旋碼的線性解碼。

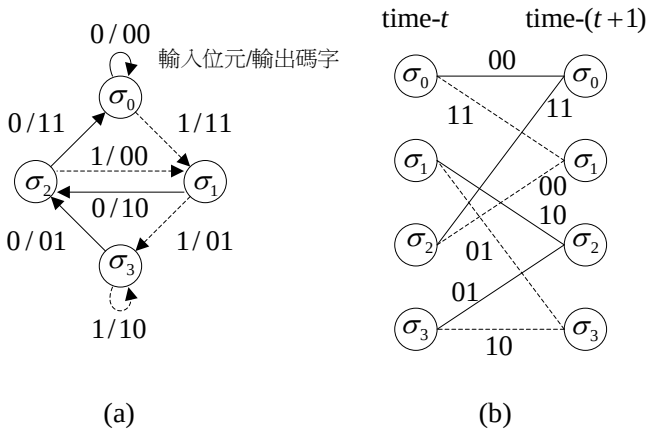


圖 4.3　　迴旋碼的(a)狀態圖與(b)柵狀圖

4.2.2 **BCJR 解碼演算法**

BCJR 演算法是在 1970 年代由 Bahl、Cocke、Jelinek 以及 Raviv 四位所提出[21]，至今仍常用在軟性輸入輸出的迴旋解碼。主要是根據柵狀圖中狀態間的轉移關係，以最大後驗機率(maximum a posteriori probability,MAP)法則求得每個對應位元的最佳解，因此能達到最小的位元錯誤率。為了增加解碼後訊息位元的可靠度，解碼過程會觀察整段接收值並配合柵狀圖碼，同時以遞迴方式可大幅降低計算複雜度。柵狀圖的構想來自於迴旋碼的編碼過程，優點在於提供解碼器進行最佳線性演算處理。而位元層級的柵狀圖如圖 4.4 所示， $m=2$ 暫存器的狀態 $\sigma_j \in \{0,1,...,2^m-1\}$，每一狀態 σ_j 依不同的輸入位元產生兩條路徑到下一個狀態 σ_{j+1}，同時產生其對應的輸出碼字。另外，狀態轉移間有記憶性，BCJR 演算法充分利用這些關連性，進行其順向與反向的遞迴運算。

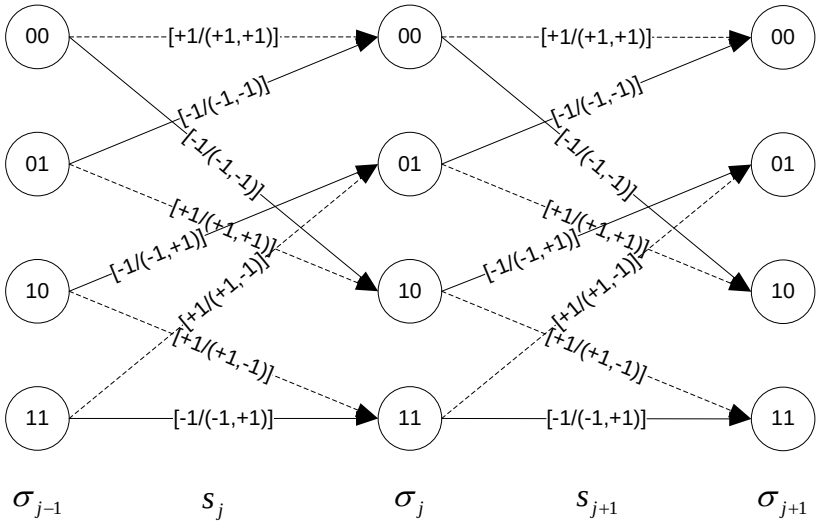


圖 4.4　　位元層級柵狀圖

根據迴旋碼的柵狀圖，我們就能依此推導位元層級的 BCJR 演算法。首先為了方便，我們在所有推導方程式中只考慮校驗子 $s_j=+1$ 的部分。而 $s_j=-1$ 也比照計算方式，以確保 $P\left(s_j=+1|W_1^L\right)$ 與 $P\left(s_j=-1|W_1^L\right)$ 相加為 1，W_1^L 為一連串接收值。柵狀圖根據目前的輸入位元以及暫存器狀態決定目前的輸出碼字，亦即目前的輸入位元與狀態共同決定唯一的狀態轉移過程又稱支線(branch)。因此我們將狀態資訊包含於 s_j 的後驗機率裡，

$$\begin{aligned} P\left(s_j=+1|W_1^L\right)&=\frac{p\left(s_j=+1,W_1^L\right)}{P\left(W_1^L\right)}=\frac{\sum_{\left(\sigma_{j-1},\sigma_j\right)\in\Sigma_j^+}p\left(\sigma_{j-1},\sigma_j,W_1^L\right)}{P\left(W_1^L\right)} \\ &=C\cdot\sum_{\left(\sigma_{j-1},\sigma_j\right)\in\Sigma_j^+}p\left(\sigma_{j-1},\sigma_j,W_1^L\right) \end{aligned} \tag{62}$$

C 為一正規化常數，使得 $\sum_{\left(\sigma_{j-1},\sigma_j\right)\in\Sigma_j^+}p\left(\sigma_{j-1},\sigma_j,W_1^L\right)=1$，而 $\sum_j^+$ 是對應輸入位元 $s_j=+1$ 的狀態轉移集合。

假設通道無記憶性以及輸入位元間彼此獨立，合併機率 $p\left(\sigma_{j-1},\sigma_j,W_1^L\right)$ 可拆成三項如下：

$$\begin{aligned} p\left(\sigma_{j-1},\sigma_j,W_1^L\right)&=p\left(\sigma_{j-1},W_1^{j-1}\right)\cdot p\left(\sigma_j,W_j|\sigma_{j-1}\right)\cdot p\left(W_{j+1}^L|\sigma_j\right) \\ &\triangleq\alpha_{j-1}\left(\sigma_{j-1}\right)\cdot\gamma_j\left(\sigma_{j-1},\sigma_j\right)\cdot\beta_j\left(\sigma_j\right) \end{aligned} \tag{63}$$

順向計量 $\alpha_j\left(\sigma_j\right)$ 可進一步展開而得遞迴循環公式：

$$\alpha_j\left(\sigma_j\right)=\sum_{s_{j-1}}\gamma_j\left(\sigma_{j-1},\sigma_j\right)\cdot\alpha_{j-1}\left(\sigma_{j-1}\right) \tag{64}$$

