# MFQ 스케줄링 기법 결과 보고서

이윤영

글로벌경영학과_2019315330

2022 년 10 월 7 일

**세부 내용:**

- 3 개의 Ready queue 를 갖는 MFQ 스케줄링 기법 구현
  - ✓ Q0: Time quantum 2 인 RR 스케줄링 기법 사용
  - ✓ Q1: Time quantum 4 인 RR 스케줄링 기법 사용
  - ✓ Q2: SPN 스케줄링 기법 사용

- 모든 프로세스는 최초에 Q0 으로 진입하며, Qi 에서 스케줄 받아 실행하고 해당 queue 의 time quantum 을 소모할 경우 Qi+ 1 로 진입

- Q2 에서의 SPN 스케줄링은 Q2 에 있는 프로세스들 만을 대상으로 스케줄링 진행함

- Q0 의 우선순위가 가장 높고, Q2 의 우선순위가 가장 낮으며, 스케줄링은 항상 높은 우선순위의 queue 에서부터 이루어짐

- 각 프로세스의 단일 CPU burst 에 대해서만 처리함

**구현:**

- 입력: (입력 파일명 input.txt 로 작성)
  - ✓ 프로세스 개수 및 각 프로세스 별 도착 시간(AT), 수행 시간(BT) 등

- 출력:
  - ✓ 스케줄링 결과 (Gantt chart)
  - ✓ 각 프로세스 별 Turnaround Time(TT) 및 Waiting Time(WT)
  - ✓ 전체 프로세스의 평균 TT 및 평균 WT

- 고려사항:
  - ✓ 기타 고려사항이나 가정이 필요할 경우, 스스로 판단하여 rule 을 정하고, 이를 결과 보고서에 명시할 것
  - ✓ 출력 형태는 각자 자유롭게 결정; 단 출력 결과를 누구나 쉽게 파악할 수 있도록 해야 함
  - ✓ 다양한 입력을 사용하여 testing 하고 그 결과를 보일 것

**설계/구현:**

사용한 언어: Python

1. 우선 텍스트 파일에 정보 입력을 다음과 같이 한다:

```
Number of Processes = 5
PID1 = 1
PID2 = 2
PID3 = 3
PID4 = 4
PID5 = 5
Arrival Time P1 = 0
Arrival Time P2 = 1
Arrival Time P3 = 3
Arrival Time P4 = 5
Arrival Time P5 = 6
Burst Time P1 = 3
Burst Time P2 = 7
Burst Time P3 = 2
Burst Time P4 = 5
Burst Time P5 = 3
```

   다음과 같이 구현하는 이유로는 사용자의 편의성을 고려하여 입력하는 값이 정확하게 구분이 될 수 있도록 하기 위함에 있다. 단 한가지 해당 구조의 단점으로는 입력되는 값들이 한자리 이상(10 이상)을 받을 수 없다. 해당 구조를 준수하며 프로그램의 정상적인 작동에 집중하여 입력 값을 한자리 로만 받을 수 있도록 하는 제약을 두었다.

2. 텍스트 파일의 값들을 변수에 딕셔너리나 리스트의 형태로 저장한다. 그리고 첫 번째 ready queue 인 Q0 에 리스트 형태로 Arrival time 순으로 정렬하고 process 들을 [Process ID 값, Burst Time 값]으로 저장한다.

3. 첫 번째 ready queue 에서는 time quantum 이 2 이기 때문에 BT 가 2 보다 클 경우, 갠트차트에 CPU 할당 시간을 2 로 저장하고 2 보다 작거나 같으면 BT 값 그대로 저장한다. 그리고 Process 들의 BT 값에 -2 를 한다. 처리가 된 process 는 Q0 에서 삭제되며 BT 값이 2 보다 큰 process 들은 Q1 으로 처리 된 순서대로 옮겨진다.

4. 두 번째 ready queue Q1 에서는 RR 기법의 time quantum 이 4 를 가지고 있기 때문에 Q0 하고 동일하게 하되 조건 값을 2 가 아닌 4 로 한다. 이전과 동일하게 BT 값이 4 를 뺀 이후에도 0 이상인 Process 들은 마지막 ready queue 인 Q2 로 옮기고 Q1 에 있는 process 는 모두 삭제한다.

