CS-204

END-SEMESTER

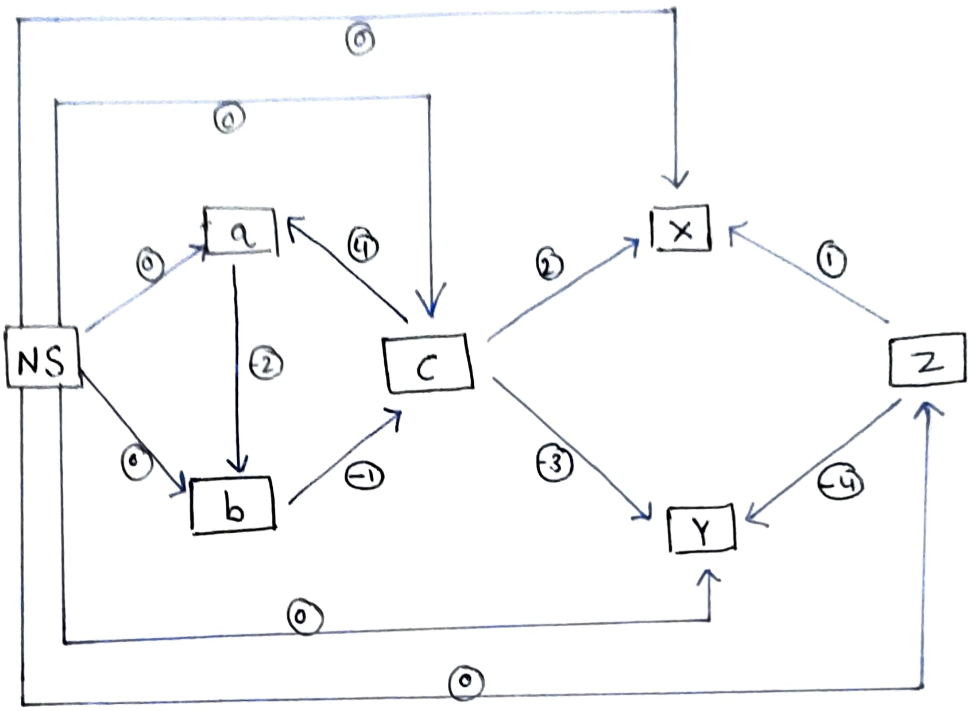
- Tarusi Mittal

- 1901CS65

- Tarusi Mittal

Tarusi Mittal
1901CS65

Que 1:- NS: New Source Vertex



→ Adding a new node NS to graph and adding 0 weight edges from NS to other edges.

→ Applying Bellman-Ford Algorithm to find the shortest paths (NS → vertex) simultaneously storing them. in vertex[ ]

(i) a: 0 x: -1

b: -2 y: -6

c: -3 z: 0

Now;

Let us take an array weight[ ] which represents the weight array of the edges.

Here;

weight [u,v] will represent the weight of edge $u \to v$

As we know;

New; weight [u,v] = weight [u,v] + vertex[u] - vertex [v]

(ii) $a \to b \Rightarrow (-2) + (0) - (-2) = \boxed{0}$

$c \to a \Rightarrow (4) + (-3) - (0) = \boxed{1}$

$b \to c \Rightarrow (-1) + (-2) - (-3) = \boxed{0}$

$c \to x \Rightarrow (2) + (-3) - (-1) = \boxed{0}$

$c \to y \Rightarrow (-3) + (-3) - (-6) = \boxed{0}$

$z \to x \Rightarrow (1) + (0) + (-1) = \boxed{2}$

$z \to y \Rightarrow (-4) + (0) - (-6) = \boxed{2}$

Que 2:- No. of elements in the array = 200 (even)

We first initialise two variables max. and min.

1. Compare arr[1] and arr [2] and store them in min and max accordingly $\longrightarrow$ 1st comparison

2. Now, we will do comparisons for every pair of two consequtive numbers.

For each pair:-

a) compare arr[i] and arr [i+1]

b) find the maximum and minimum of the pair and compare them with max and min (3 comparisons)

Total no of comparisons:

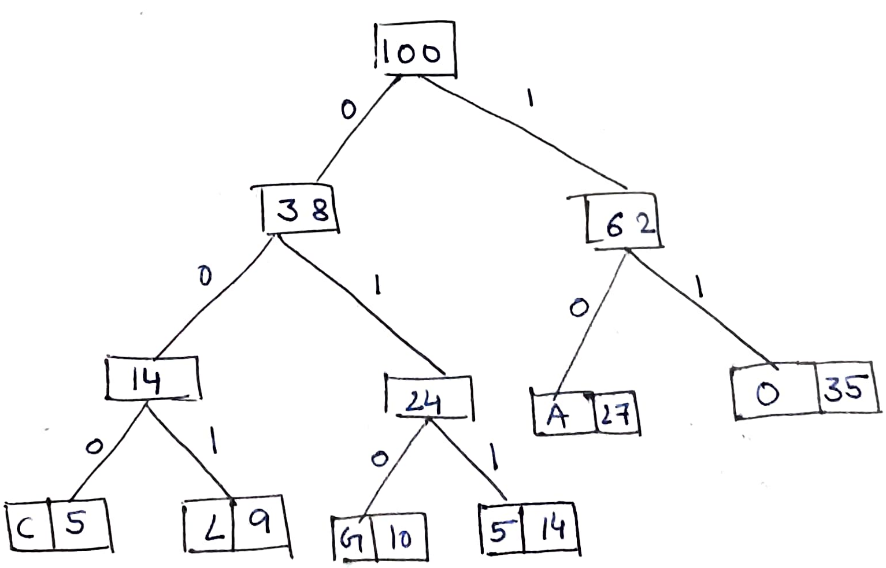
$$1 + 3\left(\frac{200}{2} - 2\right) = 298$$

(for every pair of nos three comparisons are made)

Total comparisons made are $\boxed{298}$.

