Item 1: $40 2 n= 4 W=16
Item 2: $30 5
Item 3: $50 10
Item 4: $10 5

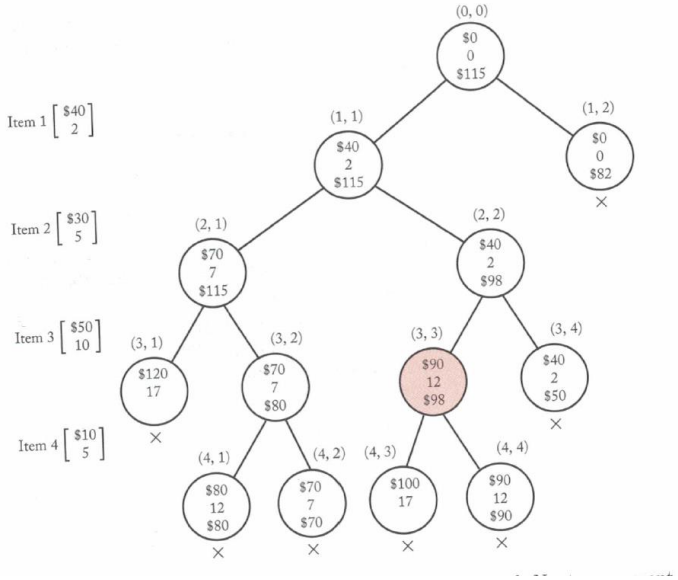

Figure 5.14
The pruned state space tree produced using backtracking in Example 5.6. Stored at each node from top to bottom are the total profit of the items stolen up to the node, their total weight, and the bound on the total profit that could be obtained by expanding beyond the node. The optimal solution is found at the node shaded in color. Each nonpromising node is marked with a cross.

Looking at state space tree -> a **path** is a candidate solution so each level represents the item so since we have 4 items, we will have room plus 4 levels.

(0,0) is always root and then (1,1) is the most left node at level 1 which is for item 1; (2, 1) is the most left node at level 2 which is for item 2; (3,1) most left node at level 3 which is for item 3; and (4,1) is most left node at level 4 which is for item 4;

Note 1: The levels will be the same in state space tree and pruned state space tree but nodes (x, y) will not be because pruned state tree will number nodes to be 1,2,3,4, etc. from left to right for the nodes that are visited.

Note 2: Each path, from root to leaf is a solution so it has to have the number of items. If we have 4 items, that means 4 nodes for each solution, one at each level.

(0, 0) root
maxprofit = 0, profit 0, weight 0, bound=115 (profit of first two items + fraction of third)
promising because weight (0) < W (16) AND bound (115) > maxprofit (0)

(1, 1) left child of root ==>**item 1**
profit = 40, weight = 2 (adding item 1 values to accumulative value)
maxprofit = 40 since weight (2) < W (16) and profit (40) > maxprofit (0)
bound=115 (current profit plus next item + fraction of third)
promising because weight (2) < W (16) AND bound (115) > maxprofit (40)

(2,1) left child of (1,1) ==> **item 2**
profit = 70, weight = 7 (adding item 2 values to accumulative value)
maxprofit = 70 since weight (7) < W (16) and profit (70) > maxprofit (40)
bound = 115 (current profit + fraction of next)
promising because weight (7) < W (16) AND bound (115) > maxprofit (70)

(3,1) left child of (2,1) ==> **item 3**
profit = 120, weight = 17 (adding item 3 values to accumulative value)
maxprofit = 70 since weight (17) < W (16) is false
non-promising because weight (17) < W (16) **false**

BACKTRACK to (3,1) parent which is (2,1) and go to **right child**

(3, 2) right child of (2,1) ==> **item 3**
Note: (**3**,y) means **3**rd level (3rd item) and (x,**2**) because it's second node in that level from left
profit=70, weight=7 (**NOT** including item 3 values)
maxprofit still 70 since weight (7) < W (16) **BUT** profit (70) > maxprofit (70) false
bound=80 (current profit plus next item (**item 4**) and since last item no fraction)
promising because weight (7) < W (16) AND bound (80) > maxprofit (70)

(4,1) left child of (3,1) ==> **item 4**
profit=80, weight=12 (adding item 4 values to accumulative value)
maxprofit = 80 since weight (12) < W (16) AND profit (80) > maxprofit (70)
bound=80 (current profit plus next item and since last item no fraction)
non-promising because weight (12) < W (16) BUT bound (80) > maxprofit (80) false

BACKTRACK to (34,1) parent which is (3,2) and go to **right child**

(4,2) right child of (3,2) ==> **item 4**
Note: do calculations not including item 4 since right child. Will be non-promising and will backtrack....