บทที่ 4
ทฤษฎีเพทริเน็ต

ความเป็นมา [8,9]

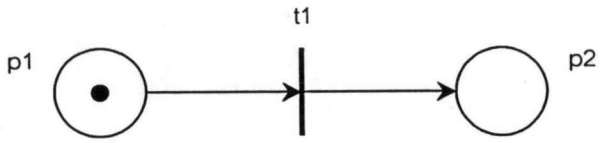
ทฤษฎีเพทริเน็ตได้ถูกริเริ่มขึ้นเมื่อปี ค.ศ. 1962 โดย Carl Adam Petri ในระหว่างการทำวิทยานิพนธ์ระดับปริญญาเอกที่มหาวิทยาลัย Darmstadt ประเทศเยอรมัน โดยมีจุดประสงค์เพื่อใช้วิเคราะห์แบบจำลองระบบสื่อสาร ซึ่งเป็นการวิเคราะห์ความสัมพันธ์ของสภาวะและเหตุการณ์ต่างๆ ที่เกิดขึ้นในระบบ ต่อมาในปี ค.ศ. 1965 เพทริเน็ตเริ่มเป็นที่รู้จักมากขึ้น เมื่อ A.W.Holt ได้นำทฤษฎีเพทริเน็ตมาใช้ในโครงการวิจัยทฤษฎีระบบสารสนเทศของ Applied Data Research Inc ประเทศสหรัฐอเมริกา หลังจากนั้นในช่วงปี ค.ศ. 1970 ถึง 1975 เป็นช่วงที่ทฤษฎีเพทริเน็ตได้รับความสนใจอย่างมาก เมื่อกลุ่มโครงสร้างการคำนวณ (Computational Structure - Group) แห่ง MIT ได้ทำการวิจัยเกี่ยวกับทฤษฎีเพทริเน็ต จึงได้เสนอบทความเกี่ยวกับเพทริเน็ตอย่างต่อเนื่องทำให้ทฤษฎีเพทริเน็ตได้พัฒนาไปอย่างรวดเร็ว

ในปัจจุบันทฤษฎีเพทริเน็ตได้ถูกนำมาใช้อย่างแพร่หลาย โดยจุดประสงค์หลักของการนำเพทริเน็ตมาใช้คือการจำลองรูปแบบของระบบเพื่อนำแบบจำลองมาวิเคราะห์ ตัวอย่างของการจำลองระบบโดยใช้เพทริเน็ต [2,3,11,12] คือ ระบบฮาร์ดแวร์ของคอมพิวเตอร์ ระบบการจัด - แบ่งทรัพยากรในคอมพิวเตอร์ ระบบกระบวนการผลิตในอุตสาหกรรม ตลอดจนการจำลองการไหลของข้อมูลในองค์กร เป็นต้น

พื้นฐานของเพทริเน็ต [8,9]

เพทริเน็ตเป็นแนวความคิดอย่างหนึ่งที่สามารถจำลองสถานะการทำงานต่างๆ ให้อยู่ในรูปแบบที่สามารถนำมาวิเคราะห์ได้ การศึกษาพฤติกรรมของระบบนั้นทำได้โดยการพิจารณาความสัมพันธ์ระหว่างสภาวะการทำงานและเหตุการณ์ที่อาจเกิดขึ้นได้ในระบบ

เพทริเน็ตสามารถอธิบายสองวิธีคือวิธีทางรูปภาพและวิธีทางคณิตศาสตร์ วิธีทางรูปภาพนั้น เพทริเน็ตสามารถเขียนจากวัตถุ 3 ชนิดคือ เพลส (Place) ทรานสิชัน (Transition) และ อาร์ก (Arc) โดยเพลสแสดงอยู่ในรูป ของวงกลม ทรานสิชันแสดงอยู่ในรูปของแท่งสี่เหลี่ยมและอาร์กแสดงอยู่ในรูปของเส้นตรง ตัวอย่างการแสดงเพทริเน็ตด้วยรูปภาพแสดงดังรูป



รูปที่ 4.1 องค์ประกอบของเพทริเน็ต

ในการแสดงความสัมพันธ์ระหว่างทรานสิชันและเพลสนั้น เริ่มจากการแยกประเภทของเพลสที่เป็นอินพุตเพลสและเอาต์พุตเพลสเมื่ออ้างอิงกับทรานสิชันใดๆ โดยเพลสที่มีอาร์กเชื่อมต่อจากเพลสไปทรานสิชัน จะเรียกเพลสนั้นว่าเป็นอินพุตเพลสของทรานสิชัน และเพลสที่มีอาร์กเชื่อมต่อจากทรานสิชันไปสู่เพลส จะเรียกเพลสนั้นว่าเป็นเอาต์พุตเพลสของทรานสิชัน

จากรูปที่ 4.1 เพทริเน็ตประกอบด้วยเพลสจำนวน 2 เพลส ทรานสิชันจำนวน 1 ทรานสิชัน และอาร์กจำนวน 2 อาร์ก จากรูปเห็นว่าเพลส p_1 เป็นอินพุตเพลสของทรานสิชัน t_1 และเพลส p_2 เป็นเอาต์พุตเพลสทรานสิชัน t_1

การแสดงสภาวะของระบบในขณะใดๆ แสดงได้โดยใช้โทเค็น (Token) โดยเพลสแต่ละเพลสอาจมีโทเค็นหรือไม่มีก็ได้ โทเค็นแสดงด้วยจุดที่อยู่ภายในเพลส เราเรียกการกระจายโทเค็นในเพทริเน็ตว่ามาร์กกิง จากรูปที่ 4.1 เห็นว่าโทเค็นอยู่ภายใน p_1 แสดงถึงสถานะของระบบในขณะนั้นคือสภาวะที่กำหนดโดยเพลส p_1 จากการแสดงสภาวะของระบบในขณะใดๆ ด้วยโทเค็นและการเปลี่ยนแปลงการกระจายโทเค็นเป็นผลจากเหตุการณ์ต่างๆ ที่เกิดขึ้นทำให้เราสามารถศึกษาพฤติกรรมการเปลี่ยนแปลงของระบบได้

เมื่อเปรียบเทียบแบบจำลองเพทริเน็ตกับการทำงานในระบบจริงนั้น อินพุตเพลสอาจถูกใช้แสดงสภาวะของระบบก่อนเกิดเหตุการณ์ใดๆ และเอาต์พุตเพลสอาจถูกใช้แสดงสภาวะของระบบหลังเกิดเหตุการณ์ต่างๆ นอกจากนี้แล้วยังอาจใช้อินพุตเพลสแสดงถึงความพร้อมของทรัพยากรในระบบและเอาต์พุตเพลสแสดงการคืนการใช้ทรัพยากรของระบบด้วย

โครงสร้างของเพทริเน็ต [8,9,11,12]

