Dynamic Programming

Dynamic Programming

- An algorithmic paradigm that solves problems by **combining the solutions to subproblems**, like the Divide-and-Conquer method.

Dynamic Programming

- Recall: Divide-and-Conquer algorithms…

Dynamic Programming

- Recall: Divide-and-Conquer algorithms partition the problems into independent subproblems, solve the subproblems recursively and then combine their solutions to solve the original problem.

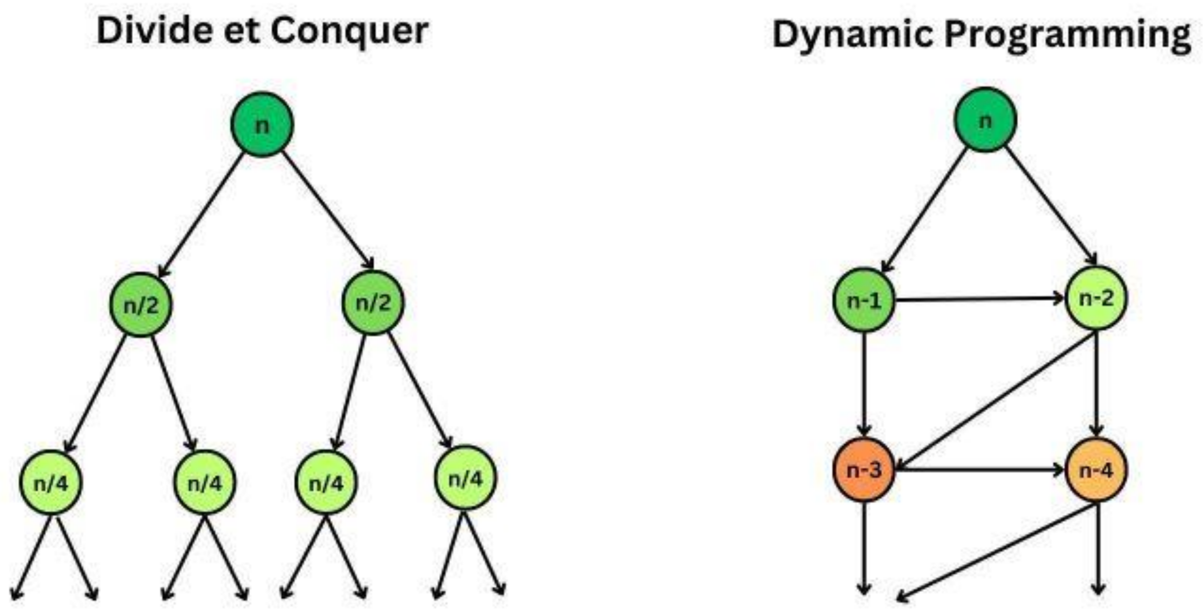
- **Dynamic programming** is applicable when subproblems are not independent.

Dynamic Programming

- In DP, subproblems **share** subsubproblems.

- In this context, divide-and-conquer algorithm does more work than necessary, repeatedly solving the common subsubproblems.

