การเปรียบเทียบเทคนิคของรหัสข้อมูลสำหรับการสื่อสารไร้สาย

Comparison of Coding Technique for Wireless Communications

นาย ดุลวิทย์ กิตติชัยวัชร์

นาย ปิยะภูมิ อัศวธิตานนท์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมสารสนเทศ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2550

การเปรียบเทียบเทคนิคของรหัสข้อมูลสำหรับการสื่อสารไร้สาย

Comparison of Coding Technique for Wireless Communications

โดย

นาย ดุลวิทย์ กิตติชัยวัชร์

นาย ปิยะภูมิ อัศวธิตานนท์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมสารสนเทศ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2550

Comparison of Coding Technique for Wireless Communications

BY

Mr. Dulravit Kittichaiwat

Mr. Piyapoom Asavathitanonta

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENT FOR THE DEGREE OF

BACHELOR IN DEPARTMENT OF INFORMATION ENGINEERING

FACULTY OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2007

หัวข้อปริญญานิพนธ์	การเปรียบเทียบเทคนิคของรหัสข้อมูลสำหรับการสื่อสารไร้สาย
ชื่อนักศึกษา	นายคุลวิทย์ กิตติชัยวัชร์ รหัสนักศึกษา 47010256
	นายปิยะภูมิ อัศวธิตานนท์ รหัสนักศึกษา 47010461
อาจารย์ที่ปรึกษา	ดร. จักรี ทีฆภาคย์วิศิษฏ์
ระดับการศึกษา	ปริญญาตรี วิศวกรรมศาสตรบัณฑิต
	สาขาวิศวกรรมสารสนเทศ
ภาควิชา	วิศวกรรมสารสนเทศ
ปีการศึกษา	2550

ปริญญานิพนธ์นี้ได้รับความเห็นชอบจากอาจารย์ที่ปรึกษาเป็นที่เรียบร้อยแล้ว

.....

(ดร. จักรี ทีฆภาคย์วิศิษฏ์)

อาจารย์ที่ปรึกษา

หัวข้อปริญญานิพนธ์	การเปรียบเทียบเทคนิคของรหัสข้อมูลสำหรับการสื่อสารไร้สาย
ชื่อนักศึกษา	นายดุลวิทย์ กิตติชัยวัชร์ รหัสประจำตัว 47010256
	นายปิยะภูมิ อัศวธิตานนท์ รหัสประจำตัว 47010461
อาจารย์ที่ปรึกษา	ดร.จักรี ทีฆภาคย์วิศิษฏ์
ระดับการศึกษา	ปริญญาตรี วิศวกรรมศาสตร์บัณฑิต
	สาขาวิศวกรรมสารสนเทศ
ภาควิชา	วิศวกรรมสารสนเทศ
ปีการศึกษา	2550

บทคัดย่อ

การเข้ารหัสช่องสัญญาณ ใช้เพื่อเพิ่มประสิทธิภาพของระบบการสื่อสารแบบไร้สาย โดยมีการใช้อัลกอริทึมในการถอดรหัส ที่หลายหลายแบบ และสามารถนำมาใช้ประโยชน์ได้ในระบบการสื่อสารแบบไร้สาย

ในโครงงานนี้การได้ทำการเปรียบเทียบประสิทธิภาพของเทคนิคการถอดรหัสแบบ Viterbi MAP และ Log-MAP ที่ใช้ในการถอดรหัส Convolutional Code และรวมถึงรูปแบบการเข้ารหัส-ถอดรหัส Turbo Codes ของระบบการสื่อสารแบบไร้สาย ภายใต้การสื่อสารผ่านช่องสัญญาณแบบ AWGN

Thesis Title	Comparison of Coding Technique for Wireless Communications
Student	Mr. Dulravit Kittichaiwat ID. 47010256
	Mr. Piyapoom Asavathitanonta ID. 47010461
Advisor	Dr. Chakree Teekapakvisit
Graduate Level	Bachelor Degree of Information Engineering
Department	Information Engineering
Academic Year	2007

Abstract

It is well known that channel coding is used to increase the performance of wireless communication systems. There are various decoding algorithms have been proposed and utilized in such systems.

In the project, the performance comparison of decoding techniques such as Viterbi, MAP and Log-MAP algorithm including Turbo codes of the convolutional code for wireless communication system under AWGN channel is investigated.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้คงมิอาจสำเร็จได้ ถ้าปราศจากความร่วมมืออย่างดียิ่งจากทุกฝ่ายที่เกี่ยวข้อง ซึ่งผู้จัดทำใคร่ขอขอบคุณทุกๆท่านที่ได้มีส่วนช่วยเหลือ แนะนำ ให้คำปรึกษา ในทุกๆด้าน

ขอขอบพระคุณ อาจารย์ จักรี ทีฆภาคย์วิศิษฏ์ ที่ได้ช่วยเหลือ ให้คำปรึกษา และให้ข้อเสนอแนะที่เป็นประโยชน์ รวมทั้งเอื้อเฟื้อข้อมูลต่างๆในการจัดทำโครงงาน จึงทำให้ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงไปด้วยดี

ขอขอบคุณพ่อและคุณแม่ ที่เป็นกำลังใจให้เสมอมา ขอบคุณเพื่อนภาคทุกๆคน ที่คอยช่วยเหลือทุกอย่าง คุณประโยชน์อันพึงมีจากโครงงานนี้ ทางผู้จัดทำขอมอบแด่ผู้มีพระคุณทุกท่านไว้ ณ โอกาสนี้

นาย ดุลวิทย์ กิตติชัยวัชร์

นาย ปิยะภูมิ อัศวธิตานนท์

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญรูป

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

บทที่ 1
บทนำ

1.1 ความเป็นมาของปริญญานิพนธ์

ในปัจจุบันการสื่อสารไร้สายได้เข้ามามีบทบาทสำคัญในชีวิตประจำวันของมนุษย์ โดยที่ความต้องการของผู้บริโภคในการรับส่งข้อมูลได้เพิ่มขึ้นอย่างต่อเนื่องและหลากหลาย จากเดิมข้อมูลที่ทำการรับส่งอาจเป็นสัญญาณพูดหรือข้อความสั้นๆเท่านั้น แต่ในปัจจุบันแนวโน้มความต้องการในการรับส่งข้อมูลของผู้บริโภคได้ขยายออกไปในวงกว้างยิ่งขึ้นกล่าวคือมีแนวโน้มในการรับส่งสัญญาณที่เป็นมัลติมีเดีย ซึ่งสัญญาณแบบมัลติมีเดียนั้นต้องการอัตราข้อมูลและความถูกต้องในการรับส่งข้อมูลสูง อีกทั้งต้องคำนึงถึงการใช้แบนด์วิดท์ซึ่งมีจำกัดให้เกิดประสิทธิภาพสูงสุดอีกด้วย ดังนั้น จึงได้มีการคิดค้นและพัฒนาเทคโนโลยีในการสื่อสารรูปแบบต่างๆ เพื่อให้สามารถรองรับความต้องการที่เพิ่มขึ้นนี้

ในการออกแบบหรือใช้งานระบบสื่อสารแบบดิจิตอลนั้น จะต้องมีการพิจารณาถึงองค์ประกอบในหลายๆส่วนด้วยกัน โดยสิ่งหนึ่งที่จะต้องมีการพิจารณาก็คือ ข้อมูลดิบที่ถูกส่งจากต้นทางไปถึงปลายทางนั้นมีข้อมูลที่เกิดความผิดพลาดขึ้นหรือไม่ ที่เกิดจากสาเหตุต่างๆหลายสาเหตุด้วยกัน โดยที่สาเหตุหลักที่จะทำให้เกิดความผิดพลาดดังกล่าวคือ การที่ระบบสื่อสารนั้นถูกรบกวนจากสัญญาณรบกวนต่างๆ ถ้าหากว่าขนาดของสัญญาณรบกวนที่เกิดขึ้นในระบบสื่อสารนั้นมีค่าที่สูง จะส่งผลให้อัตราการเกิดความผิดพลาดของข้อมูล (Bit Error Rate) ที่เกิดขึ้นมีค่าสูงตามไปด้วย ในการที่จะลดอัตราการเกิดความผิดพลาดของข้อมูลให้มีค่าที่ลดลงนั้น สามารถทำได้หลายรูปแบบด้วยกัน เช่น เพิ่มกำลังของเครื่องส่ง, การทำให้ขนาดของสัญญาณรบกวนมีค่าน้อยลง หรือ การเข้ารหัสข้อมูลเป็นต้น

สำหรับการลดอัตราการเกิดความผิดพลาดที่เกิดขึ้นในการส่งข้อมูลด้วยวิธีการเข้ารหัส (Coding) นั้น จะเป็นการนำข้อมูลดิบที่จะทำการส่งผ่านระบบสื่อสารที่เป็นข้อมูลแบบดิจิตอล มาทำการผ่านกระบวนการ “เข้ารหัส” หรือ “Encoding” เพื่อเปลี่ยนรูปแบบของข้อมูลที่จะถูกส่งผ่านระบบสื่อสารให้อยู่ในรูปแบบที่สามารถนำข้อมูลมาทำการแก้ไขความผิดพลาดของข้อมูลที่เกิดขึ้นเนื่องจากการถูกรบกวนจากสัญญาณรบกวนต่างๆได้ โดยที่ข้อมูลที่ได้จากการทำงานนั้นจะเป็นข้อมูลที่จะถูกส่งออกไปผ่านระบบสื่อสาร และเมื่อข้อมูลดังกล่าวถูกส่งมาถึงปลายทาง จะมีการนำข้อมูลที่รับได้นั้นมาผ่านกระบวนการ “ถอดรหัส” หรือ “Decoding” เพื่อเปลี่ยนรูปแบบของข้อมูลที่รับได้ให้กลับมาอยู่ในรูปของข้อมูลดิบพร้อมทั้งทำการแก้ไขข้อมูลที่คาดว่าจะเกิดความผิดพลาดขึ้น

ให้มีค่าที่ถูกต้อง โดยที่รูปแบบหรือวิธีการที่ใช้ในการเข้ารหัสข้อมูลนั้น จะมีอยู่หลายวิธีการด้วยกัน ซึ่งก่อนที่จะมีการค้นพบการเข้ารหัสแบบ Turbo Codes วิธีการเข้ารหัสแบบ Convolution Codes จะเป็นรูปแบบในการเข้ารหัสข้อมูลที่มีการนิยมใช้งานมากที่สุด โดยจะมีการนิยมใช้วิธีการถอดรหัสแบบ Map Decoding , Log-Map Decoding และ Viterbi Decoding ในการถอดรหัสข้อมูลสำหรับ Convolution Codes

ในกรณีของการเข้ารหัสแบบ Turbo Codes นั้น จะเป็นรูปแบบในการเข้ารหัสข้อมูลในระบบสื่อสารที่มีการค้นพบในปี พ.ศ. 2536(ค.ศ. 1993) โดย Claude Berrou , Alian Glavieux และ Punya Thitimajshima โดยจะเป็นรูปแบบในการเข้ารหัสข้อมูลที่มีการพัฒนาให้มีความสามารถในการป้องกันความผิดพลาดของข้อมูลที่สูงขึ้น โดยในการเข้ารหัสแบบ Turbo Codes จะมีการนำวิธีการเข้ารหัสข้อมูลแบบ Parallel Concatenate และวิธีการถอดรหัสแบบ Iterative Decoding มาใช้ในการทำงานเพื่อให้ได้รูปแบบในการเข้ารหัสข้อมูลที่มีความสามารถในการป้องกันความผิดพลาดของข้อมูลได้สูงขึ้น โดยที่ไม่มีรูปแบบของวงจรที่มีการทำงานที่ซับซ้อนมากขึ้น ผลลัพธ์ที่ได้จากการเข้ารหัสแบบ Turbo Codes นั้น สามารถที่ทำให้ข้อมูลต่างๆที่ถูกส่งผ่านระบบสื่อสาร มีอัตราการเกิดความผิดพลาดที่มีค่าน้อยลงได้ โดยไม่มีการเพิ่มกำลังของเครื่องส่งให้มีค่าสูงขึ้น ซึ่งจะส่งผลให้สามารถส่งข้อมูลโดยใช้กำลังส่งที่ลดลงได้ และสำหรับระบบสื่อสารที่มีการนำหลักการของ Turbo Codes มาทำการใช้งานนั้น จะสามารถลดความต้องการค่าอัตราส่วนระหว่างกำลังส่งข้อมูลต่อกำลังของสัญญาณรบกวน หรือ S/N ของภาคถอดรหัสลงได้จนมีค่าที่ใกล้เคียงกับค่าในทฤษฎีของ Shannon

1.2 วัตถุประสงค์ของปริญญานิพนธ์

ทดสอบประสิทธิภาพและเปรียบเทียบการถอดรหัสข้อมูลของแต่ละเทคนิคการถอดรหัสในระบบการสื่อสารแบบไร้สาย

1.3 ขอบเขตของปริญญานิพนธ์

1.3.1 ศึกษาและจำลองการเข้ารหัสแบบ Convolutional Code และการถอดรหัสแบบ MAP Coding, Log-Map Coding และ Viterbi Coding ที่นำไปประยุกต์ใช้ในระบบสื่อสารแบบไร้สาย

1.3.2 ศึกษาและจำลองการเข้ารหัสและถอดรหัสด้วย Turbo Codes แบบ Viterbi Coding และ Log-Map Coding ที่นำไปประยุกต์ใช้ในระบบสื่อสารแบบไร้สาย

1.4 เนื้อหาของปริญญานิพนธ์

บทที่ 2 กล่าวถึงทฤษฎีที่นำมาใช้ในการออกแบบและพัฒนาการเข้ารหัสแบบ Convolution และการถอดรหัสแบบ Map Decoding, Log-Map Decoding และ Viterbi Decoding และการเข้ารหัส-ถอดรหัสในรูปแบบ Turbo codes

บทที่ 3 กล่าวถึงการออกแบบจำลองโครงสร้างของการเข้ารหัสข้อมูลแบบ Convolution และการถอดรหัสข้อมูลแบบ Map Decoding, Log-Map Decoding และ Viterbi Decoding และการเข้ารหัส-ถอดรหัสแบบ Turbo codes

บทที่ 4 กล่าวถึงการทดลองและผลของการทดลองของการเข้ารหัสข้อมูลแบบ Convolution และการถอดรหัสข้อมูลแบบ Map Decoding และ Viterbi Decoding และการเข้ารหัส-ถอดรหัสแบบ Turbo Codes โดยใช้ Viterbi Decoding และ Log-Map Decoding ในการถอดรหัส

บทที่ 5 กล่าวถึงข้อสรุปที่ได้จากการทดลอง และ ปัญหาที่พบในการดำเนินโครงงานรวมถึงแนวทางแก้ไขและปรับปรุง

บทที่ 2
หลักการและทฤษฎีที่เกี่ยวข้อง

2.1 หลักการพื้นฐาน

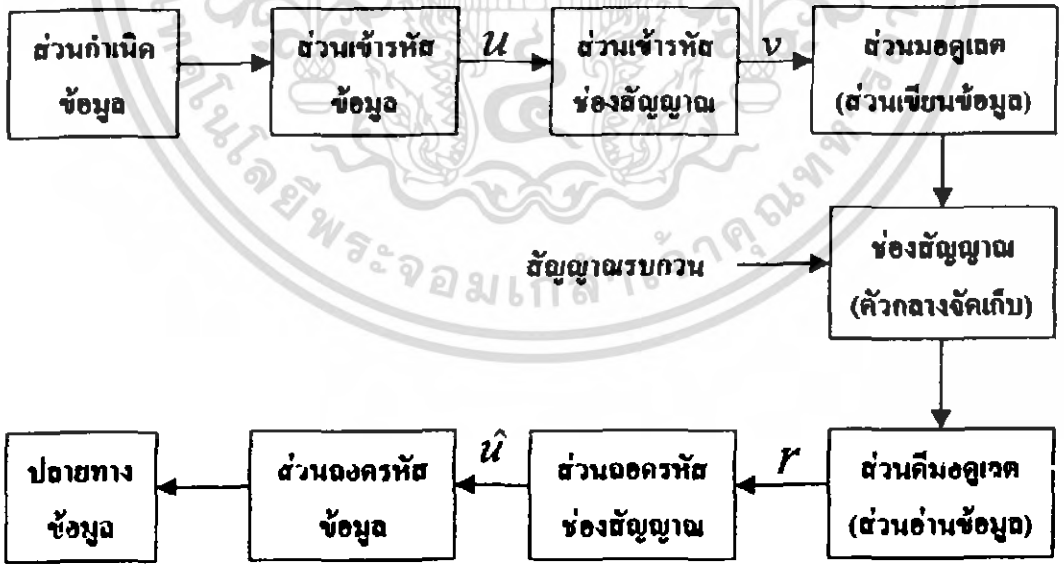
สำหรับการพัฒนาระบบสื่อสารแบบดิจิตอลนั้น จะมีรูปแบบในการพัฒนาในหลายแนวทางด้วยกัน และสำหรับแนวทางหนึ่งที่มีการพิจารณา ได้แก่ การศึกษาถึงวิธีการที่ใช้สำหรับลดความผิดพลาดของข้อมูลที่ถูกส่งผ่านระบบสื่อสาร ซึ่งเกิดจากสาเหตุต่างๆ หลายประการด้วยกัน ได้แก่ การถูกรบกวนจากสัญญาณรบกวน และ การเกิดการแทรกสอดระหว่างสัญลักษณ์ (Intersymbol Interference, ISI) ที่มีผลทำให้สัญญาณข้อมูลที่รับได้ที่ปลายทางนั้นเกิดความผิดเพี้ยนไป ตัวอย่าง เช่น กรณีของระบบสื่อสารไร้สาย ดังนั้น เพื่อที่จะลดอัตราการเกิดความผิดพลาดของข้อมูลที่ถูกส่งผ่านช่องสัญญาณ จึงมีความจำเป็นที่จะต้องใช้วิธีการสำหรับลดความผิดเพี้ยนของสัญญาณหรือข้อมูลที่ถูกส่งผ่านระบบสื่อสารให้ลดลง

ในส่วนของการป้องกันความผิดพลาดของข้อมูลที่ถูกส่งผ่านระบบสื่อสาร เนื่องมาจากการถูกรบกวนจากสัญญาณรบกวนต่างๆ ในระบบสื่อสารนั้น วิธีการหนึ่งที่มีการนำมาใช้งาน ได้แก่ วิธีการเข้ารหัสข้อมูล (Encoding) โดยสำหรับการทำงานนั้น จะเป็นการนำข้อมูลดิจิตอลที่จะทำการส่งผ่านช่องสัญญาณ มาผ่านกระบวนการเข้ารหัส (Encoding) เพื่อเปลี่ยนแปลงข้อมูลที่จะทำการส่งผ่านระบบสื่อสาร ให้อยู่ในอีกลักษณะหนึ่ง ที่มีความสามารถในการตรวจจับ และแก้ไขข้อมูลได้ เมื่อข้อมูลดังกล่าวถูกส่งมาถึงปลายทาง จะมีการนำข้อมูลที่รับได้มาผ่านกระบวนการถอดรหัส (Decoding) เพื่อทำการตรวจจับหรือทำการแก้ไขข้อมูลที่เกิดความผิดพลาดให้กลับมาถูกต้อง ดังนั้น ผลลัพธ์ที่ได้จากการใช้วิธีการนี้ นั้น จะส่งผลทำให้ค่าอัตราการเกิดความผิดพลาดของข้อมูล (Bit Error Rate, BER) นั้นมีค่าลดลง แต่เนื่องจากข้อมูลที่ได้จากการเข้ารหัสนั้น จะมีขนาดที่ใหญ่ขึ้น จึงทำให้ต้องมีการส่งข้อมูลด้วยปริมาณที่สูงขึ้นตามไปด้วย

สัญญาณรบกวน (Noise) เป็นอีกตัวแปรหนึ่งที่ส่งผลอย่างยิ่งที่อาจทำให้ข้อมูลเกิดความผิดพลาดขึ้นได้ และเพื่อป้องกันหรือให้ผลกระทบดังกล่าวนี้ลดลง จึงได้มีการพัฒนารูปแบบการป้องกันและแก้ไขแบบต่างๆ ขึ้นมาใช้งาน สำหรับวิธีการเข้ารหัส (Error Correcting Codes) เป็นแนวทางหนึ่งที่ถูกนำมาประยุกต์สำหรับการสื่อสารดิจิตอลทั่วไป ซึ่งสามารถทำให้อัตราการเกิดความผิดพลาดในการส่งข้อมูล อันเนื่องจากผลกระทบของสัญญาณรบกวนมีค่าลดลงได้ โดยมิต้องมีการเพิ่มกำลังของเครื่องส่งต้นทาง การเข้ารหัสเพื่อวัตถุประสงค์ดังกล่าวนั้นได้มีการพัฒนารูปแบบการทำงานมาอย่างต่อเนื่อง สำหรับวิธีการเข้ารหัสแบบบล็อก (Block Code) เป็นรูปแบบใน

ยุคแรกๆ รวมไปถึงรหัสแบบ Cyclic Code และรหัสแบบ Reed Solomon Code (RS) เป็นต้น ต่อมาได้มีการพัฒนารูปแบบของวิธีการเข้ารหัสแบบ Convolution Code ขึ้นมาเพื่อปรับปรุงประสิทธิภาพการแก้ไขความผิดพลาดของข้อมูลให้ดีขึ้นกว่าในอดีต รวมทั้งปรับปรุงวิธีการสร้างให้สามารถรองรับการส่งข้อมูลด้วยอัตราเร็วที่สูงขึ้นและหลากหลายเงื่อนไขมากขึ้น เช่นสำหรับระบบโทรศัพท์เคลื่อนที่ และระบบสื่อสารผ่านดาวเทียม เป็นต้น รวมถึงเพื่อรองรับกับการส่งข้อมูลในปริมาณที่สูงขึ้น (Throughput) ต่อมาจึงได้มีการพัฒนารูปแบบการเข้ารหัสให้สามารถใช้ในระบบสื่อสารที่มีการมอดูเลตแบบหลายระดับ (Multilevel Modulation) เพื่อการใช้งานในหลากหลายสภาวะได้ เช่น วิธีการเข้ารหัสแบบ Trellis-Coded Modulation (TCM) ซึ่งเป็นหนึ่งในรหัสข้อมูลที่สำคัญ และถูกนำมาใช้ในระบบสื่อสารผ่านดาวเทียม และอุปกรณ์ประเภท MODEM (Modulation Demodulation) เป็นต้น มีรูปแบบผสมของวิธีการเข้ารหัสหลายรูปแบบเข้าด้วยกันที่ถูกพัฒนาขึ้นมา เพื่อให้มีความเหมาะสมกับการประยุกต์เฉพาะกิจต่างๆ เช่น การนำรหัสแบบ Reed Solomon Codes และแบบคอนโวลูชันมาใช้งานร่วมกัน เป็นต้น จนกระทั่งมาถึงวิธีการเข้ารหัสแบบเทอร์โบ (Turbo Codes) เป็นอีกรูปแบบหนึ่งของวิธีการเข้ารหัสที่มีความสำคัญอย่างยิ่ง ในยุคของข่าวสารไร้พรมแดน

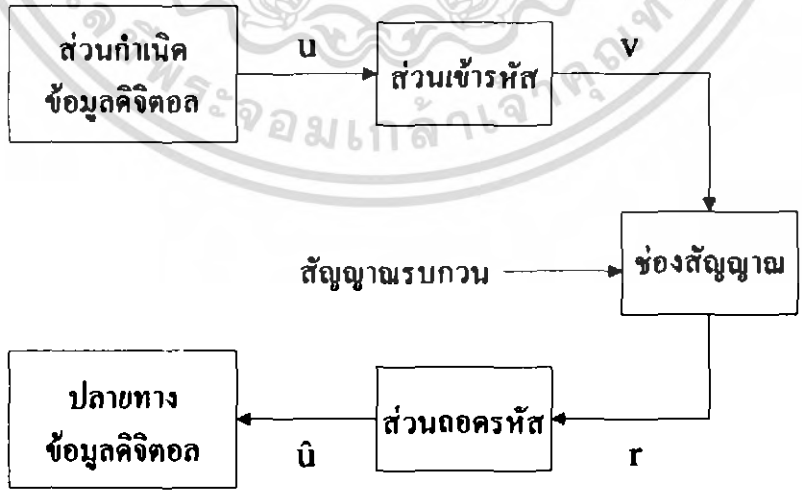
โครงสร้างของระบบการสื่อสารดิจิตอลพื้นฐาน



รูปที่ 2.1 บล็อคไดอะแกรมของการส่ง-รับข้อมูลหรือการจัดเก็บข้อมูลโดยทั่วไป

ในระบบการสื่อสารโดยทั่วไปสามารถแสดงได้ดังรูปที่ 2.1 โดยแหล่งกำเนิดข้อมูล (Information Source) จะให้กำเนิดข้อมูลข่าวสาร เช่น เสียงพูดของคน ซึ่งจะถูกทำการเข้ารหัสข้อมูล (Source Encoding) ทำให้ได้ลำดับสัญญาณดิสครีต U ซึ่งเรียกว่า ลำดับข้อมูล (Information Sequence) โดยลำดับดังกล่าวนี้จะถูกส่งเข้าไปยังตัวเข้ารหัสช่องสื่อสาร (Channel Encoder) เพื่อเพิ่มประสิทธิภาพให้แก่ระบบซึ่งจะได้ลำดับสัญญาณ v หรือเรียกว่า คำรหัส (Codeword) ออกมา แต่เนื่องจากสัญญาณดีสครีตที่ได้นี้ไม่เหมาะสมต่อการส่งผ่านช่องสัญญาณสื่อสารจึงจำเป็นต้องทำการมอดูเลตเพื่อแปลงลำดับสัญญาณดีสครีตให้อยู่ในรูปแบบคลื่น (Waveform) ที่เหมาะสมต่อการส่งผ่านช่องสัญญาณ (Channel) ไปยังด้านรับ โดยระหว่างทางของช่องสัญญาณนั้นจะประกอบด้วยสัญญาณรบกวนต่างๆที่เข้ามามีอิทธิพลต่อสัญญาณที่เราทำการส่ง โดยทางด้านรับนั้นก็จะประกอบด้วยส่วนต่างๆ ซึ่งทำหน้าที่ย้อนกลับจากที่กล่าวไว้ข้างต้น โดยจากรูปที่ 2.1 นั้นลำดับสัญญาณ r คือค่าของสัญญาณที่ได้จากการตัดสินทางด้านเครื่องรับ โดยตัวดีมอดูเลเตอร์ (Demodulator) ส่วนลำดับสัญญาณ $\hat{U}$ คือค่าของสัญญาณที่ได้หลังจากผ่านการถอดรหัสช่องสัญญาณแล้ว ซึ่งในทางอุดมคตินั้นค่าของลำดับสัญญาณ $\hat{U}$ และ U ควรจะมีค่าเหมือนกัน กล่าวคือ ไม่เกิดความผิดพลาดในการรับส่งข้อมูลเลย แต่เนื่องจากอิทธิพลของสัญญาณรบกวนจึงอาจมีผลทำให้เกิดความผิดพลาดขึ้น เพราะฉะนั้นการเข้ารหัสควบคุมความผิดพลาดจึงเป็นการลดค่าความน่าจะเป็นของความผิดพลาด (Probability of Error) ให้น้อยที่สุด

ในกรณีที่แหล่งกำเนิดข้อมูลของเรานั้นอยู่ในรูปของสัญญาณดิจิตอลอยู่แล้ว เราสามารถดัดแปลงรูปแบบโครงสร้างของระบบได้ดังรูปที่ 2.2

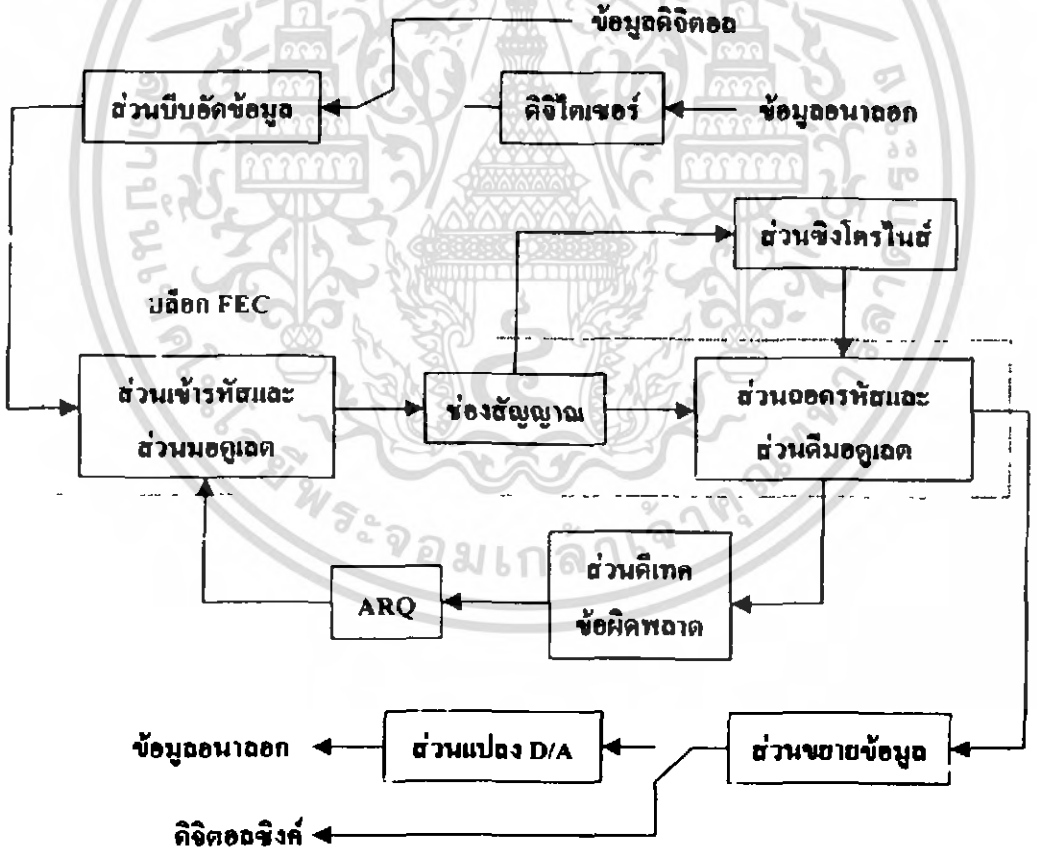


รูปที่ 2.2 แบบจำลองของระบบการเข้ารหัส

2.2 รูปแบบของการควบคุมความผิดพลาด

สำหรับระบบที่แสดงในรูปที่ 2.1 และ 2.2 นั่นเป็นระบบทิศทางเดียว (One Way System) ซึ่งการส่งสัญญาณนั้นถูกจำกัดเพียงแค่ทิศทางเดียวจากเครื่องส่งไปยังเครื่องรับ เช่น ในกรณีที่ระยะห่างของการติดต่อในระยะทางไกล เช่น ในการสื่อสารในอวกาศซึ่งระบบการควบคุมความผิดพลาดแบบนี้ เรียกว่า "Forward Error Correcting" (FEC) โดยที่รหัสที่ทำการใส่เพิ่มเข้าไปนั้นมีความสามารถในการแก้ไขความผิดพลาดที่เกิดขึ้นที่ด้านรับได้ด้วยตัวเอง

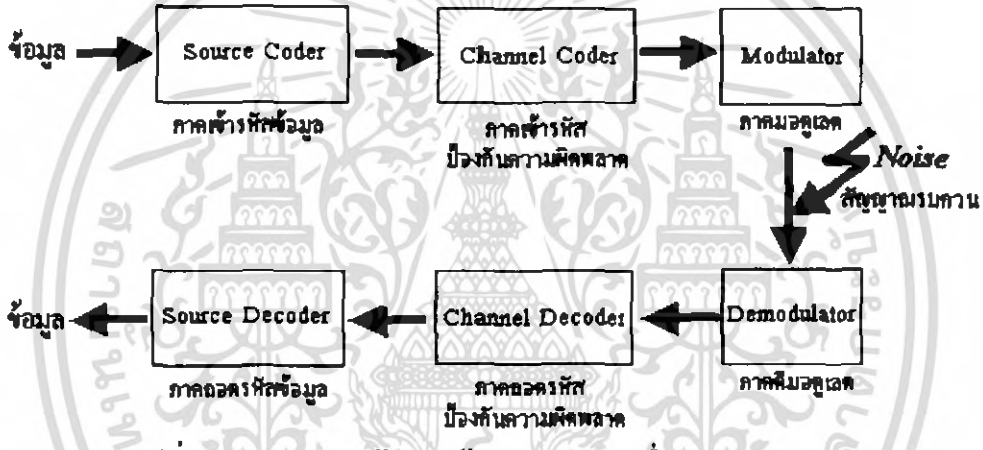
สำหรับการสื่อสารข้อมูล 2 ทิศทางนั้น ในการควบคุมความผิดพลาดยังสามารถใช้เทคนิคที่เรียกว่า "Automatic Repeat Request" (ARQ) ซึ่งหากทางด้านรับตรวจจับความผิดพลาดที่เกิดขึ้นได้ในการรับข้อมูล เครื่องรับจะส่งสัญญาณร้องขอไปยังด้านส่งเพื่อให้ส่งข้อมูลนั้นซ้ำอีกครั้ง โดยเทคนิคแบบ ARQ นั้นมีความซับซ้อนของอุปกรณ์น้อยกว่าแบบ FEC โดยโครงสร้างระบบแสดงดังรูป 2.3



รูปที่ 2.3 บล็อคของ FEC (Forward Error Control)

