บทที่ 3

การวิเคราะห์และการออกแบบ

3.1 การวิเคราะห์และการออกแบบ

แนวทางการวิเคราะห์และออกแบบระบบการค้นคืนข้อความภาษาไทยโดยใช้ต้นไม้แพ็ตในวิทยานิพนธ์นี้ พิจารณาจากลักษณะข้อมูลและวิธีการนำไปใช้งานตลอดจนถึงวิธีการค้นหาข้อความจากเอกสาร สำหรับเนื้อหาที่จะกล่าวในบทนี้ประกอบด้วย ลักษณะเอกสาร การค้นหาข้อความในเอกสาร โครงสร้างข้อมูลต้นไม้แพ็ต แพ็ตอะเรย์ โดยแสดงให้เห็นถึงรูปแบบของโครงสร้างข้อมูล ขั้นตอนการเพิ่มข้อความ และการค้นคืนข้อความ

3.2 ลักษณะเอกสาร

เอกสารส่วนใหญ่ประกอบด้วยข้อความต่างๆ เช่น เนื้อหาความรู้งานวิจัย รายงานการประชุม ข่าวสาร เป็นต้น ซึ่งส่วนใหญ่แล้วมักมีโครงสร้างเอกสารที่ไม่แน่นอนตายตัว [5] และเอกสารที่มีข้อมูลต่างภาษากัน ยังมีโครงสร้างเอกสารที่แตกต่างกันอีกด้วย อย่างไรก็ตามเอกสารไม่ว่าภาษาใดๆ ก็ตามประกอบด้วยคำหรือตัวอักษรตามภาษานั้นๆ เรียงต่อๆ กันไปจนจบเอกสารชิ้นหนึ่งๆ และคำหรือตัวอักษรดังกล่าวมีตำแหน่งเริ่มต้นที่แน่นอน แต่การกำหนดตำแหน่งสิ้นสุดของคำหรือข้อความหนึ่งๆ จำเป็นต้องอาศัยกฎทางภาษาศาสตร์ในการแบ่งคำ สำหรับเอกสารภาษาอังกฤษ การกำหนดจุดแบ่งคำหรือตำแหน่งสิ้นสุดของคำหนึ่งๆ ค่อนข้างแน่นอน เนื่องจากหลักทางภาษาศาสตร์อาศัยช่องว่างเป็นตัวแบ่งคำ แต่สำหรับเอกสารภาษาไทยมักประสบปัญหาการกำหนดหรือแบ่งคำที่ชัดเจนอยู่เสมอๆ ดังนั้น การเลือกโครงสร้างข้อมูลสำหรับการค้นคืนข้อความภาษาไทยจำเป็นต้องพิจารณาทั้งลักษณะเอกสารและการแบ่งคำเป็นองค์ประกอบหลักด้วย

3.3 การค้นหาข้อความในเอกสาร

ระบบการค้นคืนข้อความในเอกสาร เป็นเครื่องมือที่สำคัญในการค้นหาข้อความในเอกสาร เนื่องจากสามารถช่วยให้การค้นหาข้อมูลได้สะดวกรวดเร็วขึ้น เพราะได้สร้างดรรชนีเตรียมไว้ก่อนล่วงหน้า โดยการเลือกโครงสร้างข้อมูลที่เหมาะสมกับการจัดเก็บดรรชนี และการค้นคืนข้อความในเอกสาร แตกต่างจากการที่มนุษย์ค้นหาข้อความในเอกสาร โดยอาศัยเอกสารดรรชนีที่สร้างขึ้นมาเอง ที่หากผู้ใดมีเทคนิคบวกกับความชำนาญในการใช้งานบ่อยๆ ในการเปิดเอกสารดรรชนีเพื่อค้นหาข้อความในเอกสาร จะค้นหาได้รวดเร็วขึ้น สำหรับระบบการค้นคืนข้อความในเอกสารไม่อาศัยประสบการณ์อย่างเช่นมนุษย์ หากแต่ความรวดเร็วในการค้นหาข้อความขึ้นอยู่กับโครงสร้างข้อมูลที่นำมาใช้ โดยมีปัจจัยหลัก 2 ประการ ประการแรก คือ ลักษณะโครงสร้างข้อมูลที่ใช้จัดเก็บข้อความในเอกสาร และประการที่ 2 คือ วิธีการค้นคืนข้อความจากโครงสร้างข้อมูลดังกล่าว ซึ่งระบบการค้นหาข้อความที่ดี สามารถลดความซ้ำซ้อน รวมทั้งประหยัดเนื้อที่ในการจัดเก็บดรรชนีที่ใช้ในการค้นหา และสามารถระบุได้ว่า ข้อความที่ค้นหาเก็บอยู่ในเอกสารใดบ้าง รวมทั้งอยู่ในตำแหน่งใดของเอกสาร นอกเหนือจากนั้นแล้ว ยังสามารถระบุได้ว่า ข้อความนั้นๆ มีระดับความสำคัญแค่ไหนในเอกสาร เป็นต้น

