가상 메모리 관리 기법의 하나인 페이지 교체 기법 중 OPT / FIFO / LRU / Second_Chance를 구현하고 과정을 보여주는 시뮬레이터 구현

2021년 11월

김상현

차 례

1장. 프로젝트 동기/목적

가상 메모리 관리기법의 하나인 페이지 교체 기법 중에 OPT / FIFO / LRU / Second_Chance 기법을 구현하고 동작과정을 보여주는 시뮬레이터를 구현해 본다. 이를 통해 페이지 교체 기법을 파악하고 각각 관리기법의 과정을 정확히 이해한다. 그리고 각 과정의 page fault 횟수를 출력하고 각 기법마다의 page fault를 비교한다.

FIFO 구현을 age를 기준으로 했었다가 Second_chance를 구현하면서 FIFO 기법을 circular_queue처럼 구현하면 더 깔끔하게 구현이 가능하다는 점을 파악하여 이를 통해 fifo를 두 번 구현했음을 미리 밝힙니다. (1.FIFO_with_age() , 2. FIFO_circular_queue())

2장. 설계/구현

(1)기본 개념 파악

OPT : 앞으로 가장 오랫동안 참조되지 않을 page를 교체하는 기법이다. page fault를 최소화 할 수 있는 방법으로 증명이 이미 되어있다. (Optimal solution 최적의 방법) 하지만 현실에서는 이후 들어오는 page reference string을 미리 알고 있어야 가능한 방법이기에 실현 불가능한 방법이라고 할 수 있다.

FIFO : 가장 오래된 page를 교체하는 기법이다. page 가 적재한 시간(age)를 기억하고 있어야한다. 이는 자주 사용되는 page가 교체될 가능성이 높다. 즉 page fault가 평균적으로 높다고 볼 수 있다.

LRU : 가장 오랬동안 참조되지 않았던 page를 교체한다. 참조가 될 때마다 시간(age)를 기록해야하는데 이는 곧 overhead로 작용할 수 있다.

Second_chance : FIFO를 기반으로 하지만 각 page는 참조 비트를 가진다. 참조비트는 0으로 초기화 되었다가 해당 page가 참조 되었을 때 해당 페이지에 대한 참조비트를 1로 교체한다. 이후 들어오는 page 에 대해서는 참조비트가 0인 page 들 중에 교체하게 된다.

(2) 설계 아이디어

void OPT(void) {}
: page fault 가 일어날 때 어떻게 처리하는게 제일 큰 페이지 관리기법의 요점이다.

opt 기법은 이후 들어오는 page가 이후 page reference string에서 얼마나 멀리 있는지 (아예 없을 수도 있다.) 파악해야한다.

이때 count_dist(int index, int dest_var) 를 이용해서 index부터 page reference string 끝까지 dest_var이 처음 나오는 거리를 계산해서 return 하는 데 이 값이 제일 큰 page를 선택하게 되는 것이다.

```
}
for(int j = 0;j<frame_num;j++){
        if( max<(temp= count_dist(i,frame[j]))){
                max = temp;
                frame_index = j;
        }

}
frame[frame_index] = prs[i];
```

위의 코드에서 frame_index는 다음 참조될 때까지의 거리가 제일 먼 page의 virtual frame에서의 index가 될 것이다.

void FIFO()

* fifo를 새로운 배열을 선언하고 이에 각 frame의 age를 저장하는 방식으로 먼저 구현하였다. 하지만 이후 Second_chance() 함수를 구현하는 과정에서 다른 방식을 사용한다고 생각하여 circular queue의 개념으로 새로 구현하였다.

- void FIFO_with_age(void) :

FIFO의 개념은 각 page가 들어온 시간(age)를 기록해야한다. 그리고 제일 먼저 들어왔던 page를 교체하는 방식이다.

```
}
for(int j = 0;j<frame_num;j++){
        if( min>frame_input_time_arr[j]){
                min = frame_input_time_arr[j];
                frame_index = j;
        }

}
frame[frame_index] = prs[i];
input_time(frame_index,i);
```

제일 들어온 시간이 적은 (즉 age가 제일 오래된 것과 같다.) page 의 index가 위의 코드에서의 frame_index 가 될 것이다. 이후 input_time(int index, int I) 함수를 이용해서 frame_input_time[] array에 해당 page에 들어온 시간을 기록한다.

