REPORT

디스크 입출력 스케줄러의 구현

과 목 명 :	운영체제설계	
담당교수 :	곽종욱 교수님	
학 과 명 :	컴퓨터공학과	통계학과
학 번 :	21411893	21410785
성 명 :	장대건	황희
제출일자 :	2018.06.18	

목 차

1. 과제 개요

1.1 디스크 입출력 스케줄러의 구현 목표

수업시간에 학습한 디스크 입출력 스케줄러를 구현하고, 이를 바탕으로 각각의 스케줄링 알고리즘의 동작 결과를 분석하고 이해한다.

1.2 구현 알고리즘 및 사용 언어

-FCFS(샘플 소스로 제공. 구현 언어 : C#)

-SSTF, SCAN, C-SCAN, LOOK, C-LOOK 등 학습한 스케줄링 정책 중 3개 이상을 구현

-FCFS 이외에 총 3가지 정책이 기본적으로 구현되어야 함

-사용언어 : C, C++, C#, JAVA, PYTHON, PERL 등 자유

1.3 구현 내용

1.입력
-디스크 헤더의 초기 위치, 입력 요청 트랙 번호의 큐잉 순서 등
-그 외, 해당 스케줄링 알고리즘이 필요로 하는 고유 입력 요소

2.출력
-총 이동 거리

-전체 실행시간 : seek time + average rotational latency

-그 외, 해당 스케줄링 알고리즘의 고유 출력 요소

-시각적으로 스케줄링 되는 순서를 제시할 것

-스케줄링 결과 분석에 있어서, 결과 그래프 제시 및 해당 결과에 대 한 분석을 기술

2. 구현 스케줄러 종류 및 설명

1. SCAN scheduling

SCAN 알고리즘에서는 디스크 암이 디스크의 한 끝에서 시작하여 다른 끝으로 이동하며 가는 길에 있는 모든 요청을 처리 한다. 다른 한쪽 끝에 도달하면 역 방향으로 이동하면서 오는 길에 있는 요청을 처리 한다. 엘리베이터처럼 왕복하며 처리하므로 엘리베이터(elevator algorithm)이라고도 부른다.

2. C-SCAN scheduling

C-SCAN 스케줄링은 각 요청에 걸리는 시간을 좀 더 균등하게 하기 위한 SCAN의 변형이다. SCAN과 같이 C-SCAN은 한쪽 방향으로 헤드를 이동해 가면서 요청을 처리하지만 한쪽 끝에 다다르면 반대 방향으로 헤드를 이동하며 처리하는 것이 아니라 처음 시작했던 자리로 다시 되돌아가서 서비스를 시작한다. C-SCAN 스케줄링 알고리즘은 실린더들을 마지막 실린더가 처음 실린더와 맞닿은 원형 리스트로 간주 한다.

3. LOOK scheduling

SCAN scheduling은 진행방향의 요청을 모두 서비스 하면 진행 방향의 끝까지 간 후에 방향을 바꾸지만 LOOK scheduling은 SCAN 기법과 비슷하지만 진행방향에 더 이상의 요청이 없으면 역으로 스캔하는 scheduling이다. (이동방향의 끝까지 가지 않음)

4. C-LOOK scheduling

C-LOOK scheduling은 항상 바깥쪽에서 안쪽으로 움직이며 서비스를 하고 안쪽 끝까지 모든 요청을 서비스 했을 경우 다시 바깥쪽부터 탐색하는 기법이다. C-LOOK scheduling은 C-SCAN scheduling 기법과 비슷하지만 진행방향의 끝까지 가지 않고 마지막 요청 받은 곳까지만 서비스 하고 방향을 바꾸는 기법이다.