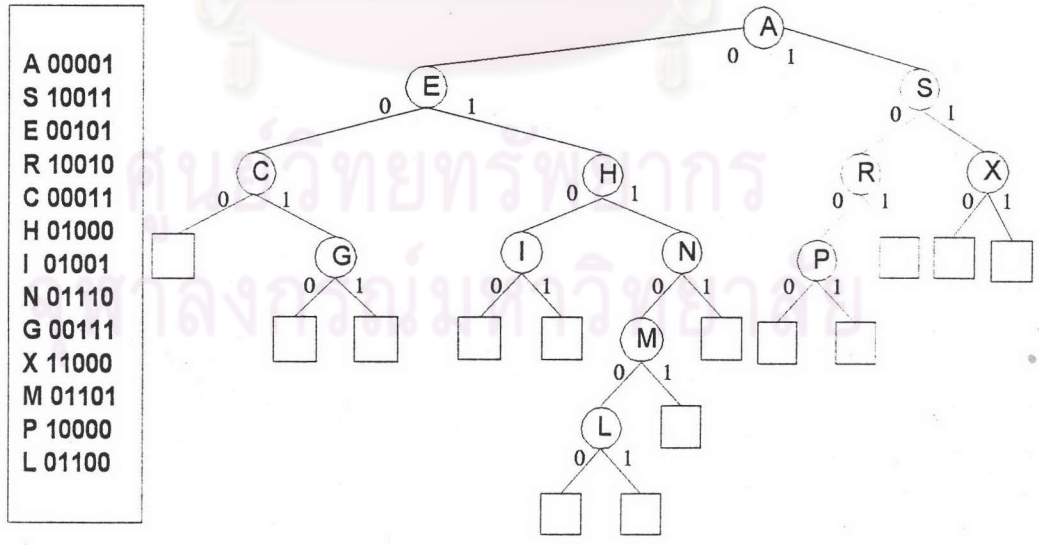
สำหรับวิธีการค้นหาข้อความในเอกสารนั้น สามารถใช้วิธีการแบบง่ายๆ ได้ ถ้าหากไม่คำนึงถึงความรวดเร็วในการค้นหา แต่โดยทั่วไปแล้วต้องการความรวดเร็ว จึงจำเป็นต้องนำ ระบบการค้นหาข้อความในเอกสารมาช่วย และระบบการค้นหาข้อความในเอกสารยังต้องนำไปใช้งานร่วมกับโปรแกรมประยุกต์ตัวอื่นๆ ดังนั้น การออกแบบโครงสร้างข้อมูลของระบบการค้นหาข้อความในเอกสารต้องคำนึงถึงความรวดเร็ว และปริมาณเนื้อที่ที่ใช้จัดเก็บดรรชนีของข้อความในเอกสาร เพื่อมิให้ขั้นตอนในการค้นหาข้อความในเอกสาร ซึ่งใช้งานร่วมกับโปรแกรมประยุกต์ตัวอื่นๆ ไปลดประสิทธิภาพในการทำงานร่วมกัน นอกจากนี้แล้วยังต้องคำนึงถึง หลักทางภาษาศาสตร์ ตามข้อความในเอกสารนั้นๆ อีกด้วย เพื่อนำมาพิจารณาออกแบบโครงสร้างข้อมูลที่มีประสิทธิภาพที่ดีที่สุดของระบบการค้นข้อความในเอกสาร

ปัจจุบันโปรแกรมประยุกต์ที่ถูกพัฒนาเป็นระบบการค้นหาข้อความในเอกสารมีอยู่มากมาย เช่น ระบบการค้นหาข้อความในเอกสารที่ใช้โครงสร้างข้อมูลดรรชนีผกผัน เป็นต้น สำหรับวิทยานิพนธ์ฉบับนี้ ได้อาศัยโครงสร้างข้อมูลต้นไม้แพ็ตเพื่อสร้างแพ็ตอะเรย์ ซึ่งมีหลักการในการพิจารณาข้อมูลในเอกสารเป็นสายอักขระที่ยาวเรียงต่อๆ กันไปจนจบเอกสารชิ้นหนึ่งๆ ซึ่งเรียกว่า

