บทที่ 3
การจัดการหน่วยประมวลกลาง
(CPU Manageme)

หน่วยประมวลมวลผลกลาง (CPU) เป็นองค์ประกอบสำคัญในระบบคอมพิวเตอร์ ถือว่าเป็นสมองของคอมพิวเตอร์ โดยมีหน้าที่หลักคือ อ่านและแปลชุดคำสั่ง ประมวลผลชุดคำสั่ง ติดต่อกับหน่วยความจำ ติดต่อรับส่งข้อมูลกับผู้ใช้ และย้ายข้อมูลตลอดจนคำสั่งระหว่างหน่วยงาน เป็นต้น ในระบบคอมพิวเตอร์ที่สามารถสลับการทำงานหลายงานพร้อมๆ กัน (Multi-tasking) ได้ระบบปฏิบัติการ (Operating System) จะต้องทำการแบ่งเวลาให้กับงานหรือกระบวนการ (Process) ในแต่ละงาน สามารถประมวลผลได้อย่างรวดเร็วเสมือนว่าทำงานได้หลายๆ งานได้ในเวลาเดียวกัน

ในบทนี้จะกล่าวถึงวิธีการจัดการกับหน่วยประมวลผลกลาง (CPU Management) ซึ่งเกี่ยวข้องกับการจัดกระบวนการ (Process) การจัดตารางการทำงาน (CPU Scheduling) การประยุกต์ใช้ขั้นตอนวิธี (Algorithm) เข้าไปช่วยในการจัดตารางการทำงานในระบบที่มีหน่วยประมวลผลกลางหลายตัว (Multiprocessor) การใช้งานระบบการประมวลผลแบบทันที (Real Time) เพื่อใช้ในการจัดกระบวนการทำงานให้เกิดประสิทธิภาพสูงสุด (ศัพท์บัญญัติราชบัณฑิตยสถาน, 2544)

3.1 การจัดตารางการทำงานของหน่วยประมวลผลกลาง (CPU Scheduling)

การจัดตารางการทำงานของหน่วยประมวลผลกลาง (CPU Scheduling) เป็นหน้าที่พื้นฐานของระบบปฏิบัติการ เกือบจะทุกระบบคอมพิวเตอร์ก่อนกระบวนการ (Process) จะมีการใช้งานทรัพยากรต่างๆ หน่วยประมวลผลกลางจะเป็นส่วนที่ใช้จัดตารางการใช้ทรัพยากรดังนี้

3.1.1 ระดับการจัดตารางการทำงาน (Level Scheduling) คือ การเลือกกระบวนการ (Process) หรือการจัดสรรทรัพยากรที่มีจำนวนจำกัดให้กับกระบวนการ แบ่งออกเป็น 2 แบบ คือ

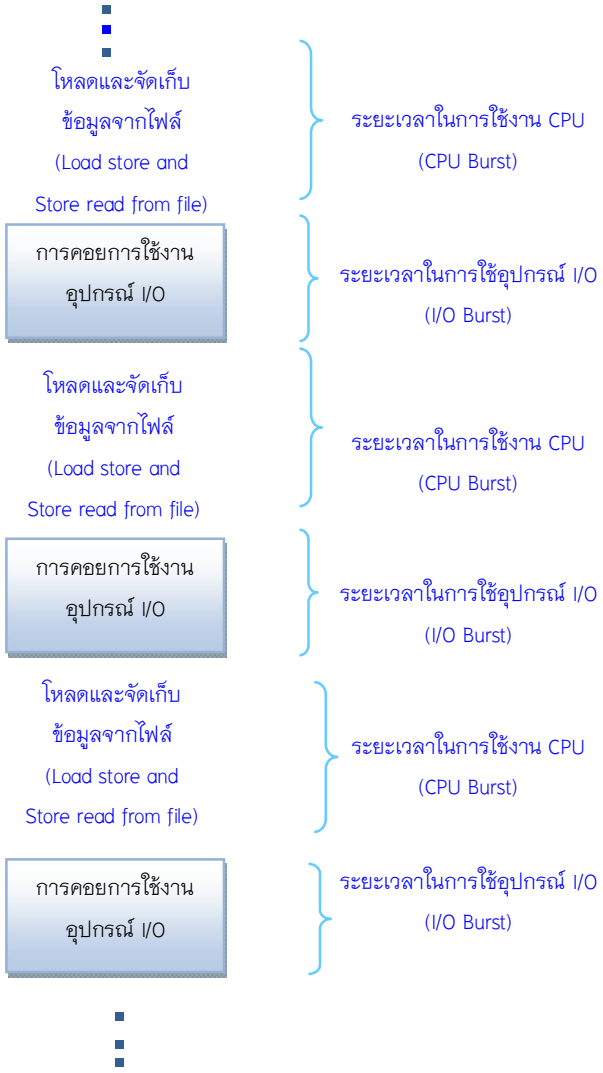
1. การจัดตารางการทำงานในระดับบน (High-Level Scheduling) เป็นการจัดลำดับงานให้กับกระบวนการ (Process) ที่จะเข้าไปใช้งานทรัพยากรใน CPU แบบกลุ่มซึ่งมีจำนวนกระบวนการ (Process) มากเกินกว่าระบบจะรับได้ในช่วงเวลาเดียวกัน ดังนั้น ระบบปฏิบัติการ

จะมีหน้าที่จัดลำดับความเหมาะสมให้แต่ละกระบวนการ (Process) เพื่อให้เกิดประโยชน์สูงสุด โดยคำนึงถึงปัจจัยต่างๆ ได้แก่ ลำดับความสำคัญ (Priority) และการเกิดภาวะการติดตาย (Deadlock) เป็นต้น