5. Q1 에서 받은 process 들을 남은 BT 값 순으로 정렬한 이후, 차례대로 처리한다. Q2 에는 Preemption 이 없기 때문에 남은 BT 값 그대로 갠트차트에 기록한다. 이렇게 process 들은 모두 처리된다.

6. Average Turnaround Time 과 Average Wait Time 은 다음과 같은 절차로 계산된다:

- Process 가 처리 될 때마다 time 이라는 리스트에 CPU 를 할당받은 시간을 숫자를 저장한다.
- Process 의 처리가 완료되면 할당 받은 시간을 저장하고 process 의 ID 를 저장한다.
- 해당 리스트를 통해 각 process 의 처리가 끝나는 시간을 도출하고 이를 TT 와 WT 값을 도출하는데 사용한다.

7. Gantt Chart 같은 경우는 process 들의 ID 와 CPU 할당 시간을 기록하여 시각화 한다.

**결과 화면:**

1.

입력 값:

```
Number of Processes = 5
PID1 = 1
PID2 = 2
PID3 = 3
PID4 = 4
PID5 = 5
Arrival Time P1 = 0
Arrival Time P2 = 1
Arrival Time P3 = 3
Arrival Time P4 = 5
Arrival Time P5 = 6
Burst Time P1 = 3
Burst Time P2 = 7
Burst Time P3 = 2
Burst Time P4 = 5
Burst Time P5 = 3
```

실행 결과:

```
Average Turnaround time:  11.8
Average Wait time:  7.8
Turnaround time of Processes:  [['P1', 11], ['P2', 19], ['P3', 3], ['P4', 13], ['P5', 13]]
Wait time of Processes:  [['P1', 8], ['P2', 12], ['P3', 1], ['P4', 8], ['P5', 10]]

Ganttchart(Visualized): P1 00 P2 00 P3 00 P4 00 P5 00 P1 0 P2 0000 P4 000 P5 0 P2 0

Ganttchart: [['P1', 2], ['P2', 2], ['P3', 2], ['P4', 2], ['P5', 2], ['P1', 1], ['P2', 4], ['P4', 3], ['P5', 1], ['P2', 1]]
```

2.

입력 값:

```
Number of Processes = 5
PID1 = 1
PID2 = 2
PID3 = 3
PID4 = 4
PID5 = 5
Arrival Time P1 = 1
Arrival Time P2 = 4
Arrival Time P3 = 2
Arrival Time P4 = 7
Arrival Time P5 = 5
Burst Time P1 = 4
Burst Time P2 = 5
Burst Time P3 = 8
Burst Time P4 = 2
Burst Time P5 = 3
```

실행 결과:

```
Average Turnaround time:  12.8
Average Wait time:  8.4
Turnaround time of Processes:  [['P1', 11], ['P2', 15], ['P3', 20], ['P4', 3], ['P5', 15]]
Wait time of Processes:  [['P1', 7], ['P2', 10], ['P3', 12], ['P4', 1], ['P5', 12]]

Ganttchart(Visualized): P1 OO P3 OO P2 OO P5 OO P4 OO P1 OO P3 OOOO P2 OOO P5 O P3 OO

Ganttchart: [['P1', 2], ['P3', 2], ['P2', 2], ['P5', 2], ['P4', 2], ['P1', 2], ['P3', 4], ['P2', 3], ['P5', 1], ['P3', 2]]
```

3.

입력 값:

```
Number of Processes = 7
PID1 = 1
PID2 = 2
PID3 = 3
PID4 = 4
PID5 = 5
PID6 = 6
PID7 = 7
Arrival Time P1 = 1
Arrival Time P2 = 4
Arrival Time P3 = 2
Arrival Time P4 = 7
Arrival Time P5 = 5
Arrival Time P6 = 8
Arrival Time P7 = 3
Burst Time P1 = 7
Burst Time P2 = 2
Burst Time P3 = 5
Burst Time P4 = 4
Burst Time P5 = 9
Burst Time P6 = 6
Burst Time P7 = 3
```