เพทริเน็ตประกอบด้วยโครงสร้างส่วนต่าง ๆ คือ $PN = (P,T,I,O,M_0)$ โดยกำหนดให้ $N = \{0,1,2,....\}$

- $P = \{p_1, p_2,..., p_n\}$ เป็นเซตของเพลส
- $T = \{t_1, t_2,..., t_n\}$ เป็นเซตของทรานสิชัน
- $P \cap T = \phi$
- $I : (P \times T) \rightarrow N$ เป็นอินพุตฟังก์ชัน แสดงโดยอาร์กจากเพลสสู่ทรานสิชัน
- $O : (P \times T) \rightarrow N$ เป็นเอาต์พุตฟังก์ชัน แสดงโดยอาร์กจากทรานสิชันสู่เพลส
- $M_0 : P \rightarrow N$ เป็นมาร์กกิงเริ่มต้น (Initial Marking)

จากรูปที่ 4.1 เห็นได้ว่า $P = \{p_1, p_2\}$, $T = \{t_1\}$

$I(p_1,t_1) = 1$, $I(p_2,t_1) = 0$

$O(p_1,t_1) = 0$, $O(p_2,t_1) = 1$

$M_0(p_1) = 1$, $M_0(p_2) = 0$

กฎการทำงานของเพทริเน็ต [8,9,11,12]

การทำงานของเพทริเน็ตพิจารณาได้จากเปลี่ยนแปลงของโทเค็นภายในเพลส โดยการเปลี่ยนแปลงของโทเค็นจะสอดคล้องกับกฎการทำงานของเพทริเน็ต ซึ่งประกอบด้วยกฎการอีนาเบิล (Enable Rule) และกฎการยิงทรานสิชัน (Firing Rule)

กฎการอีนาเบิล

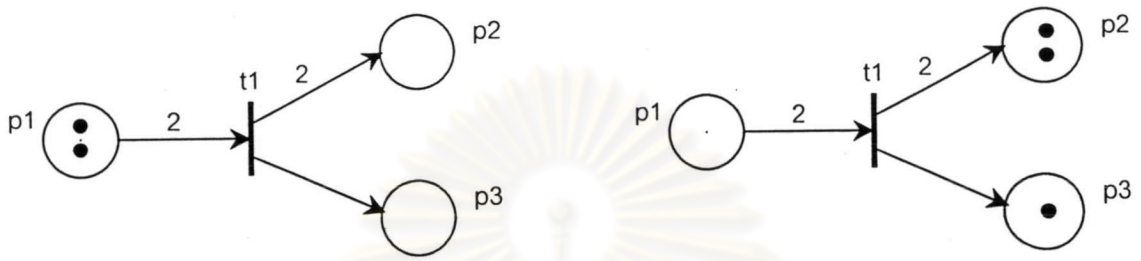
1. ทรานสิชัน t จะอีนาเบิล ถ้าทุกๆ อินพุตเพลส p ของทรานสิชัน t มีโทเค็นภายในมากกว่าหรือเท่ากับน้ำหนักของอาร์กจากเพลส p ไปที่ทรานสิชัน t

กฎการยิงทรานสิชัน

1. ถ้าทรานสิชัน t ถูกอีนาเบิลแล้ว อาจถูกยิงหรือไม่ถูกยิงทรานสิชันก็ได้ โดยการถูกยิงทรานสิ - ชันขึ้นอยู่กับเหตุการณ์ที่เกิดขึ้นในระบบ
2. การยิงทรานสิชัน t ที่ถูกอีนาเบิลอยู่จะทำให้มาร์กกิงในเพทริเน็ตเกิดการเปลี่ยนแปลง โดยโทเค็นที่มีจำนวนเท่ากับน้ำหนักของอาร์กที่ต่อจากอินพุตเพลสจะเคลื่อนจากอินพุตเพลสไป

ยังทรานสิชัน t และเคลื่อนย้ายโทเค็นจำนวนเท่ากับน้ำหนักของอาร์กที่ต่อจากทรานสิชันไปยังเอาต์พุตเพลสจากทรานสิชัน t ไปสู่เอาต์พุตเพลส

ตัวอย่างการยิงทรานสิชันแสดงดังรูปที่ 4.2



(ก) ก่อนยิงทรานสิชัน (ข) หลังยิงทรานสิชัน

รูปที่ 4.2 ตัวอย่างการยิงทรานสิชัน

จากตัวอย่างดังรูปจากรูปที่ 4.2(ก) ทรานสิชัน t_1 ถูกอีนาเบิลเนื่องจากอินพุตเพลส p_1 ของทรานสิชัน t_1 มีโทเค็นอยู่ภายใน 2 โทเค็นซึ่งเท่ากับน้ำหนักอาร์กจากเพลส p_1 ไปที่ทรานสิชัน t_1 และหลังจากถูกยิงทรานสิชัน t_1 แล้วทำให้โทเค็นจำนวน 2 โทเค็นเคลื่อนที่ออกจากเพลส p_1 และเกิดโทเค็นที่เอาต์พุตเพลส p_2 จำนวน 2 โทเค็นและเอาต์พุตเพลส p_3 จำนวน 1 โทเค็น ซึ่งเท่ากับน้ำหนักอาร์กจากที่ทรานสิชัน t_1 ไปสู่เพลส p_2 และ p_3 ตามลำดับ แสดงดังรูปที่ 4.2(ข)

สมการสเตทของเพทริเน็ต [8,9,11,12]

จากที่ได้กล่าวมาแล้ว เพทริเน็ตสามารถแสดงได้ทั้งวิธีทางรูปภาพและวิธีทางคณิตศาสตร์ ในวิธีทางคณิตศาสตร์ เพทริเน็ตสามารถอธิบายพฤติกรรมการเปลี่ยนแปลงของระบบ เมื่อเกิดเหตุการณ์ต่างๆ ได้โดยการคำนวณจากสมการสเตทของแบบจำลองเพทริเน็ต

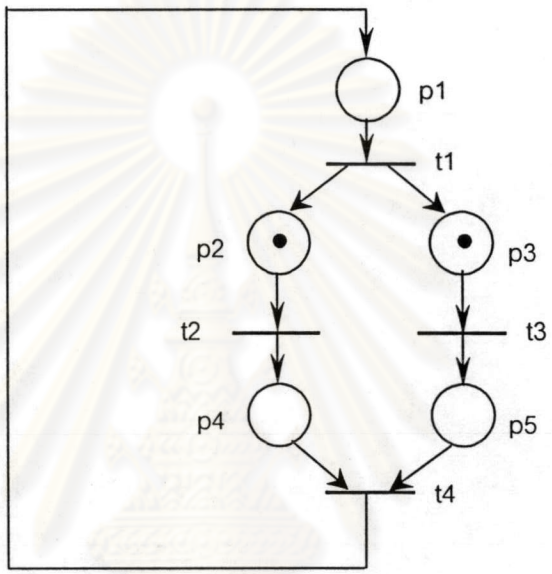
เมตริกซ์อุบัติการณ์ (Incidence Matrix) A_{ij} เป็นเมตริกซ์ที่แสดงการเชื่อมต่อระหว่างเพลสและทรานสิชันต่างๆ ภายในเพทริเน็ต โดยมีมิติเท่ากับ $n \times m$ ซึ่งมีสมาชิกทุกตัวภายในเมตริกซ์เป็นจำนวนเต็ม ซึ่ง n คือจำนวนของทรานสิชันทั้งหมด และ m คือจำนวนของเพลสทั้งหมด สมาชิกของเมตริกซ์อุบัติการณ์ a_{ij} นิยามได้ดังนี้

$$a_{ij} = a_{ij}^{+} - a_{ij}^{-}$$

โดย $a_{ij}^{+} = O(p_j, t_i)$ เป็นน้ำหนักของอาร์คที่ต่อระหว่างทรานสิชัน t_i ไปที่เอาต์พุตเพลส p_j

