CFS 구현

정제원

1. 과제 내용

리눅스에 구현되어 있는 CFS 알고리즘과 비슷하게 프로세스 스케줄링을 하도록 시뮬레이션을 하는 기능을 만들어 보았다. 본 프로그램에서는 주어진 Time Slice 만큼 시간이 주어진다. 그리고 입력 받은 인자값들은 순서대로 -2~2 사이의 나이스벨류를 가진다. 그렇게 나이스벨류를 가지고 vruntime을 설정해 준다. 그 시간동안 각각의 Ready Queue에 들어있는 프로세스들이 vruntime을 기준으로 가장 vruntime이 적은 순으로 나와서 app을 실행을 한다. 실행이 되고 나서는 다시 Ready Queue로 들어가 Ready 상태로 만들어 준다. 그렇게 Time Slice가 끝날 때까지 위 작업을 반복한다.

2. 프로그램 구조와 설계

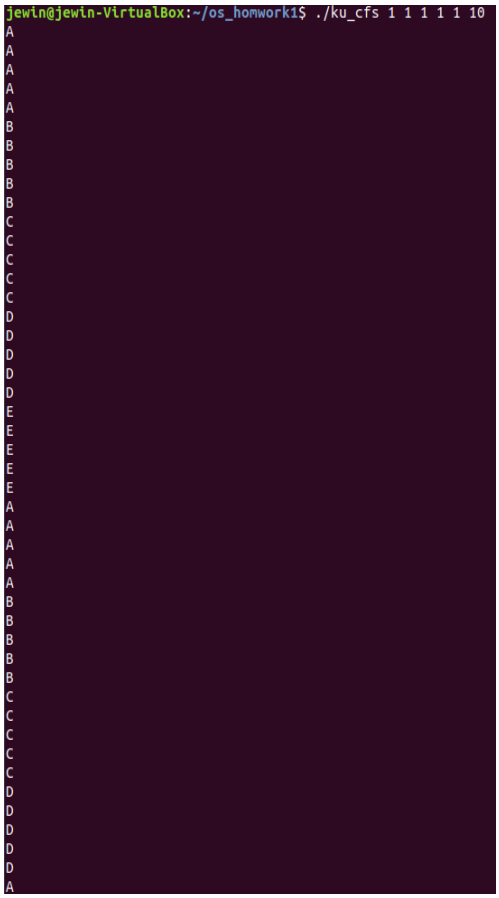
cfs를 실행할 때 인자를 총 6개 받는다. 첫 번째 인자는 나이스벨류가 -2인 프로세스의 개수, 두 번째 인자는 나이스벨류가 -1인 프로세스의 개수, 세 번째 인자는 나이스벨류가 0인 프로세스의 개수, 네 번째 인자는 나이스벨류가 1인 프로세스의 개수, 다섯번째 인자는 나이스벨류가 2인 프로세스의 개수, 여섯번째 인자는 timeslice이다. 결국 첫번째부터 5번째 인자를 합친 것이 process의 총 개수이다. n_init()과 n2_init()으로 노드 리스트를 초기화 시키고 index라는 변수를 가지고 프로세스마다 이름을 넣어주기 위해 arg배열

의 0번째에 아스키코드를 이용해 'A' +index를 해 주어 생기는 개수마다 알파벳이 인자로 들어가게 해준다. ./app의 인자로 들어가는 부분이다. 이 때 생기는 노드의 data정보는 data(vruntime), niceval(nice value), pid(pid정보)를 저장해준다. 자식 프로세스는 execl을 하고 부모 프로세스는 자식의 pid와 vruntime, nicevalue를 노드에 넣어준다. 그렇게 프로세스 개수만큼 app을 실행시켜준다. 처음 vruntime은 모두 0.0이기 때문에 초기 vruntime은 모두 0.0으로 설정해준다. Insert할 때 오름차순으로 정렬을 해주기 때문에 따로 정렬을 해줄 필요가 없다. 그렇게 모든 프로세스를 execl시켜주고 list에 insert해주고 list의 가장 첫 노드를 실행시켜주기 위해 first_node()의 pid를 가져와 SIGCONT를 해준다. First_node()의 함수는 기존의 list에 있는 노드 중 가장 앞에 있는 노드 즉, vruntime이 가장 적은 노드를 list에 빼서 실행을 구분해주는 list2에 insert2()를 이용해 넣어준다. 그렇게 StartTimer()를 이용해 timer를 돌린다. timer에서는 sigaction을 이용해 알람을 보내주고 timer_handler()를 이용해 기존에 실행중인 process를 SIGSTOP으로 멈춰주고, 기존의 실행 중이었던 프로세스의 vruntime을 리셋 해주고, 다시 list에 오름차순 삽입 정렬해준다. 그렇게 삽입을 해주고는 list2에 있는 노드를 지워주고 다시 first_node()를 이용해 list에 있는 vruntime이 가장 작은 것을 SIGCONT 해준다. 그렇게 Time Slice가 끝날 때 까지 돌려주고 Time Slice가 0 이면 기존에 할당해줬던 list, list2의 메모리를 free해준다. 이로서 프로그램이 끝난다.