ซิสตริง โดยที่ซิสตริง คือสายอักขระที่มีตำแหน่งเริ่มต้นข้อความที่แน่นอน และตำแหน่งสุดท้ายของข้อความเป็นตำแหน่งเดียวกัน คือตำแหน่งสุดท้ายของเอกสารชิ้นนั้นๆ ซึ่งเมื่อพิจารณาหลักการนี้ พบว่า มีความเหมาะสมกับข้อความลักษณะที่เป็นภาษาไทย คือประโยคภาษาไทยประกอบด้วยคำหลายๆ คำเรียงต่อๆ กัน ซึ่งคำบางคำเป็นคำซ้อน และที่สำคัญไม่มีสัญลักษณ์การแบ่งคำที่ชัดเจน ซึ่งไม่เหมือนกับภาษาอังกฤษที่มีช่องว่างเป็นตัวแบ่งคำที่ชัดเจน แต่ทั้งนี้ทั้งนั้นหลักการของต้นไม้แพ็ต ยังสามารถนำมาใช้ร่วมกับข้อมูลที่มีตัวแบ่งคำชัดเจนได้อีกด้วย

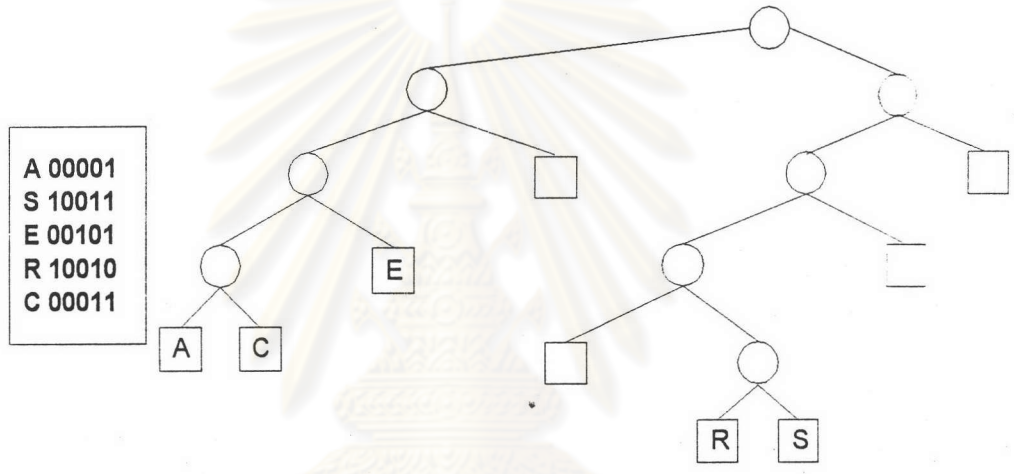
3.4 ต้นไม้ค้นหาแบบดิจิตอล (digital search tree)

ต้นไม้ค้นหาแบบดิจิตอล หมายถึงโครงสร้างข้อมูลแบบต้นไม้ที่มีการค้นหาแบบดิจิตอล [15] ต้นไม้ค้นหาแบบดิจิตอลประกอบด้วยโหนดประเภทต่างๆ เช่น โหนดราก โหนดใบ เป็นต้น กับเส้นทางเดินระหว่างโหนด ซึ่งได้จากผลลัพธ์การคำนวณระหว่างตัวเลขที่ได้จากการแปลงค่าตัวอักษรที่ประกอบเป็นดรรชนีหรือพรีฟิกซ์กับค่าที่เก็บในโหนดก่อน โดยโครงสร้างข้อมูลต้นไม้ดิจิตอลมีความแตกต่างจากโครงสร้างข้อมูลต้นไม้ที่ไม่ใช่ต้นไม้ดิจิตอล เช่น โครงสร้างข้อมูลต้นไม้แบบทวิภาค เป็นต้น ตรงที่เส้นทางเดินระหว่างโหนดของต้นไม้ดิจิตอลอาศัยผลลัพธ์จากการคำนวณระหว่างตัวเลข ไม่ได้ตรวจสอบจากการเปรียบเทียบค่าดรรชนีระหว่างโหนดกับค่าที่ต้องการค้นหาอย่างเช่นในต้นไม้ที่ไม่ใช่ดิจิตอล