- void FIFO_circular_chance(void) :

실제 구현에 있어 age를 직접적으로 새로운 배열에 저장하는 것은 overhead로써 작용한다고 생각했고 이후 단순히 다음 순서를 기억하는 pointer를 저장하면 되지 않을까라는 개념으로 접근했다.

```
                }
        }
        frame[pointer] = prs[i];
        pointer++;
        pointer%=frame_num;
}
```

pointer 변수는 frame_num의 범위 내에서만 움직일 것이다. (만약 frame_num 이 3이라면 0,1,2)

page가 교체되면서 page fault가 일어날 때마다 증가하면서 다음 frame 번호를 가리키게 될 것이다.

위의 age를 사용하여 새로운 배열과 함수를 사용하는 것보다 이방법을 사용하는 것이 더욱 쉽고 overhead 없이 구현할 수 있다.

void LRU(void) {} :

들어온 page reference string에서 참조가 되었던 시간이 가장 멀었던 page를 교체 해야하기 때문에 현재 시간 (i) - 참조된 시간(time)이 제일 큰 page를 index를 찾아야한다.

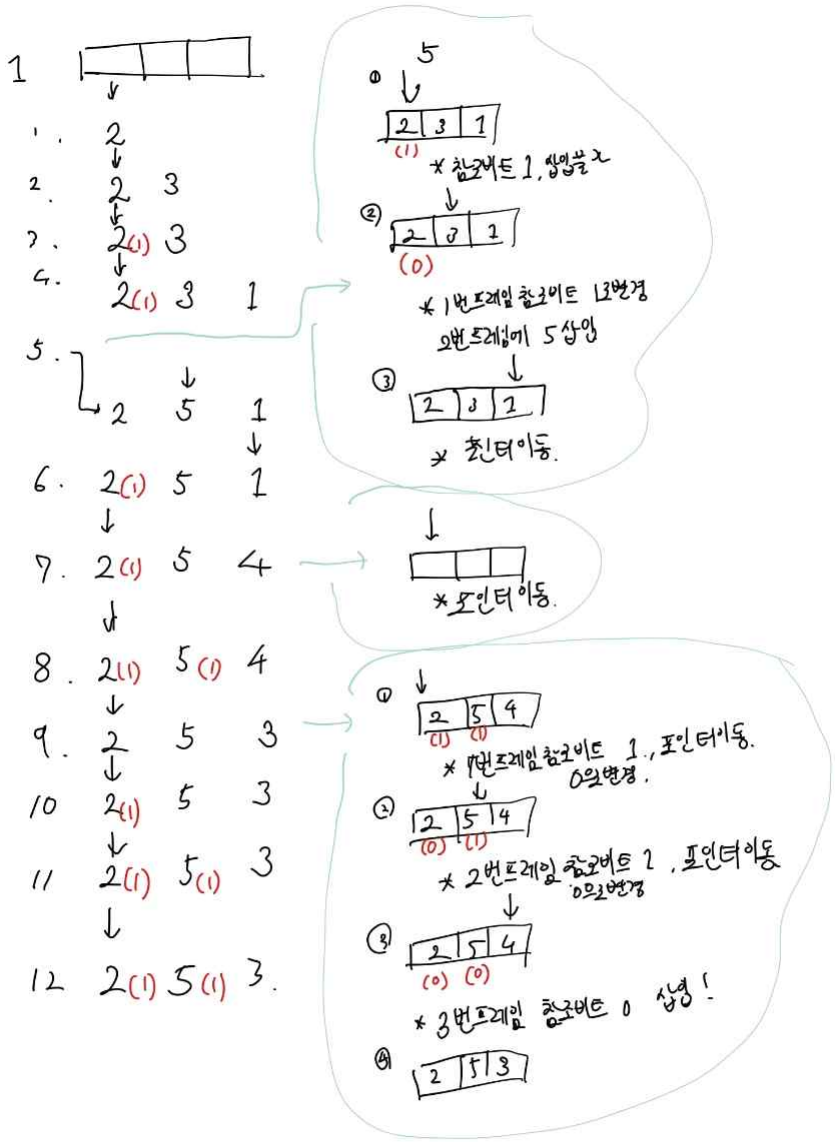
```
            }
        }
        for(int j = 0;j<frame_num;j++){
                if( max< (i-frame_input_time_arr[j])){
                        max =(i- frame_input_time_arr[j]);
                        frame_index = j;
                }

        }
        frame[frame_index] = prs[i];
        input_time(frame_index,i);
```

