Homework 3a Solutions
CSI 503 Spring 2021

Rahul Bhenjalia

March 15 2021

1 Problems

1. Every semester you take a set of courses among the remaining courses in your program of study. Suppose that for each remaining course i, c_i and v_i indicate its number of credits and how valuable the course is to your learning goals, respectively. Also, assume that your university only allows you to take up to the total of M credits each semester.

   (a) Using dynamic programming, write a recursive top-down (with memorization) algorithm BestValuePlan(c, v, n, M) that returns maximum total value of courses that you can choose for a semester among the n remaining courses. c and v are arrays of credits and values, respectively. You can define and use auxiliary functions as needed.

```
BestValuePlan(c, v, n, M):
   if n=0 or M=0 then
      return 0
   if c[n-1] <= M then
      return MAXIMUM(v[n-1]+BestValuePlan(c,v,n-1,M-c[n-1]),
             BestValuePlan(c,v,n-1,M))
   else
      return BestValuePlan(c,v,n-1,M)
```

   (b) Write a bottom-up (non-recursive) algorithm for the same solution.

```
BestValuePlan(c,v,n,M):
    for i=0 to (M+1) do
        for j=0 to (n+1) do
            table[i,j]=0
for e=0 to (n+1) do
    for w=0 to (M+1) do
    if i=0 OR w=0 then
   table[i,w]=0
```

```
else if c[i-1]<=w then
                table[i,w] = MAXIMUM(v[i-1]+table[i-1,w-c[i-
1]],table[i-1,w])
            else
                table[i,w] = table[i-1,w]
    return table[n,M]
```

(c) Extend the algorithm in the previous step to print the actual solution (set of courses).

```
BestValuePlan(c,v,n,M):
for i=0 to (M+1) do
    for j=0 to (n+1) do
    table[i,j]=0

for e=0 to (n+1) do
    for w=0 to (M+1) do
    if i=0 OR w=0 then
   table[i,w]=0
    else if c[i-1]<=w then
                table[i,w]=MAXIMUM(v[i-1]+table[i-1,w- c[i-
1]], table[i-1, w])
            else
                table[i,w]=table[i-1,w]
    s=table[n,M]
print(s)
//Printing set of courses
u=M
for i=n down-to 0 do
if s<=0 then
break
if s=table[i-1,u] then
            break
        else
            print(c[i-1])
            s=s-v[i-1]
            u=u-c[i-1]
```

(d) Draw the subproblem graph for your solution, and analyze its time complexity accordingly.

Let's take an example for drawing a tree

.

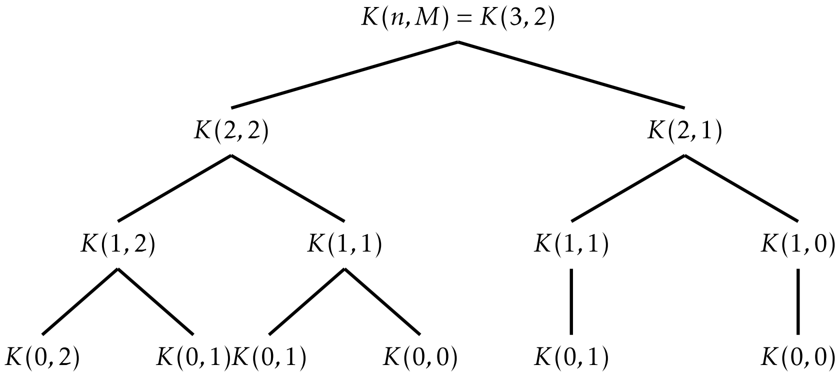
Credits of each course, $c = [1,1,1]$

.

Value of each course, $c = [10,20,30]$

.

Maximum credits you can take $M = 2$

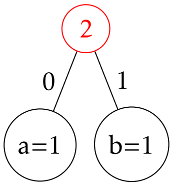


.

Considering the above algorithm in part(c), each sub-problem is redundant, the time-complexity of the algorithm is $O(2^n)$

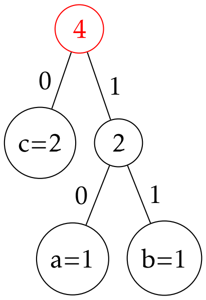
2. Make sure that you have read and understood the Huffman coding in Section 16.3 in CLRS.

   (a) Use algorithm Huffman() on Page 431 of CLRS to write an optimal Huffman code for the following set of frequencies, based on the first 8 Fibonacci numbers (a: 1, b: 1, c: 2, d: 3, e: 5, f: 8, g: 13, h: 21). Show a step-by-step construction of the tree.



.

.

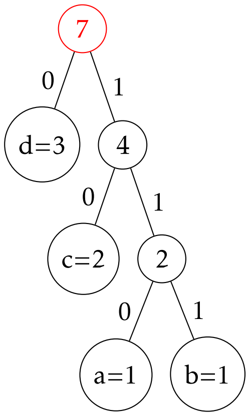


.

.