同理，反向計量 $\beta_{j-1}\left(\sigma_{j-1}\right)$ 表示為：

$$\beta_{j-1}\left(\sigma_{j-1}\right)=\sum_{s_j}\gamma_j\left(\sigma_{j-1},\sigma_j\right)\cdot\beta_j\left(\sigma_j\right) \tag{65}$$

至於路徑計量 $\gamma_j\left(\sigma_{j-1},\sigma_j\right)$，

$$\begin{aligned} \gamma_j\left(\sigma_{j-1},\sigma_j\right)&=P\left(\sigma_j|\sigma_{j-1}\right)\cdot p\left(w_j|\sigma_{j-1},\sigma_j\right) \\ &=P\left(s_j=+1\right)\cdot p\left(w_j|w_j\right) \\ &=P\left(s_j=+1\right)\cdot p\left(w_j^s|s_j=+1\right)\cdot p\left(w_j^p|w_j^p\right) \end{aligned} \tag{66}$$

其中第二式來自 $\left(\sigma_{j-1},\sigma_j\right)\in\sum_j^+$ ，第三式是假設通道無記憶性所展開而成。因為迴旋編碼的起始與最終狀態都全為零，所以順向與反向計量的初始值設定為：

$$\alpha_0\left(\sigma_0\right)=\begin{cases}1, & \sigma_0=0\\ 0, & \sigma_0\neq 0\end{cases} \tag{67}$$

$$\beta_L\left(\sigma_L\right)=\begin{cases}1, & \sigma_L=0\\ 0, & \sigma_L\neq 0\end{cases} \tag{68}$$

綜合以上所述，BCJR 演算法透過順向計量、反向計量，以及路徑計量的運算組合，求得輸入位元後驗機率的最大值。

4.3 疊代式訊源通道解碼

疊代訊源通道解碼(iterative source-channel decoder,ISCD)技術是依據渦輪原則(turbo principle)，在訊源與通道編碼器之間加入一交換器(interleaver)，以確保兩者各自的輸入序列並無相關性。同理，解碼端內部也會插入反交換器(de-interleaver)，來重建原始資料的排序。

如圖 4.5 所示，n 位元的訊源 x 經過基於 LDPC 碼的分散式訊源編碼處理，產生 m 個位元的校驗子 s，透過交換器打亂後輸入迴旋通道編碼器產生長度為 L 的碼字 w'(包含終止位元)。

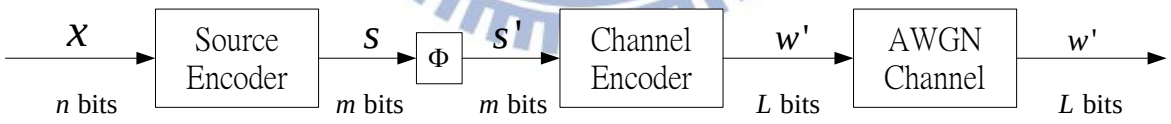


圖 4.5 ISCD 編碼端

疊代解碼器分為兩部分，一為採用 LDPC 碼的外部訊源解碼器；另一為採用迴旋碼的內部通道解碼器，而軟性訊息交換則使用 LLR(log-likelihood ratio)，如圖 4.6 所示。外部與內部解碼器之間有整體訊息的交流，而外部的 LDPC 解碼器本身也會有局部訊息交換，因此定義 (g,l) 為疊代數目，g 指的是內外部訊息疊代次數；l 為外部 LDPC 碼的最大疊代值。另外，不論是解碼器之間或者 LDPC 碼本身的額外訊息疊代過程，都必須遵守渦輪原則。接下來，進一步分析兩解碼器的工作流程。

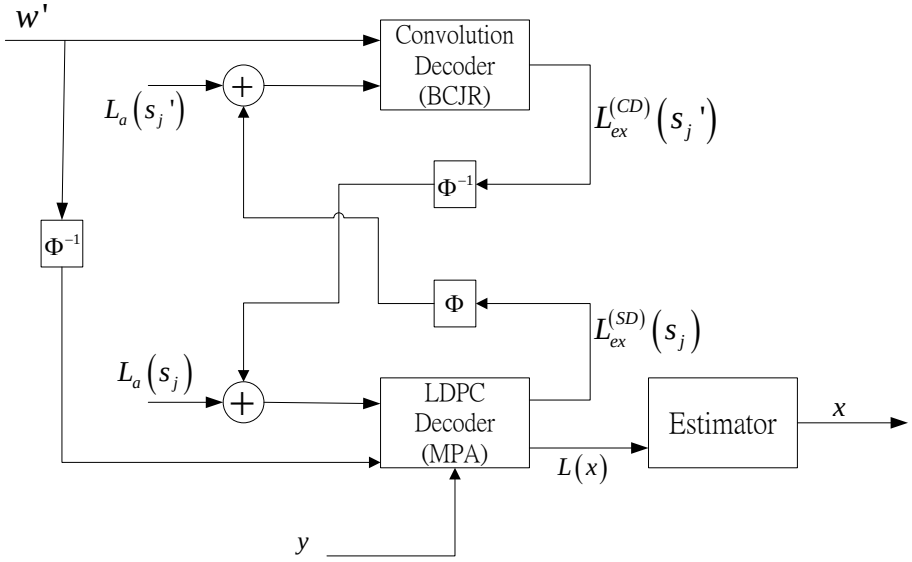


圖 4.6　　ISCD 解碼端

1. 基於 LDPC 碼的訊源解碼器：

由於校驗子有誤，我們必須採用 4.1 章節提到的擴展式 LDPC 矩陣來解碼。因此，對於新增的變數點，校驗子初始 LLR 來自迴旋碼解碼後的額外訊息 $L_{ex}^{(CD)}(s_j)$ 與通道相關訊息 $L_c(s_j)$ 以及事前資訊 $L_a(s_j)$。而原變數點的初始 LLR 則是邊訊息 y 所對應的虛擬通道計算值，若以二位元對稱通道來模擬，$L_a(x)=(-1)^{y_i}\log\left(\frac{1-p}{p}\right)$。擴展式 LDPC 碼經過訊息疊代解碼，後驗的 LLR 被用來決定訊源 x。而後驗 LLR $L^{(SD)}(s_j)$ 扣掉初始 LLR 所得的校驗子額外 LLR $L_{ex}^{(SD)}(s_j)$，則經過交換器後傳送到內部迴旋解碼器當輸入起始值。