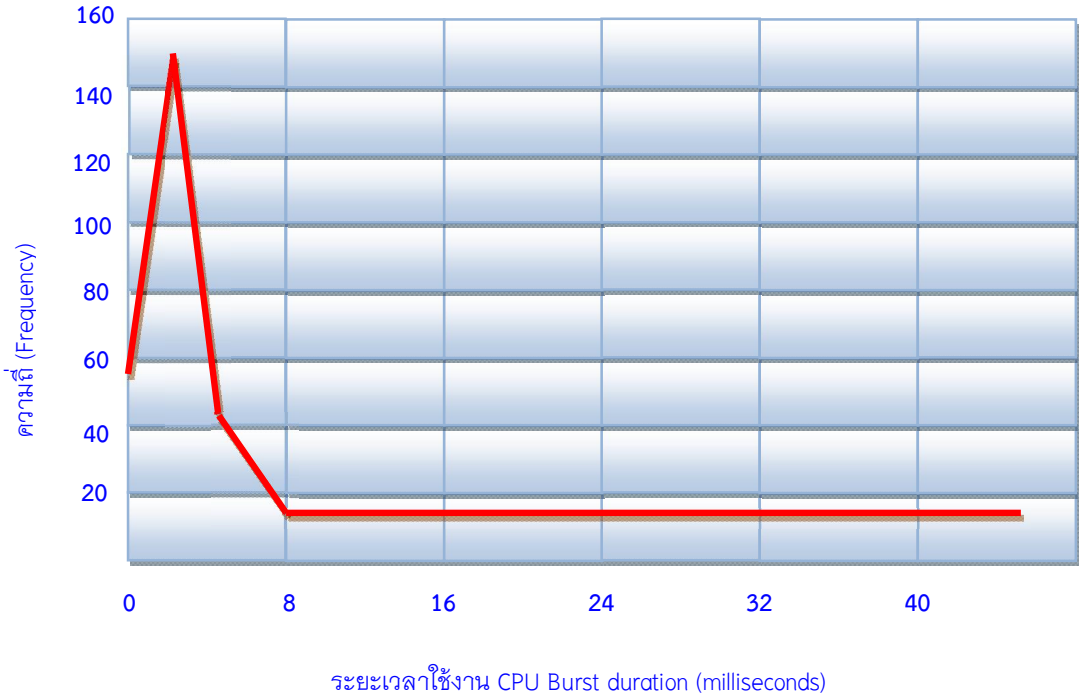
2. การจัดตารางการทำงานในระดับล่าง (Low–Level Scheduling) เป็นการเลือกและส่งกระบวนการ (Process) ที่จะเข้าไปใช้งานทรัพยากร CPU โดยเลือกกระบวนการที่มีสถานะพร้อม (Ready State) จากคิวโดยจะถูกส่งโดยตัวจัดการเวลา (Dispatcher)

3.1.2 ระยะเวลาใช้งานและรับส่งของมูลของ CPU (CPU–I/O Burst Cycle) ความสำคัญของการจัดเวลาของ CPU นั้น ขึ้นอยู่กับคุณลักษณะของกระบวนการ (Process) ซึ่งโดยทั่วๆ ไป การประมวลผลกระบวนการ (Process Execute) จะประกอบด้วยรอบเวลา (Cycle) ที่ใช้งาน CPU และเวลาที่ต้องคอยใช้งานอุปกรณ์ I/O ในขณะที่มีการประมวลผลกระบวนการ (Process Execute) จะมีการสลับการทำงานระหว่าง 2 ช่วงเวลานี้เท่านั้น คือ ระยะเวลาใช้งาน (CPU Burst) และระยะเวลาในการรับ – ส่งข้อมูล (I/O Burst) ซึ่งจะไม่เกิดขึ้นพร้อมกัน การประมวลผล (Execute) มักจะเริ่มจากการใช้งาน CPU แล้วก็จะตามด้วย การคอยการใช้งานอุปกรณ์ I/O หลังจากใช้งานอุปกรณ์ I/O เสร็จ ก็จะกลับมาใช้งาน CPU ต่อ สลับกันไปเรื่อยๆ จนกว่าจะเสร็จสิ้นการประมวลผล ซึ่งการสิ้นสุดการเสร็จสิ้นการประมวลผลนี้มักจะใช้เวลา CPU (CPU Burst) มากกว่าการรอคอยในระยะเวลาในการรับ – ส่งข้อมูล (I/O Burst) แสดงได้ดังภาพที่ 3.1

จากการค้นคว้าที่ผ่านมา ได้มีการศึกษาถึงลักษณะช่วงเวลาของการใช้งาน CPU จนสามารถมองเห็นคร่าวๆ ว่าลักษณะช่วงเวลาของการใช้งาน CPU ในแต่ละกระบวนการ (Process) จะมีรูปแบบอย่างไร ถึงแม้ว่ารูปแบบของเวลาจะมีลักษระแตกต่างกันอยู่บ้างแต่ก็มีแนวโน้มว่าแต่ละกระบวนการ(Process) จะมีคาบเวลาในการใช้งาน CPU คล้ายๆ กัน แสดงได้ดังภาพที่ 3.2



ภาพที่ 3.1 แสดงการสลับการทำงานระหว่างระยะเวลาใช้งานและระยะเวลาในการรับ– ส่งข้อมูล (Alternating sequence of CPU and I/O bursts)



ภาพที่ 3.2 แสดงฮีสโตแกรมของระยะเวลาการใช้งาน CPU (Histogram of CPU–burst times)

ตัวจัดตารางการทำงานของ CPU (CPU Scheduler)

เมื่อใดก็ตามที่ CPU ว่างระบบปฏิบัติการ (Operating System) จะต้องเข้ามาดำเนินการนำเอากระบวนการ (Process) ใดกระบวนการหนึ่งที่คอยอยู่ในคิวเข้ามาใช้งาน CPU โดยการเลือกกระบวนการ (Process) เพื่อเข้ามาใช้งาน CPU นี้ จะถูกจัดการด้วยส่วนที่เรียกว่าตัวจัดการเวลาช่วงสั้น (Short–Term Scheduler) หรือ ตัวจัดการเวลา CPU ตัวจัดการเวลานี้จะเลือกกระบวนการ (Process) ที่อยู่ในหน่วยความจำที่พร้อมในการประมวลผล (Execute) ที่สุดเพื่อให้ครอบครองเวลาการใช้งาน CPU และทรัพยากรที่เกี่ยวข้องกับกระบวนการนั้น (Process) นั้นภายใต้บล็อกควบคุมกระบวนการ (Process Control Block : PCBs) ซึ่งมีอัลกอริทึมในการจัดการคิวที่มีความพร้อม (Ready Queue) ให้เลือกใช้หลากหลาย เช่น คิวแบบมาก่อนได้รับบริการก่อน (FIFO Queue) คิวแบบเรียงตามลำดับอาวุโส (A Priority Queue) ต้นไม้ (Tree) หรือลิงค์ลิสต์แบบไม่เรียงลำดับ (Unordered Linked List) เป็นต้น

การให้สิทธิ์การจัดเวลา (Preemptive Scheduling)

การตัดสินใจของ CPU ในการเลือกว่ากระบวนการใด (Process) จะถูกประมวลผล (Execute) ขึ้นอยู่กับสถานการณ์ดังต่อไปนี้

1. เมื่อมีการกระบวนการ (Process) เปลี่ยนสถานะของจากสถานะที่ประมวลผลอยู่ (Running State) ไปเป็นสถานะคอย (Waiting State) เช่น การร้องขอใช้งานอุปกรณ์ I/O เป็นต้น
2. เมื่อมีการเปลี่ยนสถานะของ process จากสถานะรัน เป็นสถานะพร้อม
3. เมื่อมีการเปลี่ยนสถานะของ process จากสถานะคอย เป็นสถานะพร้อม
4. เมื่อ process เสร็จสิ้นไปแล้ว