frame_index 는 현시점에서 제일 참조가 된 시점이 제일 먼 index가 될 것이다. 또한 LRU에서는 참조가 될 때 마다 참조시간을 기록해야하는데 FIFO와 다르게 page fault 가 일어나지 않을 때도 시간을 기록해야한다.

void second_chance(void) {} :

제일 이해하기 어려웠던 기법이니만큼 이해하는데 사용한 노트 파일의 이미지 파일을 첨부한다.


1
1.
2.
3.
4.
5.
6.
7.
8.
9.
10
11
12
5
* 참조비트 1, 삽입할 곳
* 1번프레임 참조비트 1로변경
0번프레임에 5삽입
* 포인터이동.
* 포인터이동.
* 1번프레임참조비트 1., 포인터이동.
0으로변경.
* 2번프레임 참조비트 1 , 포인터이동
0으로변경
* 3번프레임 참조비트 0 삽입!

```
else{ //page fault uppers
        for(int j =0;j<frame_num;j++){
                if(frame[j] == 0){ // nothing in virtual frame,
                        frame[j] = prs[i];// just insert prs in virtual frame;
                        //fst[j]->time = i;
                        goto jump;
                }
        }
        while(1){
                if(fst[pointer].bit == 1 ){ //check if bit is 1;
                        fst[pointer].bit =0; // change ref bit
                        pointer++;
                        pointer%=(frame_num);//pointer should (0,1,2) if frame_num is 3
                }
                else{
                        frame[pointer] = prs[i]; //insert
                        pointer++;
                        pointer%=(frame_num);
                        break;
                }
        }
}
```

while(1) 문 안을 보면 pointer index 의 page의 ref bit가 1 인지 0인지 파악한다 1이라면 0으로 교체하고 poniter에 1을 더하고 0이라면 해당 위치에 page를 삽입하고 pointer를 삽입한다. pointer 변수는 circular queue를 표현하기 위한 변수라고 볼 수 있다.

3장수행결과

input.txt

```
1 3
2 2 3 2 1 5 2 4 5 3 2 5 2
```

input1.txt (35개)

```
1 4
2 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5
```

input2.txt (35개)

```
1 4
2 1 2 3 4 5 5 4 3 2 1 1 2 3 4 5 5 4 3 2 1 1 2 3 4 5 5 4 3 2 1 1 2 3 4 5
```

input3.txt (랜덤값)

```
1 3
2 1 2 3 4 2 4 1 3 4 5 1 2 3 5 1 2 4 3 5 1 2 4 3 5 1 4 5 2 1 1 1 2
```

결과 :

실행시에 입력 파일을 입력받는다.

```
sanghyun@ubuntu:~/os$ ./a.out
input filename : input.txt
```

```
Used_method : OPT
page reference string :   2  3  2  1  5  2  4  5  3  2  5  2

        frame   1       2       3       page fault
time
  1.            2                       F
  2.            2       3               F
  3.            2       3
  4.            2       3       1       F
  5.            2       3       5       F
  6.            2       3       5
  7.            4       3       5       F
  8.            4       3       5
  9.            4       3       5
 10.            2       3       5       F
 11.            2       3       5
 12.            2       3       5
Number of page faults : 6 times
```

