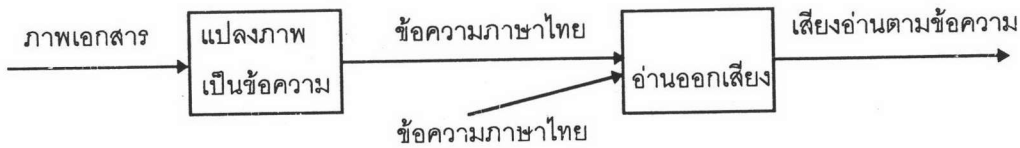
บทที่ 3

หลักการทำงานของโปรแกรม

3.1 การวิเคราะห์ข้อมูลนำเข้า

ข้อมูลนำเข้าของระบบคอมพิวเตอร์อ่านออกเสียงภาษาไทยจากเอกสาร มีอยู่ 2 ชนิด คือ

- ภาพเอกสาร อยู่ในรูปของแฟ้ม BMP ซึ่งเป็นรูปแบบแฟ้มข้อมูลภาพของบริษัท MicroSoft
- ข้อความภาษาไทย อยู่ในรูปของแฟ้มข้อความธรรมดา ซึ่งสามารถใช้โปรแกรมเอดิเตอร์ทั้งหลายสร้างหรือแก้ไขได้



รูปที่ 3.1 แสดงการทำงานหลัก ๆ ของระบบคอมพิวเตอร์อ่านออกเสียงภาษาไทยจากเอกสาร

จากรูปที่ 3.1 ผู้ใช้สามารถให้ระบบอ่านออกเสียงจากภาพเอกสาร หรือจากแฟ้มข้อความก็ได้ ซึ่งการอ่านออกเสียงจากภาพเอกสารย่อมมีความผิดพลาดมากกว่าการอ่านจากแฟ้มข้อความ เนื่องจากในขั้นตอนการแปลงภาพเป็นข้อความนั้น ความถูกต้องของข้อความที่ได้ขึ้นอยู่กับคุณภาพของภาพเอกสารที่นำมาอ่านว่ามีความชัดเจนเพียงใด

3.1.1 การวิเคราะห์ข้อมูลภาพเอกสาร

จากการวิเคราะห์ตัวอักษรในภาพเอกสาร ผู้วิจัยแบ่งตัวอักษรภาษาไทยทั้งหมดออกเป็น 5 กลุ่มตามความสูง และความกว้างของตัวอักษร คือ

1.) ตัวอักษรปกติ คือตัวอักษรที่มีความสูงและความกว้างอยู่ในเกณฑ์เฉลี่ยปกติ มี 53 ตัว ได้แก่ ก, ข, ฃ, ค, ฅ, ฆ, ง, จ, ฉ, ช, ซ, ฌ, ญ, ฑ, ฒ, ณ, ด, ต, ถ, ท, ธ, น, บ, ผ, พ, ภ, ม, ย, ร, ล, ว, ศ, ษ, ส, ห, ฬ, อ, ฮ, เ, แ, า, ฯ, ๆ, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

2.) ตัวอักษรเตี้ยอ้วน คือตัวอักษรที่มีความสูงน้อยกว่าเกณฑ์เฉลี่ย แต่มีความกว้างอยู่ในเกณฑ์เฉลี่ย มี 11 ตัว ได้แก่ ิ, ี, ึ, ื, ั, ้, ็, ่, ๊, ๋, ุ

3.) ตัวอักษรเตี้ยผอม คือตัวอักษรที่มีความสูงและความกว้างน้อยกว่าเกณฑ์เฉลี่ย มี 2 ตัว ได้แก่ ', ุ

4.) ตัวอักษรสูงอ้วน คือตัวอักษรที่มีความสูงสูงกว่าเกณฑ์เฉลี่ย และมีความกว้างอยู่ในเกณฑ์เฉลี่ย มี 8 ตัว ได้แก่ ฎ, ฏ, ฐ, ป, ฝ, ฟ, ฤ, ฦ

5.) ตัวอักษรสูงผอม คือตัวอักษรที่มีความสูงสูงกว่าเกณฑ์เฉลี่ย แต่มีความกว้างน้อยกว่าเกณฑ์เฉลี่ย มี 3 ตัว ได้แก่ โ, ใ, ไ

การแบ่งตัวอักษรออกเป็นหลายกลุ่ม จะช่วยให้การวิเคราะห์ตัวอักษรในขั้นตอนการรู้จำตัวอักษร (Character Recognition) ง่ายขึ้น เนื่องจากเป็นการเปรียบเทียบกันในกลุ่มย่อย แทนที่จะนำมาเปรียบเทียบกับตัวอักษรทั้งหมด

3.1.2 การสร้างข้อมูลตัวอักษรต้นแบบ

โดยในหัวข้อ 3.1.1 เราได้แบ่งตัวอักษรออกเป็น 5 กลุ่ม ตามความสูงและความกว้างของตัวอักษร ตัวอักษรแต่ละกลุ่มจะมีการสร้างต้นแบบและการวิเคราะห์ที่แตกต่างกัน แต่อยู่บนหลักการเดียวกัน คือ การนับจำนวนจุดตัดตัวอักษรในแนวต่างๆ รายละเอียดแนวที่ใช้นับจำนวนจุดตัดของตัวอักษรกลุ่มต่างๆ มีดังนี้

1.) กลุ่มตัวอักษรปกติ แนวที่ใช้ในการนับจำนวนจุดตัด คือ

- Top คือ เส้นในแนวนอนขอบบนของตัวอักษร
- Bottom คือ เส้นในแนวนอนขอบล่างของตัวอักษร
- Left คือ เส้นในแนวตั้งขอบซ้ายของตัวอักษร
- H1 คือ เส้นในแนวนอนที่ 1 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)
- H2 คือ เส้นในแนวนอนที่ 2 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)
- H3 คือ เส้นในแนวนอนที่ 3 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)
- H4 คือ เส้นในแนวนอนที่ 4 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)
- V1 คือ เส้นในแนวตั้งที่ 1 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)
- V2 คือ เส้นในแนวตั้งที่ 2 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)
- V3 คือ เส้นในแนวตั้งที่ 3 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)

2.) กลุ่มตัวอักษรเตี้ยอ้วน แนวที่ใช้ในการนับจำนวนจุดตัด คือ

- Top คือ เส้นในแนวนอนขอบบนของตัวอักษร
- Left คือ เส้นในแนวตั้งขอบซ้ายของตัวอักษร
- H1 คือ เส้นในแนวนอนที่ 1 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)
- V1 คือ เส้นในแนวตั้งที่ 1 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)
- V2 คือ เส้นในแนวตั้งที่ 2 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)

3.) กลุ่มตัวอักษรเตี้ยผอม แนวที่ใช้ในการนับจำนวนจุดตัด คือ

- H1 คือ เส้นในแนวนอนที่ 1 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)

- V1 คือ เส้นในแนวตั้งที่ 1 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)

4.) กลุ่มตัวอักษรสูงอ้วน แนวที่ใช้ในการนับจำนวนจุดตัด คือ

- Top คือ เส้นในแนวนอนขอบบนของตัวอักษร
- Bottom คือ เส้นในแนวนอนขอบล่างของตัวอักษร
- Left คือ เส้นในแนวตั้งขอบซ้ายของตัวอักษร
- Right คือ เส้นในแนวตั้งขอบขวาของตัวอักษร
- H1 คือ เส้นในแนวนอนที่ 1 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)
- V1 คือ เส้นในแนวตั้งที่ 1 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)

