Best-First Search with branch-and-bound pruning example: 0-1 Knapsack

The same problem from above (with W=9)

i	p_i	w_i	p_i/w_i
1	$10	2	5
2	$15	5	3
3	$10	4	2.5
4	$12	5	2.4
5	$6	3	2
6	$4	4	1

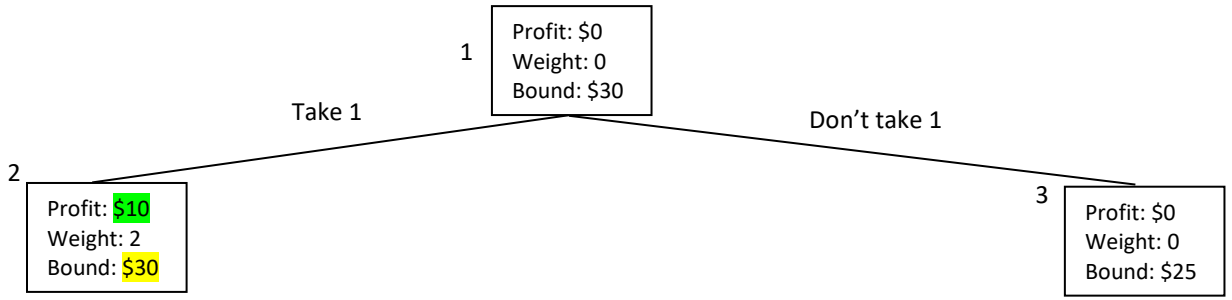
		Take?	Profit	weight	bound	Max profit	Action	Priority Queue
1	Root node		0	0	30	0	Insert 1	1 (30)
2	Remove 1: 1st child	1	10	2	30	10	Insert 2	2 (30)
3	2nd child of 1	not 1	0	0	25	10	Insert 3	2 (30), 3 (25)
4	Remove 2: 1st child	2	25	7	30	25	Insert 4	3 (25), 4 (30)
5	2nd child of 2	not 2	10	2	27.2	25	Insert 5	3 (25), 5 (27.2), 4 (30)
6	Remove 4: 1st child	3	35	11	0	25		3 (25), 5 (27.2)
7	2nd child of 4	not 3	25	7	29.8	25	Insert 7	3 (25), 5 (27.2), 7 (29.8)
8	Remove 7: 1st child	4	37	12	0	25		3 (25), 5 (27.2)
9	2nd child of 7	Not 4	25	7	29	25	Insert 9	3 (25), 5 (27.2), 9 (29)
10	Remove 9: 1st child	5	31	10	0	25		3 (25), 5 (27.2)
11	2nd child of 9	Not 5	25	7	27	25	Insert 11	3 (25), 11 (27), 5 (27.2),
12	Remove 5: 1st child	3	20	6	27.2	25	Insert 12	3 (25), 11 (27), 12 (27.2)
13	2nd child of 5	Not 3	10	2	26	25	Insert 13	3 (25), 13 (26), 11 (27)
14	Remove 12: 1st child	4	32	11	0	25		3 (25), 13 (26), 11 (27)
15	2nd child of 12	Not 4	20	6	26	25	Insert 15	3 (25), 13 (26), 15 (26), 11 (27)
16	Remove 11: 1st child	6	29	11	0	26		3 (25), 13 (26), 15 (26)
17	2nd child of 11	Not 6	25	7	25	26		3 (25), 13 (26)
18	Remove 15: 1st child	5	26	9	26	26		3 (25), 13 (26)
19	2nd child of 15	Not 5	20	6	23	26		3 (25), 13 (26)
	Remove 13							3 (25)
	Remove 3							

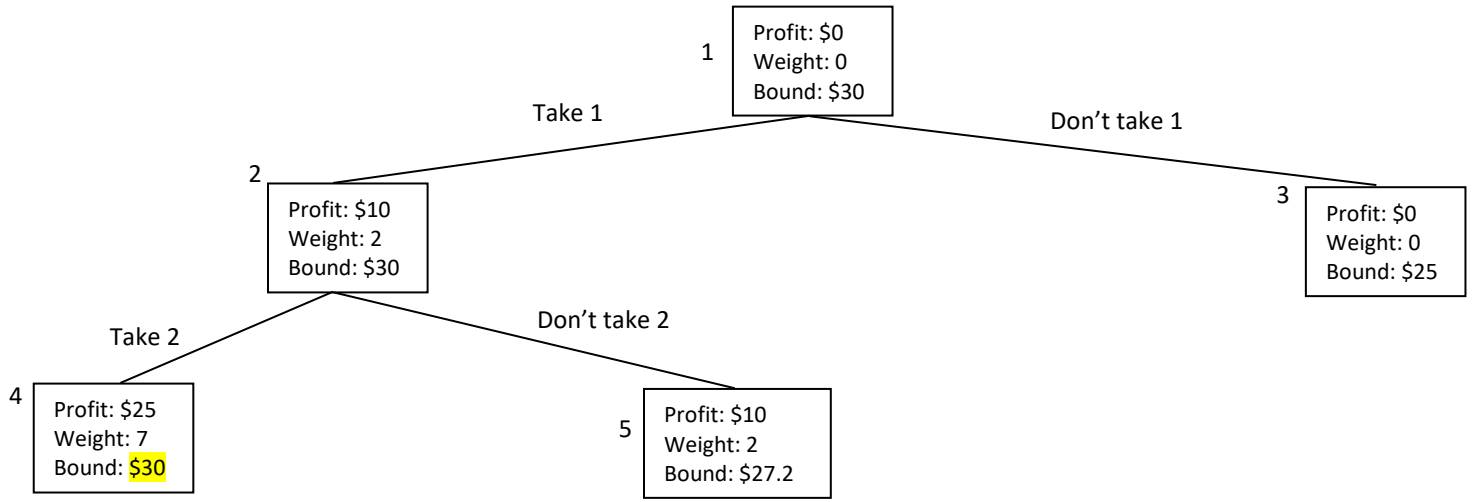
In the last 2 steps we remove the nodes and do nothing since they are no longer promising