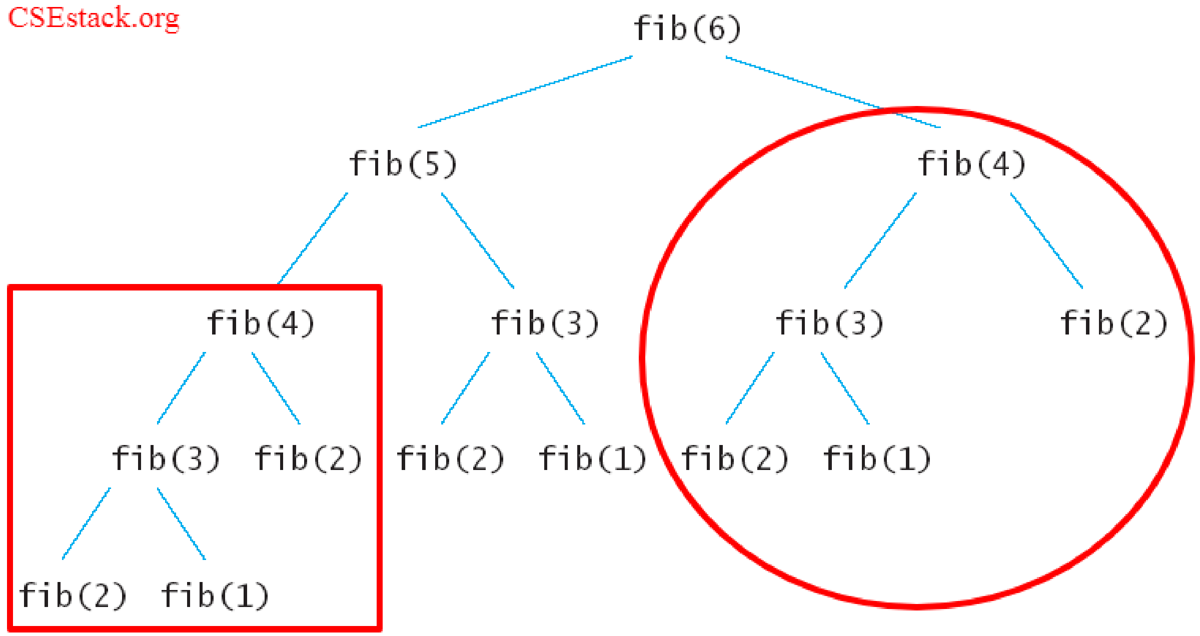
Divide and Conquer



Dynamic Programming

- A dynamic programming algorithm solves every subsubproblem just once and then saves its answer in a table, thereby **avoiding the work of recomputing** the answer every time the subsubproblem is encountered.

Dynamic Programming



Dynamic Programming

- Characterize the structure of an optimal solution
- Recursively define the value of an optimal solution
- Compute the value of an optimal solution in a bottom-up fashion
- Construct an optimal solution from computed information (can be omitted if only value of optimal solution is required)

Knapsack Problem

Introduction

- During a robbery, a burglar finds much more loot than he had expected and has to decide what to take.
- His bag (or knapsack) will hold a total weight of at most W kilograms.

Introduction

- There are n items to choose from, of weight w_1 ,w_2 ,…,w_n and money value v_1 ,v_2 ,…,v_n.

- What is the most valuable combination of items (or the combination of items that maximizes the monetary value) he can fit into his bag?

Knapsack Problem

- Practical applications:
  - W units of CPU time available (CPU scheduling)
  - bandwidth in network

How can we solve the problem?

Use **Brute-force** approach.

Question

- What is the naive / brute-force approach in solving the Knapsack problem?

- Answer: We can check every possible combination of items to know what the most valuable combination of items is

Brute-force Approach

- Given n items, how many possible combination of items are there?

- Answer: 2^n (Exponential)

Brute-force Approach

- For one combination, how long does it take to check if it is a valid combination?

- Answer: O(1) (Constant)

Brute-force Approach

- Total Running time = 2^n * O(1)
- Total Running time = O(2^n) [Exponential Time]
- 2^n possible combinations, that each take O(1) time to check for validity

Can we do better?

Duh!

PROgrammer says….

YES!

Use **Dynamic** Approach!

Defining a Subproblem

- We can do better with an algorithm based on dynamic programming

- We need to carefully identify the subproblems

Let's try this:

- If items are labeled *1..n*, then a subproblem would be to find an optimal solution for
- S_k = *{items labeled 1, 2, .. k}*

Defining a Subproblem

If items are labeled *1..n*, then a subproblem would be to find an optimal solution for S_k = *{items labeled 1, 2, .. k}*

- This is a reasonable subproblem definition.
- The question is: can we describe the final solution (S_n) in terms of subproblems (S_k)?
- Unfortunately, we can't do that.

Defining a Subproblem

- Given a knapsack with maximum capacity W, and a set S consisting of n items
- Each item i has some weight w_i and benefit value b_i (all w_i and W are integer values)
- Problem: How to pack the knapsack to achieve maximum total value of packed items?

Defining a Subproblem

- Let's add another parameter: *w*, which will represent the maximum weight for each subset of items

- The subproblem then will be to compute *V[k,w]*, *i.e.*, to find an optimal solution for S_k = *{items labeled 1, 2, .. k} in a knapsack of size w*

Recursive Formula for Subproblems

- The subproblem will then be to compute *V[k,w], i.e.,* to find an optimal solution for S_k = *{items labeled 1, 2, .. k} in a knapsack of size w*

- Assuming knowing V[i, j], where i=0,1, 2, … k-1, j=0,1,2, …w, how to derive V[k,w]?

Recursive Formula for subproblems (continued)

$$V[k,w] = \begin{cases} V[k-1,w] & \text{if } w_k > w \\ \max\{V[k-1,w], V[k-1,w-w_k]+b_k\} & \text{else} \end{cases}$$

Recursive Formula

$$V[k,w] = \begin{cases} V[k-1,w] & \text{if } w_k > w \\ \max\{V[k-1,w], V[k-1,w-w_k]+b_k\} & \text{else} \end{cases}$$

- The best subset of S_k that has the total weight $\leq w$, either contains item k or not.
- First case: $w_k > w$. Item k can't be part of the solution, since if it was, the total weight would be $> w$, which is unacceptable.
- Second case: $w_k \leq w$. Then the item k <u>can</u> be in the solution, and we choose *the case with greater value*.

