Algo :- (I) Broadcast. (To MST_Edge).

(II) If leaf node receives a broadcast.
↳ a) Compute MWOE
b) Start ConvergeCast.

(III) Internal Node : Receiving ⇒ ConvergeCast
a) Compute MWOE
b) Return the result to its parent.

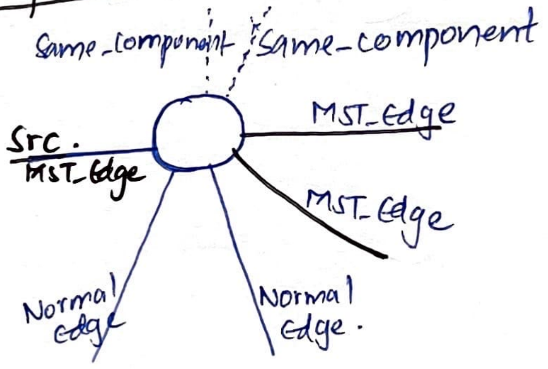
(IV) Leader on receiving convergecast msg, it computes MWOE and starts MERGE msg.

(V) A node on receiving MERGE msg,
a) returns EMPTY msg to source (acts as ACK).
b) forwards MERGE msg to all MST_Edge and normal Edge. (except minimum Edge to merge)
c) ~~Node~~ If merge msg ~~was~~ is addressed to this node. (Node starts START_MERGE msg)

(VI) If a node receives START_MERGE,
a) Checks if it is a special case **.
b) else. mark that Edge as MST_Edge.

Edge Type	Node Type
Normal_Edge	Root
MSTEdge	Leaf
Same_component.	Internal

compute MWOE

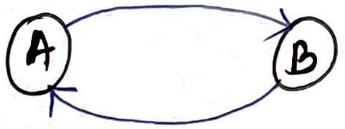


(I) Get leader of ~~the~~ all the nodes with normal edge.
↳ if leader matches :
⇒ in Same-component!

(II) Wait to get ~~Get~~ MWOE on each MST_Edge.

Then, from Normal-Edge (wt) and MWOE from ~~child~~ MST_Edge computes MWOE and returns in convergecast result.

Special case**



(VII) Node with higher UID in special case/edge would elects itself as a New_leader.

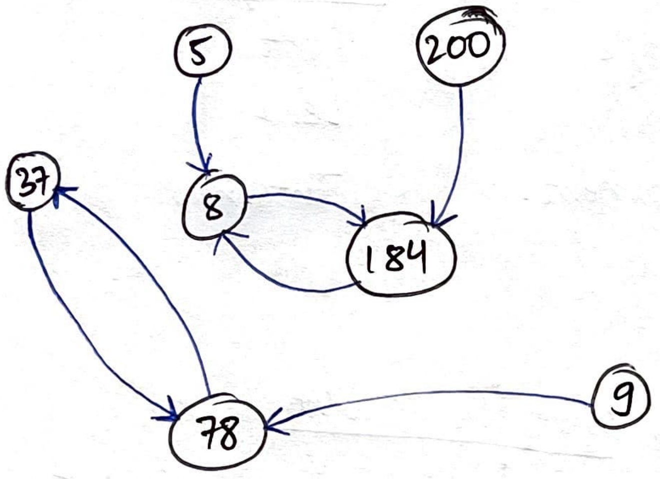
(VIII) New_ leader does a leader Broadcast.

(IX) waits for some time. i.e. sleep for some time
and again start from (I).

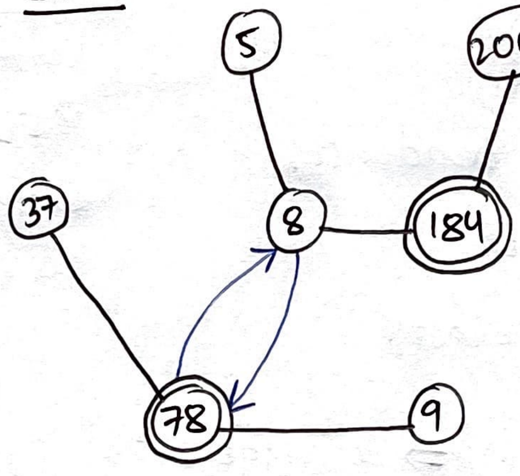
$\boxed{\text{Termination Condition}}$ → Leader if receives a null as a minEdge in converge cast.
i.e. if a leader finds that MWOE on this component
is NULL, then it starts a TERMINATION msg.

Node when receiving TERMINATION msg,
→ Prints Node Inform
→ Prints MST edge.
→ Forwards TERMINATON inform to all ~~sed~~ MST Edges.

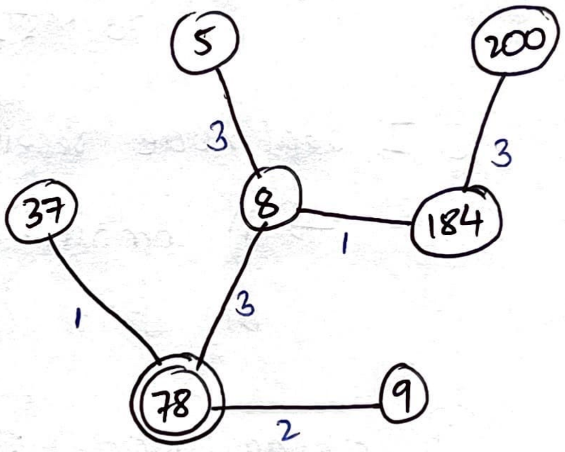
Round 0:



Round 1:



Round 2:


5
200
3
3
37
8
184
1
1
3
78
9
2