OPT

```
Used_method : FIFO_age
page reference string :   2  3  2  1  5  2  4  5  3  2  5  2

        frame   1       2       3       page fault
time
  1.            2                       F
  2.            2       3               F
  3.            2       3
  4.            2       3       1       F
  5.            5       3       1       F
  6.            5       2       1       F
  7.            5       2       4       F
  8.            5       2       4
  9.            3       2       4       F
 10.            3       2       4
 11.            3       5       4       F
 12.            3       5       2       F
Number of page faults : 9 times
```

FIFO (1)

```
Used_method : FIFO_cicular_queue
page reference string :   2  3  2  1  5  2  4  5  3  2  5  2

        frame   1       2       3       page fault
time
  1.            2                       F
  2.            2       3               F
  3.            2       3
  4.            2       3       1       F
  5.            5       3       1       F
  6.            5       2       1       F
  7.            5       2       4       F
  8.            5       2       4
  9.            3       2       4       F
 10.            3       2       4
 11.            3       5       4       F
 12.            3       5       2       F
Number of page faults : 9 times
```

FIFO (2)

```
Used_method : LRU
page reference string :   2  3  2  1  5  2  4  5  3  2  5  2

        frame   1       2       3       page fault
time
  1.            2                       F
  2.            2       3               F
  3.            2       3
  4.            2       3       1       F
  5.            2       5       1       F
  6.            2       5       1
  7.            2       5       4       F
  8.            2       5       4
  9.            3       5       4       F
 10.            3       5       2       F
 11.            3       5       2
 12.            3       5       2
Number of page faults : 7 times
```

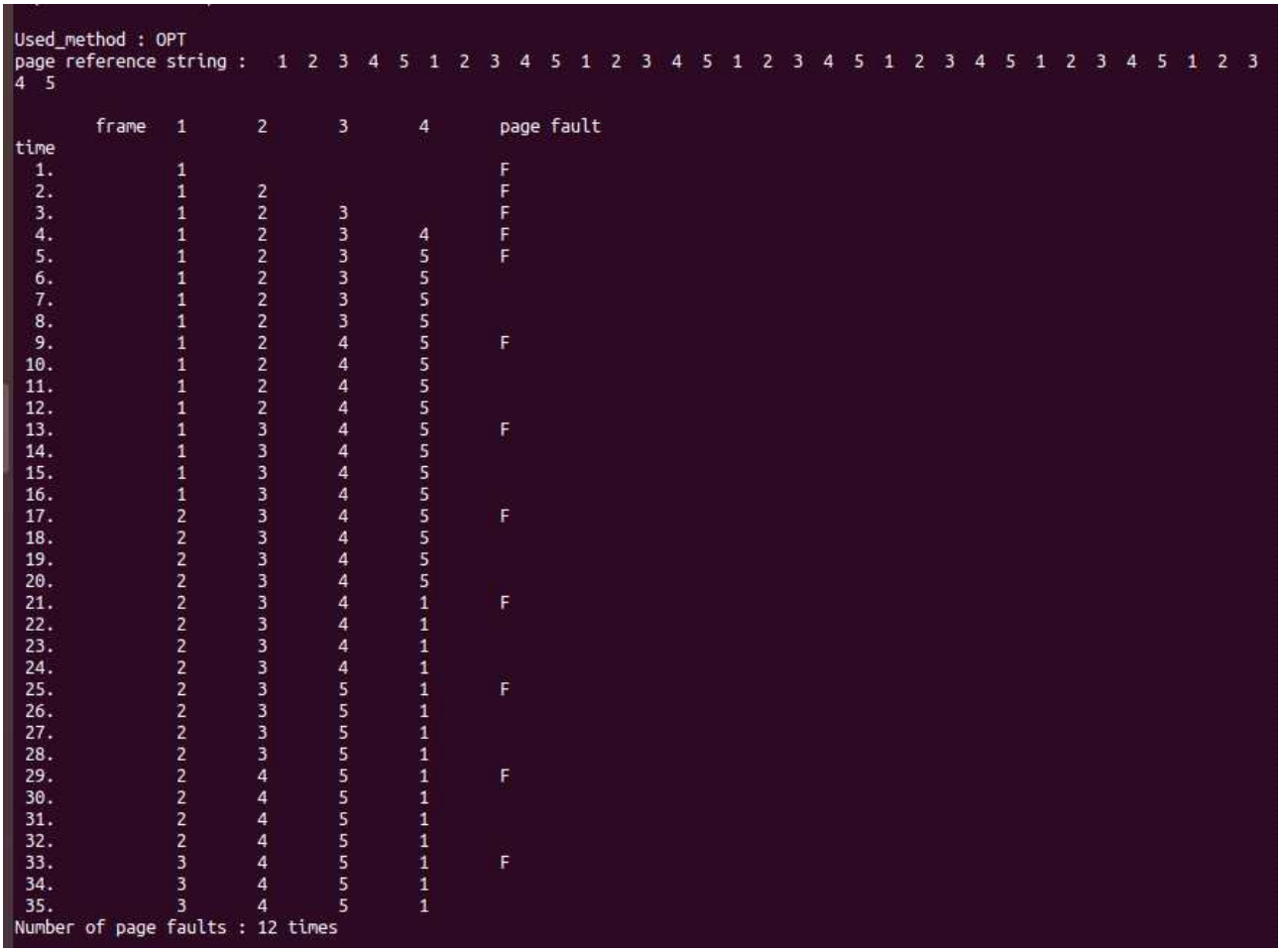
LRU

```
Used_method : Second chance
page reference string :   2  3  2  1  5  2  4  5  3  2  5  2

        frame   1       2       3       page fault
time
  1.            2                       F
  2.            2       3               F
  3.            2       3
  4.            2       3       1       F
  5.            2       5       1       F
  6.            2       5       1
  7.            2       5       4       F
  8.            2       5       4
  9.            2       5       3       F
 10.            2       5       3
 11.            2       5       3
 12.            2       5       3
Number of page faults : 6 times
```

