hw4

writing part

김신후. 21900136@handong.ac.kr

a. Describe any difficulties you experienced while doing this assignment.

Brute force, Greedy, Dynamic Programming, Branch and bound 중에 가장 어려웠던 알고리즘은 당연 Branch and bound 였습니다. Item 을 sorting하는 것은 algorithm library를 사용하였고, iteration마다 조건에 맞는 node들을 Queue 데이터 구조에 담으면서, max_benefit을 queue가 비워질 때까지 업데이트하는 그림은 수업시간에 배워서 알고 있었지만, 아이탬의 개수 (n)이 50만 되어도 소요시간이 5분이 넘어갔습니다. 왜 그렇게 오래걸리는지 분석을 해 보았습니다. 첫번째로, 쓸데없이 queue에 너무 많은 node들이 들어갔습니다. 그리고, max_benefit 의 값이 최대한 빨리 올려 bound가 상대적으로 작은 node들은 queue에 들어가지 못하게 끔 해야 할텐데, BFS 매너로 node들을 비교하다 보니 이런식으로 가면 worst time complexity가 O(2^n)이라는 최악의 알고리즘이 될 수 있다고 생각했습니다. 그래서 queue에서 FIFO로 node를 꺼내 비교하는 것이 아니라, priority queue 구조를 사용해 bound가 큰 node부터 꺼내 비교했습니다. 이것이 약간 DFS 매너인 것 같습니다. 따라서, iteration동안 max_benefit 을 최대한 빨리 올릴 수 있었고, tree 구조를 생각했을 때, level이 낮을 때부터 많은 node들을 제외시킬 수 있었습니다 . 그 결과 n이 10000개인 경우에도 17초밖에 걸리지 않는 알고리즘을 구현할 수 있었습니다.

알고리즘의 초안은 금방 짰지만, 디버깅하는 과정이 너무 오래걸리고 힘들었습니다. 한 버그는 오류메세지도 없이 프로세스가 종료되었는데, break point를 찍으면서 확인해보니, greedy alogoritm이나, branch and bound에서 value / weight ratio를 구하는 과정에서 weight이 0인 경우가 있었습니다. 0을 나누는 연산을 했기때문에, process가 종료 되버리는 것이 었습니다. 이처럼 버그의 사이즈가 작았고 제 IDE는 오류메세지를 띄워주지 않았기 때문에 더욱 찾기 힘들었습니다.

b. What are the notable differences between your code (at least the one you had in you mind) and the one you referenced on the internet?

제가 reference number 3번은 branch and bound 알고리즘입니다. 해당 알고리즘은 노드들을 queue에서 FIFO policy로 처리를 했습니다. 하지만 저의 알고리즘음 bound 별로 node에 priority를 주어 priority queue policy로 처리를 했습니다. 그 결과 FIFO policy일 때는 worst time complexity가 O(2^n)이지만, priority queue policy일 땐 O(nLg(n)) 이었습니다.

c. Describe what you had learned from this homework.

지금 OS과목에서 process간에 time-sharing이 이루어지고 있는 환경에서 scheduler가 어떤 process를 다음에 실행시킬지 결정하는 것이 전반적인 퍼포먼스에 많은 영향을 미친다고 배웠습니다. 이번에 알고리즘 과제를 하면서 그 스케줄러의 중요성과 영향을 간접적으로 느꼈습니다. Branch and Bound에 대해서 인터넷 서칭으로 조사한 결과, bound를 계산하는 법, 한 node를 queue에 넣는 조건 등은 제가 생각했던 것과 많이 비슷했습니다. 하지만, queue에서 어떤 원소를 먼저 꺼내는지 생각하는 것이 성능에 굉장히 큰 영향을 미쳤습니다. scheduler의 중요성을 배운 것, 앞으로 scheduler를 어떤 policy를 적용할지 고민할 수 있는 힘이 생긴점이 이번 과제를 통해 가장 크게 배운 것 입니다.

이번 중간고사에서 3점을 깍였는데, 그 중 1점이 knapsack 문제였습니다. knapsack 문제를 푸는 알고리즘 중에서도, 특히 branch and bound에 대해서는 자신감이 없었습니다. 하지만, 이번 과제에서 각각의 알고리즘을 직접 c++언어로 구현을 해보고, 테스트를 하는 과정을 통해, 내가 다른 사람에게 각각의 알고리즘을 설명을 해 줄 수 있겠다는 자신감이 생겼습니다.

마지막으로, priority queue를 다루면서 queue의 요소가 구조체일 때는 구조체 정의 안에 operator 매소드를 만들어 주어야 한다는 언어적인 스킬을 배울 수 있었습니다.

Number of Items	Processing time in milliseconds / Maximum benefit value
11	0 / 2575
21	7 / 3716
31	8244 / 6194

Number of Items	Processing time in milliseconds / Maximum benefit value		
	Greedy	D. P.	B. & B.
10	0 / 2009	1 / 2005	1 / 2005
100	1 / 20132	2 / 20128	1 / 20128
1000	1 / 199855	160 / 199854	4 / 199854
10000	3 / 2023405	19720 / 2023405	318 / 2023405