รูปที่ 3.1 ตัวอย่างต้นไม้ค้นหาแบบดิจิตอล

รูปที่ 3.1 เป็นตัวอย่างต้นไม้ค้นหาแบบดิจิตอลที่สร้างขึ้นตามลำดับข้อมูล คือ A, S, E, R, C, H, I, N, G, X, M, P, L ตามลำดับ โดยการค้นหาข้อมูลในต้นไม้ยังคงอาศัยการเปรียบเทียบค่าที่ต้องการค้นหากับค่าดรรชนีของแต่ละโหนดที่เดินไปถึง จนกว่าจะพบข้อมูลในโหนดใดโหนดหนึ่ง ถ้าไม่พบจะเพิ่มข้อมูลเข้าไปในต้นไม้ ดังนั้นจะมีการเปรียบเทียบค่าดรรชนีกับค่าที่ต้องการค้นหาทุกครั้งที่เดินไปถึง ซึ่งต้นไม้ค้นหาแบบดิจิตอลยังสามารถลดขั้นตอนการเปรียบเทียบตรงจุดนี้ได้อีกด้วย โดยการเก็บค่าดรรชนีไว้ที่โหนดภายนอก เท่านั้น ดังรูปที่ 3.2



รูปที่ 3.2 ตัวอย่างต้นไม้ค้นหาแบบดิจิตอลลดการเปรียบเทียบดรรชนี

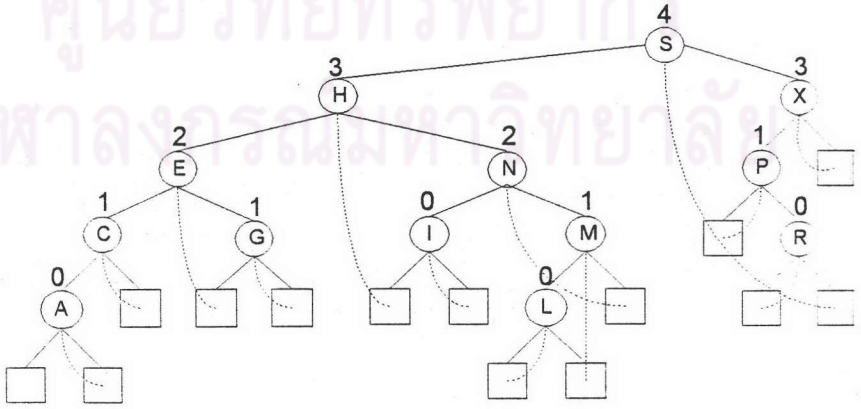
รูปที่ 3.2 จะเห็นว่าค่าดรรชนีถูกเก็บอยู่ที่โหนดสี่เหลี่ยม เรียกว่าโหนดภายนอกเท่านั้น ส่วนโหนดวงกลมไม่เก็บค่าดรรชนี เรียกว่าโหนดภายใน ดังนั้นการค้นหาข้อมูลในต้นไม้จะต้องค้นหาจากโหนดรากจนถึงโหนดภายนอกเสมอ เพื่อค้นหาค่าดรรชนีที่ใกล้เคียงหรือเหมือนกับค่าที่ต้องการค้นหา โดยการเปรียบเทียบกับค่าดรรชนีที่ค้นหาได้ที่โหนดภายนอกเพียงหนเดียวต่อการค้นหาหนึ่งครั้ง ซึ่งการค้นหาจะรวดเร็วกว่าต้นไม้ดิจิตอลตามรูปที่ 3.1 แต่ใช้เนื้อที่จัดเก็บต้นไม้มากกว่า เนื่องจากบางโหนดที่โหนดภายนอกจะเป็นโหนดว่างขึ้นอยู่กับลักษณะของข้อมูลที่นำมาสร้างต้นไม้ ซึ่งโหนดว่างเหล่านี้เกิดจากการที่ข้อมูลมีข้อมูลร่วมซ้ำๆ กันอยู่มาก