$a_{ij}^{-} = I(p_j, t_i)$ เป็นน้ำหนักของอาร์คที่ต่อระหว่างอินพุตเพลส p_j ไปที่ทรานสิชัน t_i

พิจารณาเพทริเน็ตในรูปที่ 4.3



รูปที่ 4.3 ตัวอย่างเพทริเน็ตที่ใช้หาเมตริกซ์อุบัติการณ์

จากรูปที่ 4.3 แสดงเมตริกซ์อุบัติการณ์ได้ดังนี้

$$A_{ij}^{-} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

$$A_{ij}^{+} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$A_{ij} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & 1 \\ 1 & 0 & 0 & -1 & -1 \end{bmatrix}$$

มาร์กกิงของเพทริเน็ตที่ประกอบด้วยเพลสจำนวน m เพลสสามารถแสดงโดยใช้เวกเตอร์ M_k ที่มีมิติเท่ากับ $m \times 1$ โดยสมาชิกทุกตัวในเวกเตอร์ M_k เป็นจำนวนเต็มที่มากกว่าหรือเท่ากับศูนย์ เนื่องจากโทเค็นในแต่ละเพลสต้องเป็นจำนวนเต็มที่มากกว่าหรือเท่ากับศูนย์ ตัวอย่างเช่น จากรูปที่ 4.3 มาร์กกิงเริ่มต้น $M_0 = [01100]^T$

สมการสเตทสำหรับเพทริเน็ตแสดงถึงมาร์กกิงภายในเพทริเน็ต ซึ่งเป็นผลการเปลี่ยนแปลงโทเค็นหลังจากการเกิดยิงทรานสิชันขึ้นนิยามได้ดังนี้

$$M_k = M_{k-1} + A^T u_k \qquad ;k = 1,2,3,....$$

โดย M_k คือเมตริกซ์ขนาด $n \times 1$ เป็นเมตริกซ์ที่แสดงมาร์กกิงหลังการยิงทรานสิชัน

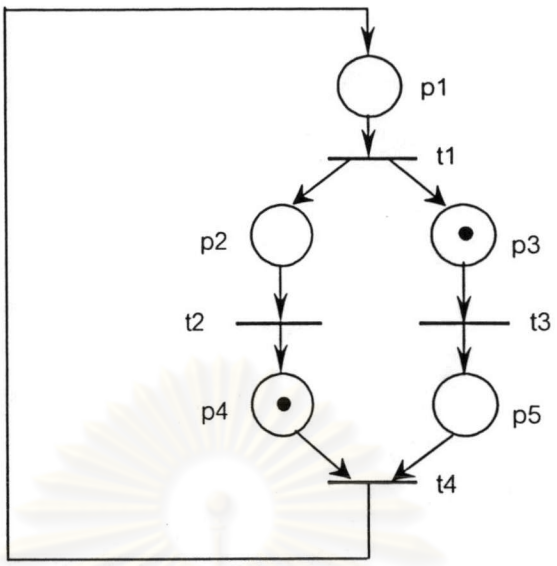
M_{k-1} คือเมตริกซ์ขนาด $n \times 1$ เป็นเมตริกซ์ที่แสดงมาร์กกิงก่อนการยิงทรานสิชัน

u_k คือเมตริกซ์ขนาด $n \times 1$ เป็นเวกเตอร์ยิงทรานสิชัน (Firing Vector)

จากตัวอย่างเพทริเน็ตในรูปที่ 4.3 หากยิงทรานสิชัน t_2 เวกเตอร์การยิงทรานสิชันคือ $u_1 = [0 \quad 1 \quad 0 \quad 0]^T$ ทำให้มาร์กกิงของเพทริเน็ตหลังจากยิงทรานสิชันคือ

$$M_1 = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 & 1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & 1 \\ 1 & 0 & 0 & -1 & -1 \end{bmatrix}^T \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ -1 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

จาก M_1 ที่คำนวณได้แสดงว่าหลังจากยิงทรานสิชัน t_2 แล้วจะมีโทเค็นอยู่ในเพลสที่ 3 และ 4 จำนวนเพลสละหนึ่งโทเค็น เมื่อแสดงเพทริเน็ตโดยใช้วิธีการใช้รูปแล้วแสดงได้ดังรูป

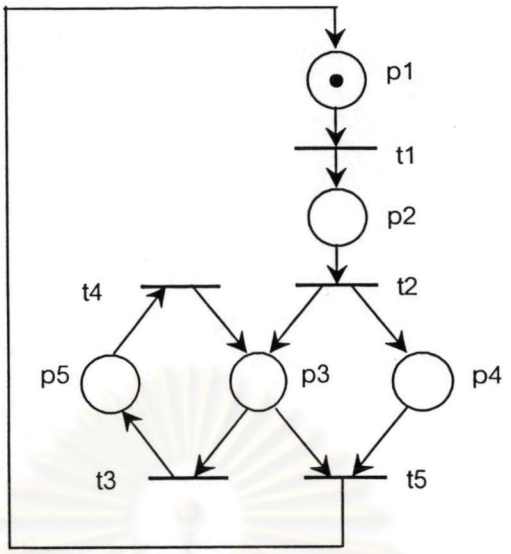


รูปที่ 4.4 เพทริเน็ตหลังยิงทรานสิชัน t_2

รีชเอบิลิตีทรี [11]