2.3 การเข้ารหัสข้อมูลในระบบสื่อสาร

การเข้ารหัสข้อมูลในระบบสื่อสาร จะเป็นวิธีการที่ใช้ในการลดความผิดพลาดในการส่งข้อมูลผ่านระบบสื่อสารซึ่งทำให้มีการเกิดความผิดพลาดลดลง เมื่อเปรียบเทียบกับกรณีของระบบสื่อสารที่ไม่มีการเข้ารหัสข้อมูล โดยสำหรับวิธีการที่ใช้ในการทำงานนั้น จะเป็นการนำข้อมูลที่เป็นข้อมูลแบบดิจิตอลที่จะทำการส่งผ่านระบบสื่อสารมาทำการเปลี่ยนแปลงรูปแบบของข้อมูลให้อยู่ในอีกลักษณะหนึ่งซึ่งถูกเรียกว่าเป็น Codeword ซึ่งจะมีคุณสมบัติพิเศษที่จะสามารถทำการแก้ไขหรือตรวจจับข้อมูลที่เกิดความผิดพลาดที่เกิดขึ้น ระหว่างการส่งข้อมูลผ่านระบบสื่อสารให้กลับมาเป็นข้อมูลที่ถูกต้องได้ เมื่อข้อมูลนั้นถูกส่งมาถึงปลายทาง ซึ่งรูปที่ 2.4 จะแสดงการทำงานของการเข้ารหัสข้อมูลในระบบสื่อสาร



รูปที่ 2.4 แผนผังการเข้ารหัสข้อมูลของระบบสื่อสารโดยสังเขป

โดยกระบวนการที่สำคัญสองกระบวนการที่ใช้ในการเข้ารหัสข้อมูลในระบบสื่อสาร ได้แก่ กระบวนการเข้ารหัส (Channel Encoder) ที่ใช้ในการเปลี่ยนแปลงข้อมูลที่จะทำการส่งให้อยู่ในรูปของ Codeword และกระบวนการถอดรหัส (Channel Decoder) ที่ใช้สำหรับการนำข้อมูลที่รับได้ที่ปลายทางนั้นมาทำการประมวลผลเพื่อหาข้อมูลที่คาดว่าต้นทางจะส่งมา ซึ่งลักษณะของข้อมูลที่ได้หลังจากกระบวนการเข้ารหัสข้อมูลที่จะถูกส่งผ่านระบบสื่อสารนั้น จะมีขนาดของข้อมูลที่มากกว่าข้อมูลที่ไม่มีการเข้ารหัสข้อมูล

2.4 รูปแบบของการเข้ารหัสข้อมูล

ในการเข้ารหัสข้อมูลในระบบสื่อสารนั้น จะมีรูปแบบของการเข้ารหัสอยู่หลายรูปแบบด้วยกันซึ่งจะสามารถแบ่งออกได้เป็น 2 รูปแบบหลักๆด้วยกัน ได้แก่ Block Codes และ Convolution Codes ซึ่งจะมีรายละเอียดดังนี้

2.4.1 การเข้ารหัสแบบ Block Codes

สำหรับการเข้ารหัสแบบ Block Codes นั้น จะเป็นการนำข้อมูลดิจิตอลที่จะทำการเข้ารหัสมาทำการแบ่งออกเป็นชุดหรือ Block ย่อยๆ ซึ่งแต่ละ Block ที่มีขนาดเท่ากับ k บิตนั้น จะถูกนำมาทำการประมวลผลเพื่อหาค่าของ Codeword ที่จะใช้แทนข้อมูลใน Block นั้นๆ ซึ่งจะมีการทำงานเช่นนี้ตั้งแต่ข้อมูล Block แรกจนถึง Block สุดท้ายจึงจะสิ้นสุดการเข้ารหัส ตัวอย่างของ Block Codes ได้แก่ Parity-Check Bit, Cyclic Codes และ Reed Solomon Codes เป็นต้น

2.4.2 การเข้ารหัสแบบ Convolutional Codes

สำหรับการเข้ารหัสแบบ Convolutional Codes นั้น จะเป็นรูปแบบของการเข้ารหัสข้อมูลที่มีความแตกต่างจาก Block Codes โดยจะเป็นการเข้ารหัสที่มีการนำข้อมูลในอดีตจำนวนหนึ่งที่ถูกป้อนเข้ามาภายในวงจรเข้ารหัสมาทำการประมวลผลร่วมกับข้อมูลที่ถูกป้อนเข้ามา ณ เวลานั้นๆ ในการคำนวณหาค่า Codeword ของข้อมูลชุดนั้น

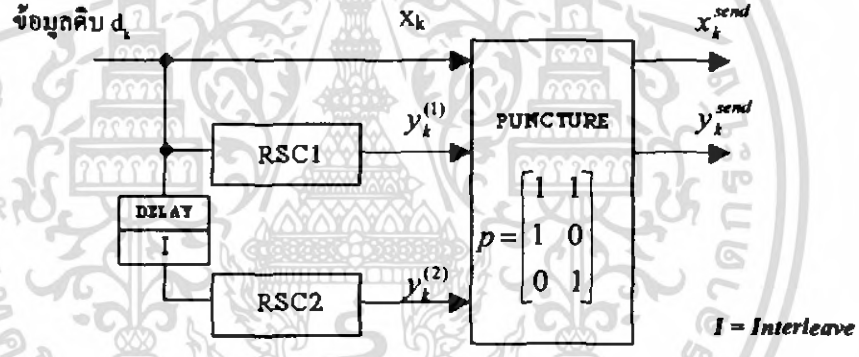
โดยสำหรับวิธีการเข้ารหัสข้อมูลทั้งสองรูปแบบนั้น จะมีลักษณะการทำงานที่เหมือนกันคือจะเป็นการนำข้อมูลที่จะทำการเข้ารหัสมาทำการแบ่งข้อมูลออกเป็นส่วนๆ หรือ Block ที่มีขนาดที่เท่าๆ กันแล้วจะมีการแทนขนาดของข้อมูลในแต่ละ Block ด้วยตัวแปร k ซึ่งในการทำงานของภาคเข้ารหัสนั้น จะเป็นการดึงข้อมูลที่จะทำการเข้ารหัสมาครั้งละ k บิต เพื่อนำมาทำการประมวลผลเพื่อหาค่าของ Codeword ที่มีขนาดเท่ากับ n บิต ซึ่งจะถูกส่งในระบบสื่อสารเพื่อแสดงถึงข้อมูลของ Block นั้น โดยที่จะมีการนิยามตัวแปรที่มีชื่อว่า Redundant Bit (r) ซึ่งจะเป็นตัวแปรที่ใช้แสดงถึงจำนวนของข้อมูลที่เพิ่มขึ้นในการเข้ารหัสข้อมูลในแต่ละครั้ง ซึ่งจะสามารถคำนวณได้จาก

$$r = n - k \tag{2.1}$$

2.5 การเข้ารหัสข้อมูลแบบ Turbo Codes

สำหรับรูปแบบในการเข้ารหัสข้อมูลแบบ Turbo Codes นั้น จะมีรูปแบบในการทำงานพื้นฐานดังวงจรในรูปที่ 2.5 โดยจะเป็นการนำวงจรเข้ารหัสข้อมูลอย่างน้อย 2 ชุดมาทำการใช้งานร่วมกันในรูปแบบของการเข้ารหัสแบบ Parallel Concatenate ซึ่งเป็นวิธีที่ใช้สำหรับการนำวงจรเข้ารหัสตั้งแต่ 2 วงจรขึ้นไปมาทำงานร่วมกันเพื่อทำให้ได้รูปแบบในการเข้ารหัสข้อมูลที่มีประสิทธิภาพในการทำงานที่สูงขึ้น โดยที่ไม่มีการใช้รูปแบบของวงจรที่มีความซับซ้อนมากขึ้น โดยสำหรับกรณีของวงจรเข้ารหัสแบบ Turbo Codes นั้น จะมีการนำการเข้ารหัสแบบ Recursive Systematic Convolution Codes (RSC) ซึ่งเป็นรูปแบบในการเข้ารหัสข้อมูลที่มีการพัฒนามาจากการเข้ารหัสแบบ Convolution Codes มาใช้ในการทำงาน โดยในการทำงานของภาคเข้ารหัสข้อมูล

ในรูปที่ 2.5 นั้น จะเป็นการนำข้อมูลดิบที่จะทำการเข้ารหัส $d_k = \{d_0, d_1, d_2, ...\}$ มาทำการเข้ารหัสข้อมูล โดยการป้อนข้อมูลดิบเข้ามาภายในวงจรเข้ารหัส RSC1 และ RSC2 ครั้งละ 1 บิต เพื่อทำการคำนวณหาค่าของข้อมูลที่ผ่านการเข้ารหัส (Codeword) ซึ่งผลลัพธ์ที่ได้นั้นจะมีลักษณะข้อมูลเป็นแบบ Systematic กล่าวคือ ข้อมูลที่ได้จากวงจรเข้ารหัสในแต่ละครั้งที่มีการทำงานนั้น จะประกอบไปด้วยข้อมูล 2 ส่วนด้วยกันได้แก่ข้อมูลในส่วนที่มีค่าเหมือนกับข้อมูลดิบ (x_k) และข้อมูลในส่วนที่ได้จากภาคเข้ารหัส RSC1และ RSC2 ($y_k^{(1)}, y_k^{(2)}$) ดังนั้นข้อมูลที่ได้จากการเข้ารหัสนั้นจะมีค่าเท่ากับ $\{x_0, y_0^{(1)}, y_0^{(2)}, x_1, y_1^{(1)}, y_1^{(2)}, x_2, y_2^{(1)}, y_2^{(2)}, ...\}$ โดยที่ข้อมูลดิบที่จะมีการนำมาใช้ในการเข้ารหัสข้อมูลสำหรับวงจร RSC2 นั้น จะมีการนำข้อมูลดิบมาผ่านกระบวนการ Interleave ก่อนที่จะมีการส่งมายังภาคเข้ารหัสเพื่อที่จะทำให้ข้อมูลที่ได้จากวงจรเข้ารหัสในแต่ละวงจรนั้นไม่มีความสัมพันธ์ (Correlate) ซึ่งกันและกัน

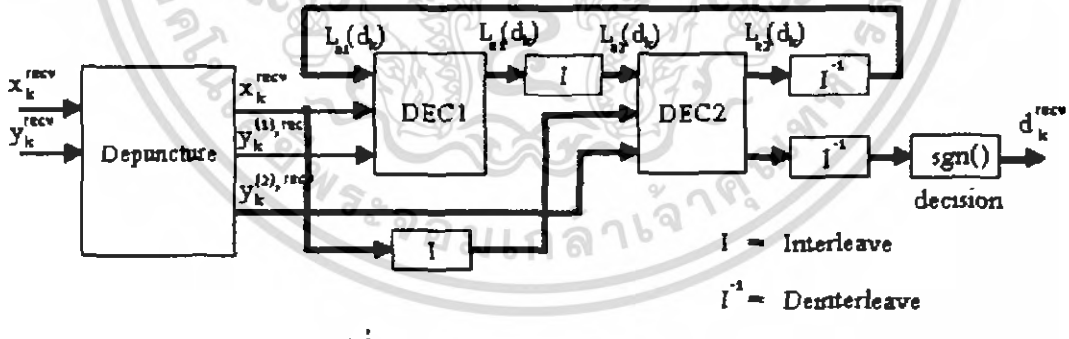


รูปที่ 2.5 วงจรเข้ารหัสแบบ Turbo Codes

และข้อมูลที่ได้จากการทำงานของภาคเข้ารหัสข้อมูล RSC1 และ RSC2 นั้นจะถูกส่งต่อมายังภาค Puncture ซึ่งจะมีหน้าที่ในการลดจำนวนของข้อมูลที่จะถูกส่งผ่านระบบสื่อสารให้มีจำนวนลดลงตามรูปแบบที่มีการกำหนดไว้ ซึ่งจะมีรูปแบบในการทำงานอยู่หลายรูปแบบด้วยกัน โดยในกรณีของวงจรเข้ารหัสแบบ Turbo Codes ในรูปที่ 2.5 นั้น จะเป็นการลดจำนวนข้อมูลที่จะส่งผ่านระบบสื่อสารโดยการส่งสัญญาณข้อมูล $y_k^{(1)}$ และ $y_k^{(2)}$ สลับกันส่งร่วมกับสัญญาณข้อมูล x_k ดังนั้นข้อมูลที่จะได้หลังจากการทำงานของภาคPunctureนั้นจะมีค่าเท่ากับ $\{x_0, y_0^{(1)}, x_1, y_1^{(2)}, x_2, y_2^{(1)}, ...\}$

2.6 ภาคถอดรหัสข้อมูลแบบ Turbo Codes

สำหรับภาคถอดรหัสข้อมูลแบบ Turbo Codes ที่ใช้สำหรับนำข้อมูลที่รับได้ที่ภาครับซึ่งเป็นข้อมูลที่ถูกเข้ารหัสแบบ Turbo Codes มาทำการเปลี่ยนรูปแบบของข้อมูลที่รับได้ให้กลับมาอยู่ในรูปของข้อมูลดิบอีกครั้ง โดยใช้ข้อมูลต่างๆที่ถูกส่งมานั้นในการตัดสินใจเลือกค่าของข้อมูลดิบที่มีความเป็นไปได้มากที่สุด เพื่อส่งออกไปเป็นผลลัพธ์ที่ได้จากการทำงานของวงจรถอดรหัส โดยสำหรับกรณีของการถอดรหัสสำหรับ Turbo Codes นั้นจะมีรูปแบบในการถอดรหัสข้อมูลเป็นแบบ Iterative Decoding ที่มีรูปแบบในการถอดรหัสเป็นแบบป้อนกลับ (Feed Back) โดยในการทำงานในแต่ละรอบการทำงานนั้น จะเป็นการนำข้อมูลที่รับได้ซึ่งเป็นข้อมูลที่ผ่านการเข้ารหัสมาทำการประมวลผลเพื่อคำนวณหาค่าความน่าจะเป็นของข้อมูลดิบที่คาดว่าจะถูกส่งมา ณ เวลาต่างๆ โดยค่าความน่าจะเป็นดังกล่าวนั้น จะถูกส่งออกมาจากวงจรถอดรหัสในรูปแบบของค่าที่เรียกว่า Extrinsic Information โดยจะเป็นข้อมูลที่จะถูกนำมาใช้ในการคำนวณในรอบต่อไปเพื่อเป็นการปรับปรุงการคำนวณต่างๆ ให้มีความแม่นยำมากยิ่งขึ้นในการประมวลผลในรอบต่อไป ดังนั้น ทุกๆครั้งที่มีการทำงาน จะทำให้ข้อมูลต่างๆที่ได้จากการคำนวณนั้นมีค่าใกล้เคียงค่าที่ถูกต้องมากยิ่งขึ้น และเมื่อมีการวนรอบในการทำงานครบตามที่กำหนดแล้วจะมีการนำข้อมูลที่เรียกว่า Log A Posteriori Probability (LAPP) ที่ได้จากการทำงานมาใช้ในการตัดสินใจเพื่อเลือกค่าของข้อมูลดิบที่จะถูกส่งออกไปเป็นผลลัพธ์ของภาคถอดรหัส โดยจะมีรูปแบบในการถอดรหัสข้อมูลดังรูปที่ 2.6



รูปที่ 2.6 วงจรถอดรหัส Turbo Codes

ซึ่งจะมีรูปแบบในการทำงานที่มีการประยุกต์มาจากวิธีการถอดรหัสแบบ Symbol-by-Symbol Maximum a Posteriori (MAP) หรือ BCJR Algorithm [4] ซึ่งถูกค้นพบโดย Bahl ในปี พ.ศ. 2517 (ค.ศ.1974) โดยสำหรับรูปแบบในการถอดรหัสข้อมูลแบบ Turbo Codes นั้น จะมีลักษณะการทำงานดังรูปที่ 2 6

ในส่วนของวิธีการถอดรหัสสำหรับ Turbo Codes ดังที่แสดงในรูปที่ 2.6 นั้น จะมีลักษณะการทำงานในรูปแบบของการถอดรหัสแบบ Iterative Decoding โดยในการถอดรหัสข้อมูลเพื่อหาค่าของข้อมูลดิบที่ถูกส่งมาในแต่ละช่วงนั้น จะเป็นการนำข้อมูลที่รับได้ในแต่ละช่วง (x_t , y_t) มาทำการ Depuncture เพื่อจัดรูปแบบของข้อมูลเพื่อกระจายข้อมูลไปยังภาคถอดรหัส DEC1 และ DEC2 ซึ่งจะเป็นภาคถอดรหัสที่ใช้ในการถอดรหัสข้อมูลที่ได้จากวงจรเข้ารหัส RSC1 และ RSC2 ของภาคเข้ารหัสตามลำดับที่มีรูปแบบในการถอดรหัสที่มีการประยุกต์มาจากวิธีการถอดรหัสแบบ Symbol-by-Symbol Maximum a Posteriori (MAP) หรือ BCJR Algorithm [4] ซึ่งถูกค้นพบโดย Bahl ในปี พ.ศ. 2517 (ค.ศ.1974) มาใช้ในการทำงาน โดยในการทำงานของภาคถอดรหัสในแต่ละวงจรนั้น จะเป็นการนำข้อมูลที่รับได้ ซึ่งจะประกอบไปด้วยข้อมูลในส่วนของข้อมูลที่รับได้ (x_t^{recv}, y_t^{recv}) และข้อมูลในส่วนที่เรียกว่า a Priori Probability (L_t) ที่ได้จากการประมวลผลของวงจรถอดรหัสวงจรก่อนมาใช้ในการทำงาน โดยจะนำค่าที่ได้รับนั้นมาใช้ในการคำนวณหาค่าที่เรียกว่า Extrinsic Information ซึ่งจะเป็นข้อมูลที่จะถูกนำไปใช้ในการปรับปรุงการทำงานของวงจรถอดรหัสชุดต่อไปให้มีการคำนวณที่มีความแม่นยำมากยิ่งขึ้น และเมื่อมีการทำงานครบตามที่กำหนดแล้ว จะมีการคำนวณหาค่าที่เรียกว่า Log A Posteriori Probability (LAPP) เพื่อนำมาใช้ในการตัดสินใจค่าของข้อมูลดิบ d_t ที่จะถูกส่งออกไปเป็นผลลัพธ์ของภาคถอดรหัส

เนื่องจากข้อมูลต่างๆที่ถูกใช้ในการทำงานของภาคเข้ารหัส RSC2 นั้น จะอยู่ในรูปของข้อมูลที่ผ่านกระบวนการ Interleave เพื่อทำให้ข้อมูลที่ได้จากวงจรเข้ารหัสทั้งสองนั้นไม่มีค่าสัมพันธ์กัน ดังนั้นเพื่อให้เกิดความสัมพันธ์กับภาคเข้ารหัส จึงจำเป็นที่จะต้องนำข้อมูลต่างๆที่จะถูกส่งมาจากวงจรถอดรหัส DEC1 มายังวงจร DEC2 มาผ่านกระบวนการ Interleave เพื่อเปลี่ยนแปลงรูปแบบของข้อมูลให้อยู่ในรูปแบบเดียวกันก่อนที่จะนำมาทำการใช้งาน และในทางกลับกัน ข้อมูลต่างๆที่จะถูกส่งออกไปจากวงจรถอดรหัส DEC2 มายังวงจร DEC1 นั้น จะต้องมีการนำข้อมูลมาผ่านกระบวนการ Deinterleave ก่อนที่จะมีการนำมาใช้งาน

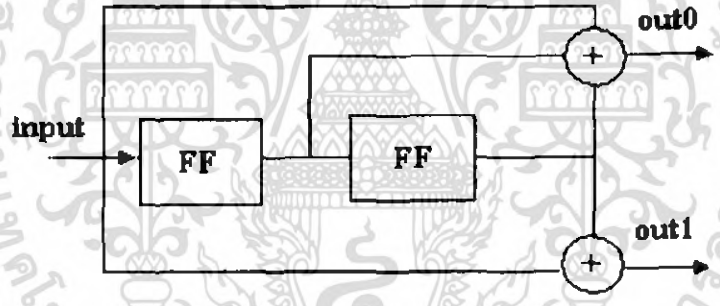
2.7 พื้นฐานในการเข้ารหัสข้อมูลสำหรับ Turbo Codes

สำหรับการเข้ารหัสข้อมูลแบบ Turbo Codes นั้น เป็นรูปแบบในการเข้ารหัสข้อมูลที่มีการนำวงจรเข้ารหัสแบบ Recursive Systematic Convolution Codes มาใช้ในการเข้ารหัสข้อมูลในรูปแบบของการทำงานแบบ Parallel Concatenate ดังนั้นในการพิจารณาถึงกระบวนการในการทำงานต่างๆของวงจรเข้ารหัสแบบ Turbo Codes นั้น จะต้องมีการพิจารณาถึงรูปแบบในการ

เข้ารหัสข้อมูลแบบ Convolution Codes และ Recursive Systematic Convolution Codes โดยจะมีรายละเอียดต่างๆดังต่อไปนี้

2.8 การเข้ารหัสแบบ Convolution

การเข้ารหัสข้อมูลแบบ Convolution เป็นความสัมพันธ์ระหว่างข้อมูลอินพุทเรียงลำดับอย่างต่อเนื่อง โดยข้อมูลเข้ามาผ่านตัว Shift-Register (Flip-Flop) และ Modulo-2 adder (Exclusive or) การหาเอาท์พุทของภาคเข้ารหัสจะทำโดยนำข้อมูลที่อยู่ใน Shift-Register บวกแบบ Modulo-2 adder ซึ่งจะมีลักษณะของวงจรแสดงดังรูปที่ 2.4 จากวงจรเข้ารหัสจะมีอัตราการเข้ารหัส (Rate) เท่ากับ ½ และค่า Constraint Length (K) เท่ากับ 3 โดยจะใช้ Generator Polynomial เพื่อแสดงตำแหน่งของใน Shift-Register ที่จะนำมาหาค่าเอาท์พุทโดยของการบวกแบบ Modulo-2 คือ g_0 เท่ากับ 7_8 และ g_1 เท่ากับ 5_8



รูปที่ 2.7 การเข้ารหัสแบบ Convolution Code ที่ K=3

จากรูปที่ 2.7 จะเป็นตัวอย่างของวงจรเข้ารหัสข้อมูลแบบ Convolutional Code โดยในการทำงานนั้นจะมีการดึงข้อมูลที่จะทำการเข้ารหัสมาครั้งละ 1 บิต (k =1) เข้ามาภายในวงจร ซึ่งจะทำให้ข้อมูลที่อยู่ในตำแหน่งต่างๆ ของ Shift-Register จากนั้นถูกเลื่อนไปอยู่ในตำแหน่งถัดไป ต่อจากนั้นจะมีการนำข้อมูลทั้งหมดที่เก็บไว้ใน Shift-Register มาทำการคำนวณเพื่อหาเอาท์พุทจำนวน 2 บิต (n = 2) ซึ่งอัตราการเข้ารหัสเท่ากับ ½ (Rate=k/n=1/2) ที่จะเป็นผลลัพธ์ส่งออกไปจากภาคเข้ารหัสข้อมูล ส่วนค่าจำนวนของ Shift-Register และอินพุทข้อมูลที่นำมาบวกแบบ Modulo-2 เรียกว่าค่า Constraint Length (K) โดยจะใช้ Generator Polynomial เพื่อแสดงตำแหน่งของใน Shift-Register ที่จะนำมาหาค่าเอาท์พุทโดยการบวกแบบ Modulo-2 ในการแสดงลักษณะของวงจรเข้ารหัส โดยสำหรับกรณีของวงจรในรูปที่ 2.4 นั้น จะมี Constraint Length = 3 และ Generator Polynomial $g_0=5_8$ และ $g_1=7_8$

2.9 การวิเคราะห์การทำงานของวงจรเข้ารหัส

จากวงจรเข้ารหัสแบบ Convolution ที่มีการรับข้อมูลที่จะทำการเข้ารหัสเข้ามาภายในวงจรครั้งละ 1 บิต จากนั้นจึงทำการคำนวณหาค่าของ Codeword ที่จะถูกส่งออกไปเป็นผลลัพธ์ของภาคเข้ารหัสจำนวน 2 บิต โดยที่ในการคำนวณหาค่า Codeword ในแต่ละครั้งนั้นจะมีการนำข้อมูลที่อยู่ใน Shift-Register จำนวน 3 บิต (K) มาใช้ในการคำนวณ ซึ่งในการวิเคราะห์การทำงานของวงจรเข้ารหัสนั้น จะมีการแทนการทำงานต่างๆ ของวงจรด้วยตัวแปรที่เรียกว่า Generator Polynomial ซึ่งจะเป็นตัวแปรที่ใช้ แสดงถึงลักษณะของการคำนวณหาผลลัพธ์ในการเข้ารหัสของ O/P แต่ละตัว โดยในกรณีของวงจรเข้ารหัสตัวอย่างในรูปที่ 2.4 นั้น จะสามารถแสดงการทำงานต่างๆของวงจรได้ด้วย Generator Polynomial ต่อไปนี้

$$g1 = [1\ 1\ 1]$$

$$g2 = [1\ 0\ 1]$$

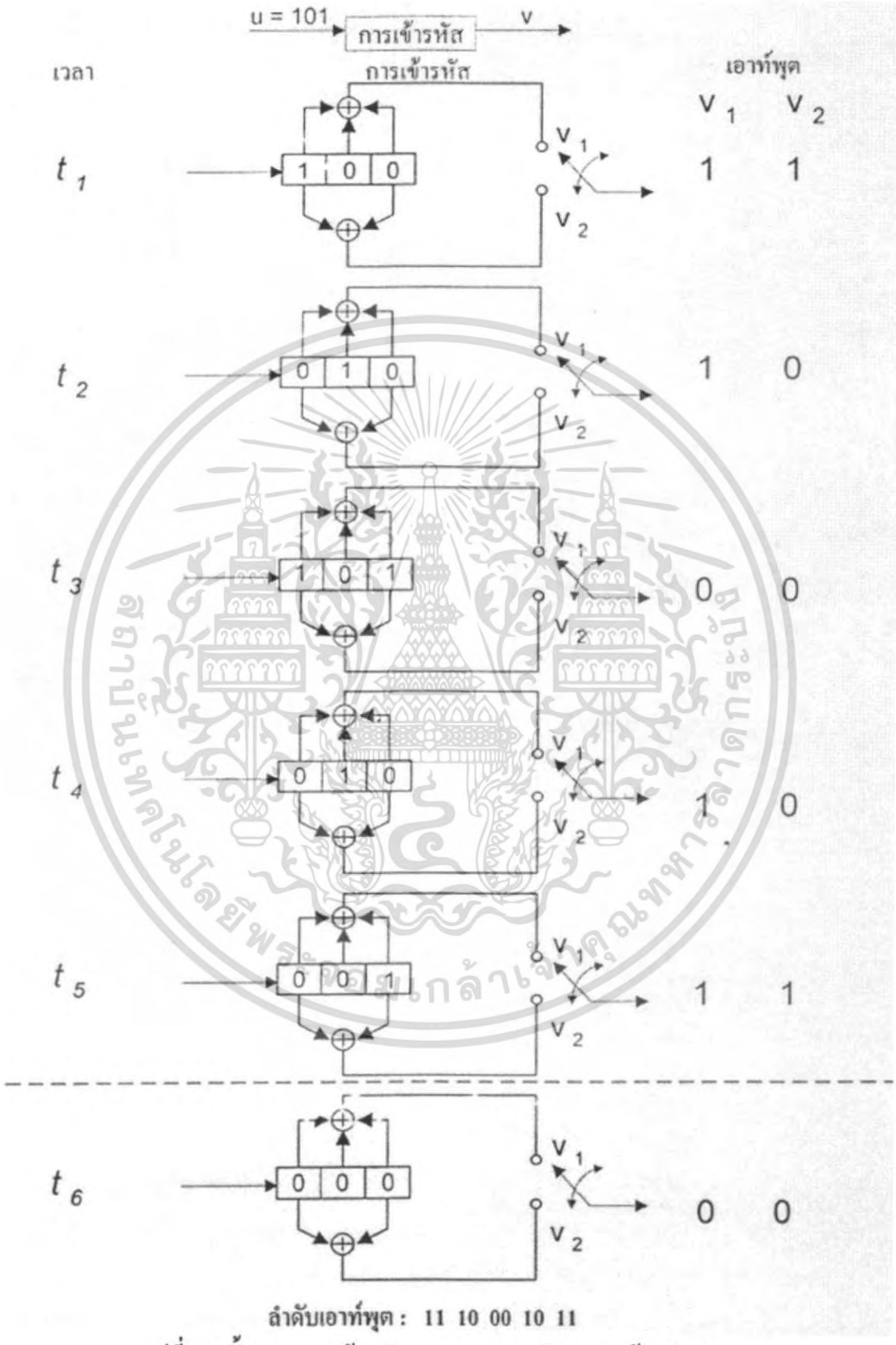
หากสมมติว่าข้อมูลอินพุตเป็น [1 0 1] ถูกทำการเข้ารหัสการประสาน โดยตัวเข้ารหัสแสดงได้ดังรูปที่ 2.7 โดยอินพุต 3 บิต ถูกชิพเข้าไปในเวลา t_1,t_2 และ t_3 ซึ่งผลที่ได้ดังรูป 2.8 ส่วนเวลา t_4 และ t_5 เป็นการชิพบิต 0 เข้าไปซึ่งเรียกว่า Zero-Padding เพื่อให้บิตข้อมูลสุดท้ายถูกชิพไปจนถึงรีจิสเตอร์ตัวสุดท้าย ส่วนเวลา t_6 เป็นการทำให้รีจิสเตอร์มีค่ากลับเป็น 0 ทั้งหมด โดยเอาท์พุตที่ได้จะมีค่าเท่ากับ 1 1 1 0 0 0 1 0 1 1 โดยสัญลักษณ์ที่อยู่ซ้ายมือสุดจะถูกส่งออกไปก่อน

ในการคำนวณหาค่าของข้อมูลที่ได้หลังจากการเข้ารหัสนั้น จะสามารถนำข้อมูลที่ได้จากค่าของ Generator Polynomial มาทำการคำนวณหาค่าของผลลัพธ์ที่ได้จากการเข้ารหัส โดยการนำข้อมูลที่จะทำการเข้ารหัสมาทำการ Convolution กับค่าของ Generator Polynomial ดังสมการ

$$O/P = [input]*gi$$

$$O/P1 = [1\ 0\ 1]*[1\ 1\ 1] = 1\ 1\ 0\ 1\ 1$$

$$O/P2 = [1\ 0\ 1]*[1\ 0\ 1] = 1\ 0\ 0\ 0\ 1$$



รูปที่ 2.8 ขั้นตอนการเข้ารหัส Convolution อัตราการเข้ารหัส ½

ดังนั้นค่าของ Codeword ที่ได้จากการเข้ารหัสข้อมูลนั้นจะเกิดจากการข้อมูลที่คำนวณได้จาก O/P1 และตามด้วย O/P2 ซึ่งจะมีค่าเท่ากับ 11 10 00 10 11 ซึ่งจากการที่การคำนวณต่างๆในการหาผลลัพธ์ของวงจรเข้ารหัสนั้น สามารถที่จะคำนวณได้จากการหาค่า Convolution ดังนั้นจึงมีการเรียกรูปแบบในการเข้ารหัสข้อมูลในลักษณะนี้ว่าเป็นการเข้ารหัสแบบ Convolution Codes และเมื่อนำมาเขียนรวมกันจะได้เป็น Generator Polynomial ของวงจรเข้ารหัสจะมีลักษณะดังนี้

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

โดยในบางกรณีนั้น จะมีการแทนค่าของ Generator Polynomial ด้วยตัวเลขฐาน 8 เพื่อแสดงถึงข้อมูลที่อยู่ใน Generator Polynomial โดยในกรณีของวงจรตัวอย่างนั้นจะมีการแทนค่าของ Generator Polynomial ด้วยตัวเลข 7_8 และ 5_8 และสำหรับข้อมูลในการเข้ารหัสข้อมูลนั้น จะมีการแบ่งข้อมูลที่จะทำการเข้ารหัสออกเป็น n ชุด โดยที่แต่ละชุดนั้นมีขนาดเท่ากับ k_0 บิต ซึ่งในการทำงานนั้นจะมีการดึงข้อมูลเข้ามาภายในวงจรในแต่ละครั้งนั้น จะมีการนำข้อมูลจำนวน k_0 บิตมาทำการประมวลผลเพื่อหาค่า Codeword จำนวน n_0 บิตเพื่อส่งออกไปจากวงจร โดยที่จะมีการทำงานในลักษณะนี้เรื่อยๆ จนกระทั่งข้อมูลทุกบิตถูกนำมาเข้ารหัสทั้งหมดและผลลัพธ์ที่ออกจากวงจรเข้ารหัสจะกลับเข้าสู่สภาวะที่มีข้อมูลทั้งหมดเป็น 0 จึงจะถือว่าเป็นการสิ้นสุดการเข้ารหัสข้อมูล ดังนั้นสำหรับการเข้ารหัสแบบ Convolution Codes นั้นจะมีค่าอัตราการเข้ารหัส (Code Rate) เท่ากับ

$$R = \frac{nk_0}{(n+L-1)n_0} \tag{2.2}$$

และในกรณีที่ค่า n มีค่ามากกว่าค่า L มากจะได้ว่าค่าอัตราการเข้ารหัสนั้นจะมีค่าเท่ากับ

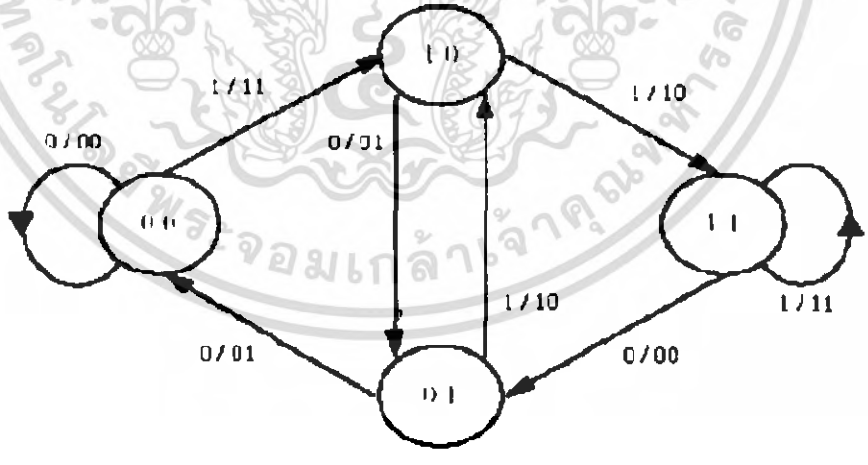
$$R = \frac{k_0}{n_0} \tag{2.3}$$

ในการวิเคราะห์การทำงานต่างๆของวงจรเข้ารหัสแบบ Convolution นั้น นอกจากจะมีการใช้สมการของ Generator Polynomial ในการแสดงลักษณะของวงจรเข้ารหัสแล้ว จะมีวิธีการแสดงการทำงานต่างๆ ของวงจรเข้ารหัส โดยการใช้รูปภาพแสดงความสัมพันธ์ระหว่างข้อมูลที่ถูกเก็บไว้

