การใช้สถิติในการสร้างตารางแจงข้อความสำหรับระบบแจงข้อความภาษาไทย

อรสา ขันเอียด[1],กนกอร ตระกูลทวีคูณ [2],เทพชัย ทรัพย์นิธิ[2], พรฤดี เนติโสภากุล[1]

[1]คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง กรุงเทพฯ

[2]ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ

Emails: a_cs24@hotmail.com, kanokorn.trakultaweekoon@nectec.or.th ,thepchai.supnithi@nectec.or.th ,ponrudee@it.kmitl.ac.th

บทคัดย่อ

บทความฉบับนี้ได้นำเสนอแนวทางการใช้สถิติภายในขั้นตอนการสร้างตารางแจงข้อความสำหรับระบบแจงข้อความภาษาไทย โดยคำนวณค่าความน่าเป็นของการเกิดลำดับสถานะให้กับทุกๆ สถานะของการแจงข้อความ เพื่อนำมาใช้ตัดสินใจเลือกเส้นทางที่มีความน่าจะเป็นสูงที่สุดในขณะที่แจงข้อความ ซึ่งจะช่วยลดขนาดของตารางแจงข้อความ และลดระยะเวลาในการแจงข้อความให้เร็วยิ่งขึ้น รวมทั้งช่วยลดภาระงานในการตรวจต้นไม้ไวยากรณ์ให้กับนักภาษาศาสตร์อีกด้วย

คำสำคัญ — การแจงข้อความ ; การแจงข้อความแบบ PGLR ; การใช้สถิติร่วมกับการแจงข้อความ

1. บทนำ

การประมวลผลภาษาธรรมชาติ (Natural Language Processing) เป็นศาสตร์แขนงหนึ่งด้านการทำให้คอมพิวเตอร์เข้าใจภาษามนุษย์หรือภาษาธรรมชาติ ซึ่งในปัจจุบันได้มีการนำเรื่องของการประมวลผลภาษาธรรมชาติมาประยุกต์ใช้ในงานด้านต่างๆ เช่น การสืบค้นข้อมูล (Information Retrieval) , การแปลภาษาด้วยเครื่องคอมพิวเตอร์ (Machine Translation) , การสรุปใจความสำคัญจากเอกสาร (Text Summarization) เป็นต้น ซึ่งการประมวลผลหรือทำให้คอมพิวเตอร์เข้าใจภาษามนุษย์นั้นเป็นเรื่องที่ยาก เนื่องจากภาษามนุษย์เป็นภาษาที่มีความหลากหลายและซับซ้อน รวมทั้งมีความกำกวมของภาษาซึ่งคำแต่ละคำอาจจะตีความได้แตกต่างกันถ้าอยู่ในประโยคที่มีบริบทแตกต่างกัน ตัวอย่างเช่น “เขาใช้กำลังกับเด็ก” ในที่นี้ “กำลัง” ทำหน้าที่เป็นนาม ส่วนประโยค “เขากำลังกินข้าว” ทำหน้าที่เป็นกริยาช่วย

ดังนั้นการแจงข้อความ (Parsing) จึงเป็นสิ่งสำคัญพื้นฐานสำหรับการประมวลผลภาษาธรรมชาติในการวิเคราะห์เชิงโครงสร้างวากยสัมพันธ์ (Syntactic Analysis) ซึ่งเป็นการตรวจสอบโครงสร้างทางไวยากรณ์ของกลุ่มคำชนิดต่างๆ ที่รวมกันเป็นวลีและประโยค การแจงข้อความภาษาไทยนั้นมีความกำกวมสูง เนื่องจากความกำกวมของภาษาซึ่งคำในภาษาไทยหลายคำไม่สามารถกำหนดหน้าที่ของคำตายตัวลงไปได้ เช่น “ขัน” เป็นได้ 4 ความหมายคือ 1)ขำขัน 2)ภาชนะ 3)การร้องของไก่ และ 4)ทำให้แน่น ซึ่งต้องอาศัยบริบทรอบข้างเข้าช่วยในการพิจารณา จึงทำให้ผลลัพธ์ของการแจงข้อความหรือต้นไม้ไวยากรณ์ (Derivation Tree) ของข้อความนั้นมีจำนวนมาก และหากข้อความที่รับเข้ามามีความยาวมากจำนวนต้นไม้ไวยากรณ์ที่ได้ก็จะหลากหลายมากขึ้นตามลำดับ แต่ต้นไม้ไวยากรณ์ที่เหมาะสมกับบริบทนั้นๆ มีเพียงต้นเดียว ซึ่งทำให้นักภาษาศาสตร์ใช้เวลานานในการตรวจสอบและเลือกต้นไม้ไวยากรณ์ที่เหมาะสมจากจำนวนต้นไม้ไวยากรณ์ทั้งหมดที่ได้จากการแจงข้อความ ดังรูปที่ 1 ตัวอย่างข้อความภาษาไทย [5] ประกอบด้วย 25 คำ ซึ่งมี 9 คำที่เป็นคำไวยากรณ์ เมื่อเข้าสู่กระบวนการแจงข้อความจะได้ต้นไม้ไวยากรณ์เป็นจำนวนมากถึง 1,469 ต้น เนื่องจากโครงสร้างภาษาไทยมีความคลุมเครือ ทำให้แต่ละคำอาจมีหลายหน้าที่รวมถึงไวยากรณ์และความหมาย เช่นคำว่า “ที่” สามารถเป็นคำนาม คำบุพบท และคำคุณวิเศษณ์ ส่วนคำว่า “คน” สามารถหมายถึงบุคคล ลักษณะนามของคน และกริยา “คน” และคำว่า “กำลัง” เป็นคำกริยาช่วยบอกการณ์ลักษณะ (aspect) และยังมีความหมายเป็นคำนามได้อีกด้วย จากปัญหาดังกล่าวจึงมีการนำสถิติ (probabilistic) เข้ามาช่วยในการแจงข้อความ เพื่อลดจำนวนของต้นไม้ไวยากรณ์ให้เหลือเพียงต้นไม้ไวยากรณ์ที่มีความน่าจะเป็นที่เหมาะสมสูงสุด แต่ส่วนใหญ่จะนำค่าสถิติมาใช้ในขั้นตอนหลังจากที่ได้ต้นไม้ไวยากรณ์ทั้งหมดจากการแจงข้อความแล้ว จึงจะคัดเลือกต้นไม้ไวยากรณ์ด้วยค่าสถิติ