3.5 โครงสร้างข้อมูลต้นไม้แพ็ต

โครงสร้างข้อมูลต้นไม้แพ็ตจะนำข้อดีของต้นไม้ค้นหาแบบดิจิตอลที่ลดขั้นตอนการเปรียบเทียบค่าดรรชนีทุกๆ โหนดกับข้อมูลที่ต้องการค้นหา โดยเก็บค่าดรรชนีไว้ที่โหนดภายนอกแทน เพื่อการเปรียบเทียบเพียงหนเดียวต่อการค้นหาข้อมูลในต้นไม้หนึ่งครั้ง รวมกับการประหยัดเนื้อที่ที่จัดเก็บต้นไม้ โดยแก้ไขปัญหาของการเกิดโหนดว่างในต้นไม้ อันเกิดขึ้นตามลักษณะข้อมูลที่นำมาสร้าง (กรณีที่เกิดข้อมูลร่วมซ้ำๆ กันมาก) โดยต้นไม้แพ็ตได้อาศัยหลักการของบิตข้ามมาประยุกต์ใช้ ทำให้ลดขนาดต้นไม้ดิจิตอลลงได้

ต้นไม้แพ็ตเป็นต้นไม้ดิจิตอลที่ประกอบด้วยซิสตริงที่เป็นไปได้ของข้อมูลชิ้นนั้นๆ โดยมีแต่ละบิตของข้อมูลที่ต้องการค้นหาตามค่าบิตข้ามของแต่ละโหนดภายในเป็นตัวระบุให้เลือกเดินไปทางไหน ถ้าบิตข้อมูลตามค่าบิตข้ามเป็น 0 จะเดินไปทางซ้าย แต่ถ้าบิตนั้นเป็น 1 ให้เดินไปทางขวา จะเห็นว่าต้นไม้แพ็ตมีบิตข้ามเป็นตัวระบุว่าจะตรวจสอบค่าบิตใดของข้อมูลที่ต้องการค้นหาเพื่อเดินไปในต้นไม้ได้อย่างถูกต้อง โดยบางครั้งอาจข้ามบิตบางบิตที่ไม่สนใจไปได้ทำให้การค้นหาเร็วขึ้น อีกทั้งยังใช้เนื้อที่จัดเก็บน้อยกว่าต้นไม้ดิจิตอลที่ไม่มีหลักการของบิตข้ามอีกด้วย ซึ่งต้นไม้แพ็ตมีคุณสมบัติเบื้องต้น ดังนี้ (รูปที่ 3.3 แสดงตัวอย่างต้นไม้แพ็ต)

1. ต้นไม้ย่อยไม่มีโหนดว่าง (null node)
2. ถ้าโหนดภายในเป็น 0 จะแยก (branch) ไปต้นไม้ย่อย (subtree) ข้างซ้าย แต่ถ้าโหนดภายในเป็น 1 จะแยกไปต้นไม้ย่อยข้างขวา
3. โหนดภายนอกเก็บค่าตัวชี้ที่ชี้ไปยังข้อมูล



รูปที่ 3.3 ตัวอย่างต้นไม้แพ็ต

สำหรับขั้นตอนการเพิ่มหรือการค้นหาข้อมูลของโครงสร้างข้อมูลต้นไม้แพ็ตได้มาจากการสร้างเส้นทางที่เริ่มต้นจากโหนดรากจนกระทั่งถึงตำแหน่งสุดท้ายของข้อมูล โดยการค้นหาข้อมูลในต้นไม้จะเริ่มค้นที่โหนดรากลงไปตามเส้นทางเดินของต้นไม้ โดยอาศัยค่าดรรชนีบิต (bit index) ในแต่ละโหนดเป็นตัวระบุว่าจะตรวจสอบค่าบิตใดในข้อมูลที่ต้องการจะค้นหา โดยถ้าค่าบิตมีค่าเป็น 1 จะเดินไปเส้นทางขวา แต่ถ้าค่าบิตเป็น 0 จะเดินไปเส้นทางซ้าย และข้อมูลของทุกโหนดตลอดเส้นทางที่เดินผ่านจะไม่ถูกตรวจสอบทั้งหมด จนกระทั่งถึงโหนดที่มีตัวชี้ขึ้น ซึ่งตัวชี้ขึ้นจะชี้โหนดที่เก็บข้อมูลในต้นไม้ที่ใช้ในการเปรียบเทียบกับข้อมูลที่ต้องการจะค้น ซึ่งเป็นข้อมูลตามค่าบิตที่เดินไปตามข้อมูลที่ต้องการจะค้น ตัวอย่างเช่น S เป็นข้อมูลที่ถูกเก็บอยู่ในต้นไม้ที่มีแบบ (Pattern) เป็น 10*11 ซึ่งได้จากการตรวจสอบบิตข้ามตามเส้นทางเดิมของต้นไม้แพ็ตจากโหนดรากลงไปจนถึงโหนดภายนอกที่ชี้ไปยังโหนด S คือพิจารณาข้อมูลตามบิตข้ามที่ 4, 3, 1, 0 ตามลำดับ โดยไม่สนใจบิตที่ 2 ดังนั้น ถ้าข้อมูลของโหนดที่ถูกชี้โดยตัวชี้ขึ้น มีค่าเท่ากับข้อมูลที่ต้องการจะค้นหาแล้วแสดงว่าการค้นหานั้นเป็นผลสำเร็จ แต่ถ้าไม่ใช่แสดงว่าการค้นหานั้นไม่เป็นผลสำเร็จ