2. 迴旋通道解碼器：

4.2 章節提到，迴旋解碼採用 BCJR 演算法求出輸入位元的後驗機率。除此之外，我們需要進一步找出迴旋通道碼的額外訊息，定義為 $P_{ex}^{(CD)}(s=+1)$。特別強調的是，通道解碼額外訊息並不一定是合法的機率組合(相加不為 1)，所以必須分開計算 $P_{ex}^{(CD)}(s=+1)$ 和 $P_{ex}^{(CD)}(s=-1)$。首先，重新考慮路徑計量並找出其額外成分。由於兩解碼端都共用同樣的通道相關訊息以及校驗子事前資訊，所以此兩項訊息不納入額外訊息部分。路徑計量可改寫如下：

$$\begin{aligned}\gamma_j\left(\sigma_{j-1},\sigma_j\right) &= P\left(s_j=+1\right)\cdot p\left(w_j^s \mid s_j=+1\right)\cdot p\left(w_j^p \mid w_j^p\right)\\ &\triangleq P\left(s_j=+1\right)\cdot p\left(w_j^s \mid s_j=+1\right)\cdot \gamma_j^{[ext]}\left(\sigma_{j-1},\sigma_j\right)\end{aligned} \tag{69}$$

其中 $\gamma_j^{[ext]}\left(\sigma_{j-1},\sigma_j\right)=p\left(w_j^p \mid w_j^p\right)$。因此後驗機率表示為：

$$\begin{aligned}P\left(s_j=+1 \mid W_1^L\right) &= C\cdot \sum_{\left(s_{j-1},s_j\right)\in\Sigma_j^+}\alpha_{j-1}\left(\sigma_{j-1}\right)\cdot\gamma_j\left(\sigma_{j-1},\sigma_j\right)\cdot\beta_j\left(\sigma_j\right)\\ &= C\cdot P\left(s_j=+1\right)\cdot p\left(w_j^s \mid s_j=+1\right)\cdot \sum_{\left(\sigma_{j-1},\sigma_j\right)\in\Sigma_j^+}\alpha_{j-1}\left(\sigma_{j-1}\right)\cdot\gamma_j^{[ext]}\left(\sigma_{j-1},\sigma_j\right)\cdot\beta_j\left(\sigma_j\right)\\ &\triangleq C\cdot P\left(s_j=+1\right)\cdot p\left(w_j^s \mid s_j=+1\right)\cdot P_{ex}^{(CD)}\left(s=+1\right)\end{aligned} \tag{70}$$

如此一來，通道解碼額外訊息定義如下:

$$P_{ex}^{(CD)}\left(s=+1\right)=\sum_{\left(\sigma_{j-1},\sigma_j\right)\in\Sigma_j^+}\alpha_{j-1}\left(\sigma_{j-1}\right)\cdot\gamma_j^{[ext]}\left(\sigma_{j-1},\sigma_j\right)\cdot\beta_j\left(\sigma_j\right) \tag{71}$$

在經過第一次整體疊代之後，來自訊源解碼的額外訊息會被用來做事前更新，

$$P\left(s_j=+1\right)\leftarrow\left[P_{ex}^{(SD)}\left(s_j=+1\right)\cdot P\left(s_j=+1\right)\right] \tag{72}$$

同時更新路徑計量：

$$\gamma_j\left(\sigma_{j-1},\sigma_j\right)=P\left(s_j=+1\right)\cdot P_{ex}^{(SD)}\left(s_j=+1\right)\cdot p\left(w_j^s \mid s_j=+1\right)\cdot\gamma_j^{[ext]}\left(\sigma_{j-1},\sigma_j\right) \tag{73}$$

最後總結後驗機率：

$$P\left(s_j=+1 \mid W_1^L\right)=C\cdot P\left(s_j=+1\right)\cdot P_{ex}^{(SD)}\left(s_j=+1\right)\cdot p\left(w_j^s \mid s_j=+1\right)\cdot P_{ex}^{(CD)}\left(s_j=+1\right) \tag{74}$$

由於我們是以 LLR 表示訊息，迴旋解碼產生的額外 LLR 可表示為：

$$L_{ex}^{(CD)}\left(s_j'\right)=\log\left(\frac{P_{ex}^{(CD)}\left(s_j'=-1\right)}{P_{ex}^{(CD)}\left(s_j'=+1\right)}\right) \tag{75}$$

另一個作法是以後驗 LLR $L^{(CD)}\left(s_j'\right)$ 扣去初始 LLR，這邊的初始 LLR 訊息包含通道相關訊息 $L_c\left(s_j'\right)$ 與事前資訊 $L_a\left(s_j'\right)$ 以及訊源解碼後的額外訊息 $L_{ex}^{(SD)}\left(s_j'\right)$。

$$L_{ex}^{(CD)}\left(s_j'\right)=L^{(CD)}\left(s_j'\right)-L_c\left(s_j'\right)-L_a\left(s_j'\right)-L_{ex}^{(SD)}\left(s_j'\right) \tag{76}$$

求出通道解碼額外訊息後，經過反交換器重新排序，當作訊源解碼的輸入起始值。如此一來，即完成一個完整的疊代過程。在第一次整體疊代時，迴旋通道碼的輸入訊息並沒有來自訊源解碼的額外資訊。只有進行到第二次疊代後，才會有反饋訊息回傳。因此綜

合以上結論，我們根據圖 4.6 歸類整體的解碼步驟如下：

1. 首先設定 $L_{ex}^{(SD)}\left(s_j'\right)=0$ 。
2. 傳遞通道相關訊息 $L_c\left(s_j'\right)$ 與位元層級的事前資訊 $L_a\left(s_j'\right)$ 以及訊源解碼輸出的額外訊息 $L_{ex}^{(SD)}\left(s_j'\right)$ 給通道解碼端，迴旋解碼器以 BCJR 演算法求出位元層級的通道解碼額外資訊 $L_{ex}^{(CD)}\left(s_j'\right)$ ，並經過反交換器後輸入到擴展式 LDPC 解碼器。
3. 擴展式 LDPC 訊源解碼採用對數域的加乘演算，變數點的初始值來自邊訊息 Y 與迴旋碼解碼後的額外訊息 $L_{ex}^{(CD)}\left(s_j\right)$ 與通道相關訊息 $L_c\left(s_j\right)$ 以及事前資訊 $L_a\left(s_j\right)$ ，其流程如下所示。

初始LLR

$$L_a(x)=(-1)^{y_i}\log\left(\frac{1-p}{p}\right)\begin{cases}o\\ \vdots \sim \text{共}n\text{個變數點(訊源部分)}\\ o\end{cases}$$

$$L_{ex}^{(CD)}\left(s_j\right)+L_c\left(s_j\right)+L_a\left(s_j\right)\begin{cases}o\\ \vdots \sim \text{共m=n-k個變數點(校驗子部分)}\\ o\end{cases}$$