สำหรับ|การ|วาง|กำลัง|ของ|คน|เสื้อ|แดง||ได้|มี|การ|วาง|บังเกอร์|รอบ|พื้นที่|ชุมนุม|เอา|น้ำมัน|ราด||รวมทั้ง|ยาง|รถยนต์|ที่|เสื่อมสภาพ|แล้ว

รูปที่ 1 ตัวอย่างข้อความภาษาไทย

จากที่กล่าวมาข้างต้นการนำสถิติเข้ามาประยุกต์ใช้กับการแจงข้อความสามารถลดจำนวนของต้นไม้ไวยากรณ์ที่มีความน่าจะเป็นของโครงสร้างไวยากรณ์ต่ำได้ ดังนั้นเราจึงมีแนวคิดในการนำสถิติมาประยุกต์ใช้กับการแจงข้อความ โดยนำสถิติมาใช้ในขั้นตอนการสร้างตารางแจงข้อความ (Parsing Table) เพื่อคำนวณค่าความน่าจะเป็นให้กับทุกๆสถานะ (State) ของการแจงข้อความ ซึ่งจะเลือกเส้นทาง (Derivation Path) ที่มีความน่าจะเป็นสูงในขณะที่แจงข้อความ ซึ่งจะช่วยลดขนาดของตารางแจงข้อความ และลดระยะเวลาในการแจงข้อความให้เร็วยิ่งขึ้น

2.งานและทฤษฎีที่เกี่ยวข้อง

จากการศึกษางานวิจัยทางด้านการแจงข้อความนั้นมีหลายวิธีด้วยกันแต่ละวิธีมีประสิทธิภาพต่างกัน การแจกแจงข้อความนั้นมีส่วนที่เกี่ยวข้องหลักๆ คือ ไวยากรณ์ที่ใช้แจกแจงข้อความ หมายถึง กฎเกณฑ์ที่ใช้อธิบายการผูกข้อความในแต่ละภาษานั้นซึ่งเรียกว่าไวยากรณ์ (Grammar) การรู้จำหรือการสร้างข้อความด้วยคอมพิวเตอร์จำเป็นต้องมีกฎไวยากรณ์เป็นฐานความรู้ทางภาษาที่เรียกว่าสูตรไวยากรณ์ [6]

ส่วนการแจกแจงข้อความนั้นเป็นวัตถุประสงค์หนึ่งของการวิเคราะห์หาความสัมพันธ์ของคำในประโยค เพื่อหาโครงสร้างลำดับชั้น ดังนั้นการวิเคราะห์โครงสร้างประโยค หมายถึง การวิเคราะห์หาความสัมพันธ์ของส่วนประกอบในประโยคว่าคำใดทำหน้าที่อะไร โดยทั่วไปโครงสร้างประโยคจะแทนด้วยรูปต้นไม้ที่เป็นลำดับชั้น กระบวนการที่ใช้แจกแจงข้อความเรียกว่า พาสซิ่ง (parsing) และโปรแกรมที่ใช้แจกแจงข้อความจะเรียกว่า พาสเซอร์ (parser) ซึ่งโครงสร้างของข้อความนั้นจะส่งผลต่อการแปลความหมายของข้อความ ดังนั้นการแจกแจงข้อความ คือ การนำไวยากรณ์โครงสร้างวลีมาตรวจสอบความถูกต้องทางไวยากรณ์ของข้อความหรือ อาจจะเรียกว่าวิธีการบอกความสัมพันธ์ของคำในข้อความ[8] โดยทั่วไปมี 2 ลักษณะด้วยกันลักษณะแรกคือ แบบบนลงล่าง (Top-down) ลักษณะของการแจกแจงข้อความแบบบนลงล่างจะเริ่มจากสัญลักษณ์เริ่มต้น จากนั้นนำกฎต่าง ๆ มาใช้กระจายโครงสร้างไปข้างหน้าจนกระทั่งได้สัญลักษณ์ที่สัมพันธ์กับองค์ประกอบต่าง ๆ ของข้อความที่นำมาแจกแจง และแบบล่างขึ้นบน (Bottom up) เป็นลักษณะของการแจกแจงข้อความแบบล่างขึ้นบน จะเริ่มจากข้อความที่นำมาแจกแจง จากนั้นนำกฎต่าง ๆ มาใช้ย้อนหลังจนกระทั่งได้โครงสร้างต้นไม้ที่มีโหนดรากเป็นสัญลักษณ์เริ่มต้น