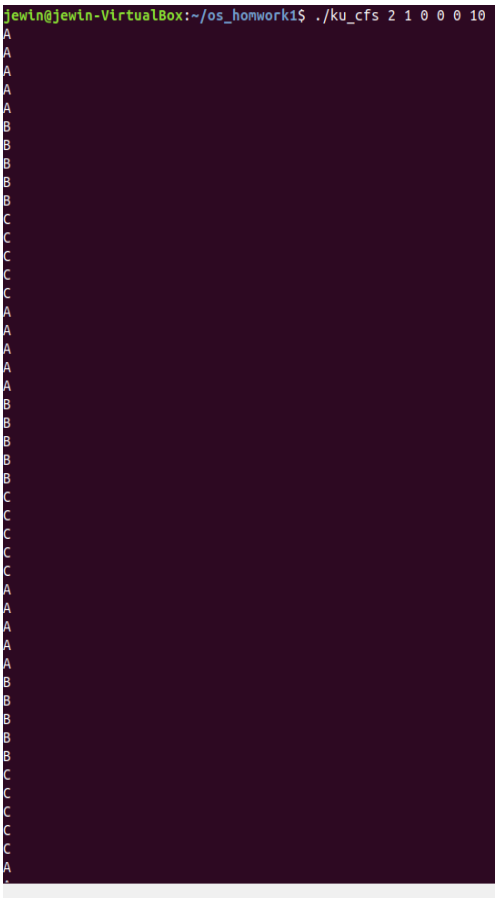
3. 결과물

- **실행 결과물**

Ex1) ./cfs 1 1 1 1 1 10



Ex2) ./cfs 2 1 0 0 0 10



(출력된 화면이 너무 길어 다 올리기에는 무리가 있어 어떤식으로 실행이 되었는지만 보이고 있음)

EX 1번 같은 경우는 process가 총 5개이므로 A, B, C, D, E가 찍히는 것을 볼 수 있고, EX 2번 같은 경우는 프로세스가 3개라서 A, B, C만 찍히는 것을 알 수있다. 그리고 초기에는 무조건 vruntime이 0이기 때문에 순서대로 찍히고 그 다음부터는 순서대로 vruntime의 순서대로 찍히는 것을 알 수 있다.

4. Description for important functions

n_init()	Functionality	Ready Queue인 노드를 초기화해준다.
	Parameters	X
	Return Value	X

n2_init()	Functionality	실행시켜야할 노드의 리스트를 초기화해준다.
	Parameters	X
	Return Value	X

ascending_insert	Functionality	ReadyQueue에 vruntime을 기준으로 오름차순 삽입 정렬을 해준다. 따라서 가장 앞 노드가 vruntime이 가장 짧은 프로세스이다.
	Parameters	(pid_t pid, double vruntime, double nv) Pid : 노드에 pid를 넣어준다. Vruntime : vruntime을 넣어준다. Nv : nice value 를 넣어준다.
	Return Value	X

insert2	Functionality	Ascending_insert시 나온 노드를 임시 저장해주는 list2에 넣어주는 함수
	Parameters	(NODE* n) n을 list2에 넣어준다.
	Return Value	X

del2	Functionality	임시 저장소인 list2에 있는 노드를 삭제시켜준다.
	Parameters	(NODE* n) n을 list2에서 빼준다.

	Return Value	X

first_node	Functionality	list에서 가장 vruntime이 낮은 즉 head 쪽에 있는 노드를 리턴해주고 list에서 삭제해준다.
	Parameters	X
	Return Value	vruntime이 가작 낮은 노드 리턴

timer_handler	Functionality	실행중인 process를 중지시키고, vruntime을 reset해준다. Reset 해준 노드를 다시 list에 오름차순 삽입 정렬을 해준다.
	Parameters	X
	Return Value	X

StartTimer	Functionality	Sigaction을 통해 1초마다 timer_handler를 불러준다.
	Parameters	X
	Return Value	X