Knapsack Algorithm

```
for w = 0 to W
   V[0,w] = 0
for i = 1 to n
   V[i,0] = 0
for i = 1 to n
   for w = 0 to W
            if w_i <= w // item i can be part of the solution
                     if b_i + V[i-1,w-w_i] > V[i-1,w]
                              V[i,w] = b_i + V[i-1,w- w_i]
                     else
                              V[i,w] = V[i-1,w]
            else V[i,w] = V[i-1,w]  // w_i > w
```

Running Time

```
for w = 0 to W
  V[0,w] = 0
for i = 1 to n
  V[i,0] = 0
for i = 1 to n
  for w = 0 to W
          < the rest of the code >
```

$O(W)$

Repeat n times

$O(W)$

What is the running time of this algorithm?

O(n*W)

Remember that the brute-force algorithm takes $O(2^n)$

Example

Let’s run our algorithm on the following data:

- n = 4 (# of elements)
- W = 5 (max weight)
- Elements (weight, benefit):
- (2,3), (3,4), (4,5), (5,6)

			0	1	2	3	4	5
b_i	w_i	0						
3	2	1						
4	3	2						
5	4	3						
6	5	4						

```
for i = 1 to n
    for w = 0 to W
        if w_i <= w
            if b_i + V[i-1,w-w_i] > V[i-1,w]
                V[i,w] = b_i + V[i-1,w- w_i]
            else
                V[i,w] = V[i-1,w]
        else V[i,w] = V[i-1,w]
```

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1						
4	3	2						
5	4	3						
6	5	4						

for w = 0 to W
    V[0,w] = 0

			0	1	2	3	4	5
b_i	w_i	0	0	0	0	0	0	0
3	2	1	0					
4	3	2	0					
5	4	3	0					
6	5	4	0					

for i = 1 to n
  V[i,0] = 0

Can the item with weight=2 fit in if knapsack capacity is 1? No!

			0	1	2	3	4	5
b_i	w_i	0	0	0	0	0	0	0
3	2	1	0	0				
4	3	2	0					
5	4	3	0					
6	5	4	0					

```
for i = 1 to n                          i=1
    for w = 0 to W                      w=1
        if w_i <= w                     2 >1
            if b_i + V[i-1,w-w_i] > V[i-1,w]
                V[i,w] = b_i + V[i-1,w- w_i]
            else
                V[i,w] = V[i-1,w]
        else V[i,w] = V[i-1,w]          V[1,1]=V[0,1]
```

Can the item with weight=2 fit in if knapsack capacity is 2? Yes!

			0	1	2	3	4	5
b_i	w_i	0	0	0	0	0	0	0
3	2	1	0	0				
4	3	2	0					
5	4	3	0					
6	5	4	0					

for i = 1 to n $\boxed{i=1}$

for w = 0 to W $\boxed{w=2}$

if $w_i <= w$ $\boxed{2=2}$

if $b_i + V[i\text{-}1, w\text{-}w_i] > V[i\text{-}1, w]$

$V[i,w] = b_i + V[i\text{-}1, w\text{-}w_i]$

else

$V[i,w] = V[i\text{-}1, w]$

else $V[i,w] = V[i\text{-}1, w]$

Benefit of adding the item + the benefit of the first i items added

Can the item with weight=2 fit in if knapsack capacity is 2? Yes!

At capacity =2, and weight of the item =2, the remaining weight that we can add is = 0

At weight capacity = 0, no item may be added so benefit = 0

			0	*1*	*2*	*3*	*4*	*5*
b_i	w_i	*0*	0	0	0	0	0	0
3	*2*	*1*	0	0				
4	*3*	*2*	0					
5	*4*	*3*	0					
6	*5*	*4*	0					

$3 + V[0, 0]=0 > V[0, 2]=0$

```
for i = 1 to n                        i=1
    for w = 0 to W                    w=2
        if w_i <= w                   2 =2
            if b_i + V[i-1,w-w_i] > V[i-1,w]
                V[i,w] = b_i + V[i-1,w- w_i]
            else
                V[i,w] = V[i-1,w]
        else V[i,w] = V[i-1,w]
```

Can the item with weight=2 fit in if knapsack capacity is 2? Yes!