ในวงจรเข้ารหัสข้อมูลที่ถูกป้อนเข้ามา ณ เวลานั้น และ ค่าของ Codeword ที่จะถูกส่งออกไป เมื่อมีข้อมูลในกรณีต่างๆป้อนเข้ามา ซึ่งรูปแบบของภาพที่ใช้แสดงการทำงานของวงจรเข้ารหัสนั้น จะมีอยู่ด้วยกัน 3 รูปแบบด้วยกัน ได้แก่ State Diagram, Tree Diagram และ Trellis Diagram ซึ่งจะมีรายละเอียดดังต่อไปนี้

2.10 State Diagram

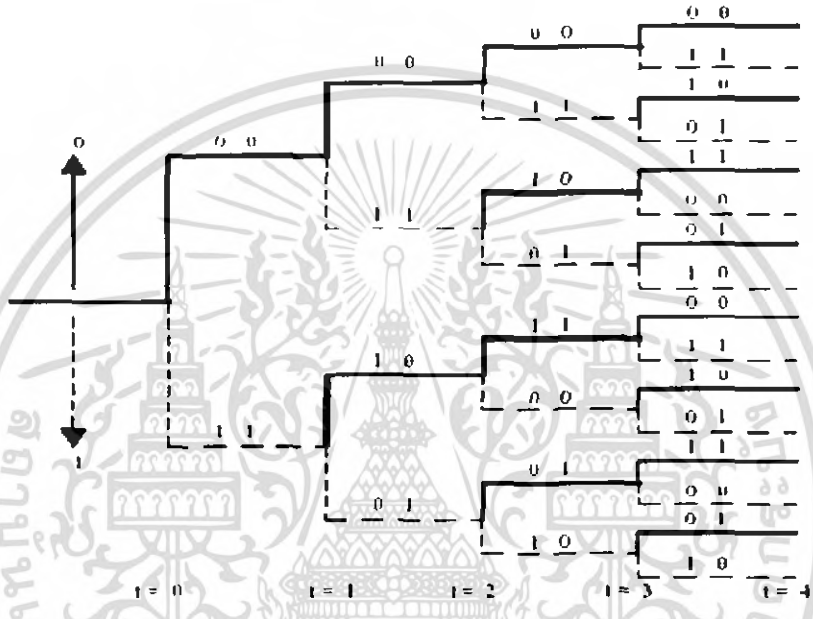
State Diagram จะแสดงค่าของข้อมูลใน Shift-Register และเอาท์พุทของตัวเข้ารหัส ซึ่งแสดงดังรูปที่ 2.9 ตัวเลขที่อยู่ในวงกลมแต่ละวงนั้น จะหมายถึงสภาวะต่างๆ ของข้อมูลที่ถูกเก็บไว้ใน Shift-Register ซึ่งในกรณีของวงจรเข้ารหัสที่ใช้เป็นตัวอย่างนั้น จำนวนสถานะ (State) ทั้งหมดเท่ากับ 4 สถานะ และ สำหรับลูกศรที่ถูกแสดงไว้ในรูปนั้นจะแสดงถึงลักษณะของการเปลี่ยนแปลงการทำงานจากสถานะหนึ่งไปเป็นอีกสถานะหนึ่ง ซึ่งจะขึ้นอยู่กับข้อมูลที่ป้อนเข้ามา ณ เวลานั้นๆ เช่นกรณีที่ข้อมูลที่เก็บไว้มีค่าเป็น 00 เมื่อมีข้อมูล 1 ป้อนเข้ามาจะมีผลทำให้ข้อมูลที่เก็บไว้ถูกเปลี่ยนไปเป็น 10 และจะมีการส่งค่า 11 ออกไปจากวงจร (แทนด้วยสัญลักษณ์ 1 / 11 ดังรูป) แต่ถ้าหากว่ามีการป้อนข้อมูล 0 เข้ามา วงจรก็จะยังคงมีสภาวะเป็น 00 เหมือนเดิมและจะมีการส่งค่า 00 ออกไปจากวงจร (แทนด้วยสัญลักษณ์ 0 / 00 ดังรูป)



รูปที่ 2.9 State Diagram

2.11 Tree Diagram

สำหรับ Tree Diagram นั้น จะเป็นการพิจารณาถึงลักษณะของการทำงานของวงจรเข้ารหัสข้อมูลโดยที่จะมีการพิจารณาถึงค่าของผลลัพธ์ที่ได้หลังจากการป้อนข้อมูลต่างๆเข้าไปในวงจรเข้ารหัสเป็นหลัก ซึ่งจะมีลักษณะของ Tree Diagram ดังรูปที่ 2.10

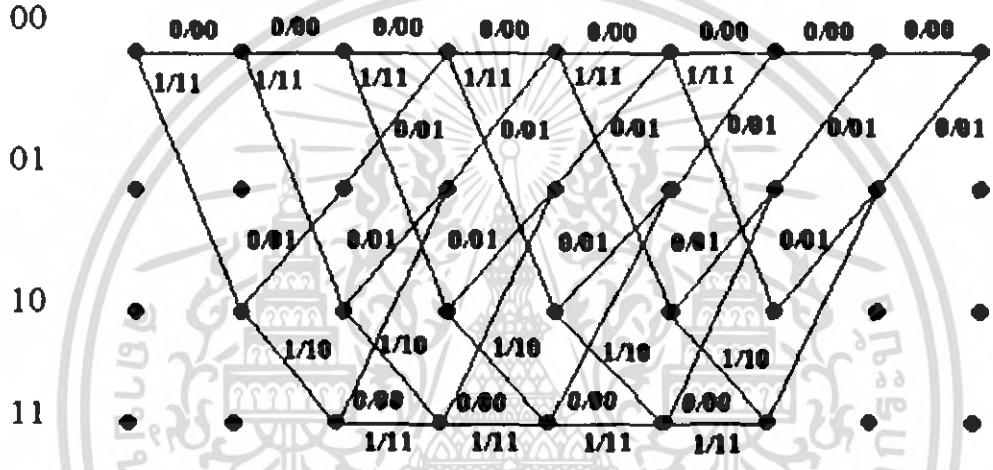


รูปที่ 2.10 Tree Diagram

สำหรับการแสดงการทำงานของวงจรเข้ารหัสแบบ *Convolution* Codes โดยใช้ Tree Diagram นั้น จะเป็นการพิจารณาการทำงานโดยการคำนึงถึงข้อมูลที่ป้อนเข้ามาและที่จะถูกส่งออกไปจากภาคเข้ารหัสเป็นหลัก ซึ่งในการพิจารณานั้น จะเริ่มต้น ณ ตำแหน่งรากของ Tree Diagram ซึ่งในกรณีของรูปที่ 2 10 นั้น จะอยู่ในตำแหน่งซ้ายมือสุดของรูป ซึ่งจะมีการนำข้อมูลที่ถูกป้อนเข้ามาภายในวงจรเข้ารหัสเป็นตัวกำหนดทิศทางการเดินทางของข้อมูลใน Tree Diagram โดยในกรณีของ Tree Diagram ตัวอย่างนั้น จะกำหนดให้มีการเลื่อนตำแหน่งไปทางข้างบนเมื่อมีการรับข้อมูล 0 เข้ามา และจะเลื่อนตำแหน่งลงล่างเมื่อมีการรับข้อมูล 1 เข้ามา ซึ่งหลังจากที่มีการเลื่อนตำแหน่งที่ใช้พิจารณาแล้ว จะมีการพิจารณาถึงข้อมูลที่จะถูกส่งออกไปจากภาคเข้ารหัส ณ เวลานั้นๆ จากข้อมูลที่อยู่เหนือเส้นทางในตำแหน่งที่มีการพิจารณา

2.12 Trellis Diagram

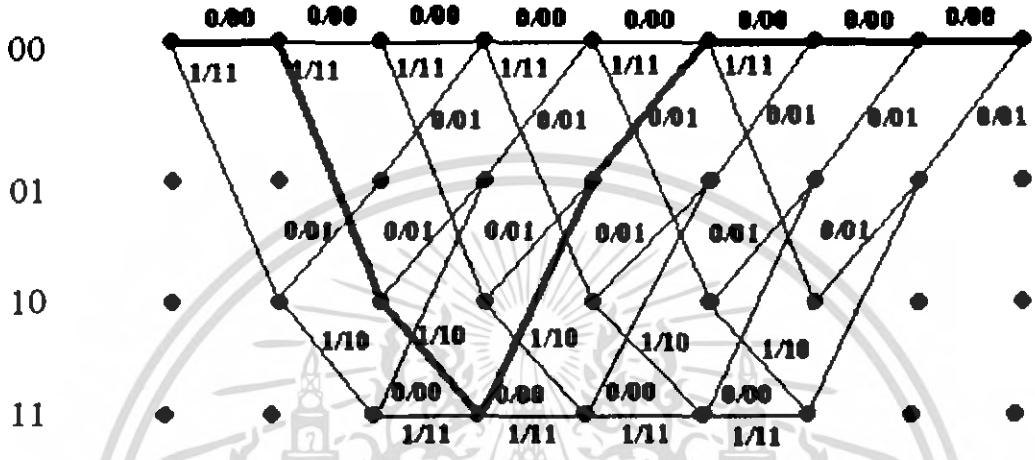
การแสดงการทำงานของวงจรเข้ารหัสโดยใช้ Trellis Diagram นั้น จะเป็นการนำการแสดงการทำงานของวงจรเข้ารหัสโดยใช้ State Diagram มาทำการเปลี่ยนแปลงรูปแบบให้อยู่ในอีกลักษณะหนึ่ง ที่แสดงถึงการเปลี่ยนแปลงของข้อมูลต่างๆภายในวงจรเข้ารหัส, ข้อมูลที่ป้อนเข้ามา และ Codeword ที่จะถูกส่งออกไป ณ เวลาต่างๆ โดยที่จะมีลักษณะของ Trellis Diagram ดังรูปต่อไปนี้



รูปที่ 2.11 Trellis Diagram

จากรูป จะเป็นการแสดงการทำงานของวงจรเข้ารหัสข้อมูลแบบ Convolution Codes ในรูปที่ 2.11 ที่มีการนำข้อมูลในอดีตจำนวน 2 บิต มาทำการประมวลผลร่วมกับข้อมูล ณ เวลานั้น (จำนวน State ทั้งหมดใน Trellis Diagram จะมีค่าเท่ากับ $2^2 = 4$ State) และจะมีข้อมูลป้อนเข้ามาภายในวงจรครั้งละ 1 บิต ($k_0=1$) ซึ่งในกรณีนี้ จะมีข้อมูลที่จะเข้ารหัสทั้งหมด 6 บิต ซึ่งเส้นทางต่างๆ ที่อยู่ใน Trellis Diagram นั้น จะแสดงถึงลักษณะการเปลี่ยนแปลง สภาวะของวงจรและตัวเลข x/y ที่อยู่เหนือทางเดินในแต่ละเส้นทางนั้น จะแสดงถึงข้อมูลที่ป้อนเข้ามา และ Codeword ที่จะถูกส่งออกไปเมื่อมีการป้อนข้อมูลนั้นเข้ามา ซึ่งเมื่อพิจารณาลักษณะของ Trellis Diagram แล้ว จะพบว่ารูปแบบของ Trellis Diagram ในแต่ละ State การทำงานนั้น จะมีลักษณะที่คล้ายกันแต่จะมีความแตกต่างกันเฉพาะส่วนหัวและท้าย ซึ่งเป็นผลมาจากข้อมูลที่เก็บอยู่ในวงจรนั้นจะมีค่าที่เริ่มต้นจากสภาวะที่มีข้อมูลเป็น 0 ทั้งหมด และจะจบลงที่สภาวะข้อมูลเป็น 0 เช่นกัน ดังนั้นเส้นทางอื่นๆ ที่ไม่ผ่านจุดที่มีข้อมูลเป็น 0 ทั้งหมด ณ จุดเริ่มต้นและจุดสุดท้ายนั้นจะไม่ถูกนำมาพิจารณา ดังนั้น

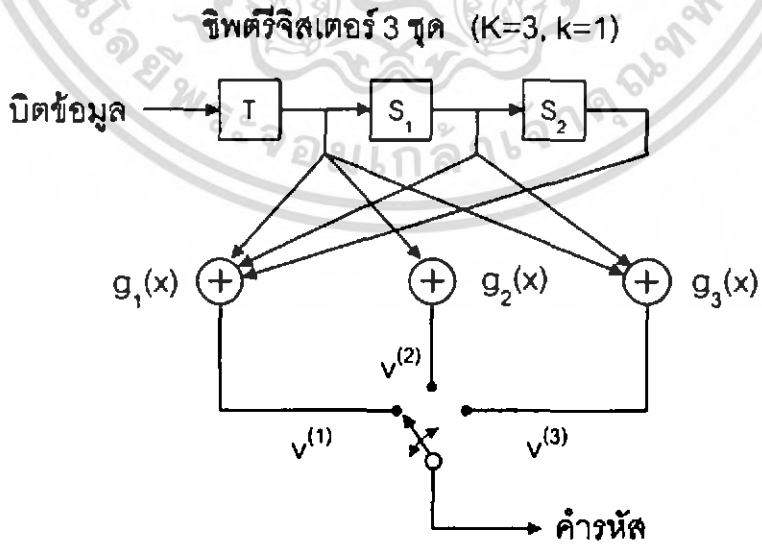
ขนาดของความยาวใน Trellis Diagram นั้นจึงขึ้นอยู่กับข้อมูลที่ถูกนำมาเข้ารหัส ตัวอย่างในการใช้งาน Trellis Diagram เช่น กรณีที่มีการป้อนข้อมูล 0 1 1 0 0 0 0 0 เข้ามาภายในวงจร จะสามารถใช้ Trellis Diagram ในการหาลักษณะของ Codeword ได้ดังรูปที่ 2.12



รูปที่ 2.12 การใช้งาน Trellis Diagram

ดังนั้นข้อมูลที่ได้หลังจากการเข้ารหัสจะมีค่าเท่ากับ 00 11 10 00 01 00 00 00

ตัวอย่างการเข้ารหัส Convolution ที่มีชิฟรีจิสเตอร์ 3 ชุด

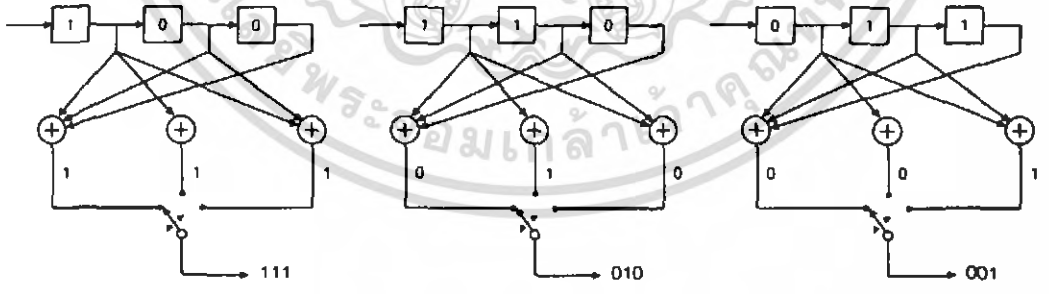


รูปที่ 2.13 ตัวอย่างวงจรเข้ารหัสคอนโวลูชัน

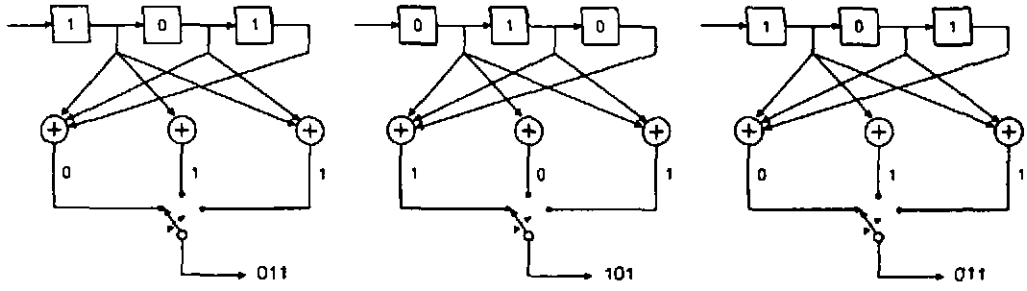
ในการทำความเข้าใจกับกลไกการทำงานของวิธีการเข้ารหัสคอนโวลูชัน เราจะอาศัยวงจรเข้ารหัสที่มีอัตราการเข้ารหัสเท่ากับ 1/3, K=3 และ k=1 ในรูปที่ 2.13 เป็นตัวอย่างในการอธิบาย จากวงจรในรูปจะเห็นว่า k=1 ดังนั้นจำนวนบิตที่จะเลื่อนเข้าสำหรับการเข้ารหัสแต่ละครั้งมีค่าเพียง 1 บิต ค่า K=3 และ k=1 หมายความว่าต้องใช้ชิฟต์รีจิสเตอร์จำนวน 3 ชุด และในรูป ประกอบด้วยวงจรพหุนามตัวกำเนิด 3 ชุด คือ $g_1(x), g_2(x)$ และ $g_3(x)$ โดยที่

$$g_1(x) = 1 + x + x^2$$
$$g_2(x) = 1$$
$$g_3(x) = 1 + x$$

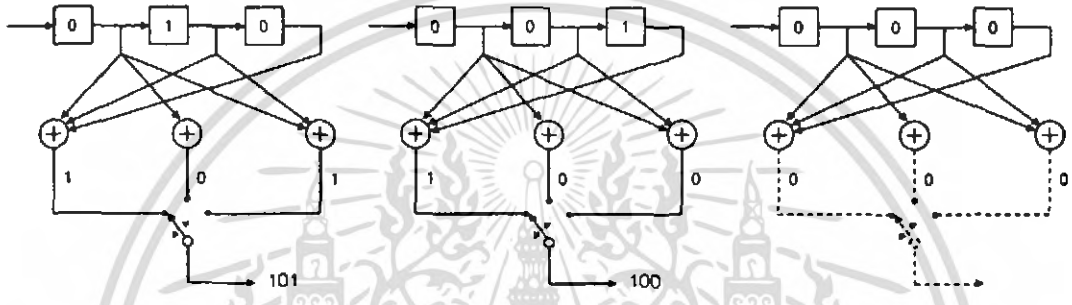
การเข้ารหัสข้อมูลแต่ละบิตจะให้เป็นคำรหัสที่มีจำนวนมากถึง 3 บิต ได้แก่ $v^{(1)}, v^{(2)}, v^{(3)}$ ค่าเหล่านี้คือค่าที่ได้จากขาออกของวงจรพหุนามของ $g_1(x), g_2(x)$ และ $g_3(x)$ นั่นเอง โดยกระบวนการอ่านค่าเหล่านี้จะวนสลับกันไป ยกตัวอย่างเช่น ถ้าข้อมูลหกบิตแรกที่ป้อนเข้าสู่วงจรเข้ารหัสมีค่าเป็น 110101 คำรหัสที่ได้จากวงจรเข้ารหัสจะมีค่าเท่ากับ 111 010 001 011 101 011 101 100 สำหรับรายละเอียดการเข้ารหัสอย่างเป็นขั้นตอนของทั้ง หกบิต สามารถดูได้จากรูปที่ 2.14 สังเกตุว่า ทุกครั้งที่จะสิ้นสุดการเข้ารหัสบิตข้อมูลแต่ละชุดจะต้องมีการเพิ่มบิตพิเศษเพิ่มเติม ที่เรียกว่า บิตหาง (Tail Bits) ต่อท้ายบิตข้อมูล ทั้งนี้ก็เพื่อปรับให้ชิฟต์รีจิสเตอร์กลับคืนสู่สถานะเดียวกับตอนเริ่มต้นซึ่งมีค่าเป็น 0 ทั้งหมด



(ก) รายละเอียดการเข้ารหัสสามบิตแรก



(ข) รายละเอียดการเข้ารหัสสามบิตที่เหลือ



(ค) รายละเอียดการเข้ารหัสบิตหางที่มีค่าเป็นศูนย์ทั้งหมด

รูปที่ 2.14 ตัวอย่างขั้นตอนการเข้ารหัสของข้อมูล 110101 โดยใช้วงจรเข้ารหัสในรูปที่ 2.13

2.13 ผลที่ได้จากการเข้ารหัสข้อมูลแบบ Convolution Codes

การเข้ารหัสแบบ Convolution Codes นั้น จะถูกนำมาใช้สำหรับเพิ่มความน่าเชื่อถือของระบบสื่อสารให้ข้อมูลที่ถูกส่งไปนั้นเกิดความผิดพลาดในการส่งข้อมูลที่ลดลง โดยที่ไม่จำเป็นต้องลดอัตราเร็วในการส่งข้อมูลลงหรือเพิ่มกำลังของเครื่องส่งให้สูงขึ้น ซึ่งในการวิเคราะห์ความสามารถในการป้องกันความผิดพลาดของการเข้ารหัสนั้น จะมีการใช้ค่า Coding Gain ในการแสดงถึงความสามารถในการป้องกันความผิดพลาดของข้อมูล ซึ่งจะเป็นค่าที่แสดงถึงความแตกต่างระหว่างกำลังงานที่ใช้ในการส่งข้อมูลในกรณีที่มีการใช้การเข้ารหัสในระบบสื่อสารกับกรณีที่ไม่มีการเข้ารหัสข้อมูลที่มีค่าของอัตราการเกิดความผิดพลาดเท่ากัน

2.14 ลักษณะของข้อมูลที่ได้จากการเข้ารหัสแบบ Convolution Codes

ในส่วนของการพิจารณาลักษณะของข้อมูลที่ได้จากการเข้ารหัสแบบ Convolution Codes สามารถแบ่งรูปแบบในการเข้ารหัสออกเป็น 2 รูปแบบด้วยกันตามลักษณะของข้อมูลที่ได้หลังจากการเข้ารหัส ได้แก่ Systematic และ Nonsystematic Convolution Codes สำหรับกรณีของการ

เข้ารหัสแบบ Systematic Convolution Codes นั้น ข้อมูลที่ผ่านการเข้ารหัสในแต่ละครั้งซึ่งถูกส่งออกมาจากวงจรเข้ารหัสนั้น จะประกอบไปด้วยข้อมูลสองส่วนด้วยกันคือส่วนของข้อมูลดิบและส่วนของข้อมูลที่ผ่านการเข้ารหัส แต่สำหรับกรณีของ Nonsystematic Convolution Codes ข้อมูลที่ได้จากการเข้ารหัสในแต่ละครั้ง จะอยู่ในรูปของข้อมูลที่ผ่านการเข้ารหัสแล้วเท่านั้น ซึ่งข้อมูลที่ได้จากการเข้ารหัสในกรณีของ Nonsystematic Convolution Codes จะมีค่าความแตกต่างของข้อมูล (Distance) ที่มีค่ามากกว่ากรณีของ Systematic Convolution Codes ซึ่งจะทำให้ความสามารถในการป้องกันความผิดพลาดของข้อมูลของ Nonsystematic Convolution Codes มีค่าที่มากกว่ากรณีของ Systematic Convolution Codes

สำหรับการถอดรหัสข้อมูลสำหรับวิธีการเข้ารหัสแบบ Convolution Codes นั้น จะมีรูปแบบในการถอดรหัสอยู่หลายวิธีการด้วยกัน โดยในการถอดรหัสนั้น จะมีการนำ Trellis Diagram ของวงจรเข้ารหัสมาใช้ในการทำงานของภาคถอดรหัส

2.15 วงจรเข้ารหัสแบบ *Recursive Systematic Convolution Codes*

สำหรับการเข้ารหัสข้อมูลแบบ Recursive Systematic Convolution Codes หรือ RSC เป็นรูปแบบของการเข้ารหัสข้อมูลที่มีการนำคุณสมบัติของการเข้ารหัสแบบ Nonsystematic และ Systematic Convolution Codes รวมเข้าด้วยกันเพื่อให้ข้อมูลที่ได้จากการเข้ารหัสนั้นมีลักษณะของข้อมูลเป็นแบบ Systematic ที่มีการส่งข้อมูลดิบไปพร้อมกับข้อมูลที่ผ่านการเข้ารหัส แต่ยังคงมีค่าความแตกต่างของข้อมูล (Distance) ซึ่งเป็นค่าที่แสดงถึงความสามารถในการป้องกันความผิดพลาดของข้อมูลมีค่าเท่ากับกรณีของ Nonsystematic ซึ่งผลที่ได้นั้นจะทำให้ความสามารถในการป้องกันความผิดพลาดของข้อมูลในกรณีของการเข้ารหัสแบบ RSC นั้นมีค่ามากกว่ากรณีของ Nonsystematic Convolution Codes ณ สภาวะของระบบสื่อสารที่มีค่า S/N ต่ำ และมีการเข้ารหัสด้วยอัตราการเข้ารหัสที่สูง สำหรับวงจรที่ใช้ในการเข้ารหัสแบบ RSC จะเป็นวงจรเข้ารหัสที่มีการดัดแปลงมาจากวงจรเข้ารหัสแบบ Nonsystematic Convolution Codes โดยการนำสัญญาณข้อมูลที่ได้จากการเข้ารหัสสัญญาณใดสัญญาณหนึ่งป้อนกลับ (Feedback) มาเป็นข้อมูลอินพุทของวงจรเข้ารหัส

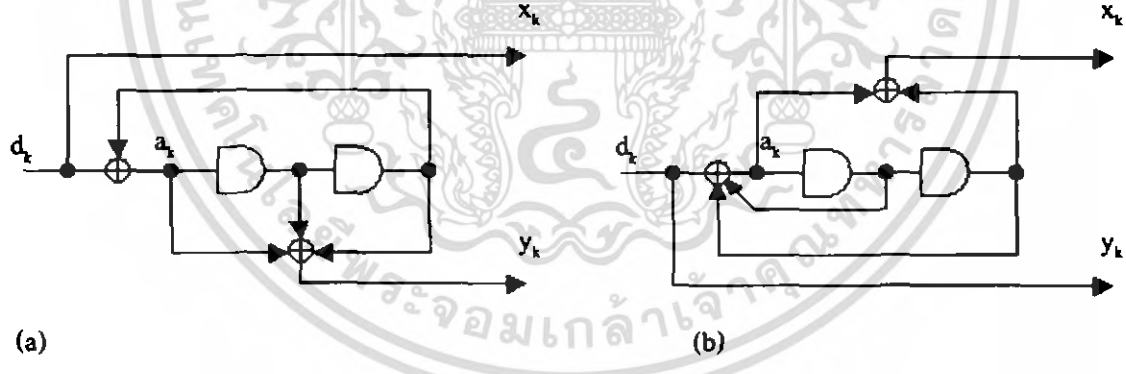
ในกรณีของวงจรเข้ารหัสแบบ RSC ที่มีอัตราการเข้ารหัส (Code Rate) เท่ากับ ½ จะมีการออกแบบโดยการพิจารณาจากวงจรเข้ารหัสแบบ Nonsystematic Convolution Codes ที่มีอัตราการเข้ารหัสเท่ากับ ½ และมีค่า Generator Matrix เท่ากับ

$$G_{NSC} = [\, g1 \;,\; g2 \,] \tag{2.4}$$

โดยที่ค่า g1 และ g2 นั้นเป็นตัวเลขฐาน 8 ซึ่งแสดงถึงรูปแบบในการนำข้อมูลที่อยู่ในวงจรเข้ารหัสมาใช้ในการคำนวณหาค่าของข้อมูลที่ได้จากการเข้ารหัสทางเอาท์พุทสัญญาณ 1 และ 2 ตามลำดับ โดยในการเปลี่ยนวงจรให้อยู่ในรูปแบบของ RSC นั้น จะมีการเปลี่ยนแปลงการทำงานของวงจรเข้ารหัสให้การทำงานในรูปของการส่งข้อมูลป้อนกลับ โดยการนำสัญญาณเอาท์พุทที่ได้จากการเข้ารหัสสัญญาณใดสัญญาณหนึ่งมาใช้คำนวณร่วมกับข้อมูลที่ถูกป้อนเข้ามา ดังนั้น ค่า Generator Matrix ของวงจรเข้ารหัสที่ถูกเปลี่ยนแปลงให้มีการทำงานในรูปแบบของ RSC นั้น จะมีค่าเท่ากับ

$$G_{RSC} = [\, 1 \;,\; g2/g1 \,] \text{ หรือ } G_{RSC} = [\, g1/g2 \;,\; 1 \,] \tag{2.5}$$

ตัวอย่างเช่นในกรณีของวงจรเข้ารหัสในรูปที่ 2.6 นั้น เมื่อมีการเปลี่ยนให้อยู่ในรูปแบบของวงจรเข้ารหัสแบบ RSC โดยการนำสัญญาณเอาท์พุท $Y_k^{(1)}$ และ $Y_k^{(2)}$ มาใช้ในการป้อนข้อมูลกลับจะมีลักษณะดังรูปที่ 2.15 (a) และ (b) ตามลำดับ



(a) (b)

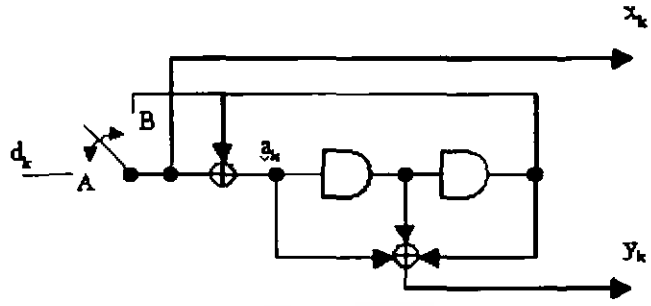
รูปที่ 2.15 ตัวอย่างวงจรเข้ารหัสแบบ RSC (a) กรณีใช้สัญญาณ $Y_k^{(1)}$ ป้อนกลับ , (b) กรณีใช้สัญญาณ $Y_k^{(2)}$ ป้อนกลับ

โดยที่ a_k นั้นจะเป็นข้อมูลที่จะถูกส่งเข้ามายังเมมโมรี่ภายในวงจรเพื่อใช้ในการคำนวณ โดยจะสามารถคำนวณหาค่าได้จากสมการ

$$a_k = \begin{Bmatrix} d_k + \sum_{i=0}^{K-1} g_{1,i} a_{k-i} & \text{mod.2} & if & x_k = d_k \\ d_k + \sum_{i=0}^{K-1} g_{2,i} a_{k-i} & \text{mod.2} & if & y_k = d_k \end{Bmatrix} \quad (2.6)$$

โดยสำหรับการทำงานต่างๆของวงจรนั้น จะมีการทำงานเช่นเดียวกับกรณีของวงจรเข้ารหัสแบบ Convolution Codes แต่เนื่องจากมีการทำงานในลักษณะของการป้อนข้อมูลกลับ ดังนั้นถ้าหากส่งข้อมูลที่เป็น 0 เข้ามาภายในวงจร อาจจะไม่สามารถทำให้สภาวะของวงจรกลับสู่สภาวะเริ่มต้นได้ ซึ่งในการใช้งานวงจรเข้ารหัสแบบ RSC ในการเข้ารหัสแบบ Turbo Codes นั้นจะมีการทำงานใน 2 ลักษณะด้วยกันได้แก่ กรณีที่ไม่ต้องมีการทำให้ข้อมูลที่ถูกเก็บไว้ในวงจรเข้ารหัสกลับสู่สภาวะที่มีข้อมูลเป็น 0 ทั้งหมดเมื่อสิ้นสุดการทำงาน และกรณีที่มีการทำให้ข้อมูลที่ถูกเก็บไว้ในวงจรเข้ารหัสกลับสู่สภาวะที่มีข้อมูลเป็น 0 ทั้งหมดเมื่อสิ้นสุดการทำงาน

