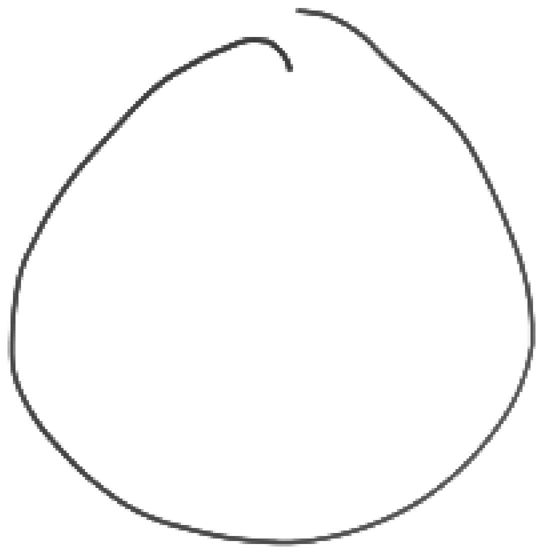
http://www.cs.uic.edu/~xiaorui/cs594

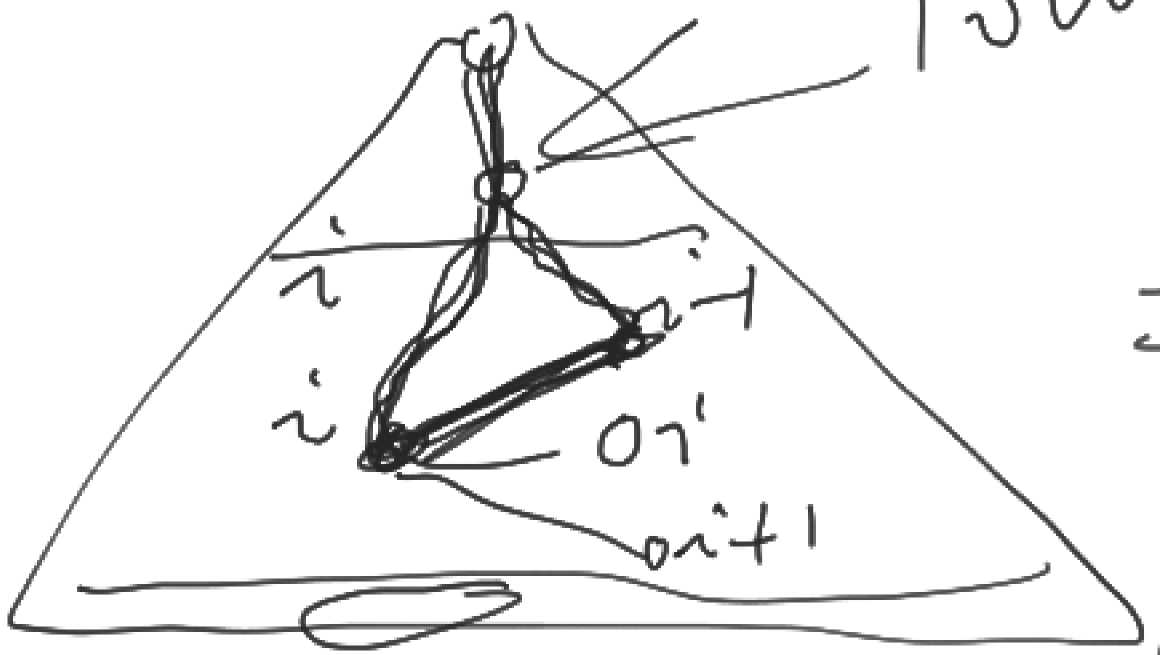
Tree (spanning tree), DFS, BFS

Claim: For a graph with minimum deg ≥ 3, there is a cycle of the graph of size $\leq O(\log n)$.