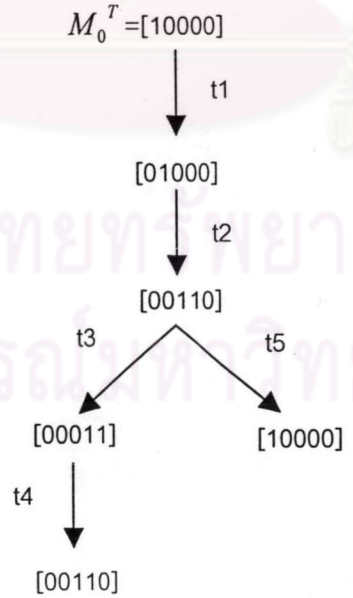
การวิเคราะห์แบบจำลองเพทริเน็ตนั้น การหาสภาวะของระบบสามารถพิจารณาได้จากมาร์กกิงของเพทริเน็ตที่เวลาใดๆ โดยเริ่มพิจารณาจากมาร์กกิงเริ่มต้น M_0 เมื่อเราทำการยิงทรานสิชันด้วยลำดับการยิงทรานสิชันชุดหนึ่งทำให้ได้มาร์กิงชุดใหม่คือ M_r ซึ่งนิยามได้ว่า มาร์กกิง M_r เป็นรีชเอเบิลมาร์กกิง (Reachable Marking) ของ M_0 ถ้ามีลำดับของการยิงทรานสิชันที่สามารถทำให้มาร์กกิงเปลี่ยนจาก M_0 เป็น M_r โดยเซตของมาร์กกิง M_r ทั้งหมดเรียกว่า เซตของรีชเอเบิลมาร์กกิง $R(M_0)$

ในการแสดงความสัมพันธ์ของการลำดับการยิงทรานสิชันและมาร์กกิง M_r เราสามารถใช้รีชเอบิลิตีทรีแสดงความสัมพันธ์ดังกล่าวได้ รีชเอบิลิตีทรีเป็นการแสดงเซตของมาร์กกิงที่สามารถเกิดขึ้นได้ในระบบ วิธีการหามาร์กกิง M_r คือการหาความเป็นไปได้ทั้งหมดของการยิงทรานสิชัน ซึ่งแสดงโดยใช้แผนภูมิต้นไม้ โดยการยิงทรานสิชันจะสิ้นสุดเมื่อการทำงานของระบบครบทุกขั้นตอนหรือสภาวะของระบบซ้ำกับสภาวะที่ผ่านมาแล้ว พิจารณาตัวอย่างเพทริเน็ตในรูป



รูปที่ 4.5 เพทริเน็ตที่ใช้วิเคราะห์รีชเอบิลิตีทรี

จากรูป มาร์กกิงเริ่มต้นของระบบคือ $M_0 = [10000]^T$ กำหนดให้เป็นโนดเริ่มต้น ซึ่งทรานสิชัน t_1 อีนาเบิลอยู่ เมื่อเราทำการยิงทรานสิชัน t_1 แล้วเราจะทำการสร้างโนดใหม่ในรีชเอบิลิตีทรี ซึ่งจะทำให้ได้มาร์กกิง $M_1 = [01000]$ ถ้าเราต้องการหาเซตของรีชเอเบิลมาร์กกิงทั้งหมด ทำได้โดยการยิงทรานสิชันต่อไปเรื่อยๆ ซึ่งหากทำการยิงทรานสิชันแล้วจะได้รีชเอบิลิตีทรีดังรูป



รูปที่ 4.6 รีชเอบิลิตีทรีของเพทริเน็ต

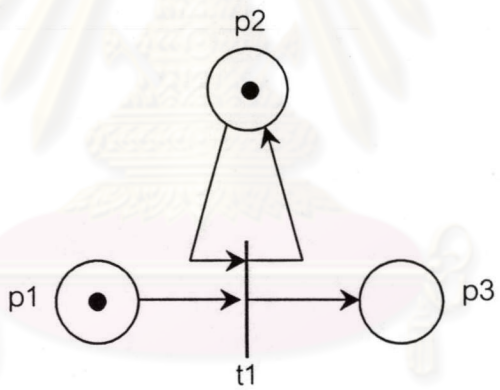
คุณสมบัติของเพทริเน็ต [11,12]

การใช้เพทริเน็ตในการจำลองการทำงานของระบบ การวิเคราะห์หาคุณสมบัติของแบบจำลองนับเป็นข้อดีที่สำคัญในการใช้เพทริเน็ตจำลองการทำงาน กล่าวคือลักษณะการทำงานของระบบสามารถแสดงได้โดยใช้คุณสมบัติของระบบ คุณสมบัติของเพทริเน็ตที่มีความสำคัญประกอบด้วย คุณสมบัติ Safeness, Boundedness, Liveness

1. Safeness

เพทริเน็ตจะมีคุณสมบัติ Safeness ถ้าทุกๆ เพลสในเพทริเน็ตมีโทเค็นอยู่ภายในไม่เกินหนึ่งตลอดการทำงาน

คุณสมบัติ Safeness เป็นคุณสมบัติที่มีความสำคัญในการสร้างอุปกรณ์ฮาร์ดแวร์ คือหากเพทริเน็ตมีคุณสมบัติข้อนี้แล้วทุกเพลสในเพทริเน็ตจะมีค่าเพียง 0 และ 1 เท่านั้น ทำให้สามารถออกแบบระบบด้วยอุปกรณ์ที่มีลักษณะเพียงการเปิด/ปิดได้



รูปที่ 4.7 เพทริเน็ตที่มีคุณสมบัติ Safeness

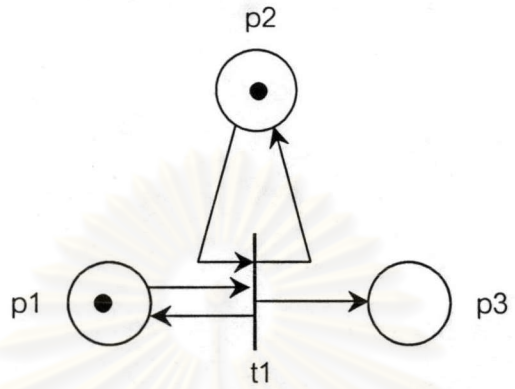
จากรูป แสดงเพทริเน็ตที่มีคุณสมบัติ Safeness เพราะตลอดการทำงาน ทั้งก่อนยิงทรานสิชันและหลังยิงทรานสิชันไม่มีเพลสใดที่มีโทเค็นเกินหนึ่ง

2. Boundedness

เพทริเน็ตจะมีคุณสมบัติ Boundedness ถ้าทุกๆ เพลสในเพทริเน็ตมีโทเค็นอยู่ภายในไม่เกินค่าคงที่จำนวนเต็มค่าหนึ่งตลอดการทำงาน

คุณสมบัติ Boundedness แสดงถึงความสามารถในการสร้างฮาร์ดแวร์ของระบบ หากเพทริเน็ตไม่มีคุณสมบัตินี้ทำให้ไม่สามารถสร้างฮาร์ดแวร์จากแบบจำลองได้

คุณสมบัติ Boundedness เป็นกรณีทั่วไปของคุณสมบัติ Safeness ซึ่งเป็นการใช้เคาน์เตอร์ (Counter) ในการนับแทนอุปกรณ์ที่มีลักษณะเพียงการเปิด/ปิดได้ ซึ่งหากแบบจำลองมีคุณสมบัติ Safeness แล้วจะมีคุณสมบัติ Boundedness ด้วย