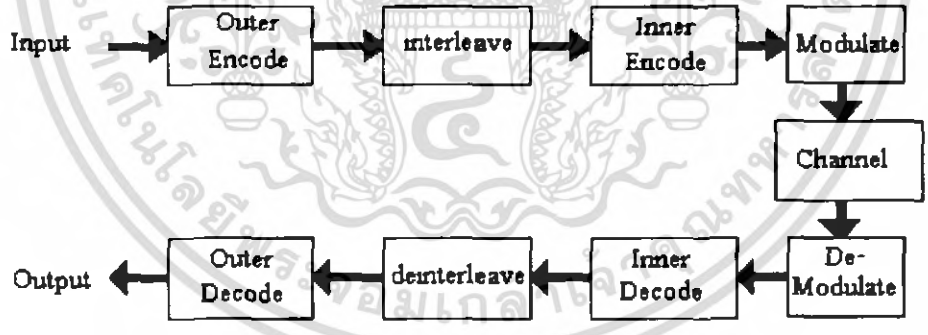
สำหรับกรณีที่ไม่มีการทำให้ข้อมูลที่ถูกเก็บไว้ในวงจรเข้ารหัสกลับสู่สภาวะที่มีข้อมูลเป็น 0 ทั้งหมดเมื่อสิ้นสุดการทำงานนั้น ภายหลังจากที่มีการป้อนข้อมูลดิบทั้งหมดแล้ว จะถือว่าเป็นการสิ้นสุดการทำงานของภาคเข้ารหัส ซึ่งข้อมูลที่อยู่ในวงจรเข้ารหัส ณ เวลานั้น อาจจะไม่มีข้อมูลที่เป็น 0 ทั้งหมด และสำหรับกรณีที่มีการทำให้ข้อมูลที่ถูกเก็บไว้ในวงจรเข้ารหัสกลับสู่สภาวะที่มีข้อมูลเป็น 0 ทั้งหมดเมื่อสิ้นสุดการทำงานนั้น หลังจากที่มีการป้อนข้อมูลดิบทั้งหมดแล้ว จะมีการส่งข้อมูลเข้ามาภายในวงจรเข้ารหัสเพื่อทำให้ข้อมูลที่อยู่ในวงจรเข้ารหัสกลับสู่สภาวะที่มีข้อมูลเป็น 0 ทั้งหมด โดยข้อมูลที่ถูกป้อนเข้ามานั้นจะต้องมีค่าที่สามารถทำให้ข้อมูล a_k ที่ได้จากการนำข้อมูลดังกล่าวมาทำการ Modulo-2 กับสัญญาณข้อมูลที่ถูกป้อนกลับมา ให้มีค่าเป็น 0 เพื่อทำให้ข้อมูลที่อยู่ในวงจรเข้ารหัสมีค่าเป็น 0 ทั้งหมด ดังเช่นกรณีของวงจรเข้ารหัสแบบ RSC ในรูปที่ 2.16 นั้น จะเป็นการดัดแปลงวงจรเข้ารหัสในรูปที่ 2.15 (a) ให้สามารถทำให้สภาวะของข้อมูลในวงจรเข้ารหัสกลับสู่สภาวะที่มีค่าเป็น 0 ได้ โดยในกรณีที่ยังมีการเข้ารหัสข้อมูลไม่ครบทุกบิต จะมีการสลับสวิตช์ไปยังตำแหน่ง A แต่เมื่อข้อมูลทั้งหมดได้ผ่านการเข้ารหัสแล้ว จะมีการสลับสวิตช์มายังตำแหน่ง B เพื่อส่งข้อมูลที่สามารถทำให้ข้อมูลในวงจรกลับสู่สภาวะที่มีค่าเป็น 0 ได้



รูปที่ 2.16 ตัวอย่างวงจรเข้ารหัสแบบ RSC

2.16 การเข้ารหัสข้อมูลแบบ Concatenation

สำหรับวิธีการเข้ารหัสข้อมูลแบบ Concatenation จะเป็นรูปแบบในการเข้ารหัสข้อมูลที่จะมีการใช้การเข้ารหัสข้อมูลตั้งแต่ 2 รูปแบบขึ้นไปมาใช้งานในการเข้ารหัสข้อมูลร่วมกัน โดยมีวัตถุประสงค์เพื่อให้ได้รูปแบบในการเข้ารหัสข้อมูลที่มีความสามารถในการป้องกันความผิดพลาดได้สูงขึ้น โดยที่ไม่มีการใช้รูปแบบของวงจรเข้ารหัสที่มีความซับซ้อน (Complex) มากขึ้น โดยที่รูปแบบในการเข้ารหัสแบบ Concatenate ในรูปแบบทั่วไปนั้น จะมีลักษณะการทำงานเป็นแบบที่เรียกว่า Serial Concatenate ซึ่งจะมีลักษณะของวงจรดังรูปที่ 2.17

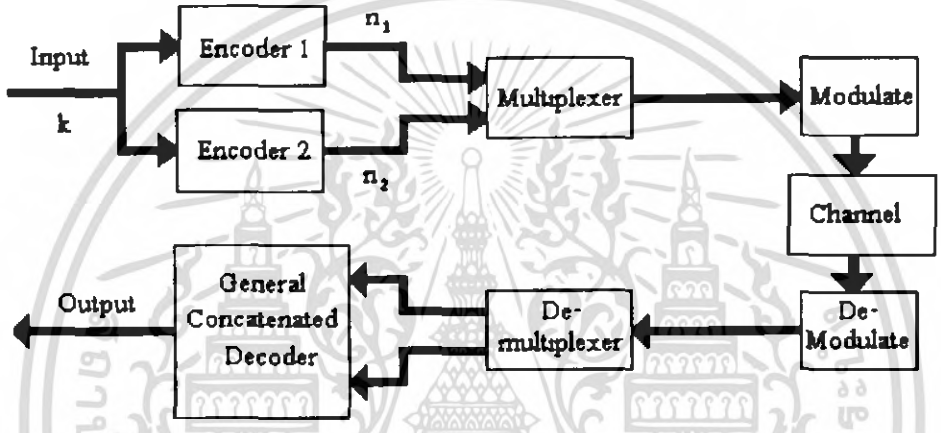


รูปที่ 2.17 การเข้ารหัสแบบ Serial Concatenate

จากรูปที่ 2.17 เป็นตัวอย่างของการนำการเข้ารหัสข้อมูล 2 รูปแบบมาทำการใช้งานร่วมกันในรูปแบบของ Serial Concatenate โดยจะเป็นการนำข้อมูลที่จะเข้ารหัสมาผ่านการเข้ารหัส 2 ครั้งด้วยกัน โดยในการเข้ารหัสในครั้งแรกนั้น จะถูกเรียกว่าเป็น Outer Encode ซึ่งข้อมูลที่ได้จากการเข้ารหัสในวงจรแรกนั้น จะถูกทำการ Interleave เพื่อป้องกันการเกิดการผิดพลาดของข้อมูลแบบ Burst Error จากนั้นจึงถูกนำมาเข้ารหัสครั้งที่สอง ซึ่งถูกเรียกว่าเป็น Inner Encoder และในทาง

กลับกัน เมื่อจะทำการถอดรหัสข้อมูลที่รับได้ที่ปลายทาง จะต้องมีการนำข้อมูลที่รับได้มาผ่านการถอดรหัส Inner Decoder จากนั้นจึงนำมาทำการ Deinterleave ข้อมูลและส่งมายังภาคถอดรหัส Outer Decoder

สำหรับการเข้ารหัสข้อมูลในรูปแบบของ Concatenate ในอีกลักษณะหนึ่ง ซึ่งจะถูกนำมาเป็นรูปแบบในการเข้ารหัสข้อมูลสำหรับ Turbo Codes นั้น จะมีลักษณะในการทำงานเป็นแบบ Parallel Concatenate ซึ่งจะมีลักษณะในการเข้ารหัสข้อมูลดังรูปที่ 2.18



รูปที่ 2.18 การเข้ารหัสแบบ Parallel Concatenate

สำหรับในการทำงานของวงจรถอดรหัสแบบ Turbo Codes นั้น จะมีการนำรูปแบบในการเข้ารหัสข้อมูลแบบ Parallel Concatenate มาใช้สำหรับการเข้ารหัสข้อมูล แต่สำหรับรูปแบบในการถอดรหัสแบบ Turbo Codes นั้น จะมีการนำรูปแบบในการถอดรหัสข้อมูลแบบ Serial Concatenated มาทำการใช้งาน

2.17 การ Interleave ข้อมูล

การ Interleave ข้อมูลนั้นเป็นกระบวนการในการเปลี่ยนแปลงการจัดเรียงข้อมูลดิจิตอล ให้มีลักษณะที่แตกต่างออกไปจากเดิม ซึ่งถูกนำมาใช้ในระบบสื่อสารเพื่อป้องกันการผิดพลาดของข้อมูลในรูปแบบของ Burst Error แต่สำหรับกรณีของวงจรเข้ารหัสแบบ Turbo Codes นั้น จะมีการนำการ Interleave มาใช้งานโดยมีจุดประสงค์เพื่อทำให้ข้อมูลที่ได้จากการเข้ารหัสจากวงจรเข้ารหัสต่างๆนั้น มีลักษณะของข้อมูลที่ไม่มีความสัมพันธ์ซึ่งกันและกัน โดยจะเป็นการนำข้อมูลคิบที่จะทำการเข้ารหัสในวงจรเข้ารหัสต่างๆ มาผ่านการ Interleave เพื่อทำการเปลี่ยนแปลงรูปแบบของข้อมูล (Permutation) เพื่อให้ข้อมูลที่ถูกเข้ารหัสนั้นมีลักษณะที่แตกต่างกัน

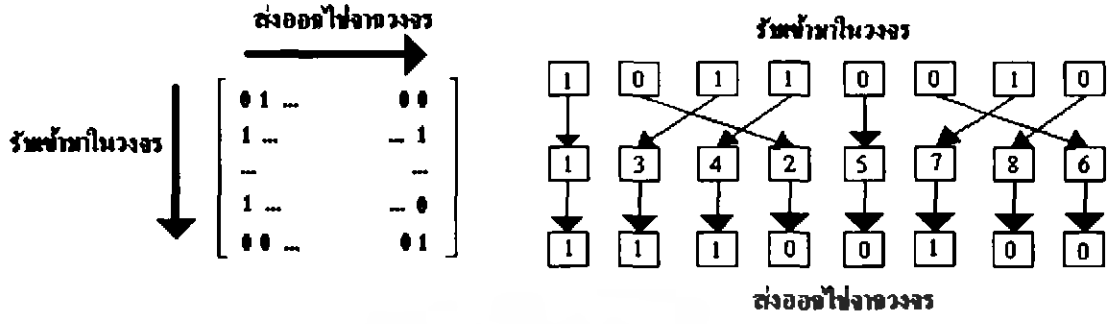
สำหรับวิธีการที่ใช้ในการ Interleave ข้อมูลนั้น จะมีรูปแบบในการทำงานอยู่หลายวิธีการด้วยกัน และในการทำงานแต่ละวิธีการนั้น จะให้ผลลัพธ์ในการทำงานที่แตกต่างกัน โดยสำหรับตัวอย่างของรูปแบบในการ Interleave ข้อมูล ได้แก่ วิธีการ Interleave ข้อมูลแบบ Block Interleave และ Random Interleave

2.18 Block Interleave

สำหรับการ Interleave ข้อมูลด้วยวิธีการ Interleave ข้อมูลแบบ Block Interleave นั้น จะเป็นวิธีที่การมีการนิยมใช้งานสำหรับระบบสื่อสาร โดยในการทำงานเพื่อเปลี่ยนแปลงรูปแบบของข้อมูลนั้นจะมีลักษณะการทำงานที่สามารถแสดงการทำงานโดยใช้เมตริกดังรูปที่ 2.19 (a) โดยในการทำงานนั้น จะเป็นการดึงข้อมูลที่จะทำการเปลี่ยนแปลงรูปแบบเข้ามาเก็บไว้ในเมมโมรี่โดยจะมีการเก็บข้อมูลเรียงจากบนลงล่าง และ ซ้ายไปขวา โดยจะมีการเก็บข้อมูลในลักษณะนี้ต่อไปเรื่อยๆ จนกระทั่งเมมโมรี่ในวงจรมีการเก็บข้อมูลไว้ทุกตำแหน่งแล้ว จากนั้นจึงส่งข้อมูลออกจากวงจรโดยการเรียงลำดับการส่งจากซ้ายไปขวา และ บนลงล่าง และจะมีการส่งข้อมูลในลักษณะนี้เรื่อยๆ จนกระทั่งข้อมูลทุกตำแหน่งในเมมโมรี่ถูกส่งออกจากวงจรแล้ว จากนั้นจึงมีการรับข้อมูลชุดต่อไปเข้ามาในวงจรเพื่อทำงานในครั้งต่อไป โดยจะมีการทำงานในลักษณะนี้ จนกระทั่งข้อมูลทั้งหมดได้ผ่านกระบวนการ Interleave แล้ว จึงสิ้นสุดการทำงาน

2.19 Random (Pseudo-Random) Interleave

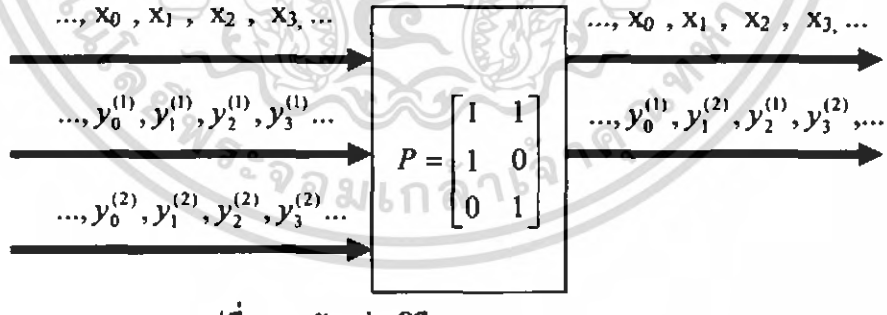
ในการ Interleave ข้อมูลด้วยวิธีการแบบ Random Interleave นั้น จะมีการใช้การสุ่มตัวเลขขึ้นมาเพื่อใช้สำหรับกำหนดรูปแบบในการเปลี่ยนแปลงตำแหน่งของข้อมูล โดยในการนำงานนั้นจะมีการนำข้อมูลมาครั้งละ 1 ชุด เพื่อทำการสลับตำแหน่งของข้อมูลให้อยู่ในตำแหน่งต่างๆตามรูปแบบของค่าที่ได้จากการสุ่ม โดยจะมีลักษณะในการทำงานดังตัวอย่างในรูปที่ 2.19 (b) ที่เป็นตัวอย่างของการ Interleave แบบ Random Interleave ที่มีการดึงข้อมูลเข้ามาภายในวงจรครั้งละ 8 บิตเพื่อเปลี่ยนแปลงตำแหน่งของข้อมูล



รูปที่ 2.19 การ Interleave ข้อมูล (a) Block Interleave , (b) Random Interleave

2.20 การ Punctured ข้อมูล

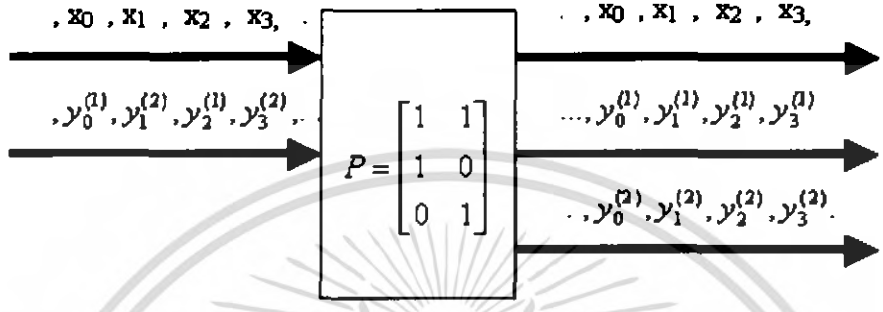
การทำงานของภาค Puncture ถูกนำมาใช้ในวงจรเข้ารหัส โดยมีวัตถุประสงค์เพื่อที่จะทำให้จำนวนบิตของข้อมูลที่จะถูกส่งผ่านระบบสื่อสารนั้น มีจำนวนที่ลดลง โดยจะเป็นการการลบข้อมูลที่จะทำการส่งบางส่วนออกไปตามรูปแบบที่มีการกำหนดไว้ โดยในการแสดงถึงรูปแบบในการทำงานนั้น จะมีการแสดงถึงรูปแบบในการลบข้อมูลที่จะทำการส่งโดยใช้ตัวแปรเมตริกที่จะแสดงถึงรูปแบบในการลบข้อมูลที่ได้จากการเข้ารหัส ดังเช่นเมื่อพิจารณาข้อมูลที่ได้จากการเข้ารหัสจากวงจรเข้ารหัสแบบ Turbo Codes ในรูปที่ 2.5 จะมีลักษณะของข้อมูลที่ได้จากการ Puncture เป็นดังรูปที่ 2.20



รูปที่ 2.20 ตัวอย่างวิธีการ Puncturing

โดยในกรณีของวงจรในรูปที่ 2.18 ซึ่งเป็นตัวอย่างที่แสดงถึงรูปในการทำงานของภาค Puncture นั้น จะเป็นการลดจำนวนบิตของข้อมูลในส่วนของข้อมูลที่ได้จากการทำงานของวงจรเข้ารหัส ($y_k^{(1)}$และ $y_k^{(2)}$) ให้มีค่าลดลง โดยจะใช้การสลับกันส่งข้อมูลระหว่างสัญญาณ $y_k^{(1)}$ และ $y_k^{(2)}$ ดังนั้นจะทำให้อัตราการเข้ารหัสข้อมูลของการเข้ารหัสนั้นมีค่าที่เพิ่มขึ้นจาก 1/3 เป็น 1/2 ในทางกลับกัน เมื่อข้อมูลต่างๆนั้นถูกส่งมาถึงปลายทาง จะต้องมีการนำข้อมูลมาทำการ

Depuncture เพื่อเปลี่ยนแปลงรูปแบบของข้อมูลที่รับได้ให้กลับมาอยู่ในรูปแบบเดิม ตัวอย่างเช่นในกรณีที่มีการ Puncture ข้อมูลที่ได้จากการถอดรหัสดังรูปที่ 2.20 จะได้ว่าเมื่อข้อมูลถูกส่งมาถึงภาครับ จะต้องมีการนำข้อมูลมาผ่านภาค Depuncture ซึ่งจะมีรูปแบบในการทำงานดังรูปที่ 2.21



รูปที่ 2.21 ตัวอย่างวิธีการ Depuncture

2.21 การถอดรหัสข้อมูล

ในการถอดรหัสสำหรับกรณีของ Convolution Codes นั้น จะมีวิธีการที่ใช้ในการถอดรหัสอยู่หลายวิธีการด้วยกัน ซึ่งสามารถที่จะแบ่งรูปแบบของการถอดรหัสออกได้เป็น 3 รูปแบบด้วยกัน ได้แก่

2.21.1 Sequential Decoding

เป็นวิธีการถอดรหัสที่มีการนำ Tree Diagram ของวงจรเข้ารหัสมาใช้ในการออกแบบการทำงานของภาคถอดรหัส โดยที่จะมีการนำข้อมูลที่รับเข้ามาได้ มาทำการเปรียบเทียบกับข้อมูลที่อยู่ในเส้นทางต่างๆใน Tree Diagram เพื่อค้นหาเส้นทางที่มีข้อมูลที่ใกล้เคียงกับข้อมูลที่รับเข้ามามากที่สุด

2.21.2 Majority-logic หรือ Threshold Decoding

เป็นวิธีการถอดรหัสที่มีการนำการคำนวณทาง Topological มาใช้ในการออกแบบการถอดรหัสเพื่อที่จะทำให้มีการถอดรหัสที่ใช้งานได้ง่ายขึ้น แต่จะทำให้ประสิทธิภาพในการทำงานนั้นลดลง

2.21.3 Viterbi Decoding

เป็นวิธีการถอดรหัสข้อมูลที่มีการนิยมใช้งานมากที่สุด ซึ่งจะมีลักษณะการทำงานเป็นแบบ Maximum-likelihood Decoding Algorithm ซึ่งจะหมายถึงว่าในการทำงานของการถอดรหัสนั้น จะ

เป็นการนำข้อมูลที่รับได้ที่ปลายทาง นั้นไปทำการประมวลผลเพื่อค้นหาเส้นทางใน Trellis Diagram ที่มีลักษณะที่ใกล้เคียงกับข้อมูลที่รับได้จากระบบสื่อสารมากที่สุด

2.22 การถอดรหัสข้อมูลสำหรับ Turbo Codes

สำหรับวิธีการที่ใช้ในการถอดรหัสข้อมูลสำหรับ Turbo Codes จะมีรูปแบบในการทำงานเป็นแบบ Iterative Decoding โดยในการทำงานนั้น จะเป็นการนำข้อมูลที่รับเข้ามาได้มาผ่านกระบวนการเพื่อคำนวณหาความเป็นไปได้ของค่าของข้อมูลดิบที่จะถูกส่งมา ณ เวลาต่างๆ ที่อยู่ในรูปของค่าที่เรียกว่า Extrinsic Information และจะเป็นข้อมูลส่วนที่ถูกนำมาใช้ในการปรับปรุงการคำนวณในการถอดรหัสครั้งต่อไปเพื่อให้มีความแม่นยำมากขึ้น โดยจะมีการวนรอบในลักษณะนี้ไปเรื่อยๆ จนกระทั่งมีการทำงานครบตามที่กำหนด จากนั้นจึงนำค่า Log A Posteriori Probability (LAPP) ที่ได้จากการทำงานมาใช้ในการตัดสินใจค่าของข้อมูลดิบที่ได้จากการทำงานของวงจรถอดรหัส

สำหรับการทำงานของวงจรถอดรหัสที่ใช้ในการถอดรหัสแบบ Turbo Codes นั้น จะมีการนำหลักการในการถอดรหัสแบบ Viterbi Coding และ Log-Map มาใช้ในกระบวนการถอดรหัสของวงจรถอดรหัส DEC1 และ DEC2 โดยจะมีรายละเอียดต่างๆในการทำงานดังต่อไปนี้

2.23 วิธีการถอดรหัสแบบ MAP

สำหรับวิธีการถอดรหัสแบบ Maximum a Posteriori หรือ MAP นั้น เป็นรูปแบบในการถอดรหัสข้อมูลที่มีการพิจาณาหาค่าของข้อมูลดิบที่ได้หลังจากการถอดรหัสโดยการพิจารณาจากนำข้อมูลจำนวน n_0 บิตที่รับได้ที่ปลายทางในแต่ละช่วงเวลามาทำการคำนวณหาความเป็นไปได้ว่าข้อมูลดิบที่ถูกส่งมา ณ เวลานั้นมีความเป็นไปได้ที่มีจะมีค่าเป็นข้อมูลใด โดยสำหรับรูปแบบและวิธีการในการทำงานนั้น จะมีวิธีการในการทำงานอยู่หลายรูปแบบด้วยกัน แต่วิธีการที่มีการนิยมใช้งานมากได้แก่วิธีการ BCJR Algorithm ซึ่งถูกค้นพบโดย Bahl ในปี พ.ศ 2517 (ค.ศ.1974) โดยในการถอดรหัสข้อมูลเพื่อหาค่าของข้อมูลดิบ d_k ที่คาดว่าจะถูกส่งมา ณ เวลานั้นๆว่าควรจะมีค่าเท่าใด โดยจะมีการพิจารณาจากค่าที่เรียกว่า Log a Posteriori Probability (LAPP) ซึ่งจะมีค่าเท่ากับ

$$L(u_k) = \log\left(\frac{P(d_k = +1 \mid R_k)}{P(d_k = -1 \mid R_k)}\right) \tag{2.7}$$

โดยตัวแปร d_k นั้นจะเป็นตัวแปรที่ใช้แทนข้อมูลดิบที่ได้จากการถอดรหัส ณ เวลานั้นๆ โดยสำหรับกรณีสมการที่ 10 นี้เป็นกรณีที่มีการส่งข้อมูลโดยใช้วิธีการมอดูเลตแบบ BPSK ดังนั้น ข้อมูลดิบที่จะถูกส่งมานั้นจะมีค่าเท่ากับ +1 หรือ –1 ซึ่งแทนข้อมูลไบนารี่ 1 หรือ 0 เท่านั้น และสำหรับค่า LAPP นี้จะเป็นตัวแปรที่จะถูกนำมาใช้สำหรับการตัดสินใจว่าข้อมูลที่จะถูกส่งมา ณ เวลานั้นๆ น่าจะมีค่าเป็นเท่าใด และตัวแปร R_k นั้นเป็นตัวแปรที่แทนสัญญาณข้อมูลที่รับได้ ณ ช่วงเวลาต่างๆ โดยถ้าหากว่าค่า LAPP ที่ได้จากการคำนวณชุดข้อมูล ณ เวลานั้นๆมีค่ามากกว่า 0 จะมีการตัดสินใจให้ข้อมูลดิบที่ได้จากการถอดรหัส ณ เวลานั้นมีค่าเท่ากับ +1 ซึ่งแทนข้อมูลไบนารี่ที่มีค่าเป็น 1 แต่ในกรณีที่ค่า LAPP ที่ได้นั้นมีค่าน้อยกว่า 0 แล้วจะมีการตัดสินใจให้ข้อมูลดิบที่ได้จากการถอดรหัส ณ เวลานั้นมีค่าเท่ากับ –1 ซึ่งแทนข้อมูลไบนารี่ที่มีค่าเป็น 0

ในกรณีของการถอดรหัสแบบ Turbo Codes นั้น เนื่องจากมีการใช้วงจรในการเข้ารหัสแบบ Recursive Systematic Convolution Codes มาใช้ในการเข้ารหัสข้อมูล ดังนั้นจึงต้องมีการเปลี่ยนแปลงรูปแบบในการคำนวณหาค่าของ LAPP เพื่อให้เหมาะสมกับการทำงาน โดยสำหรับการคำนวณหาค่า LAPP ในกรณีของการเข้ารหัสแบบ Turbo Codes นั้น จะมีการคำนวณหาค่า LAPP ของชุดข้อมูลที่รับได้ในแต่ละช่วงเวลา โดยการคำนวณจากค่าที่เรียกว่า Joint Probability ซึ่งจะมีค่าเท่ากับ

$$L(u_k) = \log\left[\frac{\sum_m \lambda_k^{1,m}}{\sum_m \lambda_k^{0,m}}\right] = \log\left[\frac{\sum_m \sum_{m'} \Pr\{d_k = 1, S_k = m, S_{k-1} = m', R_1^{k-1}, R_{k+1}^N\}}{\sum_m \sum_{m'} \Pr\{d_k = 0, S_k = m, S_{k-1} = m', R_1^{k-1}, R_{k+1}^N\}}\right] \tag{2.8}$$

เมื่อทำการเปลี่ยนแปลงรูปแบบของสมการโดยใช้กฎของเบย์ (Bayes's Rule) จะได้ว่าค่า LAPP ที่ได้นั้นจะมีค่าเท่ากับสมการ

$$L(u_k) = \log\left[\frac{\sum_m \sum_{m'} \Pr\{R_{k+1}^N \mid S_k = m\}.\Pr\{S_{k-1} = m' \mid R_1^{k-1}\}.\Pr\{d_k = 1, S_k = m, R_k \mid S_{k-1} = m'\}}{\sum_m \sum_{m'} \Pr\{R_{k+1}^N \mid S_k = m\}.\Pr\{S_{k-1} = m' \mid R_1^{k-1}\}.\Pr\{d_k = 0, S_k = m, R_k \mid S_{k-1} = m'\}}\right] \tag{2.9}$$

จะพบว่าลักษณะของสมการที่ใช้ในการคำนวณนั้นจะประกอบไปด้วยการคำนวณระหว่างค่าความน่าจะเป็น 3 ค่าด้วยกัน ดังนั้นจึงได้มีการนิยามตัวแปรขึ้นมา 3 ตัวแปรด้วยกันเพื่อใช้ในการคำนวณหาค่า Joint Probability ได้แก่ตัวแปร Forward State Matric ($\alpha_k(m)$), Reverse State

Matric ($\beta_k(m)$) และ Branch Metric ($\gamma_k(R_k, m', m)$) โดยที่ตัวแปรต่างๆนั้นจะมีการนิยามให้มีค่าเท่ากับ

$$\alpha_k(m) = \Pr\{S_k = m \mid R_1^k\} \tag{2.10}$$

$$\beta_k(m) = \frac{\Pr\{R_{k+1}^N \mid S_k = m\}}{\Pr\{R_{k+1}^N \mid R_1^k\}} \tag{2.11}$$

$$\gamma_i(R_k, m', m) = \Pr\{d_k = i, S_k = m, Rk \mid S_{k-1} = m'\} \tag{2.12}$$

โดยที่ตัวแปร $\alpha_k(m)$ และ $\beta_k(m)$ ซึ่งเป็นตัวแปรที่แสดงถึงค่าความน่าจะเป็นของข้อมูลที่ถูกเก็บไว้ในวงจรเข้ารหัสก่อน และ หลังที่จะมีการป้อนข้อมูลดิบเข้ามาภายในวงจรในกรณีต่างๆ ตามลำดับ โดยในการคำนวณนั้น จะมีการนำค่า $\gamma_i(R_k, m', m)$ ซึ่งเป็นตัวแปรที่เกิดจากการนำสัญญาณข้อมูลที่รับได้ ณ เวลาต่างๆมาเปรียบเทียบกับค่าของข้อมูลต่างๆที่จะถูกส่งออกมาจากวงจรเข้ารหัสในกรณีต่างๆ มาใช้สำหรับการคำนวณหาค่า $\alpha_k(m)$ และ $\beta_k(m)$ โดยจะมีลักษณะการคำนวณแบบย้อนกลับ (Recursive) ดังสมการ

$$\alpha_k(m) = \frac{\sum_{m'}\sum_{i=0}^{1}\gamma_i(R_k, m', m)\alpha_{k-1}(m')}{\sum_{m}\sum_{m'}\sum_{i=0}^{1}\gamma_i(R_k, m', m)\alpha_{k-1}(m')} \tag{2.13}$$

$$\beta_k(m) = \frac{\sum_{m'}\sum_{i=0}^{1}\gamma_i(R_k, m', m)\beta_{k+1}(m')}{\sum_{m}\sum_{m'}\sum_{i=0}^{1}\gamma_i(R_k, m', m)\beta_{k+1}(m')} \tag{2.14}$$

โดยในการคำนวณหาค่า $\alpha_k(m)$ และ $\beta_k(m)$ ในสมการที่ 2.14 นั้น จะมีการคำนวณหาค่าที่อยู่ในรูปของการ Normalize เพื่อให้ค่าที่ได้จากการคำนวณนั้นมีค่าอยู่ในช่วงที่ใช้งานเท่านั้น โดยในการคำนวณหาค่า $\alpha_k(m)$ และ $\beta_k(m)$ ในกรณีต่างๆนั้น จะต้องมีการกำหนดค่าเริ่มต้นในการคำนวณให้กับระบบก่อนที่จะมีการทำงาน ซึ่งจะมีรูปแบบในการกำหนดค่า $\alpha_k(m)$ เท่ากับ

$$\begin{aligned} \alpha_0(0) &= 1 \\ \alpha_0(m) &= 0 \quad \forall m \neq 0 \end{aligned} \tag{2.15}$$

และสำหรับค่าเริ่มต้นของ $\beta_k(m)$ นั้น จะมีรูปแบบในการตั้งค่าเริ่มต้นอยู่ 2 รูปแบบด้วยกันขึ้นอยู่กับรูปแบบของการทำงานของวงจรเข้ารหัส โดยสำหรับกรณีที่ไม่มีการทำให้ข้อมูลที่อยู่ในวงจรเข้ารหัสกลับสู่สภาวะที่มีค่าของข้อมูลเป็น 0 ทั้งหมดเหมือนสภาวะเริ่มต้นนั้น จะมีการคำนวณโดยกำหนดค่าเริ่มต้นของ $\beta_k(m)$ ให้มีค่าเท่ากับ

$$\beta_N(m) = \frac{1}{N} \quad \forall m \tag{2.16}$$

และสำหรับกรณีที่มีการส่งข้อมูลให้กับวงจรเข้ารหัสหลังจากข้อมูลดิบทั้งหมดผ่านการเข้ารหัสแล้ว เพื่อทำให้สภาวะของข้อมูลในวงจรกลับสู่สภาวะที่มีค่าเป็น 0 ทั้งหมดเหมือนสภาวะเริ่มต้น จะมีการคำนวณโดยกำหนดค่าเริ่มต้นของ $\beta_k(m)$ ให้มีค่าเท่ากับ

$$\beta_N(N) = 1 \quad \forall m \tag{2.17}$$

สำหรับการคำนวณหาค่า Branch Metrics, $\gamma_i(R_k, m', m)$ นั้น จะมีการคำนวณที่เกิดจากการพิจารณาค่าความน่าจะเป็นของข้อมูลซึ่งสามารถแบ่งการคำนวณออกเป็น 3 ส่วนด้วยกัน และเมื่อพิจารณาถึงลักษณะในการคำนวณค่า $\gamma_i(R_k, m', m)$ แล้ว จะมีลักษณะในการคำนวณดังสมการ

$$\gamma_i(R_k, m', m) = p(R_k / d_k = i, S_k = m, S_{k-1} = m').q(d_k = i / S_k = m, S_{k-1} = m').\pi(S_k = m \mid S_{k-1} = m') \tag{2.18}$$

โดยในส่วนของการคำนวณหาค่าความน่าจะเป็น $p(R_k / d_k = i, S_k = m, S_{k-1} = m')$ นั้น จะเป็นการนำสัญญาณข้อมูลที่รับเข้ามาได้ (R_k) ณ ช่วงเวลานั้น ซึ่งจะประกอบไปด้วยข้อมูล x_k และ y_k มาทำการเปรียบเทียบกับข้อมูลที่มีโอกาสเกิดขึ้นในกรณีต่างๆว่ามีความเป็นไปได้เท่าใด โดยสำหรับกรณีที่มีการใช้รูปแบบในการเข้ารหัสเป็นแบบ Systematic จะได้ว่าข้อมูล x_k นั้นจะมีค่าที่เท่ากับข้อมูลดิบ ซึ่งจะไม่ขึ้นอยู่กับการทำงานของวงจรเข้ารหัสดังนั้น ในส่วนของการคำนวณค่าความน่าจะเป็น $p(R_k / d_k = i, S_k = m, S_{k-1} = m')$ นั้นสามารถแยกการคำนวณในส่วนของข้อมูล x_k และ y_k ออกจากกันได้ โดยจะมีค่าเท่ากับ

$$p(R_k / d_k = i, S_k = m, S_{k-1} = m') = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2\sigma^2}\left((x_k - u_k(i,m,m'))^2 + (y_k - v_k(i,m,m')^2\right)} \quad (2.19)$$