At capacity =2, and weight of the item =2, the remaining weight that we can add is = 0

At weight capacity = 0, no item may be added so benefit = 0

			0	*1*	*2*	*3*	*4*	*5*
b_i	w_i	*0*	0	0	0	0	0	0
3	*2*	*1*	0	0	3			
4	*3*	*2*	0					
5	*4*	*3*	0					
6	*5*	*4*	0					

for $i = 1$ *to* n $\boxed{i=1}$

  for $w = 0$ *to* W $\boxed{w=2}$

    if $w_i <= w$ $\boxed{2=2}$

      if $b_i + V[i-1,w-w_i] > V[i-1,w]$

        $V[i,w] = b_i + V[i-1,w- w_i]$ $\boxed{3}$ + $\boxed{V[0, 0]=0}$

      else

        $V[i,w] = V[i-1,w]$

    else $V[i,w] = V[i-1,w]$

Can the item with weight=2 fit in if knapsack capacity is 3? Yes!

At capacity =3, and weight of the item =2, the remaining weight that we can add is = 1

At weight capacity = 1, no item may be added so benefit = 0

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3		
4	3	2	0					
5	4	3	0					
6	5	4	0					

for $i = 1$ to n $\boxed{i=1}$

  for $w = 0$ to W $\boxed{w=3}$

    if $w_i <= w$ $\boxed{2<3}$

      if $b_i + V[i-1,w-w_i] > V[i-1,w]$

        $V[i,w] = b_i + V[i-1,w-w_i]$

      else

        $V[i,w] = V[i-1,w]$

    else $V[i,w] = V[i-1,w]$

$\boxed{3} + \boxed{V[0, 1]=0} > \boxed{V[0, 3]=0}$

Can the item with weight=2 fit in if knapsack capacity is 3? Yes!

At capacity =3, and weight of the item =2, the remaining weight that we can add is = 1

At weight capacity = 1, no item may be added so benefit = 0

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3		
4	3	2	0					
5	4	3	0					
6	5	4	0					

for i = 1 to n $\boxed{i=1}$

for w = 0 to W $\boxed{w=3}$

if $w_i <= w$ $\boxed{2<3}$

if $b_i + V[i\text{-}1,w\text{-}w_i] > V[i\text{-}1,w]$

$V[i,w] = b_i + V[i\text{-}1,w\text{-} w_i]$ $\boxed{3}$ + $\boxed{V[0, 1]=0}$

else

$V[i,w] = V[i\text{-}1,w]$

else $V[i,w] = V[i\text{-}1,w]$

Can the item with weight=2 fit in if knapsack capacity is 4? Yes!

At capacity =4, and weight of the item =2, the remaining weight that we can add is = 2

At weight capacity = 2, no item may be added so benefit = 0

			0	1	2	3	4	5
b_i	w_i	0	0	0	0	0	0	0
3	2	1	0	0	3	3		
4	3	2	0					
5	4	3	0					
6	5	4	0					

$3 + V[0, 2]=0 > V[0, 4]=0$

for i = 1 to n $i=1$

    for w = 0 to W $w=4$

        if $w_i <= w$ $2 < 4$

            if $b_i + V[i-1,w-w_i] > V[i-1,w]$

                $V[i,w] = b_i + V[i-1,w- w_i]$

            else

                $V[i,w] = V[i-1,w]$

        else $V[i,w] = V[i-1,w]$

Can the item with weight=2 fit in if knapsack capacity is 4? Yes!

At capacity =3, and weight of the item =2, the remaining weight that we can add is = 1

At weight capacity = 1, no item may be added so benefit = 0

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	
4	3	2	0					
5	4	3	0					
6	5	4	0					

for $i = 1$ *to* n $i=1$

    for $w = 0$ *to* W $w=4$

        if $w_i <= w$ $2 < 4$

            if $b_i + V[i\text{-}1, w\text{-}w_i] > V[i\text{-}1, w]$

                $V[i,w] = b_i + V[i\text{-}1, w\text{-} w_i]$ $3 + V[0, 2]=0$

            else

                $V[i,w] = V[i\text{-}1, w]$

        else $V[i,w] = V[i\text{-}1, w]$

Can the item with weight=2 fit in if knapsack capacity is 5? Yes!

At capacity =5, and weight of the item =2, the remaining weight that we can add is = 3

At weight capacity = 3, no item may be added so benefit = 0

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	
4	3	2	0					
5	4	3	0					
6	5	4	0					

3 + V[0, 3]=0 > V[0, 4]=0

for i = 1 to n   *i=1*

    for w = 0 to W   *w=5*

        if w_i *<= w*   *2 <5*

            if $b_i + V[i-1,w-w_i] > V[i-1,w]$

                $V[i,w] = b_i + V[i-1,w- w_i]$

            else

                $V[i,w] = V[i-1,w]$

        else $V[i,w] = V[i-1,w]$

Can the item with weight=2 fit in if knapsack capacity is 4? Yes!