ต้นไม้แพ็ตจะมีค่าดรรชนีบิตของแต่ละโหนดในต้นไม้จะลดลงตามระดับของต้นไม้ และการค้นหาในต้นไม้แพ็ตครั้งหนึ่งๆ จะสิ้นสุดที่โหนดที่ถูกชี้โดยตัวชี้ขึ้น แล้วนำค่าที่เก็บในโหนดนี้ไปเปรียบเทียบกับค่าที่ต้องการจะค้นหา รูปที่ 3.4 แสดงฟังก์ชันการค้นหาข้อมูลในต้นไม้แพ็ต

```
static struct node
        { int key, info, b; struct node *l, *r; };
static struct node *head;
int patriciasearch(int v)
        {
        struct node *p, *x;
        p = head; x = head->l;
        while (p->b > x->b)
            {
                p = x;
                x = (bits(v, x->b, 1)) ? x->r : x->l;
            }
```

```
    if (v == x->key) return x->info; else return -1;
}
```

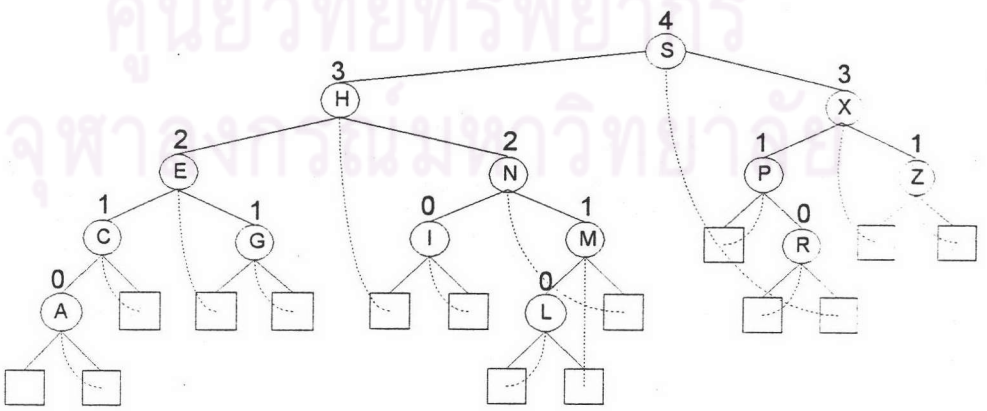
รูปที่ 3.4 ฟังก์ชันการค้นหาข้อมูลในต้นไม้แพ็ต

ฟังก์ชันนี้ทำหน้าที่หาข้อมูลที่เก็บค่าตามตัวแปร V เช่น ถ้าต้องการจะค้นหา Z ที่มีรหัส 11010 ในรูปที่ 3.3 เริ่มต้นจะเดินไปตามเส้นทางขวา แล้วเดินต่อไปตามเส้นทางทางขวาของโหนด X ซึ่งจะพบกับตัวชี้ขึ้นและตัวชี้ขึ้นตัวนี้ชี้ไปยังโหนด X แล้วนำค่าโหนด X ไปเปรียบเทียบกับข้อมูลที่ต้องการจะค้นหา คือ Z พบว่าไม่เท่ากันดังนั้นการค้นหาข้อมูล Z จึงไม่เป็นผลสำเร็จ

