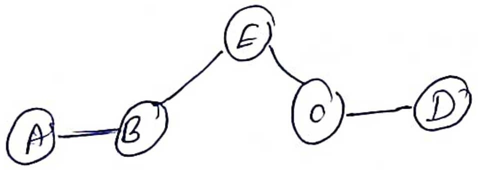
disjoint set ADT:

- → set u, set v
- → Merge u & v
- → detect cycle.

2 operations
- → find
- → union

Eg

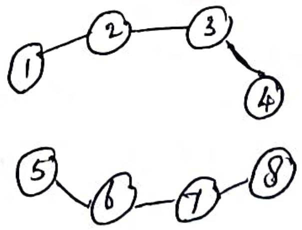
→ two different graph connect these two by using another vertex called 'E'



→ New vertex should not belongs to any set

Eg-1

$u = \{1, 2, 3, 4\}$
$v = \{5, 6, 7, 8\}$



→ find '5', its available in set 'v'
→ find '1' " " " set 'u'

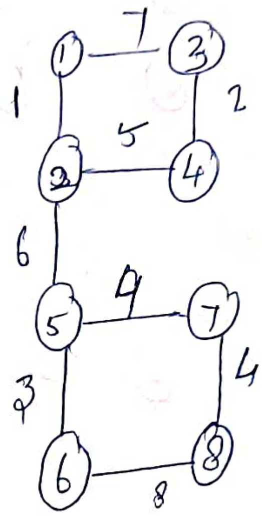
→ Union operation

$u \cap v = \emptyset$

$u \cup v = \{1, 2, 3, 4, 5, 6, 7, 8\}$ → Connect (4, 8) to perform union.

→ find (~~4~~, 8) = its available in one set, we can know that its cycle.

Eg., universal set $U = \{1, 2, 3, 4, 5, 6, 7, 8\}$



Consider edge weight & joint vertices, form set

$S_1 = \{1, 2\}$ weight - 1

Remove 1,2 from universal set

{~~1~~, ~~2~~, ~~3~~, ~~4~~, ~~5~~, ~~6~~, ~~7~~, ~~8~~}

$S_2 = \{3, 4\}$ - 2

$S_3 = \{5, 6\}$ - 3

$S_4 = \{7, 8\}$ - 4

→ Next weight '5' (2,4) its available in set 1 & 2, perform union

$S_1 \cup S_2 = \{1, 2, 3, 4\}$ - S_6

→ weight '6' (2,5), available in S_6, ~~S_2~~ & S_3

S_7: {1, 2, 3, 4, 5, 6}

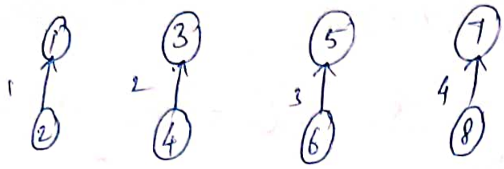
→ '7' (1, 3) if we connect this it will form cycle because its available in same set.

→ '8' {6, 8}, its there in different set perform union

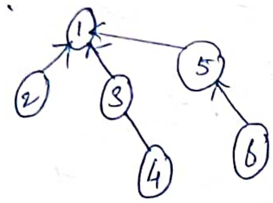
{1, 2, 3, 4, 5, 6, 7, 8}

→ '9' {5, 7} its in same set so it will form cycle.

→ sentinel node.



5



"Dynamic Equivalence Problem:

→ DEP essentially support 3 operations on set of elements.

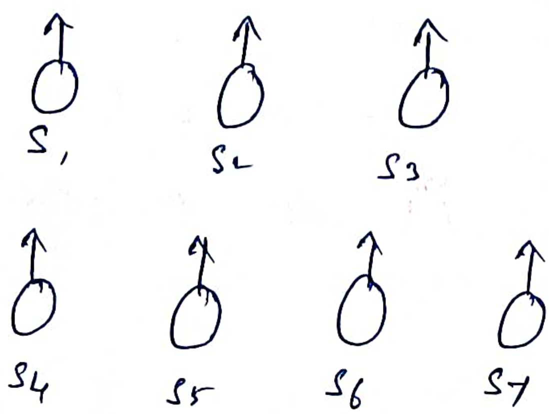
→ Equivalence relation is defined by 3 operations.

1. Make set
2. find
3. Union.

Makeset

→ This operation is used to create a set with element.

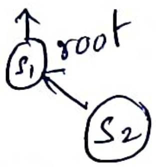
eg.,



Union:

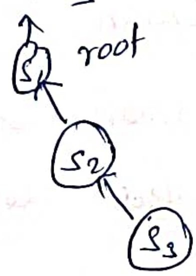
→ which merges two equivalence set, and create new set.