訊息疊代演算(MPA)

$$L(x)\begin{cases}o\\ \vdots \sim \text{共}n\text{個變數點(訊源部分)}\\ o\end{cases}$$

$$L^{SD}\left(s_j\right)\begin{cases}o\\ \vdots \sim \text{共m=n-k個變數點(校驗子部分)}\\ o\end{cases}$$

後驗LLR

其中 $L(x)$ 用來硬性判定訊源 x ，而 $L^{(SD)}\left(s_j\right)$ 則用來計算訊源解碼的額外訊息。

$$L_{ex}^{(SD)}\left(s_j\right)=L^{(SD)}\left(s_j\right)-L_{ex}^{(CD)}\left(s_j\right)-L_c\left(s_j\right)-L_a\left(s_j\right)$$

4. 將 $L_{ex}^{(SD)}\left(s_j\right)$ 經過交換器後，得到 $L_{ex}^{(SD)}\left(s_j'\right)$ 並輸入到迴旋解碼器當初始 LLR。如此便完成一次完整的疊代。
5. 重複步驟 2，直到整體疊代次數最大值，最後計算重建後訊源的錯誤率。

在第 3 章我們曾介紹兩種虛擬通道，包含無記憶性的二位元對稱通道以及記憶性的 Gilbert 通道。因此在疊代訊源通道解碼的系統模擬中，我們也考慮當訊源 X 通過 Gilbert 通道產生邊訊息 Y，並與 BSC 通道的模擬結果比較。疊代解碼過程類似，不同處在於邊訊息 Y 本身的雜訊帶有記憶性，所以進行擴展式 LDPC 訊源解碼時，必須採用 3.3 章節的方法，將 Gilbert 通道的狀態資訊納入考慮，方能更精準判定錯誤的情況。所以邊訊息 Y 給予變數點的初始 LLR，會在每一次局部疊代之後做更新。至於校驗子的 LLR 與相同於 BSC 通道的情況，包含通道解碼的額外訊息與事前資訊以及通道相關訊息。

第5章　實驗模擬與結果分析

在前面的章節已介紹基於 LPDC 碼的分散式訊源編碼運算，接下來根據校驗子傳輸有無錯誤再分兩大區塊來進行模擬。首先 5.1 章節先說明傳輸無誤情況下，基本實驗環境的設定，之後針對兩種虛擬通道 BSC 與 Gilbert，各自探討其解碼的效能。5.2 節則是在有傳輸錯誤情況下，疊代式合併訊源通道解碼在 BSC 與 Gilbert 兩種虛擬通道的模擬結果。

5.1　傳輸無誤的實驗環境設定

整體架構如圖 3.3 所示，訊源 X 與邊訊息 Y 為互相獨立且均勻分布的二位元訊號，以 LDPC 碼製作 DSC 編碼的校驗子生成器，解碼則採用對數域的加乘演算法。至於訊源相關模型，則分別考慮 BSC 及 Gilbert 兩種虛擬通道。

5.1.1　BSC 虛擬通道

[1] 參數設定:

訊源 X 與邊訊息 Y 的關聯性，是以二位元對稱通道來模擬，並考慮各種不同的交叉錯誤率 p，同時參考[14]的數據設定長度為 10000，並採用碼率為 1/2 的規則與不規則 LDPC 碼，其中前者規則的程度分布 $(w_c, w_r) = (3.6)$。不規則的程度分布表示如下，以 density-evolution 做變數點程度的最佳化，其中程度最大值為 20：

$$\lambda(x) = 0.4579x + 0.3238x^2 + 0.0214x^3 + 0.0593x^5 + 0.0389x^6$$
$$+0.0248x^7 + 0.0088x^8 + 0.0177x^{18} + 0.0474x^{19}$$
$$\rho(x) = 0.6748x^7 + 0.3252x^8$$

以 100 次疊代為最大限度，進行 100 組訊源解碼，之後求得平均位元錯誤率(bit error rate,BER)。至於不同 BSC 的交叉錯誤率 p 所對應的熵值 $H(X|Y) = H(p)$，結果顯示如圖 5.1。

[2] 實驗結果：

	BSC 交叉錯誤率	熵值 H(p)	解碼後 BER
規則 LDPC 碼	0.085	0.4196	0.027
	0.08	0.4022	0.0045
	0.076	0.3879	0.000358
	0.074	0.3807	0.0000344
不規則 LDPC 碼	BSC 交叉錯誤率	熵值 H(p)	解碼後 BER
	0.09	0.4364	0.0003226
	0.092	0.4431	0.0016
	0.094	0.4497	0.0058
	0.096	0.4562	0.0135
	0.098	0.4626	0.0334

表 5.1　基於 LDPC 碼的 DSC 模擬數據

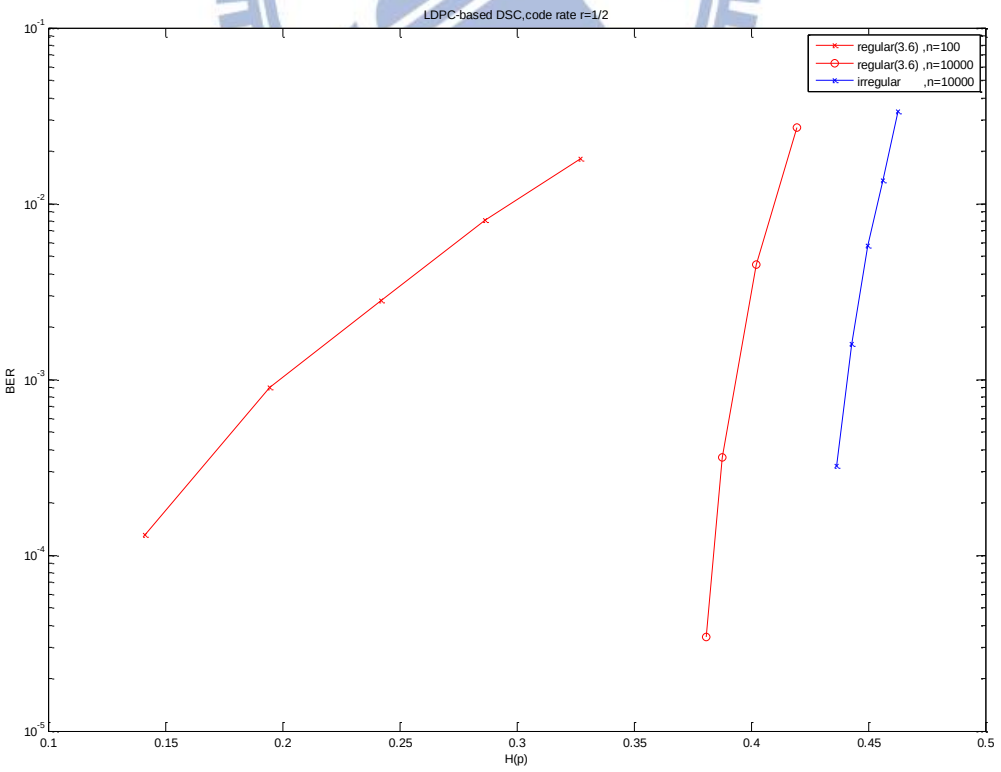


圖 5.1　基於 LDPC 碼的 DSC 模擬結果

[3] 結果分析與討論:

在相同長度且碼率一樣的情況下，經過最佳化處理的不規則 LDPC 碼之效果較好，從圖 5.1 來看，同樣的 BER，不規則碼的熵值比規則碼的大許多，代表在相同的 BSC 虛擬通道下，修正能力相對較強，原因與矩陣內 1 的分布情況有關。另外我們也模擬不同長度的 LDPC 碼對於分散式訊源編碼的影響，長度越長解碼效果越好。這也反應在線性區塊碼的最短距離上，隨著 n 變大，LDPC 碼的最短距離越大。因此，LDPC 碼運用在分散式訊源編碼，遠比渦輪碼(turbo code)更能接近 Slepian-Wolf 壓縮極限。

5.1.2 Gilbert 虛擬通道

[1] 參數設定：

整體架構以及訊源 X 的產生與之前的設定相同，但邊訊息 Y 改以 Gilbert 通道產生，且狀態轉移機率固定為 $g=b=0.01$。我們改變差狀態下的錯誤率 P_e^b，並算出對應的平均通道錯誤率 $\overline{P}_e$。LDPC 碼則是採用長度為 2000，碼率為 1/2 的規則分布 $(w_c, w_r)=(3.6)$。

[2] 實驗結果：

	虛擬通道平均 BER	解碼後 BER
修正的 LDPC 解碼	0.25	0.233
	0.15	0.1185
	0.1	0.033
	0.075	0.0099
	0.06	0.0012
	0.05	0.000261
傳統 BSC 的 LDPC 解碼	虛擬通道平均 BER	解碼後 BER
	0.25	0.2441
	0.15	0.1454
	0.1	0.0723
	0.075	0.0224
	0.06	0.0037
	0.05	0.0006275

表 5.2　Gilbert 虛擬通道的 LDPC-DSC 解碼數據

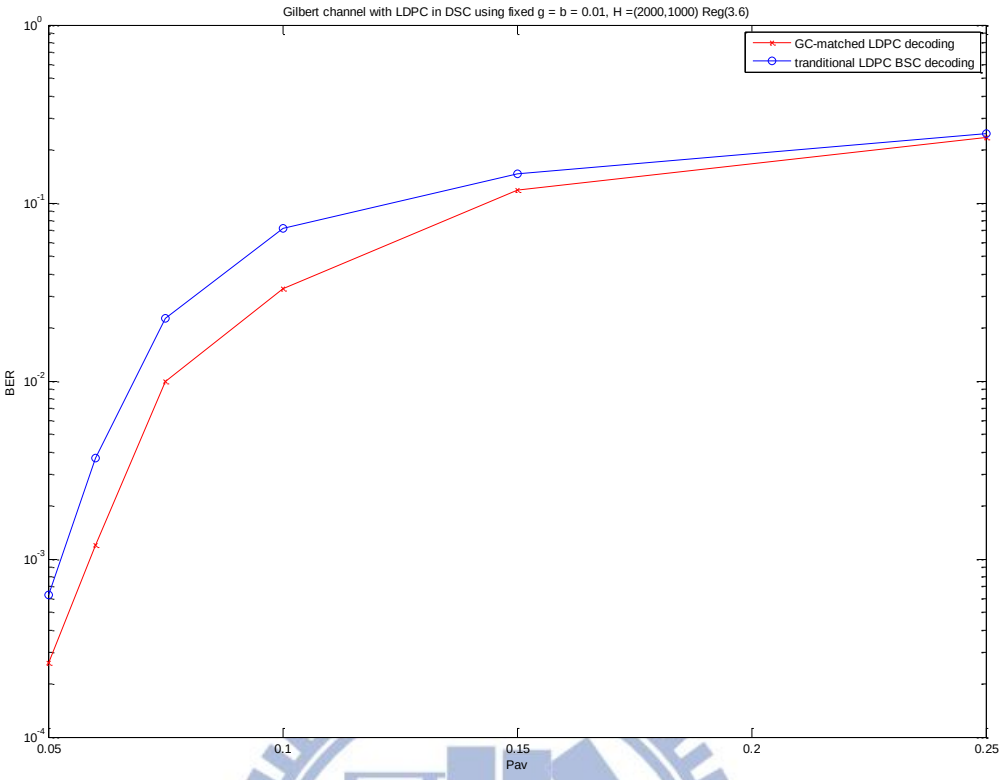


圖 5.2 Gilbert 虛擬通道的 LDPC-DSC 解碼結果

[3] 結果分析與討論:

如圖 5.2 所示，我們以兩種不同方式進行分散式訊源解碼。其一為傳統 BSC 解碼，也就是虛擬通道特性未知的情況下，將錯誤序列當成無記憶性並仿照 3.2 章節方式解碼；另一為 3.3 節提到的「修正」疊代訊息解碼，接收端事先已知 Gilbert 虛擬通道的模型參數，並針對記憶性錯誤序列做狀態估計。結果顯示，修正的 LDPC 疊代訊息解碼(又稱 Gilbert-matched LDPC decoding)，效能上會比傳統 BSC 疊代訊息解碼更好。原因在於我們對記憶性錯誤序列做額外的狀態估測，因此解碼端更能掌握錯誤發生的相關性，也有助於還原出訊源 X。

5.2 傳輸有誤的實驗環境設定

整體架構如圖 4.6 所示，訊源 X 與邊訊息 Y 為互相獨立且均勻分布的二位元訊號，訊源編碼器利用 LDPC 碼產生其校驗子，解碼採用對數域的加乘演算法；通道編碼器則採用遞迴式系統迴旋碼，解碼使用 BCJR 演算法。

5.2.1 BSC 虛擬通道

[1] 參數設定:

訊源 X 與邊訊息 Y 的關聯性，我們以二位元對稱通道來模擬，並以[16]上的數據做為參考，擴展式 LDPC 碼是基於$(n,k)=(2000,1000)$的規則 LDPC 碼，其程度分布$(w_c,w_r)=(3.6)$。遞迴式系統迴旋碼的碼率為 1/2，其生成矩陣$G(D)=\left[1,\ \frac{1+D^2}{1+D+D^2}\right]$。BSC 的交叉錯誤率固定為 0.0533，通道 SNR 介於-5 到 5dB 之間，以(g,l)代表整體疊代與局部疊代最大次數，進行 100 組訊源通道解碼，並求得平均錯誤率(bit error rate,BER)。如圖 5.3 所示。

[2] 實驗結果：

解碼器	通道 SNR(dB)					
	-5	-4	-3	-2	-1	0
BCJR 1 iter.	0.04471	0.038935	0.02948	0.02067	0.014545	0.011625
(1,6)	0.048625	0.042805	0.03064	0.016095	0.00553	0.002215
(1,12)	0.049195	0.041735	0.029585	0.009	0.000235	0
BCJR 2 iter.	0.03525	0.025795	0.01815	0.01344	0.01118	0.010525
(2,6)	0.04425	0.032875	0.01648	0.00641	0.002265	0.00158
(2,12)	0.044875	0.034175	0.01273	0.00116	0	0
解碼器	通道 SNR(dB)					
	1	2	3	4	5	
BCJR 1 iter.	0.01053	0.01021	0.010175	0.01017	0.01017	
(1,6)	0.001505	0.0013	0.00128	0.00125	0.00125	
(1,12)	0	0	0	0	0	
BCJR 2 iter.	0.01027	0.01021	0.01017	0.01017	0.01017	
(2,6)	0.001315	0.00126	0.00125	0.00125	0.00125	
(2,12)	0	0	0	0	0	

表 5.3 ISCD 模擬數據

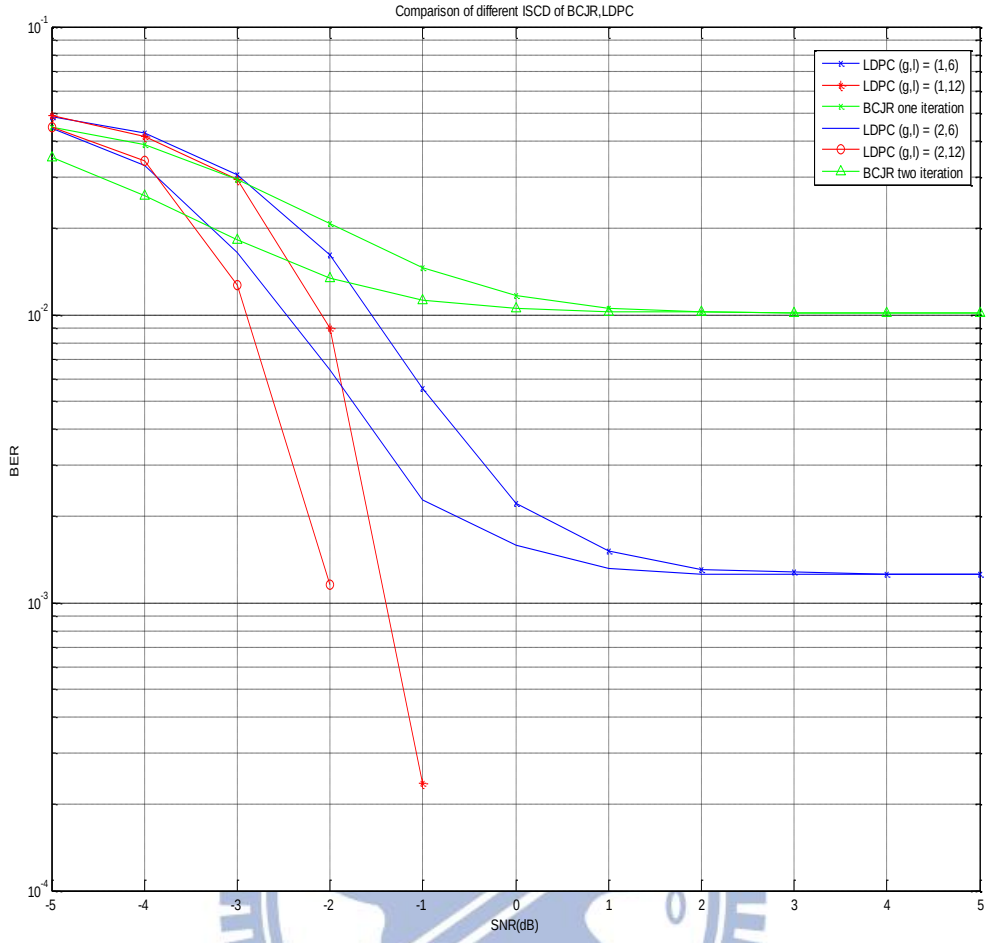


圖 5.3　ISCD 模擬結果

[3] 結果分析與討論:

針對長度為 2000 的 LDPC 碼，我們選擇了規則的分布方式，主因在[22]中表示此短長度的 LDPC 碼，規則的效能會比不規則的好。另外，我們也將訊源 LDPC 碼改成同碼率且相同長度的遞迴式系統迴旋碼來做為比較。由於 LDPC 碼解碼能力遠優於迴旋碼，因此在 SNR 為-1dB 處，兩者曲線瞬間產生差距。再者，同樣都以 LDPC 碼為訊源編解碼端，也會因(g,l)疊代次數不同而有不一樣的結果。舉例來說，固定整體疊代次數g，局部疊代次數l越大，其錯誤率相較下越小，因為 LDPC 碼內部訊息交換使得校驗子正確性越高，進而影響整個系統效能。反之，固定l，改變g的次數，次數越多效果也越好，原因在於通道與訊源之間的額外訊息會正向幫助系統提升解碼能力。

5.2.2 Gilbert 虛擬通道

[1] 參數設定：

合併式疊代訊源通道解碼架構同上一小節，邊訊息 Y 改以 Gilbert 通道產生。根據相關 Gilbert 通道參數研究以及方便模擬，狀態轉移機率固定為 $g = b = 0.01$，另外通道平均錯誤率固定為 $\overline{P}_e = 0.05$，擴展式 LDPC 訊源碼與遞迴式系統迴旋碼的參數與前一小節一樣，結果如圖 5.4 所示。