สำหรับการเพิ่มข้อมูลในต้นไม้ [15] แบ่งได้เป็น 2 แบบ คือ

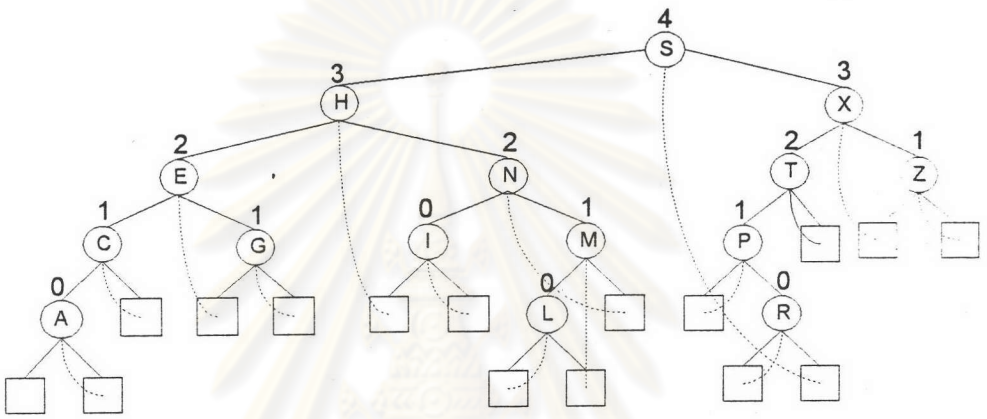
1. การเพิ่มข้อมูลที่โหนดภายนอกของต้นไม้
2. การเพิ่มข้อมูลที่โหนดภายในของต้นไม้

การเพิ่มข้อมูลที่โหนดภายนอกของต้นไม้ เริ่มจากการค้นหาข้อมูลที่ต้องการจะเพิ่มในต้นไม้ก่อน ถ้าไม่พบจึงจะเพิ่มเข้าในต้นไม้ แต่ถ้าพบจะไม่เก็บข้อมูลซ้ำเข้าไปในต้นไม้ เช่น การเพิ่ม Z ที่มีรหัส 11010 เข้าไปในรูปที่ 3.3 ตามที่ได้อธิบายไว้ข้างต้นเกี่ยวกับการค้นหาข้อมูลในต้นไม้ พบว่าการค้นหา Z สิ้นสุดที่โหนด X ถ้าเพิ่ม Z เข้าไปจะต้องเปลี่ยนตัวชี้ขึ้นให้ชี้ไปยังโหนด Z โหนดใหม่แล้วตัวชี้ขึ้นสองตัวของโหนดใหม่นี้ ตัวหนึ่งจะชี้ขึ้นไปยังโหนด X และอีกตัวหนึ่งจะชี้ขึ้นไปยังโหนด Z ตามค่าดรรชนีบิตที่ได้จากการเปรียบเทียบค่าบิตที่แตกต่างกันของข้อมูล X กับข้อมูล Z ส่งผลให้โหนดที่เป็นโหนดภายนอกกลายเป็นโหนดภายในและโหนดใหม่ที่เพิ่มเข้าไปในต้นไม้กลายเป็นโหนดภายนอกแทน ดังรูปที่ 3.5



รูปที่ 3.5 การเพิ่มข้อมูลที่โหนดภายนอกของต้นไม้แพ็ต

การเพิ่มข้อมูลที่โหนดภายในของต้นไม้คล้ายกับแบบแรกแต่ซับซ้อนกว่า สมมติว่าต้องการเพิ่มข้อมูล T ที่มีรหัส 10100 จากการค้นหาข้อมูล T พบว่าการค้นหาสิ้นสุดที่ P คือ 10000 ซึ่งมีรูปแบบเป็น 10*0* (ไม่สนใจบิตที่ 2 กับ 0) จากการเปรียบเทียบข้อมูล T กับ P พบว่าค่าบิตแตกต่างกันที่บิตที่สอง และจากคุณสมบัติที่ว่าค่าดรรชนีบิตจะลดลงตามระดับของต้นไม้ ดังนั้นถ้าเพิ่มข้อมูล T จะต้องเพิ่มระหว่างโหนด X กับ P และมีตัวชี้ขึ้นชี้ไปยังโหนดตัวเอง ส่งผลให้ทราบว่าโหนด P กับโหนด R มีค่าบิตที่สองเหมือนกัน ดังรูปที่ 3.6