Best-First Search with branch-and-bound pruning example: 0-1 Knapsack

i	p_i	w_i	p_i/w_i
1	10	2	5
2	15	5	3
3	10	4	2.5
4	12	5	2.4
5	6	3	2
6	4	4	1

W=9

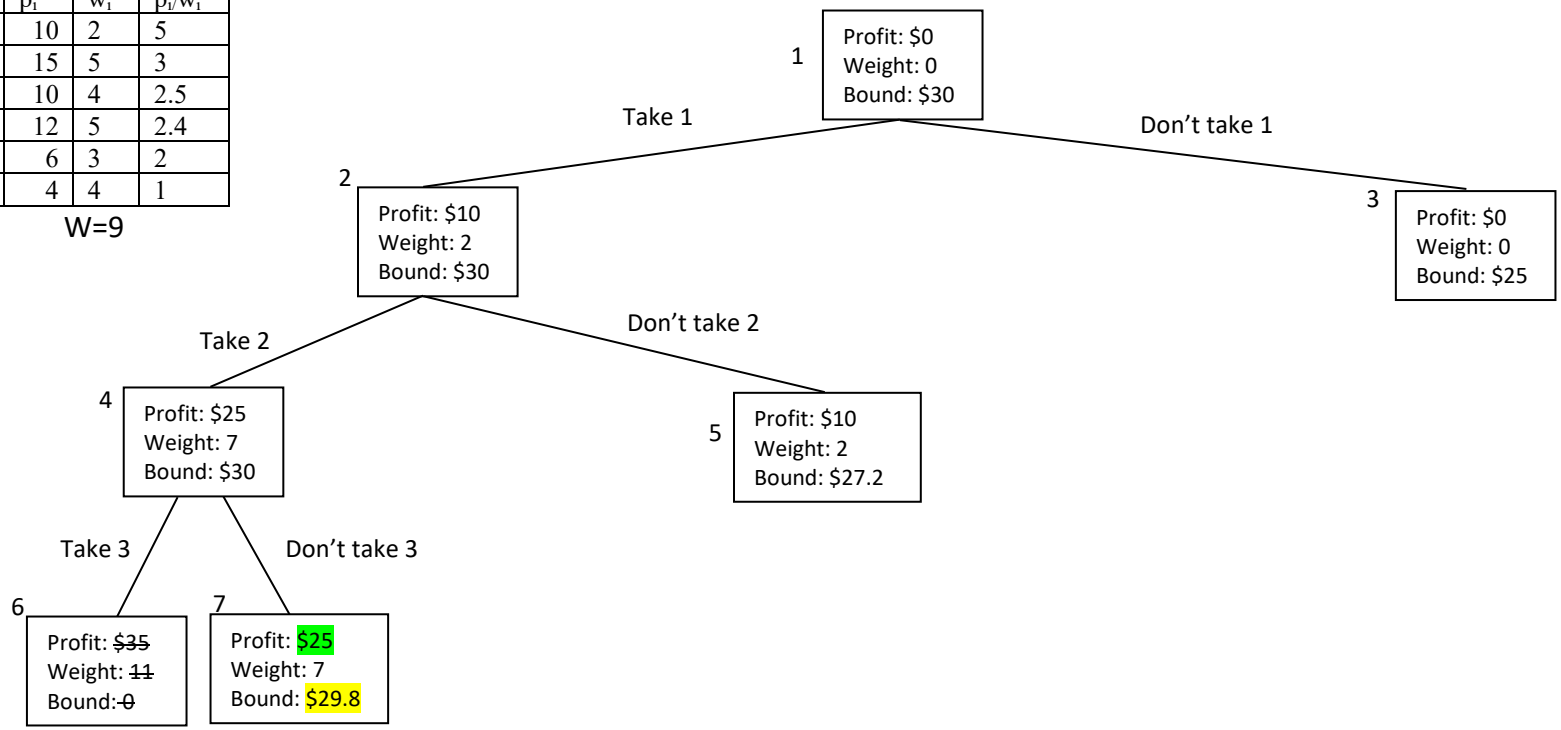




Best-First Search with branch-and-bound pruning example: 0-1 Knapsack