ตัวจัดการเวลา (Dispatcher)

เป็นคอมโพเนนต์ (Component) ที่สำคัญอีกตัวหนึ่งที่เกี่ยวข้องกับฟังก์ชันในการจัดเวลาในหน่วยประมวลผลกลาง (CPU) ซึ่งเป็นโมดูลที่ทำหน้าที่ควบคุมการครอบครองเวลาใน CPU ของกระบวนการ (Process) โดยฟังก์ชันนี้จะประกอบด้วย

- การย้ายไปในส่วนของอธิบายขยายความ (Switching Context)
- การย้ายไปในส่วนของผู้ใช้ (User Mode)
- การกระโดด (Jumping) ไปยังตำแหน่งที่เหมาะสมของชุดคำสั่ง เพื่อที่จะเริ่มรันชุดคำสั่งใหม่อีกครั้ง

นอกจากนี้ตัวจัดการเวลา (Dispatcher) ควรมีการทำงานที่เร็วที่สุดเท่าที่จะทำได้ เพราะว่าตัวจัดการเวลาจะต้องทำงานทุกครั้งที่มีการย้ายกระบวนการ (Process) ซึ่งเวลาที่ถูกใช้ไปกับการทำการแบบนี้เรียกว่า Dispatch Latency

เกณฑ์ในการจัดตารางการทำงาน (Scheduling Criteria)

ใช้ในการเปรียบเทียบความแตกต่างของอัลกอริทึมที่ใช้สำหรับกระบวนการ (Process) ในการเข้ามาใช้งาน CPU หรือเปรียบเทียบวิธีการจัดตารางการทำงาน CPU ว่าวิธีใดเหมาะสมที่สุด ดังนี้

1. การใช้ CPU ให้เกิดประโยชน์ (CPU Utilization) สูงสุด เช่น 40 % สำหรับการบรรจุเบา (Lightly Load) และไม่ควรเกิน 90 % สำหรับการบรรจุหนัก (Hardily Load)
2. ปริมาณงาน (Throughput) ขณะที่มีการประมวลผล จะมีจำนวนกระบวนการ (Process) ที่ทำงานเสร็จสมบูรณ์ภายในหนึ่งหน่วยเวลา (Per Time Unit) เท่าใด

3. การวนรอบการทำงาน (Turnaround Time) คือ ระยะเวลาที่กระบวนการ (Process) เริ่มเข้าไปประมวลผลจนกระทั้งทำงานเสร็จสมบูรณ์ หรืออาจจะเป็นเวลาที่กระบวนการ (Process) รอเพื่อที่จะเข้าไปใช้งาน CPU รออยู่ในแถวคอยเพื่อเปลี่ยนสถานะเป็นพร้อม (Ready) ในการประมวลผล (Execute) บน CPU หรือรอเพื่อใช้งานอุปกรณ์ I/O เป็นต้น

4. เวลาที่ใช้คอย (Waiting Time) รออยู่ในแถวคอยเพื่อเปลี่ยนสถานะเป็นพร้อม (Ready)

5. เวลาในการตอบสนอง (Respond Time) ระยะเวลาที่กระบวนการ (Process) ใช้ในการสนองต่อคำสั่งจากผู้ใช้ที่ร้องขอ (Request) จนกระทั่งได้ผลลัพธ์ส่งกลับมายังผู้ใช้ โดยที่เวลาในการตอบสนองมักจะขึ้นอยู่กับความเร็ว (Speed) ของอุปกรณ์แสดงผล (Output Device)

3.2 อัลกอริทึมสำหรับการจัดตารางการทำงาน (Scheduling Algorithms)

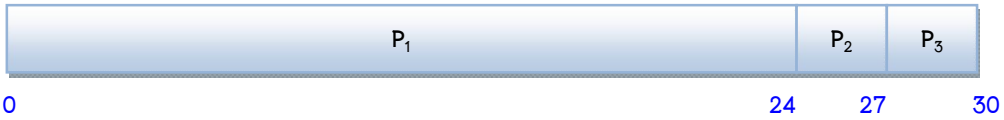
หน้าที่หลักของตัวจัดการคิวคือ การคัดเลือกกระบวนการซึ่งรออยู่ในสถานะพร้อมที่เหมาะสมที่สุดให้ปรับเปลี่ยนให้อยู่ในสถานะรัน (Run) เพื่อได้ครอบครองและใช้งานหน่วยประมวลผลกลาง ดังนั้นจึงเป็นหน้าที่หลักของงระบบปฏิบัติการ โดยใช้ชื่อเรียกว่าตัวส่ง (dispatcher) ทำการคัดเลือกกระบวนการและเรียกให้ตัวส่ง (dispatcher) ส่งกระบวนการที่ถูกเลือกแล้วเข้าไปใช้งานในหน่วยประมวลผลกลาง โดยมีขั้นตอนวิธี (Algorithm) ในการจัดการดังนี้ (ศัพท์บัญญัติ ราชบัณฑิตยสถาน, 2544)

1. การจัดเวลาแบบมาก่อนได้ก่อน (FCFS : First-come First-served Scheduling)

การจัดคิวแบบมาก่อนได้ก่อน FCFS (first-come-first-served) เป็นวิธีที่ง่ายที่สุด คือ กระบวนการไหนเข้ามารอในคิวก่อนก็จะได้ครอบครองหรือใช้บริการซีพียูก่อน ตามลำดับเวลาของการเข้ามาอยู่ในคิว คือ "มาก่อนได้รับบริการก่อน" กระบวนการที่ได้ครอบครองซีพียูจะทำงานไปจนเสร็จ โดยไม่ระยะเวลาอื่นมาเกี่ยวข้อง แต่ถ้ากระบวนการมีการเรียกใช้งานอุปกรณ์อินพุต-เอาต์พุต (I/O Device) หรือรอเหตุการณ์บางอย่าง ของกระบวนการนั้นอยู่จำเป็นจะต้องปลดปล่อยซีพียู และออกจากสถานะรัน (Running State) ไปอยู่ในสถานะขัดจังหวะ (Interrupted) เมื่อใดที่กระบวนการทำงานเสร็จสิ้นลงหรือเกิดเหตุการณ์ที่กำลังรออยู่กระบวนการนั้นก็จะปรับเปลี่ยนสถานะกลับเข้าไปต่อท้ายคิวใหม่อีกครั้ง ซึ่งการเปลี่ยนสถานะของกระบวนการโดยใช้การจัดคิวแบบ FCFS ซึ่งจะเห็นว่าแตกต่างกับรูปแสดงการเปลี่ยนสถานะของกระบวนการที่เคยกล่าวมาคือ ไม่มีการเปลี่ยนสถานะของกระบวนการจากสถานะรัน (Running State) มายังสถานะพร้อม (Ready State) โดยตรง