BFS tree:



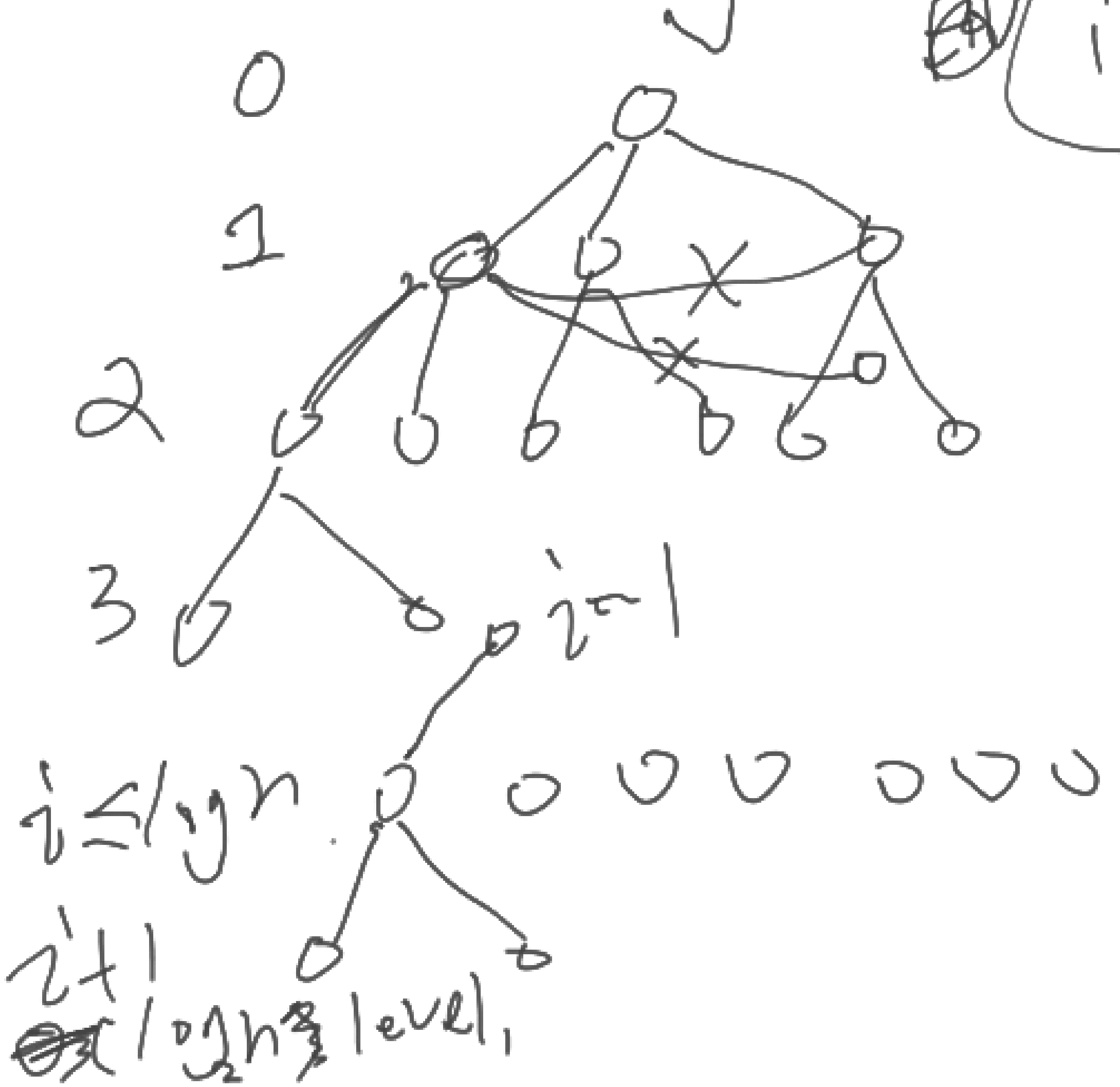
1. Edges connect vertices at same level or adjacent level.
2. If $\exists$ non-tree edge has one endpoint at level i, $\Rightarrow$ cycle of size $\leq 2i+2$



lowest common ancester

$\Rightarrow$ to show

$\exists$ a non-tree edge in BFS tree at level $O(\log n)$.



#vertices	level
1	0
≥ 2	1
≥ 4	2
$\vdots$	$\vdots$
$\geq 2^i$	i
$\geq 2^{\log n} = n$	$\log_2 n$

$\# \; 1+2+4 \cdots 2^{\log n}$

$\geq 2 \cdot n - 1$

contradiction, n vertices.

$\square$

Dynamic Graph connectivity.