3. 출력결과 및 분석

3.1 SCAN

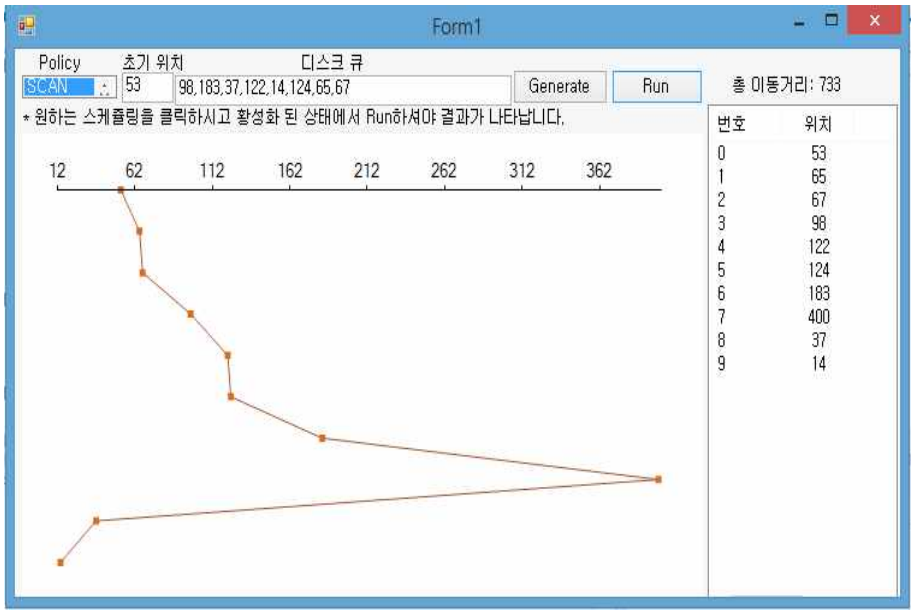

그림 1 SCAN Scheduling 첫번째 시뮬레이션

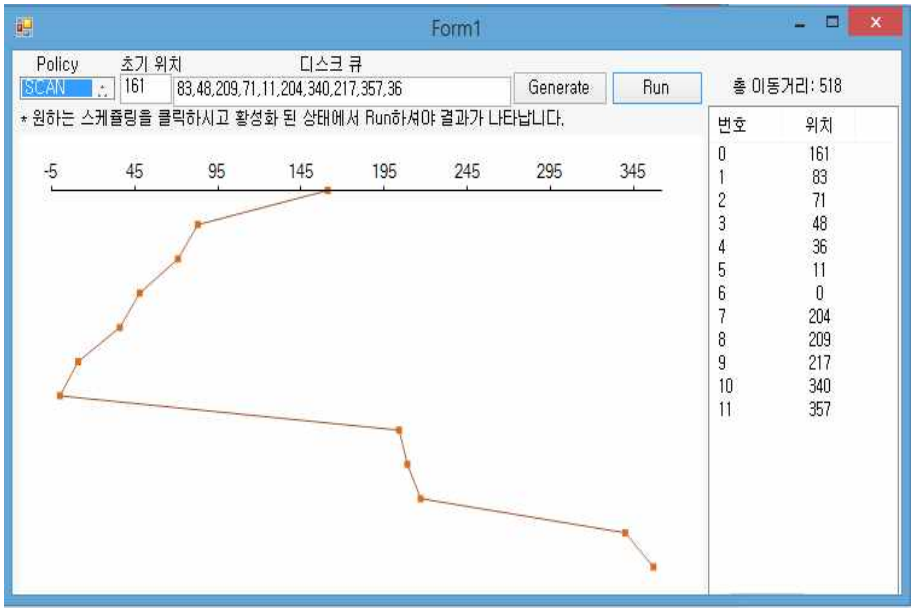

그림 2 SCAN Scheduling 두번째 시뮬레이션

3.2 C-SCAN

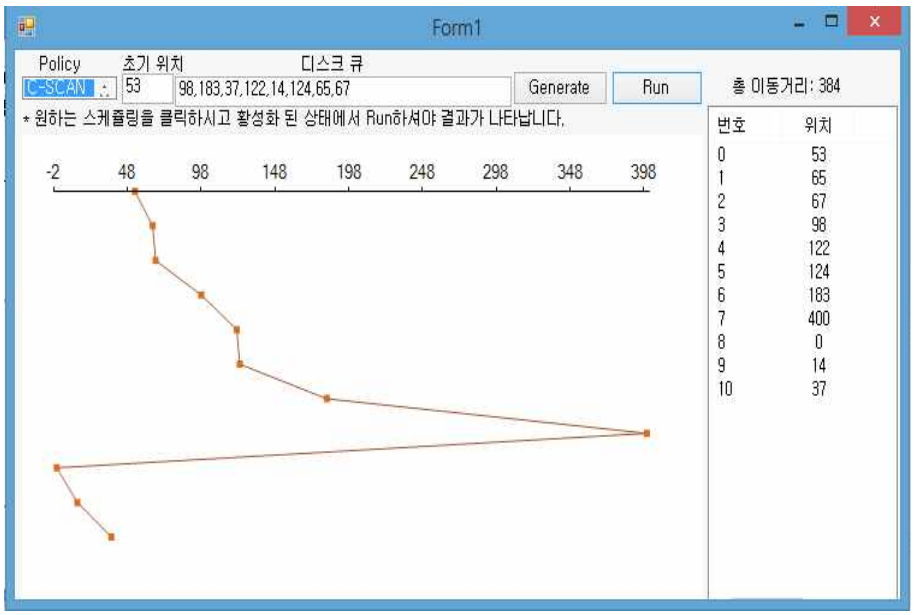

그림 3 C-SCAN Scheduling 첫번째 시뮬레이션

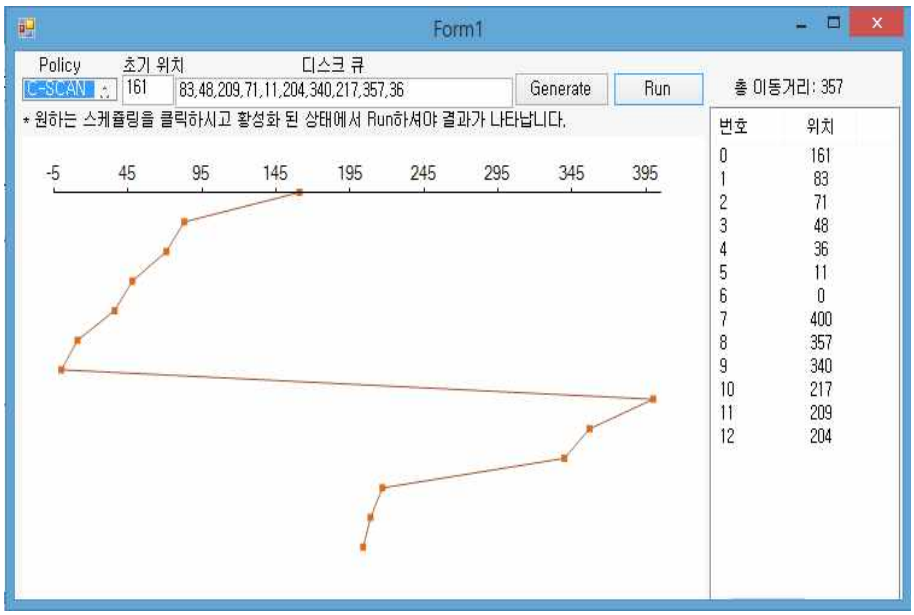

그림 4 C-SCAN Scheduling 두번째 시뮬레이션

3.3 LOOK

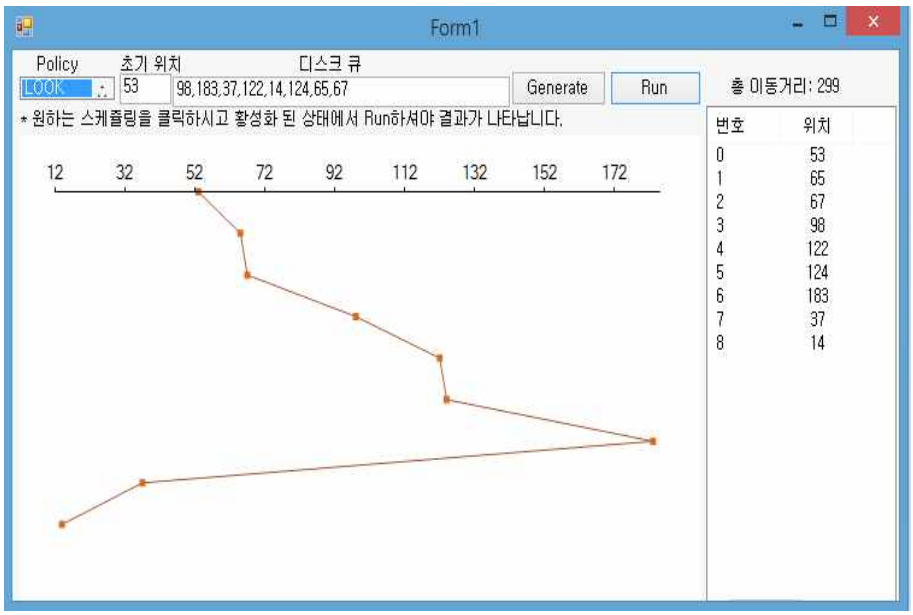

그림 5 LOOK Scheduling 첫번째 시뮬레이션

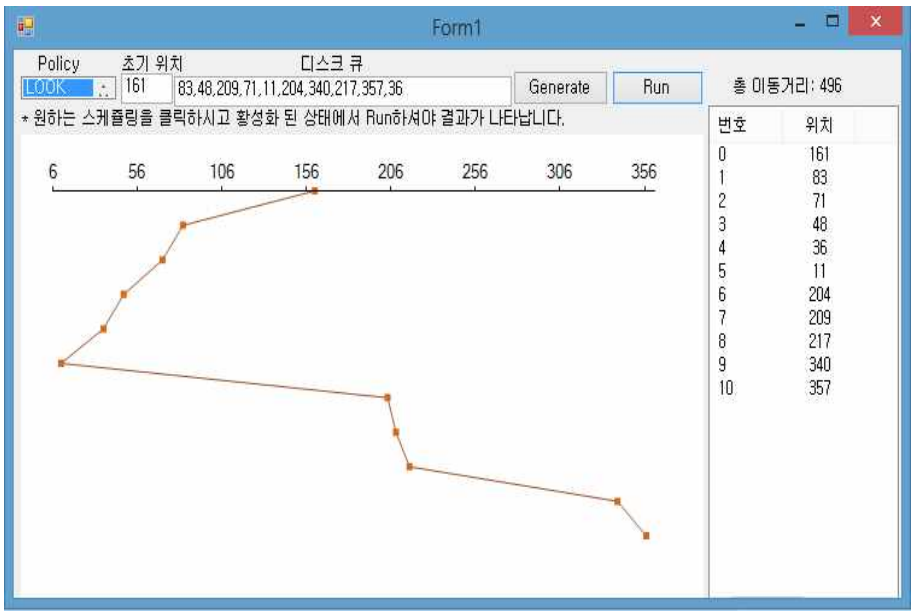

그림 6 LOOK Scheduling 두번째 시뮬레이션

3.4 C-LOOK