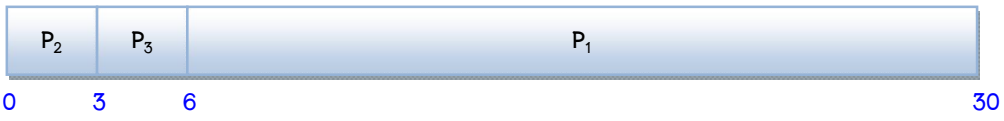
กระบวนการ (Process)	เวลาในการใช้งาน CPU (Burst Time)
P_1	24
P_2	3
P_3	3

ตัวอย่าง ถ้ามีกระบวนการ (Process) เข้ามาใช้งานตามลำดับดังนี้ P_1, P_2, P_3 และใช้วิธีการแบบ FCFS แสดงอยู่ในรูปแบบ Gantt chart ได้ดังนี้



P_1 ใช้เวลาในการคอย 0 มิลลิวินาที (millisecond), P_2 ใช้เวลาในการคอย 24 มิลลิวินาที (millisecond), P_3 ใช้เวลาในการคอย 27 มิลลิวินาที (millisecond) ดังนั้นใช้เวลาเฉลี่ยในการค่อยเป็น (0 + 24 + 27)/3 = 17 มิลลิวินาที (millisecond)

แต่ถ้ากระบวนการ (Process) เข้ามาใช้งานเปลี่ยนลำดับเป็น P_2, P_3, P_1 แสดงอยู่ในรูปแบบ Gantt chart ได้ดังนี้



ดังนั้นใช้เวลาเฉลี่ยในการค่อยเป็น (6 + 0 + 3)/3 = 3 มิลลิวินาที (millisecond) ซึ่งจะเห็นว่าเวลาเฉลี่ยในการคอยลดลง ดังนั้นสรุปได้ว่าเวลาเฉลี่ยในการคอยจะขึ้นอยู่กับวิธีการเรียงลำดับหรืออัลกอริทึมที่เลือกใช้และอาจเปลี่ยนแปลงไปตามขนาดของกระบวนการ (Process) ที่เข้ามาใช้งานเวลาในหน่วยประมวลผลกลาง

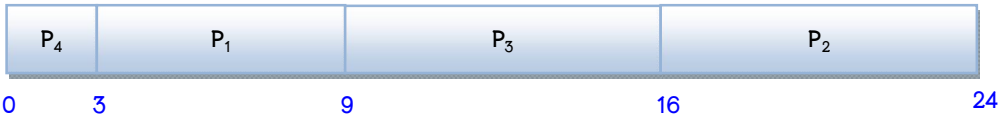
2. การจัดเวลาแบบงานสั้นทำก่อน (SJF: Short–Job–First Scheduling)

การจัดคิวแบบงานสั้นทำก่อน SJN (shortest job next) การคัดเลือกกระบวนการด้วยวิธีนี้จะคัดเลือกเอากระบวนการที่ต้องการใช้เวลาในการทำงานในหน่วยประมวลผลกลางน้อยที่สุดทำให้กระบวนการที่ต้องการเวลาในการทำงานน้อยจบออกไปได้เร็วขึ้น จำนวน

กระบวนการในระบบที่รออยู่ในคิวก็จะมีจำนวนลดลง ซึ่งก็จะทำให้เวลาทำงานโดยเฉลี่ยเสร็จเร็วขึ้น แต่วิธีการจัดลำดับคิวแบบนี้ก็มีข้อเสียตรงที่หากกระบวนการที่ต้องการเวลาในการทำงานนานก็จะต้องเสียเวลาค่อย (Waiting times) มากด้วยเช่นกัน

กระบวนการ (Process)	เวลาในการใช้งาน CPU (Burst Time)
P_1	6
P_2	8
P_3	7
P_4	3

ตัวอย่าง ถ้ามีกระบวนการ (Process) เข้ามาใช้งานตามลำดับดังนี้ P_1 , P_2, P_3, P_4 และใช้วิธีการแบบ SJF แสดงอยู่ในรูปแบบ Gantt chart ได้ดังนี้



P_1 ใช้เวลาในการคอย 3 มิลลิวินาที (millisecond), P_2 ใช้เวลาในการคอย 16 มิลลิวินาที (millisecond), P_3 ใช้เวลาในการคอย 9 มิลลิวินาที (millisecond) และ P_4 ใช้เวลาในการคอย 0 มิลลิวินาที (millisecond) ดังนั้นใช้เวลาเฉลี่ยในการค่อยเป็น (3 + 16 + 9 + 0)/4 = 7 มิลลิวินาที (millisecond) แต่ถ้าใช้วิธีการแบบ FCFS จะใช้เวลาเฉลี่ยในการค่อยถึง (0 + 6 +14 + 21)/4 = 10.25 มิลลิวินาที (millisecond)

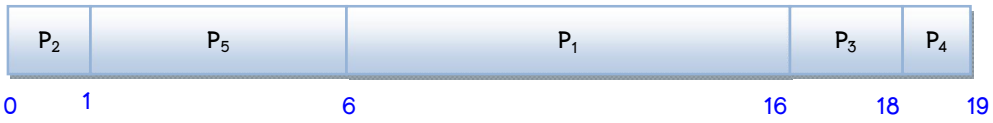
3. การจัดเวลาตามลำดับความสำคัญ (Priority Scheduling)

การจัดคิวแบบลำดับความสำคัญ (priority queue) คิว แบบลำดับความสำคัญมีลักษณะแตกต่างกับคิวธรรมดา ภายในคิวจะมีการจัดเรียงลำดับของกระบวนการต่าง ๆ ตามลำดับความสำคัญของกระบวนการนั้น กระบวนการที่อยู่ต้นคิวจะมีลำดับความสำคัญมากที่สุด และลดลงเรื่อย ๆ กระบวนการที่อยู่ท้ายคิวคือกระบวนการที่มีลำดับความสำคัญต่ำสุด การคัดเลือกกระบวนการจะเอากระบวนการที่อยู่ต้นคิว (มีลำดับความสำคัญสูงสุด) เข้าไป