5.) กลุ่มตัวอักษรสูงผอม แนวที่ใช้ในการนับจำนวนจุดตัด คือ

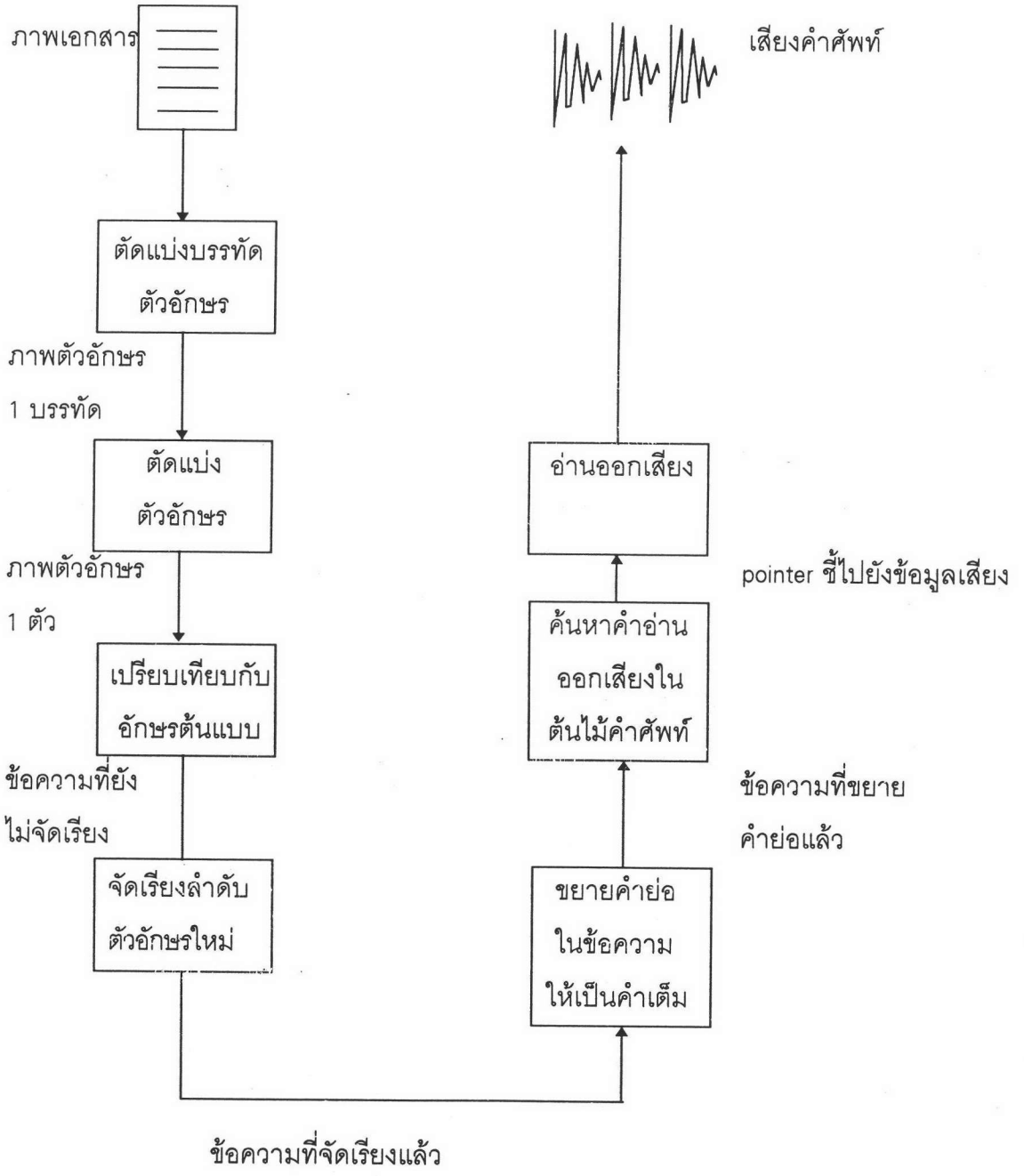
- V1 คือ เส้นในแนวนอนที่ 1 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)
- V2 คือ เส้นในแนวตั้งที่ 2 (ค่าที่แท้จริงกำหนดอยู่ในแฟ้มรูปแบบตัวอักษร)

ต่อไปนี้จะเป็นการแสดงการสร้างข้อมูลตัวอักษรต้นแบบ โดยสมมติให้มีตัวอักษรที่จะนำมาสร้างต้นแบบ 3 ตัว คือ ตัวอักษร “ก”, “ข” และ “ค” ตามลำดับ รูปที่ 3.2 แสดงการสร้างข้อมูลต้นแบบตัวอักษร “ก”, “ข” และ “ค”

	T	B	L	H1	H2	H3	H4	V1	V2	V3
	1	2	1	2	2	2	2	1	2	1
	2	1	1	3	3	3	2	2	1	1
	1	2	1	2	3	2	2	2	3	1

รูปที่ 3.2 แสดงการสร้างข้อมูลต้นแบบตัวอักษร “ก”, “ข” และ “ค”

จากรูปที่ 3.1 ขยายออกได้เป็นรูปที่ 3.3

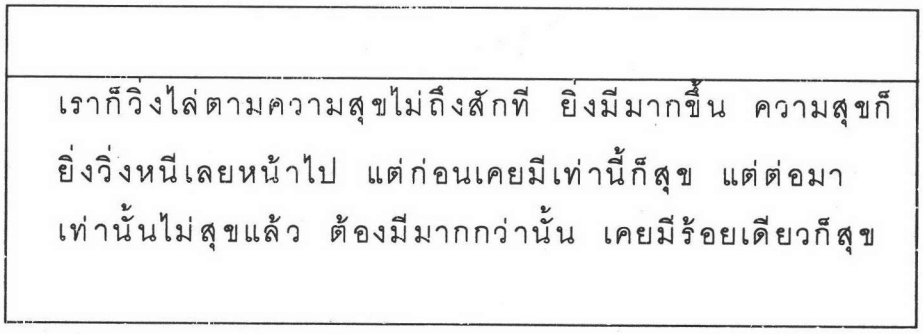


รูปที่ 3.3 แสดงการทำงานของระบบคอมพิวเตอร์อ่านออกเสียงภาษาไทยจากเอกสาร

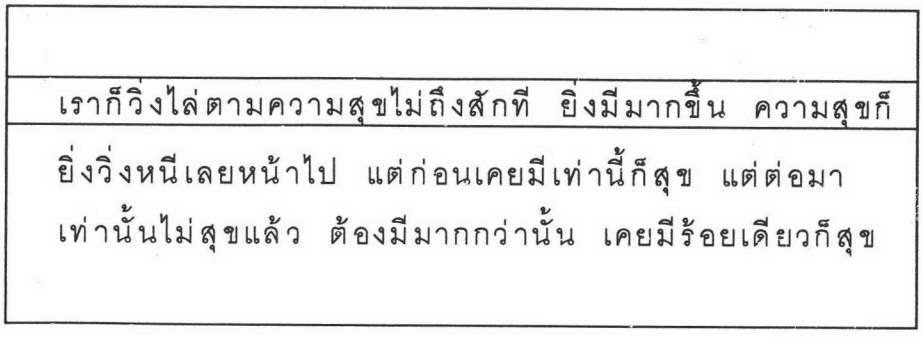
3.2 ขั้นตอนการแปลงภาพเอกสารเป็นข้อความ

3.2.1 การตัดแบ่งบรรทัดตัวอักษร

ในการแปลงภาพเอกสารเป็นข้อความ จะใช้วิธีการตัดแบ่งภาพเอกสารออกเป็นส่วนย่อย ๆ จนสุดท้ายได้เป็นภาพตัวอักษรแต่ละตัวเพื่อนำไปวิเคราะห์ ในขั้นตอนแรกของการตัดแบ่งเอกสาร จะตัดภาพเอกสารออกมาทีละบรรทัด โดยเริ่มสแกนในแนวนอน จากแนวนอนบนสุดของภาพเอกสาร ลงมาทีละแถวจนกว่าจะพบส่วนใดส่วนหนึ่งของตัวอักษรก็จะหยุดและยึดเป็นขอบบนของบรรทัดตัวอักษร ดังรูปที่ 3.4 หลังจากนั้นจะสแกนในแนวนอนลงมาจนกว่าจะไม่พบส่วนใดส่วนหนึ่งของตัวอักษรก็จะได้บรรทัดของภาพตัวอักษรมา 1 บรรทัด ดังรูปที่ 3.5 ซึ่งจะนำเข้าไปสู่ขั้นตอนการตัดแบ่งตัวอักษรต่อไป



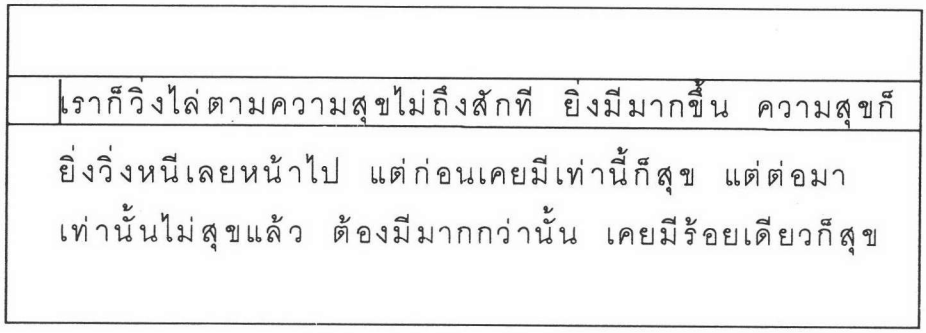
รูปที่ 3.4 การสแกนหาขอบบนของบรรทัดตัวอักษร



