บทที่ 3

ทฤษฎีเพทริเน็ต

<u>ความเป็นมา</u>

ทฤษฎีเพทริเน็ตเป็นทฤษฎีที่ริเริ่มขึ้นในช่วงคริสตศักราชที่ 1962 โดย Carl Adam Petri ในงานวิทยานิพนธ์ปริญญาเอกของเขา ซึ่งให้แนวความคิดพื้นฐานสำหรับการติดต่อสื่อสารระหว่างอุปกรณ์แบบอะซิงโครนัส (Asynchronous) ของระบบคอมพิวเตอร์ โดยบรรยายความสัมพันธ์ระหว่างเหตุการณ์ที่เกิดขึ้นในระบบด้วยทฤษฎีเพทริเน็ต ในปี ค.ศ. 1965 A.W. Holt และคนอื่น ๆ ในโครงการทฤษฎีระบบสารสนเทศ (Information System Theory Project) ของสถาบัน Applied Data Research (ADR) สนใจงานวิทยานิพนธ์ของ Petri และ พัฒนาสร้างข้อกำหนดและรูปแบบของเพทริเน็ต รวมทั้งได้แสดงให้เห็นว่าเพทริเน็ตสามารถประยุกต์ใช้กับระบบคอนเคอเร็นท์ (Concurrent System) [6] ต่อมามีการวิจัยเกี่ยวกับทฤษฎีเพทริเน็ตในระบบต่างๆ กันอย่างกว้างขวาง เช่น ระบบโพโทคอลสื่อสาร (Communication Protocol) ระบบฐานข้อมูลแบบกระจาย (Distribution Database System) ระบบมัลติโปรเซสเซอร์ (Multiprocessor System) และระบบอุตสาหกรรมการผลิต (Manufacturing System) เป็นต้น [7,8]

<u>นิยามพื้นฐาน</u>

ในการศึกษาทฤษฎีเพทริเน็ตจำเป็นต้องทราบถึงนิยามพื้นฐานของเพทริเน็ต เพื่อใช้ในการหาแบบจำลองและวิเคราะห์ระบบที่ซับซ้อน นิยามพื้นฐานของเพทริเน็ตมีรากฐานมาจากทฤษฎีของแบ็ก (Bag Theory) [ภาคผนวก ก] โดยจะกล่าวถึงโครงสร้างและกฎการทำงานของเพทริเน็ต

1. <u>โครงสร้างของเพทริเน็ต</u> [6,8,9]

นิยามที่ 3.1 : โครงสร้างพื้นฐานโดยทั่วไปของเพทริเน็ต PN = (P, T, I, O, M_0) โดยที่

(i) $P = \{p_1, p_2, ..., p_n\}$,n เป็นเลขจำนวนเต็มที่ $n \geq 0$

(ii) $T = \{t_1, t_2, ..., t_m\}$,m เป็นเลขจำนวนเต็มที่ $m \geq 0$

(iii) $P \cap T = \varnothing$

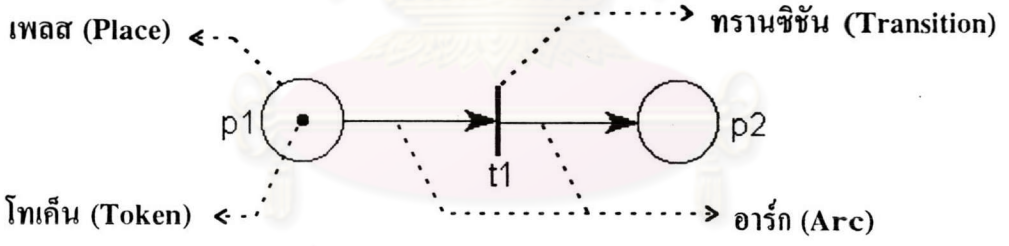
(iv) $I : (P \times T) \rightarrow N$ เป็นอินพุตฟังก์ชัน (Input Function) : O⟶|

(v) $O : (P \times T) \rightarrow N$ เป็นเอาต์พุตฟังก์ชัน (Output Function) : |⟶O

โดยที่ $N = \{0, 1, 2, ...\}$

(vi) $M_0 : P \rightarrow N$ เป็นมาร์กกิงเริ่มต้น (Initial Marking)

เพทริเน็ตประกอบด้วยองค์ประกอบพื้นฐานเพียงโนด (node) 2 ชนิด คือ เพลส (Place) ซึ่งแทนด้วยรูปวงกลม และ ทรานซิชัน (Transition) ซึ่งแทนด้วยรูปแท่งสี่เหลี่ยม โดยโนดทั้งสองถูกเชื่อมต่อกันด้วยอาร์ก (Arc) ดังแสดงในรูปที่ 3.1



รูปที่ 3.1 องค์ประกอบพื้นฐานของเพทริเน็ต

จากรูปที่ 3.1 เพลส p_1 เป็นอินพุตเพลสของทรานซิชัน t_1 เนื่องจากมีอาร์กชี้จากเพลส p_1 มายังทรานซิชัน t_1 และเพลส p_2 เป็นเอาต์พุตเพลสของทรานซิชัน t_1 เนื่องจากมีอาร์กชี้จากทรานซิชัน t_1 มายังเพลส p_2 ฟังก์ชันอินพุตของทรานซิชัน t_1 ($I(t_1)$) คือ เซ็ตของอินพุตเพลสของทรานซิชัน t_1 ทั้งหมด และ ฟังก์ชันเอาต์พุตของทรานซิชัน t_1 ($O(t_1)$) คือ เซ็ตของเอาต์พุตเพลสของทรานซิชัน t_1 ทั้งหมด

มาร์กกิงเริ่มต้น M_0 แทนจำนวนโทเค็นในแต่ละเพลสในขณะเริ่มต้นการทำงาน ซึ่งการทำงานของเพทริเน็ตถูกควบคุมด้วยการกระจายของโทเค็น ในรูปที่ 3.1 $M_0 = (1, 0)$

ตัวอย่างที่ 3.1 แสดงโครงสร้างของแบบจำลองเพทริเน็ต

$PN = (P, T, I, O, M_0)$

$P = \{p_1, p_2, p_3, p_4, p_5\}$

$T = \{t_1, t_2, t_3, t_4\}$

$I(t_1) = \{p_1, p_5\}$ $O(t_1) = \{p_3\}$

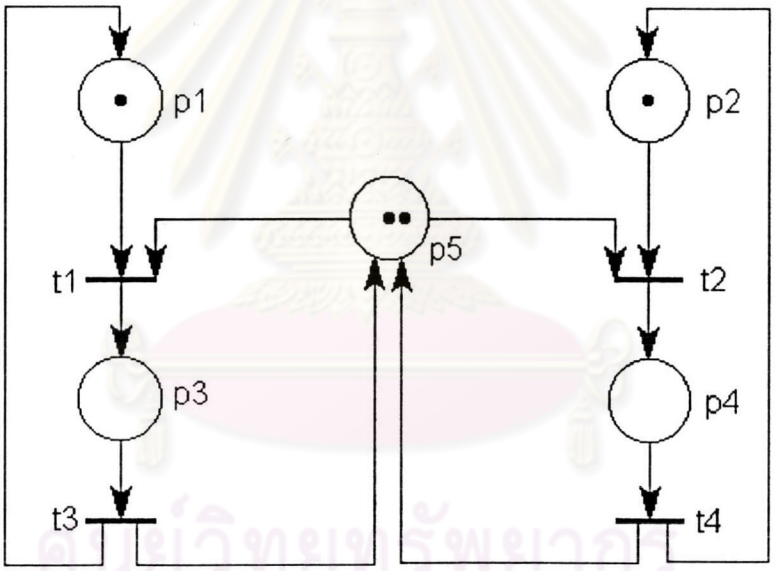
$I(t_2) = \{p_2, p_5\}$ $O(t_2) = \{p_4\}$