Second_Chance

● input1.txt를 입력했을 때는 OPT 기법을 제외하고는 모두 35번 전부 page fault가 발생했다.



```
Used_method : OPT
page reference string :   1  2  3  4  5  1  2  3  4  5  1  2  3  4  5  1  2  3  4  5  1  2  3  4  5  1  2  3  4  5  1  2  3
4  5

        frame   1       2       3       4       page fault
time
  1.            1                               F
  2.            1       2                       F
  3.            1       2       3               F
  4.            1       2       3       4       F
  5.            1       2       3       5       F
  6.            1       2       3       5
  7.            1       2       3       5
  8.            1       2       3       5
  9.            1       2       4       5       F
 10.            1       2       4       5
 11.            1       2       4       5
 12.            1       2       4       5
 13.            1       3       4       5       F
 14.            1       3       4       5
 15.            1       3       4       5
 16.            1       3       4       5
 17.            2       3       4       5       F
 18.            2       3       4       5
 19.            2       3       4       5
 20.            2       3       4       5
 21.            2       3       4       1       F
 22.            2       3       4       1
 23.            2       3       4       1
 24.            2       3       4       1
 25.            2       3       5       1       F
 26.            2       3       5       1
 27.            2       3       5       1
 28.            2       3       5       1
 29.            2       4       5       1       F
 30.            2       4       5       1
 31.            2       4       5       1
 32.            2       4       5       1
 33.            3       4       5       1       F
 34.            3       4       5       1
 35.            3       4       5       1
Number of page faults : 12 times
```

위 사진은 그 중 OPT에 대한 결과이다. 확인 할 수 있는 건 다음에 들어오는 page reference string에서 제일 참조가 멀리 될 값에 page exchange가 일어나기 때문에 처음 5초 까지는 page fault가 계속 일어나지만 그 이후에는 5번에 한번 씩만 fault가 일어난다.

