Definitions. A \( k \)-colouring of a graph \( G = (V, E) \) is a labelling \( f : V \to \{1, 2, \ldots, k\} \) of the nodes such that for each edge \( \{u, v\} \in E \) we have \( f(u) \neq f(v) \). A weak \( k \)-colouring of a graph \( G = (V, E) \) is a labelling \( f : V \to \{1, 2, \ldots, k\} \) of the nodes such that each non-isolated node \( u \) has a neighbour \( v \) with \( f(u) \neq f(v) \). As usual, \( n \) denotes the number of nodes.

Question 1: Covering maps. Prove that the following problem cannot be solved at all with deterministic PN-algorithms:

- Graph family: cycle graphs.
- Local inputs: a weak 2-colouring.
- Local outputs: a 3-colouring.

You can use the following textbook result (without proving it): covering maps preserve local outputs.

Question 2: Local neighbourhoods. Prove that the following problem cannot be solved in time \( o(n) \) with deterministic PN-algorithms:

- Graph family: path graphs with at least 3 nodes.
- Local inputs: nothing.
- Local outputs: a weak 2-colouring.

You can use the following textbook result (without proving it): isomorphic radius-\( T \) neighbourhoods imply identical local outputs for time-\( T \) deterministic algorithms.

Question 3: Randomised algorithms. Here is a randomised PN-algorithm that finds a weak 2-colouring in cycle graphs. Each node maintains a colour \( c \in \{1, 2\} \) and a state \( s \in \{0, 1\} \). Initially \( s \leftarrow 0 \) and \( c \) is chosen uniformly at random from \( \{1, 2\} \). When \( s = 0 \), the node is still running. When \( s = 1 \), the node stops and outputs \( c \). In each round, the algorithm proceeds as follows:

- All nodes (both running and stopped) send the current colour \( c \) to their neighbours.
- If \( s = 0 \) (the node is still running):
  - If the current value of \( c \) is different from the current colour of at least one neighbour, set \( s \leftarrow 1 \) and stop and output \( c \).
  - Otherwise, pick a new colour \( c \) uniformly at random from \( \{1, 2\} \).

(a) Prove that if the algorithm stops, it outputs a weak 2-colouring. (b) Prove that there is a constant \( p > 0 \) such that for any round \( t \), any node that is still running during round \( t \) will stop in round \( t \) with probability at least \( p \).

Question 4: Hardness of colouring. Prove: it is not possible to find a weak 2-colouring of a path graph in time \( O(1) \) with deterministic LOCAL-algorithms.

You can use the following textbook result (without proving it): it is not possible to find a 3-colouring of a cycle graph in time \( O(1) \) with deterministic LOCAL-algorithms.

Alternatively, you can also use Ramsey’s theorem (without proving it).