เทคนิคในการแจกแจงข้อความนั้นมีหลายวิธีด้วยกัน แต่ละวิธีให้ผลการทำงานที่มีประสิทธิภาพต่างกัน การแจงข้อความที่เกี่ยวข้องกับบทความนี้คือ

2.1 GLR Parser

ใน GLR Parser หรือ (Generalized Left-to-right Rightmost Derivation parser) ถูกเสนอขึ้นโดย Masaru Tomita [6] ในปี 1984 กระบวนการทำงานทั่วไปมีความคล้ายคลึงกับ LR Parser แต่มีความพิเศษตรงที่วิเคราะห์เป็นแบบขนาน กล่าวคือจะมีกราฟมากกว่าหนึ่งชั้นในสายการทำงาน ในกระบวนการทำงานจะวิเคราะห์แจกแจงข้อความโดยมองจากซ้ายไปขวา เช่น “ A dog chases the cat.” ตัวอย่างข้อความจากมนุษย์เราจะมองจากซ้ายไปขวาโดยอัตโนมัติ “A dog” ทำหน้าที่เป็นประธาน “the cat” ทำหน้าที่เป็นกรรม ส่วน “chases the cat” เป็นภาคแสดงของข้อความ เมื่อนำส่วนต่างๆมารวมกันจะกลายเป็นข้อความที่สมบูรณ์ วิธีการทำงานของ GLR Parser จะใช้โครงสร้างสแต็กผ่านตัวดำเนินการต่างๆ [1]

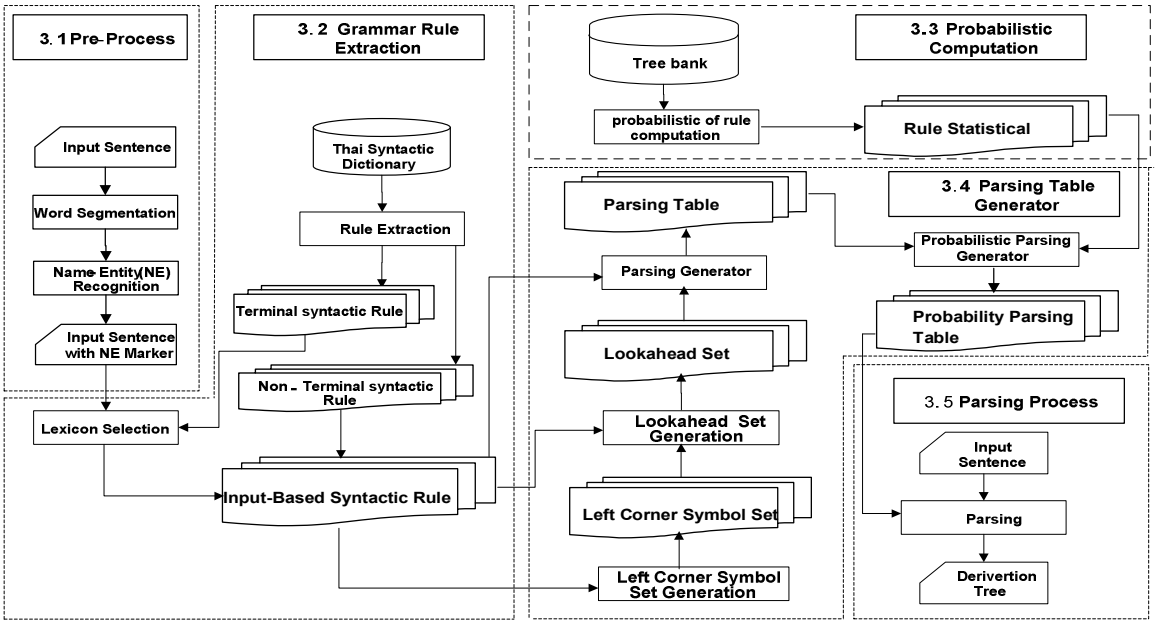
การทำงานของ parser เริ่มจากรับข้อมูลเข้ามาแล้วทำการวิเคราะห์คำ โดยจะพิจารณาเป็นระดับ POS (Part of speech) คือ การแบ่งชนิดของคำตามหลักไวยากรณ์เพื่ออธิบายหน้าที่และความสัมพันธ์ของคำต่างๆในข้อความ จากนั้นเข้ากระบวนการวิเคราะห์คำทีละหน่วยโดยพิจารณาร่วมกับตารางแจงข้อความที่สร้างได้ ภายในตารางประกอบด้วยตัวดำเนินการต่างๆ เมื่อ Input ที่รับเข้ามาอยู่ในสถานะใดๆก็ให้ทำตามตัวดำเนินนั้นๆ