เมื่อ u_k และ v_k เป็นค่าของข้อมูลที่จะได้จากการทำงานของภาคเข้ารหัสเมื่อมีการป้อนข้อมูลดิบมีค่าเท่ากับ i และข้อมูลที่อยู่ในวงจรเข้ารหัสก่อนและหลังจากที่มีการป้อนข้อมูลเข้ามามีค่าเป็น m' และ m ตามลำดับ และสำหรับค่าความน่าจะเป็น $q(d_k = i \mid S_k = m, S_{k-1} = m')$ จะเป็นค่าที่เกิดจากการวิเคราะห์ความเป็นไปได้ของข้อมูลที่อยู่ในวงจรเข้ารหัสหลังและก่อนที่มีการป้อนข้อมูลดิบเข้ามาภายในวงจรและค่าของข้อมูลที่ได้จากการเข้ารหัส ในช่วงเวลาการทำงานต่างๆว่ามีความเป็นไปได้ที่จะมีค่าเป็นไปตามนั้นมากน้อยเพียงใด โดยในกรณีของวงจรเข้ารหัสแบบ Convolution Codes นั้น เนื่องจากรูปแบบในการทำงานที่มีรูปแบบที่แน่นอน ดังนั้นค่า $q(d_k = i / S_k = m, S_{k-1} = m')$ ที่ได้นั้น จะมีค่าแค่เพียง 0 หรือ 1 เท่านั้น และสำหรับค่าความน่าจะเป็น $\pi(S_k = m \mid S_{k-1} = m')$ นั้น จะเป็นค่าที่เกิดจากการคาดคะเนค่าของข้อมูลดิบที่จะถูกป้อนเข้ามาภายในวงจรเข้ารหัส ณ เวลานั้นๆ โดยทั่วไปจะมีค่าเท่ากับ $P(d_k = 1) = P(d_k = 0) = 1/2$

ดังนั้นสมการที่ใช้สำหรับคำนวณหาค่า LAPP นั้นจะมีค่าเท่ากับ

$$L(d_k) = \log \frac{\sum_{m}\sum_{m'} \gamma_1(R_k, m', m)\alpha_{k-1}(m')\beta_k(m)}{\sum_{m}\sum_{m'} \gamma_0(R_k, m', m)\alpha_{k-1}(m')\beta_k(m)} \quad (2.20)$$

ดังนั้น สำหรับขั้นตอนที่ใช้สำหรับการถอดรหัสข้อมูลโดยใช้วิธีการ MAP นั้น จะมีขั้นตอนในการทำงานดังต่อไปนี้

1. นำข้อมูลที่รับเข้ามาได้มาทำการคำนวณหาค่า Branch Metrics, $\gamma_k(R_k, m', m)$ ในกรณีต่างๆ โดยใช้สมการที่ 2.18
2. นำค่า Branch Metrics ที่ได้จากการคำนวณมาใช้ในการคำนวณหาค่า Forward State Matric , $\alpha_k(m)$ โดยใช้สมการที่ 2.13
3. นำค่า Branch Metrics ที่ได้จากการคำนวณมาใช้ในการคำนวณหาค่า Reverse State Matric , $\beta_k(m')$ โดยใช้สมการที่ 2.14
4. ทำการคำนวณหาค่า LAPP ของข้อมูลบิตที่ k โดยใช้สมการที่ 2.20

5. ทำการตัดสินใจค่าของข้อมูลดิบที่ได้จากการถอดรหัสโดยการพิจารณาจากค่า LAPP ที่ได้จากการคำนวณ โดยที่ถ้าค่า LAPP มีค่ามากกว่า 0 จะมีการตัดสินใจให้ข้อมูลดิบ d_t ที่ได้จากการถอดรหัส ณ เวลานั้นมีค่าเป็น +1 แต่ถ้าหากมีค่าน้อยกว่า 0 แล้ว จะตัดสินใจให้ข้อมูลดิบ d_t มีค่าเป็น -1

2.24 การถอดรหัสแบบ Viterbi Decoder

การถอดรหัสแบบ Viterbi จะมีลักษณะการทำงานเป็นแบบ *Maximum-likelihood* Decoding Algorithm ซึ่งผลลัพธ์ที่ได้จากการทำงานนั้น จะได้เส้นทางเพียงหนึ่งเส้นทางจากเส้นทางทั้งหมดใน Trellis Diagram ที่มีลักษณะที่เหมือนกับข้อมูลที่รับได้มากที่สุด

2.25 รูปแบบสำหรับการถอดรหัสแบบ Viterbi Decoder

สำหรับการถอดรหัสแบบ Viterbi Decoding นั้น จะมีรูปแบบสำหรับการถอดรหัสที่ใช้งานอยู่ 2 ลักษณะด้วยกัน ได้แก่ แบบ Hard Decision และ Soft Decision

2.25.1 Hard Decision

สำหรับการทำงานของวงจรถอดรหัสที่ใช้กระบวนการตัดสินใจแบบ Hard Decision นั้น จะเป็นการพิจารณาข้อมูลที่รับเข้ามา โดยการพิจารณาว่าข้อมูลที่รับเข้ามาในแต่บิตนั้น มีค่าของข้อมูลเป็น 0 หรือ 1 เท่านั้น

2.25.2 Soft Decision

การถอดรหัสที่มีการใช้กระบวนการตัดสินใจแบบ Soft Decision จะเป็นการพิจารณาถึงข้อมูลที่รับเข้ามาได้โดยการทำการตัดสินใจระดับของข้อมูลที่รับเข้ามาได้โดยการแบ่งระดับของสัญญาณที่ใช้ในการ คำนวณหาค่า Metric ที่มากกว่า 2 ระดับ ซึ่งผลลัพธ์ที่ได้นั้น จะได้ข้อมูลรายละเอียดของข้อมูลที่ส่งมาที่มากกว่ากรณีของ Hard Decision ซึ่งข้อมูลที่ได้จากการตัดสินใจ (Soft-Output) นั้น จะถูกนำมาใช้ในการคำนวณหาค่า Metrics เพื่อเปรียบเทียบข้อมูลที่รับเข้ามา ณ เวลานั้นๆ กับข้อมูลที่อยู่ในเส้นทางต่างๆณ เวลานั้น ซึ่งจะมีรูปแบบที่ใช้ในการคำนวณที่แตกต่างกันไป

2.26 Viterbi Algorithm

โดยในการทำงานต่างๆนั้นจะต้องมีการคำนวณหาความแตกต่างระหว่างข้อมูลที่รับเข้ามาและค่าที่อยู่ในเส้นทางต่างๆ เพื่อใช้ในกระบวนการตัดสินใจ โดยกระบวนการที่ใช้ในการทำการหาเส้นทางที่ดีที่สุดนั้น จะใช้วิธีการทำงานที่มีชื่อว่า Viterbi Algorithm ซึ่งจะเป็นกระบวนการที่ใช้ในการค้นหาเส้นทางที่อยู่ใน Trellis Diagram ที่มีลักษณะที่ใกล้เคียงกับข้อมูลที่รับได้มากที่สุด เพื่อที่จะนำข้อมูลในเส้นทางนั้นมาคำนวณหาค่าของข้อมูลที่ถูกส่งมา โดยที่ในกระบวนการค้นหาเส้นทางที่เหมาะสมที่สุดโดยใช้ Viterbi Algorithm นั้น จะมีขั้นตอนในการทำงานดังต่อไปนี้

1) พิจารณาแบ่งข้อมูลที่รับเข้ามาออกเป็นข้อมูลย่อยๆจำนวน m ช่วง ซึ่งแต่ละช่วงนั้นมีขนาดของข้อมูลเท่ากับ n_0 บิต

2) ทำการวาด Trellis Diagram ที่มีจำนวน State ในการทำงานเท่ากับ m State โดยจมีการพิจารณาเฉพาะเส้นทางที่มีความเป็นไปได้ว่าจะถูกส่งมาเท่านั้น โดยสำหรับที่ State ของ Trellis Diagram ตั้งแต่ L-1 ขึ้นไปนั้น ให้วาดเฉพาะเส้นทางที่จะพุ่งเข้าหาสภาวะของวงจรที่มีข้อมูลเป็น 0 ทั้งหมด

3) กำหนดค่าตัวแปร l = 1 และทำการกำหนดค่าเริ่มต้นของตัวแปร Metric ในสภาวะเริ่มต้นที่มีข้อมูลเป็น 0 ทั้งหมด ให้มีค่าของ Metric เท่ากับ 0

4) ทำการคำนวณหาค่าความแตกต่างของข้อมูล (Distance) ระหว่างข้อมูลที่รับได้ชุดที่ l กับข้อมูลในเส้นทางในการเปลี่ยนแปลงสถานะใน Trellis Diagram จาก State ที่ l ไปเป็น l+1

5) นำค่าที่คำนวณได้นั้นไปบวกกับค่า Metric สะสมของ State l เพื่อคำนวณหาค่าของ Metric สะสมใน State ที่ l+1 เพื่อใช้ในการตัดสินใจเลือกเส้นทางที่เหมาะสมที่สุด ในการเปลี่ยนแปลงข้อมูลไปยัง State นั้นๆ โดยในแต่ละ State นั้น จะมีจำนวนเส้นทางทั้งหมดจำนวน $2k_0$ เส้นทางที่จะพุ่งเข้า State เดียวกัน

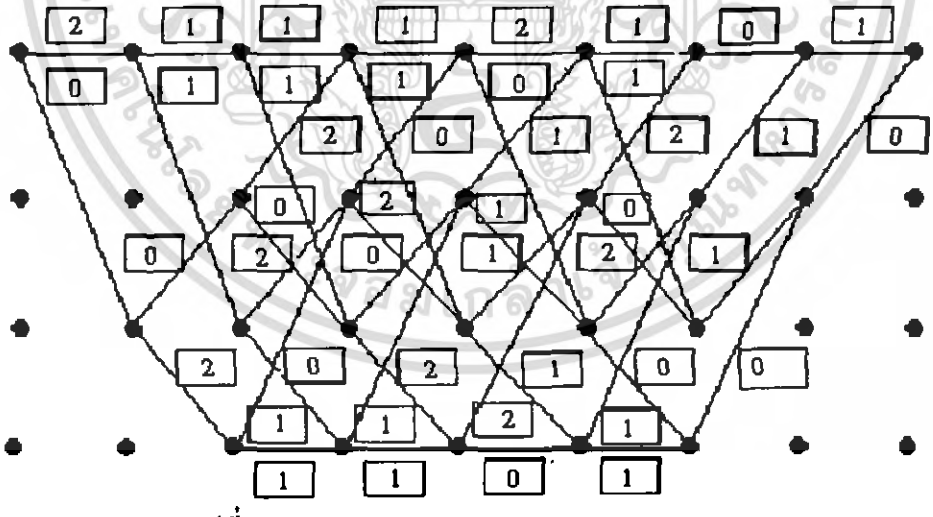
6) พิจารณา ณ ตำแหน่งใน State ที่ l+1 ในแต่ละ State นั้น ทำการเลือกเส้นทางที่มีค่า Metric สะสมที่มีค่าน้อยที่สุดที่พุ่งเข้าหาในแต่ละ State โดยที่เส้นทางที่ถูกเลือกนั้น จะถูกเรียกว่า "Survivor" ซึ่งจะเป็นเส้นทางที่ถูกเก็บไว้ทำการคำนวณใน State ต่อไป และสำหรับเส้นทางอื่นๆที่ไม่ได้ถูกเลือกนั้น จะถูกเรียกว่า "Forgetting" โดยจะถูกลบทิ้งออกไปจากระบบการตัดสินใจ

7) ถ้าหากว่า l นั้นมีค่าเท่ากับ m แล้วให้ทำงานในขั้นตอนต่อไปได้ แต่ถ้ายังมีค่าน้อยกว่าจะมีต้องมีการเพิ่มค่า l ขึ้นอีก 1 จากนั้นจึงกลับไปทำงานที่ขั้นตอนที่ 4 ใหม่

8) เริ่มต้นพิจารณา ณ State ที่ m+1 ที่มีสภาวะของข้อมูลสถานะเป็น 0 ทั้งหมด ทำการเลือกเส้นทางที่เป็น "Survival" ซึ่งเป็นเส้นทางที่ถูกเลือกที่เหลืออยู่ย้อนกลับไปจนกระทั่งถึงสภาวะเริ่มต้นของการทำงานที่มีสภาวะในการทำงานเป็น 0 ทั้งหมด ซึ่งเส้นทางที่ได้นั้น จะเป็นเส้นทางที่มี

ลักษณะที่ใกล้เคียงกับข้อมูลที่รับเข้ามามากที่สุด ซึ่งจะถูกนำไปใช้ในการคำนวณหาข้อมูลข่าวสารที่ถูกส่งมา โดยข้อมูลข่าวสารที่จะถูกส่งออกไปจากภาคถอดรหัสนั้น จะเป็นการส่งข้อมูลทั้งหมดที่อยู่ในเส้นทางส่งออกไปยกเว้นยกเว้นข้อมูล 0 จำนวน $k_0(L-1)$ บิต ที่อยู่ท้ายสุดนั้น จะถูกตัดทิ้งไป

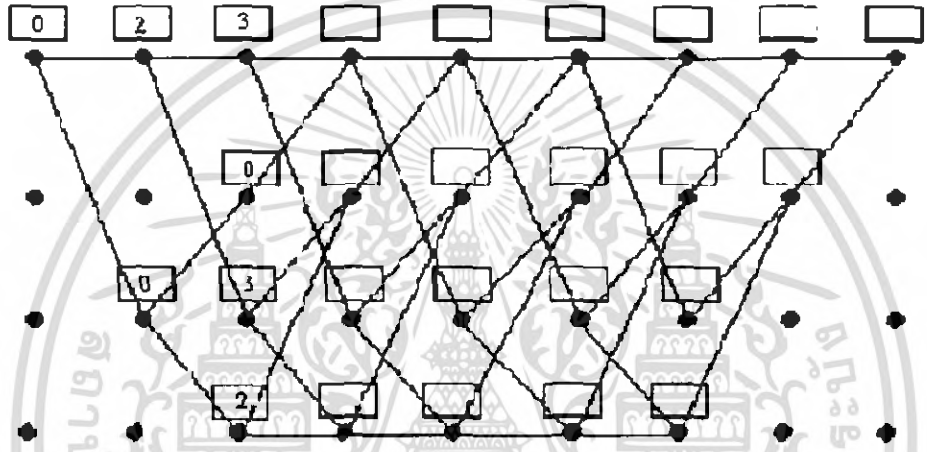
ตัวอย่างเช่น ในกรณีของวงจรถอดรหัสสำหรับข้อมูลที่ถูกเข้ารหัสด้วยวงจรในรูปที่ 2.22 ซึ่งมีการป้อนข้อมูลดิบจำนวน 6 บิตมีค่าเท่ากับ 1 0 1 0 1 1 ซึ่งเมื่อทำการเข้ารหัสแล้วจะได้ Codeword มีค่าเท่ากับ 1 1 0 1 1 0 0 1 1 0 1 0 0 0 0 1 จากนั้นพิจารณาในกรณีที่ข้อมูลของ Codeword ที่รับได้ที่ปลายทางนั้นเกิดความผิดพลาดในการตีความขึ้นจำนวน 1 บิต จึงทำให้ข้อมูลที่รับได้มีค่าเท่ากับ 1 1 0 1 1 0 0 1 1 1 1 0 0 0 0 1 ดังนั้นในการถอดรหัสโดยใช้วิธีการ Viterbi Decoding นั้นจะทำงานโดยเริ่มต้นจากการนำข้อมูลที่ได้นั้นมาทำการแบ่งออกเป็นชุดๆที่มีขนาดเท่ากับจำนวนของ Codeword ที่วงจรเข้ารหัสส่งออกมาในแต่ละช่วงเวลาซึ่งในกรณีนี้มีค่าเท่ากับ 2 ดังนั้นข้อมูลต่างๆจะถูกแบ่งออกเป็นชุดดังนี้ 11 01 10 01 11 10 00 01 ซึ่งในการทำงานนั้นจะมีการพิจารณาข้อมูลทีละชุดเพื่อนำมาเปรียบเทียบกับข้อมูลที่อยู่ในเส้นทางต่างๆใน Trellis Diagram โดยการทำงานในช่วงแรกนั้นจะเป็นการหาความแตกต่างระหว่างข้อมูลที่รับได้และข้อมูลที่อยู่ในเส้นทางต่างๆ ซึ่งมีลักษณะการคำนวณดังรูปที่ 2.22



รูปที่ 2.22 การถอดรหัสแบบ Viterbi Decoding (1)

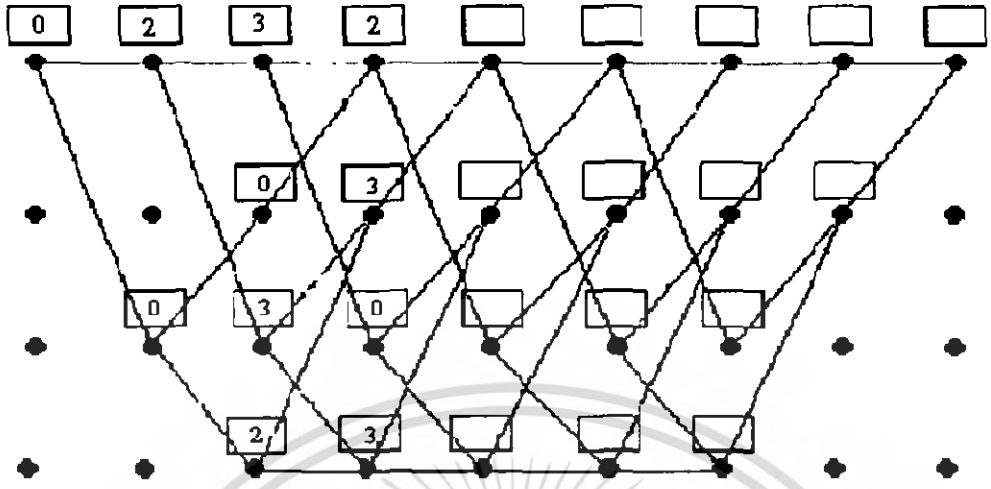
ซึ่งจะเป็นการนำข้อมูลที่รับได้ในแต่ละช่วงเวลามาทำการเปรียบเทียบกับข้อมูลที่อยู่ในเส้นทางต่างๆ โดยในช่วงเวลาที่ 1 นั้น จะเป็นการนำข้อมูล 11 ที่รับได้มาทำการเปรียบเทียบเพื่อหาจำนวนบิตของข้อมูลที่มีความแตกต่างกัน โดยเมื่อเปรียบเทียบกับเส้นทางที่ลากจาก Node เริ่มต้นที่มีข้อมูล

เป็น 0 ทั้งหมดมายัง Node ที่มีข้อมูลเป็น 00 ในช่วงเวลาถัดไปซึ่งมีข้อมูลในเส้นทางเป็น 0/00 ซึ่งจะพบว่าจะมีค่า Codeword ที่แตกต่างกันจำนวน 2 บิต และเมื่อเปรียบเทียบกับเส้นทางที่ลากไปยัง Node ที่มีข้อมูลเป็น 10 จะพบว่ามีความแตกต่างเท่ากับ 0 บิต และใน ช่วงเวลาที่ 2 จะเป็นการนำข้อมูล 01 ที่รับได้ ณ เวลานั้นมาทำการเปรียบเทียบกับเส้นทางต่างๆทั้งหมด ณ เวลานั้นๆ และจะมีการคำนวณเช่นนี้ไปเรื่อยๆจนครบทุกเส้นทางจากนั้นในการทำงานขั้นตอนต่อไปจะเป็นการคำนวณหาค่า Metric สะสมของเส้นทาง โดยสำหรับ 2 ช่วงเวลาแรกนั้นจะมีการทำงานดังรูปที่ 2.23



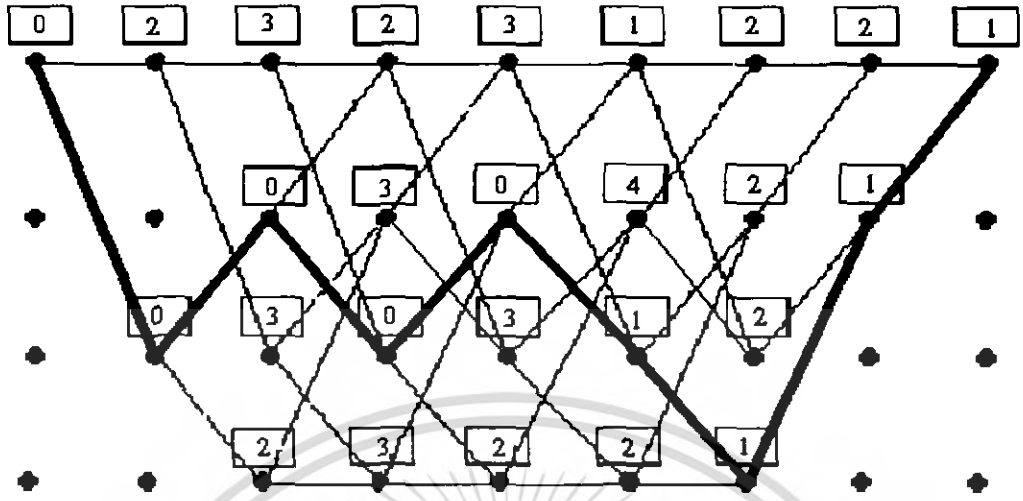
รูปที่ 2.23 การถอดรหัสแบบ Viterbi Decoding (2)

โดยในการทำงานนั้นจะเริ่มต้นที่ Node 00 ที่อยู่ซ้ายบนสุดของรูป ซึ่งกำหนดให้มีค่า Metric เป็น 0 จากนั้นจึงพิจารณา Node ถัดไปที่เชื่อมต่อกับ Node 00 ซึ่งมี 2 Node ด้วยกัน โดยจะทำการนำ Metric ที่อยู่ในเส้นทางนั้นมาบวกกับ Metric สะสมที่ถูกเก็บไว้ยัง Node ที่พิจารณา ซึ่ง ณ เวลานี้คือ Node 00 ที่อยู่ซ้ายบนสุด และผลลัพธ์ที่ได้นั้นจะถูกเก็บไว้ใน Node ที่อยู่ปลายทางของเส้นทางนั้นๆ ซึ่งจะมีการทำงานเช่นนี้ไปเรื่อยๆจนกระทั่งมีการพบว่า มีเส้นทางมากกว่า 1 เส้นทางที่พุ่งเข้าหา Node เดียวกันดังเช่นในช่วงเวลาที่ 3 จนถึงช่วงเวลาที่ 8 ซึ่งเมื่อเกิดเหตุการณ์นี้ขึ้นจะต้องมีกระบวนการตัดสินใจเพื่อเลือกเส้นทางที่ใช้ในการคำนวณเฉพาะ เส้นทางที่มีค่า Metric สะสมน้อยที่สุดเท่านั้นมาพิจารณา โดยในช่วงเวลาที่ 3 นั้นจะมีการคำนวณดังรูปที่ 2.24



รูปที่ 2.24 การถอดรหัสแบบ Viterbi Decoding (3)

โดยสำหรับการพิจารณาค่า Metric สะสมที่ Node 00 ในช่วงเวลาที่ 3 นั้น จะเป็นการตัดสินใจเลือกเส้นทางระหว่างเส้นทางที่มาจาก Node 00 และ Node 01 ของช่วงเวลาที่ 2 ซึ่งจะทำการเลือกเส้นทางที่มีค่า Metric สะสมที่น้อยที่สุดเท่านั้น โดยสำหรับเส้นทางที่มาจาก Node 00 จะมีค่า Metric สะสมเท่ากับ 3+1 เท่ากับ 4 และเส้นทางที่มาจาก Node 01 มีค่าเท่ากับ 0+2 เท่ากับ 2 ดังนั้นจึงเลือกเส้นทางที่มาจาก Node 01 มาใช้ในการคำนวณซึ่งจะมีการทำงานในลักษณะนี้เรื่อยๆ จนกระทั่งคำนวณ Metric สะสมครบทุกช่วงเวลา จากนั้นจึงนำข้อมูลที่ได้นั้นมาใช้ในการค้นหาเส้นทาง โดยการเริ่มต้นที่ Node 00 ที่อยู่ขวาสุดของรูป จากนั้นจึงทำการมองย้อนกลับใน Node ที่อยู่ซ้ายมือที่มีเส้นทางต่อกับ Node 00 โดยจะทำการเลือกเส้นทางที่ต่อกับ Node ที่มีค่า Metric สะสมที่มีค่าน้อยที่สุดเพื่อเลือกเป็นเส้นทางที่ใช้งานจากนั้นจึงย้ายจุดที่พิจารณาไปยัง Node ถัดไปทางซ้ายมือที่อยู่ปลายทางของเส้นทางที่เลือกไว้ ซึ่งจะมีกระบวนการตัดสินใจเช่นเดิมซ้ำอีกครั้ง โดยจะมีการทำงานเช่นนี้ไปเรื่อยๆจนมีการค้นหาเส้นทางครบทุกช่วงเวลา ซึ่งผลลัพธ์ที่ได้จากการทำงาน จะมีลักษณะดังรูปที่ 2.25



รูปที่ 2.25 การถอดรหัสแบบ Viterbi Decoding (4)

หลังจากนั้น จะเป็นการนำข้อมูลที่อยู่ในเส้นทางที่ถูกเลือกมาใช้ในการหาข้อมูลดิบที่ส่งมา โดยการพิจารณาถึงข้อมูลที่อยู่ในเส้นทางส่วนต่างๆ ซึ่งจากรูปนั้นจะได้ว่าข้อมูลดิบที่ถอดรหัสได้ จะมีค่าเท่ากับ 1 0 1 0 1 1 0 0 โดยใน 2 บิตสุดท้ายจะไม่ถูกนำมาพิจารณาเนื่องจากเป็นข้อมูลที่ถูกป้อนเข้ามาเพื่อใช้ในการคำนวณ ดังนั้นข้อมูลดิบที่ได้จากการถอดรหัสจะมีค่าเท่ากับ 1 0 1 0 1 1

2.27 ตัวแปร Extrinsic Information

เมื่อทำการพิจารณาสมการ ที่ใช้สำหรับคำนวณหาค่า LAPP นั้น จะพบว่าในกรณีที่มีการเข้ารหัสเป็นแบบแบบ Nonsystematic แล้ว ข้อมูลในส่วนที่มีค่าเหมือนกับข้อมูลดิบนั้น เนื่องจากเป็นตัวแปรที่ไม่มีความสัมพันธ์กับข้อมูลต่างๆที่อยู่ในวงจรเข้ารหัส ดังนั้นจึงสามารถที่จะแยกการคำนวณในส่วนของข้อมูลที่เหมือนกับข้อมูลดิบและข้อมูลส่วนที่มีการเข้ารหัสออกจากกันได้ ดังนั้นสมการที่ใช้ในการคำนวณหาค่า LAPP จะเท่ากับ

$$L(u_k) = \log\frac{p(x_k \mid d_k = 1)}{p(x_k \mid d_k = 0)} + \log\frac{\sum_m\sum_{m'}\gamma_1(y_k, m', m)\alpha_{k-1}(m')\beta_k(m)}{\sum_m\sum_{m'}\gamma_0(y_k, m', m)\alpha_{k-1}(m')\beta_k(m)} \tag{2.21}$$

$$L(u_k) = \log\frac{p(x_k \mid d_k = 1)}{p(x_k \mid d_k = 0)} + L_e \tag{2.22}$$

โดยในส่วนของการคำนวณส่วนแรกในสมการนั้น จะเป็นการนำข้อมูลที่รับได้ในส่วนของข้อมูลที่เหมือนกับข้อมูลดิบ x_k มาทำการคำนวณ และในส่วนของการคำนวณส่วนหลังของสมการนั้น (W_k) จะเป็นข้อมูลส่วนที่ได้จากการนำข้อมูลที่เข้ารหัส y_k มาใช้ในการคำนวณ โดยในกรณีทั่วๆไปนั้นจะมีค่าที่มีเครื่องหมายเหมือนกับข้อมูลดิบ d_k ดังนั้นจึงสามารถนำข้อมูลในส่วนนี้มาใช้สำหรับเพิ่มความถูกต้องในการคำนวณหาค่า LAPP ให้มีความถูกต้องมากขึ้นได้ โดยการส่งข้อมูลในส่วนนี้ป้อนกลับไปใช้ในการเข้ารหัสข้อมูลชุดเดิมใหม่ได้ ซึ่งจะทำให้ค่า LAPP ที่ได้จากการคำนวณนั้นมีความถูกต้องมากยิ่งขึ้นได้ ซึ่งจะมีการเรียกค่าของข้อมูลในส่วนนี้ว่าเป็นค่า "Extrinsic Information" ซึ่งจะสามารถคำนวณได้จากการคำนวณหาค่า LAPP ของวงจรถอดรหัสโดยไม่พิจารณาถึงข้อมูลในส่วนที่มีค่าเหมือนกับข้อมูลดิบ

$$L_e = L(d_k)|_{x_k=0} = \log \frac{\sum_{m}\sum_{m'} \gamma_1(y_k, m', m)\alpha_{k-1}(m')\beta_k(m)}{\sum_{m}\sum_{m'} \gamma_0(y_k, m', m)\alpha_{k-1}(m')\beta_k(m)} \tag{2.23}$$

โดยที่เมื่อพิจารณาถึงรูปแบบในการนำค่า Extrinsic Information มาใช้ในการปรับปรุง การทำงานต่างๆของวงจรถอดรหัสนั้น จะพบว่าค่า Extrinsic Information ที่ได้จากการถอดรหัสในแต่ละครั้งนั้น สามารถที่จะนำมาใช้สำหรับเปลี่ยนแปลงค่าความน่าจะเป็น $\pi(S_k = m \mid S_{k-1} = m')$ ที่ใช้สำหรับการคำนวณของวงจรเข้ารหัสที่จะทำงานถัดไป ที่เป็นค่าความน่าจะเป็นที่เกิดจากการคาดคะเนค่าของข้อมูลที่ได้จากการถอดรหัส d_k ณ เวลาต่างๆ ในสมการที่ (2.21) ซึ่งจะถูกนำมาใช้ในการคำนวณหาค่า Branch Metrics, $\gamma_k(R_k, m', m)$ ให้มีค่าที่แม่นยำมากยิ่งขึ้น ซึ่งผลลัพธ์ที่ได้นั้น จะเป็นการทำให้การคำนวณต่างๆ ในการถอดรหัสข้อมูลนั้นมีความถูกต้องมากขึ้นกว่าการถอดรหัสในครั้งก่อน

2.28 วิธีการถอดรหัสแบบ Iterative Decoding

ดังนั้นสำหรับการทำงานของวงจรถอดรหัสในรูปแบบของการถอดรหัสแบบ Iterative Decoding โดยการนำหลักการของการถอดรหัสแบบ MAP มาทำการประยุกต์ใช้งาน จะได้ว่าค่า LAPP ที่ได้จากการทำงานของวงจรถอดรหัส DEC1 และ 2 นั้น ข้อมูลที่ได้จากการถอดรหัสนั้นจะประกอบด้วยการคำนวณ 3 ส่วนด้วยกันได้แก่ ข้อมูลในส่วนของ APP ที่ได้จากการทำงานของภาคของรหัสในครั้งก่อน ส่วนของข้อมูลที่ได้จากการคำนวณข้อมูล x_k ที่รับได้ และส่วนของ Extrinsic

Information ที่จะถูกคำนวณขึ้นในระหว่างที่มีการถอดรหัสและจะเป็นข้อมูลส่วนที่จะถูกส่งออกไปจากวงจรถอดรหัสเพื่อใช้เป็นค่า APP ในการถอดรหัสของวงจรถอดรหัสวงจรต่อไป โดยสำหรับกรณีที่มีการพิจารณาถึงการใช้งานการเข้ารหัสแบบ Turbo Codes ภายใต้ระบบสื่อสารที่มีสัญญาณรบกวนเป็นแบบเกาส์ (Gaussian) นั้นจะได้ว่า

$$L(d_k) = L_a + L_c x_k + L_e \tag{2.24}$$

โดยในส่วนของการคำนวณค่า L_a ในเทอมแรกนั้น จะเป็นส่วนของการคำนวณเกี่ยวกับค่า a Priori Probability ที่ได้จากกระบวนการถอดรหัสครั้งก่อนที่ถูกส่งมา และ ในเทอมที่สองนั้นจะเป็นการคำนวณเกี่ยวกับค่า x_k ที่รับได้ โดยจะมีการนำค่าที่รับได้นั้นมาคูณด้วยค่าความเชื่อมั่นของสัญญาณ (Reliability Value), L_c มีค่าเท่ากับ $2/\sigma$ และในเทอมสุดท้ายนั้นจะเป็นการคำนวณในส่วนของค่า Extrinsic Information ซึ่งจะได้มาจากการนำข้อมูลที่เข้ามาในวงจรเข้ารหัสมาทำการคำนวณ และจะเป็นค่าที่จะถูกส่งออกจากวงจรถอดรหัสเพื่อใช้เป็นค่า a Priori Probability ของวงจรถัดไปเพื่อใช้สำหรับกระบวนการถอดรหัส ดังนั้นเมื่อพิจารณาถึงการทำงานของวงจรถอดรหัสสำหรับ Turbo Codes ในรูปที่ 2.6 นั้นจะได้ว่าลักษณะการทำงานต่างๆของวงจรถอดรหัสนั้น จะมีลักษณะในการทำงานเพื่อทำการถอดรหัสข้อมูลในแต่ละชุดนั้น จะมีลักษณะการทำงานดังต่อไปนี้

1. ทำการ Depuncture ข้อมูลที่รับได้เพื่อกระจายข้อมูลไปยังวงจรถอดรหัสต่างๆ
2. เมื่อข้อมูลถูกส่งมาเข้ายังภาคถอดรหัส DEC1 จะมีการคำนวณหาค่า LAPP โดยใช้ข้อมูลในส่วนของ $L_a, L_c x_k$ และ $L_c y_k^{(1)}$ โดยใช้วิธีการ MAP โดยจะมีขั้นตอนในการทำงานดังนี้

ทำการคำนวณหาค่า Branch Metrics, $\gamma_k(R_k, m', m)$ โดยใช้สมการที่ 2.18

นำค่า Branch Metrics ที่ได้จากการคำนวณมาใช้ในการคำนวณหาค่า Forward State Matric, $\alpha_k(m)$ โดยใช้สมการที่ 2.13

นำค่า Branch Metrics ที่ได้จากการคำนวณมาใช้ในการคำนวณหาค่า Reverse State Matric, $\beta_k(m')$ โดยใช้สมการที่ 2.14

ทำการคำนวณหาค่า LAPP ของข้อมูลบิตที่ k โดยใช้สมการที่ 2.20

3. ทำการคำนวณหาค่า Extrinsic Information , L_e ที่จะถูกส่งไปยังวงจรถอดรหัส DEC2 เพื่อใช้เป็นค่า a Priori Probability ในการทำงานต่อไป โดยพิจารณาจากสมการที่ 2.24