รูปที่ 3.5 การสแกนหาขอบล่างของบรรทัดตัวอักษร

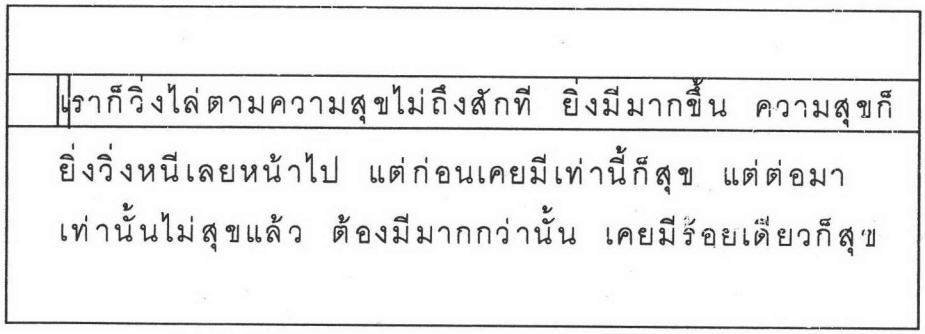
3.2.2 การตัดแบ่งตัวอักษร

เมื่อได้ภาพตัวอักษรมา 1 บรรทัดแล้ว จะทำการตัดแบ่งเป็น block ตัวอักษร โดยการสแกนในแนวตั้ง เริ่มตั้งแต่ขอบซ้ายสุดของบรรทัดตัวอักษรมาทางขวา จนกว่าจะพบส่วนใดส่วนหนึ่งของตัวอักษร ดังรูปที่ 3.6



รูปที่ 3.6 การสแกนหาขอบซ้ายของตัวอักษร

เมื่อพบส่วนใดส่วนหนึ่งของตัวอักษรแล้ว จะยึดแนวนั้นเป็นขอบซ้ายของ block ตัวอักษร แล้วทำการสแกนในแนวตั้งไปทางขวาต่อไปจนกว่าจะไม่พบส่วนใดส่วนหนึ่งของตัวอักษร ก็จะได้ block ของภาพตัวอักษรมา 1 block ดังรูป 3.7



รูปที่ 3.7 การสแกนหาขอบขวาของตัวอักษร

ซึ่งภาพตัวอักษร 1 block ที่ได้นี้อาจจะไม่ใช่ภาพของตัวอักษร 1 ตัวก็ได้ เนื่องจากภาษาไทยเรามีสระล่างและสระบน เช่น คำว่า อยู่ จะได้ ' อ ' เป็น 1 block และอีก 1 block จะมีภาพตัวอักษร ' ยู่ ' อยู่ใน block เดียวกัน ดังนั้น ใน 1 block ที่ได้ จึงต้องมีการตัดแบ่งออกเป็น block ย่อย ๆ จนกว่าจะได้ภาพตัวอักษร 1 ตัว ต่อ 1 block

ซึ่งจากการวิเคราะห์ภาพเอกสารแล้ว พบว่ารูปแบบของ block ตัวอักษร แบ่งออกได้เป็น 8 แบบคือ

1.) block ที่มีภาพตัวอักษรอยู่ 1 ตัว เช่น

2.) block ที่มีพยัญชนะ 1 ตัว และสระล่าง 1 ตัว เช่น

3.) block ที่มีพยัญชนะ 1 ตัว และสระบน 1 ตัว เช่น

4.) block ที่มีพยัญชนะ 1 ตัว, สระบนและสระล่างอย่างละตัว เช่น

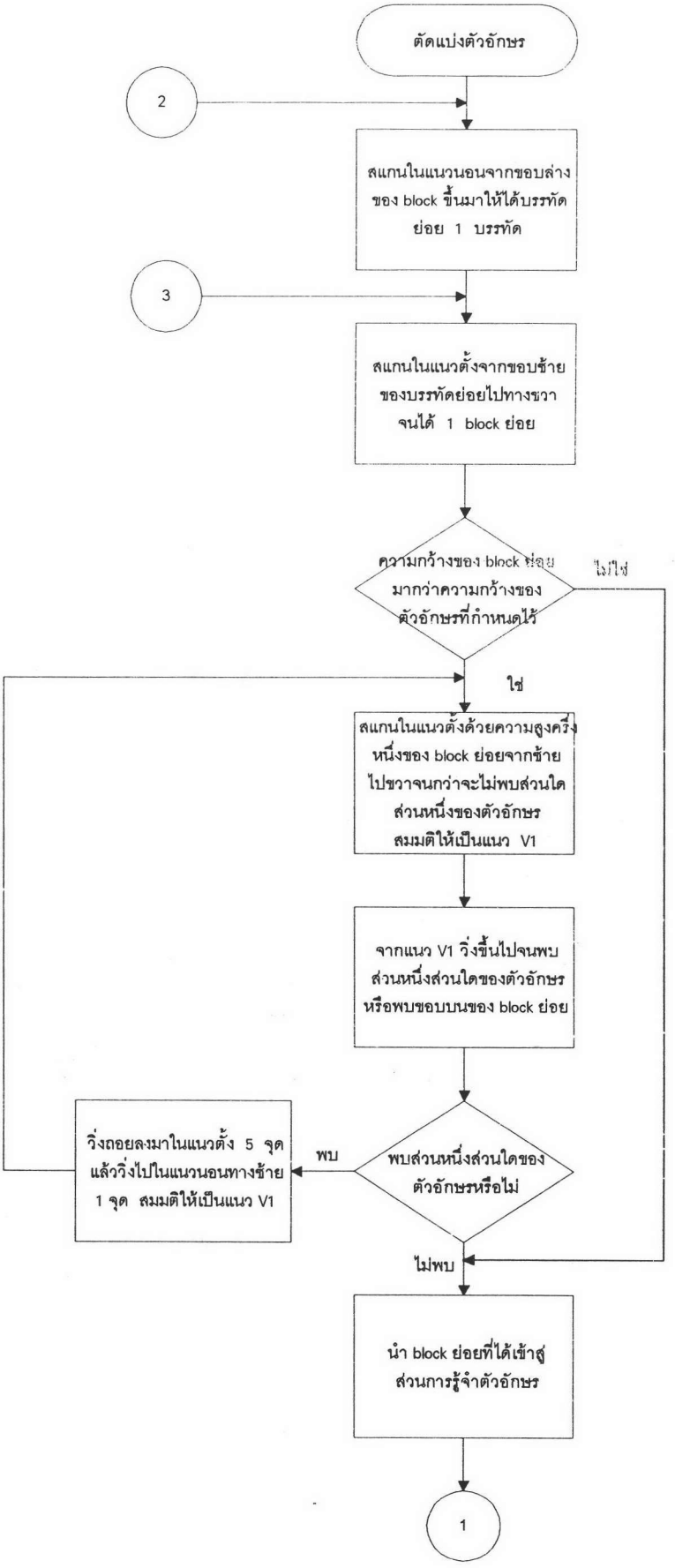
5.) block ที่มีพยัญชนะ 1 ตัว และสระบน 2 ตัว เช่น

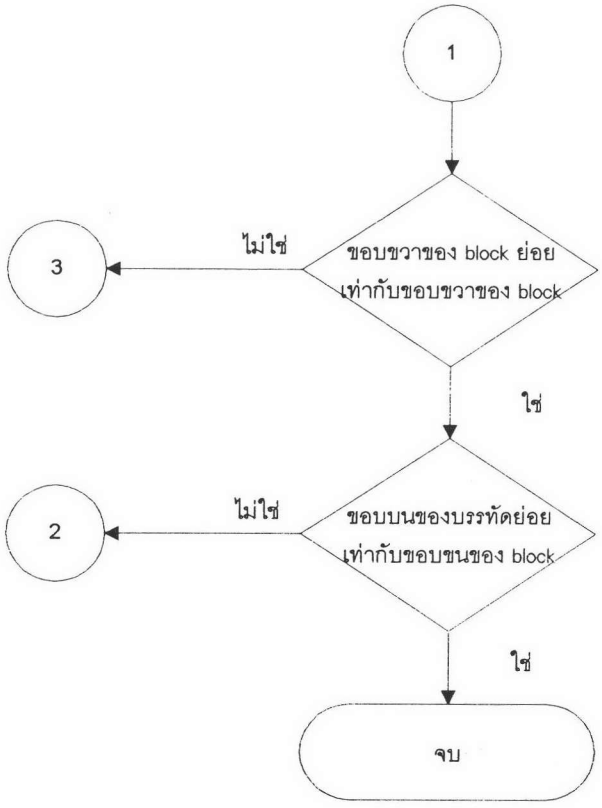
6.) block ที่มีพยัญชนะ 1 ตัว และสระ 1 ตัวไขว้กันอยู่ เช่น

7.) block ที่มีพยัญชนะ 2 ตัว และสระ 1 ตัวไขว้กันอยู่ เช่น

8.) block ที่มีพยัญชนะ 2 ตัว และสระ 2 ตัวไขว้กันอยู่ เช่น

จากรูปแบบ block ตัวอักษรทั้งหมด ทำให้เขียนวิธีการตัดแบ่ง block ย่อย ให้ได้ภาพตัวอักษร 1 ตัว ใน 1 block เป็น flow chart ดังรูปที่ 3.8