$I(t_3) = \{p_3\}$ $O(t_3) = \{p_1, p_5\}$

$I(t_4) = \{p_4\}$ $O(t_4) = \{p_2, p_5\}$

$M_0 = (1, 1, 0, 0, 2)$

จากโครงสร้างนี้แสดงแบบจำลองเพทริเน็ตได้ดังรูปที่ 3.2



รูปที่ 3.2 แบบจำลองเพทริเน็ต

2. <u>กฎการทำงานของเพทริเน็ต</u> [6,8]

การทำงานของเพทริเน็ตถูกควบคุมโดยจำนวนและการกระจายของโทเค็นในแต่ละเพลส เพทริเน็ตทำงานโดยการยิงทรานซิชัน (firing transition) ทรานซิชันยิงโดยการขจัดโทเค็นจากอินพุตเพลสของทรานซิชัน และสร้างโทเค็นใหม่ที่เอาต์พุตเพลสของทรานซิชันนั้น ซึ่งแสดงรายละเอียดดังนิยามที่ 3.2 และ 3.3

นิยามที่ 3.2 : ทรานซิชัน $t_j \in T$ ในเพทริเน็ตถูกอีนาเบิล (Enable) ก็ต่อเมื่อ

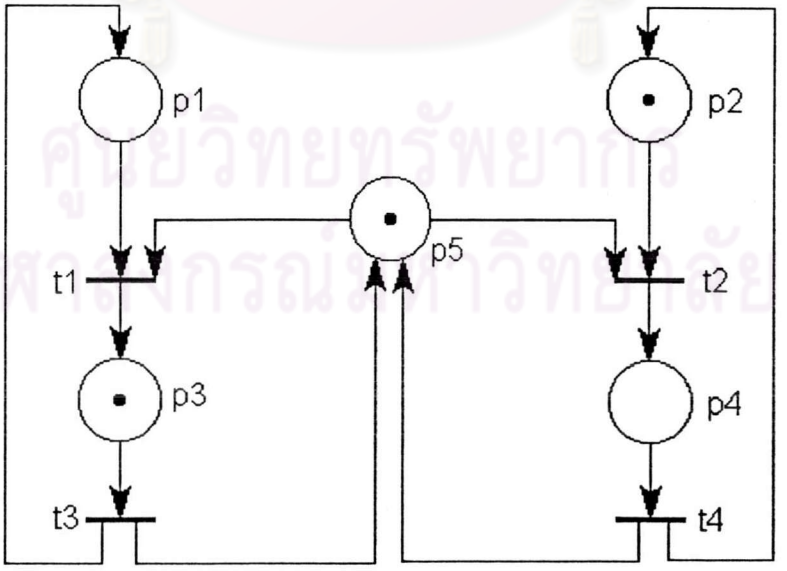
$$M(p_i) \geq \#(p_i, I(t_j)) \qquad ,\forall p_i \in P$$

นิยามที่ 3.3 : ทรานซิชันซึ่งถูกอีนาเบิลสามารถยิงได้โดยขจัดโทเค็นจากแต่ละอินพุตเพลส และสร้างโทเค็นใหม่ในแต่ละเอาต์พุตเพลสของทรานซิชันนั้น ซึ่งจะได้ผลลัพธ์เป็นมาร์กกิงใหม่ M′ เมื่อ

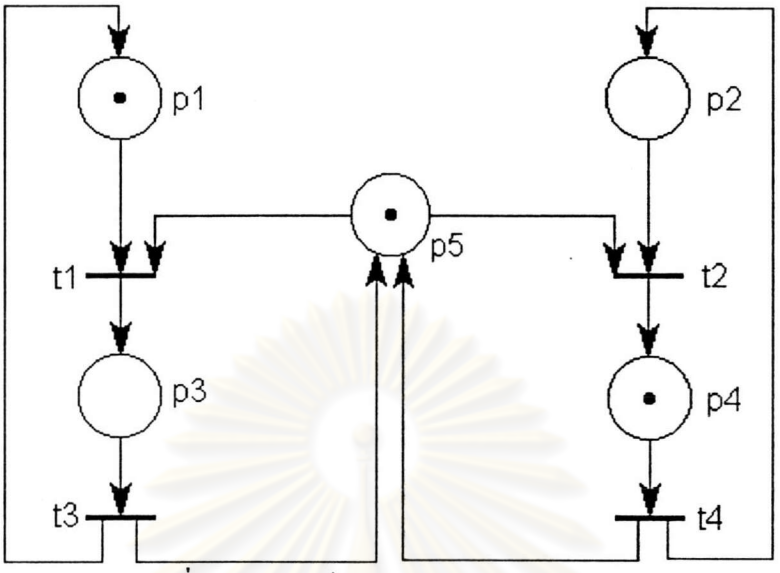
$$M'(p_i) = M(p_i) - \#(p_i, I(t_j)) + \#(p_i, O(t_j)) \qquad (3\text{-}1)$$

ทรานซิชันจะยิงได้ก็ต่อเมื่อทรานซิชันนั้นถูกอีนาเบิล นั่นหมายความว่าทุก ๆ อินพุตเพลสของทรานซิชันนั้นมีจำนวนโทเค็นอยู่อย่างน้อยเท่ากับจำนวนอาร์กที่ชี้มายังทรานซิชันนั้น ซึ่งในการยิงของทรานซิชันจะทำให้เกิดการเปลี่ยนแปลงในมาร์กกิงของเพทริเน็ตไปเป็นมาร์กกิงใหม่ M′ และ การยิงของทรานซิชันสามารถดำเนินต่อไปเรื่อย ๆ ตราบใดที่ยังมีทรานซิชันอย่างน้อยหนึ่งทรานซิชันที่อีนาเบิลอยู่ แต่ถ้าไม่มีทรานซิชันใดอีนาเบิลเลยการทำงานของเพทริ-เน็ตก็จะหยุดทำงาน

ตัวอย่างที่ 3.2 จากรูปที่ 3.2 จะเห็นได้ว่ามี 2 ทรานซิชันที่อีนาเบิล คือ ทรานซิชัน t_1 และ t_2 ดังนิยามที่ 3.2 ซึ่งแสดงผลลัพธ์จากการยิงของทรานซิชันได้ดังรูปที่ 3.3 และ 3.4



รูปที่ 3.3 ผลลัพธ์จากการยิงของทรานซิชัน t_1



รูปที่ 3.4 ผลลัพธ์จากการยิงของทรานซิชัน t_2

3. ปริภูมิสถานะของเพทริเน็ต (Petri Net State Space) [6]

สถานะของเพทริเน็ตถูกกำหนดโดยมาร์กกิงของเพทริเน็ต การยิงของทรานซิชันแทนการเปลี่ยนสถานะของเพทริเน็ต โดยการเปลี่ยนมาร์กกิงของเพทริเน็ต ปริภูมิสถานะของเพทริเน็ตซึ่งมี n เพลส เป็นเซ็ตของมาร์กกิงทั้งหมด คือ N^n โดยที่ N เป็นเซ็ตของจำนวนเต็มบวกใด ๆ กับ จำนวนเต็มศูนย์ การเปลี่ยนสถานะเนื่องจากการยิงของทรานซิชันซึ่งถูกกำหนดโดยฟังก์ชันการเปลี่ยนสถานะ δ (next-state function)

