2.10.2017

Branch-and-bound

Knapsack problem

	1	2	3	4
w	2	5	10	5
p	40	30	50	10

$n = 4$

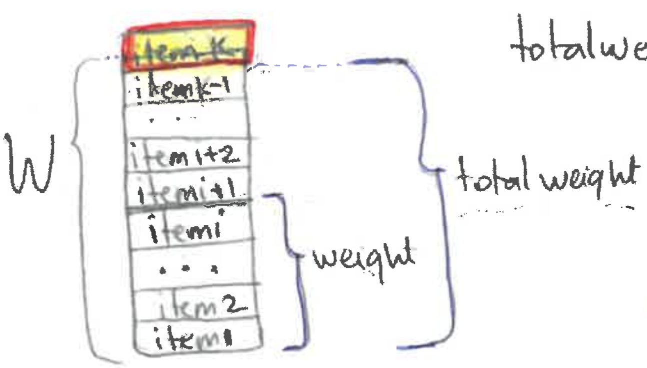
$W = 16$

item	profit p_i	weight w_i	p_i/w_i
1	\$40	2	\$20
2	\$30	5	\$6
3	\$50	10	\$5
4	\$10	5	\$2

- assume that the items are sorted in decreasing order of p_i/w_i
- for each node in the search tree:
  - weight - total weight of the items selected so far
  - profit - total profit of the items selected so far
  - bound - function which computes an upperbound of the profit that can be obtained with this partial solution
  - maxprofit - best profit so far
- if bound < maxprofit, then the node is nonpromising

- How do we compute the bound?

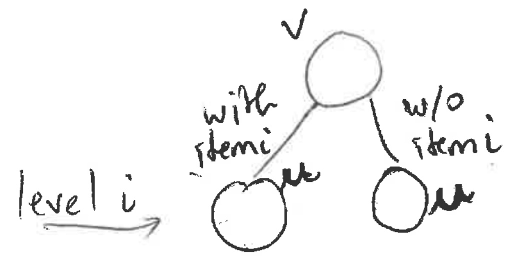
Assume that
- current node is at level i
- item k would bring the weight above W



$$\text{totalweight} = \text{weight} + \sum_{j=i+1}^{k-1} w_j$$

$$\text{bound} = \text{profit} + \sum_{j=i+1}^{k-1} p_j + \frac{(W - \text{totalweight})}{w_k} \cdot p_k$$

Knapsack with BreadthFS w/ Branch-and-Bound pruning (n,p[ ],w[ ],W,maxprofit)




```
Q = ∅
r.level = 0; r.profit = 0; r.weight = 0
maxprofit = 0
ENQUEUE(Q,r)
while Q ≠ ∅
   v = DEQUEUE(Q)
   u.level = v.level +1
   u.weight = v.weight + w[u.level]
   u.profit = v.profit + p[u.level]
   if (u.weight ≤ W and u.profit > maxprofit)
      maxprofit = u.profit
   if bound(u) > maxprofit
      ENQUEUE(Q,u)
   u.weight = v.weight
   u.profit = v.profit
   if bound(u) > maxprofit
      ENQUEUE(Q,u)
```

set u as the child of v that includes the next item

set u as the child of v that does not include the next item

RT analysis

- The number of nodes is:

geometric series

$$\leq 1 + 2 + 2^2 + 2^3 + \ldots + 2^n = \frac{2^{n+1}-1}{2-1} = 2\cdot 2^n - 1$$

total number of nodes $= O(2^n)$

- bound( ) function takes $O(n)$

$\boxed{\text{Total RT} = O(n \cdot 2^n)}$

Branch-and-bound with Breadth FS

Knapsack Problem

maxprofit = ~~0~~

40 (1)
~~70~~ (3)
90 (7)

