Review Heap sort

Full- A binary tree is **full** if each node has 0 or 2 child nodes
Complete – A **full** binary tree with all leaf nodes at the same level

What is a Heap?
What is heap Property?

Is the sequence (23,17,14,6,13,10,1,5,7,12) a max heap? Show it.

In a heap A[1…n]
What are **children** of A[i] ?
What is parent of A[i] ?

If n is the number of nodes in a heap of height h

What is the Min # of nodes ?
What is the Max # of nodes ?

What is the exact value of h in terms of n?
A complete binary tree with n nodes has height h given by

$h=\lfloor \lg n \rfloor$ used in time to adjust nodes in Heapify algorithm

What is the Min number of **leaf** nodes?
What is the Max number of **leaf** nodes?

In a heap with n nodes has exactly $\lfloor n/2 \rfloor$ internal nodes, $\lceil n/2 \rceil$ leaves (external)

$\lfloor n/2 \rfloor$ internal and $\lceil n/2 \rceil$ external

Show that with the array representation for storing n-element heap, the leaves are the nodes indexed by $\lfloor n/2 \rfloor+1$, $\lfloor n/2 \rfloor+2$, $\lfloor n/2 \rfloor+3$, …, n.

Write function for AdjustMaxHeap(A,k) with invariants?

Do

AdjustMaxHeap(A,3) on the array A =(27,17,3,16,13,10,1,5,7,12,4,8,9,0)

Build a maxHeap on the array A=(5,3,17,10,84,19,6,22,9)

Write function for Build MaxHeap with invariants

Prove that Complexity to Build MaxHeap: O(n)

Given: an array of size n
What is heap sort? How to do heap-sort?
Write the code with invariants.

1) Given the following recursive function to **insert an element x into the highest zero element** of an array `a`**:**

   What are the best, worst, and average time of Insert(x,a,n) using the recursion tree method. Verify your solutions using the substitution method.

2) Consider the recursive version of **Binary Search** that finds x in A[1..n] sorted ascending, if it exists and return its index. If x is not in A[1..n], the program returns 0. For simplicity, assume n is a power of 2.
   a. Develop a recurrence for the running time T(n) of BinarySearch using the recursion tree method.

   b. Solve the recurrence from part a. Verify it using the substitution method.

   c. Problem 4-3 – space complexity of parameter passing

3 a) Show the solution of following recurrence for n a power of 2:

$T(n)=2T(n/2) + n$	$n>1$
$T(1)=1$	otherwise

b) 4.2-1 (but assume n is a power of 2 so the "floor" function disappears)

$T(n)=3T(\text{floor}(n/2)) + n$	$n>1$

4) 4.2-4

   $T(n) = T(n-a) + T(a) + cn.$

5) Solve Problem 4-1 a, c, e, g show all work
   a. $T_n = 2T_{n/2} + n^3$
   c. $T_n = 16T_{n/4} + n^2$
   e. $T_n = 7T_{n/2} + n^2$
   g. $T_n = T_{n-1} + n$

6) Problem 4-4 a, h

   b. $T_n = 3T_{n/2} + n \lg n$
   h. $T_n = T_{n-1} + \lg n$ □ (n lg n)

Review
Chapter 15,16 Topics

- Example Problems

- $15.1 Car Scheduling-- **dynamic 'programming'**
- $15.4String matching: acd, adc -- **dynamic 'programming'**
- $16.1 Real-Time deadline schedules – **greedy algorithm**
- $16.2 0-1 Knapsack: 1,2,3,4 W=5 -- **dynamic 'programming'**
- $16.2 fractional Knapsack: 1,2,3,4 W=5 -- **greedy algorithm**

What is dynamic programming? Give examples?
What is Greedy Algorithm? Give examples?

What is principle of optimality?

Note. optimal value is unique, but optimal solution is not.

Formally state
0-1 KNAPSACK PROBLEM

Example 0-1 Knapsack , find optimal solution via Greedy and Dynamic approach

W = 5, and 4 weights and their values
Sorting the value/weight and taking them in that order may not optimize:

i	w_i	v_i	v_i / w_i
1	3	12	4
2	1	10	10
3	3	20	6.66
4	2	15	7.5

Example 0-1 Knapsack , find optimal solution via Greedy and Dynamic approach
W = 5, and 4 weights and their values

i	w_i	v_i	v_i / w_i
1	2	12	6
2	1	10	10
3	3	20	6.66
4	2	15	7.5

Dynamic Programming
Create **(n+1)x(W+1)** table.

Let c[i,j] be the optimal value of a sack of size j by using a subset of i items
c[i,j] = if i=0 or j=0
Recursive calls
c[i,j] values are calculated as follows:
c[i,j] = c[i-1,j] if $w_i>j$, -- w_i cannot be used
c[i,j] = max(v_i+ c[i-1,j- w_i], c[i-1,j]) otherwise -- whether w_i will lead to optimal value

apply it to the following example
Example: W =5,
$w_1 =1$, $w_2 = 2$, $w_3 = 3$,
$v_1 = 6$, $v_2 = 10$, $v_3 = 12$,

a)Weights ordered by value per unit weight {1, 2, 3}.

b) weights given in the order {2, 1, 3}.

Question? Do we need to fill the whole last row? Not necessarily? just c[n,m] needs to be computed.