รูปที่ 3.6 การเพิ่มข้อมูลที่โหนดภายในของต้นไม้แพ็ต

สำหรับฟังก์ชันการเพิ่มข้อมูลทั้งสองแบบสามารถใช้ฟังก์ชันเดียวกันได้ดังรูปที่ 3.7

```
int patriciainseart(int v, int info)
        {
        struct node *p, *t, *x;
        int i = maxb;
        p = head; t = head->l;
        while (p->b > t->b)
           { p = t; t = (bits(v, t->b, 1)) ? t->r : t->l; }
        if (v == t->key) return;
        while (bits(t->key, i, 1) == bits(v, i, 1)) i--;
        p = head; x = head->l;
        while (p->b > x->b && x->b > i)
```

```
        { p = x; x = (bits(v, x->b, 1)) ? x->r : x->l; }
    t = (struct node *) malloc(sizeof *t);
    t->key = v; t->info = info; t->b = i;
    t->l = (bits(v, t->b, 1)) ? x : t;
    t->r = (bits(v, t->b, 1)) ? t : x;
    if (bits(v, p->b, 1)) p->r = t; else p->l = t;
}
```

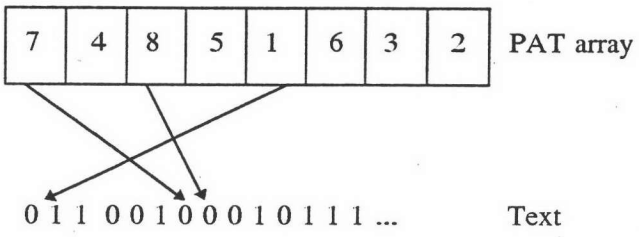
รูปที่ 3.7 ฟังก์ชันการเพิ่มข้อมูลในต้นไม้แพ็ต

3.6 การแทนต้นไม้แพ็ตด้วยอะเรย์

เนื่องจากต้นไม้แพ็ตค่อนข้างใช้เนื้อที่ในการจัดเก็บมาก คือขนาดของต้นไม้ขึ้นอยู่กับจำนวนโหนดทั้งหมด (ทั้งโหนดภายในและโหนดภายนอก) ซึ่งโหนดแต่ละโหนดจะต้องเก็บ

1. บิตข้าม (skip bit)
2. ดรรชนีชี้ทางซ้าย (left pointer)
3. ดรรชนีชี้ทางขวา (right pointer)
4. ดรรชนีชี้ซิสตริง (position of Sistring)

จะเห็นว่าสามารถจะลดขนาดของโหนดแต่ละโหนดได้ โดยการนำเทคนิคของอะเรย์เข้ามาช่วยแก้ไขปัญหาดังกล่าวเพื่อให้มีประสิทธิภาพมากขึ้น ซึ่งผู้คิดค้นวิธีการนี้ คือ Manber and Myers [11] และได้ตั้งชื่อว่าอะเรย์เติมหลัง (suffix array) หรือแพ็ตอะเรย์ (PAT array) โดยมีหลักการ คือนำโหนดภายนอกของต้นไม้แพ็ตมาเก็บลงในอะเรย์แถวเดี่ยว (single array) เรียงต่อๆ กัน ไปผลที่ได้ คือซิสตริงต่างๆ ถูกเก็บเรียงต่อกันไปตามลำดับอยู่ในอะเรย์ ดังตัวอย่างการแทนต้นไม้แพ็ตจากรูปที่ 2.6 ด้วยอะเรย์ ดังนี้



รูปที่ 3.8 การแทนต้นไม้แพ็ตด้วยอะเรย์

3.7 การค้นหาในแพ็ตอะเรย์ (PAT array)

การค้นหาในแพ็ตอะเรย์แบบเติมหน้าและแบบช่วงนั้นสามารถทำได้ในลักษณะเดียวกันกับการค้นหาในต้นไม้แพ็ต คืออาศัยวิธีการค้นแบบทวิภาคโดยอ้อม (indirect binary search) โดยทำการค้นแบบทวิภาค (binary search) กับอะเรย์ของโหนดภายนอก ซึ่งเป็นดรรชนีของข้อมูล และมีประสิทธิภาพเป็น O(log n) [16] โดยที่ n คือ จำนวนโหนดทั้งหมดในต้นไม้

ส่วนในการค้นหาแบบข้างเคียงและการค้นหาแบบนัยสำคัญสูงสุดหรือความถี่สูงสุดจำเป็นต้องอาศัยโครงสร้างข้อมูลอื่นเพิ่มเติม ดังรายละเอียดในเอกสารของ Manber and Myers [11]