i	p_i	w_i	p_i/w_i
1	10	2	5
2	15	5	3
3	10	4	2.5
4	12	5	2.4
5	6	3	2
6	4	4	1

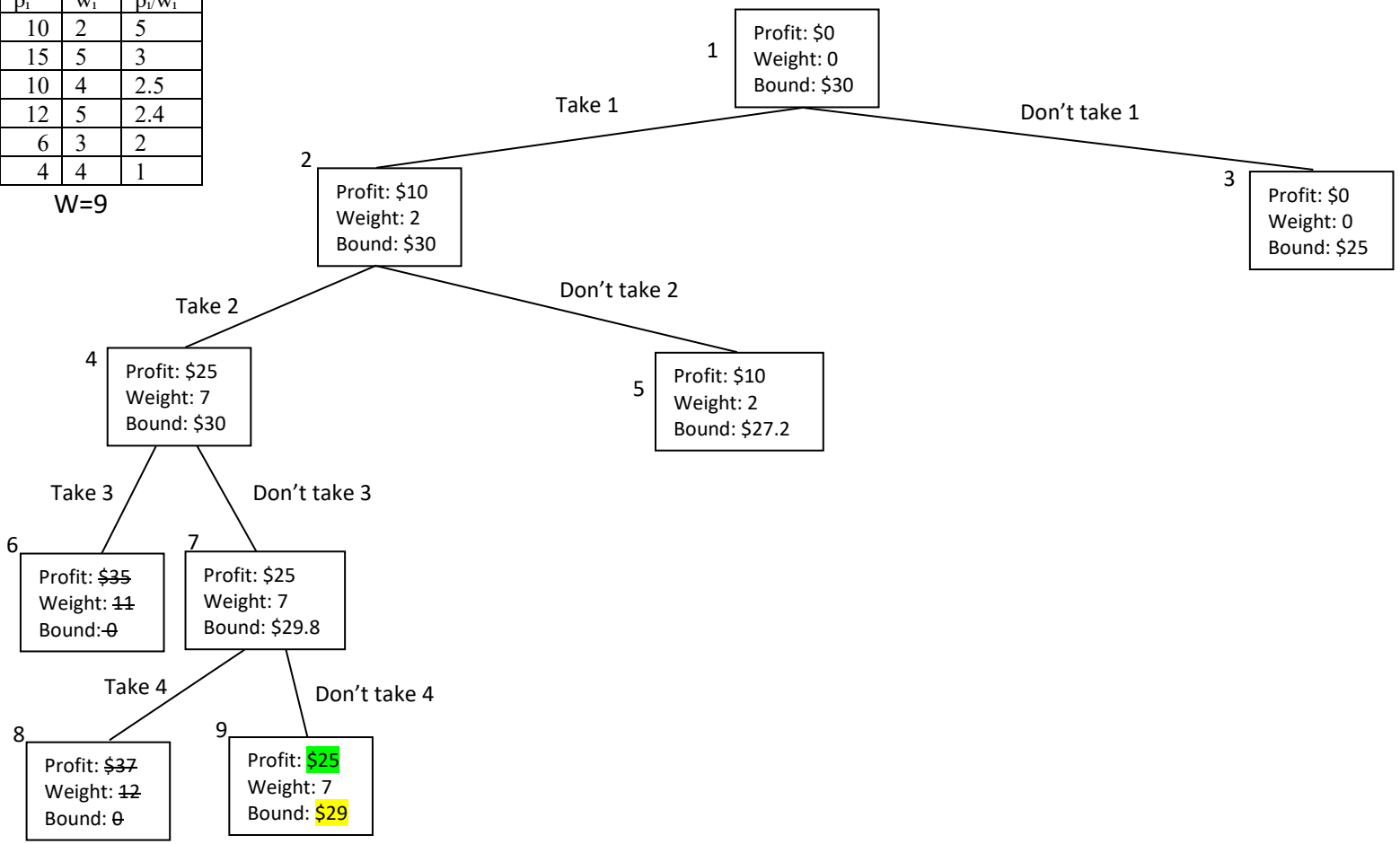
W=9



Best-First Search with branch-and-bound pruning example: 0-1 Knapsack

i	p_i	w_i	p_i/w_i
1	10	2	5
2	15	5	3
3	10	4	2.5
4	12	5	2.4
5	6	3	2