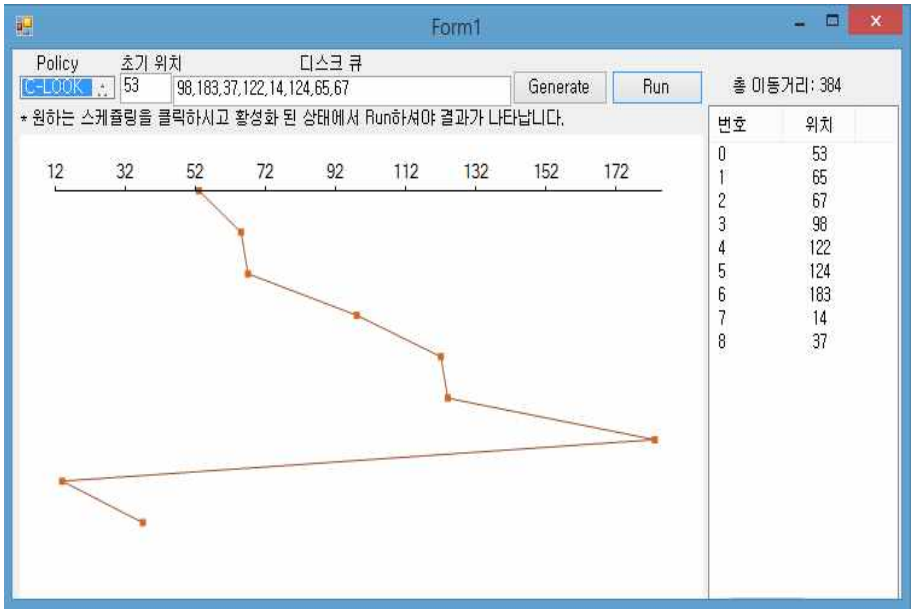

그림 7 C-LOOK Scheduling 첫번째 시뮬레이션

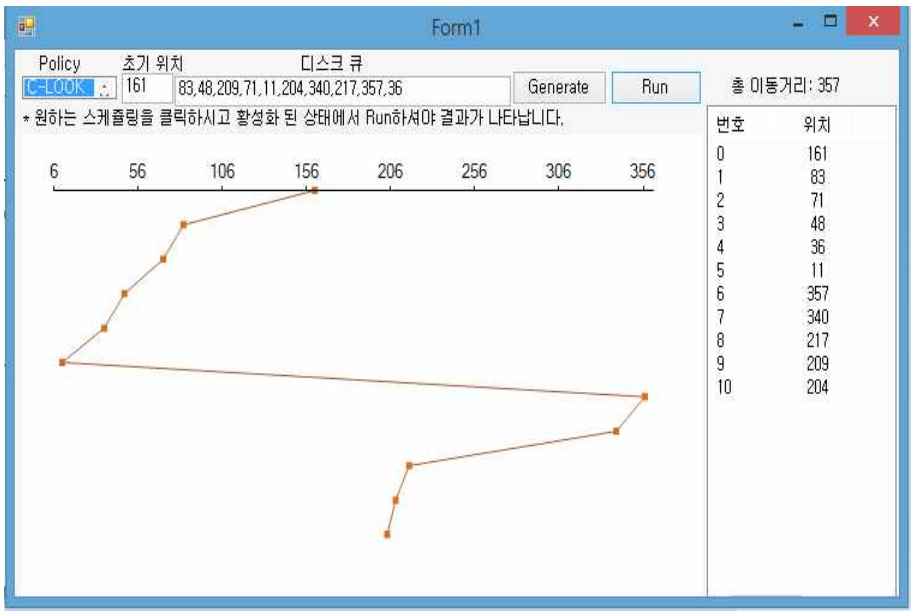

그림 8 C-LOOK Scheduling 두번째 시뮬레이션

3.5 scheduling algorithm 분석 표

표 1 Scheduling algorithm 반복시뮬레이션 결과

반복횟수	scheduling algorithm에 따른 실린더 이동거리				
	FCFS	SCAN	C-SCAN	LOOK	C-LOOK
1회차	640	733	384	299	384
2회차	1906	381	381	352	352
3회차	1846	370	370	356	356
4회차	1383	334	334	229	299
5회차	1973	736	398	676	398
6회차	1506	518	395	484	395
7회차	1282	666	312	644	312
8회차	1571	351	351	333	333
9회차	1878	628	387	590	387