ดังนั้นค่า Extrinsic Information ที่ได้ที่อยู่ในรูปแบบของข้อมูล Logarithm นั้น จะคำนวณได้จากการนำค่า LAPP ที่ได้จากการถอดรหัสมาทำการลบด้วยค่า L_a และ $L_c.x_k$

4. เมื่อวงจรถอดรหัส DEC2 ได้รับข้อมูลในส่วนของ a Priori Probability ที่ได้มาจากค่า Extrinsic ของวงจรถอดรหัส DEC1 ซึ่งถูกผ่านการเปลี่ยนแปลงรูปแบบของข้อมูลโดยภาค Interleave จะมีการนำข้อมูลดังกล่าวมาใช้ในการคำนวณหาค่า LAPP ของวงจรถอดรหัส DEC2 โดยใช้วิธีการ MAP ซึ่งจะมีรูปแบบในการทำงานเช่นเดียวกับภาคถอดรหัส DEC1
5. ถ้าหากว่าการวงรอบการคำนวณยังไม่ครบตามที่กำหนด จะทำการคำนวณหาค่า Extrinsic Information, L_e ที่จะถูกส่งไปยังวงจรถอดรหัส DEC1 เพื่อใช้เป็นค่า a Priori Probability ในการทำงานต่อไป แล้วจึงกลับไปทำงานที่ (2) อีกครั้ง
6 นำค่า LAPP ที่ได้จากการทำงานของวงจรถอดรหัส DEC2 มาทำการตัดสินใจค่าของข้อมูลดิบ d_k ที่จะถูกส่งออกเป็นผลลัพธ์ของภาคถอดรหัสแบบ Turbo Codes
7. จบการทำงานของการถอดรหัส

ซึ่งผลลัพธ์ที่ได้จากการถอดรหัสในรูปแบบของ *Iterative Decoding* นั้น ถ้าหากว่ามีการวนรอบในการถอดรหัสมากขึ้นเท่าใด ก็จะยิ่งทำให้ค่าของตัวแปรต่างๆที่ใช้ในการคำนวณของภาคเข้ารหัสนั้นมีค่าที่แม่นยำมากยิ่งขึ้นตามไปด้วย และจะส่งผลให้ข้อมูลที่ได้จากการถอดรหัสข้อมูลนั้นมีค่าที่มีความถูกต้องมากขึ้นตามไปด้วย แต่เนื่องจากในกระบวนการในการทำงานต่างๆของภาคถอดรหัสที่มีการนำหลักการถอดรหัสแบบ MAP มาประยุกต์ใช้งานนั้น จะมีการคำนวณที่มีความซับซ้อนสูง ดังนั้นจึงได้มีการพัฒนารูปแบบในการคำนวณต่างๆ ให้มีการทำงานที่มีความซับซ้อนน้อยลง ดังเช่นตัวอย่างของวิธีการแบบ Max-Log-Map และ Log-MAP ที่จะมีการบรรยายในหัวข้อถัดไป

2.29 การลดความซับซ้อนในการคำนวณในการถอดรหัสข้อมูล

สำหรับกระบวนการถอดรหัสแบบ Iterative Decoding ที่มีการนำหลักการในการถอดรหัสแบบ MAP มาทำการประยุกต์ใช้งานนั้น จะมีจุดประสงค์ในการทำงานเพื่อทำการคำนวณหาค่า APP เพื่อนำมาใช้สำหรับการตัดสินใจ โดยที่เมื่อพิจารณาถึงรูปแบบในการทำงานต่างๆของวิธีการถอดรหัสแบบ MAP แล้ว จะพบว่าการคำนวณต่างๆที่ใช้ในการทำงานนั้น จะมีลักษณะในการคำนวณที่มีความซับซ้อนในการทำงานมาก ดังนั้นจึงได้มีการพัฒนารูปแบบในการคำนวณต่างๆ ให้มีลักษณะในการคำนวณที่มีความซับซ้อนน้อยลง

วิธีการหนึ่งที่ถูกนำมาใช้ได้แก่การพิจารณาการคำนวณค่าต่างๆอยู่ในรูปของค่า Logarithm ซึ่งผลลัพธ์ที่ได้นั้นจะทำให้การคำนวณต่างๆที่อยู่ในรูปของการคูณนั้น จะเปลี่ยนมาอยู่ในรูปแบบของการบวกแทน ซึ่งจะมีผลทำให้วงจรที่ใช้ในการคำนวณนั้นมีความซับซ้อนที่ลดลง โดยจะสามารถทำให้การคำนวณที่อยู่ในรูปของการคูณนั้น จะเปลี่ยนมาอยู่ในรูปแบบของการบวก นอกจากนั้นจะมีผลทำให้ค่าของข้อมูลที่ได้จากการคำนวณนั้น มีอัตราการเปลี่ยนแปลงที่ลดลงซึ่งจะทำให้สามารถป้องกันการเกิดข้อมูลที่มีค่าเกินช่วงที่พิจารณาได้ (Overflow) แต่ในการคำนวณต่างๆ เมื่อพิจารณาค่าอยู่ในรูปของ Logarithm นั้น จะยังมีการคำนวณที่มีความซับซ้อนอยู่ เนื่องจากจะมีการคำนวณในรูปแบบของการหาค่าจากฟังก์ชัน $\ln(e^{\delta_1} + ... + e^{\delta_{n-1}})$ ซึ่งจะต้องใช้ การทำงานที่มีความซับซ้อนในการหาค่าของข้อมูล ในหัวข้อต่อไปนี้จะกล่าวถึงรูปแบบในการคำนวณแบบ Max-Log-Map และ Log-MAP ซึ่งเป็นวิธีการที่ใช้ในการลดความซับซ้อนในการคำนวณของภาคถอดรหัสที่มีลักษณะในการคำนวณที่อยู่ในรูปของฟังก์ชันดังกล่าวให้น้อยลง โดยจะมีรายละเอียดในการทำงานดังนี้

2.30 วิธีการ MAX-Log-MAP

เมื่อพิจารณาถึงลักษณะในการคำนวณของภาคถอดรหัสเพื่อคำนวณหาค่า LAPP และ Extrinsic Information จะพบว่าในการคำนวณของภาคถอดรหัสที่มีรูปแบบในการทำงานที่เป็นการพัฒนามาจากวิธีการถอดรหัสแบบ MAP จะพบว่าในการคำนวณต่างๆของภาคถอดรหัสนั้น จะมีการเริ่มต้นจากการคำนวณหาค่า Branch Metrics , $\gamma_k(R_k, m', m)$ เพื่อนำมาใช้ในการคำนวณหาค่าของตัวแปรต่างๆ โดยเมื่อพิจารณาถึงลักษณะในการคำนวณหาค่า Branch Metrics ในสมการที่ 2.18 นั้นจะพบว่าลักษณะของสมการที่ใช้สำหรับการคำนวณหาค่า Branch Metrics จะมีลักษณะของสมการอยู่ในรูปของ $K_1.e^{K_2}$ โดยที่ K_1 และ K_2 นั้นค่าต่างๆที่ได้จากการคำนวณ และในการนำค่า Branch Metrics มาใช้ในการคำนวณหาค่าของ Forward และ Reverse State Metrics $(\alpha_k(m), \beta_k(m))$นั้น จะมีลักษณะในการคำนวณดังสมการที่ 2.13 ดังนั้นรูปแบบในการคำนวณนั้น จะอยู่ในรูปแบบของสมการ

$$\alpha_k(m) = \frac{\sum_i e^{\delta_i}}{\alpha_k}, \beta_k(m) = \frac{\sum_i e^{\delta_i}}{\beta_k} \qquad (2.25)$$

เมื่อ δ_i เป็นตัวแปรที่ได้จากการคำนวณ และสำหรับตัวแปร α_k และ β_k นั้น เป็นค่าผลรวมของค่า $\alpha_k(m)$ และ $\beta_k(m)$ ในช่วงเวลาที่พิจารณา (t=k) ตามลำดับ ซึ่งถูกนำมาหารเพื่อทำให้ข้อมูลอยู่ในรูปของค่า Normalize และสำหรับการนำค่าของ $\alpha_k(m)$ และ $\beta_k(m)$ มาใช้ในการคำนวณหาค่า LAPP เพื่อใช้ในการตัดสินใจหรือคำนวณหาค่า Extrinsic Information นั้น จะมีลักษณะของสมการที่ใช้สำหรับการคำนวณดังนี้

$$L(d_k) = \log \frac{\sum_{m}\sum_{m'} \gamma_1(R_k, m', m) e^{A_{k-1}(m')} e^{B_k(m)}}{\sum_{m}\sum_{m'} \gamma_1(R_k, m', m) e^{A_{k-1}(m')} e^{B_k(m)}} \tag{2.26}$$

โดยจะเป็นรูปแบบของการคำนวณของสมการที่ 2.23 เมื่อพิจารณาถึงการนำค่า $\alpha_k(m)$ และ $\beta_k(m)$ มาใช้ในการคำนวณในรูปแบบของค่า Logarithm ซึ่งจะพบว่ามีลักษณะในการคำนวณเช่นเดียวกับกรณีของการหาค่า $\alpha_k(m)$ และ $\beta_k(m)$ และจะเห็นได้ว่าในการคำนวณต่างๆ ของภาคถอดรหัสนั้น มีการคำนวณที่มีความซับซ้อน ดังนั้นจึงได้มีการพัฒนารูปแบบในการคำนวณที่มีความซับซ้อนที่น้อยลง โดยสำหรับการลดความซับซ้อนในการคำนวณของภาคถอดรหัสข้อมูล โดยใช้วิธีการ MAX-Log-MAP นั้น จะเป็นการใช้การประมาณค่าในการคำนวณต่างๆแทนการคำนวณสมการโดยตรง ซึ่งจะมีลักษณะในการประมาณค่าของสมการดังนี้

$$\ln(e^{\delta_1} + \ldots + e^{\delta_n}) \approx \max_{i \in \{1, \ldots, n\}} \delta_i \tag{2.27}$$

จากสมการ จะเป็นรูปแบบในการประมาณค่าของสมการที่มีการคำนวณที่ซับซ้อนให้มีการคำนวณที่มีความซับซ้อนน้อยลง โดยสำหรับรูปแบบของสมการดังกล่าวนั้น จะเป็นลักษณะของสมการที่จะต้องมีการคำนวณในการหาค่าตัวแปรต่างๆในกระบวนการถอดรหัส และเมื่อมีการนำหลักการในการประมาณค่านี้มาทำการใช้งาน จะเป็นการทำให้การคำนวณหาค่าต่างๆนั้น ลดความซับซ้อนในการคำนวณลงเหลือเพียงแค่การคำนวณหาค่าสูงสุดของข้อมูลเท่านั้น และเมื่อพิจารณาถึงการคำนวณหาค่าแต่เนื่องจากกระบวนการที่ใช้ในการคำนวณต่างๆนั้น เป็นเพียงแค่การประมาณการ ดังนั้นประสิทธิภาพในการทำงานของวงจรถอดรหัสที่มีการนำวิธีการ MAX-Log-MAP มาใช้งานนั้น จะมีค่าที่ต่ำกว่ากรณีที่มีการคำนวณด้วยวิธีการ MAP

2.31 วิธีการ Log-MAP

เนื่องจากวิธีการลดความซับซ้อนในการคำนวณของภาคถอดรหัสด้วยวิธีการ MAP-Log-MAP นั้น จะมีลักษณะในการคำนวณที่เป็นเพียงการประมาณการเท่านั้น ดังนั้นเมื่อมีการนำวิธีการ MAX-Log-MAP มาใช้ในการทำงานนั้น จะมีผลทำให้ประสิทธิภาพในการทำงานของภาคถอดรหัสนั้นมีค่าลดลง ดังนั้น ในการที่จะทำให้ประสิทธิภาพในการทำงานของภาคถอดรหัสนั้น มีค่าสูงขึ้น จะต้องมีการปรับปรุงรูปแบบในการคำนวณให้มีค่าที่ใกล้เคียงค่าที่แท้จริงมากยิ่งขึ้น โดยการเพิ่มการคำนวณในส่วนของ Jacobi Logarithm ในสมการ ดังนั้นรูปแบบของสมการที่ใช้สำหรับการคำนวณในกรณีของ Log-MAP นั้น จะมีลักษณะดังนี้

$$\ln(e^{\delta_1} + e^{\delta_2}) = \max(\delta_1, \delta_2) + \ln\left(1 + e^{-|\delta_2 - \delta_1|}\right) = \max(\delta_1, \delta_2) + f_c\left(|\delta_1 - \delta_2|\right) \tag{2.28}$$

โดยที่การคำนวณในส่วนท้ายของสมการที่แทนด้วยสัญลักษณ์ $f_c(.)$ นั้น จะเป็นส่วนที่ถูกนำมาทำการคำนวณเพื่อทำให้การคำนวณนั้น มีค่าที่ใกล้เคียงค่าที่แท้จริงมากขึ้น ดังนั้น เมื่อมีการนำรูปแบบของสมการดังกล่าวมาใช้ในการคำนวณ จะได้ลักษณะในการคำนวณดังสมการ

$$\begin{aligned} &\ln(e^{\delta_1} + \ldots + e^{\delta_n}) \\ &= \ln(\Delta + e^{\delta_n}) \\ &= \max(\ln\Delta, \delta_n) + f_c\left(\ln\Delta - \delta_n\right) \\ &= \max(\delta\ , \delta_n) + f_c\left(\delta - \delta_{n1}\right) \end{aligned} \tag{2.29}$$

เมื่อ Δ มีค่าเท่ากับ $e^{\delta_1} + \ldots + e^{\delta_{n-1}}$ ซึ่งผลที่ได้จากการทำงานของภาคถอดรหัสโดยใช้วิธีการ Log-MAP นั้น จะมีประสิทธิภาพในการทำงานที่สูงขึ้นเมื่อเปรียบเทียบกับกรณีที่มีการใช้วิธีการ MAX-Log-MAP แต่ในการคำนวณต่างๆนั้น จะมีความซับซ้อนมากขึ้น เนื่องจากมีการคำนวณหาค่า Jacobi Logarithm ซึ่งสามารถที่จะลดความซับซ้อนในการคำนวณในส่วนของการคำนวณหาค่า Jacobi Logarithm ได้โดยการใช้การประมาณค่าที่ได้จากการคำนวณจากฟังก์ชัน $f_c(.)$ และทำการบันทึกไว้ในเมมโมรี่ในวงจร

2.32 การนำวิธีการเข้ารหัสแบบ Turbo Codes ไปใช้งาน

เนื่องจากการเข้ารหัสแบบ Turbo Codes เป็นรูปแบบในการเข้ารหัสข้อมูลในระบบสื่อสารที่มีประสิทธิภาพในการลดอัตราการเกิดความผิดพลาดของข้อมูลสูงเมื่อเปรียบเทียบกับรูปแบบในการเข้าข้อมูลแบบอื่นๆ ดังนั้นจึงได้มีการนำวิธีการเข้ารหัสข้อมูลแบบ Turbo Codes มาประยุกต์ใช้ในการทำงานในรูปแบบต่างๆหลายประเภทด้วยกัน ได้แก่

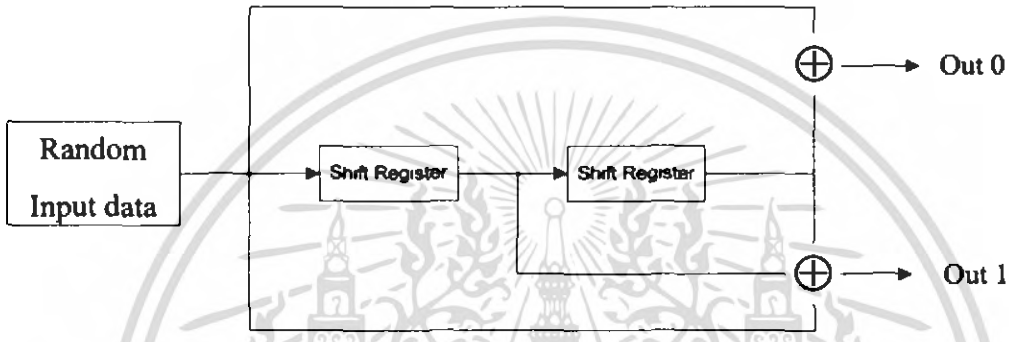
2.33 *Deep-Space Communications*

สำหรับการส่งข้อมูลผ่านระบบสื่อสารในอวกาศนั้น ได้มีการนำวิธีการเข้ารหัสข้อมูลแบบ Turbo Codes มาใช้ในการลดอัตราการเกิดความผิดพลาดของข้อมูลที่ถูกส่งให้มีค่าที่ลดลง หรือ ทำให้สามารถส่งข้อมูลด้วยกำลังส่งที่ลดลงได้ โดยสำหรับกรณีที่มีการนำวิธีการเข้ารหัสข้อมูลในรูปแบบของ Turbo Codes มาใช้ในระบบสื่อสารในอวกาศนั้น สามารถที่จะใช้วิธีการ Interleave ข้อมูลครั้งละจำนวนมากๆได้ จึงส่งผลทำให้ข้อมูลที่ได้จากวงจรเข้ารหัสนั้นมีความเป็นอิสระต่อกันมากขึ้น และจะทำให้ค่า Coding Gain ที่ได้จากการเข้ารหัสนั้นมีค่าที่สูงขึ้นตามไปด้วย

บทที่ 3

การออกแบบและการสร้าง

3.1 โครงสร้างการเข้ารหัสแบบ Convolutional Code



รูปที่ 3.1 การเข้ารหัสแบบ Convolution Code ที่ K=3

โครงสร้างการเข้ารหัสแบบ Convolutional Code จะมีการทำงานโดยดึงข้อมูลที่จะทำการเข้ารหัสมาครั้งละ 1 บิต (k=1) เข้ามาภายในวงจร ซึ่งจะทำให้ข้อมูลที่อยู่ในตำแหน่งต่างๆ ของ Shift-Register จากนั้นถูกเลื่อนไปอยู่ในตำแหน่งถัดไป ต่อจากนั้นจะมีการนำข้อมูลทั้งหมดที่เก็บไว้ใน Shift-Register มาทำการคำนวณเพื่อหาเอาท์พุทจำนวน 2 บิต (n = 2) ซึ่งอัตราการเข้ารหัสเท่ากับ ½ (Code Rate=k/n=1/2) ที่จะเป็นผลลัพธ์ส่งออกไปจากภาคเข้ารหัสข้อมูล ส่วนค่าจำนวนของ Shift-Register และอินพุทข้อมูลที่นำมาบวกแบบ Modulo-2 เรียกว่าค่า Constraint Length (K) โดยจะใช้ Generator Polynomial เพื่อแสดงตำแหน่งของใน Shift-Register ที่จะนำมาหาค่าเอาท์พุทโดยการบวกแบบ Modulo-2 ในการแสดงลักษณะของวงจรเข้ารหัส โดยสำหรับกรณีของวงจรในรูปที่ 3.1 นั้น จะมี Generator Polynomial $g_0=5_8$ และ $g_1=7_8$ หรือ สามารถเขียนให้อยู่ในรูปเลขฐาน 2 ได้เป็น g_0 = 1 0 1 และ g_1 = 1 1 1 ซึ่งหมายถึง ที่ g_0 จะมีการดึงข้อมูลมาทำ Exclusive or เฉพาะตำแหน่งที่ค่า g มีค่าเป็น 1 เท่านั้น คือ นำข้อมูลที่ตำแหน่งที่ 1 กับ 3 มาคิดเท่านั้น และที่ g_1 ก็จะนำข้อมูลที่ตำแหน่ง 1,2 และ 3 มาทำ Exclusive or

โดย Input Data สามารถสร้างโดยใช้ฟังก์ชั่น rand() โดยค่าที่ได้ออกมาจากการจะเป็นทศนิยมระหว่าง 0 ถึง 1ซึ่งสามารถปัดค่าให้เป็น 0 และ 1 โดยใช้ฟังก์ชั่น *round()* และจะมีการดึงค่าจากอินพุตเข้าไปทีละ 1 บิต และได้มีการกำหนดให้ 1 เฟรมมี 270 บิต

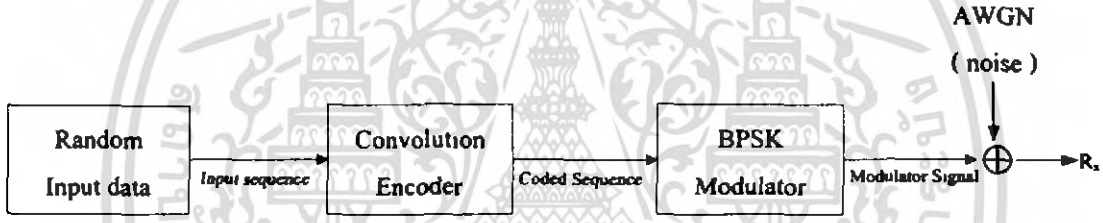
3.2 ค่าที่ได้ออกมาจากการเข้ารหัสแบบ Convolutional Code

จากรูปที่ 3.1 จะเห็นว่าเมื่อป้อนอินพุตเข้าไป 1 บิตจะได้เอาพุตออกมา 2 บิต ซึ่งเราสามารถเขียนเอาพุตที่ได้จากการเข้ารหัสได้ ดังนี้

[$out_1 0$ $out_1 1$, $out_2 0$ $out_2 1$, $out_3 0$ $out_3 1$, $out_4 0$ $out_4 1$, ...]

จากนั้นเราจะนำค่าที่ได้จากการเข้ารหัสแบบ Convolutional นี้ไปทำการ Modulation แบบ BPSK (Binary Phase Shift Keying) ซึ่งถ้า Output ที่ได้จากการเข้ารหัสเป็น 1 จะ Modulation ออกมาได้เป็น 1 และถ้า Output ที่ได้จากการเข้ารหัสเป็น 0 จะ Modulation ออกมาได้เป็น -1

จากนั้นเราก็จะนำค่าที่ได้จากการ Modulation แบบ BPSK นี้ไปบวกกับ สัญญาณรบกวนแบบสัญญาณรบกวนเกาส์สีขาวแบบบวก (Additive White Gaussian Noise : AWGN) แล้วจึงนำค่าข้อมูลที่บวกกับ Noise แล้วส่งไปให้ภาค Decode ซึ่งจะแสดงบล็อคไดอะแกรมได้ดังรูป



รูปที่ 3.2 การจำลองการทำงานของภาคส่ง

3.3 การถอดรหัสข้อมูลแบบ MAP (Maximum a Posteriori)

ในการถอดรหัสข้อมูลเพื่อหาค่าของข้อมูลดิบ d_k ที่คาดว่าจะถูกส่งมา ณ เวลานั้นๆว่าควรจะมีค่าเท่าใด โดยจะมีการพิจารณาจากค่าที่เรียกว่า Log-likelihood Ratio ซึ่งจะมีค่าเท่ากับ

$$L(u_k) = \log\left(\frac{P(d_k = +1 \mid R_k)}{P(d_k = -1 \mid R_k)}\right) \tag{3.1}$$

โดยตัวแปร c_i นั้นจะเป็นตัวแปรที่ใช้แทนข้อมูลดิบที่ได้จากการถอดรหัส ณ เวลานั้นๆ โดยสำหรับกรณีสมการที่ 3.1 นี้เป็นกรณีที่มีการส่งข้อมูลโดยใช้วิธีการมอดูเลตแบบ BPSK ดังนั้นข้อมูลดิบที่จะถูกส่งมานั้นจะมีค่าเท่ากับ +1 หรือ -1 ซึ่งแทนข้อมูลไบนารี่ 1 หรือ 0 เท่านั้น และสำหรับค่า Log-likelihood Ratio นี้จะเป็นตัวแปรที่จะถูกนำมาใช้สำหรับการตัดสินใจว่าข้อมูลที่จะถูกส่งมา ณ เวลานั้นๆ น่าจะมีค่าเป็นเท่าใด และตัวแปร R_k นั้นเป็นตัวแปรที่แทนสัญญาณข้อมูลที่รับได้ ณ ช่วงเวลาต่างๆ โดยถ้าหากว่าค่า Log-likelihood Ratio ที่ได้จากการคำนวณชุดข้อมูล ณ

เวลานั้นๆมีค่ามากกว่า 0 จะมีการตัดสินใจให้ข้อมูลดิบที่ได้จากการถอดรหัส ณ เวลานั้นมีค่าเท่ากับ +1 ซึ่งแทนข้อมูลไบนารี่ที่มีค่าเป็น 1 แต่ในกรณีที่ค่า Log-likelihood Ratio ที่ได้นั้นมีค่าน้อยกว่า 0 แล้วจะมีการตัดสินใจให้ข้อมูลดิบที่ได้จากการถอดรหัส ณ เวลานั้นมีค่าเท่ากับ –1 ซึ่งแทนข้อมูลไบนารี่ที่มีค่าเป็น 0 ซึ่งสามารถเขียนให้อยู่ในรูปวงจรถอดรหัสการตัดสินใจแบบฮาร์ดโดยการเปรียบเทียบ $L(u_k)$ กับ 0

$$L(u_k)=\begin{cases}1, & \text{if } \Lambda(d_k) > 0\\ 0, & \text{Ortherwise}\end{cases} \tag{3.2}$$

จะพบว่าลักษณะของสมการที่ใช้ในการคำนวณนั้นจะประกอบไปด้วยการคำนวณระหว่างค่าความน่าจะเป็น 3 ค่าด้วยกัน ดังนั้นจึงได้มีการนิยามตัวแปรขึ้นมา 3 ตัวแปรด้วยกันเพื่อใช้ในการคำนวณหาค่า Joint Probability ได้แก่ตัวแปร Forward State Matric ($\alpha_t(I)$), Backward State Metric ($\beta_t(I)$) และ Branch Matric ($\gamma'_t(I',I)$) โดยที่ตัวแปรต่างๆนั้นจะมีการนิยามให้มีค่าเท่ากับ

$$\alpha_k(m) = \Pr\{S_k = m \mid R_1^k\} \tag{3.3}$$

$$\beta_k(m) = \frac{\Pr\{R_{k+1}^N \mid S_k = m\}}{\Pr\{R_{k+1}^N \mid R_1^k\}} \tag{3.4}$$

$$\gamma_t(R_k, m', m) = \Pr\{ d_k = i, S_k = m, Rk \mid S_{k-1} = m'\} \tag{3.5}$$

S_{k-1} กับ S_k เป็นสถานะของวงจรถอดรหัสที่เวลา k-1 กับ k ตามลำดับ

โดยที่ตัวแปร $\alpha_k(m)$ และ $\beta_k(m)$ ซึ่งเป็นตัวแปรที่แสดงถึงค่าความน่าจะเป็นของข้อมูลที่ถูกเก็บไว้ในวงจรเข้ารหัสก่อน และ หลังที่จะมีการป้อนข้อมูลดิบเข้ามาภายในวงจรในกรณีต่างๆตามลำดับ โดยในการคำนวณนั้น จะมีการนำค่า $\gamma_t(R_k, m', m)$ ซึ่งเป็นตัวแปรที่เกิดจากการนำสัญญาณข้อมูลที่รับได้ ณ เวลาต่างๆมาเปรียบเทียบกับค่าของข้อมูลต่างๆที่จะถูกส่งออกมาจากวงจรเข้ารหัสในกรณีต่างๆ มาใช้สำหรับการคำนวณหาค่า $\alpha_k(m)$ และ $\beta_k(m)$ โดยจะมีลักษณะการคำนวณแบบย้อนกลับ (Recursive) ดังสมการ

$$\alpha_k(m) = \frac{\sum_{m}\sum_{i=0}^{1}\gamma_i(R_k, m', m)\alpha_{k-1}(m')}{\sum_{m}\sum_{m'}\sum_{i=0}^{1}\gamma_i(R_k, m', m)\alpha_{k-1}(m')} \tag{3.6}$$

$$\beta_k(m) = \frac{\sum_{m}\sum_{i=0}^{1}\gamma_i(R_k, m', m)\beta_{k+1}(m')}{\sum_{m}\sum_{m'}\sum_{i=0}^{1}\gamma_i(R_k, m', m)\beta_{k+1}(m')} \tag{3.7}$$

เมื่อ u_k และ v_k เป็นค่าของข้อมูลที่จะได้จากการทำงานของภาคเข้ารหัสเมื่อมีการป้อนข้อมูลดิบมีค่าเท่ากับ i และข้อมูลที่อยู่ในวงจรเข้ารหัสก่อนและหลังจากที่มีการป้อนข้อมูลเข้ามามีค่าเป็น m' และ m ตามลำดับ และสำหรับค่าความน่าจะเป็น $q(d_k = i \mid S_k = m, S_{k-1} = m')$ จะเป็นค่าที่เกิดจากการวิเคราะห์ความเป็นไปได้ของข้อมูลที่อยู่ในวงจรเข้ารหัสหลังและก่อนที่มีการป้อนข้อมูลดิบเข้ามาภายในวงจรและค่าของข้อมูลที่ได้จากการเข้ารหัส ในช่วงเวลาการทำงานต่างๆว่ามีความเป็นไปได้ที่จะมีค่าเป็นไปตามนั้นมากน้อยเพียงใด โดยในกรณีของวงจรเข้ารหัสแบบ Convolution Codes นั้น เนื่องจากรูปแบบในการทำงานที่มีรูปแบบที่แน่นอน ดังนั้นค่า $q(d_k = i / S_k = m, S_{k-1} = m')$ ที่ได้นั้น จะมีค่าแค่เพียง 0 หรือ 1 เท่านั้น และสำหรับค่าความน่าจะเป็น $\pi(S_k = m \mid S_{k-1} = m')$ นั้น จะเป็นค่าที่เกิดจากการคาดคะเนค่าของข้อมูลดิบที่จะถูกป้อนเข้ามาภายในวงจรเข้ารหัส ณ เวลานั้นๆ โดยทั่วไปจะมีค่าเท่ากับ $P(d_k = 1) = P(d_k = 0) = 1/2$

ดังนั้น สำหรับขั้นตอนที่ใช้สำหรับการถอดรหัสข้อมูลโดยใช้วิธีการ MAP นั้น จะมีขั้นตอนในการทำงานดังต่อไปนี้

1. นำข้อมูลที่รับเข้ามาได้มาทำการคำนวณหาค่า Branch Metrics, $\gamma_k(R_k, m', m)$ ในกรณีต่างๆ โดยใช้สมการที่ 2.18
2. นำค่า Branch Metrics ที่ได้จากการคำนวณมาใช้ในการคำนวณหาค่า Forward State Matric , $\alpha_k(m)$ โดยใช้สมการที่ 3.6
3. นำค่า Branch Metrics ที่ได้จากการคำนวณมาใช้ในการคำนวณหาค่า Reverse State Matric , $\beta_k(m')$ โดยใช้สมการที่ 3.7
4. ทำการคำนวณหาค่า LAPP ของข้อมูลบิตที่ k โดยใช้สมการ

$$L(d_k) = \log \frac{\sum_{m}\sum_{m'} \gamma_1(R_k, m', m)\alpha_{k-1}(m')\beta_k(m)}{\sum_{m}\sum_{m'} \gamma_0(R_k, m', m)\alpha_{k-1}(m')\beta_k(m)}$$

5 ทำการตัดสินใจค่าของข้อมูลดิบที่ได้จากการถอดรหัสโดยการพิจารณาจากค่า LAPP ที่ได้จากการคำนวณ โดยที่ถ้าค่า LAPP มีค่ามากกว่า 0 จะมีการตัดสินใจให้ข้อมูลดิบ d_k ที่ได้จากการถอดรหัส ณ เวลานั้นมีค่าเป็น +1 แต่ถ้าหากมีค่าน้อยกว่า 0 แล้ว จะตัดสินใจให้ข้อมูลดิบ d_k มีค่าเป็น -1

3.4 การถอดรหัสข้อมูลแบบ Log-MAP

จะเป็นการนำวิธีการของ MAP มาเพิ่มการคำนวณในส่วนของ Jacobi Logarithm ในสมการ ดังนั้นรูปแบบของสมการที่ใช้สำหรับการคำนวณในกรณีของ Log-MAP นั้น จะมีลักษณะดังนี้

$$\ln(e^{\delta_1} + e^{\delta_2}) = \max(\delta_1, \delta_2) + \ln\left(1 + e^{-|\delta_2 - \delta_1|}\right) = \max(\delta_1, \delta_2) + f_c\left(|\delta_1 - \delta_2|\right) \qquad (3.8)$$

โดยที่การคำนวณในส่วนท้ายของสมการที่แทนด้วยสัญลักษณ์ $f_c(.)$ นั้น จะเป็นส่วนที่ถูกนำมาทำการคำนวณเพื่อทำให้การคำนวณนั้น มีค่าที่ใกล้เคียงค่าที่แท้จริงมากขึ้น ดังนั้น เมื่อมีการนำรูปแบบของสมการดังกล่าวมาใช้ในการคำนวณ จะได้ลักษณะในการคำนวณดังสมการ

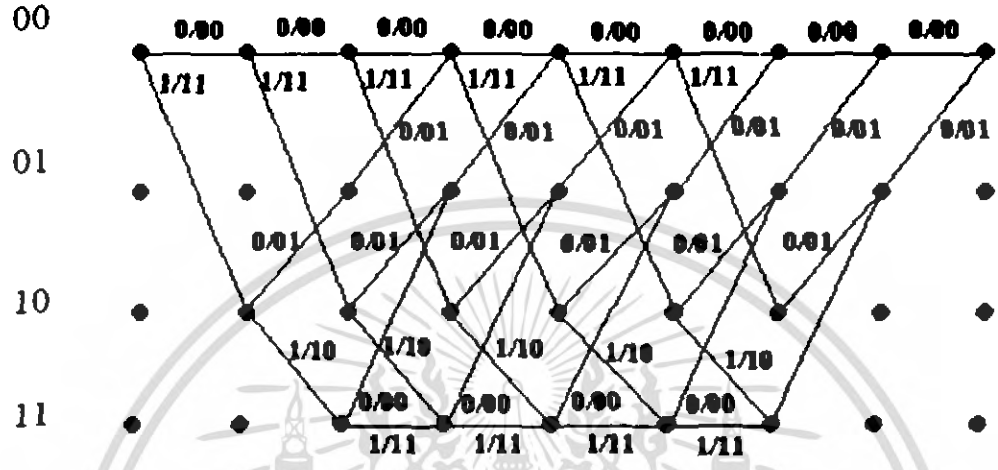
$$\begin{aligned} &\ln(e^{\delta_1} + \ldots + e^{\delta_n}) \\ &= \ln(\Delta + e^{\delta_n}) \\ &= \max(\ln\Delta, \delta_n) + f_c\left(\ln\Delta - \delta_n\right) \\ &= \max(\delta\ , \delta_n) + f_c\left(\delta - \delta_{n1}\right) \end{aligned} \qquad (3.9)$$