- Input is large.
- Input changes. — small but frequent

assumption

Dynamic Algo

Maintain some data structure so that updates can be quickly incorporated.

o(n) time to handle update.

Example: sum.

input: n integers, $a_1, a_2 \cdots a_n$.

update: $a_i \longrightarrow a_i'$

goal: sum of $a_i = S$

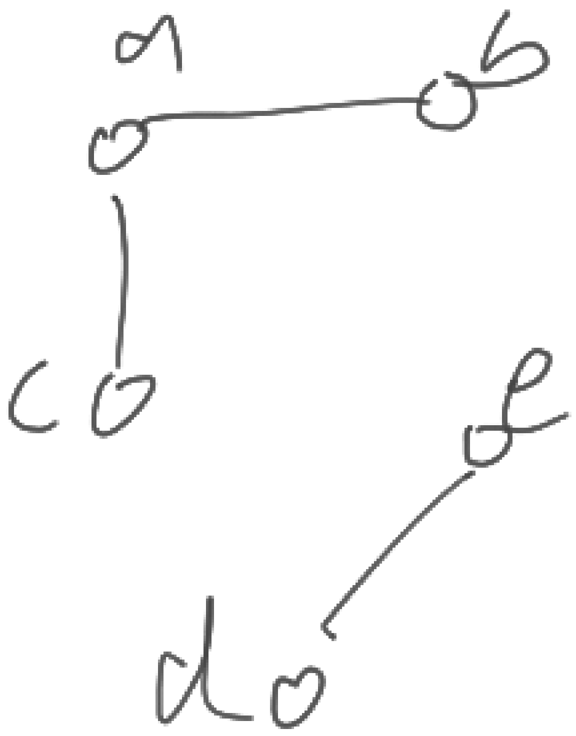
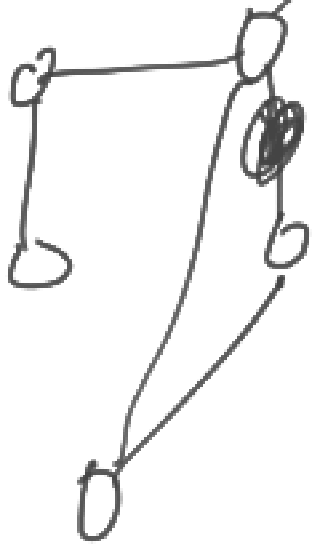
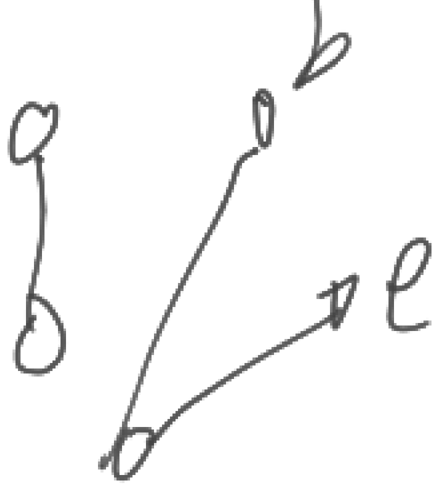
Claim: $O(1)$ time to maintain sum.

if $a_i \longrightarrow a_i'$

$S \longrightarrow \boxed{S + a_i' - a_i}$

Dynamic Connectivity $O(n)$.

Input: graph with n vertices.
Updates: add or delete an edge.
query: Is x connected to y or not?

Initial	query (a,e)	Add (b,d)	query (a,e)	Delete (a,b)	query (a,e)	query (b,e)
	No		Yes.		No	Yes

Thm: Dynamic connectivity can be solved in $O(\log n)$ time (randomized).

undirected graph G

deterministic $O(n^{0.5})$

Q: Which data structure is enough for connectivity?