實驗結果：

解碼器	通道 SNR(dB)					
	-5	-4	-3	-2	-1	0
GC(2,6)	0.036563	0.025983	0.013536	0.006596	0.003836	0.003057
BSC(2,6)	0.038521	0.027588	0.015364	0.007394	0.004453	0.003553
GC(2,12)	0.034981	0.02231	0.009009	0.00205	0.000589	0.000355
BSC(2,12)	0.038053	0.026484	0.012499	0.003395	0.001018	0.000542
解碼器	通道 SNR(dB)					
	1	2	3	4	5	
GC(2,6)	0.002832	0.002802	0.002792	0.00279	0.00279	
BSC(2,6)	0.003337	0.003252	0.003255	0.003255	0.003255	
GC(2,12)	0.000333	0.00033	0.000323	0.000323	0.000323	
BSC(2,12)	0.000495	0.00048	0.00048	0.00048	0.00048	

表 5.4　Gilbert 虛擬通道的 ISCD 數據

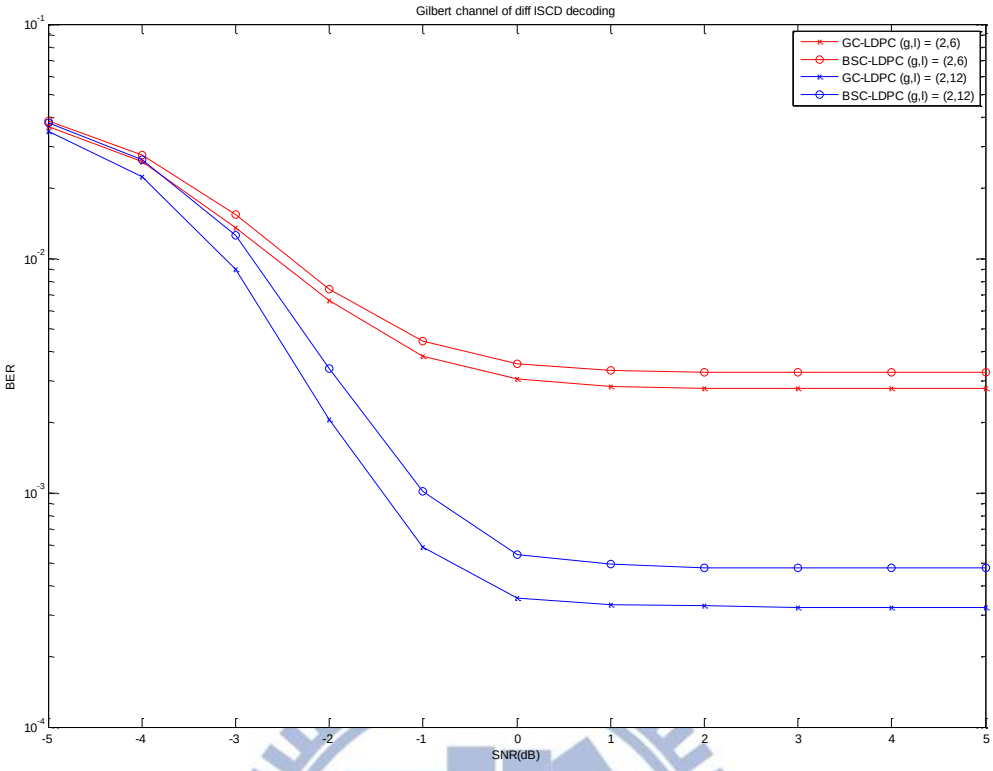


圖 5.4　　Gilbert 虛擬通道的 ISCD

[3] 結果分析與討論:

由於 Gilbert 虛擬通道產生的錯誤序列具有記憶性，所以理當採用 Gilbert-matched LDPC decoding 為基礎，並將矩陣擴展以對抗校驗子的錯誤情形。在此，同 5.1.2 小節的方式，我們將訊源解碼端以兩種方式來做解碼。一為傳統 BSC 的解碼，另一為 Gilbert-matched LDPC 解碼。結果顯示，有將錯誤序列額外狀態資訊考慮進去會提升解碼效能。另外(g,l)的疊代次數多寡，也與前一小節結果相似。

第6章　結論與未來展望

LDPC 碼，為一線性區塊通道碼，且對於資料傳輸與通道儲存提供了接近容量的效能。由 Gallager 在 1960 年的博士論文[1]中首次被提出，卻一直到近十年來才被熱烈討論，主要是當時 VLSI 技術尚未成熟。LDPC 碼產生容易，例如隨機建構方式，且解碼演算複雜度與碼的區塊長度呈線性關係，再加上其可觀的效能，使得 LDPC 碼有許多不同的應用，例如 2003 年，英特爾提議在 10GBASE-T 標準中採用該方式，而制定該規格的 IEEE802.3an 工作小組於 2004 年 8 月通過採用 LDPC 碼。另外，2004 年 6 月訂定的 HDTV 衛星電視廣播規格標準 DVB-S2(Digital Video Broadcasting Satellite Version2)，也採用 LDPC 碼；日本電信電話株式會社(NTT Communications)在 2004 年 12 月開始導入的影像配訊服務「OCN Theater」，也是決定採用 LDPC 碼。因此無論是通訊的高速化、通訊距離的延長還是傳送品質的提升，都可以從 LDPC 碼來獲得效果。由於 LDPC 碼具有優異的錯誤修正機能，即編碼增益(Coding Gain)，可以高速大容量化並改善惡劣的傳送品質，因此無線網路的高速化、高速行動通訊、數位衛星廣播等用途可獲得改善，而量子密碼的通訊在惡劣的傳送迴路下也無須重送。

基於 LDPC 碼的 DSC 模擬結果顯示，整體效能比渦輪碼更好，且接近 Slepian-Wolf 極限。除此之外，在長區塊的 LDPC 矩陣下，我們會以不規則的分布作為解碼首選，而程度分布多樣化，如何求其最佳分布方式也為一研究議題。但由於此先期研究使用的訊源與邊訊息皆為人工合成，在實際應用上的效果有待進一步驗證。另外，訊源相關模型都假設為均勻分布來進行模擬，對此非均勻分布的研究較貼近現實通訊環境，如何運用非均勻訊源的資訊來幫助解碼亦為一研究方向。以遠距醫療看護系統而言，心電圖訊源的 DSC 架構[23]，主要是根據分散式訊源編碼理論，兩個有相關性的訊源在各自在獨立編碼的情況下，仍能以合併解碼模式取得較低的理論熵值。以此針對 MIT-BIH 資料庫的多導極心電訊號，分析建立其訊源相關模型並視之為虛擬傳輸通道。若以 LDPC 碼做為校驗子生成器，必須面對碼字長度的問題與如何運用心電訊源分布資訊來幫助解碼。若得以解決，其效能或許會超越以迴旋碼設計的校驗子生成機制。另外，如何判定虛擬通道的記憶特性亦為重要研究議題，未來將發展相關訊源的虛擬通道模型化技術，並採用適當的通道匹配解碼方式。