ครอบครองซีพียูก่อน ดังนั้นถึงแม้ว่ากระบวนการที่เข้าคิวทีหลังแต่มีลำดับความสำคัญสูงกว่าก็อาจได้ เข้าไปครอบครองซีพียูก่อน เช่น กระบวนการ P_5 เข้าคิวเป็นกระบวนการหลังสุด แต่จะได้ครอบครองซีพียูก่อนกระบวนการ P_3 และ P_4 เพราะมีความสำคัญสูงกว่า แต่ถ้ากรณีที่กระบวนการมีลำดับความสำคัญเท่ากันจะใช้วิธีมาก่อนได้ก่อนในการช่วยพิจารณา

กระบวนการ (Process)	เวลาในการใช้งาน CPU (Burst Time)	ลำดับความสำคัญ (Priority)
P_1	10	3
P_2	1	1
P_3	2	4
P_4	1	5
P_5	5	2

ตัวอย่าง ถ้ามีกระบวนการ (Process) เข้ามาใช้งานตามลำดับดังนี้ P_1, P_2, P_3, P_4, P_5 และใช้วิธีการแบบตามลำดับความสำคัญ Priority Queue แสดงอยู่ในรูปแบบ Gantt chart ได้ดังนี้

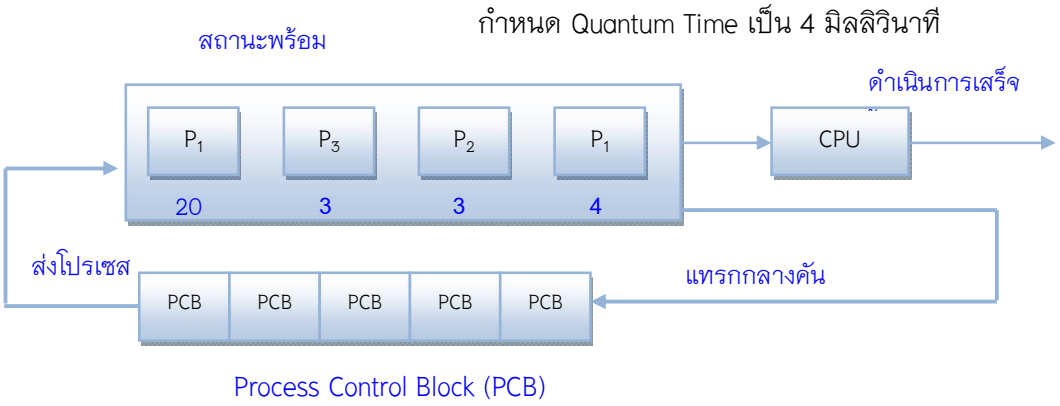


P_1 ใช้เวลาในการคอย 6 มิลลิวินาที (millisecond), P_2 ใช้เวลาในการคอย 0 มิลลิวินาที (millisecond), P_3 ใช้เวลาในการคอย 16 มิลลิวินาที (millisecond) และ P_4 ใช้เวลาในการคอย 18 มิลลิวินาที (millisecond) และ P_1 ใช้เวลาในการคอย 1 มิลลิวินาที (millisecond) ดังนั้นใช้เวลาเฉลี่ยในการค่อยเป็น (6 + 0 + 16 + 18 + 1)/5 = 8.2 มิลลิวินาที (millisecond)

4. การจัดเวลาแบบวนรอบ (RR : Round-Robin Scheduling)

การจัดคิวแบบการวนรอบ (round-robin) เป็นการจัดลำดับคิวแบบมีการวนรอบ ลักษณะคือมีการคัดเลือกกระบวนการในคิวในรูปแบบ FCFS คือ "มาก่อนได้รับบริการก่อน" แต่ต่างกันตรงมีการครอบครองซีพียูของกระบวนการในสถานะใช้งาน (Running State) ที่ถูกจำกัดเวลาไว้ด้วยระยะเวลาควอนตัม (Quantum Time) ทำให้กระบวนการที่ต้องการเวลาในการทำงานนานจะต้องเปลี่ยนสถานะวนรอบ (round-robin) ระหว่างสถานะ พร้อม (Ready State) และสถานะใช้งาน (Running State) การจัดคิวแบบการวนรอบสามารถแก้ปัญหาการคอยนาน

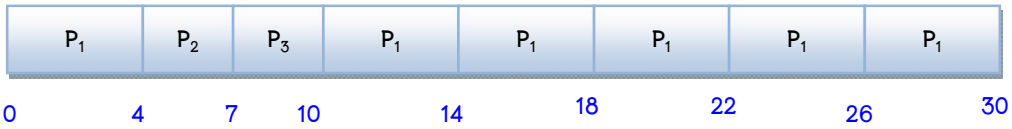
ของแต่ละกระบวนการที่ต้องการเวลาทำงานในหน่วยประมวลผลกลางน้อย ๆ ถ้าระบบกำหนดเวลาควอนตัมเป็น 4 มิลลิวินาที กระบวนการ P_1 ต้องมีการวนรอบเปลี่ยนสถานะระหว่างสถานะรันและสถานะพร้อม 6 ครั้ง กระบวนการ P_2 1 ครั้งกระบวนการ P_3 1 ครั้ง เมื่อกระบวนการ P_1 เข้าไปอยู่ในสถานะรันครั้งแรกและกลับออกมา กระบวนการ P_2 จะได้ครอบครองซีพียูได้และ ทำงานเสร็จกระบวนการ P2 จบและออกจากระบบไปเลย กระบวนการถัดไปที่จัดได้ครอบครองซีพียูคือ P_3 กระบวนการ P_3 จะครอบครองซีพียู 3 มิลลิวินาทีจนกระทั่งกระบวนการ P_3 จบ เหลือกระบวนการ P_1 เพียงกระบวนการเดียว จนเข้ามาดำเนินการใช้งานหน่วยประมวลผลกลางจนเสร็จสิ้น โดยมีตัวควบคุมการส่งกระบวนการให้เข้าไปในคิวที่เรียกว่า Process Control Block (PCB) แสดงได้ดังภาพที่ 3.3 เป็นแผนภาพแสดงการจัดคิวแบบการวนรอบ (round-robin) ของกระบวนการทั้ง 3 กระบวนการ



ภาพที่ 3.3 แสดงการครอบครองหน่วยประมวลผลกลางแบบวนรอบ (round-robin)