10회차	1500	512	342	492	342
합계	15485	5229	3654	4455	3558
평균	1548.5	522.9	365.4	445.5	355.8
표준편차	378.62	151.65	27.58	146.56	33.10

우선, 각 스케쥴링 기법에 따라 실린더 이동거리에 차이가 있는지 확인하기 위해 두가지 가정을 세웠다.

1. 각 회차별 Head와 Disk queue는 무작위 추출한 수를 사용한다.
2. 각 회차에서 같은 Head와 Disk queue를 가지고 스케쥴링 기법만 다르게하여 실린더 이동거리를 잰다.

결과적으로 C-SCAN과 C-LOOK스케쥴링 기법이 거의 비슷한 평균을 보였다. 아주 미미하지만 평균은 C-LOOK스케쥴링이 더 낮다. 하지만 표준편차는 C-SCAN이 C-LOOK보다 약 6정도 작은 것을 볼 수 있다. 표본이 작기 때문에 단정 지을수는 없지만, 평균으로 보았을 때 C-SCAN과 C-LOOK은 차이가 거의 없다고 볼 수 있고, 표준편차가 C-SCAN이 더 낮은 것을 보아 시뮬레이션 반복시에도 C-SCAN의 반응편차가 더 적을 것임을 알 수 있다. 따라서 만약 4개의 스케쥴러 중 선택해야 한다면 C-SCAN방식을 선택 할 것이다.

1순위. 반응속도의 평균, 2순위. 반응속도의 편차로 스케쥴러의 순위를 매기자면 (C-SCAN>C-LOOK>LOOK>SCAN>>>>>FCFS)순이다.

4. Conclusion

저희는 이번 디스크 입출력 스케줄링(SCAN, C-SCAN, LOOK, C-LOOK) 구현을 통해 좀 더 스케줄링 기법에 대해 자세히 알 수 있었습니다. 각 스케줄링의 종류와 처리하는 방법, 장점과 단점에 대해 이해를 할 수 있었고 각 스케줄링의 차이에 대해 알 수 있었습니다.

이번 과제를 하면서 조원과의 협력을 통해 구현을 하여 좋은 결과를 얻을 수 있었습니다. 또한 나머지 scheduling들을 구현 하지 못한 점에 아쉬움을 느꼈습니다.

마지막으로 과제와 수업을 통해 OS에 대하여 많은 점을 알 수 있어서 매우 의미 있는 시간이 되었습니다.