ตัดแบ่งตัวอักษร
2
สแกนในแนวนอนจากขอบล่างของ block ขึ้นมาให้ได้บรรทัดย่อย 1 บรรทัด
3
สแกนในแนวตั้งจากขอบซ้ายของบรรทัดย่อยไปทางขวาจนได้ 1 block ย่อย
ความกว้างของ block ย่อย มากว่าความกว้างของตัวอักษรที่กำหนดไว้
ไม่ใช่
ใช่
สแกนในแนวตั้งด้วยความสูงครึ่งหนึ่งของ block ย่อยจากซ้ายไปขวาจนกว่าจะไม่พบส่วนใดส่วนหนึ่งของตัวอักษร สมมติให้เป็นแนว V1
จากแนว V1 วิ่งขึ้นไปจนพบส่วนหนึ่งส่วนใดของตัวอักษร หรือพบขอบบนของ block ย่อย
พบส่วนหนึ่งส่วนใดของตัวอักษรหรือไม่
พบ
วิ่งถอยลงมาในแนวตั้ง 5 จุด แล้ววิ่งไปในแนวนอนทางซ้าย 1 จุด สมมติให้เป็นแนว V1
ไม่พบ
นำ block ย่อยที่ได้เข้าสู่ส่วนการรู้จำตัวอักษร
1

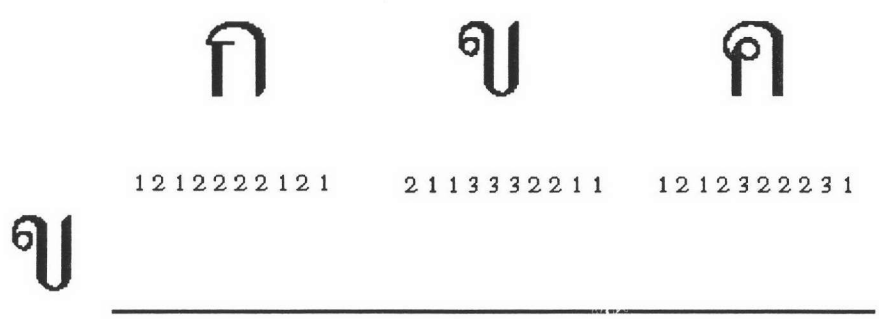


รูปที่ 3.8 Flow Chart แสดงการตัดแบ่งตัวอักษร

3.2.3 การเปรียบเทียบกับอักษรต้นแบบ

เป็นขั้นตอนการนำ block ภาพตัวอักษรที่ได้มาวิเคราะห์ว่าเป็นอักษรตัวใด โดยใช้วิธีวิ่งในแนวนอน และแนวตั้งของ block ภาพตัวอักษร แล้วนับจำนวนจุดที่ตัดกับตัวอักษรในแต่ละแนว จากจำนวนจุดตัดที่นับได้ จะนำไปเปรียบเทียบกับจำนวนจุดตัดของตัวอักษรต้นแบบทุกตัว ผลลัพท์ที่ได้คือ รหัสแอสกีของตัวอักษรแม่แบบที่จำนวนจุดตัดในแต่ละแนวใกล้เคียงกับจำนวนจุดตัดของ block ภาพตัวอักษรที่นำมาวิเคราะห์มากที่สุด

ต่อไปนี้เป็นการแสดงการวิเคราะห์ตัวอักษร โดยการเปรียบเทียบกับอักษรต้นแบบ ซึ่งสมมติให้มีตัวอักษรต้นแบบ 3 ตัว คือ ตัวอักษร “ก”, “ข” และ “ค” และสมมติให้ตัวอักษรที่นำมาเปรียบเทียบ คือ ตัวอักษร “ข” ดังรูปที่ 3.9



รูปที่ 3.9 แสดงตัวอักษรและข้อมูลตัวอักษรต้นแบบ

วิธีเปรียบเทียบโดยการหาความแตกต่างของตัวอักษรที่นำไปเปรียบเทียบกับตัวอักษรต้นแบบ ซึ่งคำนวณด้วยสูตร

nTop = | pTop - xTop |

nBottom = | pBottom - xBottom |

nLeft = | pLeft - xLeft |

nH1 = | pH1 - xH1 |

nH2 = | pH2 - xH2 |

nH3 = | pH3 - xH3 |

nH4 = | pH4 - xH4 |

nV1 = | pV1 - xV1 |

nV2 = | pV2 - xV2 |

nV3 = | pV3 - xV3 |

โดยที่

nTop คือ ความแตกต่างของจำนวนจุดตัดในแนว Top ระหว่างตัวอักษรที่นำไปเปรียบเทียบกับตัวอักษรต้นแบบ

pTop คือ จำนวนจุดตัดในแนว Top ของตัวอักษรต้นแบบ

xTop คือ จำนวนจุดตัดในแนว Top ของตัวอักษรที่นำไปเปรียบเทียบ

nBottom คือ ความแตกต่างของจำนวนจุดตัดในแนว Bottom ระหว่างตัวอักษรที่นำไปเปรียบเทียบกับตัวอักษรต้นแบบ

pBottom คือ จำนวนจุดตัดในแนว Bottom ของตัวอักษรต้นแบบ

xBottom คือ จำนวนจุดตัดในแนว Bottom ของตัวอักษรที่นำไปเปรียบเทียบ

nLeft คือ ความแตกต่างของจำนวนจุดตัดในแนว Left ระหว่างตัวอักษรที่นำไปเปรียบเทียบกับตัวอักษรต้นแบบ

pLeft คือ จำนวนจุดตัดในแนว Left ของตัวอักษรต้นแบบ

xLeft คือ จำนวนจุดตัดในแนว Left ของตัวอักษรที่นำไปเปรียบเทียบ

nH1 คือ ความแตกต่างของจำนวนจุดตัดในแนว H1 ระหว่างตัวอักษรที่นำไปเปรียบเทียบกับตัวอักษรต้นแบบ

pH1 คือ จำนวนจุดตัดในแนว H1 ของตัวอักษรต้นแบบ

xH1 คือ จำนวนจุดตัดในแนว H1 ของตัวอักษรที่นำไปเปรียบเทียบ

nH2 คือ ความแตกต่างของจำนวนจุดตัดในแนว H2 ระหว่างตัวอักษรที่นำไปเปรียบเทียบกับตัวอักษรต้นแบบ

pH2 คือ จำนวนจุดตัดในแนว H2 ของตัวอักษรต้นแบบ

xH2 คือ จำนวนจุดตัดในแนว H2 ของตัวอักษรที่นำไปเปรียบเทียบ

nH3 คือ ความแตกต่างของจำนวนจุดตัดในแนว H3 ระหว่างตัวอักษรที่นำไปเปรียบเทียบกับตัวอักษรต้นแบบ

pH3 คือ จำนวนจุดตัดในแนว H3 ของตัวอักษรต้นแบบ

xH3 คือ จำนวนจุดตัดในแนว H3 ของตัวอักษรที่นำไปเปรียบเทียบ

nH4 คือ ความแตกต่างของจำนวนจุดตัดในแนว H4 ระหว่างตัวอักษรที่นำไปเปรียบเทียบกับตัวอักษรต้นแบบ

pH4 คือ จำนวนจุดตัดในแนว H4 ของตัวอักษรต้นแบบ

xH4 คือ จำนวนจุดตัดในแนว H4 ของตัวอักษรที่นำไปเปรียบเทียบ

nV1 คือ ความแตกต่างของจำนวนจุดตัดในแนว V1 ระหว่างตัวอักษรที่นำไปเปรียบเทียบกับตัวอักษรต้นแบบ

pV1 คือ จำนวนจุดตัดในแนว V1 ของตัวอักษรต้นแบบ

xV1 คือ จำนวนจุดตัดในแนว V1 ของตัวอักษรที่นำไปเปรียบเทียบ

nV2 คือ ความแตกต่างของจำนวนจุดตัดในแนว V2 ระหว่างตัวอักษรที่นำไปเปรียบเทียบกับตัวอักษรต้นแบบ

