# Operating system Homework\_2

Page fault

นาย ดรันภพ เป็งคำตา 580610642

**รูปแบบการทดลอง**

เป็นการทดลองเพื่อทดสอบการเกิด page fault ของอัลกอริทึม 3 แบบ คือ First In First Out (FIFO), Optimal และ Least Recently Used (LRU) โดยจะมีการสุ่ม reference string ขึ้นมาเพื่อทดสอบการทำงาน และจะให้แต่ละอัลกอริทึมทำงานตาม reference string นั้น ๆ โดย ภายในการทดลองเดียกัน ทั้ง3 อัลกอริทึมจะทำงานบน reference string ชุดเดียวกัน เพื่อทำการเปรียบเทียบ ประสิทธิภาพของการทำงาน และในแต่ละการทดลอง จะทำการทดลองย่อย ด้วยการเปลี่ยน ขนาดของ frame size ตั่งแต่ 1 ไปจนถึง ความหลากหลายของ reference string (Variety) และนำมา plot กราฟเพื่อแสดงถึงผลของการทำงาน

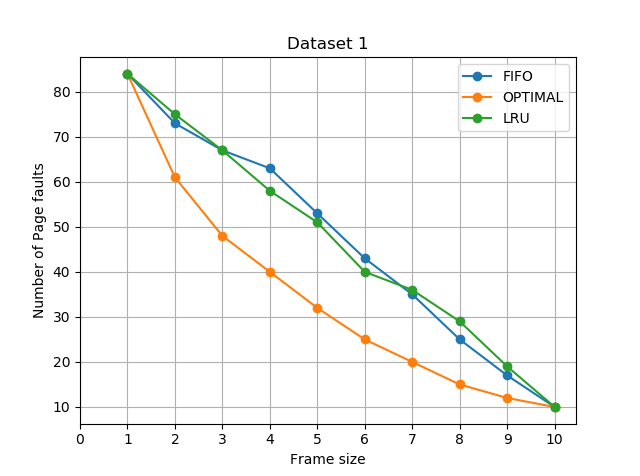
**การทดลองที่ 1**

**Reference String**

**4 5 1 4 6 0 0 5 9 5 3 3 7 7 9 9 5 6 0 9 9 3 4 1 6 8 5 2 2 0 2 7 0 7 7 2 4**

**2 7 9 1 8 0 0 3 7 4 8 0 5 7 9 4 2 2 4 1 9 6 4 4 9 3 4 7 8 9 9 0 3 0 2 1 5**

**1 5 0 6 7 5 9 9 2 7 0 0 6 6 9 5 8 9 9 9 6 0 6 7 0 3**

reference string ความยาว 100 ตัวอักษร มีค่าที่แตกต่างกัน 10 ค่า คือ { 0, 1, 2, 3, 4, ,5, 6, 7, 8, 9 } ที่มีโอกาศในการปรากฏเท่ากันทุกค่า ทดลองด้วย frame ขนาด 1 - 10

**การทดลอง 2**

**Reference String**

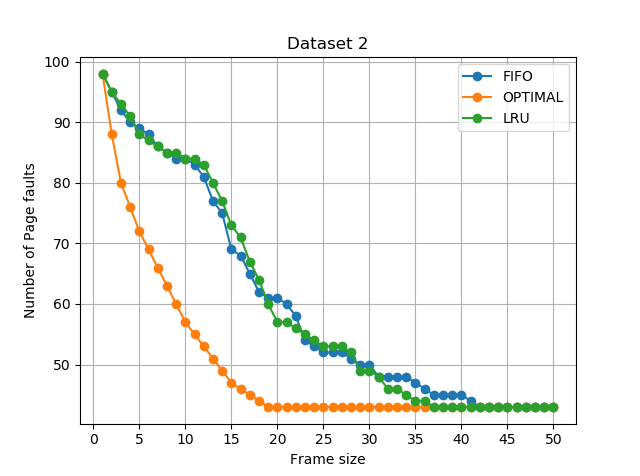
**20 19 27 38 42 5 13 5 16 1 13 37 12 24 6 30 4 3 19 11 32 17 25 26 15**

**26 42 13 16 29 5 30 19 47 11 9 17 12 34 39 48 47 36 37 20 27 40 30 16 27**

**25 40 12 39 43 14 14 21 2 22 34 9 32 14 37 38 9 10 33 9 47 16 1 10 16 2**

**2 26 31 43 44 35 44 5 36 14 38 9 34 10 49 35 47 34 46 19 1 20 18 32**

reference string ความยาว 100 ตัวอักษร มีค่าที่แตกต่างกัน 50 ค่า คือ 0 - 50 ที่มีโอกาศในการปรากฏเท่ากันทุกค่า และ ทดลองด้วย frame ขนาด 1 – 50



**สรุปผลการทดลอง**

จากการทดลองทั้ง 2 การทดลองทำให้เห็นว่า Optimal Algorithm ให้ผลการทำงานได้ดีที่สุด และในการทดลองที่ 2 จะเห็นว่า ตั้งแต่ frame ขนาด 19 เป็นต้นไป Optimal Algorithm การเกิด page fault จะเป็นคงคงที่ คือเกิด 43 ครั้ง

สำหรับ FIFO และ LRU จะเห็นได้ว่า การเกิดpage fault จะมีค่าใกล้เคียงกัน ในทั้ง 2 การทดลอง จึงได้ตั้งสมมุติฐานว่า ทั้งสองอัลกอริทึม จะทำงานให้ผลที่ขึ้นอยู่กับ การกระจายตัวของ reference string เนื่องจากทั้ง 2 การทดลองนี้มีการกระตัวแบบเดียวกัน จึงทำให้ไม่สามารถแยกความสามารถของทั้งสองอัลกอริทึมออกจากกันได้

**Code**

ส่วนของการเกิด page fault

def pageFault(frameSize, reference, type=0):

fault = 0

frame = np.full((frameSize), -1)

for i in range(len(reference)):

idx = checkPage(frame, reference[i])

if(idx == -1):

fault += 1

insIdx = checkPage(frame, -1)

if insIdx == -1: // Page fault Handler part

if type == 0:

frame = FIFO(frame, reference[i])

elif type == 1:

frame = OPTIMAL(frame, reference[i], reference[i:])

elif type == 2:

frame = LRU(frame, reference[i], reference[:i])

else:

frame[insIdx] = reference[i]

return fault

**FIFO**

def FIFO(frame, i):

return np.append(frame, i)[1:]

**Optimal**

def OPTIMAL(frame, o, reference):

nframe = np.copy(frame)

tmp = np.full((len(nframe)), len(reference))

for i in range(len(reference)):

for j in range(len(nframe)):

if tmp[j] == len(reference):

if reference[i] == nframe[j]:

tmp[j] = i

nframe[np.argmax(tmp)] = o

return nframe

**LRU**

def LRU(frame, o, reference):

nframe = np.copy(frame)

tmp = np.full((len(nframe)), 0)

for i in range(len(tmp)):

if tmp[i] == 0:

for j in range(len(reference)):

if nframe[i] == reference[len(reference) - j - 1]:

tmp[i] = len(reference) - j - 1

break

nframe[np.argmin(tmp)] = o

return nframe

**CheckPage**

def checkPage(frame, target):

index = -1

for i in range(len(frame)):

if frame[i] == target:

index = i

break

return index

**ใช้สำหรับตรวจสอบว่า มีข้อมูลดังกล่าวอยู่ใน frame หรือไม่**

**Reference string Generator**

def genReference(length, variety):

return np.random.choice(variety, length)