At capacity =3, and weight of the item =2, the remaining weight that we can add is = 1

At weight capacity = 1, no item may be added so benefit = 0

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0					
5	4	3	0					
6	5	4	0					

for i = 1 to n $\boxed{i=1}$

for w = 0 to W $\boxed{w=5}$

if $w_i <= w$ $\boxed{2 < 5}$

if $b_i + V[i\text{-}1,w\text{-}w_i] > V[i\text{-}1,w]$

$V[i,w] = b_i + V[i\text{-}1,w\text{-}\ w_i]$ $\boxed{3}$ + $\boxed{V[0, 3]=0}$

else

$V[i,w] = V[i\text{-}1,w]$

else $V[i,w] = V[i\text{-}1,w]$

Can the item with weight=3 fit in if knapsack capacity is 1? No!

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0				
5	4	3	0					
6	5	4	0					

for i = 1 to n $\boxed{i=2}$

  for w = 0 to W $\boxed{w=5}$

    if $w_i <= w$ $\boxed{3 > 1}$

      if $b_i + V[i-1,w-w_i] > V[i-1,w]$

        $V[i,w] = b_i + V[i-1,w- w_i]$

      else

        $V[i,w] = V[i-1,w]$

    else $V[i,w] = V[i-1,w]$ $\boxed{V[2, 1]=V[1, 1]}$

Can the item with weight=3 fit in if knapsack capacity is 2? No!

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3			
5	4	3	0					
6	5	4	0					

for i = 1 to n $i=2$

    for w = 0 to W $w=5$

        if $w_i <= w$ $3 > 2$

            if $b_i + V[i-1,w-w_i] > V[i-1,w]$

                $V[i,w] = b_i + V[i-1,w- w_i]$

            else

                $V[i,w] = V[i-1,w]$

        else $V[i,w] = V[i-1,w]$ $V[2, 2]=V[1, 2]$

Can the item with weight=3 fit in if knapsack capacity is 3? Yes!

At capacity =3, and weight of the item =3, the remaining weight that we can add is = 0

At weight capacity = 0, no item may be added so benefit = 0

			0	1	2	3	4	5
b_i	w_i	0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4		
5	4	3	0					
6	5	4	0					

4 + $V[1, 0]=0$ > $V[1, 3]=0$

$for\ i = 1\ to\ n$ $\boxed{i=2}$

$\quad for\ w = 0\ to\ W$ $\boxed{w=3}$

$\quad\quad if\ w_i <= w$ $\boxed{3=3}$

$\quad\quad\quad if\ b_i + V[i-1,w-w_i] > V[i-1,w]$

$\quad\quad\quad\quad V[i,w] = b_i + V[i-1,w- w_i]$

$\quad\quad\quad else$

$\quad\quad\quad\quad V[i,w] = V[i-1,w]$

$\quad\quad else\ V[i,w] = V[i-1,w]$

Can the item with weight=3 fit in if knapsack capacity is 3? Yes!

At capacity =3, and weight of the item =3, the remaining weight that we can add is = 0

At weight capacity = 0, no item may be added so benefit = 0

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4		
5	4	3	0					
6	5	4	0					

for i = 1 to n   i=2

    for w = 0 to W   w=3

        if $w_i <= w$ 3 =3

            if $b_i + V[i-1,w-w_i] > V[i-1,w]$

                $V[i,w] = b_i + V[i-1,w- w_i]$ 4 + V[1, 0]=0

            else

                $V[i,w] = V[i-1,w]$

        else $V[i,w] = V[i-1,w]$

Can the item with weight=3 fit in if knapsack capacity is 4? Yes!

At capacity =4, and weight of the item =3, the remaining weight that we can add is = 1

At weight capacity = 1, no item may be added so benefit = 0

			0	*1*	*2*	*3*	*4*	*5*
b_i	w_i	*0*	0	0	0	0	0	0
3	*2*	*1*	0	0	3	3	3	3
4	*3*	*2*	0	0	3	4		
5	*4*	*3*	0					
6	*5*	*4*	0					

for $i = 1$ *to* n $\boxed{i=2}$

for $w = 0$ *to* W $\boxed{w=4}$

if $w_i <= w$ $\boxed{3 < 4}$

if $b_i + V[i-1,w-w_i] > V[i-1,w]$ $\boxed{4}$ + $\boxed{V[1, 1]=0}$ > $\boxed{V[2, 3]=0}$

$V[i,w] = b_i + V[i-1,w- w_i]$

else

$V[i,w] = V[i-1,w]$

else $V[i,w] = V[i-1,w]$

Can the item with weight=3 fit in if knapsack capacity is 4? Yes!