pV2 คือ จำนวนจุดตัดในแนว V2 ของตัวอักษรต้นแบบ

xV2 คือ จำนวนจุดตัดในแนว V2 ของตัวอักษรที่นำไปเปรียบเทียบ

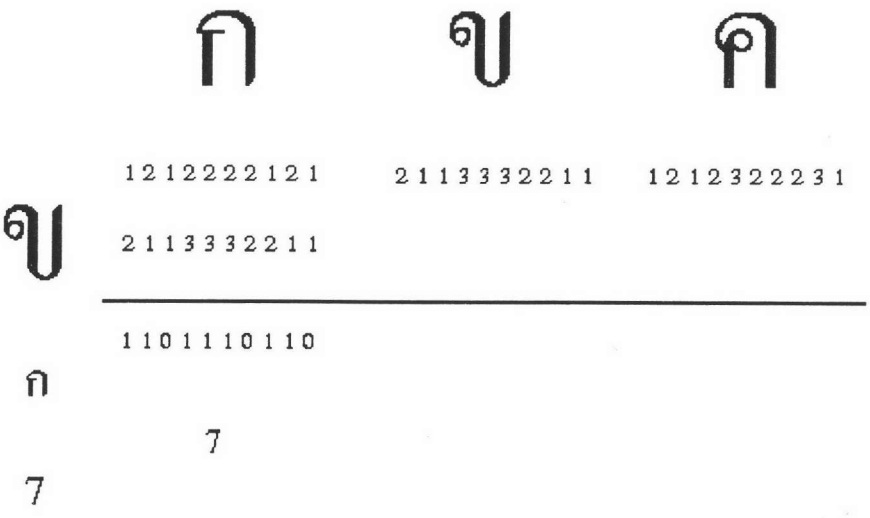
nV3 คือ ความแตกต่างของจำนวนจุดตัดในแนว V3 ระหว่างตัวอักษรที่นำไปเปรียบเทียบกับตัวอักษรต้นแบบ

pV3 คือ จำนวนจุดตัดในแนว V3 ของตัวอักษรต้นแบบ

xV3 คือ จำนวนจุดตัดในแนว V3 ของตัวอักษรที่นำไปเปรียบเทียบ

ผลลัพธ์ที่ได้จากการเปรียบเทียบกับตัวอักษรต้นแบบ คือ รหัสแอสกีของตัวอักษรต้นแบบที่ผลรวมความแตกต่าง (nTop + nBottom + nLeft + nH1 + nH2 + nH3 + nH4 + nV1 + nV2 + nV3) มีค่าน้อยที่สุด ดังรูปที่ 3.10 แสดงการเปรียบเทียบตัวอักษร “ข” กับอักษรต้นแบบ “ก” ได้ผลรวมความแตกต่างเท่ากับ 7 ซึ่งน้อยกว่าค่าตั้งต้น (99) ที่ตั้งไว้

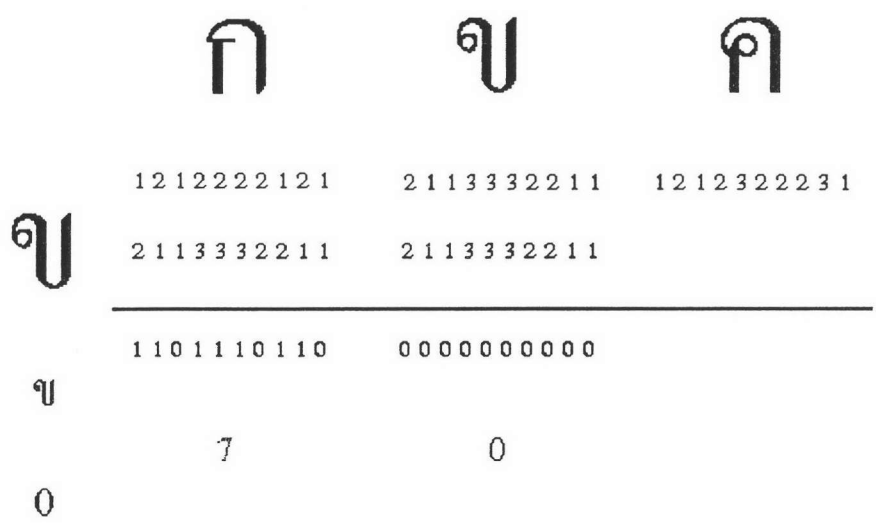
ดังนั้น การเปรียบเทียบตัวอักษร “ข” กับอักษรต้นแบบ “ก” ได้ผลลัพธ์เป็น “ก” โดยมีค่าความแตกต่างน้อยที่สุดเท่ากับ 7



รูปที่ 3.10 แสดงการเปรียบเทียบตัวอักษร “ข” กับตัวอักษรต้นแบบ “ก”

เมื่อนำตัวอักษร “ข” ไปเปรียบเทียบกับอักษรต้นแบบตัวต่อไป คือ “ข” จะได้ผลรวมความแตกต่างเท่ากับ 0 ซึ่งน้อยกว่าค่าความแตกต่างน้อยที่สุด

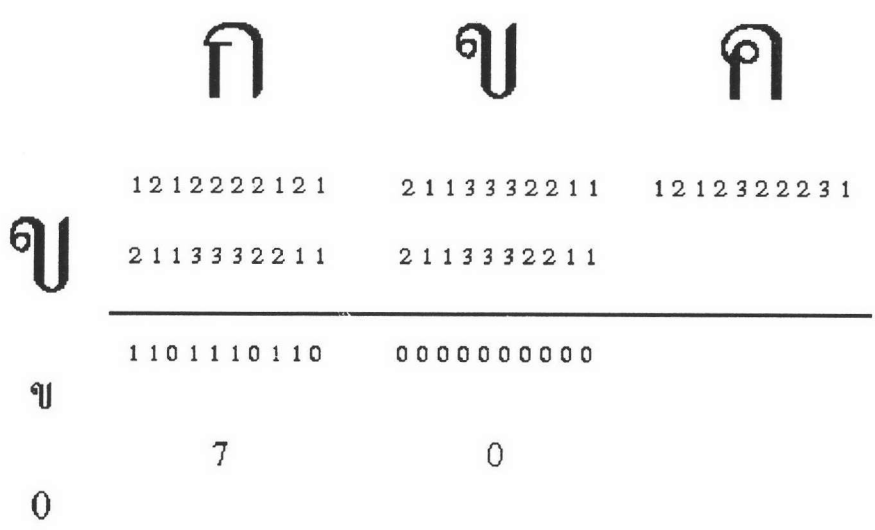
ดังนั้น การเปรียบเทียบตัวอักษร “ข” กับอักษรต้นแบบ “ข” ได้ผลลัพธ์เป็น “ข” โดยมีค่าความแตกต่างน้อยที่สุดเท่ากับ 0 ดังรูปที่ 3.11



รูปที่ 3.11 แสดงการเปรียบเทียบตัวอักษร “ข” กับตัวอักษรต้นแบบ “ข”

เมื่อนำตัวอักษร “ข” ไปเปรียบเทียบกับอักษรต้นแบบตัวต่อไป คือ “ค” จะได้ผลรวมความแตกต่างเท่ากับ 6 ซึ่งมากกว่าค่าความแตกต่างน้อยที่สุด

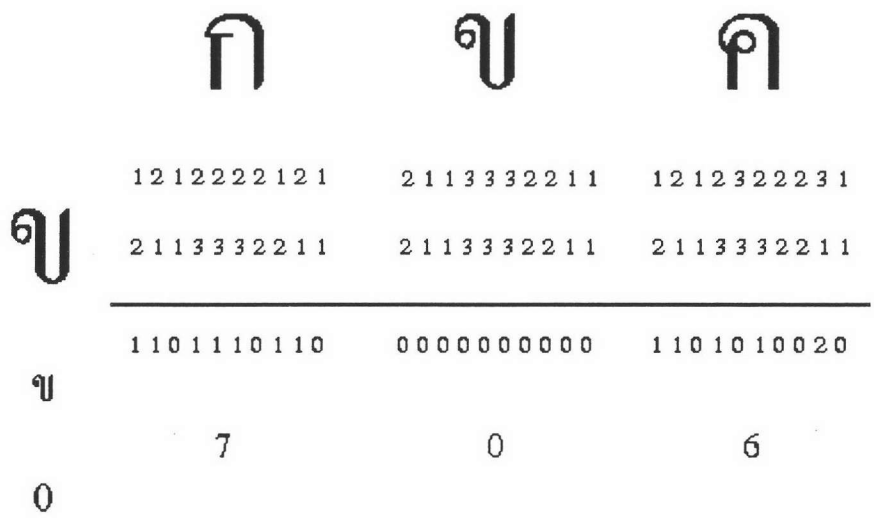
ดังนั้น ผลลัพธ์ยังคงเป็น “ข” และค่าความแตกต่างน้อยที่สุดก็ยังคงเท่ากับ 0 ดังรูปที่ 3.12



รูปที่ 3.11 แสดงการเปรียบเทียบตัวอักษร “ข” กับตัวอักษรต้นแบบ “ข”

เมื่อนำตัวอักษร “ข” ไปเปรียบเทียบกับอักษรต้นแบบตัวต่อไป คือ “ค” จะได้ผลรวมความแตกต่างเท่ากับ 6 ซึ่งมากกว่าค่าความแตกต่างน้อยที่สุด

ดังนั้น ผลลัพธ์ยังคงเป็น “ข” และค่าความแตกต่างน้อยที่สุดก็ยังคงเท่ากับ 0 ดังรูปที่ 3.12