Que 3: Total count of the letters:

A	- 27		A	— 10
L	- 9		L	— 001
G	- 10	$\longrightarrow$	G	— 010
O	- 35		O	— 11
C	- 5		C	— 000
S	- 14		S	— 011



No. of bits used for A = 27×2 = 54

L = 4×9 = 36

G = 3×10 = 30

O = 2×35 = 70

C = 4×5 = 20

S = 2×14 = 28

Total bits = = 238

Total bits used = $\boxed{238}$ for encoding

Que 4: The given array:

13, 19, 9, 5, 12, 8, 7, 4, 21, 2, 6, 11

low = 0

high = 12

pivot = arr [high − 1) = 11

initialyzing i = low − 1 = −1

for j = 0 to < high − 1

j=0	13	19	9	5	12	8	7	4	21	2	6	11	i = −1
j=1	13	19	9	5	12	8	7	4	21	2	6	11	i = −1
j=2	9	19	13	5	12	8	7	4	21	2	6	11	i = 0
j=3	9	5	13	19	12	8	7	4	21	2	6	11	i = 1
j=4	9	5	13	19	12	8	7	4	21	2	6	11	i = 1
j=5	9	5	8	19	12	13	7	4	21	2	6	11	i = 2
j=6	9	5	8	7	4	13	19	4	21	2	6	11	i = 3
j=7	9	5	8	7	4	13	19	12	21	2	6	11	i = 4
j=8	9	5	8	7	4	13	19	12	21	2	6	11	i = 4
j=9	9	5	8	7	4	2	19	12	21	13	6	11	i = 5
j=10	9	5	8	7	4	2	6	12	21	13	19	11	i = 6

Ans:

9	5	8	7	4	2	6	11	21	13	19	12

ie the array after 1st iteration of quick sort.

Que 5:- Tree edges: The edges that are part of dfs tree.

A tree having k edges has k+1 nodes.

Now, lets assume number of connected components are x

lets assume first component has k_1 tree edges, second k_2 and so on...

$\therefore k_1 + k_2 + k_3 + \cdots + k_x = k$

Summation of nodes of tree $= n$

$(k_1+1) + (k_2+1) + (k_3+1) + \cdots + (k_x+1) = n$

$(k_1 + k_2 + k_3 + \cdots + k_x) + x = n$

$k + x = n$

$\boxed{x = n - k}$

So, the number of connected components is $\boxed{n-k}$

Que 6: Let us say that there is a set 'S' with m elements that are of the size s1, s2, ..., sm.

As we know that the subset-sum problem says that with input set $(x_1, x_2, \ldots x_m)$ asks if any subset exsists, sum of whose elements is equal to a given no N.

So, the given problem is same as the subset-sum problem.

#1: Subset Sum is NP

Given a proposed set T, we need to test if $\sum_{i \in T} s_i = B$.
Adding up at most m numbers, each of size B takes $O(m \log B)$ time, linear in input size.

#2: 3 SAT is NP Complete

As SAT is an NP complete problem.
SAT $\propto$ 3 SAT
(corollary of SAT NP complete reduction)

Hence 3 SAT is NP complete.

#3: Subset Sum is NP Complete

It will be a reduction of 3SAT

Defining numbers x_i and $\overline{x_i}$ and a target B.
Such that one can take only one of x_i and $\overline{x_i}$.

Now, lets say we have vectors instead of numbers.
Two vectors can be added component wise. Now we have to see whether there is a subset whose sum equals a specified vector.

Now, lets ~~tady~~ input of 3SAT be ϕ by having n clauses and m variabls. The vectors will have length.

$n+m$:- whether first m positions specify the variable taken a the rest n positions records the clauses each litoral is

Let ϕ =

$$(x_2 \vee x_3 \vee \overline{x_4}) \wedge (x_1 \vee \overline{x_3} \vee x_4) \wedge (x_1 \vee \overline{x_2} \vee x_4)$$

the vector $x_i \Rightarrow$

$x_1 = (1,0,0,0;\ 0,1,1)$ $\qquad$ $\overline{x_1} = (1,0,0,0;\ 0,0,0)$

$x_2 = (0,1,0,0;\ 1,0,0)$ $\qquad$ $\overline{x_2} = (0,1,0,0;\ 0,0,1)$

$x_3 = (0,0,1,0;\ 1,0,0)$ $\qquad$ $\overline{x_3} = (0,0,1,0;\ 0,1,0)$

$x_4 = (0,0,0,1;\ 0,1,1)$ $\qquad$ $\overline{x_4} = (0,0,0,1;\ 1,0,0)$

A target B of all 1's would force selection of exactly one of each variable and its negation.

How, some clauses, might have true, literal.

$B = (1,1,1,1;\ 3,3,3)$

Adding vectors x_i & $\bar{x}_i$, that can be used to round sum up to B.

ex. $x_1 = (0,0,0,0;\ 1,0,0)$ $\quad \bar{x}_i = (0,0,0,0;\ 1,0,0)$

To reach 3 in a component atleast 1 must be supplied by a literal.

<u>Thus</u>:

we have built a set of vectors and a target vector such that there is a subset of vectors, that sums to a target vector exactly when the boolean formula has a satisfying assignment.

Now, think of vectors just as a number in decimal like.

$B = 1111333$

$x_1 = 1000011$

$\bar{x}_1 = 1000000$; we just need $\bar{x}_1$ and x_1 which sum upto B.

Thus we have reduced 3 SAT problem to subset sum problem i.e 3SAT $\propto$ subset sum problem.

Thus subset sum problem is NP complete.