At capacity =4, and weight of the item =3, the remaining weight that we can add is = 1

At weight capacity = 1, no item may be added so benefit = 0

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	
5	4	3	0					
6	5	4	0					

for i = 1 to n $\boxed{i=2}$

    for w = 0 to W $\boxed{w=4}$

        if $w_i <= w$ $\boxed{3 < 4}$

            if $b_i + V[i\text{-}1, w\text{-}w_i] > V[i\text{-}1, w]$

                $V[i,w] = b_i + V[i\text{-}1, w\text{-}w_i]$ $\boxed{4}$ + $\boxed{V[1, 1]=0}$

            else

                $V[i,w] = V[i\text{-}1, w]$

        else $V[i,w] = V[i\text{-}1, w]$

Can the item with weight=3 fit in if knapsack capacity is 4? Yes!

At capacity =5, and weight of the item =3, the remaining weight that we can add is = 2

At weight capacity = 2, item 1 is already added with a benefit = 3, so we add it to benefit of item 4 = 4

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	
5	4	3	0					
6	5	4	0					

$for\ i = 1\ to\ n$ $\boxed{i=2}$

$\quad for\ w = 0\ to\ W$ $\boxed{w=5}$

$\quad\quad if\ w_i <= w$ $\boxed{3 < 5}$

$\quad\quad\quad if\ b_i + V[i-1,w-w_i] > V[i-1,w]$ $\boxed{4}$ + $\boxed{V[1, 2]=3}$ > $\boxed{V[2, 3]=0}$

$\quad\quad\quad\quad V[i,w] = b_i + V[i-1,w- w_i]$

$\quad\quad\quad else$

$\quad\quad\quad\quad V[i,w] = V[i-1,w]$

$\quad\quad else\ V[i,w] = V[i-1,w]$

Can the item with weight=3 fit in if knapsack capacity is 4? Yes!

At capacity =5, and weight of the item =3, the remaining weight that we can add is = 2

At weight capacity = 2, item 1 is already added with a benefit = 3, so we add it to benefit of item 4 = 4

Total benefit now is 7.

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0					
6	5	4	0					

for i = 1 to n $\boxed{i=2}$

for w = 0 to W $\boxed{w=5}$

if $w_i <= w$ $\boxed{3 < 5}$

if $b_i + V[i-1,w-w_i] > V[i-1,w]$

$V[i,w] = b_i + V[i-1,w- w_i]$ $\boxed{4}$ + $\boxed{V[1, 2]=3}$

else

$V[i,w] = V[i-1,w]$

else $V[i,w] = V[i-1,w]$

Can the item with weight=4 fit in if knapsack capacity is 1? No!

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0				
6	5	4	0					

for i = 1 to n $i=3$

for w = 0 to W $w=1$

if $w_i <= w$ $4 > 1$

if $b_i + V[i-1,w-w_i] > V[i-1,w]$

$V[i,w] = b_i + V[i-1,w- w_i]$

else

$V[i,w] = V[i-1,w]$

else $V[i,w] = V[i-1,w]$ $V[3,1]=V[2,1]$

Can the item with weight=4 fit in if knapsack capacity is 2? No!

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3			
6	5	4	0					

for i = 1 to n $i=3$

  for w = 0 to W $w=2$

    if $w_i <= w$ $4 > 2$

      if $b_i + V[i-1,w-w_i] > V[i-1,w]$

        $V[i,w] = b_i + V[i-1,w- w_i]$

      else

        $V[i,w] = V[i-1,w]$

    else $V[i,w] = V[i-1,w]$ $V[3,2]=V[2,2]$

Can the item with weight=4 fit in if knapsack capacity is 2? No!

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4		
6	5	4	0					

for i = 1 to n $i=3$

for w = 0 to W $w=3$

if $w_i <= w$ $4 > 3$

if $b_i + V[i-1,w-w_i] > V[i-1,w]$

$V[i,w] = b_i + V[i-1,w- w_i]$

else

$V[i,w] = V[i-1,w]$

else $V[i,w] = V[i-1,w]$ $V[3,3]=V[2,3]$

Can the item with weight=4 fit in if knapsack capacity is 4? Yes!

At capacity =4, and weight of the item =4, the remaining weight that we can add is = 0

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4		
6	5	4	0					

for i = 1 to n $\boxed{i=3}$

for w = 0 to W $\boxed{w=4}$

if $w_i <= w$ $\boxed{4 = 4}$

if $b_i + V[i-1,w-w_i] > V[i-1,w]$ $\boxed{5}$ + $\boxed{V[2, 0]=0}$ > $\boxed{V[2, 4]=0}$

$V[i,w] = b_i + V[i-1,w- w_i]$

else

$V[i,w] = V[i-1,w]$

else $V[i,w] = V[i-1,w]$

Can the item with weight=4 fit in if knapsack capacity is 4? Yes!