① $S_1 \cup S_2$



a

② a $\cup S_3$



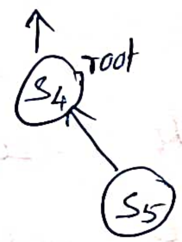
b

Note: Consider s_1 & s_2 having similar set of elements.

→ If the sets having different element, then it makes disjoint.

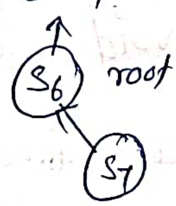
$S_1 \cup S_2 = \phi$

③ $S_4 \cup S_5$



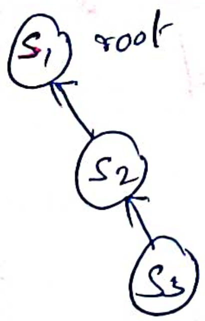
c

④ $S_6 \cup S_7$



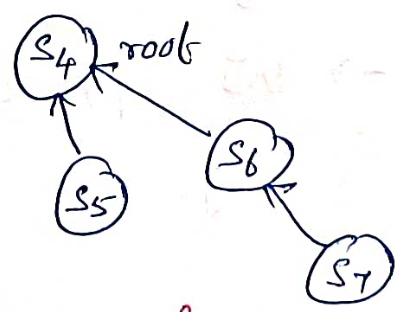
d

⑤ a ∪ b



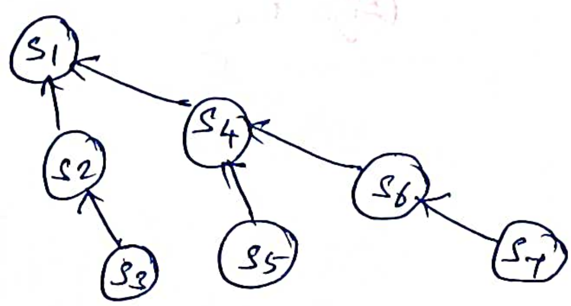
e

⑥ c ∪ d



f

⑦ e ∪ f



$\{S_1, S_2, S_3\}$ $\{S_1, S_4, S_5\}$ $\{S_1, S_4, S_6, S_7\}$

~~Quick find~~

→ Approaches to solve DEP:

① Quick find

② Quick union

↳ smart Union

→ Path Compression

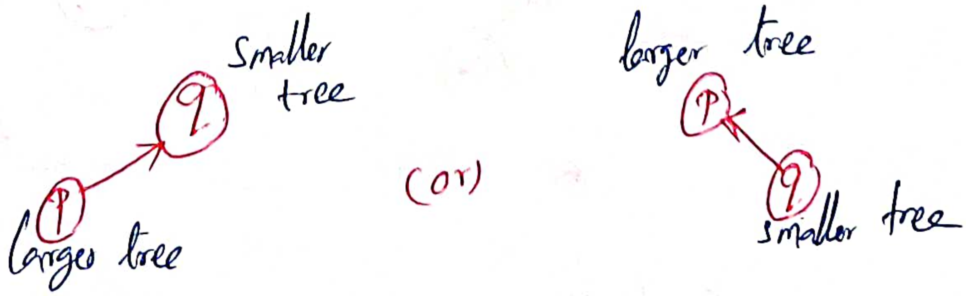
Smart Union: Algorithms:

→ Avoid tall trees.

→ Keep track of the size (i-e, no of objects) of each tree and always to link the root of smaller tree to root of larger tree, breaking ties by any method.

→ This approach is called Union by size.

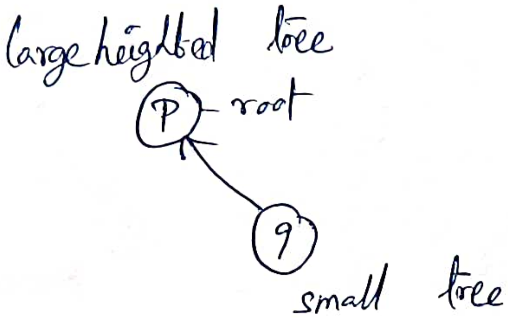
Eg.,



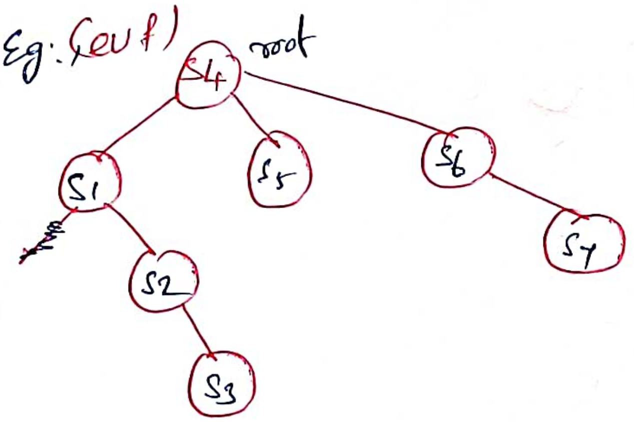
Union by height:

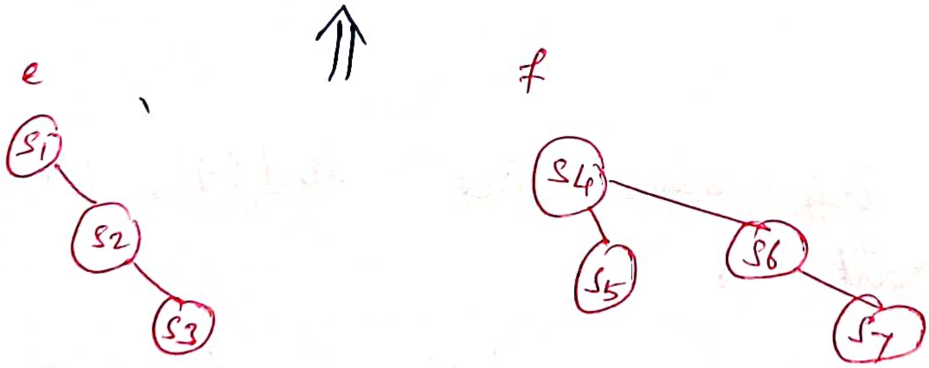
which tracks the height instead size

Eg.,



Eg: (e ∪ f)



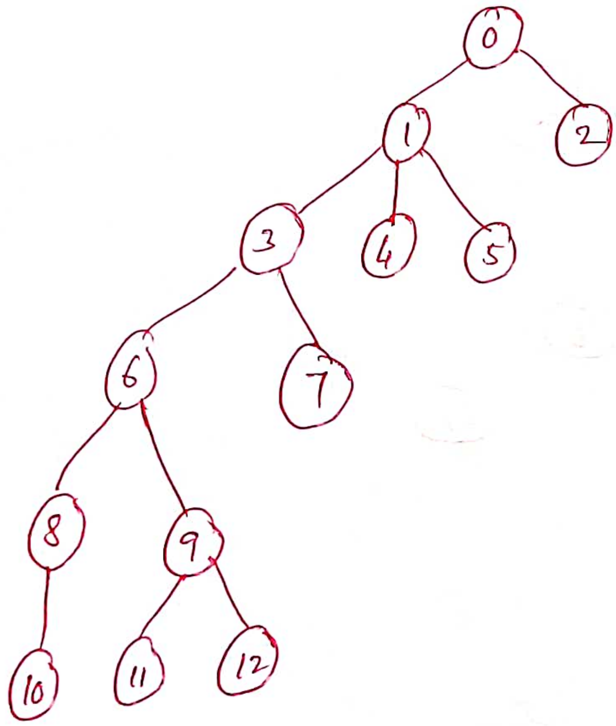


In this above sets e & f, e is having 3 sets & f is having 4 sets, so 'f' is having highest height when comparing with 'e', so S_4 in set f is considered as root, and rest of the tree is connected with root 's_4'.

Path Compression:

→ This operation is performed during find.

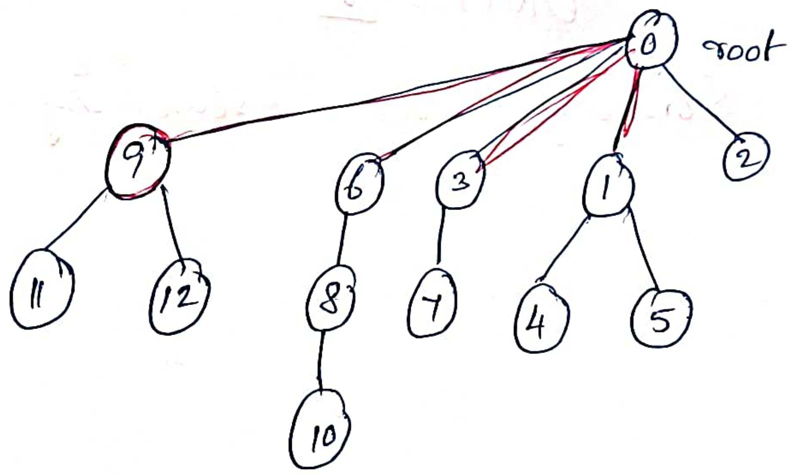
Eg →



→ In this above tree find (9), Perform (call) disjoint set

→ Path to reach '9' is {0, 1, 3, 6, 9}

→ all the nodes are connected directly to the root node, after finding '9', disconnected all the subtrees.


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