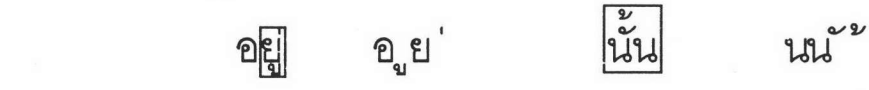
รูปที่ 3.12 แสดงการเปรียบเทียบตัวอักษร “ข” กับตัวอักษรต้นแบบ “ค”

ปรากฏว่าเมื่อเปรียบเทียบกับตัวอักษรต้นแบบจนครบทุกตัวแล้ว ผลลัพธ์ที่ได้ คือ รหัสแอสกีของตัวอักษร “ข”

จะเห็นได้ว่าผลลัพธ์จากการวิเคราะห์ตัวอักษรวิธีนี้ อาจจะไม่ถูกต้อง 100 เปอร์เซนต์ เป็นแต่เพียงการคำนวณหาตัวอักษรต้นแบบที่มีความแตกต่างจากตัวอักษรที่นำมาวิเคราะห์น้อยที่สุด โดยการเปรียบเทียบจากจำนวนจุดตัดในแต่ละแนวเท่านั้นเอง

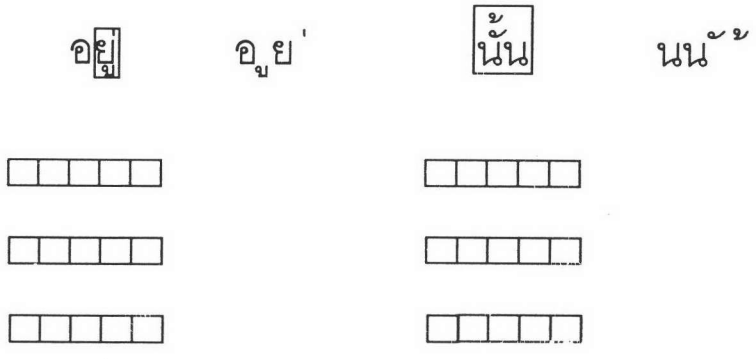
3.2.4 การจัดเรียงลำดับตัวอักษรใหม่

จากรูปแบบ block ภาพตัวอักษรทั้ง 8 แบบในหัวข้อ 3.2.2 จะเห็นว่าถึงแม้จะสามารถตัดแบ่งตัวอักษรออกมาได้เป็น 1 ตัวอักษรต่อ 1 block และสามารถนำ block ภาพตัวอักษรย่อยแต่ละ block มาวิเคราะห์จนรู้ว่าเป็นอักษรตัวใดแล้ว แต่ลำดับตัวอักษรที่ได้ยังมีบางกรณีที่ไม่ถูกต้อง เช่น



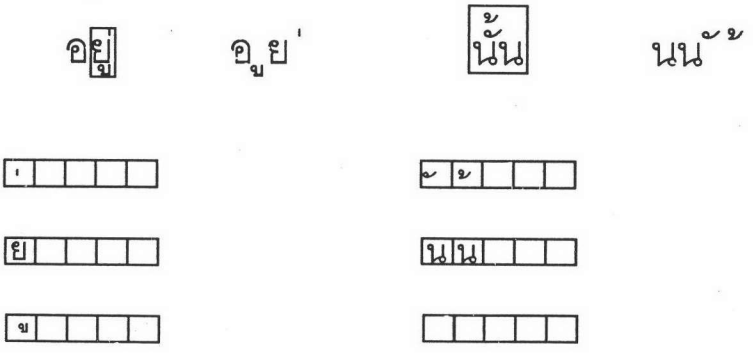
จากการตัดแบ่งตัวอักษร และเปรียบเทียบกับอักษรต้นแบบแล้ว จะได้ข้อความ “อ ุย ่” และ “นน ั ้” ซึ่งไม่ถูกต้อง ดังนั้นจึงจำเป็นต้องการจัดเรียงตัวอักษรใหม่ โดยมีวิธีการดังนี้

1.) สร้าง array ของตัวอักษรขึ้นมา 3 ชุด ชุดละ 5 element เป็น array ของพยัญชนะ, สระล่างและสระบนอย่างละชุด ดังรูป 3.13



รูปที่ 3.13 แสดงการสร้าง Array ของพยัญชนะ, สระบนและสระล่างเพื่อใช้ในการจัดเรียงลำดับตัวอักษร

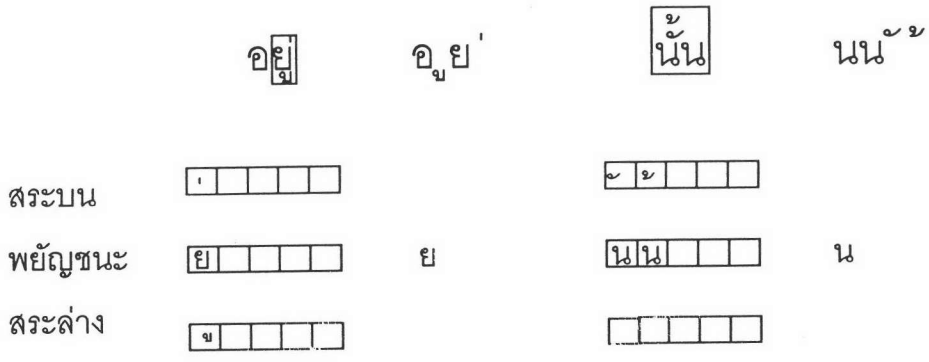
2.) หลังจากขั้นตอนการรู้จำตัวอักษรแล้ว ให้เก็บตัวอักษรที่วิเคราะห์ได้ลงใน array ทั้ง 3 ตามชนิดของตัวอักษร ดังรูป 3.14



รูปที่ 3.14 แสดงการเก็บตัวอักษรที่วิเคราะห์ได้ลงใน Array ทั้ง 3 ตามชนิดตัวอักษร

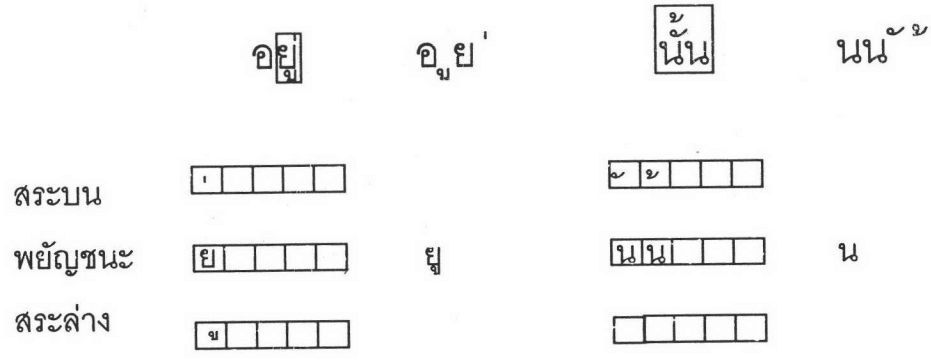
3.) จัดเรียงลำดับตัวอักษรใหม่ โดยใช้หลัก

3.1.) อ่านพยัญชนะตัวที่ 1 จาก array ของพยัญชนะเข้ามา ดังรูป 3.15



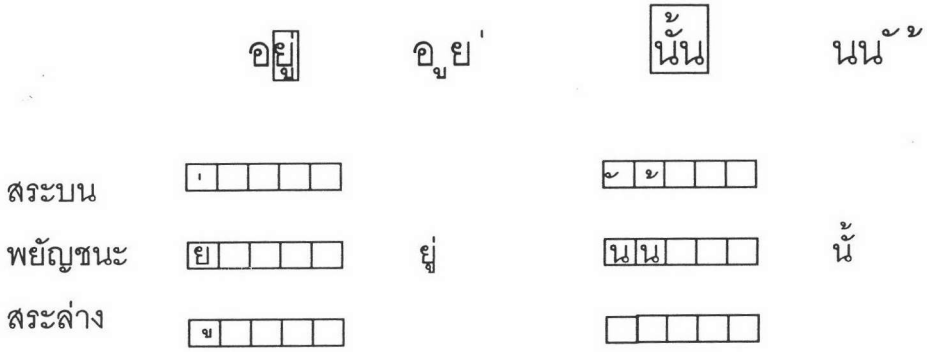
รูปที่ 3.15 แสดงการอ่านพยัญชนะตัวที่ 1 จาก Array ของพยัญชนะ