กระบวนการ (Process)	เวลาในการใช้งาน CPU (Burst Time)
P_1	24
P_2	3
P_3	3

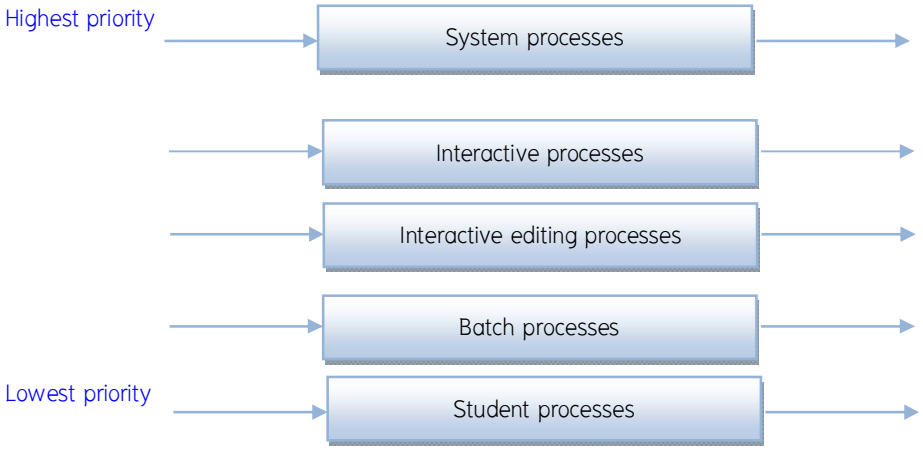
ตัวอย่าง ถ้ามีกระบวนการ (Process) เข้ามาใช้งานตามลำดับดังนี้ P_1, P_2, P_3 และใช้วิธีการแบบวนรอบ (Round-Robin) และใช้เวลาควอนตัม (Time Quantum) เป็น 4 มิลลิวินาที (millisecond) แสดงอยู่ในรูปแบบ Gantt chart ได้ดังนี้



P_1 ใช้เวลาในการคอย 0 มิลลิวินาที (millisecond), P_2 ใช้เวลาในการคอย 4 มิลลิวินาที (millisecond), P_3 ใช้เวลาในการคอย 3 มิลลิวินาที (millisecond) ดังนั้นใช้เวลาเฉลี่ยในการค่อยเป็น [(10–4)+ 4 + 7]/3 = 17/3 = 5.66 มิลลิวินาที (millisecond)

5. การจัดเวลาแบบคิวหลายระดับ (Multilevel Queue Scheduling)

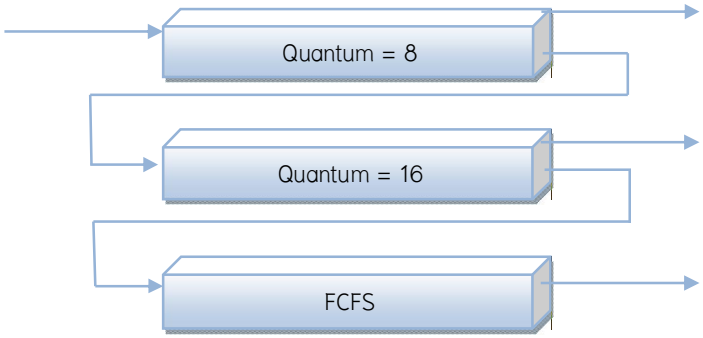
เป็นวิธีการจัดคิวโดยการแบ่งคิวออกเป็นระดับ (Partition) แสดงได้ดังภาพที่ 3.4 มีการกำหนดคุณลักษณะของแต่ละกระบวนการ (Process) เช่น ขนาดของหน่วยความจำ ลำดับความสำคัญของกระบวนการ และชนิดของกระบวนการ แต่ละคิวจะมีอัลกอริทึมการจัดเวลาเป็นของตัวเอง และมีจัดลำดับความสำคัญของแต่ละกระบวนการในรูปแบบกระบวนการลำดับความสำคัญต่ำสุด (Lowest Priority) ไปจนถึงกระบวนการลำดับความสำคัญสูงสุด (Highest Priority)



ภาพที่ 3.4 แสดงการจัดเวลาแบบคิวหลายระดับ (Multilevel Queue Scheduling)

6. การจัดเวลาแบบคิวหลายระดับแบบย้อนกลับ (Multilevel Feedback Queue Scheduling)

เป็นการจัดเวลาแบบลำดับคิวเพื่อให้เกิดประสิทธิภาพสูงสุด จึงมีการผสมผสานการจัดการลำดับคิวหลาย ๆ เทคนิคมาประยุกต์รวมเข้าด้วยกัน โดยคำนึงถึงลำดับความสำคัญ (Piority Queue) ของงาน ทำให้งานที่มีความสำคัญเหมือนกันอยู่ในคิวเดียวกันและให้งานสำคัญน้อย ๆ อยู่ในคิวที่สำคัญน้อยด้วยเช่นกัน แสดงได้ดังภาพที่ 3.5



ภาพที่ 3.5 แสดงการจัดเวลาแบบคิวหลายระดับแบบย้อนกลับ (Multilevel Feedback Queue Scheduling)

3.3 การจัดตารางการทำงานของระบบหลายกระบวนการ (Multiple-Processor Scheduling)

การทำงานในลักษณะที่มีหน่วยประมวลผลกลางหลายตัวต่างก็มีระบบการจัดเวลาของตัวเอง จะต้องมีการออกแบบอย่างระมัดระวังในเรื่องของการที่กระบวนการแต่ละกระบวนการอาจจะต้องการใช้ข้อมูลในฐานข้อมูลในเวลาเดียวกัน รวมทั้งต้องระวังไม่ให้หน่วยประมวลผลกลางว่างงานพร้อมกัน

ในระบบคอมพิวเตอร์บางระบบที่มีการพัฒนาเพื่อให้ได้ประสิทธิภาพสูงสุด ได้มีการใช้หน่วยประมวลผลกลางแยกต่างหากระหว่างงานภายในของระบบ และกับงานที่เป็นของผู้ใช้ ซึ่งการทำงานแยกกันของหน่วยประมวลผลกลางแบบนี้ ทำให้การออกแบบระบบปฏิบัติการมีความซับซ้อนน้อยลงมาก เนื่องจากมีหน่วยประมวลผลกลางเพียงตัวเดียวที่จะเข้าไปใช้ข้อมูลของระบบปฏิบัติการ เพราะว่า หน่วยประมวลผลกลางตัวอื่นๆ ไม่ว่าจะกี่ตัวก็ตาม จะเข้าถึงได้เฉพาะข้อมูลของผู้ใช้ภายนอกเท่านั้น

3.4 การจัดตารางการทำงานของระบบแบบทันที (Real-Time Scheduling)

Hard real-time คือระบบที่สามารถทำงานใดงานหนึ่งให้เสร็จตามเวลาที่กำหนดได้ ซึ่งงานที่จะรับเข้ามาแต่ละงานนั้นจะมีความต้องการของเวลาที่ต้องการให้ เสร็จมาด้วย ดังนั้นตัวจัดเวลาจะต้องเป็นตัวตัดสินใจว่าจะรับงานเข้ามาทำหรือไม่