6	4	4	1

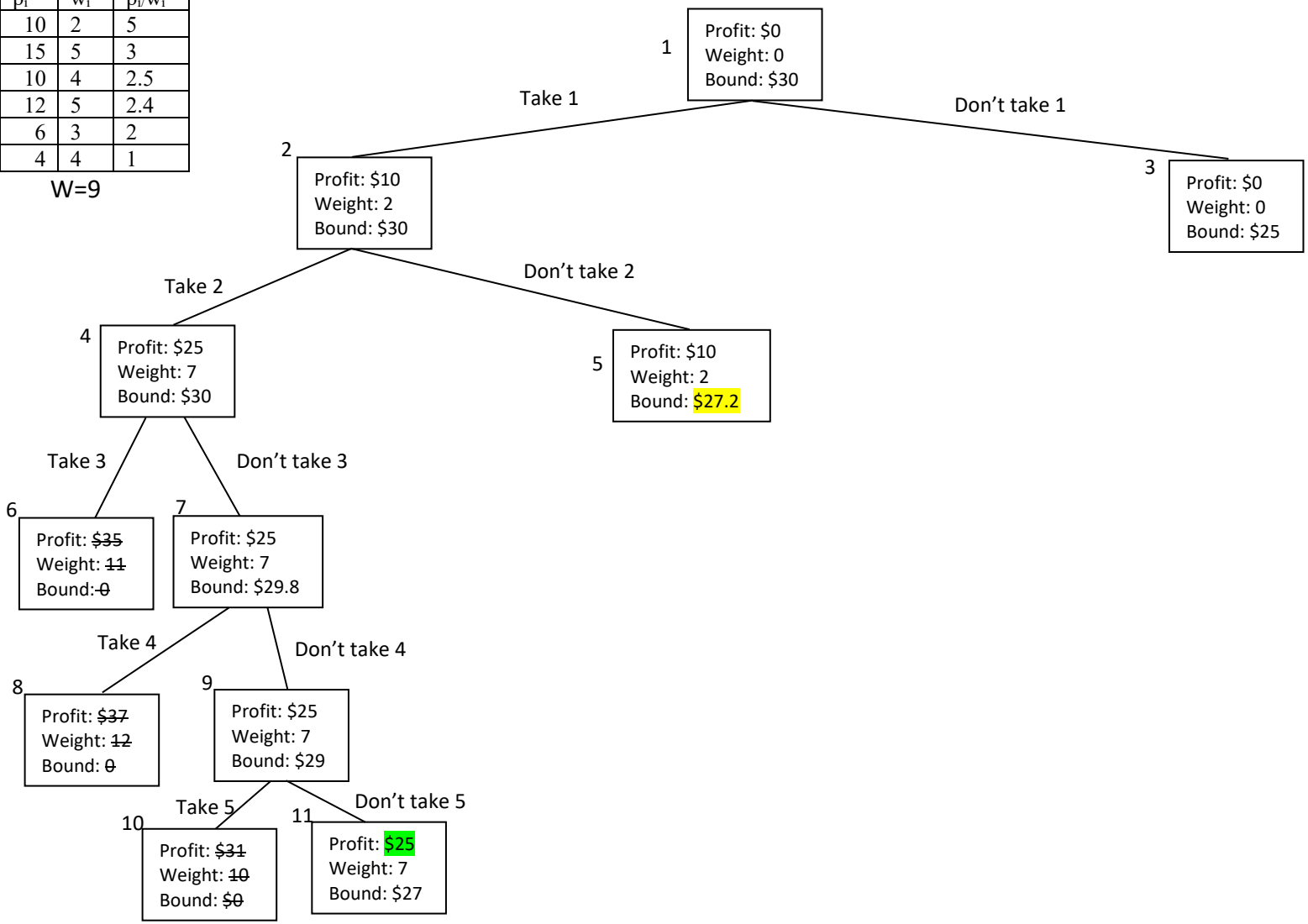
W=9



Best-First Search with branch-and-bound pruning example: 0-1 Knapsack

i	p_i	w_i	p_i/w_i
1	10	2	5
2	15	5	3
3	10	4	2.5
4	12	5	2.4
5	6	3	2
6	4	4	1

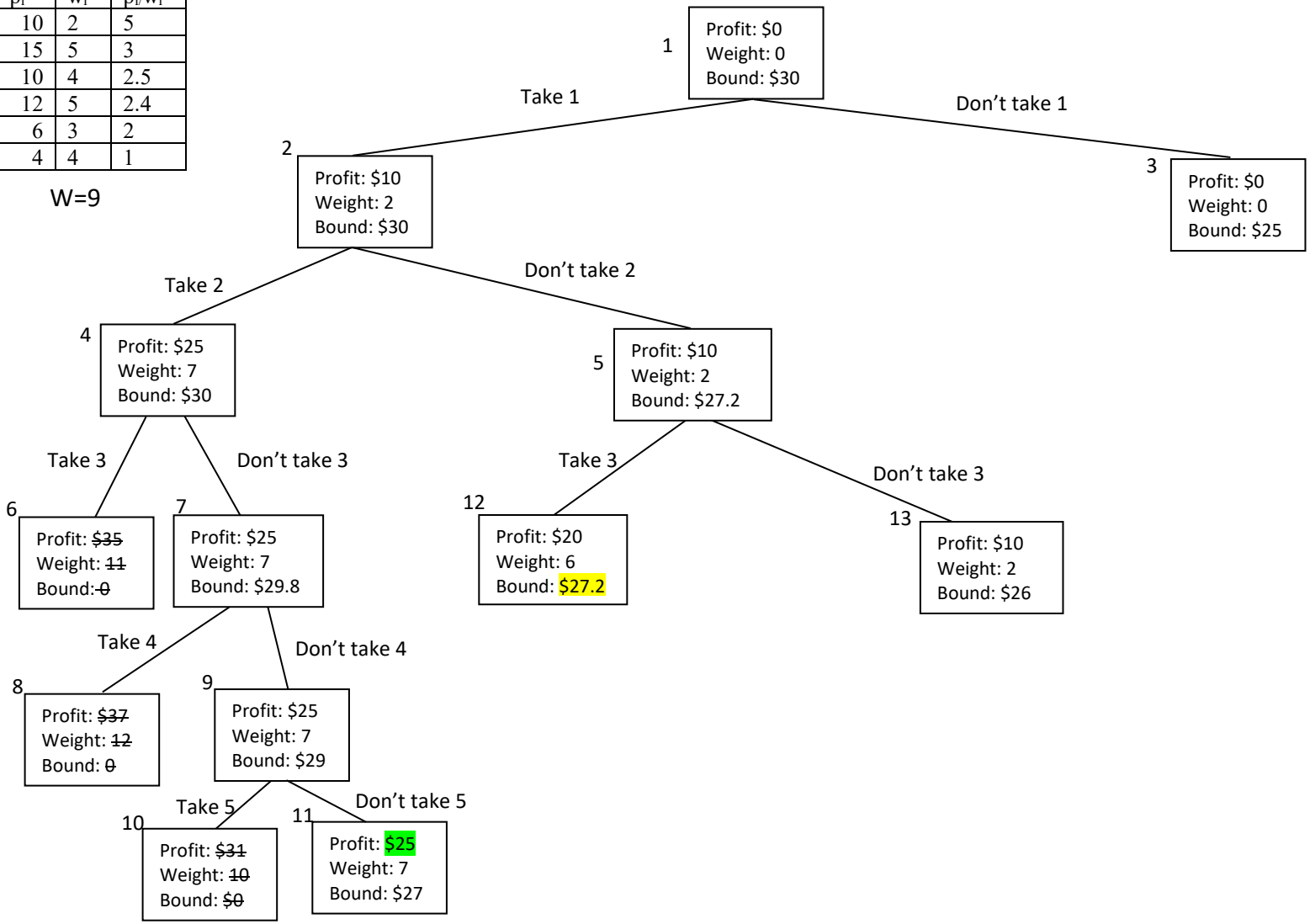
W=9