นิยามที่ 3.4 : ฟังก์ชันการเปลี่ยนสถานะ $\delta: N^n \times T \rightarrow N^n$ สำหรับ $PN = (P, T, I, O, M_0)$ และทรานซิชัน $t_j \in T$ ถูกกำหนด ก็ต่อเมื่อ

$$M(p_i) \geq \#(p_i, I(t_j)) \qquad ,\forall p_i \in P$$

ถ้า $\delta(M,t_j)$ ถูกกำหนด ดังนั้น $\delta(M,t_j) = M'$ ดังสมการที่ (3-1)

นิยามที่ 3.5 : สำหรับเพทริเน็ต $PN = (P, T, I, O, M_0)$ มาร์กกิง M' เป็นอิมมีเดียตลีรีชเอเบิล (Immediately Reachable) จาก M_0 ถ้ามีทรานซิชัน $t_j \in T$ ซึ่ง $\delta(M_0,t_j) = M'$

เราสามารถขยายแนวความคิดนี้โดยกำหนดเซ็ตของรีชเอเบิลมาร์กิง(Reachable Marking) สำหรับเพทริเน็ตที่ให้ ถ้า M' เป็นอิมมีเดียตลีรีชเอเบิลจาก M_0 และ M'' เป็นอิมมีเดียตลีรีชเอเบิลจาก M' ดังนั้นเราสามารถเรียก M'' ว่าเป็นรีชเอเบิล (Reachable) จาก M_0 เรากำหนด

ให้เซ็ตของรีชเอเบิลมาร์กกิง($R(M_0)$) ของเพทริเน็ตเป็นมาร์กกิงทั้งหมดซึ่งเป็นรีชเอเบิลจาก M_0 ถ้ามีลำดับของการยิงทรานซิชันซึ่งทำให้เกิดการเปลี่ยนแปลงจากมาร์กกิง M_0 ไปสู่มาร์กกิง M'' แสดงว่ามาร์กกิง M'' อยู่ในเซ็ตของรีชเอเบิลมาร์กกิง

นิยามที่ 3.6 : ส่วนขยายของฟังก์ชันเปลี่ยนสถานะถูกกำหนดสำหรับมาร์กกิง M และ ลำดับของทรานซิชัน $\sigma \in T$ โดย

$$\delta(M,t_j\sigma) = \delta(\delta(M,t_j) , \sigma) \qquad (3\text{-}2)$$

ตัวอย่างที่ 3.3 จากโครงสร้างของเพทริเน็ตในตัวอย่างที่ 3.1 โดยมี $M_0 = (1, 1, 0, 0, 2)$ มีทรานซิชัน t_1 และ t_2 ถูกอีนาเบิล ดังนั้นมี 2 อิมมีเดียตลีรีชเอเบิล คือ (0, 1, 1, 0, 1) กับ (1, 0, 0, 1, 1)

ถ้าลำดับของทรานซิชันเป็น t_1 t_2 t_3 ก็จะได้ผลของมาร์กกิงเป็น (1, 0, 0, 1, 1)

4. <u>การแทนแบบจำลองเพทริเน็ตด้วยเมตริกซ์</u> [6]

แบบจำลองเพทริเน็ตสามารถแทนด้วยเมตริกซ์หนึ่งที่เรียกว่า อินซิเด็นท์เมตริกซ์ (Incident Matrix : C) โดยกำหนดผ่านเมตริกซ์ 2 เมตริกซ์ได้แก่ C^+ และ C^- ซึ่งแทนเมตริกซ์ของเอาต์พุตฟังก์ชันและอินพุตฟังก์ชันตามลำดับ สำหรับแบบจำลองเพทริเน็ตที่มี n ทรานซิชัน และ m เพลส อินซิเด็นท์เมตริกซ์ที่ใช้แทนแบบจำลองเพทริเน็ตนี้มีขนาด $n \times m$ โดยมีค่าดังสมการที่ (3-3)

$$C = C^+ - C^- \qquad (3\text{-}3)$$

จากรูปที่ 3.2 เราสามารถแทนแบบจำลองเพทริเน็ตด้วยเมตริกซ์ดังนี้

$$C^+ = \begin{array}{c} \begin{matrix} P_1 & P_2 & P_3 & P_4 & P_5 \end{matrix} \\ \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix} \end{array} \begin{matrix} t_1 \\ t_2 \\ t_3 \\ t_4 \end{matrix} \qquad (3\text{-}4)$$

$$C^- = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \qquad (3\text{-}5)$$

$$\mathbf{C} = \begin{bmatrix} -1 & 0 & 1 & 0 & -1 \\ 0 & -1 & 0 & 1 & -1 \\ 1 & 0 & -1 & 0 & 1 \\ 0 & 1 & 0 & -1 & 1 \end{bmatrix} \quad (3\text{-}6)$$

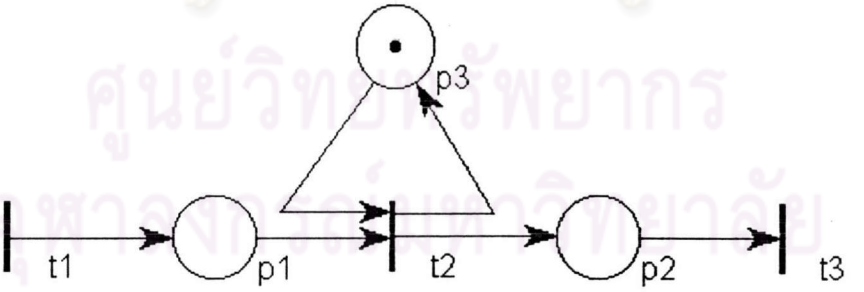
คุณสมบัติของเพทริเน็ต [6,7,8]

1. Safeness

คุณสมบัติ Safeness หมายถึง การที่ไม่มีจำนวนโทเค็นในเพลสใดเลยที่มีค่ามากกว่าหนึ่งตลอดการทำงานของเพทริเน็ตนั้น

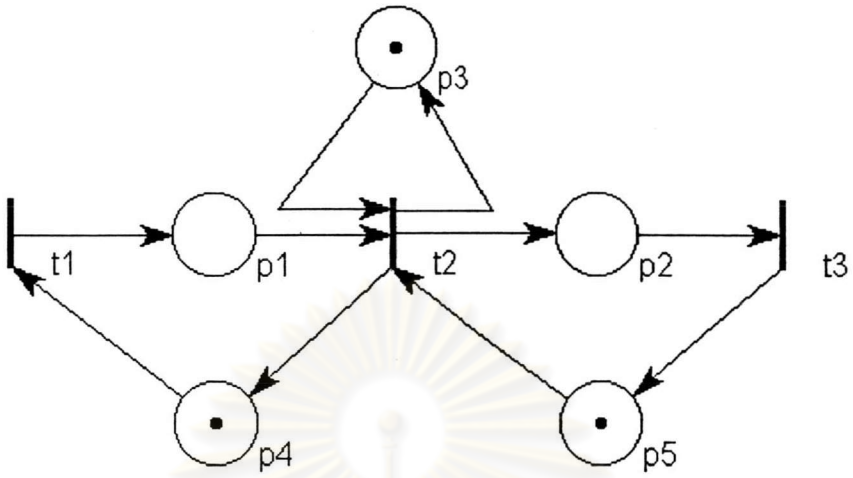
นิยามที่ 3.7 : เพลส $p_i \in P$ ของเพทริเน็ต $PN = (P, T, I, O, M_0)$ มีคุณสมบัติ safeness ถ้า $\forall M' \in R(M_0)$, $M'(p_i) \leq 1$ และ เพทริเน็ตมีคุณสมบัติ safeness ถ้าแต่ละเพลสในเพทริเน็ตนั้นมีคุณสมบัติ safeness