DSC 的虛擬通道，除了 BSC 與 Gilbert 之外，二位元非對稱通道(binary asymmetric channel, BAC)也為一值得深入探討的議題。由於 0 與 1 各自的交叉錯誤率不同，迫使

LDPC 碼在設定軟性初始值時必須做修正。如何找出適合的方式建立初始 LLR，且若搭配非均勻訊源其複雜度值得研究。針對 Gilbert 虛擬通道，雖然我們考慮了狀態點的訊息，結果也證實能提昇解碼效益，不過與傳統 BSC 解碼之間差異不大，代表 LDPC 碼本身並不適用於狀態估測。如何修正解碼演算方式或者加強矩陣本身對於記憶性通道判斷力，都是可以努力的方向。

ISCD 模擬結果顯示，疊代次數多寡與效能成正向關係，且 LDPC 訊源解碼器優於傳統迴旋碼，此外我們也可以 LDPC 碼替換通道迴旋碼建構基於 LDPC 碼的訊源通道解碼。另外，擴展式 LDPC 矩陣本身並不是低密度分布，因此針對矩陣本身做低密度化或改採同樣長度的 LDPC 矩陣來進行解碼，效能上也許能進一步提升。大幅減少運算時間的研究也是重要議題，這點對於即時通訊系統相當重要。最後，LDPC 碼本身仍存在許多有待開發的潛能，建立一高效能低複雜度的解碼方式依舊是最熱門的議題。

參考文獻

[1] R. G. Gallager, "*Low-Density Parity-Check Codes*," Cambridge, MA: MIT Press, 1963.

[2] V. Zyablov and M. Pinsker, "Estimation of the Error-Correction Complexity of Gallager Low-Density Codes," *Probl. Pered. Inform.*, vol. 11, pp. 23–26, Jan. 1975.

[3] G. A. Margulis, "Explicit Construction of Graphs without Short Cycles and Low Density Codes," *Combinatorica*, vol. 2, no. 1, pp. 71–78, 1982.

[4] R. M. Tanner, "A Recursive Approach to Low Complexity Codes," *IEEE Trans. Inf.Theory*,vol. IT-27, no. 5, pp. 533–547, Sep. 1981.

[5] M. Sipser and D. A. Spielman, "Expander codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 1710–1722, Nov. 1996.

[6] D. J. C. MacKay and R. M. Neal, "Near Shannon Limit Performance of Low Density Parity Check Codes," *Electron. Lett.*, vol. 32, pp. 1645–1646, Aug. 1996.

[7] D. J. C. MacKay, "Good Error Correcting Codes Based on Very Sparse Matrices," *IEEE Trans. Inf. Theory*, vol. 45, no. 2, pp. 399–431, Mar. 1999.

[8] S.Y. Chung, J. G. D. Forney, T. Richardson, and R. Urbanke, "On the Design of Low-Density Parity-Check Codes within 0.0045 dB of the Shannon Limit," *IEEE Commun.Lett.*, vol. 5, no. 2, pp. 58–60, Feb. 2001.

[9] X.Y. Hu, E. Eleftheriou, and D.M. Arnold, "Progressive Edge-Growth Tanner Graphs," *Proc. IEEE Global Telecommunications Conf. (GLOBECOM)*, San Antonio, TX, Nov. 2001, pp. 995–1001.

[10] X.Y. Hu, E. Eleftheriou, and D.M. Arnold, "Regular and Irregular Progressive Edge-Growth Tanner Graphs," *IEEE Trans. Inf. Theory*, vol. 51, no. 1, pp. 386–398, Jan. 2005.

[11] D. Slepian and J. K. Wolf, "Noiseless Coding of Correlated Information Sources," *IEEE Trans. Inform. Theory*, vol. 19, pp. 471-480, Jul. 1973.

[12] S. S. Pradhan and K. Ramchandran, "Distributed Source Coding Using Syndromes (DISCUS): design and construction," *Proc. of Data Compression Conf.(DCC)*, pp. 158-167, Mar. 1999.

[13] T. Murayama, "Statistical Mechanics of Linear Compression Codes in Network Communication," *Europhysics Lett.*, 2001. Preprint

[14] A. D. Liveris, Z. Xiong and C. N. Geoghiades, "Compression of Binary Sources with Side Information Using Low-Density Parity-Check Codes,"2002.

[15] A. W. Eckford, F. R. Kschischang and S. Pasupathy, "Analysis of Low-Density Parity-Check Codes for the Gilbert-Elliott Channel, " *IEEE Trans.Inform. Theory*, vol.

51, no. 11, pp. 3872-3889, Nov. 2005.

[16] R. Hu, R. Viswanathan, and J. Li, "A New Coding Scheme for the Noisy-Channel Slepian-Wolf Problem: Separate Design and Joint Decoding," *Proc. Globecom'04*, Nov. 2004.

[17] David MacKay's Gallager code resources, "Source Code for Progressive Edge Growth Parity-Check Matrix Construction, " [Online]. Available: http://www.cs.toronto.edu/~mackay/S0.html#PEG_ECC.html.

[18] Peiyu Tan, Kai Xie, and Jing Li, "Slepian-Wolf Coding Using Parity Approach and Syndrome Approach," in Proceeding of 41st Annual Conference on Information Sciences and Systems, March 14-16 2007, Baltimore, MD, pp. 708-713.

[19] H.S. Wang and N. Moayeri, "Finite-state Markov Channel - A Useful Model for Radio Communication Channels, " *IEEE Trans. Vehicular Tech.*, vol. 44,no. 1, pp. 163-171, Feb. 1995.

[20] H.S. Wang and P.C. Chang, "On Verifying the First-Order Markovian Assumption for a Rayleigh Fading Channel Model, " *IEEE Trans. Vehicular Tech.*, vol. 45, no. 2, pp. 353-357, May 1996.

[21] L.R. Bahl, J. Cocke, F. Jeinek and J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate, " IEEE *Trans. Inform. Theory*, vol. IT-20, pp. 248-287, March 1974.

[22] D. J. C. MacKay and M. C. Davey, "Evaluation of Gallager Codes for Short Block Length and High Rate Applications," *Proc. IMA Workshop on Codes, Systems and Graphical Models*, pp. 113-130, 1999.

[23] 陳柏強，「基於分散式訊源編碼架構的心電圖儀」，國立交通大學碩士論文，民國一百年。