โดย Δ มีค่าเท่ากับ $e^{\delta_1} + \ldots + e^{\delta_{n-1}}$

3.5 การถอดรหัสข้อมูลแบบ Viterbi Coding

การถอดรหัสข้อมูลแบบ Viterbi จะสามารถคำนวณหาข้อมูลอินพุตได้จากเทลลิสไดอะแกรม ซึ่งจะเป็นกระบวนการที่ใช้ในการค้นหาเส้นทางที่อยู่ในเทลลิสไดอะแกรม ที่มีลักษณะที่ใกล้เคียงกับข้อมูลที่รับได้มากที่สุดเพื่อที่จะนำข้อมูลในเส้นทางนั้นมาคำนวณหาค่าของข้อมูลที่

ถูกส่งมา โดยเราสามารถนำข้อมูลที่ได้จากการเข้ารหัสแบบ Convolutional Code มาทำการหาข้อมูลอินพุต



รูปที่ 3.3 Trellis Diagram

โดยเราจะนำข้อมูลในอดีตจำนวน 2 บิต มาทำการประมวลผลร่วมกับข้อมูล ณ เวลานั้น (จำนวน State ทั้งหมดใน Trellis Diagram จะมีค่าเท่ากับ $2^2 = 4$ state) และจะมีข้อมูลป้อนเข้ามาภายในวงจรครั้งละ 1 บิต โดยที่ในกระบวนการค้นหาเส้นทางที่เหมาะสมที่สุดโดยใช้ Viterbi Algorithm นั้น จะมีขั้นตอนในการทำงานดังต่อไปนี้

1) พิจารณาแบ่งข้อมูลที่รับเข้ามาออกเป็นข้อมูลย่อยๆจำนวน m ช่วง ซึ่งแต่ละช่วงนั้นมีขนาดของข้อมูลเท่ากับ n_0 บิต

2) ทำการวาด Trellis Diagram ที่มีจำนวน State ในการทำงานเท่ากับ m State โดยจะมีการพิจารณาเฉพาะเส้นทางที่มีความเป็นไปได้ว่าจะถูกส่งมาเท่านั้น

3) กำหนดค่าตัวแปร $l = 1$ และทำการกำหนดค่าเริ่มต้นของตัวแปร Metric ในสภาวะเริ่มต้นที่มีข้อมูลเป็น 0 ทั้งหมด ให้มีค่าของ Metric เท่ากับ 0

4) ทำการคำนวณหาค่าความแตกต่างของข้อมูล (Distance) ระหว่างข้อมูลที่รับได้ชุดที่ l กับข้อมูลในเส้นทางในการเปลี่ยนแปลงสถานะใน Trellis Diagram จาก State ที่ l ไปเป็น l+1

5) นำค่าที่คำนวณได้นั้นไปบวกกับค่า Metric สะสมของ State l เพื่อคำนวณหาค่าของ Metric สะสมใน State ที่ l+1 เพื่อใช้ในการตัดสินใจเลือกเส้นทางที่เหมาะสมที่สุด ในการเปลี่ยนแปลงข้อมูลไปยัง State นั้นๆ โดยในแต่ละ State นั้น จะมีจำนวนเส้นทางทั้งหมดจำนวน $2k_0$ เส้นทางที่จะพุ่งเข้า State เดียวกัน

6) พิจารณา ณ ตำแหน่งใน State ที่ l+1 ในแต่ละ State นั้น ทำการเลือกเส้นทางที่มีค่า Metric สะสมที่มีค่าน้อยที่สุดที่พุ่งเข้าหาในแต่ละ State โดยที่เส้นทางที่ถูกเลือกนั้น จะถูกเรียกว่า "Survivor" ซึ่งจะเป็นเส้นทางที่ถูกเก็บไว้ทำการคำนวณใน State ต่อไป และสำหรับเส้นทางอื่นๆที่ไม่ได้ถูกเลือกนั้น จะถูกเรียกว่า "Forgetting" โดยจะถูกลบทิ้งออกไปจากระบบการตัดสินใจ

7) ถ้าหากว่า l นั้นมีค่าเท่ากับ m แล้วให้ทำงานในขั้นตอนต่อไปได้ แต่ถ้ายังมีค่าน้อยกว่าจะมีต้องมีการเพิ่มค่า l ขึ้นอีก 1 จากนั้นจึงกลับไปทำงานที่ขั้นตอนที่ 4 ใหม่

8) เริ่มต้นพิจารณา ณ State ที่ m+1 ที่มีสภาวะของข้อมูลสถานะเป็น 0 ทั้งหมด ทำการเลือกเส้นทางที่เป็น "Survival" ซึ่งเป็นเส้นทางที่ถูกเลือกที่เหลืออยู่ย้อนกลับไปจนกระทั่งถึงสภาวะเริ่มต้นของการทำงานที่มีสภาวะในการทำงานเป็น 0 ทั้งหมด ซึ่งเส้นทางที่ได้นั้น จะเป็นเส้นทางที่มีลักษณะที่ใกล้เคียงกับข้อมูลที่รับเข้ามามากที่สุด ซึ่งจะถูกนำไปใช้ในการคำนวณหาข้อมูลข่าวสารที่ถูกส่งมา โดยข้อมูลข่าวสารที่จะถูกส่งออกไปจากภาคถอดรหัสนั้น จะเป็นการส่งข้อมูลทั้งหมดที่อยู่ในเส้นทางส่งออกไปยกเว้นข้อมูลที่อยู่ 2 บิตท้ายสุดนั้น จะถูกตัดทิ้งไป

ซึ่งถ้าจะนำมาเขียนให้อยู่ในรูปของสมการจะสามารถคำนวณหาเส้นทางโดยหาค่าความแตกต่างน้อยที่สุดได้ดังตามขั้นตอนต่อไปนี้

ขั้นตอนที่ 1 Branch Metric Generation

ขั้นตอนนี้คำนวณหาค่า Branch Metric (BM) จากข้อมูลอินพุตที่รับเข้ามา r กับค่าเอาต์พุตของการเข้ารหัส C การคำนวณหาค่า Branch Metric ต้องคำนวณทุกๆ สาขาหรือ Branch การคำนวณหาค่า Branch Metric แสดงดังสมการต่อไปนี้

$$BM_{i,j,n} = (r_n - C_{i,j})^2 \tag{3.10}$$

โดย ค่า BM แทนค่า Branch Metric ระหว่าง State i ไปยัง State j ณ เวลา n

ค่า r แทนค่า ข้อมูลอินพุตที่รับเข้ามา ณ เวลา n

ค่า C แทนค่าเอาต์พุตของการเข้ารหัสระหว่าง State i ไปยัง State j ณ เวลา n

ขั้นตอนที่ 2 Survivor Path และ Path Metric Update

ขั้นตอนนี้คำนวณหาค่า Survivor Path และ Path Metric จากจำนวน State การทำงานทั้งหมดค่า Path Metric ที่เลือกไว้เพื่อใช้ในการหาค่า Path Metric ครั้งต่อไป (Update) ส่วนค่า

Survivor Path เป็นค่าที่ใช้ในการตัดสินใจหาค่าเอาต์พุต โดยการคำนวณหาค่า Survivor Path และ Path Metric นั้นค่าของ Branch Metric และ Path Metric จะถูกเข้าด้วยกัน ซึ่งผลการบวกนั้นมีสองค่าที่เข้ามาในแต่ละจุดเชื่อมต่อ (Trellis Node) ของ Trellis Diagram โดยค่า Path Metric เป็นค่าที่เลือกจากค่าผลบวกที่น้อยกว่า ส่วนค่า Survivor Path เป็น State การทำงานที่น้อยกว่าจากการเลือก Path Metric ซึ่งแสดงดัง

$$PM_{j,n} = \min(PM_{i,n-1} + BM_{i,j,n}, PM_{i+1,n-1} + BM_{i+1,j,n}) \tag{3.11}$$

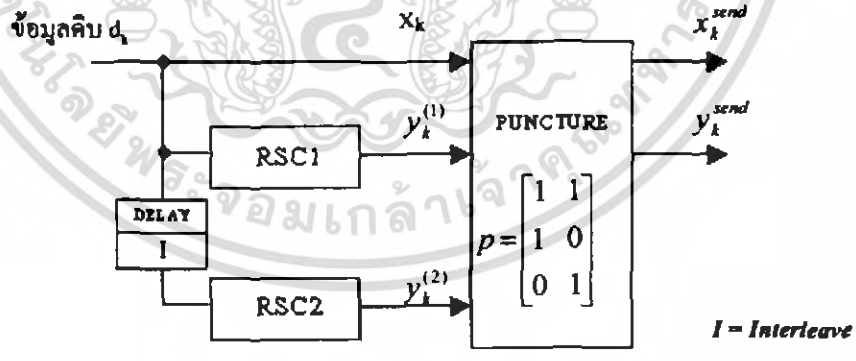
โดย ค่า PM แทน Path Metric ระหว่าง State i ไปยัง State j ณ เวลา n

ค่า C แทนค่าเอาต์พุตของการเข้ารหัสระหว่าง State i ไปยัง State j ณ เวลา n

ขั้นตอนที่ 3 Optimum Paths Trace Back

ขั้นตอนนี้เป็นขั้นตอนการตัดสินใจหาค่าเอาต์พุต โดยใช้ค่า Survivor Path ในแต่ละ State ที่บันทึกไว้มาตัดสินใจเลือกเส้นทางของข้อมูล โดยการตัดสินใจหาเส้นทางของข้อมูลจะเริ่มจาก Survivor Path ในอดีต (Trace Back)

3.6 โครงสร้างการเข้ารหัสข้อมูลแบบ Turbo Codes



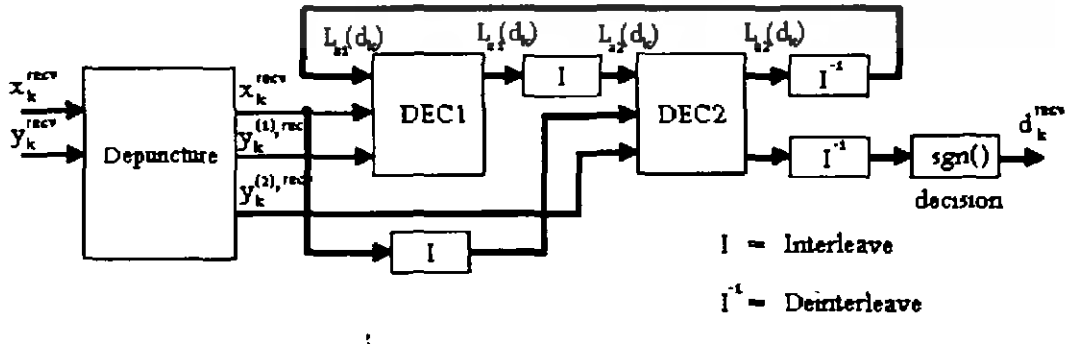
รูปที่ 3.4 วงจรเข้ารหัสแบบ Turbo Codes

สำหรับรูปแบบในการเข้ารหัสข้อมูลแบบ Turbo Codes นั้น จะมีรูปแบบในการทำงานพื้นฐานดังวงจรในรูปที่ 3.4 โดยจะเป็นการนำวงจรเข้ารหัสข้อมูลอย่างน้อย 2 ชุดมาทำการใช้งานร่วมกันในรูปแบบของการเข้ารหัสแบบ Parallel Concatenate ซึ่งเป็นวิธีที่ใช้สำหรับการนำวงจรเข้ารหัสตั้งแต่ 2 วงจรขึ้นไปมาทำงานร่วมกันเพื่อทำให้ได้รูปแบบในการเข้ารหัสข้อมูลที่มี

ประสิทธิภาพในการทำงานที่สูงขึ้น โดยที่ไม่มีการใช้รูปแบบของวงจรที่มีความซับซ้อนมากขึ้น โดยสำหรับกรณีของวงจรเข้ารหัสแบบ Turbo Codes นั้น จะมีการนำการเข้ารหัสแบบ Recursive Systematic Convolution Codes (RSC) [2] ซึ่งเป็นรูปแบบในการเข้ารหัสข้อมูลที่มีการพัฒนามาจากการเข้ารหัสแบบ Convolution Codes มาใช้ในการทำงาน โดยในการทำงานของภาคเข้ารหัสข้อมูลในรูปที่ 3.4 นั้น จะเป็นการนำข้อมูลดิบที่จะทำการเข้ารหัส $d_k = \{d_0, d_1, d_2, ...\}$ มาทำการเข้ารหัสข้อมูล โดยการป้อนข้อมูลดิบเข้ามาภายในวงจรเข้ารหัส RSC1 และ RSC2 ครั้งละ 1 บิต เพื่อทำการคำนวณหาค่าของข้อมูลที่ผ่านการเข้ารหัส (Codeword) ซึ่งผลลัพธ์ที่ได้นั้นจะมีลักษณะข้อมูลเป็นแบบ Systematic กล่าวคือ ข้อมูลที่ได้จากวงจรเข้ารหัสในแต่ละครั้งที่มีการทำงานนั้น จะประกอบไปด้วยข้อมูล 2 ส่วนด้วยกัน ได้แก่ข้อมูลในส่วนที่มีค่าเหมือนกับข้อมูลดิบ (x_k) และข้อมูลในส่วนที่ได้จากภาคเข้ารหัส RSC1 และ RSC2 ($y_k^{(1)}, y_k^{(2)}$) ดังนั้นข้อมูลที่ได้จากการเข้ารหัสนั้นจะมีค่าเท่ากับ $\{x_0, y_0^{(1)}, y_0^{(2)}, x_1, y_1^{(1)}, y_1^{(2)}, x_2, y_2^{(1)}, y_2^{(2)}, ...\}$ โดยที่ข้อมูลดิบที่จะมีการนำมาใช้ในการเข้ารหัสข้อมูลสำหรับวงจร RSC2 นั้น จะมีการนำข้อมูลดิบมาผ่านกระบวนการ Interleave ก่อนที่จะมีการส่งมายังภาคเข้ารหัสเพื่อที่จะทำให้ข้อมูลที่ได้จากวงจรเข้ารหัสในแต่ละวงจรนั้นไม่มีความสัมพันธ์ (Correlate) ซึ่งกันและกัน

และข้อมูลที่ได้จากการทำงานของภาคเข้ารหัสข้อมูล RSC1 และ RSC2 นั้นจะถูกส่งต่อมายังภาค Puncture ซึ่งจะมีหน้าที่ในการลดจำนวนของข้อมูลที่จะถูกส่งผ่านระบบสื่อสารให้มีจำนวนลดลงตามรูปแบบที่มีการกำหนดไว้ ซึ่งจะมีรูปแบบในการทำงานอยู่หลายรูปแบบด้วยกัน โดยในกรณีของวงจรเข้ารหัสแบบ Turbo Codes ในรูปที่ 1 นั้น จะเป็นการลดจำนวนข้อมูลที่จะส่งผ่านระบบสื่อสาร โดยการส่งสัญญาณข้อมูล $y_k^{(1)}$ และ $y_k^{(2)}$ สลับกันส่งร่วมกับสัญญาณข้อมูล x_k ดังนั้นข้อมูลที่จะได้หลังจากการทำงานของภาค Puncture นั้นจะมีค่าเท่ากับ $\{x_0, y_0^{(1)}, x_1, y_1^{(2)}, x_2, y_2^{(1)}, ...\}$

3.7 โครงสร้างการถอดรหัสข้อมูลแบบ Turbo Codes



รูปที่ 3.5 วงจรถอดรหัส Turbo Codes

สำหรับภาคถอดรหัสข้อมูลแบบ Turbo Codes ที่ใช้สำหรับนำข้อมูลที่รับได้ที่ภาครับซึ่งเป็นข้อมูลที่ถูกเข้ารหัสแบบ Turbo Codes มาทำการเปลี่ยนรูปแบบของข้อมูลที่รับได้ให้กลับมาอยู่ในรูปของข้อมูลดิบอีกครั้ง โดยใช้ข้อมูลต่างๆที่ถูกส่งมานั้นในการตัดสินใจเลือกค่าของข้อมูลดิบที่มีความเป็นไปได้มากที่สุด เพื่อส่งออกไปเป็นผลลัพธ์ที่ได้จากการทำงานของวงจรถอดรหัส โดยสำหรับกรณีของการถอดรหัสสำหรับ Turbo Codes นั้นจะมีรูปแบบในการถอดรหัสข้อมูลเป็นแบบ Iterative Decoding ที่มีรูปแบบในการถอดรหัสเป็นแบบป้อนกลับ (Feed Back) โดยในการทำงานในแต่ละรอบการทำงานนั้น จะเป็นการนำข้อมูลที่รับได้ซึ่งเป็นข้อมูลที่ผ่านการเข้ารหัสมาทำการประมวลผลเพื่อคำนวณหาค่าความน่าจะเป็นของข้อมูลดิบที่คาดว่าจะถูกส่งมา ณ เวลาต่างๆ โดยค่าความน่าจะเป็นดังกล่าวนั้น จะถูกส่งออกมาจากวงจรถอดรหัสในรูปแบบของค่าที่เรียกว่า Extrinsic Information โดยจะเป็นข้อมูลที่จะถูกนำมาใช้ในการคำนวณในรอบต่อไปเพื่อเป็นการปรับปรุงการคำนวณต่างๆ ให้มีความแม่นยำมากยิ่งขึ้นในการประมวลผลในรอบต่อไป ดังนั้น ทุกๆครั้งที่มีการทำงาน จะทำให้ข้อมูลต่างๆที่ได้จากการคำนวณนั้นมีค่าใกล้เคียงค่าที่ถูกต้องมากยิ่งขึ้น และเมื่อมีการวนรอบในการทำงานครบตามที่กำหนดแล้วจะมีการนำข้อมูลที่เรียกว่า Log A Posteriori Probability (LAPP) ที่ได้จากการทำงานมาใช้ในการตัดสินใจเพื่อเลือกค่าของข้อมูลดิบที่จะถูกส่งออกไปเป็นผลลัพธ์ของภาคถอดรหัส โดยจะมีรูปแบบในการถอดรหัสข้อมูลดังรูปที่ 3.5

ในส่วนของวิธีการถอดรหัสสำหรับ Turbo Codes ดังที่แสดงในรูปที่ 3.5 นั้น จะมีลักษณะการทำงานในรูปแบบของการถอดรหัสแบบ Iterative Decoding โดยในการถอดรหัสข้อมูลเพื่อหาค่าของข้อมูลดิบที่ถูกส่งมาในแต่ละช่วงนั้น จะเป็นการนำข้อมูลที่รับได้ในแต่ละช่วง (x_k , y_k) มาทำการ Depuncture เพื่อจัดรูปแบบของข้อมูลเพื่อกระจายข้อมูลไปยังภาคถอดรหัส *DEC1* และ *DEC2* ซึ่งจะเป็นภาคถอดรหัสที่ใช้ในการถอดรหัสข้อมูลที่ได้จากวงจรเข้ารหัส RSC1 และ 2 ของภาคเข้ารหัสตามลำดับที่มีรูปแบบในการถอดรหัสที่มีการประยุกต์มาจากวิธีการถอดรหัสแบบ Symbol-by-Symbol Maximum a Posterion (MAP) หรือ BCJR Algorithm [4] ซึ่งถูกค้นพบโดย Bahl ในปี พ.ศ. 2517 (ค.ศ 1974) มาใช้ในการทำงาน โดยในการทำงานของภาคถอดรหัสในแต่ละวงจรนั้นจะเป็นการนำข้อมูลที่รับได้ ซึ่งจะประกอบไปด้วยข้อมูลในส่วนของข้อมูลที่รับได้ (x_k^{rec}, y_k^{rec}) และข้อมูลในส่วนที่เรียกว่า a Priori Probability (L_e) ที่ได้จากการประมวลผลของวงจรถอดรหัสวงจรก่อนมาใช้ในการทำงาน โดยจะนำค่าที่ได้รับนั้นมาใช้ในการคำนวณหาค่าที่เรียกว่า Extrinsic Information ซึ่งจะเป็นข้อมูลที่จะถูกนำไปใช้ในการปรับปรุงการทำงานของวงจรถอดรหัสชุดต่อไปให้มีการคำนวณที่มีความแม่นยำมากยิ่งขึ้น และเมื่อมีการทำงานครบตามที่กำหนดแล้ว จะมีการคำนวณหาค่าที่เรียกว่า Log A Posteriori Probability (LAPP) เพื่อนำมาใช้ในการตัดสินใจค่าของข้อมูลดิบ d_k ที่จะถูกส่งออกไปเป็นผลลัพธ์ของภาคถอดรหัส

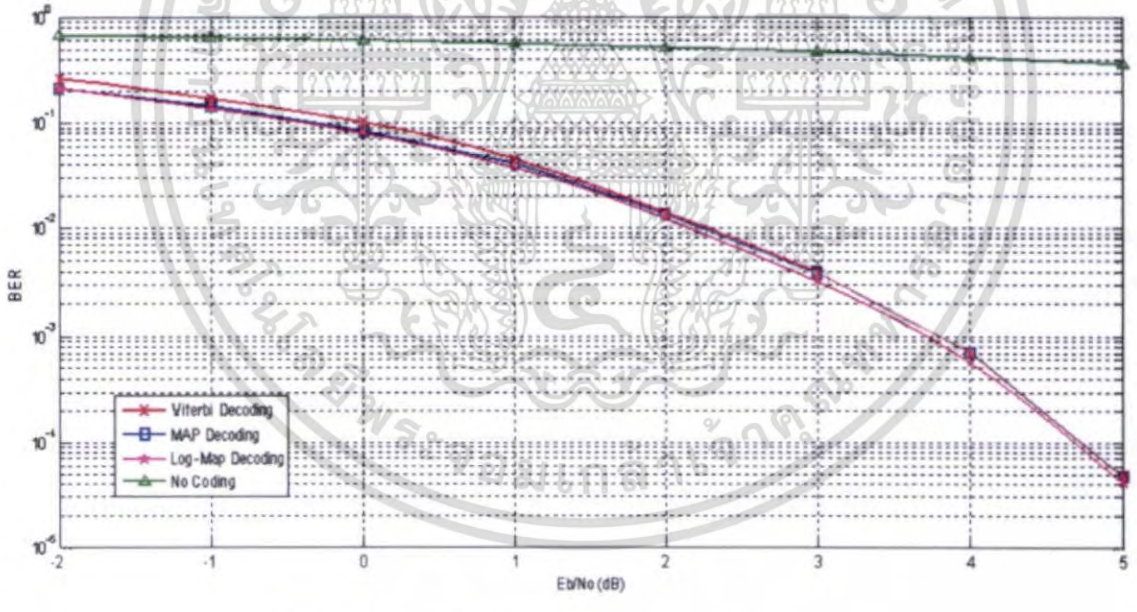
เนื่องจากข้อมูลต่างๆที่ถูกใช้ในการทำงานของภาคเข้ารหัส RSC2 นั้น จะอยู่ในรูปของข้อมูลที่ผ่านกระบวนการ Interleave เพื่อทำให้ข้อมูลที่ได้จากวงจรเข้ารหัสทั้งสองนั้นไม่มีค่าสัมพันธ์กัน ดังนั้นเพื่อให้เกิดความสัมพันธ์กับภาคเข้ารหัส จึงจำเป็นที่จะต้องนำข้อมูลต่างๆที่จะถูกส่งมาจากวงจรถอดรหัส DEC1 มายังวงจร DEC2 มาผ่านกระบวนการ Interleave เพื่อเปลี่ยนแปลงรูปแบบของข้อมูลให้อยู่ในรูปแบบเดียวกันก่อนที่จะนำมาทำการใช้งาน และในทางกลับกัน ข้อมูลต่างๆที่จะถูกส่งออกไปจากวงจรถอดรหัส DEC2 มายังวงจร DEC1 นั้น จะต้องมีการนำข้อมูลมาผ่านกระบวนการ Deinterleave ก่อนที่จะมีการนำมาใช้งาน

บทที่ 4

การทดลองและผลการทดลอง

ในการทดลองจะทำการวัดประสิทธิภาพของการเข้ารหัสแบบ Convolutional Code และมีการถอดรหัสแบบ Viterbi Decoding, MAP Decoding และ Log-Map Decoding แบบพื้นฐาน กับการเข้ารหัสในรูปแบบ Turbo Codes และถอดรหัสด้วย Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Viterbi Decoding และ Log-Map Decoding และทำการเปรียบเทียบการกำหนดค่า Gerenator Metrix และเปรียบเทียบการกำหนดขนาดจำนวนบิตต่อหนึ่งเฟรม ในการเข้ารหัสแบบ Turbo Codes โดยวัดประสิทธิภาพจากค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูล (BER) ที่มีค่าอัตราส่วนสัญญาณต่อสัญญาณรบกวน (Signal-to-Noise Ratio : *SNR*) ต่างๆกัน

4.1 ผลค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลแบบที่มีการถอดรหัสแบบ MAP Decoding Viterbi Decoding และ Log-Map Decoding กับการส่งข้อมูลที่ไม่มีการเข้ารหัส

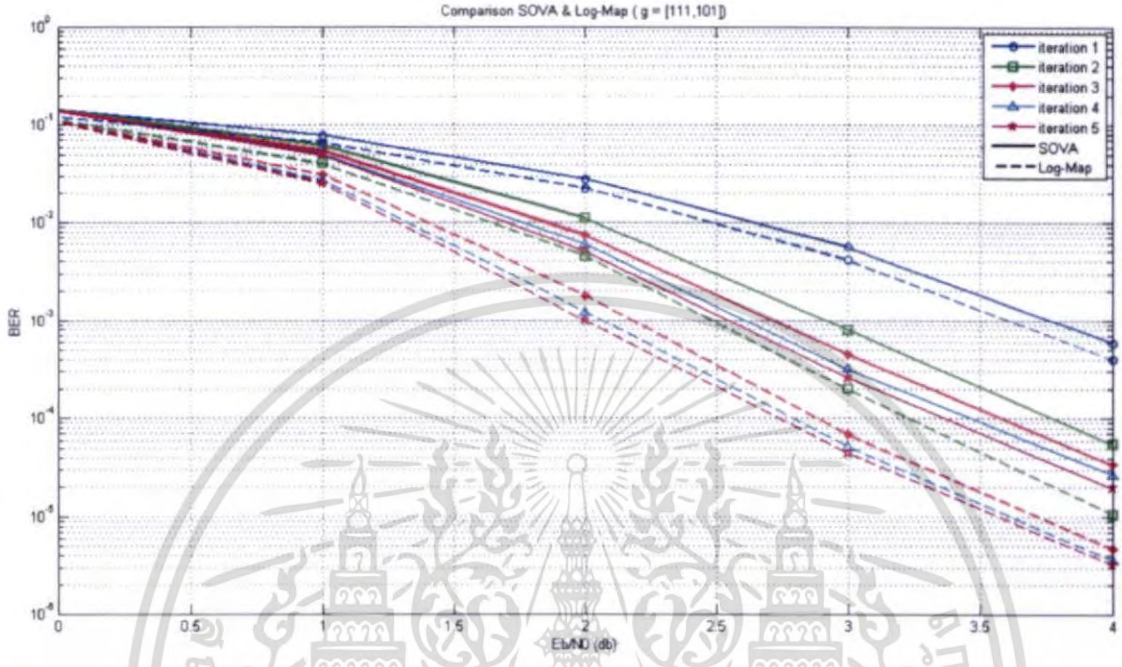


รูปที่ 4.1 ผลการทดลองค่าเฉลี่ยอัตราการผิดพลาดของบิตข้อมูลที่มีการเข้ารหัสแบบ Convolutional Code และมีการถอดรหัสแบบ Viterbi Decoding, MAP Decoding และ Log-Map Decoding กับการส่งข้อมูลที่ไม่มีการเข้ารหัส ที่แต่ละ SNR

โดยการเข้ารหัสแบบ Convolutional Code ที่ทำการทดลองนั้นจะมีค่า G (Generator Metrix) เท่ากับ [1 0 1 , 1 1 1] ซึ่งจะทำให้มีอัตราของข้อมูลเข้าต่อข้อมูลออกหรือ Code rate เป็น ½ และได้กำหนดให้แต่ละเฟรมมี 270 บิตและได้มีการมอดูเลตสัญญาณแบบ BPSK และได้นำสัญญาณรบกวนแบบ AWGN (Additive White Gaussian Noise · AWGN)มาเป็นสัญญาณรบกวนในการทดลอง โดยในการทดลองนี้สัญญาณเข้ารหัสที่ได้จากการเข้ารหัสและมอดูเลตที่ภาคส่งจะถูกบวกสัญญาณรบกวนแล้วนำมาคำนวณการถอดรหัสเลยซึ่งไม่ได้ผ่านสายอากาศแต่อย่างใด และได้ทำการหาค่าเฉลี่ยของ BER ที่ค่าอัตราส่วนสัญญาณต่อสัญญาณรบกวน (Signal-to-Noise Ratio : *SNR*) ตั้งแต่ -2 เดซิเบลถึง 5 เดซิเบลมากกว่า 50000 เฟรม

จะเห็นว่าการส่งข้อมูลแบบไม่มีการถอดรหัสนั้นจะมีค่าเฉลี่ยอัตราการผิดพลาดของบิตข้อมูล (BER) ที่สูงกว่าการส่งข้อมูลแบบมีการเข้ารหัสแบบ Convolutional Code แล้วถอดรหัสแบบ Viterbi Decoding, MAP Decoding, Log-Map Decoding ที่ทุกๆ SNR โดยจะเห็นว่าการส่งข้อมูลแบบมีการเข้ารหัสนั้นเมื่อส่งในภาวะที่มีค่า SNR สูงๆนั้นจะทำให้ได้ค่า BER ที่น้อยมากๆ แต่การส่งข้อมูลแบบไม่มีการเข้ารหัสเมื่อเพิ่ม SNR ขึ้น ค่า BER จะลดลงเพียงเล็กน้อยเท่านั้น

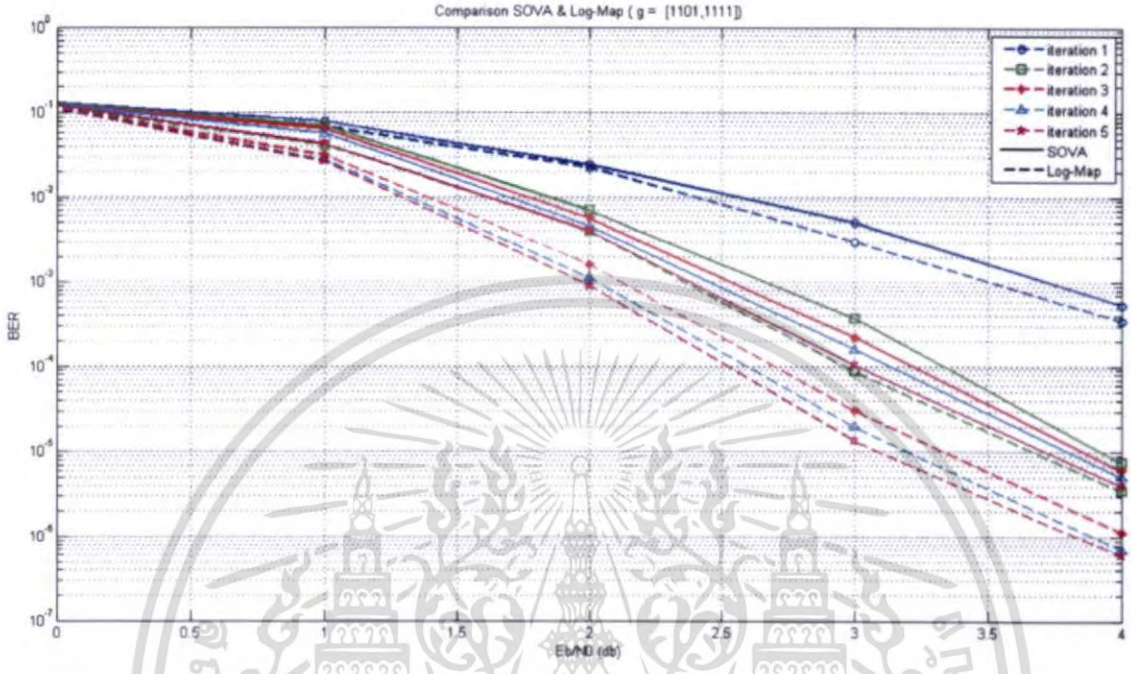
4.2 ผลค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลแบบที่มีการเข้ารหัสในรูปแบบ Turbo Codes ที่กำหนดค่า G (Generator Metrix) = [111,101] และการถอดรหัสในรูปแบบ Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Viterbi Decoding และ Log-Map Decoding



รูปที่ 4.2 ผลการทดลองค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลแบบที่มีการเข้ารหัสในรูปแบบ Turbo Codes ที่กำหนดค่า G (Generator Metrix) = [111,101] และการถอดรหัสในรูปแบบ Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Viterbi Decoding และ Log-Map Decoding และที่มีการวนซ้ำในการถอดรหัสตั้งแต่ 1 ถึง 5 รอบ