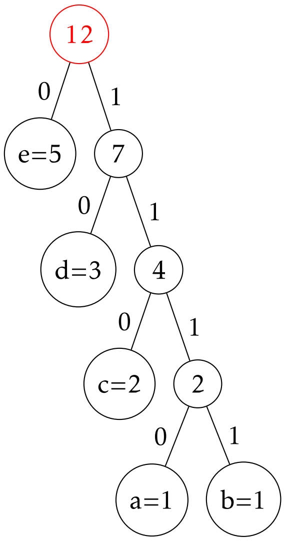
.

.


7
0
1
d=3
4
0
1
c=2
2
0
1
a=1
b=1

.

.

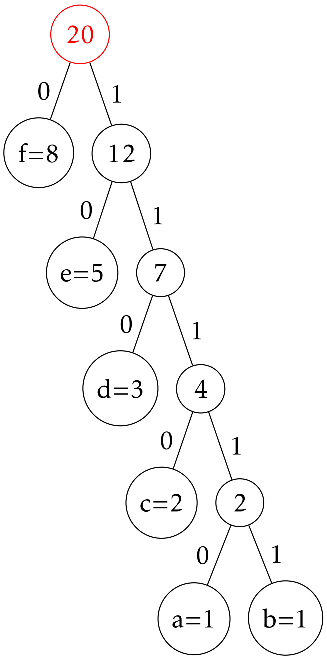

12
0
1
e=5
7
0
1
d=3
4
0
1
c=2
2
0
1
a=1
b=1

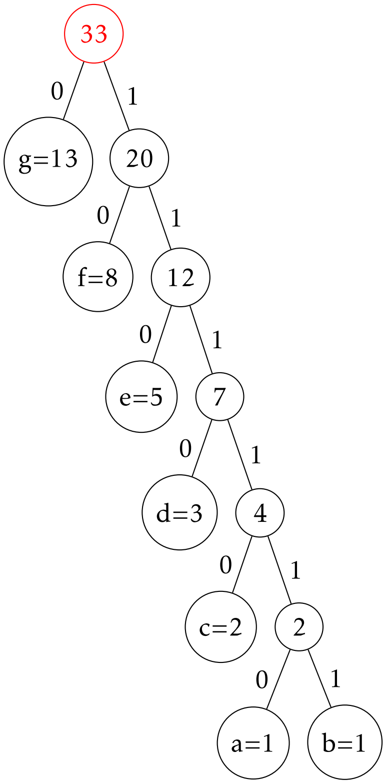
.

.

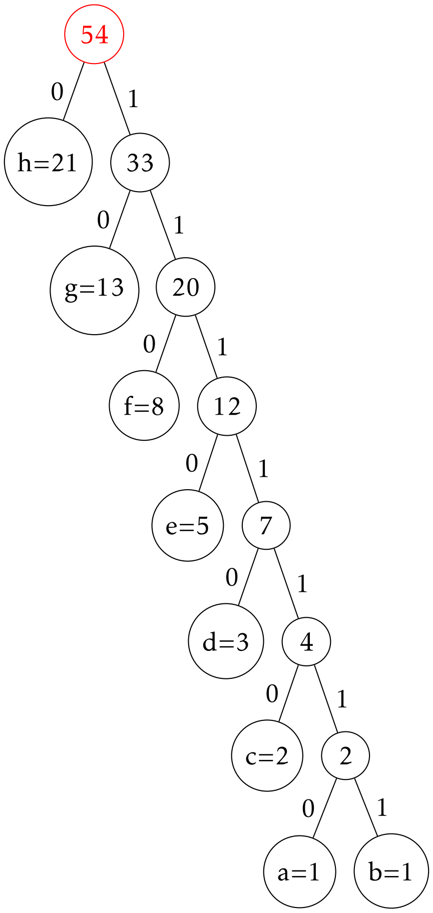
.

.


20
0
1
f=8
12
0
1
e=5
7
0
1
d=3
4
0
1
c=2
2
0
1
a=1
b=1


33
0
1
g=13
20
0
1
f=8
12
0
1
e=5
7
0
1
d=3
4
0
1
c=2
2
0
1
a=1
b=1

.



.

.

.

.

Huffman codes are as follows:

.

$a = 1111111$
$b = 1111110$
$c = 111110$
$d = 11110$
$e = 1110$
$f = 110$
$g = 10$
$h = 0$

.

.

.

(b) Generalize your answer to find the optimal code when the frequencies are the first n

Fibonacci numbers. Justify your answer.

.

As mentioned, the tree is just a long limb with n leaves just hanging. The recurrence in Fibonacci series implies:
$F_{n+2} = \sum_{i=0}^{n} F_i + 1$

.

As mentioned, the tree is just a long limb with n leaves just hanging. The recurrence in Fibonacci series implies:
$F_{n+2} = \sum_{i=0}^{n} F_i + 1$

We can prove the given equation but INDUCTION.

.

For base case: $1, 1, 2, 3$ provide sufficient result.

.

We assume that the equality holds for all Fibonacci numbers smaller than F_{n+2}

.

Hence we prove correctness for F_{n+2}:

.

$F_{n+2} = F_{n+1} + F_n = \sum_{i=0}^{n-1} F_i + 1 + F_n = \sum_{i=0}^{n} F_i + 1$

.

.

We can conclude that $F_{n+2} < \sum_{i=0}^{n-1} F_i + 1$

.

Which clearly say $F_{n+2} < F_{n+1}$

.

Hence, our generalized equation holds.