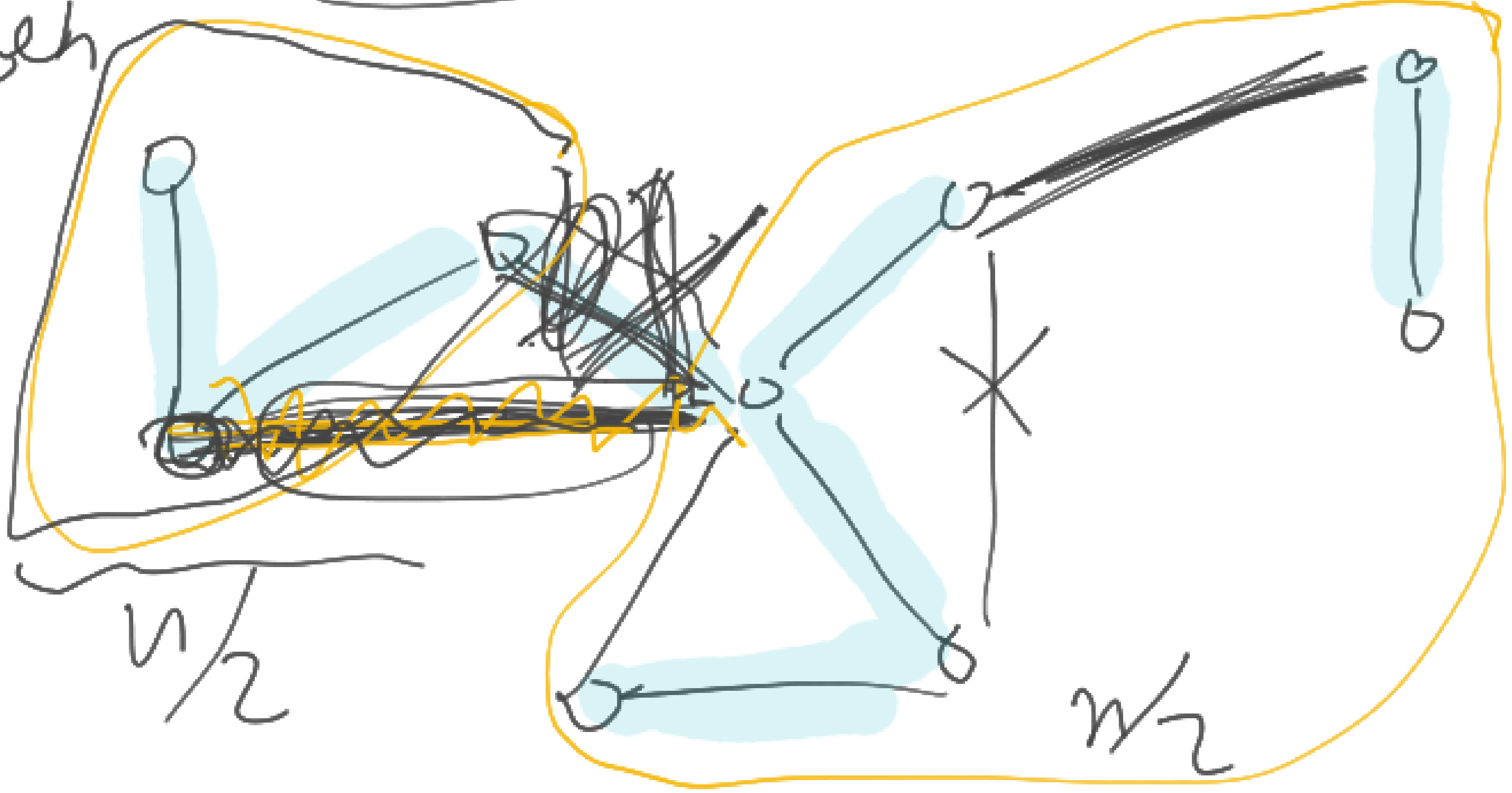
A: A spanning forest F (spanning trees for each CC).

two vertices are connected in $F \Longleftrightarrow$ connected in G

Goal: Maintain a spanning forest F of G.

$O(\log n)$ time

Insert an edge between two vertices in same CC



⇒ No update is required.

Delete a non-tree edge ⇒ No update to F.