3.2.) อ่านสระล่างจาก array ของสระล่างทั้งหมดเข้ามาทีละตัว ดังรูป 3.16



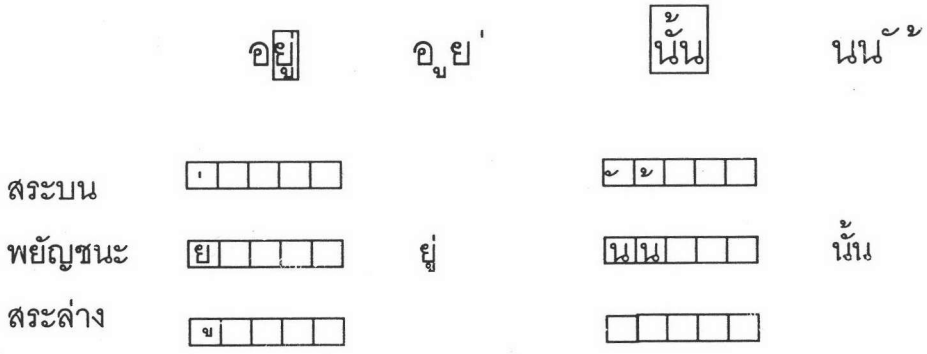
รูปที่ 3.16 แสดงการอ่านสระล่างจาก Array ของสระล่างทั้งหมดเข้ามาทีละตัว

3.3.) อ่านสระบนจาก array ของสระบนทั้งหมดเข้ามาทีละตัว ดังรูป 3.17



รูปที่ 3.17 แสดงการอ่านสระบนจาก Array ของสระบนทั้งหมดเข้ามาทีละตัว

3.4.) อ่านพยัญชนะตัวที่ 2 และตัวต่อๆไปจาก array ของพยัญชนะเข้ามาทีละตัวจนหมด ดังรูป 3.18



รูปที่ 3.18 แสดงการอ่านพยัญชนะตัวที่ 2 และตัวต่อๆไปจาก Array ของพยัญชนะเข้ามาทีละตัวจนหมด

จากวิธีการข้างต้นจะทำให้ได้ลำดับตัวอักษรใหม่ที่ถูกต้อง

3.3 ขั้นตอนการแปลงข้อความให้เป็นเสียง

3.3.1 การขยายคำย่อในข้อความให้เป็นคำเต็ม

ข้อความที่จะนำมาอ่านออกเสียงในบางครั้งอาจจะมีคำย่อ เช่น ด.ช., ด.ญ., ม.ร.ว. เป็นต้น ปนอยู่ในข้อความด้วย ซึ่งหากเราเก็บคำศัพท์ “ด.ช.” และเสียงอ่านคำว่า

“เด็กชาย” ไว้ในพจนานุกรม ก็จะทำให้เกิดความซ้ำซ้อนของข้อมูลเสียงกับคำว่า “เด็ก” และคำว่า “ชาย” ในพจนานุกรมเสียง

ดังนั้นเพื่อลดความซ้ำซ้อนของข้อมูล จึงให้มีโปรเซสในการค้นหาคำย่อในข้อความที่จะนำมาอ่านออกเสียง แล้วขยายคำย่อนั้นเป็นคำเต็ม แล้วจึงนำข้อความที่ขยายแล้วมาอ่านออกเสียงต่อไป

ในขั้นตอนการขยายคำย่อในข้อความให้เป็นคำเต็ม นอกจากจะใช้ในการขยายคำย่อให้เป็นคำเต็มตามปกติแล้ว ยังใช้ในการแก้ไขตัวอักษรที่ผิดปกติจากขั้นตอนการแปลงภาพเอกสารเป็นข้อความได้อีกด้วย เช่น ในขั้นตอนการแปลงภาพเอกสารเป็นข้อความตัวอักษรสระแอ “แ” จะถูกวิเคราะห์เป็นตัวอักษรสระเอ 2 ตัวติดกัน “เ เ” หรือตัวอักษรสระอะ “ะ” จะถูกวิเคราะห์เป็นตัวอักษรไม้หันอากาศ 2 ตัว “ั ั” ซึ่งเราสามารถแก้ไขได้ง่ายๆ ด้วยการใช้ขั้นตอนการขยายคำย่อในข้อความให้เป็นคำเต็ม โดยการเติมข้อมูลคำย่อและคำเต็มดังนี้เพิ่มเข้าไปในแฟ้มคำย่อ

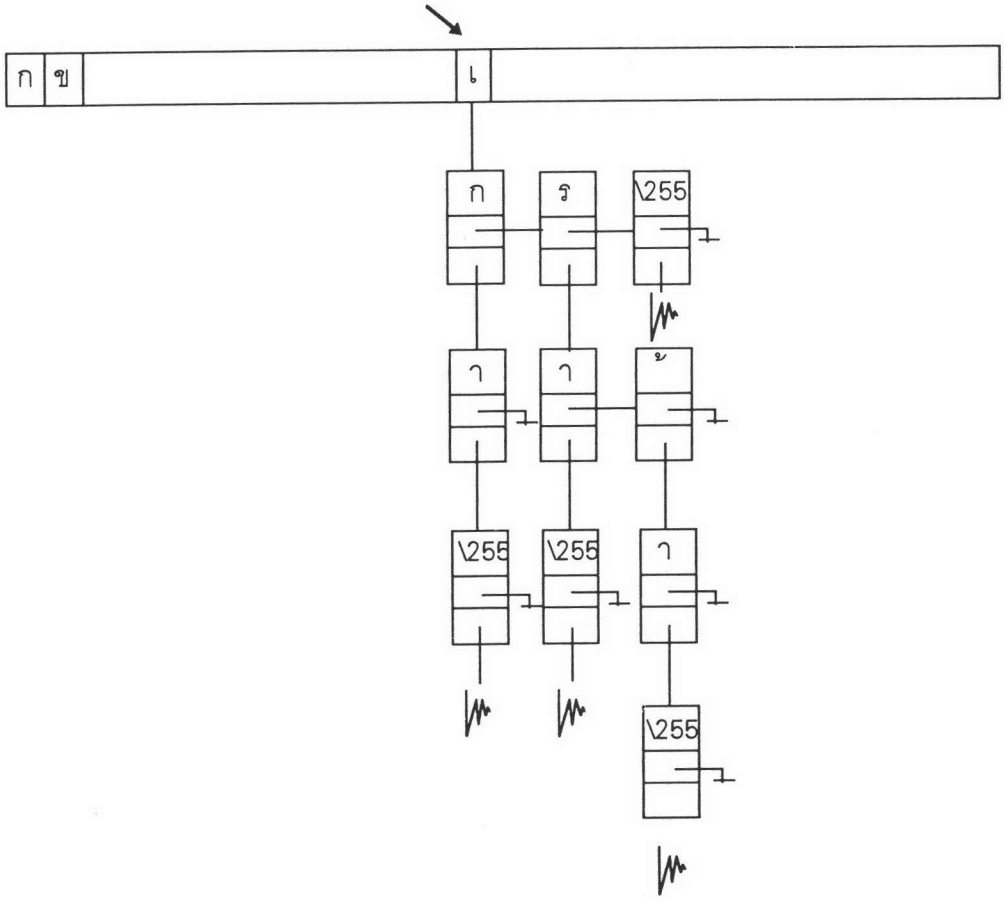
เ เ=แ

ั ั=ะ

3.3.2 การค้นหาคำอ่านออกเสียงในต้นไม้คำศัพท์

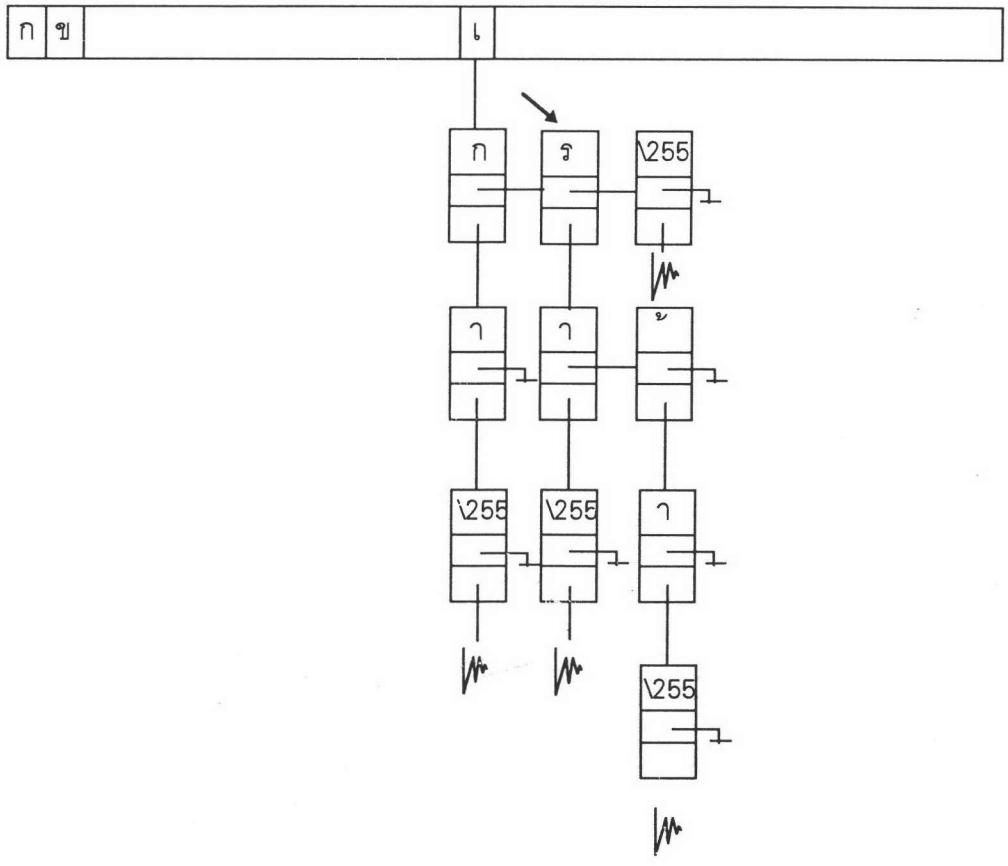
หลังจากได้ข้อความที่ผ่านการขยายคำย่อต่างๆแล้ว ก็นำข้อความนั้นมาค้นหาในต้นไม้คำศัพท์ โดยมีขั้นตอนดังนี้ เช่น ข้อความที่ต้องการอ่านออกเสียง คือ “เราก็วิ่งไล่.....”