At capacity =4, and weight of the item =4, the remaining weight that we can add is = 0

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	
6	5	4	0					

for $i = 1$ *to* n $i=3$

    for $w = 0$ *to* W $w=4$

        if $w_i <= w$ $4 = 4$

            if $b_i + V[i-1,w-w_i] > V[i-1,w]$

                $V[i,w] = b_i + V[i-1,w- w_i]$ 5 $+$ $V[2, 0]=0$

            else

                $V[i,w] = V[i-1,w]$

        else $V[i,w] = V[i-1,w]$

Can the item with weight=4 fit in if knapsack capacity is 5? Yes!

At capacity =5, and weight of the item =5, the remaining weight that we can add is = 1.

At capacity 1, benefit is 0 as well.

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	
6	5	4	0					

```
for i = 1 to n                              i=3
    for w = 0 to W                          w=5
        if w_i <= w                         4 < 5
            if b_i + V[i-1,w-w_i] > V[i-1,w]
                V[i,w] = b_i + V[i-1,w- w_i]
            else
                V[i,w] = V[i-1,w]
        else V[i,w] = V[i-1,w]
```

5 + V[2, 1]=0 < V[2, 5]=0

Can the item with weight=4 fit in if knapsack capacity is 5? Yes!

At capacity =5, and weight of the item =5, the remaining weight that we can add is = 0.

At capacity 0, benefit is 0 as well.

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	7
6	5	4	0					

for i = 1 to n $\boxed{i=3}$

    for w = 0 to W $\boxed{w=5}$

        if $w_i <= w$ $\boxed{4 < 5}$

            if $b_i + V[i\text{-}1,w\text{-}w_i] > V[i\text{-}1,w]$

                $V[i,w] = b_i + V[i\text{-}1,w\text{-}\ w_i]$

            else

                $V[i,w] = V[i\text{-}1,w]$ $\boxed{V[3,\ 5]=V[2,\ 5]}$

        else $V[i,w] = V[i\text{-}1,w]$

Can the item with weight=5 fit in if knapsack capacity is 1? No!

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	7
6	5	4	0	0				

for i = 1 to n $i=4$

for w = 0 to W $w=1$

if $w_i <= w$ $5 > 1$

if $b_i + V[i-1,w-w_i] > V[i-1,w]$

$V[i,w] = b_i + V[i-1,w- w_i]$

else

$V[i,w] = V[i-1,w]$

else $V[i,w] = V[i-1,w]$ $V[4, 1]=V[3, 1]$

Can the item with weight=5 fit in if knapsack capacity is 2? No!

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	7
6	5	4	0	0	3			

for i = 1 to n $i=5$

for w = 0 to W $w=2$

if $w_i <= w$ $5 > 2$

if $b_i + V[i-1,w-w_i] > V[i-1,w]$

$V[i,w] = b_i + V[i-1,w- w_i]$

else

$V[i,w] = V[i-1,w]$

else $V[i,w] = V[i-1,w]$ $V[4, 2]=V[3, 2]$

Can the item with weight=5 fit in if knapsack capacity is 3? No!

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	7
6	5	4	0	0	3	4		

for i = 1 to n $\boxed{i=5}$

for w = 0 to W $\boxed{w=3}$

if $w_i <= w$ $\boxed{5 > 3}$

if $b_i + V[i-1,w-w_i] > V[i-1,w]$

$V[i,w] = b_i + V[i-1,w- w_i]$

else

$V[i,w] = V[i-1,w]$

else $V[i,w] = V[i-1,w]$ $\boxed{V[4, 3]=V[3, 3]}$

Can the item with weight=5 fit in if knapsack capacity is 4? No!

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	7
6	5	4	0	0	3	4	5	

for i = 1 to n $\boxed{i=5}$

for w = 0 to W $\boxed{w=4}$

if $w_i <= w$ $\boxed{5 > 4}$

if $b_i + V[i-1,w-w_i] > V[i-1,w]$

$V[i,w] = b_i + V[i-1,w- w_i]$

else

$V[i,w] = V[i-1,w]$

else $V[i,w] = V[i-1,w]$ $\boxed{V[4, 4]=V[3, 4]}$

Can the item with weight=5 fit in if knapsack capacity is 5? Yes!

At capacity =5, and weight of the item =5, the remaining weight that we can add is = 0

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	7
6	5	4	0	0	3	4	5	

for $i = 1$ *to* n $i=4$
    for $w = 0$ *to* W $w=5$
        if $w_i <= w$ $5 = 5$
            if $b_i + V[i-1,w-w_i] > V[i-1,w]$
                $V[i,w] = b_i + V[i-1,w- w_i]$
            else
                $V[i,w] = V[i-1,w]$
        else $V[i,w] = V[i-1,w]$

$$6 + V[3, 0]=0 < V[3, 5]=7$$

Can the item with weight=5 fit in if knapsack capacity is 5? Yes!