เมื่อผ่านกระบวนอัลกอริทึมการแจงข้อความแล้ว ข้อความที่ไม่สามารถผ่านการแจงต้นไม้ได้เกิดจาก อาจไม่พบคำในพจนานุกรมสามารถแก้ไขโดยการเพิ่มคำลงพจนานุกรมหรือข้อความไม่เป็นไปตามกฎไวยากรณ์ ส่วน ข้อความใดที่ให้ผลการวิเคราะห์เป็นโครงสร้างหลายแบบยังต้องอาศัยนักภาษาศาสตร์วิเคราะห์และเลือกโครงสร้างต้นไม้ที่ซึ่งต้องใช้เวลานานและหากข้อความที่ต้องการแจงนี้มากก็จะใช้เวลามากตามลำดับ

2.2 PGLR Parser

PGLR Parser หรือ (Probabilistic Generalized Left-to-right Rightmost Derivation parser)เป็นวิธีการแจงข้อความโดยพัฒนาต่อเนื่องมาจากการแจงแบบ GLR [1] ซึ่งจะใช้การคำนวณทางสถิติเข้ามาร่วมพิจารณากับการแจงข้อความแบบ GLR ดังที่กล่าวไปแล้วใน 2.1 การทำงานจะดำเนินการคำนวณความน่าจะเป็นของกฎที่ใช้ในการแจงข้อความ เมื่อทำการแจงทุกทางที่เป็นไปได้ตามวิธีของ GLR Parser หลังจากนั้นจะทำการเลือกต้นไม้โดยวิเคราะห์จากความน่าจะเป็นของกฎที่ใช้แจงของต้นไม้ที่ได้แต่ละต้น ความน่าจะเป็นของต้นไม้แต่ละต้นหาได้จากผลรวมของความน่าจะเป็นของกฎไวยากรณ์ที่ใช้ไปในต้นไม้นั้นๆ สุดท้ายจะทำการเลือกต้นไม้ที่มีค่าความน่าจะเป็นสูงสุดเป็นผลลัพธ์

นอกจากนี้ยังมีวิธีการทำงาน PGLR ที่อาศัยเงื่อนไขการทำงานตัวดำเนินการ 2 ชนิดที่แตกต่างกันในตาราง LR Table คือ shift และ reduce โดยสถานะหลังจากการ reduce คือการยึดอินพุตเดิมไว้ในสถานะเดิม ในขณะที่ตัวดำเนินการ shift จะอ่านอินพุตใหม่เข้ามา ทำการจัดกลุ่มสถานะเป็น 2 แบบคือกลุ่มสถานะของตัวดำเนินการ shift อีกกลุ่มคือกลุ่มสถานะของตัวดำเนินการ reduce และประมาณความน่าจะเป็นของกลุ่มสถานะตัวดำเนินการที่ต่างกันเหล่านี้[3] โดยสามารถประมาณความน่าจะเป็นของตัวดำเนินการปัจจุบันได้จากสถานะส่วนบนสุดของสแต็กปัจจุบันแทนที่จะเป็นทั้งสแต็ก การใส่ความน่าจะเป็นให้กับตาราง LR Table นั้นจะใส่จากการแยกกลุ่มสถานะตามสถานะตัวดำเนินการ ทั้งสถานะตัวดำเนินการ shift และ reduce สถานะสังเกตได้จากคุณสมบัติกลุ่มของสถานะสำหรับตาราง LR Table ที่ตรงกันเนื่องจากสถานะใดๆสามารถเชื่อมต่อไวยากรณ์ได้ชนิดเดียวเท่านั้น สถานะในกลุ่ม reduce การทำงานจะดูจากการเปลี่ยนแปลงในส่วน goto ในตาราง LR Table อีกส่วนจะเป็นส่วนของกลุ่มสถานะ Shift เมื่อใช้การแจงแบบ GLR ทดลองกับคลังคำเพื่อสกัดลำดับการทำงานของตัวดำเนินการจะได้ความถี่การทำงานแต่ละตัวดำเนินการและเพิ่มส่วนของการนับแต่ละตัวดำเนินการที่ปรากฏในตารางเพิ่มเติม สุดท้ายแล้วความน่าจะเป็นแต่ละตัวดำเนินการ จะคำนวณตามกลุ่มสถานะตัวดำเนินการ ผลการเพิ่มความน่าจะเป็นในแต่ละตัวดำเนินการ ในตาราง parsing นั้น กลุ่มสถานะ shift จะให้ผลรวมความ



รูปที่ 2. แสดงภาพรวมการทำงานของระบบแจงข้อความแบบ PGLR

น่าจะเป็นในสถานะนั้นๆเท่ากับ 1 แต่บางส่วนของกลุ่มสถานะ reduce ความน่าจะเป็นของตัวดำเนินการ ในแต่ละ element และสัญลักษณ์อินพุต มีผลรวมเป็น 1