Best-First Search with branch-and-bound pruning example: 0-1 Knapsack

i	p_i	w_i	p_i/w_i
1	10	2	5
2	15	5	3
3	10	4	2.5
4	12	5	2.4
5	6	3	2
6	4	4	1

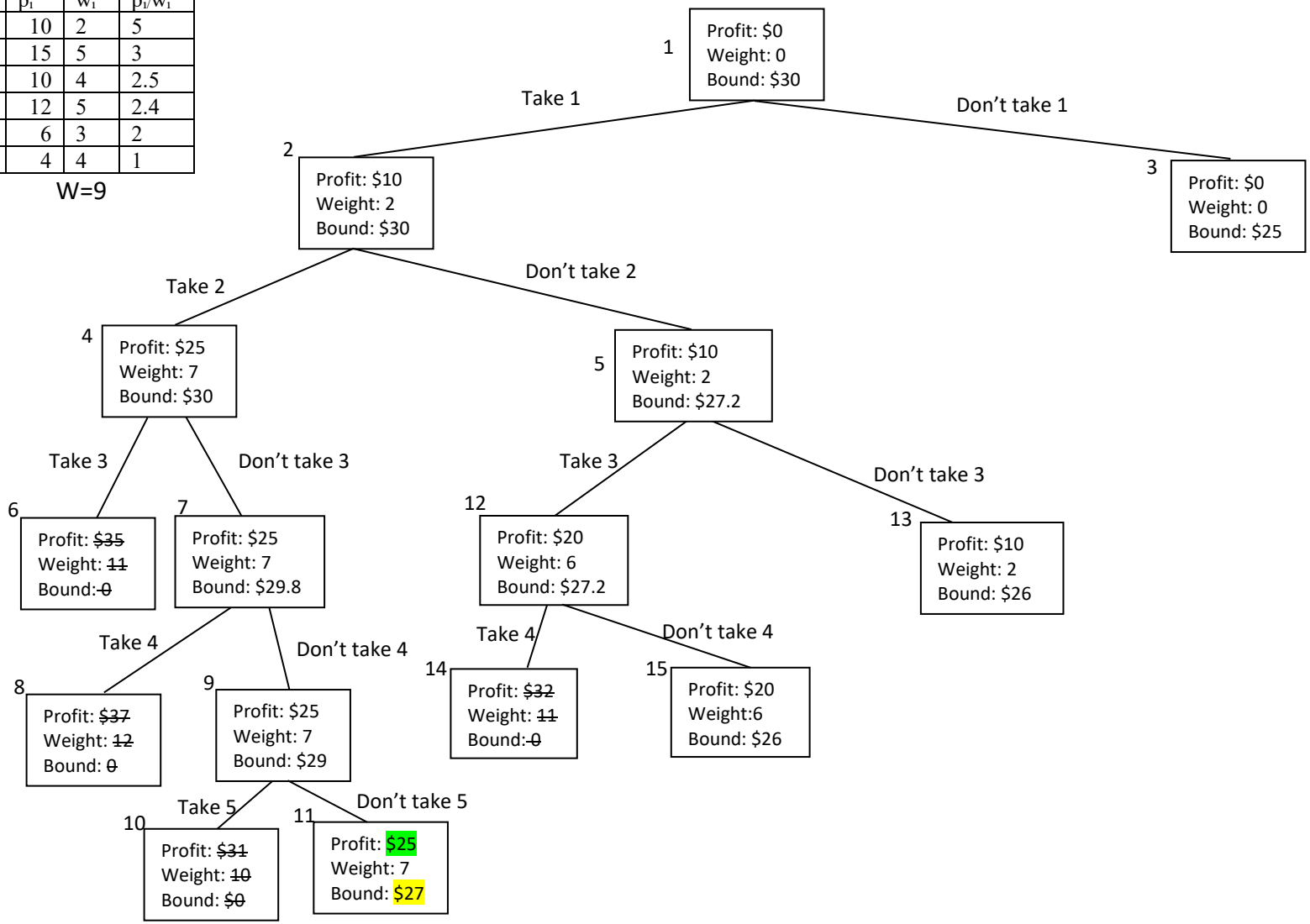
W=9



Best-First Search with branch-and-bound pruning example: 0-1 Knapsack

i	p_i	w_i	p_i/w_i