ในการนำเพทริเน็ตไปทำการใช้ในอุปกรณ์ฮาร์ดแวร์จริงนั้นคุณสมบัตินี้เป็นคุณสมบัติที่จำเป็นมากข้อหนึ่ง เนื่องจากถ้าเพลสหนึ่งมีคุณสมบัติ Safeness แสดงว่าจำนวนโทเค็นในเพลสนั้นมีค่าเป็น 0 หรือ 1 ดังนั้นเพลสนั้นสามารถถูกสร้างด้วยฟลิบฟลอบ (Flip-Flop)



รูปที่ 3.5 แบบจำลองเพทริเน็ตที่ไม่มีคุณสมบัติ Safeness

จากรูปที่ 3.5 แสดงแบบจำลองเพทริเน็ตที่ไม่มีคุณสมบัติ Safeness เนื่องจากทรานซิชัน t_1 อีนาเบิลตลอดเวลาไม่ว่าจะมีการกระจายของโทเค็นเป็นอย่างไร ดังนั้นถ้ามีลำดับการยิงทรานซิชันเป็น t_1t_1 จะทำให้เพลส p_1 มีจำนวนโทเค็นเกินหนึ่งได้ ซึ่งเราสามารถแก้ไขให้แบบจำลองเพทริเน็ตนี้มีคุณสมบัติ Safeness ได้ดังรูปที่ 3.6



รูปที่ 3.6 การทำให้แบบจำลองเพทริเน็ตในรูปที่ 3.5 มีคุณสมบัติ Safeness

2. Boundedness

เพลสจะมีคุณสมบัติ Boundedness หรืออาจเรียกว่า k-safe หรือ k-bounded ก็ต่อเมื่อจำนวนโทเค็นในแต่ละเพลสไม่มีโอกาสมากกว่าค่าคงที่จำนวนเต็ม k ซึ่งเป็นค่าที่น้อยที่สุดที่ยังคงทำให้จำนวนโทเค็นในแต่ละเพลสน้อยกว่าค่า k อยู่ตลอดการทำงาน

นิยามที่ 3.8 : เพลส $p_i \in P$ ของเพทริเน็ต $PN = (P, T, I, O, M_0)$ มีคุณสมบัติ k-bounded ถ้า $\forall M' \in R(M), M'(p_i) \leq k$

จะเห็นได้ว่าการที่เพลสนั้น ๆ มีคุณสมบัติ 1-safe หรือ 1-bounded นั่นคือเพทริเน็ตนั้นจะมีคุณสมบัติ Safeness นั่นเอง กล่าวคือเพทริเน็ตที่มีคุณสมบัติ Safeness จะมีคุณสมบัติ Boundedness เสมอ เพลสที่ไม่มีคุณสมบัติ Boundedness ไม่สามารถนำไปสร้างในอุปกรณ์ฮาร์ดแวร์จริง

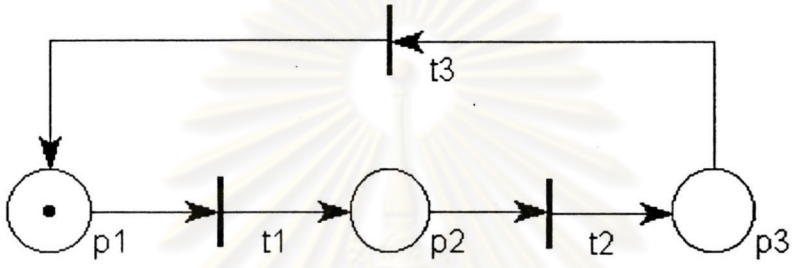
3. Conservation

วิธีการจะตรวจสอบว่าเพทริเน็ตนั้นมีคุณสมบัติ Conservation หรือไม่ก็คือ การตรวจดูจำนวนโทเค็นทั้งหมดของเพทริเน็ตนั้นในแต่ละขั้นของการเปลี่ยนแปลงตลอดการทำงานของเพทริเน็ตนั้น

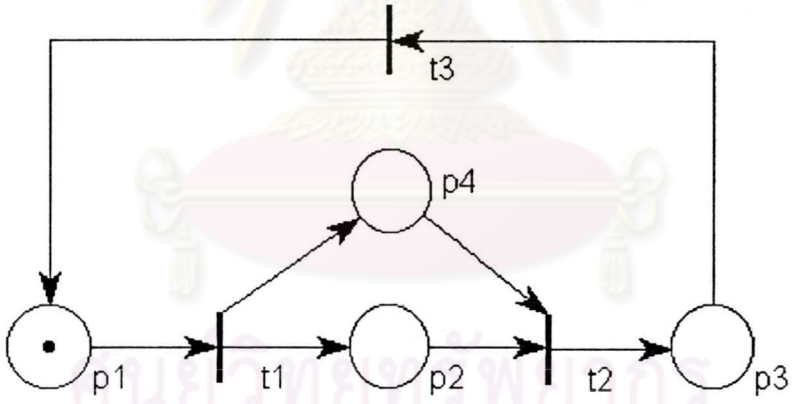
นิยามที่ 3.9 : เพทริเน็ตมีคุณสมบัติ Conservation ถ้า $\forall M' \in R(M_0)$,

$$\sum M'(p_i) = \sum M_0(p_i) \qquad (3\text{-}7)$$

โดยทั่วไปถ้าทุก ๆ ทรานซิชันของเพทริเน็ตมีจำนวนอินพุตเพลสเท่ากับจำนวนเอาต์พุตเพลสแล้ว เพทริเน็ตนั้นจะมีคุณสมบัติ Conservation



รูปที่ 3.7 แบบจำลองเพทริเน็ตที่มีคุณสมบัติ Conservation



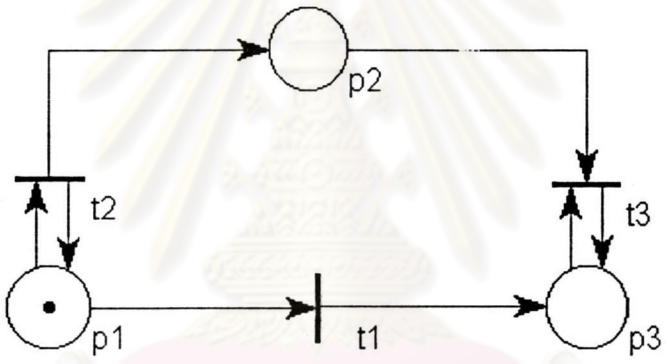
รูปที่ 3.8 แบบจำลองเพทริเน็ตที่ไม่มีคุณสมบัติ Conservation

จากรูปที่ 3.8 แบบจำลองเพทริเน็ตไม่มีคุณสมบัติ Conservation เนื่องจากมาร์กกิงเริ่มต้นมีค่าเท่ากับ (1, 0, 0, 0) ซึ่งผลรวมของจำนวนโทเค็นในทุก ๆ เพลสมีค่าเท่ากับ 1 แต่เมื่อมีการยิงทรานซิชัน t_1 ได้มาร์กกิงใหม่มีค่าเท่ากับ (0, 1, 0, 1) ซึ่งผลรวมของจำนวนโทเค็นในทุก ๆ เพลสมีค่าเท่ากับ 2 ทำให้สมการ (3-7) ไม่เป็นจริง

4. Liveness

การมีคุณสมบัติ Liveness ของเพทริเน็ต คือ การที่ไม่มีทรานซิชันหรือชุดของทรานซิชันใดเลยในเพทริเน็ตนั้นที่ไม่สามารถยิงได้ตลอดช่วงการทำงานของเพทริเน็ตนั้น การที่เพทริเน็ตไม่มีคุณสมบัติ Liveness ทำให้ระบบสามารถเกิดสถานะ Deadlock ซึ่งสถานะ Deadlock เป็นสถานะที่ระบบไม่สามารถทำงานต่อไปได้