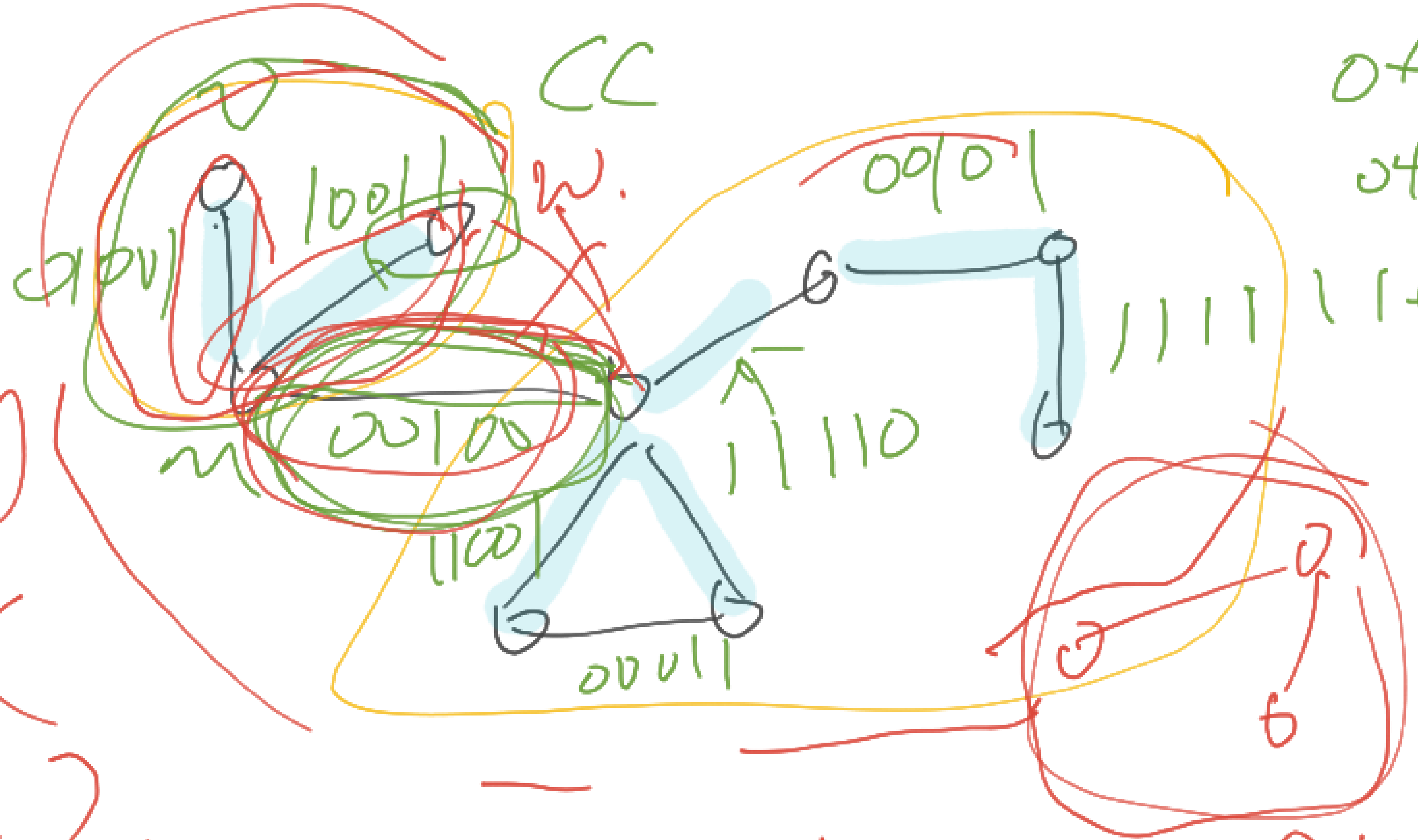
Insert an edge connecting two different CCs $\Rightarrow$ add new edge to F

Delete a tree edge
- Find a replacement (if possible)

IDEA: "sample" an edge going out of one CC

IDEA:

(edge going out of CC is unique).



$0+0=0$

$0+1=1$

$1+1=0$

give each edge (XOR) binary random name

$S_v = 01001$ $\quad S_u = 01001 + 10011 + 00100$

$= 11110$

$S_w = 10011$ $\quad S_{CC} = \bigoplus_{v \in CC} S_v = 01001 + 11110 + 10011$

$= 10111 + 10011 = 00100$

$S_{cc} = \bigoplus_{v \in CC} = 01001 + 11110 + 10011$

$\cancel{= 10001 + 10011}$

$\cancel{= 00010}$

$\boxed{\begin{aligned} &= 10000 + 10011 \\ &= 0010 \end{aligned}}$

$= 00100$