1	10	2	5
2	15	5	3
3	10	4	2.5
4	12	5	2.4
5	6	3	2
6	4	4	1

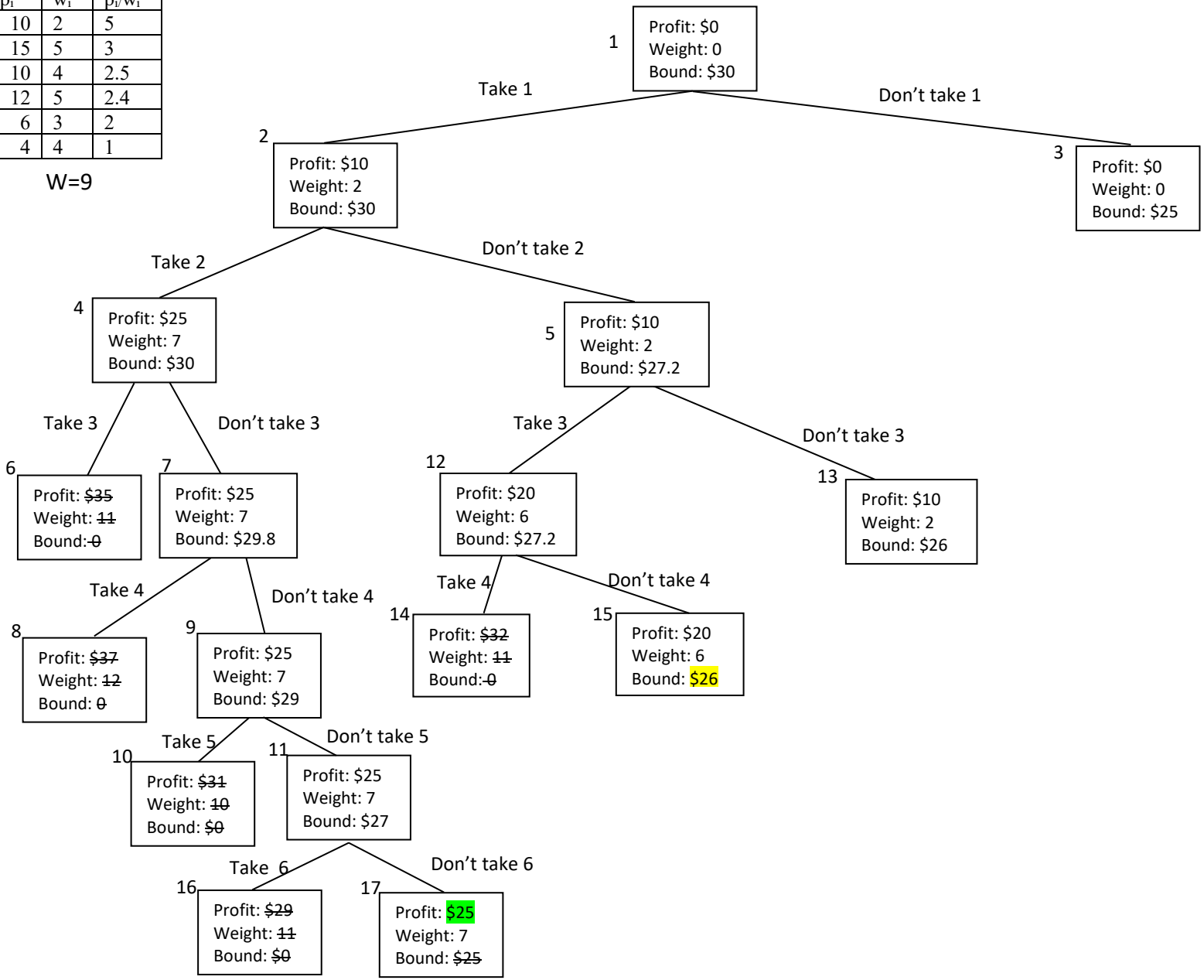
W=9



Best-First Search with branch-and-bound pruning example: 0-1 Knapsack

i	p_i	w_i	p_i/w_i
1	10	2	5
2	15	5	3
3	10	4	2.5
4	12	5	2.4
5	6	3	2
6	4	4	1