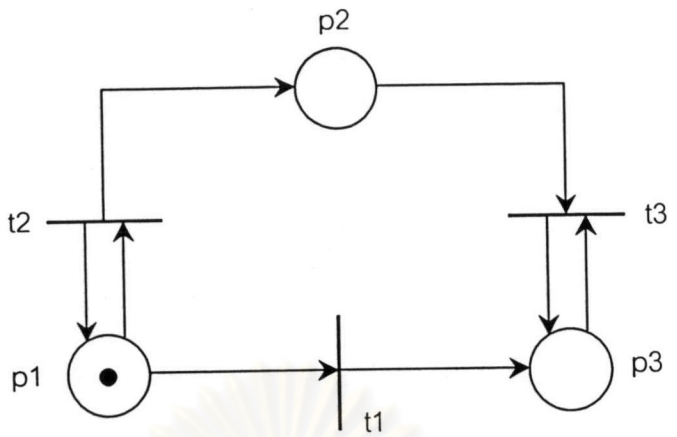
รูปที่ 4.8 เพทริเน็ตที่ไม่มีคุณสมบัติ Boundedness

จากรูป แสดงเพทริเน็ตที่ไม่มีคุณสมบัติ Boundedness เนื่องจากทรานสิชัน t_1 อีนาเบิลตลอดการทำงาน ทำให้เมื่อยิงทรานสิชัน t_1 สองครั้งติดกัน ทำให้เพลส p_2 มีโทเค็นสองโทเค็น ดังนั้นหากเมื่อยิงทรานสิชัน t_1 ไปเรื่อย ๆ ทำให้ไม่สามารถหาค่าคงที่ที่ทำให้จำนวนที่แสดงจำนวนโทเค็นในเพลส p_2 ได้

3. Liveness

เพทริเน็ตจะมีคุณสมบัติ Liveness เมื่อเทียบกับมาร์กกิงเริ่มต้น M_0 ถ้าทุกๆ รีช - เอเบิลมาร์กกิง M_r ซึ่งเป็นสมาชิกอยู่ในเซตของรีชเอเบิลมาร์กกิง $R(M_0)$ สามารถยิงทรานสิชันบางทรานสิชันได้ จากคุณสมบัติดังกล่าว ในระหว่างการทำงานของระบบ ที่มาร์กกิง M_r ใดๆ ถ้าไม่มีทรานสิชันใดสามารถยิงได้ทำให้ไม่สามารถเปลี่ยนสถานะของระบบได้

คุณสมบัติ Liveness สามารถบอกถึงโอกาสของการหยุดทำงานของระบบได้ โดยหากเพทริเน็ตไม่มีคุณสมบัติ Liveness แล้วทำให้ระบบสามารถเกิด Deadlock ได้ สภาวะ Deadlock คือสภาวะที่ระบบหยุดทำงาน ไม่สามารถทำงานต่อได้



รูปที่ 4.9 เพทริเน็ตที่ไม่มีคุณสมบัติ Liveness

แบบจำลองเพทริเน็ตในรูปไม่มีคุณสมบัติ Liveness เมื่อเทียบกับมาร์กกิงเริ่มต้น $M_0 = [100]$ เนื่องจากหลังจากยิงทรานสิชัน t_1 แล้วทำให้มาร์กกิงของระบบคือ $M_1 = [001]$ นั่นคือรีชเอเบิลมาร์กกิง $M_r = [001]$ ไม่มีทรานสิชันใดเลยสามารถยิงทรานสิชันได้

การวิเคราะห์คุณสมบัติของเพทริเน็ตนั้นสามารถพิจารณาได้จากโนดต่างๆ ของรีชเอบิลิตีทรีดังนี้

1. เพทริเน็ตมีคุณสมบัติ Safeness ถ้าทุกๆ เพลสในทุกๆ โนดของรีชเอบิลิตีทรีมีโทเค็นอยู่ไม่เกินหนึ่ง
2. เพทริเน็ตมีคุณสมบัติ Boundedness ถ้าทุกๆ เพลสในทุกๆ โนดของในรีชเอบิลิตีทรีโทเค็นอยู่ภายในไม่เกินค่าคงที่จำนวนเต็ม
3. เพทริเน็ตมีคุณสมบัติ Liveness เมื่อเทียบกับมาร์กกิงเริ่มต้น M_0 ถ้าทุกๆ โนดแสดงมาร์กกิงที่สามารถยิงทรานสิชันได้

ตัวอย่างการจำลองระบบโดยใช้เพทริเน็ต [8]

ขั้นตอนการจำลองระบบทางกายภาพโดยใช้เพทริเน็ตนั้น เริ่มจากการกำหนดสภาวะและเหตุการณ์ที่อาจเกิดได้ โดยเหตุการณ์ที่เกิดขึ้นจะทำให้สภาวะของระบบเปลี่ยนไป โดยสภาวะก่อนเกิดเหตุการณ์ (Pre-Condition) จะถูกแทนด้วยอินพุตเพลสในเพทริเน็ต และสภาวะหลังจากเกิดเหตุการณ์ (Post-Condition) จะถูกแทนด้วยเอาต์พุตเพลส จากนั้นจึงนำเอาสถานะดังกล่าวมาเขียนความสัมพันธ์ร่วมกันตามเงื่อนไขต่างๆ ตามที่กำหนดไว้

พิจารณาตัวอย่างขั้นตอนการผลิตในโรงงานอุตสาหกรรม เงื่อนไขการทำงานคือ เครื่องจักรจะเริ่มทำงานได้ก็ต่อเมื่อมีใบสั่งสินค้าและเครื่องจักรว่างจากการทำงาน โดยเครื่องจักรจะทำงานอย่างต่อเนื่องจนเสร็จแล้วจึงหยุดทำงาน จากเงื่อนไขการทำงานดังกล่าวสามารถสรุปสภาวะและเหตุการณ์ของระบบได้ดังนี้

สภาวะประกอบด้วย p_1 : โรงงานรอใบสั่งสินค้า

p_2 : ใบสั่งสินค้าอยู่ที่โรงงานแต่ยังไม่ได้สั่งให้ผลิต

p_3 : เครื่องจักรกำลังทำงาน

p_4 : งานที่สั่งทำการผลิตเสร็จสิ้นแล้ว

p_5 : เครื่องจักรไม่ได้ทำงาน

p_6 : จัดส่งสินค้าให้ลูกค้า

เหตุการณ์ประกอบด้วย t_1 : โรงงานได้รับใบสั่งสินค้า

t_2 : สั่งให้เริ่มทำการผลิต

t_3 : เครื่องจักรทำการผลิตเสร็จ

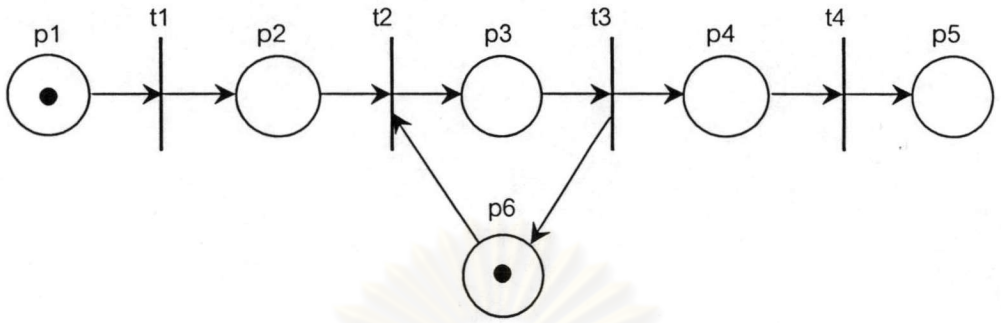
t_4 : มีคำสั่งให้ส่งสินค้าที่เสร็จแล้วออกไป