Write iterative solution with invariants

Trace path: we are interested in determining which weights are included.
Give criteria to trace path and show that the complexity is: in Θ(n) time

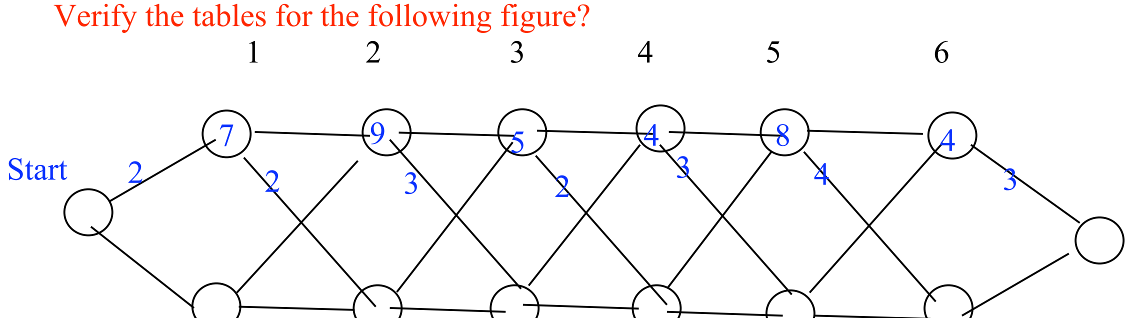
Describe the LCS problem?

What is the Recursive algorithm worst complexity.
Give iterative solution with invariants.

Show that
1. GRAMPRO and PROGRAM: many common subsequences here, LCS is GRAM
2. 2613564 and 5642613: many common subsequences here, LCS is 2613
3. LCS of 1232412 and 243121: 2412, 2312, 2321
4. LCS: AGCGA and CAGATAGAG is AGGA

How determine the actual path determine a path from the optimal length?

In the Car assembly problem with two assembly lines,
Verify the tables for the following figure?



Complete

4 2 1 1 2 1

8 5 6 5 4 7 2

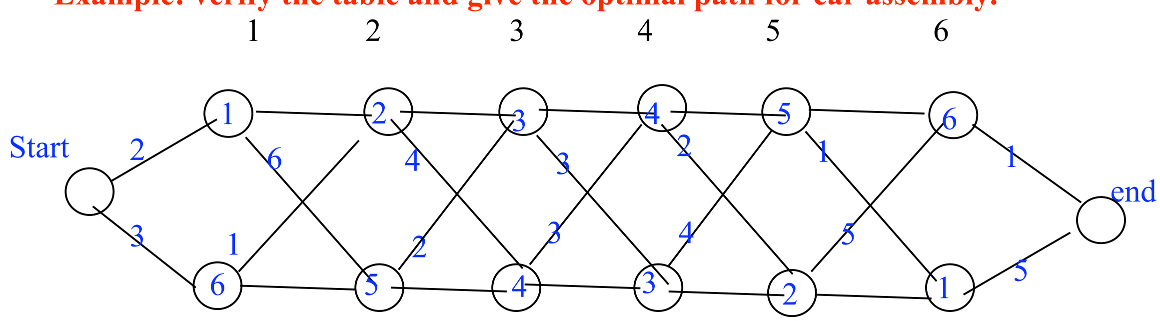
j	1	2	3	4	5	6	exit
$f_1[j]$	9	18	22	26	34	35	5
$f_2[j]$	12	16	22	27	31	38	1

f* = 38, l*=2 – last station is $S_{2,6}$

j	2	3	4	5	6
$l_1[j]$	1	2	1	1	2
$l_2[j]$	1	2	2	2	2

Give the optimal path for car assembly.

Example: verify the table and give the optimal path for car assembly.

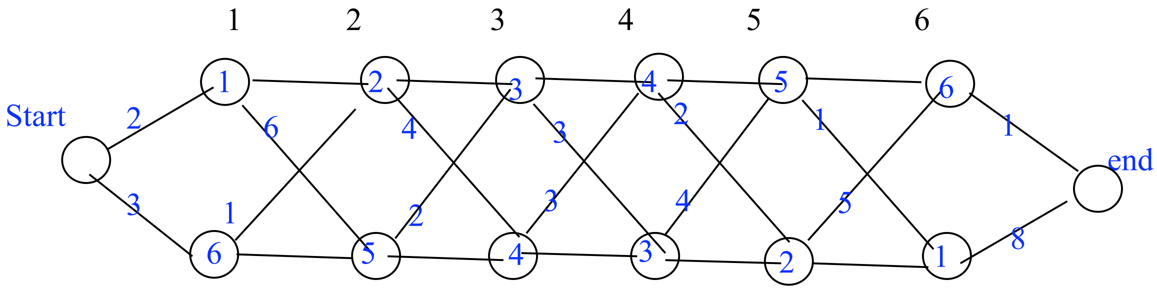


j	1	2	3	4	5	6	exit
$f_1[j]$	3	5	8	12	17	23	1
$f_2[j]$	9	14	13	14	16	17	5

f* = 22, l*=2 – last station is $S_{2,6}$
The sequence of stations is $S_{1,1}$, $S_{1,2}$, $S_{1,3}$, $S_{2,4}$, $S_{2,5}$, $S_{2,6}$

j	2	3	4	5	6
$l_1[j]$	1	1	1	1	1
$l_2[j]$	1,2	1	1	1,2	2

Example same as above with slight change at the end



j	1	2	3	4	5	6	exit
$f_1[j]$	3	5	8	12	17	23	1
$f_2[j]$	9	14	13	14	16	17	8

$f^* = 24$, $l^*=1$ – last station is $S_{1,6}$
The sequence of stations is $S_{1,1}$, $S_{1,2}$, $S_{1,3}$, $S_{1,4}$, $S_{1,5}$, $S_{1,6}$

j	2	3	4	5	6
$l_1[j]$	1	1	1	1	1
$l_2[j]$	1,2	1	1	1,2	2

Is the optimal value unique?
Is the optimal solution unique?
Justify your answers with examples?