นิยามที่ 3.10 : ทรานซิชัน t ในเพทริเน็ต $PN = (P, T, I, O, M_0)$ มีคุณสมบัติ Liveness ถ้า $\forall M' \in R(M_0)$, มีมาร์กิงรีชเอเบิลจาก M' ซึ่งทรานซิชัน t ถูกอีนาเบิล เพทริเน็ตมีคุณสมบัติ Liveness ถ้าแต่ละทรานซิชันมีคุณสมบัติ Liveness



รูปที่ 3.9 แบบจำลองเพทริเน็ตที่ไม่มีคุณสมบัติ Liveness

จากรูปที่ 3.9 แบบจำลองเพทริเน็ตไม่มีคุณสมบัติ Liveness เนื่องจากมีมาร์กกิง (0, 0, 1) ซึ่งเป็นรีชเอเบิลมาร์กกิงของมาร์กิงเริ่มต้น (1, 0, 0) โดยการยิงของทรานซิชัน t_1 และสำหรับมาร์กกิง (0, 0, 1) ทำให้ไม่มีทรานซิชันใดเลยอีนาเบิล คือ ทรานซิชัน t_1 ไม่อีนาเบิลเนื่องจากไม่มีโทเค็นในเพลส p_1 ทรานซิชัน t_2 ไม่อีนาเบิลเนื่องจากไม่มีโทเค็นในเพลส p_1 และทรานซิชัน t_3 ไม่อีนาเบิลเนื่องจากไม่มีโทเค็นในเพลส p_2

<u>การจำลองระบบด้วยเพทริเน็ต</u> [6,10]

แบบจำลองเพทริเน็ตสามารถจำลองการเกิดของเหตุการณ์หลายชนิดและกิจกรรมต่าง ๆ ที่เกิดขึ้นในระบบได้ การจำลองระบบด้วยเพทริเน็ตใช้หลักการเหตุการณ์และเงื่อนไข (Events and Conditions) โดยเหตุการณ์คือสิ่งที่เกิดขึ้นในระบบ การเกิดของเหตุการณ์จะถูกควบคุมโดย

สถานะของระบบ ซึ่งสถานะของระบบนี้สามารถที่จะบรรยายได้ด้วยเงื่อนไข เงื่อนไขคือลักษณะของสถานะของระบบที่สามารถบรรยายด้วยค่าทางตรรกะ เช่นเงื่อนไขเป็นจริงหรือเท็จ การที่เหตุการณ์จะเกิดขึ้นได้นั้นเงื่อนไขบางข้อจำเป็นต้องจะเป็นจริง เงื่อนไขดังกล่าวนี้เป็นเงื่อนไขก่อนเกิดเหตุการณ์ เมื่อเหตุการณ์ดังกล่าวเกิดขึ้นแล้วก็จะทำให้เงื่อนไขใหม่เกิดขึ้น ซึ่งเรียกว่าเป็นเงื่อนไขหลังเกิดเหตุการณ์ ดังนั้นจุดสำคัญในการหาแบบจำลองเพทริเน็ตโดยทั่วไปคือ

แทนเพลสด้วยเงื่อนไข และ แทนทรานซิชันด้วยเหตุการณ์

ตัวอย่างที่ 3.4 เครื่องจักรเครื่องหนึ่งจะทำงานเมื่อมีใบสั่งสินค้าเข้ามาในโรงงานเมื่อเครื่องจักรได้ทำงานตามใบสั่งแล้ว ก็จะส่งชิ้นงานนั้นต่อไป

เงื่อนไข

ก) เครื่องจักรกำลังคอยงาน

ข) ใบสั่งมาถึงโรงงาน

ค) เครื่องจักรกำลังทำงานตามใบสั่ง

ง) งานเสร็จตามใบสั่ง

เหตุการณ์

1) ใบสั่งมาถึงโรงงาน

2) เครื่องจักรเริ่มทำงานตามใบสั่ง

3) เครื่องจักรทำงานเสร็จตามใบสั่ง

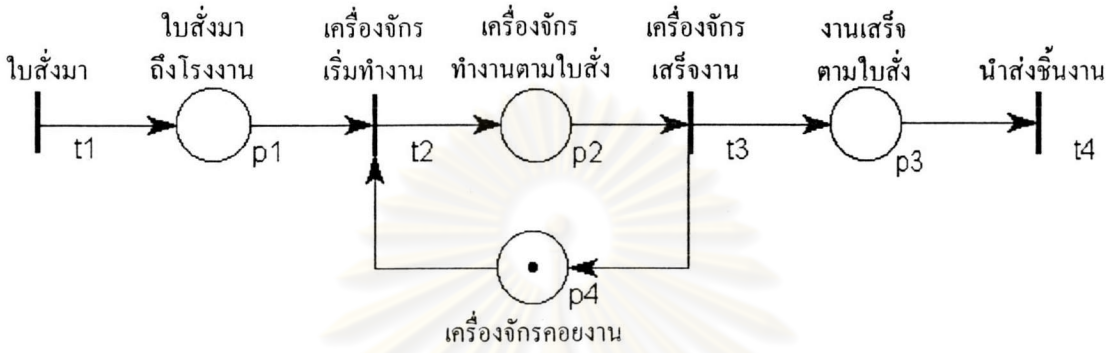
4) ชิ้นงานถูกนำส่ง

เหตุการณ์	เงื่อนไขก่อนเกิด	เงื่อนไขหลังเกิด
1	-	ข
2	ก , ข	ค
3	ค	ก , ง
4	ง	-

ตารางที่ 3.1 เหตุการณ์และเงื่อนไขของโรงงาน

การมองระบบเป็น 2 ส่วน คือ เงื่อนไขและเหตุการณ์ตามตัวอย่างข้างต้นทำให้สามารถจำลองระบบให้อยู่ในรูปแบบจำลองเพทริเน็ตได้โดยง่าย โดยการแทนเงื่อนไขด้วยเพลส และ เหตุการณ์ด้วยทรานซิชัน อินพุตเพลสของทรานซิชันเป็นเงื่อนไขก่อนเกิด ส่วนเอาต์พุตเพลสของ

ทรานซิชันเป็นเงื่อนไขหลังเกิด การแสดงว่าเงื่อนไขจริงหรือเท็จแทนได้ด้วยโทเค็นที่อยู่ในเพลส เมื่อทรานซิชันถูกยิงโทเค็นที่แทนเงื่อนไขก่อนเกิดว่าจริงจะหายไปและเกิดโทเค็นขึ้นในเงื่อนไขหลังเกิดแทน ซึ่งแสดงแบบจำลองเพทริเน็ตได้ดังรูปที่ 3.10



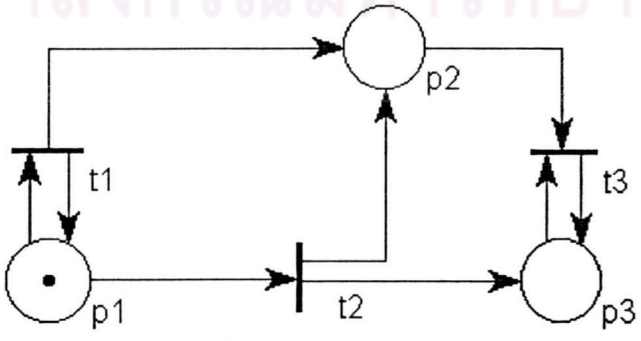
รูปที่ 3.10 แบบจำลองเพทริเน็ตของโรงงาน

<u>วิธีการวิเคราะห์แบบจำลองเพทริเน็ต</u> [6,8]