3. ภาพรวมการทำงานของระบบ

การทำงานของระบบจะทำงานโดยนำ GLR ที่มีมาพัฒนาเพิ่มเติมโดยนำสถิติความน่าจะเป็นมาใช้ภายในขั้นตอนการสร้างตารางแจงข้อความเพื่อนำมาใช้ตัดสินใจเลือกเส้นทางที่มีความน่าจะเป็นสูงที่สุดในขณะที่แจงข้อความ นอกจากนี้ได้พัฒนาส่วนที่ช่วยสนับสนุนและรองรับสำหรับภาษาไทยโดยมีโครงสร้างการทำงานของระบบดังรูปที่ 2 แบ่งออกเป็น 5 ส่วนดังนี้

3.1 Pre-Process

ขั้นตอนนี้จะรับข้อความดิบผ่านกระบวนการแบ่งคำและส่งคำเหล่านี้ไปกำกับนิพจน์ระบุนาม (กลุ่มนิพจน์ที่ใช้เรียกหรือระบุถึงสิ่งใดๆ ที่เป็นชื่อเฉพาะ เช่น ชื่อบุคคล ชื่อองกรค์ ชื่อสถานที่ และวัน เวลา เป็นต้น) ซึ่งนิพจน์เหล่านี้มักเกิดปัญหาในการประมวลผลข้อความคืออาจจะไม่เจอในพจนานุกรม

3.2 Grammar Rule Extraction

กระบวนการนี้แบ่งออกเป็น 2 ขั้นตอนย่อยได้แก่ Rule Extraction และ Lexicon Selection มีรูปแบบการทำงานดังนี้

3.2.1 Rule Extraction

กระบวนการนี้ทำหน้าที่ประมวลผลพจนานุกรม เพื่อแยกกฎไวยากรณ์ออกเป็น 2 ส่วน คือรายการข้อมูลคู่คำ (Terminal Rule) และ รายการกฎไวยากรณ์สำหรับการแจงไวยากรณ์ (Non-Terminal Rule) โดยทำการใส่กฎใดๆก็ได้ เช่น CG(Categorial Grammar) , POS ซึ่งในงานนี้จะอ้างอิงจาก CG [6] เป็นหลัก

3.2.2 Lexicon Selection

ขั้นตอนนี้จะนำข้อความที่ถูกกำกับนิพจน์เข้าสู่ฟังก์ชันเลือกกฎไวยากรณ์ ระบบจะเลือกกฎไวยากรณ์ที่เหมาะสมกับข้อความ แล้วจะนำไปใช้สร้างตารางแจงข้อความต่อไป

3.3 Probabilistic Computation

ในโมดูลนี้จะทำการเก็บค่าสถิติเพื่อใช้เป็นค่าสถิติตั้งต้น ซึ่งกระบวนการ probabilistic of rule computation จะทำการคำนวณค่าสถิติสำหรับแต่ละกฎจาก Treebank [5] ที่ผ่านการคัดเลือกแล้วว่ามีโครงสร้างที่ถูกต้อง จะนับการเกิดของกฎนั้นๆ เทียบกับกฎอื่นๆที่มีวลีเป็นชนิดเดียวกัน [8]โดยใช้สมการ

$$\widetilde{P}(A \to \alpha) = \frac{C\xi(A \to \alpha)}{\sum_{\alpha' \in (N \cup T)*} C\xi(A \to \alpha')} \qquad (1)$$

โดย

$\widetilde{P}$ ค่าความน่าจะเป็นโดยประมาณ

$C\xi$ ฟังก์ชันในการนับจำนวนกฎที่เกิดขึ้นใน โครงสร้างต้นไม้ที่อยู่ในคลังข้อความ $\xi(Rule \mapsto N+)$

N เป็นเซตของสัญลักษณ์ไม่สิ้นสุด(non-terminal)

T เป็นเซตของสัญลักษณ์สิ้นสุด (terminal)

ค่าสถิติที่คำนวณได้ดังกล่าวเก็บไว้ใน ไฟล์ rule statistical เพื่อนำไปใช้ในขั้นตอนการ Parsing

3.4 Parsing Table Generator

สำหรับการสร้างตารางแจงข้อความเริ่มจากข้อความนำเข้าซึ่งประกอบด้วยกฎไวยากรณ์สำหรับแจงข้อความ ซึ่งได้มาจากขั้นตอนของ Grammar Rule Extraction จะได้กฎไวยากรณ์สำหรับการแจงข้อความ เพื่อนำมาใช้สำหรับสร้างตารางการแจงข้อความ โดยขั้นตอนสร้างตารางประกอบด้วย 3 ขั้นตอน คือ นำกฎมาสร้าง Left Corner Symbol Set หรือ First Symbol Set โดยการหา leftmost ของคำที่จะแจงไวยากรณ์ จากนั้นนำมาใช้หา Lookahead Set ซึ่งเป็นชุดข้อมูลของคำถัดไปที่เป็นไปได้ทั้งหมดของกฎไวยากรณ์ ทั้งหมดนี้จะนำมาพิจารณาเพื่อใช้ในการสร้างตารางสำหรับแจงข้อความ ในส่วนนี้ผู้วิจัยจะทำการนับกฎที่ได้ถูกใช้ไปในระหว่างการแจกแจงข้อความไว้ด้วยเพื่อไปคำนวณค่าสถิติเพิ่มเติม จะได้ผลการแจกแจงข้อความที่มีความถูกต้องแม่นยำมากขึ้น ขั้นตอนในการสร้างตารางแจกแจงข้อความ