เมื่อพิจารณาการทำงานของเครื่องจักรเห็นว่าการสั่งให้เริ่มทำการผลิต จะต้องมีสภาวะของระบบสองอย่างเกิดขึ้นก่อน คือต้องมีใบสั่งสินค้าและเครื่องจักรไม่ได้ทำงานอยู่ และสภาวะที่เกิดหลังจากเหตุการณ์สั่งทำการผลิตคือเครื่องจักรกำลังทำงานและหลังจากทำงานเสร็จเครื่องจักรก็จะหยุดทำงาน จากการกำหนดสภาวะการทำงาน สามารถสรุปตารางสภาวะก่อนและหลังการเกิดเหตุการณ์ได้ดังนี้

เหตุการณ์	สภาวะก่อนเกิดเหตุการณ์	สภาวะหลังเกิดเหตุการณ์
t_1	p_1	p_2
t_2	p_2, p_6	p_3
t_3	p_3	p_4, p_6
t_4	p_4	p_5

ตารางที่ 4.1 ตารางสภาวะก่อนและหลังเกิดเหตุการณ์

แสดงเพทริเน็ตได้ดังรูป



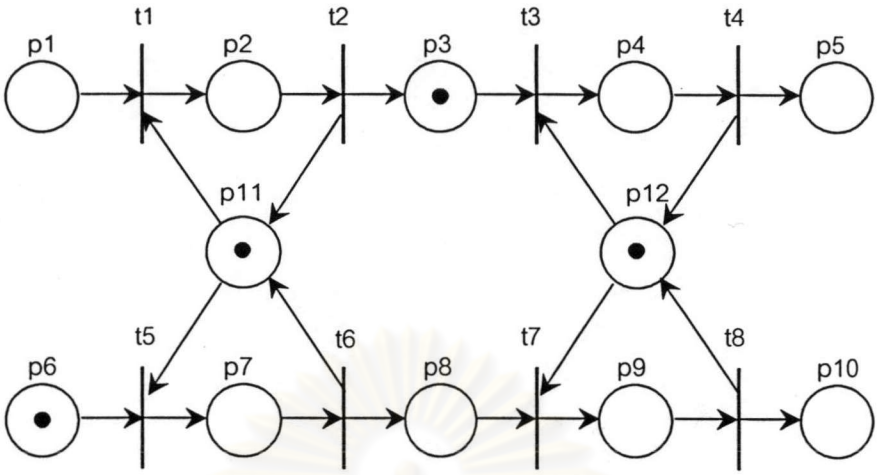
รูปที่ 4.10 เพทริเน็ตแสดงการจำลองขั้นตอนการผลิตในโรงงาน

การจำลองระบบโดยใช้เพทริเน็ตเป็นการจำลองการทำงานในสถานะการทำงานของระบบ สามารถแทนความหมายด้วยส่วนต่างๆ ของเพทริเน็ตดังนี้ สภาวะแทนด้วยเพลส เหตุการณ์แทนด้วยทรานสิชัน สภาวะก่อนเกิดเหตุการณ์แทนด้วยอินพุตเพลส สภาวะก่อนเกิดเหตุการณ์แทนด้วยเอาต์พุตเพลส การเกิดขึ้นของเหตุการณ์แทนด้วยการยิงทรานสิชัน และสภาวะของระบบในขณะนั้นแสดงโดยโทเค็นในเพลส เมื่อเกิดการยิงทรานสิชันจะเคลื่อนย้ายโทเค็นจากเพลสที่แสดงสภาวะก่อนเกิดเหตุการณ์ไปที่เพลสหลังเกิดเหตุการณ์

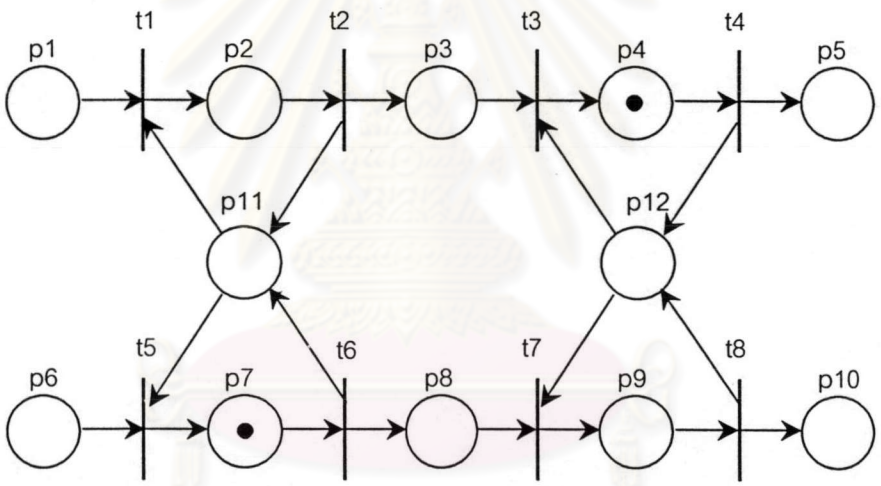
การทำงานพร้อมกันและการทำงานชนกัน [8]

การทำงานพร้อมกัน (Concurrency Task) และการทำงานชนกัน (Conflict Task) เป็นเหตุการณ์สำคัญที่อาจเกิดขึ้นได้ในทางปฏิบัติ โดยเหตุการณ์นี้สามารถใช้เพทริเน็ตอธิบายเหตุกาณ์รวมถึงพิจารณาทางเลือกที่เหมาะสมในการแก้ปัญหาได้