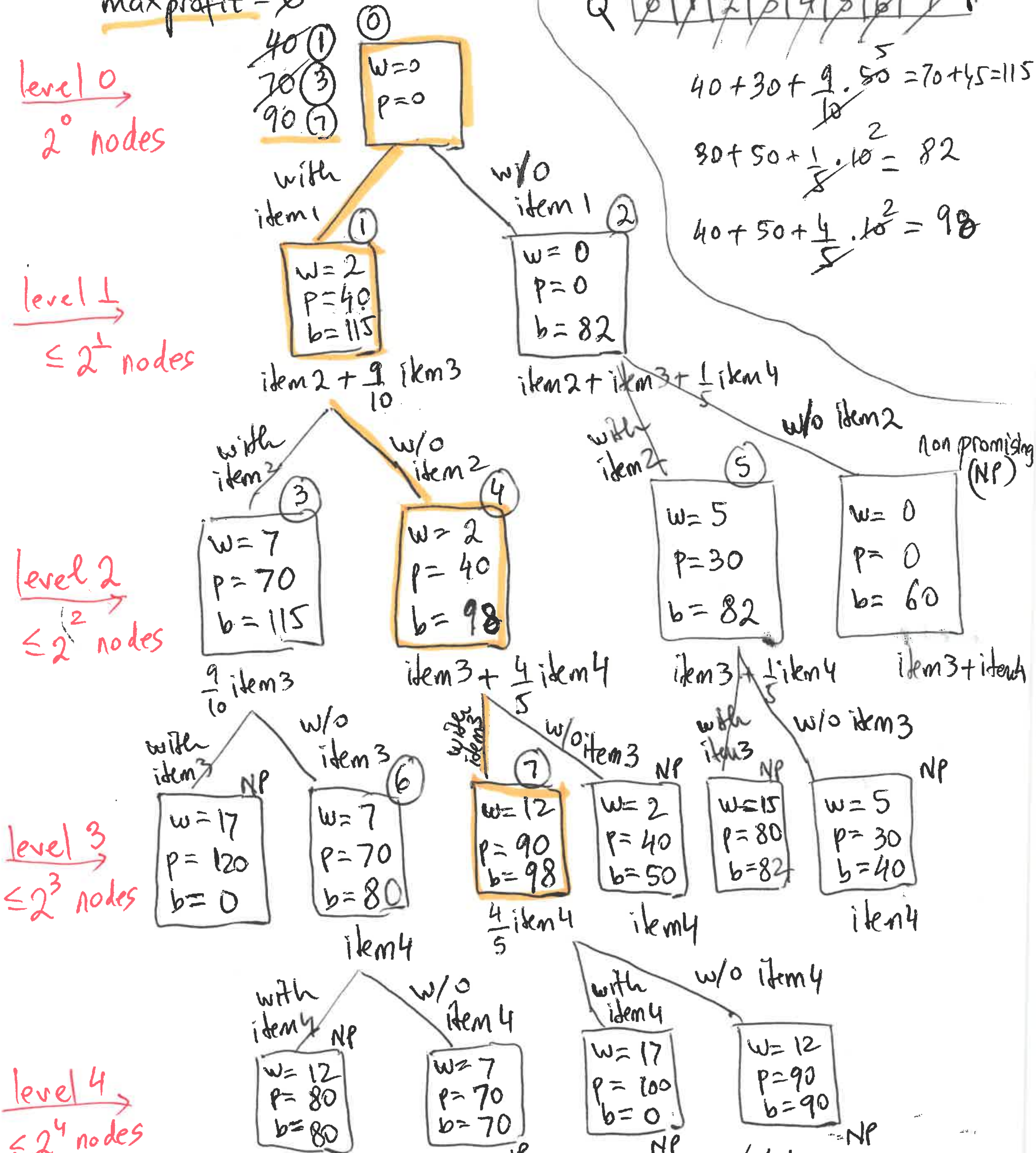
FIFO

Q | ~~0~~ | ~~1~~ | ~~2~~ | ~~3~~ | ~~4~~ | ~~5~~ | ~~6~~ | ~~7~~ |

$40+30+\frac{9}{\cancel{10}}\cdot\cancel{50}^{5} = 70+45 = 115$

$30+50+\frac{1}{\cancel{5}}\cdot\cancel{10}^{2} = 82$

$40+50+\frac{4}{\cancel{5}}\cdot\cancel{10}^{2} = 98$



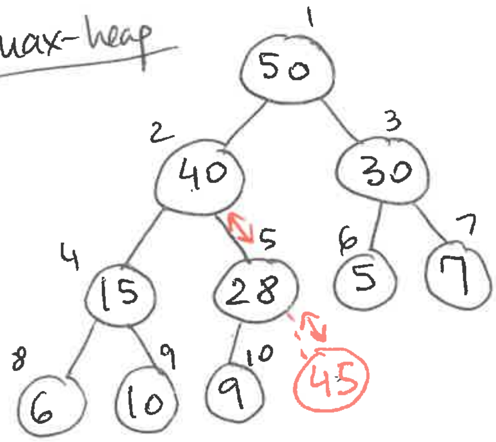
Solution = (item 1, item 3) → total weight = 12, total profit = 90 max-priority queue – implemented using a max-heap

CLRS chapter 6.5 page 162

- **insert**( ) and **remove**( ) operations take $O(\lg n)$ for an n-element queue

$\boxed{\text{example}}$

max-heap

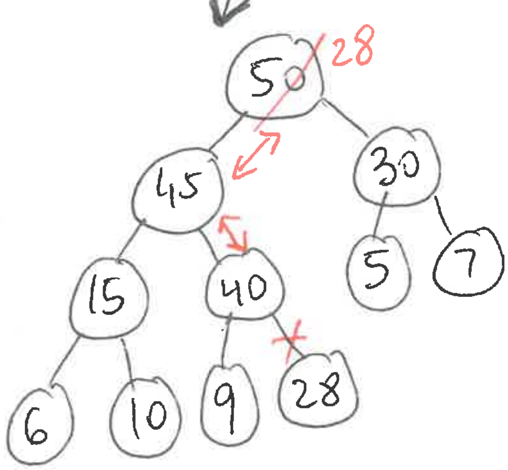


A

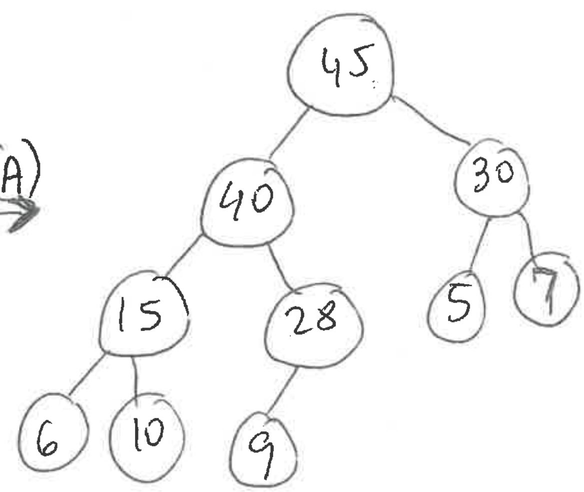
1	2	3	4	5	6	7	8	9	10
50	40	30	15	28	5	7	6	10	9

max-heap $\Rightarrow$ height $= \Theta(\lg n)$

insert(A, 45)



remove(A)



A =

50	45	30	15	40	5	7	6	10	9	28

A =

45	40	30	15	28	5	7	6	10	9

Knapsack-BestFS-Branch-and-Bound(n,p[],w[],W,maxprofit)

```
PQ = ∅
r.level = r.profit = r.weight = 0
maxprofit = 0
r.bound = bound(r)
insert(PQ,r)
while PQ ≠ ∅
    v = remove(PQ)                          → O(lg n)
    if v.bound > maxprofit
        u.level = v.level + 1
        u.weight = v.weight + w[u.level]
        u.profit = v.profit + p[u.level]
        if (u.weight ≤ W and u.profit > maxprofit)
                maxprofit = u.profit
        u.bound = bound(u)                  → O(n)
        if bound(u) > maxprofit
                insert(PQ,u)                → O(lg n)
        u.weight = v.weight
        u.profit = v.profit
        u.bound = bound(u)                  → O(n)
        if u.bound > maxprofit
                insert(PQ,u)
```