Soft real-time คือระบบที่แบ่งเวลาธรรมดาที่มีการให้ระดับความสำคัญแก่งานบางประเภท หรืองานที่ถูกเลือกไว้ล่วงหน้าว่าเป็นงานเร่งด่วน ซึ่งอาจทำให้เกิดปัญหาของการทำงานในระดับต่ำ ๆ อาจไม่ได้รับเวลาของหน่วยประมวลผลกลางเลย

3.5 รูปแบบของขั้นตอนวิธี (Algorithm Evolution)

การกำหนดกฎเกณฑ์และวิธีการเลือกใช้อัลกอริทึมมักจะถูกกำหนดจากการเข้าไปใช้งาน **หน่วยประมวลผลกลาง** โดยมีเวลาในการโต้ตอบ (Respond Time) หรือปริมาณงาน (Throughput) ที่ได้ของแต่ละกระบวนการ ดังนั้นการเลือกใช้ขั้นตอนวิธี (Algorithm) จะต้องคำนึงถึงการวัดประสิทธิภาพการใช้งานโดยพิจารณาจากสิ่งต่อไปนี้(ศัพท์บัญญัติราชบัณฑิตยสถาน, 2544)

1. การใช้ประโยชน์จาก**หน่วยประมวลผลกลาง**ได้สูงสุดภายใต้ข้อกำหนดของเวลาในการตอบสนอง (Maximum Respond Time) ให้ได้อย่างรวดเร็วภายในหนึ่งวินาที (หนึ่งหน่วยเวลา)
2. การใช้ประโยชน์จาก**หน่วยประมวลผลกลาง**ได้สูงสุดของปริมาณงาน (Throughput) ที่ได้เมื่อเทียบกับอัตราส่วนของเวลาในการประมวลผล (Execution Time) กับจำนวนกระบวนการที่ทำงานเสร็จสมบูรณ์ภายในหนึ่งหน่วยเวลา ตั้งแต่กระบวนการนั้นถูกส่งไปใช้งานจนกระทั่งได้รับผลลัพธ์กลับมาโดยมีอัลกอริทึมหรือรูปแบบให้เลือกใช้ดังนี้

- **Deterministic Modeling**

  วิธีนี้เป็นวิธีการคัดเลือกที่เรียกว่า analytic evaluation ซึ่งจะนำเอาขั้นตอนวิธี (Algorithm) ชนิดต่างๆ และลักษณะของงานมาสร้างสูตร เพื่อใช้ในการคำนวณหาตัวเลขของประสิทธิภาพที่สามารถวัดและเปรียบเทียบได้

- **Queuing Models**

  ลักษณะของงานที่เข้ามาในระบบคอมพิวเตอร์นั้นมักจะมีลักษณะที่ไม่แน่นอน ในแต่ละวันที่มีการใช้ระบบคอมพิวเตอร์นั้น งานต่างๆ ที่เข้ามาอาจมีลักษณะที่ไม่ซ้ำกันเลย อย่างไรก็ตามมีสิ่งหนึ่งที่เราอาจจะสามารถทำนายหรือกำหนดได้ ก็คือการกระจายของเวลาในการใช้หน่วยประมวลผลกลางและการใช้เลือกอุปกรณ์นำเข้าหรือส่งออก (Input/output) ซึ่งเราสามารถที่จะกำหนดแบบคร่าวๆ ได้ ซึ่งก็เช่นเดียวกันกับเวลาของการมาถึงระบบของงานต่างๆ ที่เราก็สามารถกำหนดไว้แบบไม่เฉพาะเจาะจงเช่นกัน

- **Simulations**

  วิธีการจำลองระบบ การที่เราจะเลือกวิธีการ หรือเลือกใช้ขั้นตอนวิธี (Algorithm) ที่ถูกต้องต่อระบบใดๆ อย่างเป็นจริงเป็นจังแล้ว เราสามารถใช้วิธีการของการจำลองระบบ ซึ่งวิธีการนี้จะสามารถคำนวณตัวเลขต่างๆ ออกมาได้อย่างเที่ยงตรงมากขึ้น การทำการจำลองระบบในที่นี้จะเกี่ยวข้องกับการใช้ชุดคำสั่งคอมพิวเตอร์ ซึ่ง

จะต้องมีการเขียนชุดคำสั่งเพื่อใช้เป็นตัวแทนหรือหุ่นจำลองของระบบต่างๆ ในคอมพิวเตอร์ นอกจากนี้ยังต้องมี การเขียนชุดคำสั่งเพื่อเป็นตัวแทนของสิ่งแวดล้อมที่เกี่ยว ข้องกับระบบคอมพิวเตอร์นั้นๆ อีกด้วย

- **Implementation**

  วิธีการทดลองนำไปใช้งานจริง ซึ่งจำให้พบข้อผิดพลาดที่เกิดขึ้นได้ง่ายกว่าการสร้างแบบจำลองเพื่อมาทดสอบ โดยการสร้างขั้นตอนวิธี (Algorithm) ชนิดต่างๆ เพื่อทดลองใช้กับชุดคำสั่งกับระบบงานจริงๆ ที่ถูกกำหนดขึ้นมา

สรุป

การจัดการกับหน่วยประมวลผลกลาง (CPU Management) จึงเกี่ยวข้องกับการจัดกระบวนการ (Process) การจัดตารางการทำงาน (CPU Scheduling) การประยุกต์ใช้ขั้นตอนวิธี (Algorithm) เข้าไปช่วยในการจัดตารางการทำงานให้กับระบบคอมพิวเตอร์เพื่อให้เกิดประสิทธิภาพสูงสุด โดยมีขั้นตอนวิธี (Algorithm) ให้เลือกใช้หลายแบบด้วยกัน เช่น

1. การจัดเวลาแบบมาก่อนได้ก่อน (FCFS : First-come First-served Scheduling) เป็นอัลกอริทึมประเภทที่ไม่สามารถแทรกกลางคันได้ (Non-preemptive) เหมาะกับการใช้งานในระบบงานแบบกลุ่ม (Batch System) มีข้อดีคือง่ายต่อการประยุกต์ใช้งานแต่ก็มีข้อเสียที่ตามมาคือจะไม่สามารถคาดเดาเหตุการณ์ของแต่ละกระบวนการในการเข้าใช้งานหน่วยประมวลผลกลางได้

2. การจัดเวลาแบบงานสั้นทำก่อน (SJF: Short-Job-First Scheduling) เป็นอัลกอริทึมประเภทที่ไม่สามารถแทรกกลางคันได้ (Non-preemptive) เหมาะกับการใช้งานในระบบงานแบบกลุ่ม (Batch System) มีข้อดีคือ มีค่าเฉลี่ยเวลาในการรอคอยในการใช้งานหน่วยประมวลผลกลางแต่ละกระบวนการน้อยที่สุด ข้อเสียคืออาจทำให้เกิดปัญหาการขาดแคลนทรัพยากร (Starvation)