3.4.1 Parsing Generator

การสร้างตาราง LR Parsing Table ซึ่งเป็นขั้นตอนสำคัญในการแจงต้นไม้ไวยากรณ์ โดยใช้อัลกอริทึมการสร้างตารางแจงข้อความ [4]

3.4.2 Probabilistic Parsing Generator

เมื่อได้ตาราง parsing table จะทำการหาค่าความน่าจะเป็นแต่ละ Action โดยการหาความน่าจะเป็นของการเชื่อมต่อเมทริกซ์ก่อนซึ่งจะทำการสร้างเมทริกซ์การเกิด Lookahead ที่ต่อเนื่องกัน การเชื่อมต่อเมทริกซ์จะเป็นดังเงื่อนไขต่อไปนี้ หาก

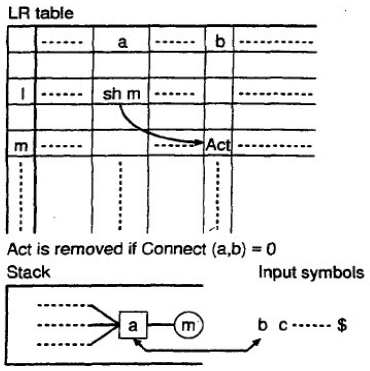
$$\text{Connect}(a_i,b_j) = \begin{cases} 1 & \text{If } b_j \text{ can follow } a_i \\ 0 & \text{otherwise} \end{cases}$$

โดยที่ b_j คือ terminal ตัวที่ j

a_i คือ terminal ตัวที่ i

ทำการสร้างตาราง Matrix การเกิดต่อเนื่องกันของ Lookhead มีมิติเท่ากับ Lookahead x Lookahead ค่าในแต่ละ element คือ 0 หรือ 1 ตามเงื่อนไขที่ได้กล่าวมาข้างต้น

เมื่อได้ตารางการเชื่อมต่อเมทริกซ์ทำการหาค่าความน่าจะเป็นดังตัวอย่างแสดงในรูปต่อไปนี้



รูปที่ 3. แสดงการเชื่อมต่อเมทริกซ์และตาราง Parsing Table

สมมุติว่ามี action A ใน state m ด้วย Lookahead b ดังรูป action A จะทำงานเมื่อ Connect(a_i,b_j) $\neq 0$ (คือ b สามารถเกิดขึ้นตาม a) แต่ถ้าหาก Connect(a_i,b_j) = 0 แสดงว่า b ไม่สามารถเกิดขึ้นตาม a ฉะนั้น Action A ใน state m กับ Lookahead b ไม่สามารถทำงานได้สามารถลบ Action A นี้ออกจากตาราง Parsing Table ได้ ต่อมาเมื่อได้การเชื่อมต่อเมทริกซ์แล้วจะทำการหาค่าความน่าจะเป็นการเชื่อมต่อกันภายใน จาก *Connect(a,* b) = 0 หมายความว่า b ไม่สามารถเกิดตาม a ได้ แต่การเชื่อมต่อเมทริกซ์ มีค่าเป็น 0 หรือ 1 เท่านั้นซึ่งเกี่ยวข้องกับความน่าจะเป็นในแต่ละองค์ประกอบ โดยความน่าจะเป็นการเชื่อมต่อเมทริกซ์แต่ละองค์ประกอบนั้นหาได้จาก

$$\text{PConnect}(a,b) = P(b|a) \qquad (2)$$

โดยที่

$\forall a,b \in V_T$ (เซตของ Terminal symbols)

ถ้า PConnect(a,b) = 0 หมายความว่า a และ b ไม่สามารถเกิดต่อเนื่องกันได้ PConnect(a,b) $\neq 0$ หมายความว่า b สามารถเกิดต่อเนื่อง a ด้วยความน่าจะเป็น P(b|a)

เมื่อได้ตาราง Probability Connection Matrix เข้าสู่อัลกอริทึมการหาความน่าจะเป็นของแต่ละ Action โดยผ่านอัลกอริทึม 2 ดังนี้

อัลกอริทึมที่ 1

1. Generate an LR table from Parsing Table Generator Algorithm .

2.Removal of actions:

For each shift action *shm* with lookahead a in the LR table , delete actions in the state m with lookalahead b if *PConnect(a, b) = 0*

3. Compact the parsing table if possible.

4. Incorporation of constraints into the parsing table:

For each action *shm* with lookahead a in the Parsing table , let

$$P = \sum_{i=1}^{N} PConnect(a,bi) \qquad (3)$$

where {bi : i = 1,... ,N} is the set of lookaheads for state m. For each action Aj in state rn with lookahead bi, assign a probability p to action Aj:

$$p = \frac{P(bi \mid a)}{P \times n} = \frac{PConnect(a, bi)}{P \times n} \quad (4)$$

where n is the number of conflict actions in state m with lookahead *bi.* The denominator is dearly a normalization factor.