set u as the child of v that includes the next item

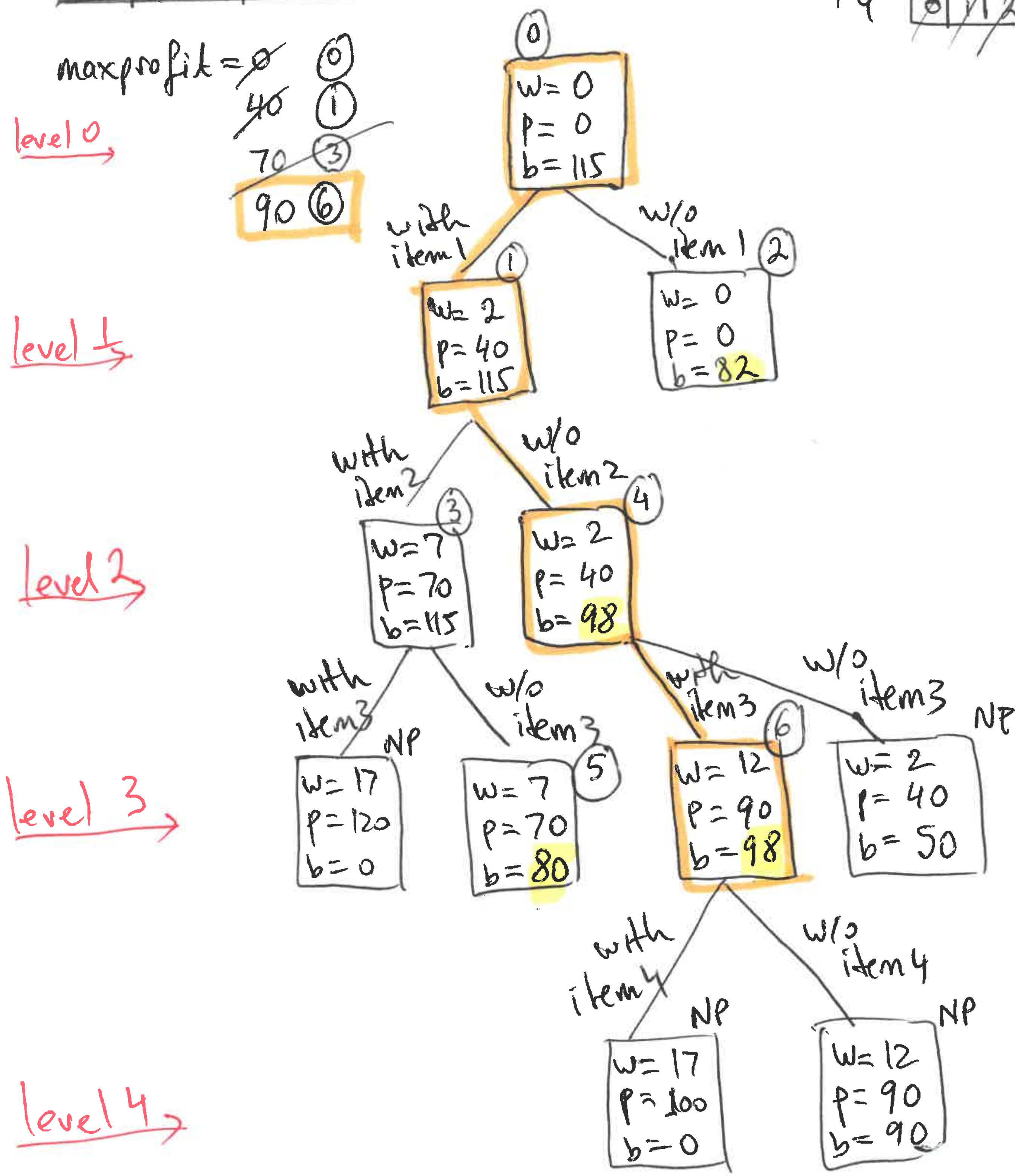
set u as the child of v that does not include the next item

$RT = O(n \cdot 2^n)$

Branch-and-bound with Best FS

Knapsack problem

PQ ~~0~~ ~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ ~~6~~



Solution = (item 1, item 3)