ในการวิเคราะห์แบบจำลองเพทริเน็ตสามารถหาคุณสมบัติของแบบจำลองเพทริเน็ตได้ เช่น Safeness, Boundedness, Liveness และ Conservation เป็นต้น โดยมีวิธีในการวิเคราะห์ 2 วิธีหลัก ๆ คือ

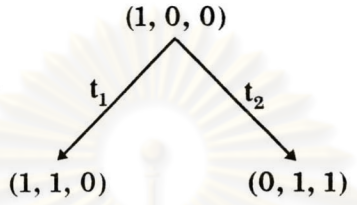
1. <u>วิธีการวิเคราะห์โดยใช้รีชเอบิลิตีทรี (Reachability Tree)</u>

รีชเอบิลิตีทรีเป็นตัวแทนสำหรับเซ็ตของรีชเอเบิลมาร์กกิงในเพทริเน็ต ซึ่งทรีหรือส่วนหนึ่งของทรีนี้จะแสดงมาร์กกิงทั้งหมดของการเปลี่ยนแปลงของเพทริเน็ตนั้นจากมาร์กกิงเริ่มต้น



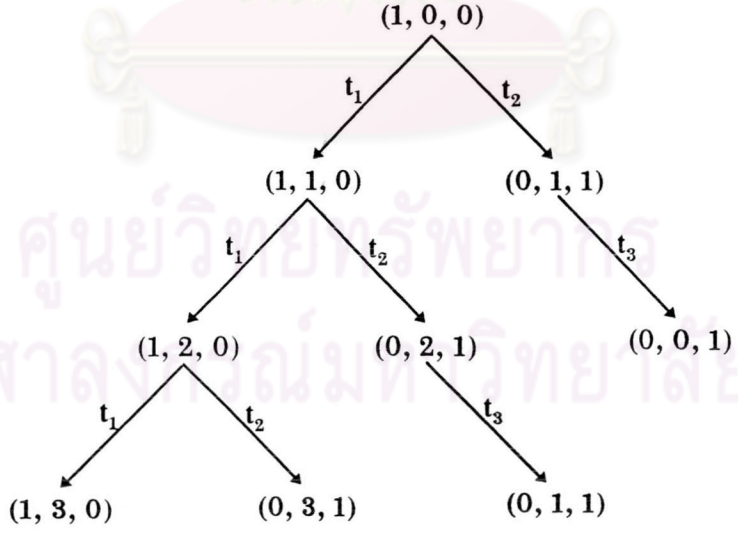
รูปที่ 3.11 แบบจำลองเพทริเน็ตที่ใช้ในการศึกษาวิธีการวิเคราะห์โดยใช้รีชเอบิลิตีทรี

ในตอนแรกกำหนดมาร์กกิงเริ่มต้นเป็น (1, 0, 0) ซึ่งจะพบว่าทั้งทรานซิชัน t_1 และ t_2 สามารถยิงได้ โดยมาร์กกิงใหม่ที่เกิดจากการยิงทรานซิชัน t_1 มีค่าเป็น (1, 1, 0) และมาร์กกิงใหม่ที่เกิดจากการยิงทรานซิชัน t_2 มีค่าเป็น (0, 1, 1)



รูปที่ 3.12 การสร้างรีชเอบิลิตีทรีจากรูปที่ 3.11 ในการยิงขั้นแรก

ด้วยวิธีการเช่นนี้จะทำให้สามารถเขียนรีชเอบิลิตีทรีได้ แต่ถ้าเขียนต่อไปเรื่อย ๆ สามารถทำให้เกิดรีชเอบิลิตีทรีที่มีขนาดไม่จำกัด ดังจะเห็นได้จากรูปที่ 3.11 เป็นแบบจำลองเพทริเน็ตที่มีจำนวนเพลสและทรานซิชันจำกัดแต่สามารถทำให้เกิดรีชเอบิลิตีทรีมีขนาดไม่จำกัดดังรูปที่ 3.13 ซึ่งจะทำการวิเคราะห์ไม่ได้



รูปที่ 3.13 การสร้างรีชเอบิลิตีทรีจากรูปที่ 3.11 ในการยิงขั้นที่ 3

เพื่อกำหนดให้รีชเอบิลิตีทรีมีขอบเขตเสมอ จึงมาพิจารณาชนิดของมาร์กกิงที่เกิดใหม่ พบว่าจะมีโนดทั้งหมด 3 ประเภทที่จะช่วยในการทำให้รีชเอบิลิตีทรีมีขอบเขต คือ

1. เทอร์มินัลโนด (Terminal Node) เป็นโนดที่ไม่มีทรานซิชันใดที่อีนาเบิล และสามารถยิงได้

2. โนดซ้ำซ้อน (Duplicate Node) เป็นโนดที่มีจำนวนโทเค็นในทุก ๆ เพลสซ้ำกับโนดที่อยู่ในทรีก่อนหน้า

3. โนดที่มีจำนวนโทเค็นในบางเพลสมีค่าเพิ่มขึ้นเป็นอนันต์ โนดชนิดนี้เราจะให้ ω แทนการมีค่าเพิ่มขึ้นเป็นอนันต์ของจำนวนโทเค็นในเพลสนั้น โดยมีพีชคณิตของ ω เป็น

$$\omega \pm a = \omega \tag{3-8}$$

$$\omega > a \tag{3-9}$$

$$\omega \geq \omega \tag{3-10}$$

เมื่อ a เป็นค่าคงที่

การสร้างรีชเอบิลิตีทรีสำหรับเพทริเน็ตมีขั้นตอนวิธีดังต่อไปนี้

1. ให้โนดเริ่มต้นมีค่าเท่ากับมาร์กกิงเริ่มต้นและให้ค่าสถานะของโนดเป็นโนดใหม่

2. ขณะที่ยังมีโนดใหม่ทำดังต่อไปนี้

2.1 เลือกโนดใหม่ (มาร์กกิง M)

2.2 ถ้าโนดใหม่มีจำนวนโทเค็นในทุก ๆ เพลส ซ้ำกับโนดที่อยู่ในทรีก่อนหน้า ให้สถานะของโนดใหม่นั้นมีค่าเป็นโนดซ้ำซ้อน

2.3 ถ้าไม่มีทรานซิชันใดเลยสามารถยิงได้จากโนดใหม่นี้ ให้สถานะของโนดใหม่มีค่าเป็นเทอร์มินัลโนด

2.4 สำหรับโนดใหม่ที่ไม่เป็นโนดซ้ำซ้อนหรือเทอร์มินัลโนดทำดังต่อไปนี้

2.4.1 สร้างโนด (M′) ที่เป็นผลมาจากการยิงทรานซิชัน t_j ของมาร์กกิง M

2.4.2 ในทรีจากรากถึงมาร์กกิง M ถ้ามีมาร์กกิง M″ใด ๆ ซึ่งทำให้ $M'(p) \geq M''(p)$ สำหรับแต่ละเพลส p และ $M'(p) \neq M''(p)$ แล้วจึงแทน M′(p) ด้วย ω สำหรับแต่ละเพลส p ที่ $M'(p) > M''(p)$

2.4.3 วาดอาร์กชื่อ t_j จากโนด M ไปยังโนด M′ และให้สถานะของโนดนั้นมีค่าเป็นโนดใหม่

ผลที่ได้จากการวิเคราะห์โดยใช้รีชเอบิลิตีทรี

1. การตรวจสอบคุณสมบัติ Safeness นี้ สามารถดูได้จากจำนวนโทเค็น ในทุก ๆ เพลส ในทุก ๆ โนดของรีชเอบิลิตีทรี จะต้องมีค่าไม่มากกว่าหนึ่ง