실행 결과:

```
Average Turnaround time:  21.428571428571427
Average Wait time:  11.428571428571429
Turnaround time of Processes:  [['P1', 32], ['P2', 4], ['P3', 19], ['P4', 21], ['P5', 31], ['P6', 24], ['P7', 19]]
Wait time of Processes:  [['P1', 25], ['P2', 2], ['P3', 14], ['P4', 17], ['P5', 22]]

Ganttchart(Visualized): P1 OO P3 OO P7 OO P2 OO P5 OO P4 OO P6 OO P1 OOOO P3 OOO P7 O P5 OOOO P4 OO P6 OOOO P1 O P5 OOO

Ganttchart: [['P1', 2], ['P3', 2], ['P7', 2], ['P2', 2], ['P5', 2], ['P4', 2], ['P6', 2], ['P1', 4], ['P3', 3], ['P7', 1], ['P5', 4], ['P4', 2], ['P6', 4
```

**실행결과를 보는 방법 및 제약 조건:**

- 제약 조건:
  - ✓ Process 개수, Process ID, Burst Time, 그리고 Arrival Time 은 모두 10 미만, 즉 한 자리 수 여야 한다.
  - ✓ 텍스트 파일 마지막 문장에서 맨 끝에 있는 텍스트 다음에는 공간이 없어야 한다. 한번 프로그램을 실행하면 마지막 문장에 띄어쓰기가 되는데, 이 상태에서 프로그램을 한번 더 실행하면 에러가 나기 때문에 띄어쓰기가 된 공간을 지워줘야 재실행이 가능하다.
  - ✓ 입력 값은 모두 정해져 있는 자리에 입력을 해야 하며, 띄어쓰기나 Backspace 를 하면 안된다.
  - ✓ Process 수를 추가하려면 우선 최 상단의 Process 수를 변경하고, 정해져 있는 형식 그대로 라인들을 추가해야 한다. 즉, 프로세스가 5 개인 상태에서 6 개로 바꾸려면:
    - Number of Process 를 6 으로 변경
    - PID = 5 밑에 PID = 6 추가,
    - Arrival Time P5 = 6 밑에 Arrival Time P5 = n 추가,
    - Burst Time P5 = 3 밑에 Burst Time P6 = n 추가
      (복사 붙여넣기를 하고 값만 바꾸면 편하다.)

이해를 돕기 위한 사진:

```
Number of Processes = 5
PID1 = 1
PID2 = 2
PID3 = 3
PID4 = 4
PID5 = 5
Arrival Time P1 = 0
Arrival Time P2 = 1
Arrival Time P3 = 3
Arrival Time P4 = 5
Arrival Time P5 = 6
Burst Time P1 = 3
Burst Time P2 = 7
Burst Time P3 = 2
Burst Time P4 = 5
Burst Time P5 = 3
```

- 실행결과를 보는 법:
  - ✓ Turnaround time of Processes 와 Wait time of processes 는 각 Process 의 PID 에 대한 Turnaround time 과 Wait time 을 보여준다. 예를 들어 Turnaround time of Processes 같은 경우, 리스트 안에 첫번째 값 (예:P1)이 PID 이며, 그 다음에 오는 숫자가 Turnaround time 이다.

✓ Gantt chart 의 경우, Process 의 ID 가 먼저 나오며, 해당 Process 가 CPU 를 얼만큼 할당 받았는지 보여준다. “O”가 1 Time 을 뜻하며 “OO”인 경우에는 2 번의 Time 동안 CPU 를 할당 받았다는 뜻이 된다. 아래에 이해를 돕기 위한 결과 화면이 있다.

```
Average Turnaround time:  11.8
Average Wait time:  7.8
Turnaround time of Processes:  [['P1', 11], ['P2', 19], ['P3', 3], ['P4', 13], ['P5', 13]]
Wait time of Processes:  [['P1', 8], ['P2', 12], ['P3', 1], ['P4', 8], ['P5', 10]]

Ganttchart(Visualized): P1 OO P2 OO P3 OO P4 OO P5 OO P1 O P2 OOOO P4 OOO P5 O P2 O

Ganttchart: [['P1', 2], ['P2', 2], ['P3', 2], ['P4', 2], ['P5', 2], ['P1', 1], ['P2', 4], ['P4', 3], ['P5', 1], ['P2', 1]]
```