5. For each action A with lookahead a in state 0, assign A a probability p = p(a/#), where "#" is the sentence beginning marker.

6. Assign a probability p = *1/n* to each action. A in state m with lookahead symbol a that has not been assigned a probability, where n is the number of conflict actions in state m with lookahead symbol a.

7. Return the table T produced at the completion of Step 6 as the probability parsing table.

จะได้ผลลัพธ์เป็นตาราง Parsing Table ที่มีค่าความน่าจะเป็นของแต่ละ action (probability parsing table)

3.5 Parsing Process

เมื่อได้ตาราง Parsing Table แล้วเริ่มต้นขั้นตอนนี้จะอ่านข้อความรับเข้าอีกครั้ง และนำผลลัพธ์จากโมดูล Parsing Table Generator คือ ตารางสำหรับแจงข้อความเพื่อสร้างต้นไม้ไวยากรณ์ (Probability Parsing Table) นำคำในข้อความเข้านั้นเปรียบเทียบกับ action ใน parsing table เลือกaction ที่มีความน่าจะเป็นกระทำก่อน แล้วเก็บลง stack โดยจะทำต่อเนื่องจากซ้ายไปขวาจนจบข้อความ โดยใช้อัลกอริทึมต่อไปนี้

อัลกอริทึม 2

```
s = State number
        a = Input
        For i to n in input sentence
        select the best action according to the probability of action
        If action is shift (Sh[n])
                Then
                shift the next input symbol and the state s onto the
                stack
        If action is reduce (Re[n] or A->β)
                Then
                pop items from the stack depend on production
                number (where n is a production number)
                push A and Si where Si=goto[Sn-r,A]
        If action is accept
                Then
                select the best derivation tree according
                to the probability of each derivation tree and
                return best derivation tree
        If error or empty entry in parsing table
                Then
                parser detected an error and return error
```

หากผลลัพธ์จากการ parsing เป็น Accept ก็จะคืนผลลัพธ์เป็นต้นไม้ไวยากรณ์ของข้อความเข้าออกไป โดยเมื่อการวิเคราะห์ข้อความได้ต้นไม้แต่ละต้นนั้นจะทำการคำนวณค่าความน่าจะเป็นของต้นไม้แต่ละต้นหาได้จากผลคูณของค่าความน่าจะเป็นของกฎ ทั้งหมดที่ใช้อยู่ในโครงสร้างต้นไม้นั้นๆ ต้นไม้สุดท้ายที่วิเคราะห์ได้หากวิเคราะห์ได้โครงสร้างต้นไม้ที่มีมากกว่า 1 โครงสร้างจะเลือกจากต้นไม้ที่มีความน่าจะเป็นสูงสุด ประสิทธิภาพการทำงานจะขึ้นอยู่กับความละเอียดของแกรมม่าที่ใช้ ปริมาณ tree ที่ใช้เรียนรู้และความซักซ้อนของข้อความตั้งต้น

4. ตัวอย่างการแจงข้อความ

การแจงข้อความจะแสดงลำดับการเกิดเส้นทางการแจงข้อความ จากตัวอย่างข้อความ “เด็กขโมยสมุด” จะเห็นได้ว่าข้อความนี้สามารถเป็นได้ทั้งข้อความสมบูรณ์หรือเป็นนามวลี แต่ในบริบทนั้นๆจะมีโครงสร้างไวยากรณ์ที่ถูกต้องอยู่เพียงแบบเดียว

ตารางที่ 1 แสดงผลการสร้างตาราง Parsing Table ของ เด็กขโมยสมุด

Parsing Table					
	$	np	s	vi	vt
0		Sh[2]	Sh[1]		
1	Sh[5]				
2				Sh[6]	Sh[7]
3	Re[1]				
4	Re[3]				
5	Acc				
6	Re[5]				
7		Sh[9]			

ตัวอย่างการทำงานเพื่อให้เห็นภาพชัดเจนจึงขอยกตัวอย่างประโยคนี้ ประโยคดังกล่าวเข้าสู่อัลกิริทึมขั้นแรกจะได้ตาราง LR Parsing Table สำหรับความสัมพันธ์ระหว่าง LR Parsing Table และตารางLR ประกอบไปด้วย สัญลักษณ์ Lookahead และ state การทำงาน ให้ action shift7 ใน state1 กับ Lookahead vt หลังจาก parser ทำงาน action shift7 นั้น Lookahead vt ก็จะถูก push สู่ Top stack และเลื่อนการทำงานไปยัง state 7 (action shift7 ให้ใช้ POS นั้นเทียบ Parsing Table แล้วกระโดดไป state ถัดไปตามที่ระบุไว้ในตาราง นั้นคือ ไป state 7)

action shift9 ใน state 7 กับ Lookahead np ทำงานได้ ถ้าหาก connect(vt,np) $\neq 0$ นั่นคือ Lookahead np สามารถเกิดตาม Lookahead vt

ได้ ดังนั้นค่า element ใน ตาราง Connection Matrix ที่ np เกิดตามหลัง vt เท่ากับ 1 แต่หาก connect(vt,np) =0 นั่นคือ Lookahead np ไม่สามารถเกิดตาม Lookahead vt ได้ ค่าใน element นี้เป็น 0