● input2.txt를 입력하였을 때는 다양한 패턴을 볼 수 있었다.

```
Used_method : OPT
page reference string :   1  2  3  4  5  5  4  3  2  1  1  2  3  4  5  5  4  3  2  1  1  2  3  4  5  5  4  3  2  1  1  2  3
4  5

       frame   1       2       3       4       page fault
time
  1.           1                               F
  2.           1       2                       F
  3.           1       2       3               F
  4.           1       2       3       4       F
  5.           5       2       3       4       F
  6.           5       2       3       4
  7.           5       2       3       4
  8.           5       2       3       4
  9.           5       2       3       4
 10.           1       2       3       4       F
 11.           1       2       3       4
 12.           1       2       3       4
 13.           1       2       3       4
 14.           1       2       3       4
 15.           5       2       3       4       F
 16.           5       2       3       4
 17.           5       2       3       4
 18.           5       2       3       4
 19.           5       2       3       4
 20.           1       2       3       4       F
 21.           1       2       3       4
 22.           1       2       3       4
 23.           1       2       3       4
 24.           1       2       3       4
 25.           5       2       3       4       F
 26.           5       2       3       4
 27.           5       2       3       4
 28.           5       2       3       4
 29.           5       2       3       4
 30.           1       2       3       4       F
 31.           1       2       3       4
 32.           1       2       3       4
 33.           1       2       3       4
 34.           1       2       3       4
 35.           5       2       3       4       F
Number of page faults : 11 times
```

OPT : 11 번

```
Used_method : FIFO_age
page reference string :   1  2  3  4  5  5  4  3  2  1  1  2  3  4  5  5  4  3  2  1  1  2  3  4  5  5  4  3  2  1  1  2  3
4  5

       frame   1       2       3       4       page fault
time
  1.           1                               F
  2.           1       2                       F
  3.           1       2       3               F
  4.           1       2       3       4       F
  5.           5       2       3       4       F
  6.           5       2       3       4
  7.           5       2       3       4
  8.           5       2       3       4
  9.           5       2       3       4
 10.           5       1       3       4       F
 11.           5       1       3       4
 12.           5       1       2       4       F
 13.           5       1       2       3       F
 14.           4       1       2       3       F
 15.           4       5       2       3       F
 16.           4       5       2       3
 17.           4       5       2       3
 18.           4       5       2       3
 19.           4       5       2       3
 20.           4       5       1       3       F
 21.           4       5       1       3
 22.           4       5       1       2       F
 23.           3       5       1       2       F
 24.           3       4       1       2       F
 25.           3       4       5       2       F
 26.           3       4       5       2
 27.           3       4       5       2
 28.           3       4       5       2
 29.           3       4       5       2
 30.           3       4       5       1       F
 31.           3       4       5       1
 32.           2       4       5       1       F
 33.           2       3       5       1       F
 34.           2       3       4       1       F
 35.           2       3       4       5       F
Number of page faults : 20 times
```

FIFO : 20 번

```
Used_method : LRU
page reference string :   1  2  3  4  5  5  4  3  2  1  1  2  3  4  5  5  4  3  2  1  1  2  3  4  5  5  4  3  2  1  1  2  3
4  5

       frame   1       2       3       4       page fault
time
  1.           1                               F
  2.           1       2                       F
  3.           1       2       3               F
  4.           1       2       3       4       F
  5.           5       2       3       4       F
  6.           5       2       3       4
  7.           5       2       3       4
  8.           5       2       3       4
  9.           5       2       3       4
 10.           1       2       3       4       F
 11.           1       2       3       4
 12.           1       2       3       4
 13.           1       2       3       4
 14.           1       2       3       4
 15.           5       2       3       4       F
 16.           5       2       3       4
 17.           5       2       3       4
 18.           5       2       3       4
 19.           5       2       3       4
 20.           1       2       3       4       F
 21.           1       2       3       4
 22.           1       2       3       4
 23.           1       2       3       4
 24.           1       2       3       4
 25.           5       2       3       4       F
 26.           5       2       3       4
 27.           5       2       3       4
 28.           5       2       3       4
 29.           5       2       3       4
 30.           1       2       3       4       F
 31.           1       2       3       4
 32.           1       2       3       4
 33.           1       2       3       4
 34.           1       2       3       4
 35.           5       2       3       4       F
Number of page faults : 11 times
```