การทำงานพร้อมกันเป็นเหตุการณ์ที่แสดงถึงความสามารถของระบบในการทำงานมากกว่าหนึ่งอย่าง พิจารณาแบบจำลองเพทริเน็ตในรูปที่ 4.11 เห็นได้ว่าทรานสิชัน t_3 และ t_5 อีนาเบิลอยู่เมื่อทำการยิงทรานสิชัน t_3 และ t_5 พร้อมกันจะทำให้โทเค็นเคลื่อนที่ไปที่เพลส 4 และ 7 ซึ่งเห็นได้ว่าระบบทำงานอยู่ที่เพลส p_4 และ p_7 และทรานสิชันที่ t_3 และ t_5 คือการสั่งให้ระบบเริ่มทำงานแล้วหมายความว่าระบบกำลังทำงานทั้งสองอย่างอยู่นั่นเอง



(ก) ก่อนยิงทรานซิชัน t_3 และ t_5

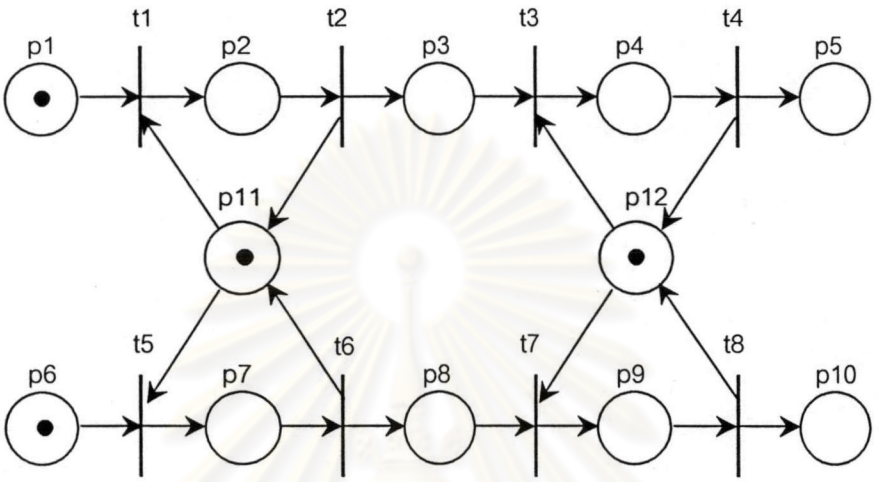


(ข) หลังยิงทรานซิชัน t_3 และ t_5

รูปที่ 4.11 รูปแบบการทำงานพร้อมกัน

การทำงานชนกันเป็นเหตุการณ์ที่แสดงว่าระบบสามารถทำงานอย่างใดอย่างหนึ่งเท่านั้น พิจารณาแบบจำลองเพทริเน็ตดังรูปที่ 4.12 ทรานสิชันที่ t_1 และ t_5 ถูกอีนาเบิลอยู่เมื่อเราทำการยิงทรานสิชัน t_1 แล้วทำให้ทรานสิชัน t_5 สูญเสียอีนาเบิลไป ในทางกลับกันการยิงทรานสิ - ชัน t_5 แล้วจะทำให้ทรานสิชันที่ t_1 สูญเสียอีนาเบิลไปเช่นเดียวกัน ซึ่งเป็นเหตุการณ์ดังกล่าวเกิดจากความสัมพันธ์กันของเหตุการณ์นั่นเอง ซึ่งหากมองว่าเพลสที่ p_2 และ p_7 คือสภาวะการทำงานของระบบและทรานสิชัน t_1 และ t_7 คือการสั่งให้ระบบเริ่มทำงานแล้วหมายความว่าระบบ

สามารถทำงานอย่างใดอย่างหนึ่งเท่านั้น ไม่สามารถทำงานสองอย่างในเวลาเดียวกันได้ และเพลสที่ทำหน้าที่แสดงความพร้อมของทรัพยากรในระบบนั้น จำนวนโทเค็นในเพลสจะแสดงถึงจำนวนทรัพยากรที่พร้อมทำงานหรือความสามารถในการรองรับการทำงานของระบบด้วย



รูปที่ 4.12 รูปแบบการทำงานขนกัน

ไทม์เพทริเน็ต [9]

จากกฎการทำงานของเพทริเน็ต เราสามารถหาสภาวะของระบบหลังจากการเกิดเหตุการณ์ต่างๆ แต่นอกจากการเกิดเหตุการณ์ใดๆ ภายในระบบแล้วยังอาจขึ้นกับเวลาที่เปลี่ยนไปด้วย ในการทำงานของระบบที่มีเวลาเป็นตัวกำหนดเราสามารถใช้เพทริเน็ตที่มีเงื่อนไขของเวลาหรือไทม์เพทริเน็ต (Timed Petri Net) เป็นตัวอธิบายการทำงานของระบบ โดยมีลักษณะการทำงานคือทรานสิชันจะถูกอีนาเบิลได้เมื่อเวลาเปลี่ยนไป

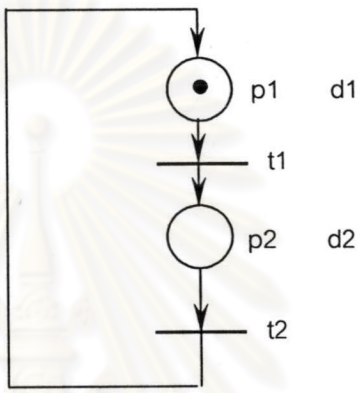
โครงสร้างของไทม์เพทริเน็ตประกอบด้วย

- $PN = (P, T, I, O, M_0)$
- $D = \{d_1, d_2, ..., d_n\}$ เป็นเซตของเวลาในแต่ละเพลส
  โดย n เป็นจำนวนเต็มที่มีค่ามากกว่าหรือเท่ากับศูนย์

โครงสร้างของไทม์เพทริเน็ตคือเพทริเน็ตที่มีเวลาการทำงานในแต่ละเพลสเพิ่มเติม กล่าวคือ โทเค็นใดๆ จะเคลื่อนที่ออกจากเพลส p_i ได้ก็ต่อเมื่อโทเค็นนั้นอยู่ในเพลส p_i แล้วเป็นเวลาอย่างน้อย d_i

จากเงื่อนไขข้างต้น ทำให้เราสามารถกล่าวได้อีกนัยหนึ่งว่า เมื่อเวลาผ่านไปแล้ว ทำให้โทเค็นในเพทริเน็ตนั้นพร้อมที่จะเคลื่อนที่ และทรานสิชันของไทม์เพทริเน็ตจะอีนาเบิลได้ก็ต่อเมื่อโทเค็นในเพทริเน็ตนั้นพร้อมที่จะเคลื่อนที่