ตารางที่ 2 แสดงผลการสร้างตาราง Connection Matrix ของ เด็กชโมยสมุด

Connection Matrix					
	$	np	s	vi	vt
np	1			1	1
s	1				
vi	1				
vt		1			

เมื่อได้ตาราง Connection Matrix จะนำมาสร้างตาราง Probability Connection Matrix ตัวอย่าง

PConnect(a,b) = P(b|a)

PConnect(vt,np) = P(np|vt)

ตารางที่ 3 แสดงผลการสร้างตาราง Probability Connection Matrix ของ เด็กชโมยสมุด

Probability Connection Matrix					
	$	np	s	vi	vt
np	0.32			1	1
s	0.32				
vi	0.32				
vt		1			

เมื่อได้ตาราง Probability Connection Matrix แล้วจะมาทำการหา ความน่าจะเป็นแต่ละ action ซึ่งทำการหาความน่าจะเป็นรวมในแต่ละ state ก่อนจากสมการ (3) จะได้

$$P = PConnect(vt,np)$$

$$= 1$$

จากนั้นหาความน่าจะเป็นแต่ละ action จากสมการ 4 จะได้

$$p = \frac{PConnect(\mathrm{vt},\mathrm{np})}{P \times n}$$

$$p = \frac{1}{1 \times 1}$$

ดังนั้น ความน่าจะเป็นของ action shift9 ใน state 7 เท่ากับ 1.0 ดังในตาราง Probability Parsing Table จบกระบวนการจะได้ต้นไม้ไวยากรณ์ซึ่งจะรวมกันได้เป็นประโยคที่สมบูรณ์

5. บทสรุปและงานในอนาคต

บทความนี้อธิบายถึงกรอบแนวคิดการพัฒนาอัลกอริทึมการแจกแจงข้อความโดยอาศัยความน่าจะเป็นเข้ามาช่วยในขั้นตอนการสร้างตารางแจงข้อความเพื่อเพิ่มประสิทธิภาพการทำงานของการแจงข้อความ ช่วยลดงานในการตรวจหาต้นไม้ไวยากรณ์ที่ถูกต้องโดยความประสิทธิภาพจะขึ้นอยู่กับความละเอียดของไวยากรณ์ที่ใช้ ปริมาณต้นไม้ที่ใช้เรียนรู้และความซับซ้อนของประโยคที่รับเข้ามา ในอนาคตจะทำการพัฒนาและทดสอบประสิทธิภาพการทำงานของการแจงข้อความแบบ PGLR นี้ให้การวิเคราะห์มีความแม่นยำมากยิ่งขึ้น จะได้เป็นพื้นฐานของงานวิจัยที่เกี่ยวข้องกับการประมวลผลภาษาธรรมชาติสามารถนำไปใช้เพื่อพัฒนาระบบต่างๆต่อไป

เอกสารอ้างอิง

[1] Briscoe, T. and Carroll, J. (1993). “Generalized Probabilistic LR Parsing of Natural Language (Corpora) with Unification-Based Grammars”. Computational Linguistics, Vol.19, No.1, pages 25-59.

[2] Hirold Ima/ and Hozumi Tanaka (1998) “A Method of Incorporating Bigram Constraints into an LR Table and Its Effectiveness in Natural Language Processing”. In D.M.W. Powers (ed.) NeMLaP3/CoNLL98: New Methods in Language Processing and Computational Natural Language Learning, ACL, pp 225-233.

[3] Inui, K., Sornlertlamvanich, V., Tanaka, H. and Tokunaga, T. 1997. A New Formalization of Probabilistic GLR Parsing. Proceedings of the 5th International Workshop on Parsing Technologies.

[4] Ruangrajitpakorn T, Trakultaweekoon K, Supnithi T. (2002). “Building Thai CG tree bank using LRparser”, IEICETRANS.FUNDAMENTALS/COMMUN. /ELECTRON. /INF. & SYST., VOL. E85-A/B/C/D, No. 1

[5] Ruangrajitpakorn Taneth and Supnithi Thepchai. 2010. “A Current Status of Thai Categorial Grammars and Their Applications”, Proceedings of the 8th Workshop on Asian Language Resources, (Coling 2010, Beijing,Chaina)

[6] Ruangrajitpakorn T, Trakultaweekoon K, Supnithi T, “A Syntactic Resource for Thai: CG Treebank” Proceedings of the 7th Workshop on Asian Language Resources, ACL-IJCNLP 2009, pages 96–102

[7] Tomita M.,(1987), “An efficient augmented-context-free parsing algorithm” , Computational Linguistics, v.13 n.1-2, p.31-46.

[8] อัสนีย์ ก่อตระกูล. **การประมวลผลภาษามนุษย์ด้วยคอมพิวเตอร์ : เส้นทางสู่การพัฒนาระบบสารสนเทศอัจฉริยะ**.หน่วยปฏิบัติการวิจัยเชี่ยวชาญเฉพาะการประมวลผลภาษาธรรมชาติและเทคโนโลยีสารสนเทศอัจฉริยะ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์ , 2549.หน้า 195-231.