LRU : 11번

```
Used_method : Second chance
page reference string :   1  2  3  4  5  5  4  3  2  1  1  2  3  4  5  5  4  3  2  1  1  2  3  4  5  5  4  3  2  1  1  2  3
4  5

       frame   1       2       3       4       page fault
time
  1.           1                               F
  2.           1       2                       F
  3.           1       2       3               F
  4.           1       2       3       4       F
  5.           5       2       3       4       F
  6.           5       2       3       4
  7.           5       2       3       4
  8.           5       2       3       4
  9.           5       2       3       4
 10.           5       1       3       4       F
 11.           5       1       3       4
 12.           5       1       2       4       F
 13.           5       1       2       3       F
 14.           4       1       2       3       F
 15.           4       1       5       3       F
 16.           4       1       5       3
 17.           4       1       5       3
 18.           4       1       5       3
 19.           4       2       5       3       F
 20.           4       2       5       1       F
 21.           4       2       5       1
 22.           4       2       5       1
 23.           3       2       5       1       F
 24.           3       2       4       1       F
 25.           5       2       4       1       F
 26.           5       2       4       1
 27.           5       2       4       1
 28.           5       3       4       1       F
 29.           5       3       4       2       F
 30.           5       1       4       2       F
 31.           5       1       4       2
 32.           5       1       4       2
 33.           5       1       3       2       F
 34.           4       1       3       2       F
 35.           4       1       5       2       F
Number of page faults : 21 times
```

Second_change : 21번

● input3.txt를 입력하였을 때

```
Used_method : OPT
page reference string :   1  2  3  4  2  4  1  3  4  5  1  2  3  5  1  2  4  3  5  1  2  4  3  5  1  4  5  2  1  1  1  2

       frame   1       2       3       page fault
time
  1.           1                       F
  2.           1       2               F
  3.           1       2       3       F
  4.           1       2       4       F
  5.           1       2       4
  6.           1       2       4
  7.           1       2       4
  8.           1       3       4       F
  9.           1       3       4
 10.           1       3       5       F
 11.           1       3       5
 12.           2       3       5       F
 13.           2       3       5
 14.           2       3       5
 15.           2       3       1       F
 16.           2       3       1
 17.           4       3       1       F
 18.           4       3       1
 19.           4       5       1       F
 20.           4       5       1
 21.           4       5       2       F
 22.           4       5       2
 23.           4       5       3       F
 24.           4       5       3
 25.           4       5       1       F
 26.           4       5       1
 27.           4       5       1
 28.           2       5       1       F
 29.           2       5       1
 30.           2       5       1
 31.           2       5       1
 32.           2       5       1
Number of page faults : 14 times

Used_method : FIFO_age
```

OPT : 14번

```
Used_method : FIFO_age
page reference string :   1  2  3  4  2  4  1  3  4  5  1  2  3  5  1  2  4  3  5  1  2  4  3  5  1  4  5  2  1  1  1  2

       frame   1       2       3       page fault
time
  1.           1                       F
  2.           1       2               F
  3.           1       2       3       F
  4.           4       2       3       F
  5.           4       2       3
  6.           4       2       3
  7.           4       1       3       F
  8.           4       1       3
  9.           4       1       3
 10.           4       1       5       F
 11.           4       1       5
 12.           2       1       5       F
 13.           2       3       5       F
 14.           2       3       5
 15.           2       3       1       F
 16.           2       3       1
 17.           4       3       1       F
 18.           4       3       1
 19.           4       5       1       F
 20.           4       5       1
 21.           4       5       2       F
 22.           4       5       2
 23.           3       5       2       F
 24.           3       5       2
 25.           3       1       2       F
 26.           3       1       4       F
 27.           5       1       4       F
 28.           5       2       4       F
 29.           5       2       1       F
 30.           5       2       1
 31.           5       2       1
 32.           5       2       1
Number of page faults : 18 times

Used_method : FIFO_circular_queue
```