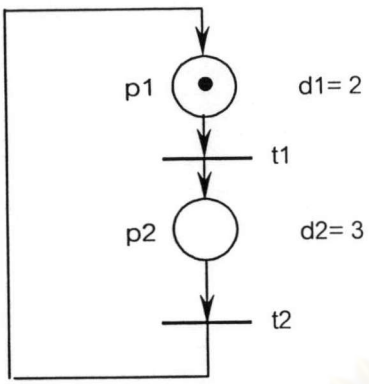
พิจารณาการทำงานในรูปที่ 4.13



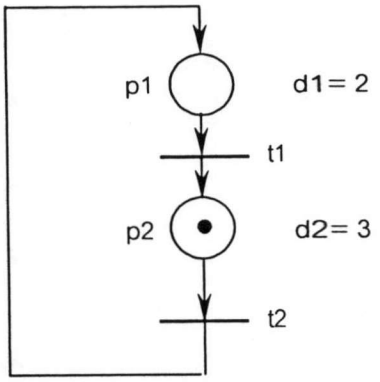
รูปที่ 4.13 ตัวอย่างของไทม์เพทริเน็ต

จากรูปเมื่อทรานสิชัน t_1 ถูกยิงแล้วโทเค็นจะเคลื่อนที่ไปที่เพลส p_2 และในขณะนั้นโทเค็นจะยังไม่พร้อมที่จะเคลื่อนที่ จนกระทั่งเวลาผ่านไปเป็นเวลา d_2 โทเค็นจะอยู่ในสภาวะพร้อมที่จะเคลื่อนที่ และทำให้ทรานสิชัน t_2 ถูกอีนาเบิล หลังจากนั้นเมื่อใดที่ทรานสิชัน t_2 ถูกยิงทรานสิชัน โทเค็นจะเคลื่อนที่ไปที่เพลส p_1 และในลักษณะเดียวกันโทเค็นในเพลส p_1 จะไม่พร้อมที่จะเคลื่อนที่จนกระทั่งเวลาผ่านไปเป็นเวลา d_1 โทเค็นดังกล่าวจึงพร้อมที่จะเคลื่อนที่และทำให้ทรานสิชัน t_1 ถูกอีนาเบิลเพื่อรอการยิงทรานสิชันต่อไป

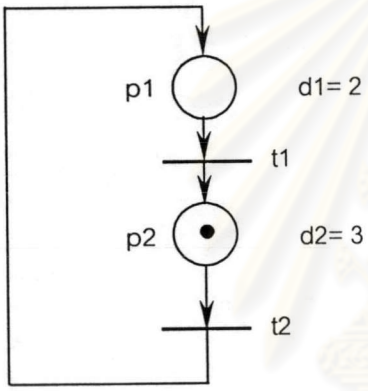
จากรูปของไทม์เพทริเน็ตในรูปที่ 4.13 กำหนดให้ $d_1 = 2$ และ $d_2 = 3$ แสดงการทำงานของเพทริเน็ตได้ดังรูป



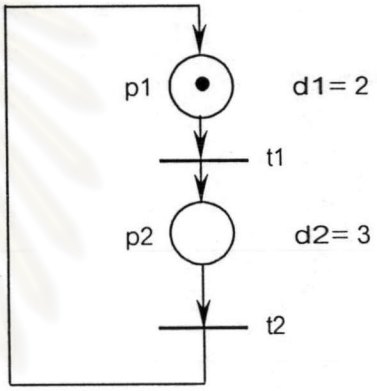
(ก) $0 \le t \le x$



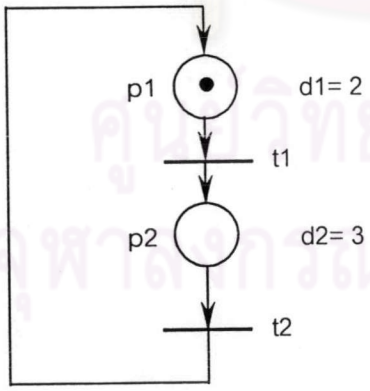
(ข) $x \le t \le x+3$



(ค) $x+3 \le t \le x+3+y$



(ง) $x+3+y \le t \le x+3+y+2$



(จ) $x+3+y+2 \le t \le .....$

รูปที่ 4.14 การเคลื่อนที่ของโทเค็นในไทม์เพทริเน็ต

จากรูปกำหนดให้ x และ y เป็นจำนวนจริงที่มีค่ามากกว่าหรือเท่ากับศูนย์ (ในกรณีที่ $x=0$ และ $y=0$ ทำให้เกิดกรณีที่ทำงานได้เร็วที่สุดคือเป็นการยิงทรานสิชันทันทีที่อีนาเบิล) การทำงานอธิบายได้ดังนี้

รูปที่ 4.14 (ก) แสดงสภาวะเริ่มต้นของการทำงานคือโทเค็นอยู่ที่เพลส p_1 เป็นเวลานานแล้ว ดังนั้นโทเค็นในเพลส p_1 จะพร้อมเคลื่อนที่และทรานสิชัน t_1 ถูกอีนาเบิล

รูปที่ 4.14 (ข) เมื่อถึงเวลา $t=x$ แล้วทรานสิชัน t_1 จะถูกยิงทำให้โทเค็นเคลื่อนที่จากเพลส p_1 ไปยังเพลส p_2 เนื่องจากที่เพลส p_2 มีเวลาในการทำงานคือ $d_2=3$ ดังนั้นที่เวลา $x<t<x+3$ โทเค็นที่อยู่ในเพลส p_2 จะยังไม่พร้อมที่จะเคลื่อนที่

รูปที่ 4.14 (ค) เมื่อถึงเวลา $t=x+3$ ซึ่งเป็นเวลาที่โทเค็นอยู่ในเพลส p_2 ครบเวลา ดังนั้นโทเค็นจะพร้อมที่จะเคลื่อนที่และทรานสิชัน t_2 ถูกอีนาเบิล จนกระทั่งเวลาผ่านไปเป็นเวลา y

รูปที่ 4.14 (ง) เมื่อถึงเวลา $t=x+3+y$ แล้วทรานสิชัน t_2 จะถูกยิงทำให้โทเค็นเคลื่อนที่จากเพลส p_2 ไปยังเพลส p_1 จากนั้นโทเค็นจะยังไม่พร้อมที่จะเคลื่อนที่จนกระทั่งโทเค็นอยู่ในเพลส p_1 เป็นเวลา $d_1=2$

รูปที่ 4.14 (จ) เมื่อถึงเวลา $t=x+3+y+2$ แล้วโทเค็นจะพร้อมที่จะเคลื่อนที่และทรานสิชัน t_1 ถูกอีนาเบิล และจะรอการยิงทรานสิชันต่อไป

สรุป

เพทริเน็ตเป็นเครื่องมือที่สามารถใช้จำลองและวิเคราะห์การทำงานของระบบได้ โดยสามารถแสดงได้ทั้งวิธีทางรูปภาพและวิธีทางคณิตศาสตร์ ขั้นตอนการนำเพทริเน็ตไปวิเคราะห์เริ่มจากการกำหนดสภาวะและเหตุการณ์ที่อาจเกิดขึ้นได้ในระบบ เขียนแบบจำลองเพื่อการแสดงความสัมพันธ์ของสภาวะและเหตุการณ์ต่างๆ จากนั้นจึงนำแบบจำลองที่สร้างขึ้นมาทำการวิเคราะห์สภาวะการทำงานของระบบ