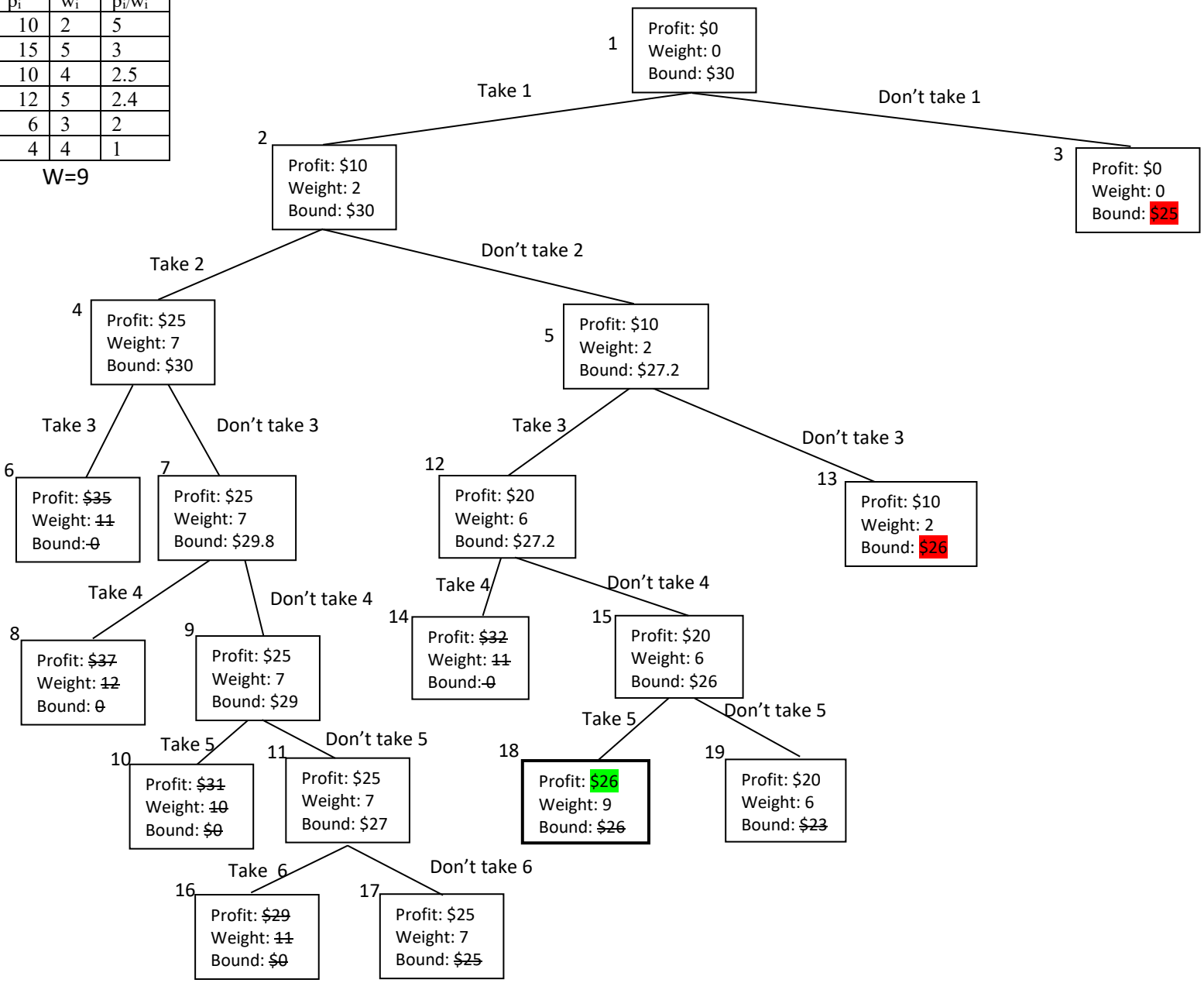
W=9



Best-First Search with branch-and-bound pruning example: 0-1 Knapsack

i	p_i	w_i	p_i/w_i
1	10	2	5
2	15	5	3
3	10	4	2.5
4	12	5	2.4
5	6	3	2
6	4	4	1

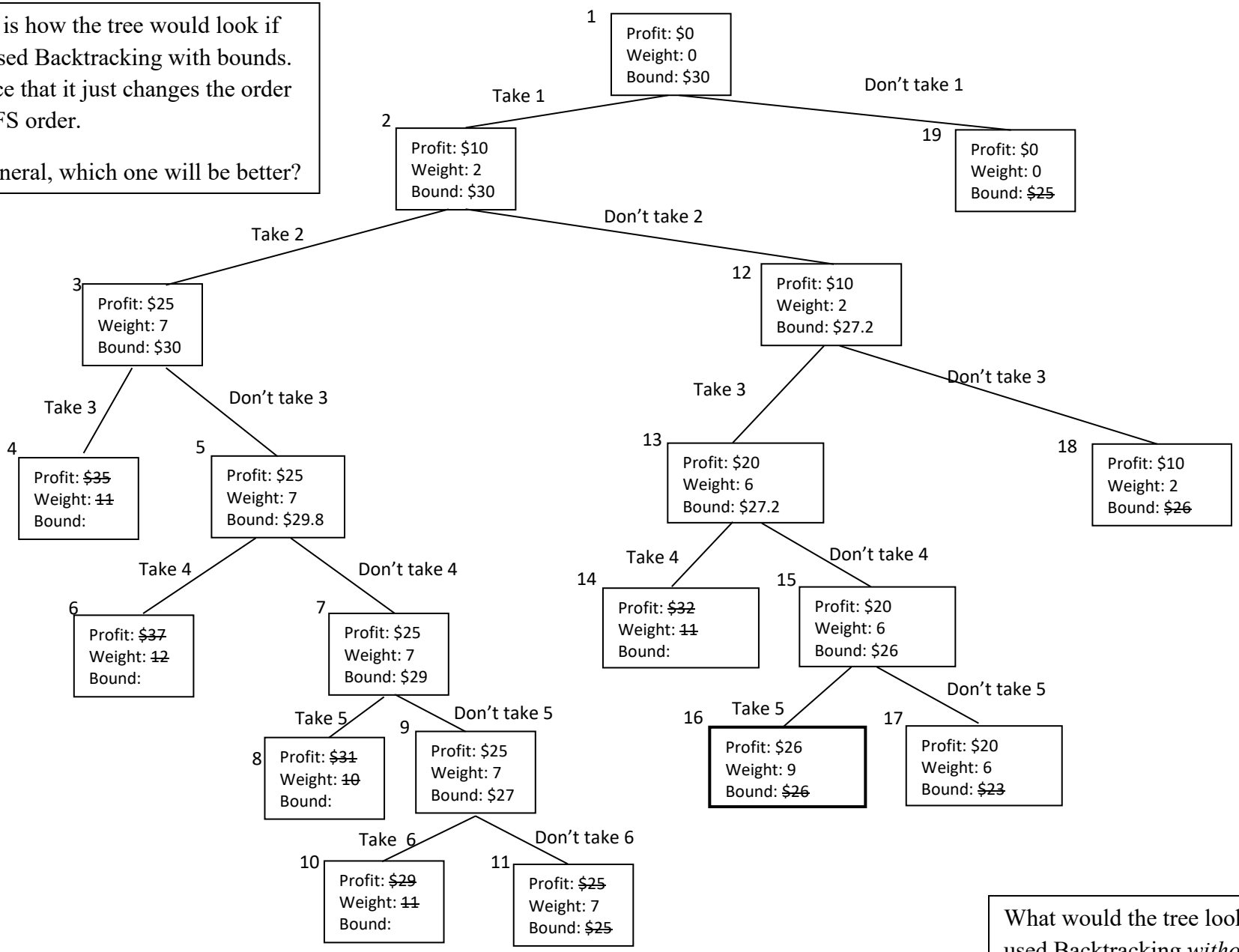
W=9



Best-First Search with branch-and-bound pruning example: 0-1 Knapsack

Here is how the tree would look if we used Backtracking with bounds. Notice that it just changes the order to DFS order.

In general, which one will be better?



What would the tree look like if we used Backtracking *without* bounds?