Exam 2024 b

Question 1:

a) steps

- each node discovers his neighbors with their id and give the minimum IP color the edge with the concatenation of the IPS, meaning IP1+IP2.

That will take O (1) time, in one round you are done.

Number of colors is m (the number of edges).

Proving correctness:

We assume that there is a edge (one or more that violates the edge coloring), that mean there is a v,u,x where v+u=v+x, that mean x=u and that contradict the unique ids.

b) We will use the same idea of the previous algorithm, but in a different way

- step one assign each color binary value from one to c length(log(c)) .

- run the previous algorithm, when we will use only the color of the minimum and join the minimum color and the port number of this minimum color. O(1)

- now for c rounds we will fix the issues

For color in reverse ( c ):

* Check neighbors if there is an issue with one of the edges.
* If issue:
* - we fix the issue by adding our port number to the string of the color

correctness, each node does not have a neighbored that is the same color,

And each node selected to set the edges color is not a neighbored of another one cause then there will be a contradiction.

Each node will correct his local issue starting from the highest value which will never choose an edge and so can have issue (the smallest edge can never have issue in his neighbored hood) and add a unique value that is his port number.

Color number: c + max(degree)^2

Time O(c)

Message=o(3m)=o(m)

**Better written with Chat**

**Question 1:**

**(A) Steps**

1. Each node discovers its neighbors along with their IDs.
2. Each edge is assigned a color based on the concatenation of the **minimum** of the two node IDs, i.e., IP1 + IP2.
3. This process takes O(1) time since it completes in a single round.
4. The number of colors used is **m** (the number of edges).

**Proof of Correctness:**

* Assume there exists at least one edge that violates the edge-coloring condition.
* This would mean that for some nodes v,u,xv, u, xv,u,x, we have: v+u=v+xv + u = v + xv+u=v+x which implies x=ux = ux=u, contradicting the assumption that all node IDs are unique.
* Therefore, the edge-coloring is correct.

**(B) Algorithm for the Anonymous Network**

We use a similar idea from the previous algorithm but apply it differently.

1. **Assign Binary Color Values:**
   * Each node assigns a **binary value** to its color, with values ranging from **1 to c** (i.e., binary length log(c)).
2. **Initial Edge Coloring:**
   * Run the previous algorithm, but instead of using node IDs, we use **only the color of the minimum node** and concatenate it with the port number of that node.
   * This takes O(1) time.
3. **Fix Coloring Conflicts Over ccc Rounds:**
   * Iterate over colors in **reverse order** (starting from the highest).
   * Each node checks if it has a conflicting edge color with any of its neighbors.
   * If a conflict is found, the node **fixes the issue** by appending its port number to the edge color string.

**Proof of Correctness:**

* Each node has a **unique color**, ensuring no direct neighbor has the same color.
* When a node sets an edge color, it **does not conflict** with another node performing the same action, as otherwise, there would be a contradiction.
* The process starts with the highest color values, ensuring that any edge adjustments propagate **without causing new conflicts**.

**Complexity Analysis:**

* **Number of Colors Used**: c+(max (degree)^2)
* **Time Complexity**: O(c)
* **Message Complexity**: O(3m) =O(m)