2. การตรวจสอบคุณสมบัติ Boundedness นี้สามารถดูได้จากจำนวนโทเค็นในทุกๆ เพลสของทุก ๆ โนดของรีชเอบิลิตีทรี จะต้องมีค่าไม่มากกว่าจำนวนเต็มค่าหนึ่ง นั่นคือไม่สามารถเป็น ω ได้

3. การตรวจสอบคุณสมบัติ Conservation นี้ สามารถดูได้จากจำนวนโทเค็น ในทุก ๆ เพลสของรีชเอบิลิตีทรีจะต้องบวกกันได้ค่าคงที่ค่าหนึ่งสำหรับทุก ๆ โนด

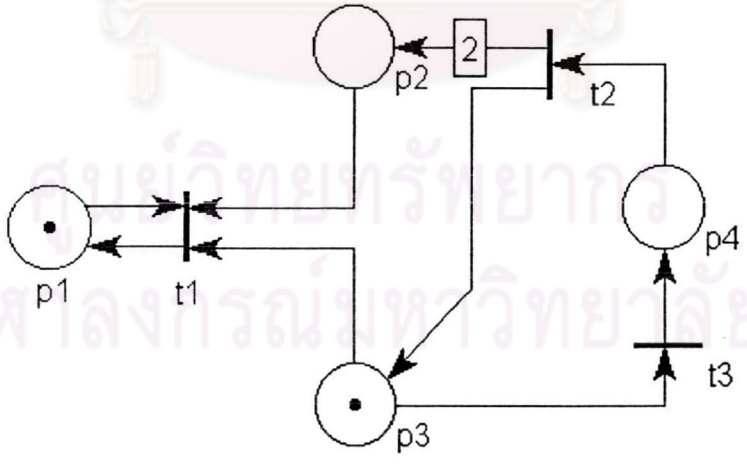
4. การตรวจสอบคุณสมบัติ Liveness นี้ ทุก ๆ โนดของรีชเอบิลิตีทรีจะต้องไม่เป็นเทอร์มินัลโนด

2. <u>วิธีการวิเคราะห์โดยใช้สมการเมตริกซ์</u>

ในการวิเคราะห์โดยใช้สมการเมตริซ์ เราจะแทนแบบจำลองเพทริเน็ตให้อยู่ในรูปของอินซิเดนท์เมตริกซ์ C มีขนาด m แถว (จำนวนทรานซิชัน) และ n คอลัมน์ (จำนวนเพลส) โดยกำหนดให้มี m-vector X ซึ่งมีค่าเป็นจำนวนครั้งที่ทรานซิชันในตำแหน่งนั้นถูกยิงผ่านไป จะได้ว่า

$$M'(p) = M(p) + (X \bullet C) \qquad (3\text{-}11)$$

ซึ่งจากสมการที่ (3-11) สามารถหาค่าสถานะถัดไปได้ ถ้ารู้ค่าสถานะปัจจุบันและทรานซิชันที่ต้องการยิง



รูปที่ 3.14 ตัวอย่างแบบจำลองเพทริเน็ตในการวิเคราะห์โดยใช้สมการเมตริกซ์

จากรูปที่ 3.14 สามารถหาอินซิเดนท์เมตริกซ์ได้ดังนี้

$$\mathbf{C}^{-} = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{3-12}$$

$$\mathbf{C}^{+} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3-13}$$

จากสมการที่ (3-3) ได้

$$\mathbf{C} = \begin{bmatrix} 0 & -1 & -1 & 0 \\ 0 & 2 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix} \tag{3-14}$$

มาร์กกิงเริ่มต้นคือ $M_0(p) = (1, 0, 1, 0)$ และ ทรานซิชัน t_3 สามารถยิงได้จะก่อให้เกิดสถานะถัดไป $M'(p)$ จากสมการที่ (3-11) ซึ่งมีค่าเป็น

$$\begin{aligned} \mathbf{M'(p)} &= (1,\ 0,\ 1,\ 0) + (0,\ 0,\ 1) \bullet \begin{bmatrix} 0 & -1 & -1 & 0 \\ 0 & 2 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix} \\ &= (1,\ 0,\ 1,\ 0) + (0,\ 0,\ -1,\ ,1) \\ &= (1,\ 0,\ 0,\ 1) \end{aligned} \tag{3-15}$$

ด้วยมาร์กกิงเดิม ถ้ามีลำดับการยิง $\sigma = t_3t_2t_3t_2t_1$ ซึ่งสามารถแทนได้โดยเวคเตอร์การยิง $f(\sigma) = (1, 2, 2)$ ได้จะก่อให้เกิดสถานะถัดไป $M'(p)$ ซึ่งมีค่าเป็น

$$\begin{aligned} \mathbf{M'(p)} &= (1,\ 0,\ 1,\ 0) + (1,\ 2,\ 2) \bullet \begin{bmatrix} 0 & -1 & -1 & 0 \\ 0 & 2 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix} \\ &= (1,\ 0,\ 1,\ 0) + (0,\ 3,\ -1\ \ 0) \\ &= (1,\ 3,\ 0,\ 0) \end{aligned} \tag{3-15}$$

ด้วยมาร์กกิงเริ่มต้นเดิมต้องการทราบว่ามาร์กกิง (1, 8, 0, 1) จะเกิดได้ต้องมีลำดับการยิงทรานซิชันเป็นอย่างไร

$$(1,\ 8,\ 0,\ 1) = (1,\ 0,\ 1,\ 0) + \mathbf{X} \bullet \begin{bmatrix} 0 & -1 & -1 & 0 \\ 0 & 2 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix} \qquad (3\text{-}16)$$

$$(0,\ 8,\ -1,\ 1) = \mathbf{X} \bullet \begin{bmatrix} 0 & -1 & -1 & 0 \\ 0 & 2 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

$$\mathbf{X} = (0,\ 4,\ 5)$$

ดังนั้นจะได้ลำดับการยิงเป็น $\sigma = t_3t_2t_3t_2t_3t_2t_3t_2t_3$

ผลที่ได้จากการวิเคราะห์โดยใช้สมการเมตริกซ์

1. Conservation การตรวจสอบคุณสมบัติ Conservation นี้ สามารถดูได้จากสมการ C.w = 0 เมื่อ w = (1, 1, 1, ...,1) โดยมีขนาด (m × 1)

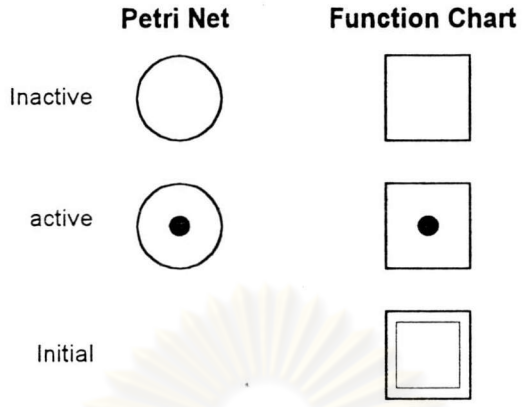
2. เมื่อแก้สมการที่ (3-11) สามารถหาค่าสถานะถัดไปได้ ถ้ารู้ค่ามาร์กกิงปัจจุบัน และ ทรานซิชันที่ต้องการยิงเนื่องจาก C ขึ้นกับแบบจำลองของเพทริเน็ตดังตัวอย่าง

<u>ความสัมพันธ์ระหว่างเพทริเน็ตกับฟังก์ชันชาร์ต</u> [11,12]