3. การจัดเวลาตามลำดับความสำคัญ (Priority Scheduling) เป็นอัลกอริทึมประเภทที่ไม่สามารถแทรกกลางคันได้ (Non-preemptive) เหมาะกับการใช้งานในระบบงานแบบกลุ่ม (Batch System) มีข้อดีคือ ลำดับงานที่มีความสำคัญจะได้รับการประมวลผลก่อนเสมอ ข้อเสียคืออาจทำให้เกิดปัญหาการขาดแคลนทรัพยากร (Starvation)

4. การจัดเวลาแบบวนรอบ (RR : Round-Robin Scheduling) เป็นอัลกอริทึมประเภทที่สามารถแทรกกลางคันได้ (Preemptive) เหมาะกับการใช้งานในระบบงานแบบโต้ตอบ (Interactive System) มีข้อดีคือ สามารถตอบสนองต่อผู้ใช้งานได้อย่างรวดเร็ว ข้อเสียคือต้องมีการกำหนดระยะเวลาการใช้งาน (Quantum Time) ที่แน่นอนให้กับแต่ละกระบวนการ

5. การจัดเวลาแบบคิวหลายระดับ (Multilevel Queue Scheduling) เป็นอัลกอริทึมประเภทที่สามารถแทรกกลางคันได้หรือไม่สามารถแทรกกลางคันได้ (Preemptive/Non-Preemptive) เหมาะกับการใช้การใช้งานในระบบงานแบบกลุ่ม (Batch System) และงานในระบบงานแบบโต้ตอบ (Interactive System) มีข้อดีคือ มีความยืดหยุ่นในการใช้งาน (Flexibility) ข้อเสียคือมี overhead ในการจากการตรวจสอบลำดับคิวการจัดตารางงานของระบบตัวประมวลผลหลายตัว (Multiprocessor)

คำถามทบทวน

1. การเลือกกระบวนการหรือจัดสรรทรัพยากรที่มีจำกัดให้กับกระบวนการแบ่งออกเป็นกี่แบบอะไรบ้าง

2. จงเปรียบเทียบและแสดงให้เห็นว่าระยะเวลาใช้งาน (CPU Burst) และระยะเวลาในการรับ – ส่งข้อมูล (I/O Burst) มีความเกี่ยวพันกันอย่างไร

3. จงแสดงขั้นตอนการให้สิทธิ์การจัดเวลา (Preemptive Scheduling) การตัดสินใจของหน่วยประมวลผลกลางในการเลือกว่ากระบวนการใด (Process) จะถูกประมวลผล (Execute) ขึ้นอยู่กับสถานการณ์ใดบ้าง

4. ตัวจัดการเวลา (Dispatcher) มีหน้าที่การทำงานอย่างไร

5. เกณฑ์ที่ใช้เปรียบเทียบความแตกต่างของอัลกอริทึมที่ใช้สำหรับกระบวนการ (Process) ในการเข้ามาใช้งานหน่วยประมวลผลกลางหรือเปรียบเทียบวิธีการจัดตารางการทำงานหน่วยประมวลผลกลางว่าวิธีใดเหมาะสมที่สุดทำได้อย่างไรบ้าง

6. จากตารางที่ให้มาจงใช้อัลกอริทึมต่อไปนี้แสดงวิธีการหาเวลาเฉลี่ยในการค่อยว่าเป็นเท่าใด

กระบวนการ (Process)	เวลาในการใช้งาน CPU (Burst Time)	ลำดับความสำคัญ (Priority)
P_1	15	5
P_2	5	1
P_3	2	2
P_4	4	4
P_5	8	3

6.1 การจัดตารางเวลาแบบมาก่อนได้ก่อน (FCFS: First-come First-served Scheduling)

6.2 การจัดตารางเวลาแบบงานสั้นทำก่อน (SJF: Short-Job-First Scheduling)

6.3 การจัดเวลาตามลำดับความสำคัญ (Priority Scheduling)

7. จากตารางข้างล่างที่ให้มา จงแสดงวิธีเวลาเฉลี่ยของการจัดเวลาแบบวนรอบ (RR: Round-Robin Scheduling) โดยกำหนดให้เวลาควอนตัม (Quantum Time) 4 มิลลิวินาทีพร้อมวาดรูปประกอบคำอธิบายและแสดงรูปแบบ Gantt chart โดยมีตัวควบคุมการส่งกระบวนการเข้าไปในคิวที่เรียกว่า Process Control Black (PCB)

กระบวนการ (Process)	เวลาในการใช้งาน CPU (Burst Time)
P_1	27
P_2	2
P_3	3
P_4	8

8. จงอธิบายความแตกต่างระหว่างการจัดเวลาแบบคิวหลายระดับ (Multilevel Queue Scheduling) และการจัดเวลาแบบคิวหลายระดับแบบย้อนกลับ (Multilevel Feedback Queue Scheduling)

9. จงอธิบายความแตกต่างระหว่าง hard real-time และ soft real-time

10. จงอธิบายองค์ประกอบการเลือกใช้อัลกอริทึมจะต้องคำนึงถึงการวัดประสิทธิภาพในการใช้งานให้ได้ประโยชน์สูงสุดต่อไปนี้

10.1 Deterministic Modeling

10.2 Queue Models

10.3 Simulations

10.4 Implematation

เอกสารอ้างอิง

ราชบัณฑิตยสถาน. (2544). *ศัพท์บัญญัติ ราชบัณฑิตยสถาน*. ค้นเมื่อ 26 กรกฎาคม 2556, จาก: http://rirs3.royin.go.th/coinages

หน่วยประมวลผลกลาง. (2556). *วิกิพีเดีย*. ค้นเมื่อ 26 กรกฎาคม 2556, จาก: http://th.wikipedia.org/wiki

พีรพร หมุนสนิท, สุธี พงศาสุกุลชัย, อัจจิมา เลี้ยงอยู่. (2553). *ระบบปฏิบัติการ: Operating Systems*. กรุงเทพฯ : เคทีพี แอนด์ คอนซัลท์.

พีระพนธ์ โสพัศสถิตย์. (2552). *ระบบปฏิบัติการ*. Operating Systems. กรุงเทพฯ : สำนักพิมพ์จุฬาลงกรณ์มหาวิทยาลัย.

Silberschartz, Galvin, Gangne. (2011). *Operating System Concepts*. 8 th (ed), New York: McGra Hill.