ขั้นที่ 1 จะค้นหาตัวอักษร “เ” ใน array ตัวอักษร ก็จะได้ค่า index ของ array ตัวอักษรที่ชี้ไปยังตัวอักษร “เ” ดังรูปที่ 3.19



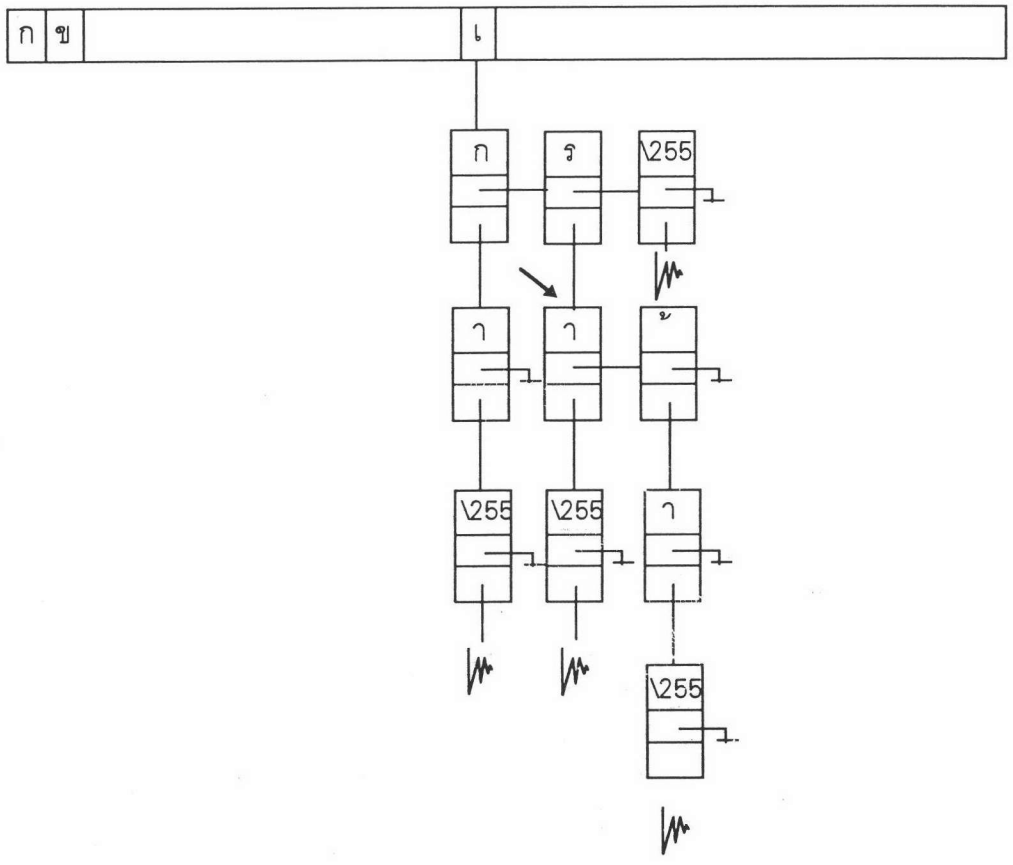
รูปที่ 3.19

ขั้นที่ 2 ค้นหาตัวอักษร “ร” ในโหนดระดับที่ 1 ปรากฏว่าพบ ดังรูปที่ 3.20



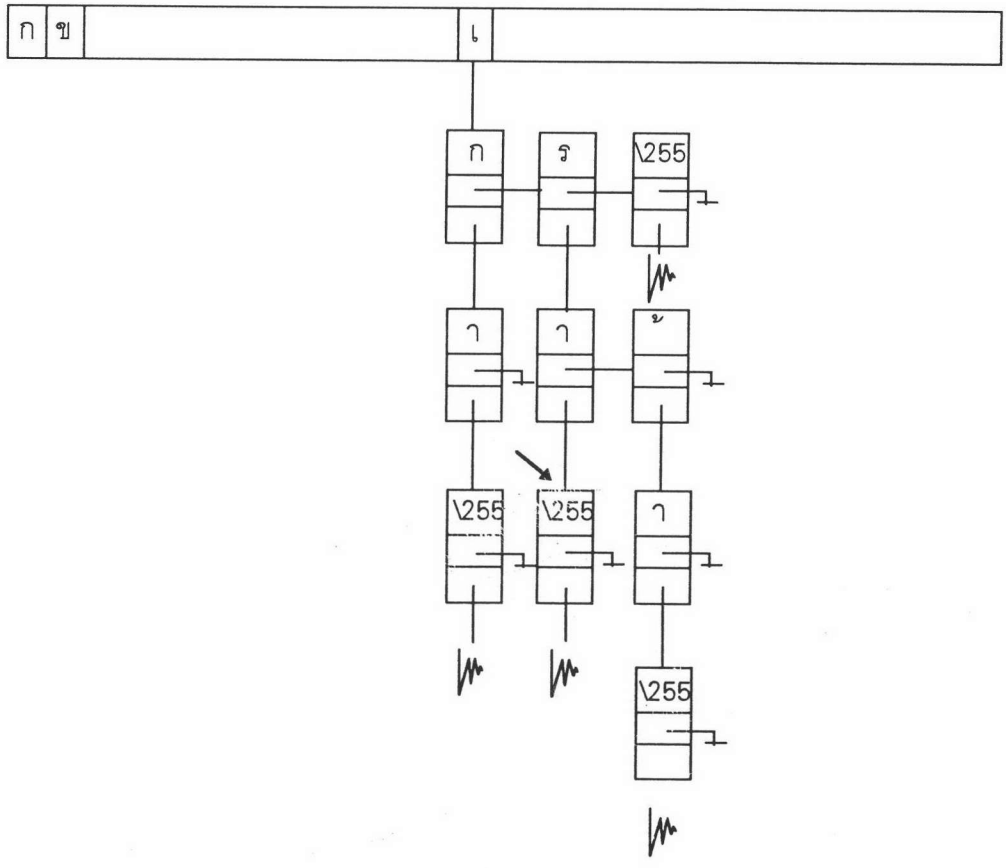
รูปที่ 3.20

ขั้นที่ 3 ค้นหาตัวอักษร “า” ในโหนดระดับที่ 2 ปรากฏว่าพบ ดังรูปที่ 3.21



รูปที่ 3.21

ขั้นที่ 4 ค้นหาตัวอักษร "ก" ในโหนดระดับที่ 3 ปรากฏว่าไม่พบ แสดงว่าคำสิ้นสุดที่นี่ ดังรูปที่ 3.22



รูปที่ 3.22

ขั้นที่ 5 ตรวจสอบดูว่าตัวอักษรของโหนดสุดท้ายที่พบมีค่าเท่ากับรหัสแอสกี 255 หรือไม่ ถ้าใช่ pointer ที่ชี้ลงของโหนดนั้น คือ pointer ที่ชี้ไปยังเสียงของคำศัพท์นั้น ในที่นี้คือคำว่า "เรา" ในแฟ้มข้อมูลเสียง ถ้าตัวอักษรของโหนดสุดท้ายที่พบไม่เท่ากับรหัสแอสกี 255 แสดงว่าไม่มีคำศัพท์นั้นในพจนานุกรมของระบบ

3.3.3 การอ่านออกเสียง

เมื่อได้ pointer ที่ชี้ไปยังข้อมูลเสียงคำศัพท์ที่ค้นหาแล้ว ระบบจะทำการอ่านข้อมูลเสียงจากแฟ้มข้อมูลเสียง ณ จุดที่ pointer ชี้เข้ามาเก็บไว้ใน buffer ที่เตรียมไว้ก่อนแล้ว แล้วจึงทำการอ่านออกเสียงจาก buffer นั้น