ฟังก์ชันชาร์ตเป็นภาษารูปภาพ ซึ่งมีโนด 2 ชนิด คือ สเต็ปและทรานซิชัน โดยมีเส้นเชื่อมต่อ (Directed Link) เป็นตัวเชื่อมระหว่างสเต็ปและทรานซิชัน เนื่องจากการพัฒนาฟังก์ชัน-ชาร์ตมีรากฐานมาจากทฤษฎีเพทริเน็ต ดังนั้นการแทนรูปทางกราฟฟิกของฟังก์ชันชาร์ตแบบดั้งเดิมเหมือนกับการแทนรูปทางกราฟฟิกของเพทริเน็ต สเต็ปสามารถถูกแทนด้วยเพลสในเพทริเน็ต และทรานซิชันสามารถถูกแทนด้วยทรานซิชันในเพทริเน็ต ซึ่งเราสามารถแสดงรายละเอียดความสัมพันธ์ได้ดังนี้

1. <u>สเต็ปและเพลส</u>

สเต็ปและเพลสถูกแสดงดังรูปที่ 3.15 สเต็ปมีเพียง 2 สถานะ คือ แอกทีฟ(Active) และ ไม่แอกทีฟ ซึ่งการแอกทีฟของสเต็ปสามารถถูกแทนโดยโทเค็นในเพทริเน็ต แต่ต้องมีจำนวนโทเค็นในเพลสไม่เกิน 1 โทเค็น สำหรับสเต็ปเริ่มต้นสามารถถูกแทนด้วยมาร์กกิงเริ่มต้นของเพทริเน็ตโดยกำหนดให้ มีโทเค็น 1 โทเค็นในเพลสเริ่มต้น



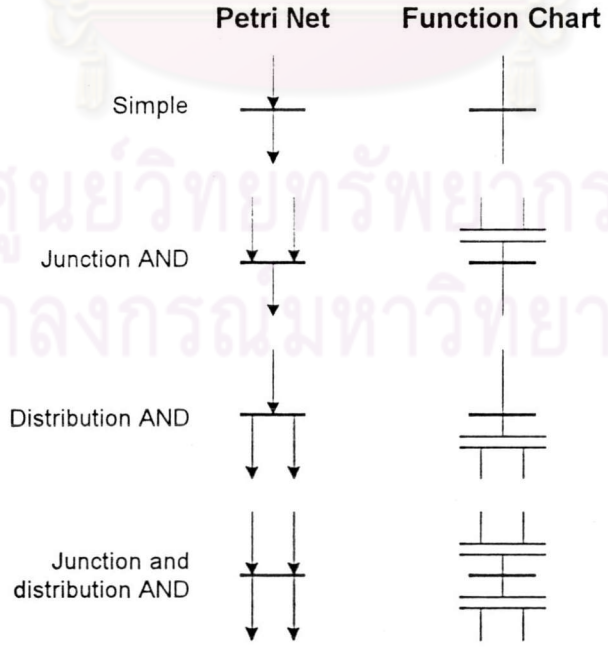
รูปที่ 3.15 การแทนรูปทางกราฟฟิกของสเต็ปและเพลส

2. ทรานซิชัน

ทรานซิชันในฟังก์ชันชาร์ตสามารถถูกแทนด้วยทรานซิชันในเพทริเน็ตได้โดยตรง เนื่องจากรูปภาพทางกราฟฟิกเหมือนกัน

3. เส้นเชื่อมต่อและอาร์ก

เส้นเชื่อมต่อและอาร์กถูกแสดงดังรูปที่ 3.16 เส้นเชื่อมต่อโดยทั่วไปถ้าไม่มีลูกศรแสดงทิศทางแสดงว่าชี้จากข้างบนลงข้างล่าง



รูปที่ 3.16 การแทนรูปทางกราฟฟิกของเส้นเชื่อมต่อ

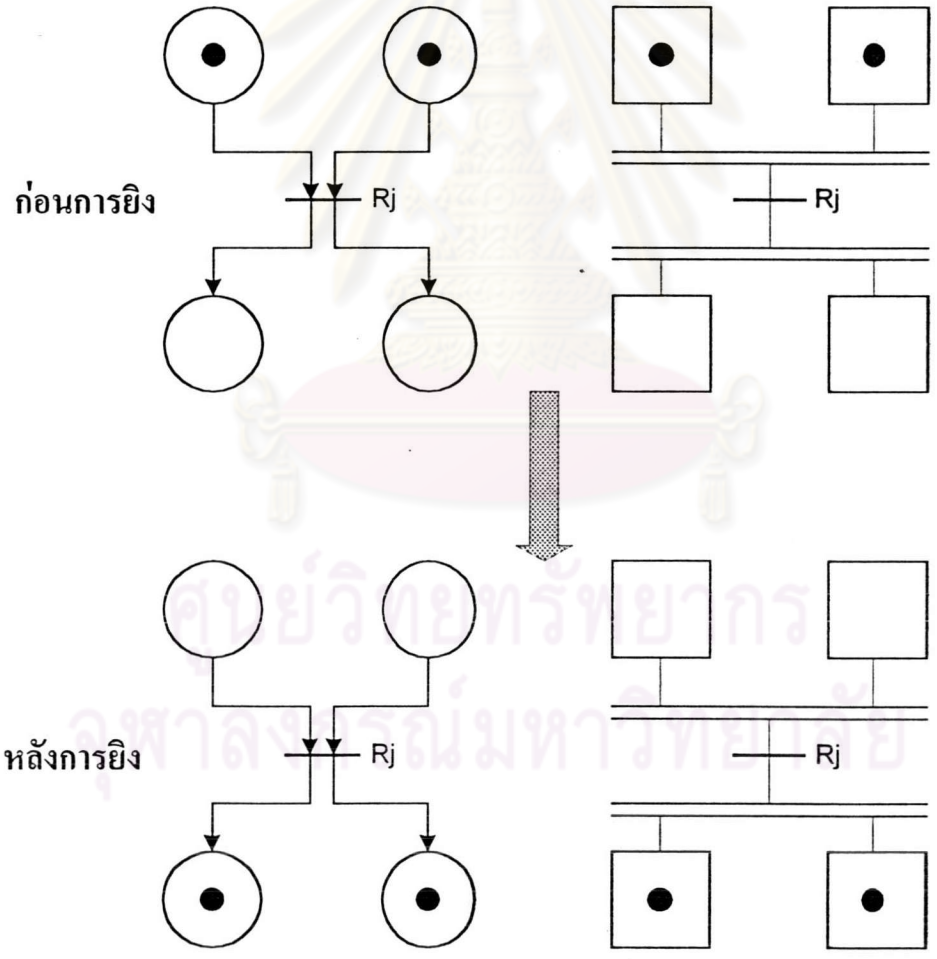
4. <u>การยิงของทรานซิชัน</u>

เงื่อนไขการยิงทรานซิชันในฟังก์ชันชาร์ต คือ

4.1 สเต็ปที่ชี้ไปยังทรานซิชันนั้นทั้งหมดต้องแอกทีฟ

4.2 เงื่อนไขของทรานซิชันเป็นจริง

จะเห็นได้ว่าเงื่อนไขในข้อ 4.1 เหมือนกับการอีนาเบิลของทรานซิชันในเพทริเน็ต ดังนั้นถ้าต้องการให้การยิงทรานซิชันของเพทริเน็ตเหมือนกับฟังก์ชันชาร์ต ต้องเพิ่มเงื่อนไขการยิงของทรานซิชันนอกจากทรานซิชันอีนาเบิลแล้ว ยังต้องมีเงื่อนไขของทรานซิชันเป็นจริงด้วย ตัวอย่างแสดงการยิงของทรานซิชันแสดงดังรูปที่ 3.17



รูปที่ 3.17 ตัวอย่างการยิงทรานซิชัน