FIFO :18번

```
Used_method : LRU
page reference string :   1  2  3  4  2  4  1  3  4  5  1  2  3  5  1  2  4  3  5  1  2  4  3  5  1  4  5  2  1  1  1  2

       frame   1       2       3       page fault
time
  1.           1                       F
  2.           1       2               F
  3.           1       2       3       F
  4.           4       2       3       F
  5.           4       2       3
  6.           4       2       3
  7.           4       2       1       F
  8.           4       3       1       F
  9.           4       3       1
 10.           4       3       5       F
 11.           4       1       5       F
 12.           2       1       5       F
 13.           2       1       3       F
 14.           2       5       3       F
 15.           1       5       3       F
 16.           1       5       2       F
 17.           1       4       2       F
 18.           3       4       2       F
 19.           3       4       5       F
 20.           3       1       5       F
 21.           2       1       5       F
 22.           2       1       4       F
 23.           2       3       4       F
 24.           5       3       4       F
 25.           5       3       1       F
 26.           5       4       1       F
 27.           5       4       1
 28.           5       4       2       F
 29.           5       1       2       F
 30.           5       1       2
 31.           5       1       2
 32.           5       1       2
Number of page faults : 25 times
```

LRU : 25번

```
Used_method : Second chance
page reference string :   1  2  3  4  2  4  1  3  4  5  1  2  3  5  1  2  4  3  5  1  2  4  3  5  1  4  5  2  1  1  1  2

       frame   1       2       3       page fault
time
  1.           1                       F
  2.           1       2               F
  3.           1       2       3       F
  4.           4       2       3       F
  5.           4       2       3
  6.           4       2       3
  7.           4       2       1       F
  8.           4       3       1       F
  9.           4       3       1
 10.           4       3       5       F
 11.           4       1       5       F
 12.           4       1       2       F
 13.           3       1       2       F
 14.           3       5       2       F
 15.           3       5       1       F
 16.           2       5       1       F
 17.           2       4       1       F
 18.           2       4       3       F
 19.           5       4       3       F
 20.           5       1       3       F
 21.           5       1       2       F
 22.           4       1       2       F
 23.           4       3       2       F
 24.           4       3       5       F
 25.           1       3       5       F
 26.           1       4       5       F
 27.           1       4       5
 28.           2       4       5       F
 29.           2       1       5       F
 30.           2       1       5
 31.           2       1       5
 32.           2       1       5
Number of page faults : 25 times
```

Second_chance : 25번

4장 결론 및 보충 할 점

결과를 비교해보았을 때 OPT 기법이 압도적으로 page fault가 적다. 하지만 언급했던 거처럼 현실적으로 이후 들어올 page를 예측하는 것은 불가능에 가깝다.

다른 3개기법을 비교해 보자.

FIFO의 경우 평균적으로 비슷한 page fault 횟수를 나타내고 패턴의 영향을 제일 덜 받았다.
LRU의 경우 반복적으로 주어지는 page reference string에서 강점이 나타났다. (input2.txt 참고)
Second_chance 의 경우 page fault에 있어서 특이점을 찾지 못했다.

현재 제일 많이 사용하는 페이지 관리기법은 LRU라고 한다. 실생활에서 우리는 자주 사용하는 프로그램이 있고 자주 사용하지 않는 프로그램이 명확히 나누어져 있다. 그래서 같은 page 가 반복적으로 나타나는 현실의 경우에서는 LRU 관리 기법이 제일 효과적일 것이라고 예측한다.

page reference string의 패턴에 따라 효과적인 기법이 각각 다르다고 볼 수 있다. 하지만 역시 현실에서는 단순히 page fault 가 적다고 효율적인 관리기법이라고 하기는 어렵다. overhead도 생각해야하며 다른 영향을 미치는 factor 도 존재할 것이다. 하여 제일 효율적/ 좋은 관리기법은 콕 집어 말할 수 없을 것 같다.

개인 적인 생각을 추가하자면 opt 관리 기법이 완벽히 불가능하다고 생각하지는 않다. 사용자의 CPU 이용 패턴에는 특정 패턴들이 존재할 것이고 이를 저장해 놓거나 빅데이터의 개념으로 접근 한다면 특정 상황에서는 OPT 관리기법을 사용할 수 있는 시간이 존재할 것이라고 생각한다.