At capacity =5, and weight of the item =5, the remaining weight that we can add is = 0

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	7
6	5	4	0	0	3	4	5	7

for i = 1 to n $i=4$

  for w = 0 to W $w=5$

    if $w_i <= w$ $5 = 5$

      if $b_i + V[i-1,w-w_i] > V[i-1,w]$

        $V[i,w] = b_i + V[i-1,w- w_i]$

      else

        $V[i,w] = V[i-1,w]$ $V[4,5]=V[3, 5]$

    else $V[i,w] = V[i-1,w]$

7 is the maximum benefit that we can have given the included items.

Now, how to find the actual items?

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	7
6	5	4	0	0	3	4	5	7

for i = 1 to n

  for w = 0 to W

    if $w_i <= w$

      if $b_i + V[i-1,w-w_i] > V[i-1,w]$

        $V[i,w] = b_i + V[i-1,w- w_i]$

      else

        $V[i,w] = V[i-1,w]$

    else $V[i,w] = V[i-1,w]$

All of the information we need is in the table. V[n,W] is the maximal value of items that can be placed in the Knapsack.

i=4 *k=5*

Let i=n and k=W

if $V[i,k] \neq V[i-1,k]$ *then*

mark the i^{th} *item as in the knapsack*

$i = i-1,\ k = k-w_i$

else

$i = i-1$ *// Assume the* i^{th} *item is* *not* *in the knapsack*

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	7
6	5	4	0	0	3	4	5	7

All of the information we need is in the table.
V[n,W] is the maximal value of items that can be placed in the Knapsack.

i=4 *k=5*

Let i=n and k=W

if V[i,k] ≠ V[i–1,k] then *V[4, 5] == V[3, 5]*

mark the i^{th} item as in the knapsack

$i = i–1$, $k = k-w_i$

else

$i = i–1$ *i=4-1=3*

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	7
6	5	4	0	0	3	4	5	7

All of the information we need is in the table.
V[n,W] is the maximal value of items that can be placed in the Knapsack.

$i=3$ $k=5$

Let i=n and k=W

if $V[i,k] \neq V[i-1,k]$ *then* $V[3, 5] == V[2, 5]$

mark the i^{th} *item as in the knapsack*

$i = i-1,\ k = k-w_i$

else

$i = i-1$ $i=2$

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	7
6	5	4	0	0	3	4	5	7

All of the information we need is in the table.
V[n,W] is the maximal value of items that can be placed in the Knapsack.

$i=2$ $k=5$

Let $i=n$ *and* $k=W$

if $V[i,k] \neq V[i-1,k]$ *then* — $V[2, 5] \mathrel{!=} V[1, 5]$

mark the i^{th} *item as in the knapsack*

$i = i-1,\ k = k-w_i$

else

$i = i-1$

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	7
6	5	4	0	0	3	4	5	7

All of the information we need is in the table.
$V[n,W]$ is the maximal value of items that can be placed in the Knapsack.

Let $i=n$ and $k=W$

if $V[i,k] \neq V[i-1,k]$ then $V[2, 5] \mathrel{!=} V[1, 5]$

mark the i^{th} item as in the knapsack

$i = i-1,\ k = k-w_i$ $i=1$ $k=5-3=2$

else

$i = i-1$

b_i	w_i		0	1	2	3	4	5
		0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	7
6	5	4	0	0	3	4	5	7

All of the information we need is in the table.
$V[n,W]$ is the maximal value of items that can be placed in the Knapsack.

Let $i=n$ and $k=W$

if $V[i,k] \neq V[i-1,k]$ then

mark the i^{th} item as in the knapsack

$i = i-1$, $k = k-w_i$ $i=0$ $k=2-2=0$

else

$i = i-1$

			0	1	2	3	4	5
b_i	w_i	0	0	0	0	0	0	0
3	2	1	0	0	3	3	3	3
4	3	2	0	0	3	4	4	7
5	4	3	0	0	3	4	5	7
6	5	4	0	0	3	4	5	7

Example 2

	Item1	Item2	item3	item4
Value	5	3	4	7
Weight	3	2	1	4

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0													
2	0													
3	0													
4	0													

References

- Dasgupta, C.H. et al, Algorithms, 2006
- MITOpenCourseware
- Slides adapted from Arup Guha's Computer Science II Lecture notes: http://www.cs.ucf.edu/~dmarino/ucf/cop3503/lectures/