จากรูปที่ 4.2 แสดงการเปรียบเทียบค่าเฉลี่ยอัตราผิดพลาดของบิตข้อมูลของการเข้ารหัสแบบ Turbo Codes โดย Recursive Systematic Convolutional ที่ทำการทดลองนั้นจะมีค่า G (Generator Metrix) เท่ากับ [1 1 1 , 1 0 1] และได้กำหนด Puncture ให้มีอัตราของข้อมูลเข้าต่อข้อมูลออกหรือ Code Rate เป็น ½ และได้กำหนดให้แต่ละเฟรมมี 270 บิตและได้มีการมอดูเลตสัญญาณแบบ BPSK และได้นำสัญญาณรบกวนแบบ AWGN (Additive White Gaussian Noise : AWGN) มาเป็นสัญญาณรบกวนในการทดลอง ซึ่งในการถอดรหัส Turbo Codes นั้นได้ทำการเปรียบเทียบเทคนิคระหว่าง Viterbi Decoding และ Log-Map Decoding ที่แต่ละจำนวนรอบในการวนซ้ำตั้งแต่ 1 ถึง 5 รอบ โดยผลที่ได้จากการเปรียบเทียบนั้นจะเห็นว่า เทคนิคแบบ Log-Map Decoding จะสามารถลดค่า BER ได้ดีกว่าเทคนิคแบบ Viterbi Decoding อยู่เล็กน้อย และจำนวนรอบในการวนซ้ำยิ่งมากก็จะช่วยลดค่า BER ได้มากตามไปด้วย แต่ยิ่งจำนวนวนรอบมากขึ้นก็จะใช้เวลาในการถอดรหัสมากขึ้นด้วย

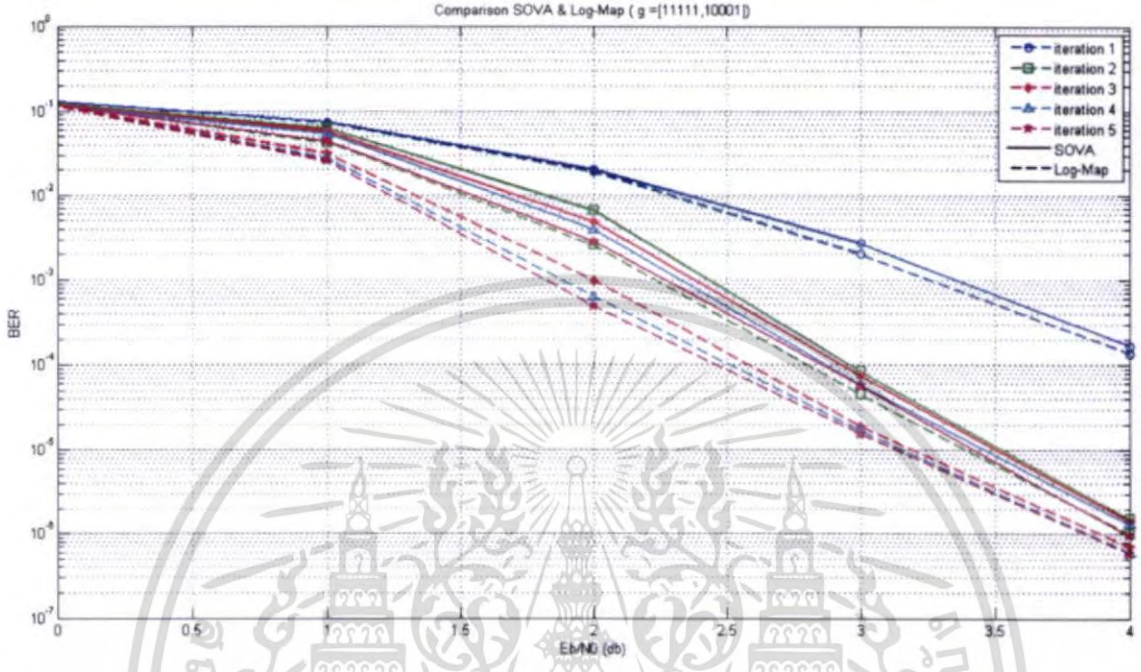
4.3 ผลค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลแบบที่มีการเข้ารหัสในรูปแบบ Turbo Codes ที่กำหนดค่า G (Generator Metrix) = [1101,1111] และการถอดรหัสในรูปแบบ Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Viterbi Decoding และ Log-Map Decoding



รูปที่ 4.3 ผลการทดลองค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลแบบที่มีการเข้ารหัสในรูปแบบ Turbo Codes ที่กำหนดค่า G (Generator Metrix) = [111,101] และการถอดรหัสในรูปแบบ Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Viterbi Decoding และ Log-Map Decoding และที่มีการวนซ้ำในการถอดรหัสตั้งแต่ 1 ถึง 5 รอบ

จากรูปที่ 4.3 แสดงการเปรียบเทียบค่าเฉลี่ยอัตราผิดพลาดของบิตข้อมูลของการเข้ารหัสแบบ Turbo Codes โดย Recursive Systematic Convolutional ที่ทำการทดลองนั้นจะมีค่า G (Generator Metrix) เท่ากับ [1101,1111] และได้กำหนด Puncture ให้มีอัตราของข้อมูลเข้าต่อข้อมูลออกหรือ Code Rate เป็น ½ และได้กำหนดให้แต่ละเฟรมมี 270 บิตและได้มีการมอดูเลตสัญญาณแบบ BPSK และได้นำสัญญาณรบกวนแบบ AWGN (Additive White Gaussian Noise : AWGN) มาเป็นสัญญาณรบกวนในการทดลอง ซึ่งในการถอดรหัส Turbo Codes นั้นได้ทำการเปรียบเทียบเทคนิคระหว่าง Viterbi Decoding และ Log-Map Decoding ที่แต่ละจำนวนรอบในการวนซ้ำตั้งแต่ 1 ถึง 5 รอบ โดยผลที่ได้จากการเปรียบเทียบนั้นจะเห็นว่า เทคนิคแบบ Log-Map Decoding จะสามารถลดค่า BER ได้ดีกว่าเทคนิคแบบ Viterbi Decoding อยู่เล็กน้อย และจำนวนรอบในการวนซ้ำยิ่งมากก็จะช่วยลดค่า BER ได้มากตามไปด้วย แต่ยิ่งจำนวนวนรอบมากขึ้นก็จะใช้เวลาในการถอดรหัสมากขึ้นด้วย

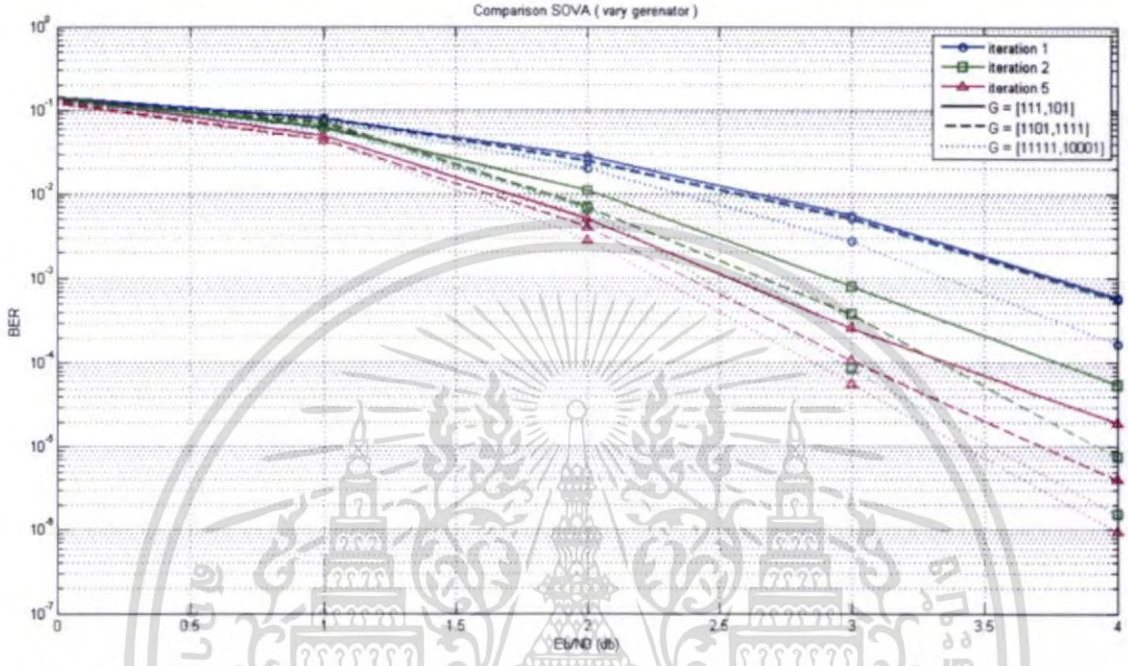
4.4 ผลค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลแบบที่มีการเข้ารหัสในรูปแบบ Turbo Codes ที่กำหนดค่า G (Generator Metrix) = [11111,10001] และการถอดรหัสในรูปแบบ Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Viterbi Decoding และ Log-Map Decoding



รูปที่ 4.4 ผลการทดลองค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลแบบที่มีการเข้ารหัสในรูปแบบ Turbo Codes ที่กำหนดค่า G (Generator Metrix) = [11111,10001] และการถอดรหัสในรูปแบบ Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Viterbi Decoding และ Log-Map Decoding และที่มีการวนซ้ำในการถอดรหัสตั้งแต่ 1 ถึง 5 รอบ

จากรูปที่ 4.4 แสดงการเปรียบเทียบค่าเฉลี่ยอัตราผิดพลาดของบิตข้อมูลของการเข้ารหัสแบบ Turbo Codes โดย Recursive Systematic Convolutional ที่ทำการทดลองนั้นจะมีค่า G (Generator Metrix) เท่ากับ [11111,10001] และได้กำหนด Puncture ให้มีอัตราของข้อมูลเข้าต่อข้อมูลออกหรือ Code Rate เป็น ½ และได้กำหนดให้แต่ละเฟรมมี 270 บิตและได้มีการมอดูเลตสัญญาณแบบ BPSK และได้นำสัญญาณรบกวนแบบ AWGN (Additive White Gaussian Noise : AWGN) มาเป็นสัญญาณรบกวนในการทดลอง ซึ่งในการถอดรหัส Turbo Codes นั้นได้ทำการเปรียบเทียบเทคนิคระหว่าง Viterbi Decoding และ Log-Map Decoding ที่แต่ละจำนวนรอบในการวนซ้ำตั้งแต่ 1 ถึง 5 รอบ โดยผลที่ได้จากการเปรียบเทียบนั้นจะเห็นว่า เทคนิคแบบ Log-Map Decoding จะสามารถลดค่า BER ได้ดีกว่าเทคนิคแบบ Viterbi Decoding อยู่เล็กน้อย และจำนวนรอบในการวนซ้ำยิ่งมากก็จะช่วยลดค่า BER ได้มากตามไปด้วย แต่ยิ่งจำนวนวนรอบมากขึ้นก็จะใช้เวลาในการถอดรหัสมากขึ้นด้วย

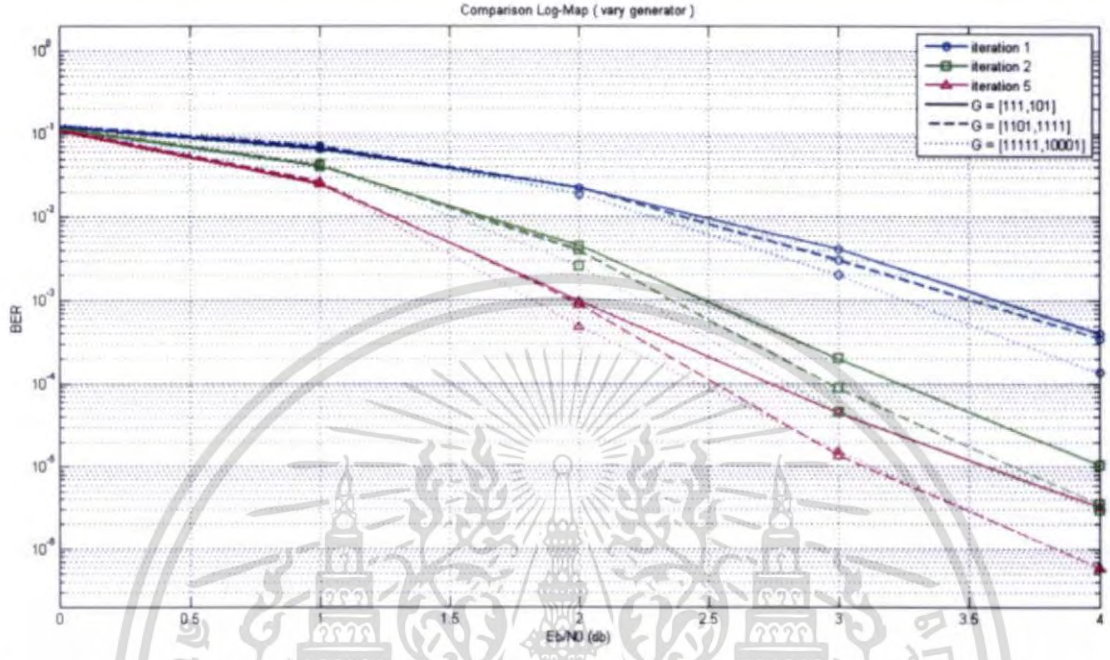
4.5 ผลค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลแบบที่มีการเข้ารหัสในรูปแบบ Turbo Codes ที่กำหนดค่า G (Generator Metrix) แตกต่างกัน และการถอดรหัสในรูปแบบ Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Viterbi Decoding



รูปที่ 4.5 ผลการทดลองค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลแบบที่มีการเข้ารหัสในรูปแบบ Turbo Codes ที่กำหนดค่า G (Generator Metrix) = [111,101], [1101,1111], [11111,10001] และการถอดรหัสในรูปแบบ Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Viterbi Decoding ที่มีการวนซ้ำในการถอดรหัส 1 , 2 และ 5 ตามลำดับ

จากรูป 4.5 แสดงการเปรียบเทียบค่าเฉลี่ยอัตราผิดพลาดของบิตข้อมูลของการเข้ารหัสแบบ Turbo Codes โดย Recursive Systematic Convolutional ที่ทำการทดลองนั้นได้ทำการกำหนดค่า G (Generator Metrix) เท่ากับ [111,101], [1101,1111], [11111,10001] เพื่อเปรียบเทียบประสิทธิภาพ และได้กำหนด punctureให้มีอัตราของข้อมูลเข้าต่อข้อมูลออกหรือ Code rate เป็น ½ และได้กำหนดให้แต่ละเฟรมมี 270 บิตและได้มีการมอดูเลตสัญญาณแบบ BPSK และได้นำสัญญาณรบกวนแบบ AWGN(Additive White Gaussian Noise : AWGN) มาเป็นสัญญาณรบกวนในการทดลอง และมีการถอดรหัส Turbo Codes โดยใช้เทคนิค Viterbi Decoding โดยมีจำนวนรอบในการวนซ้ำที่ 1 ,2 และ 5 ตามลำดับ ซึ่งค่า G (Generator Metrix) สามารถที่จะช่วยลดค่า BER ได้ลดได้เป็นอย่างมาก โดยถ้า G (Generator Metrix) มีความยาวมากก็จะช่วยลดค่า BER ได้มากแต่ก็จะใช้เวลาในการเข้ารหัสและถอดรหัสนานขึ้นตามไปด้วย

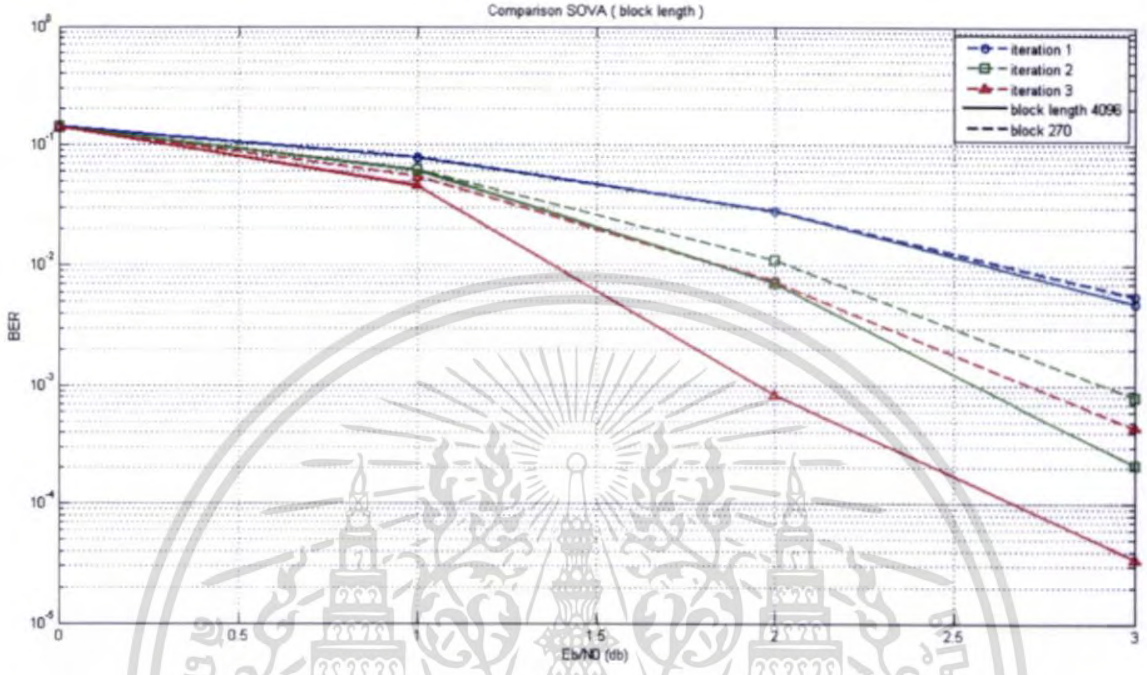
4.6 ผลค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลแบบที่มีการเข้ารหัสในรูปแบบ Turbo Codes ที่กำหนดค่า G (Generator Metrix) แตกต่างกัน และการถอดรหัสในรูปแบบ Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Log-Map Decoding



รูปที่ 4.6 ผลการทดลองค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลแบบที่มีการเข้ารหัสในรูปแบบ Turbo Codes ที่กำหนดค่า G (Generator Metrix) = [111,101], [1101,1111], [11111,10001] และการถอดรหัสในรูปแบบ Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Viterbi Decoding ที่มีการวนซ้ำในการถอดรหัส 1 , 2 และ 5 ตามลำดับ

จากรูป 4.6 แสดงการเปรียบเทียบค่าเฉลี่ยอัตราผิดพลาดของบิตข้อมูลของการเข้ารหัสแบบ Turbo Codes โดย Recursive Systematic Convolutional ที่ทำการทดลองนั้นได้ทำการกำหนดค่า G (Generator Metrix) เท่ากับ [111,101], [1101,1111], [11111,10001] เพื่อเปรียบเทียบประสิทธิภาพ และได้กำหนด Puncture ให้มีอัตราของข้อมูลเข้าต่อข้อมูลออกหรือ Code Rate เป็น ½ และได้กำหนดให้แต่ละเฟรมมี 270 บิตและได้มีการมอดูเลตสัญญาณแบบ BPSK และได้นำสัญญาณรบกวนแบบ AWGN (Additive White Gaussian Noise : AWGN) มาเป็นสัญญาณรบกวนในการทดลอง และมีการถอดรหัส Turbo Codes โดยใช้เทคนิค Log-Map Decoding โดยมีจำนวนรอบในการวนซ้ำที่ 1 ,2 และ 5 ตามลำดับ ซึ่งค่า G (Generator Metrix) สามารถที่จะช่วยลดค่า BER ได้ลดได้เป็นอย่างมาก โดยถ้า G (Generator Metrix) มีความยาวมากก็จะช่วยลดค่า BER ได้มากแต่ก็จะใช้เวลาในการเข้ารหัสและถอดรหัสนานขึ้นตามไปด้วย

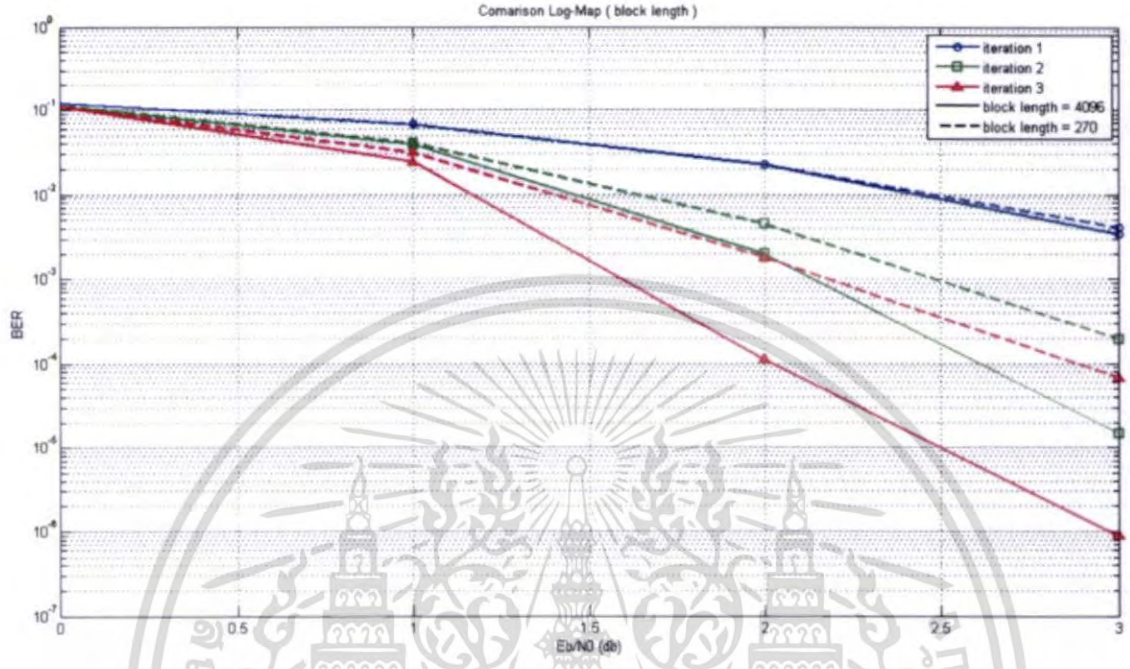
4.7 ผลค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลแบบที่มีการเข้ารหัสในรูปแบบ Turbo Codes ที่กำหนดค่าขนาดจำนวนบิตต่อหนึ่งเฟรมแตกต่างกัน และการถอดรหัสในรูปแบบ Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Viterbi Decoding



รูปที่ 4.7 ผลการทดลองค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลแบบที่มีการเข้ารหัสในรูปแบบ Turbo Codes ที่กำหนดค่าขนาดจำนวนบิตต่อหนึ่งเฟรมเท่ากับ 270 บิตและ 4096 บิต และถอดรหัสในรูปแบบ Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Viterbi Decoding ที่มีการวนซ้ำในการถอดรหัส 1, 2 และ 3 ตามลำดับ

จากรูป 4.7 แสดงการเปรียบเทียบค่าเฉลี่ยอัตราผิดพลาดของบิตข้อมูลของการเข้ารหัสแบบ Turbo Codes โดย Recursive Systematic Convolutional ที่ทำการทดลองนั้นได้ทำการกำหนดค่า G (Generator Metrix) เท่ากับ [111,101] และได้กำหนด Puncture ให้มีอัตราของข้อมูลเข้าต่อข้อมูลออกหรือ Code Rate เป็น ½ และได้กำหนดให้แต่ละเฟรมมี 270 บิตและ 4096 บิตตามลำดับเพื่อเปรียบเทียบประสิทธิภาพ และได้มีการมอดูเลตสัญญาณแบบ BPSK และได้นำสัญญาณรบกวนแบบ AWGN (Additive White Gaussian Noise : AWGN) มาเป็นสัญญาณรบกวนในการทดลอง และมีการถอดรหัส Turbo Codes โดยใช้เทคนิค Viterbi Decoding โดยมีจำนวนรอบในการวนซ้ำที่ 1, 2 และ 3 ตามลำดับ ซึ่งขนาดของจำนวนบิตต่อหนึ่งเฟรมที่มีขนาดใหญ่จะสามารถช่วยลดค่า BER ได้มากกว่าขนาดของจำนวนบิตต่อหนึ่งเฟรมขนาดเล็ก

4.8 ผลค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลแบบที่มีการเข้ารหัสในรูปแบบ Turbo Codes ที่กำหนดค่าขนาดจำนวนบิตต่อหนึ่งเฟรมแตกต่างกัน และการถอดรหัสในรูปแบบ Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Log-Map Decoding



รูปที่ 4.8 ผลการทดลองค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลแบบที่มีการเข้ารหัสในรูปแบบ Turbo Codes ที่กำหนดค่าขนาดจำนวนบิตต่อหนึ่งเฟรมเท่ากับ 270 บิตและ 4096 บิต และถอดรหัสในรูปแบบ Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Log-Map Decoding ที่มีการวนซ้ำในการถอดรหัส 1, 2 และ 3 ตามลำดับ

จากรูป 4.8 แสดงการเปรียบเทียบค่าเฉลี่ยอัตราผิดพลาดของบิตข้อมูลของการเข้ารหัสแบบ Turbo Codes โดย Recursive Systematic Convolutional ที่ทำการทดลองนั้นได้ทำการกำหนดค่า G (Generator Metrix) เท่ากับ [111,101] และได้กำหนด Puncture ให้มีอัตราของข้อมูลเข้าต่อข้อมูลออกหรือ Code Rate เป็น ½ และได้กำหนดให้แต่ละเฟรมมี 270 บิตและ 4096 บิตตามลำดับเพื่อเปรียบเทียบประสิทธิภาพ และได้มีการมอดูเลตสัญญาณแบบ BPSK และได้นำสัญญาณรบกวนแบบ AWGN (Additive White Gaussian Noise : AWGN) มาเป็นสัญญาณรบกวนในการทดลอง และมีการถอดรหัส Turbo Codes โดยใช้เทคนิค Log-Map Decoding โดยมีจำนวนรอบในการวนซ้ำที่ 1, 2 และ 3 ตามลำดับ ซึ่งขนาดของจำนวนบิตต่อหนึ่งเฟรมที่มีขนาดใหญ่จะสามารถช่วยลดค่า BER ได้มากกว่าขนาดของจำนวนบิตต่อหนึ่งเฟรมขนาดเล็ก

บทที่ 5
บทวิจารณ์และบทสรุป

5.1 สรุปผลการทดลอง

การเข้ารหัสแบบ Convolutional Code และมีการถอดรหัสแบบ Viterbi Decoding, MAP Decoding และ Log-Map Decoding แบบพื้นฐาน กับ การเข้ารหัสในรูปแบบ Turbo Codes และถอดรหัสด้วย Turbo Codes โดยใช้เทคนิคในการถอดรหัสแบบ Viterbi Decoding และ Log-Map Decoding โดยใช้โปรแกรมแมทแลบ ในบทที่ 4 สามารถที่จะสรุปได้ว่า

จากผลการทดลองรูปที่ 4.1 แสดงให้เห็นว่าการส่งข้อมูลแบบมีการเข้ารหัสนั้นจะช่วยลดค่า BER ได้เป็นอย่างมากซึ่งเมื่อในภาวะที่ส่งด้วยค่า SNR มากๆจะเห็นว่าค่า BER ที่เกิดขึ้นนั้นจะมีค่าน้อยมากๆ แต่การส่งข้อมูลแบบที่ไม่มีการเข้ารหัสนั้นจะเห็นว่าค่า BER ที่เกิดขึ้นนั้นจะมีค่ามากตลอดทุกๆในแต่ละ SNR ซึ่งจากผลการทดลองจะสามารถสรุปได้ว่า การส่งข้อมูลแบบที่มีการเข้ารหัสนั้นจะช่วยลดค่า BER ได้เป็นอย่างมากและในการถอดรหัสข้อมูลแบบ Log-Map Decoding, Map Decoding และ Viterbi Decoding จะมีประสิทธิภาพในการลดค่า BER ได้มากไปน้อยตามลำดับ

จากผลการทดลองรูปที่ 4.2, 4.3 และ 4.4 แสดงให้เห็นว่าการใช้เทคนิคในการเข้ารหัสและถอดรหัสแบบ Turbo Codes เข้ามาช่วยจะสามารถลดค่า BER ได้มากกว่าการเข้ารหัส Convolutional Code และถอดรหัสแบบ Viterbi Decoding กับ Log-Map Decoding แบบพื้นฐาน โดยการวนซ้ำในการถอดรหัสนั้นจะสามารถช่วยแก้ไขบิตข้อมูลที่ผิดพลาดได้ดียิ่งขึ้น ซึ่งสามารถสังเกตได้จากค่า BER ที่ลดต่ำลดเมื่อเพิ่มจำนวนรอบในการวนซ้ำของการถอดรหัส

จากผลการทดลองรูปที่ 4.5 และ 4.6 แสดงให้เห็นว่าในการเข้ารหัสแบบ Convolutional Code นั้นขนาดของ Generator Metrix นั้นจะมีส่วนต่อประสิทธิภาพในการส่งข้อมูลเพราะยิ่งขนาดของ Generator Metrix ขนาดใหญ่ก็จะทำให้ คำรหัส ที่ถูกส่งออกไปนั้นมีความหลากหลายและมีความยาวมากยิ่งขึ้นจึงทำให้ข้อมูลเกิดความน่าจะเป็นในการผิดพลาดลดน้อยลง ซึ่งสามารถสังเกตได้จากค่า BER ที่ลดต่ำลงเมื่อเพิ่มขนาดของ Generator Metrix

จากผลการทดลองรูปที่ 4.7 และ 4.8 แสดงให้เห็นว่าในการเข้ารหัสแบบ Convolutional Code นั้นขนาดของจำนวนบิตต่อหนึ่งเฟรม นั้นจะมีส่วนต่อประสิทธิภาพในการส่งข้อมูลเพราะในการถอดรหัสแบบ Viterbi Decoding และ Log-Map Decoding นั้นจะทำการถอดรหัสเป็นเฟรมๆ โดยถ้าในหนึ่งเฟรมมีขนาดใหญ่ก็จะช่วยให้เกิดความผิดพลาดในการถอดรหัสลดน้อยลง ซึ่งสามารถสังเกตได้จากค่า BER ที่ลดต่ำลงเมื่อเพิ่มขนาดของจำนวนบิตต่อหนึ่งเฟรม

5.2 ปัญหาที่พบในระหว่างการดำเนินโครงการ

ในการดำเนินโครงงานจะต้องมีขั้นตอนในการเก็บข้อมูลของค่าเฉลี่ยอัตราความผิดพลาดของบิตข้อมูลซึ่งจะเป็นการเฉลี่ยค่าของแต่เฟรมจำนวนมาก และต้องทำในแต่ละอัตราส่วนสัญญาณต่อสัญญาณรบกวนในสมมุติฐานที่ต่างกัน ขั้นตอนนี้จึงเป็นขั้นตอนที่ใช้เวลานานมากในขั้นตอนการดำเนินโครงงาน เป็นผลทำให้การดำเนินงานเป็นไปอย่างล่าช้า อีกทั้งถ้าหากการกำหนดสมมุติฐานต่างๆในการจำลองระบบเกิดความผิดพลาดจะทำให้ยิ่งทำให้การดำเนินงานล่าช้ามากขึ้นไปอีก เพราะ ต้องทำการเก็บข้อมูลใหม่อีกครั้งเพื่อความถูกต้องของข้อมูล และข้อมูลเนื้อหาที่เกี่ยวข้องกับการเข้ารหัส-ถอดรหัสแบบ Turbo Codes ที่เป็นภาษาไทยนั้นยังมีให้ศึกษาน้อยอยู่ จึงทำให้การศึกษาในช่วงแรกของการดำเนินโครงงานเป็นไปได้อย่างล่าช้าอีกทั้งยังต้องเริ่มต้นศึกษาวิธีการใช้โปรแกรมแมทแลบ อย่างละเอียดเพื่อเป็นพื้นฐานในการจำลองการเข้ารหัสและถอดรหัสข้อมูล

5.3 แนวทางการแก้ปัญหาและพัฒนา

ในการแก้ปัญหาที่กล่าวมาข้างต้นได้นำโปรแกรมที่เขียนขึ้นไปทำการประมวลผลที่สำนักวิจัยและบริการคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังโดยการพัฒนาต่อไปจำเป็นต้องพัฒนาการเขียนโปรแกรมในการจำลองการทำงานของระบบให้มีประสิทธิภาพมากขึ้นเพื่อที่จะทำให้สามารถเก็บข้อมูลมีความรวดเร็วมากยิ่งขึ้น และในโครงงานได้จำลองช่องการสื่อสารแบบส่ง 1 สายอากาศและรับ 1 สายอากาศ (SISO) ซึ่งในการพัฒนาต่อไปควรจะทำการจำลองช่องการสื่อสารแบบส่งหลายสายอากาศและรับหลายสายอากาศ (MIMO) และเพิ่มช่องสัญญาณแบบที่มีการลดทอน (Fading Channel) เพื่อที่จะได้นำไปประยุกต์ใช้ได้จริงในระบบการสื่อสารแบบไร้สาย

บรรณานุกรม

Branka Vucetic, Jinhong Yuan. 2000. **TURBO CODES.** The University of Sydney, Sydney Australia.

G.D. Forney, Jr. **"The Viterbi algorithm"**,Proc. IEEE, Vol.61, No. 3, pp.268-278, March 1973

DSL Lab (Digital System Laboratory), **www.kmitl.ac.th/dslabs**, E12-1106, Department of information, Faculty of Engineering, King Mongkut's Institute of Technology Ladkrabang.

John G Proakis. 2001. **Digital Communications.** Fourth Edition. McGraw-Hill International Edition.

John Litva, and Titus Kwok-Yeung Lo. 1996. **Digital beamforming in wireless communications.** Artech House, Inc.

Theodore S. Rappaport. 2002. ***Wireless Communication: Principles & Practice.*** Second Edition. New Jersey. Prentice-Hall, Inc.

A.J. Viterbi, **"Error bounds for convolutional codes and an asymptotically optimum decoding algorithm"**, IEEE Trans. Inform Theory, Vol.IT-13, No.2, pp.260-269, April 1967