INFORMATICS.
OLYMPIAD.CH
INFORMATIK-OLYMPIADE
OLYMPIADES D'INFORMATIQUE
OLIMPIADI DELL'INFORMATICA

Rerouting

Joël Huber

8 January 2022

Problem Summary

Mouse Stofl is planning ferry routes for the Indonesian Mouse Ferry Organization

Problem Summary

Mouse Stofl is planning ferry routes for the Indonesian Mouse Ferry Organization

Given two labelled trees A and B, find minimal sequence of moves transforming A to B

Problem Summary

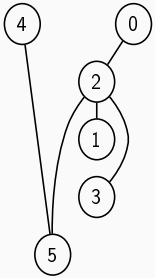
Mouse Stofl is planning ferry routes for the Indonesian Mouse Ferry Organization

Given two labelled trees A and B, find minimal sequence of moves transforming A to B

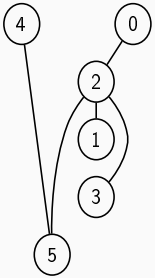
Move: Replace any existing edge by any edge

Example

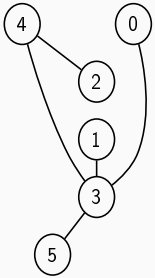
Tree A:



Current Tree:

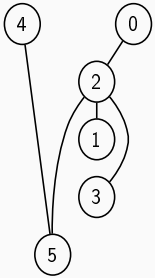


Tree B:

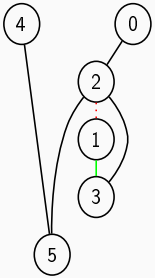


Example

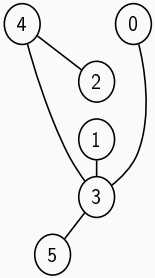
Tree A:



Current Tree:

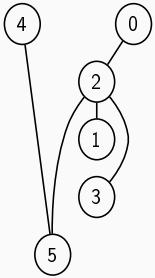


Tree B:

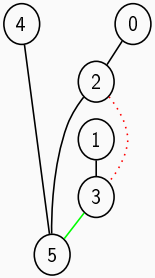


Example

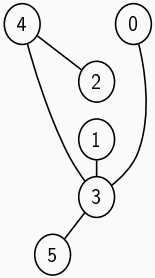
Tree A:



Current Tree:

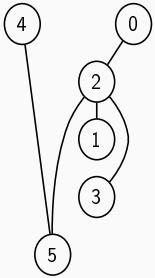


Tree B:

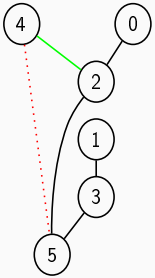


Example

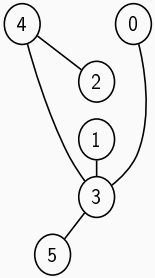
Tree A:



Current Tree:

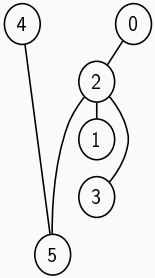


Tree B:

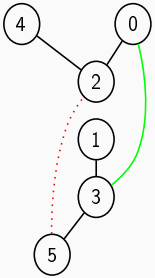


Example

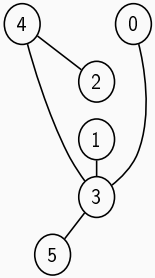
Tree A:



Current Tree:

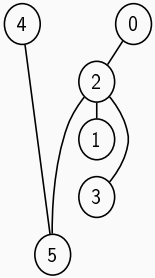


Tree B:

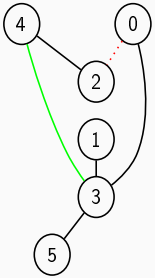


Example

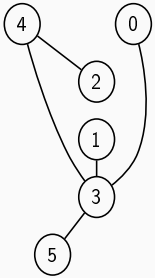
Tree A:



Current Tree:

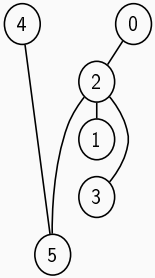


Tree B:

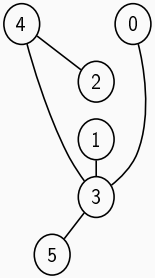


Example

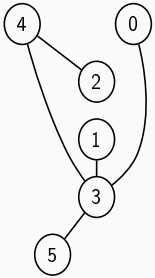
Tree A:



Current Tree:



Tree B:



Observation 1

Observation 1

For every edge, there is a replacement edge.

Observation 1

For every edge, there is a replacement edge.

Every edge is "removable" and "replaceable" at any point in time.

Observation 1

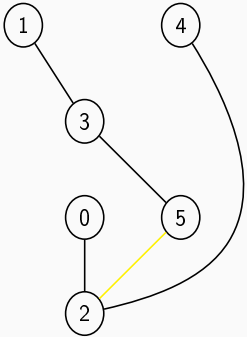
For every edge, there is a replacement edge.

Every edge is "removable" and "replaceable" at any point in time.

"Replaceable" = We can replace it with an edge from B

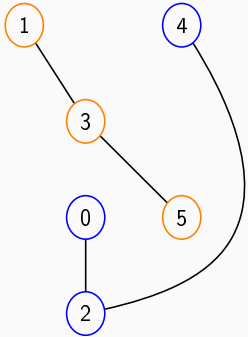
Observation 1

Let's see why



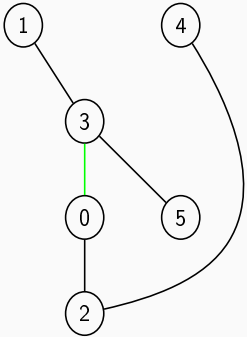
Observation 1

Let's see why



Observation 1

Let's see why



Observation 2

Every edge in A that doesn't appear in B needs to be changed

Lower Bound: Number of edges in A which aren't in B

Observation 2

Every edge in A that doesn't appear in B needs to be changed

Lower Bound: Number of edges in A which aren't in B

Observation 2: We can always achieve this bound.

Observation 2

Every edge in A that doesn't appear in B needs to be changed

Lower Bound: Number of edges in A which aren't in B

Observation 2: We can always achieve this bound.

Why? We can always select an edge still in our tree but not in B. By obs. 1, we can replace this edge

Observation 2

Every edge in A that doesn't appear in B needs to be changed

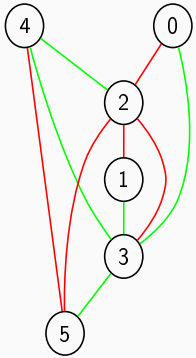
Lower Bound: Number of edges in A which aren't in B

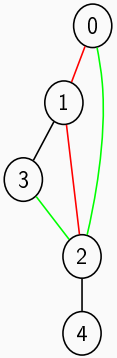
Observation 2: We can always achieve this bound.

Why? We can always select an edge still in our tree but not in B. By obs. 1, we can replace this edge

This observation solves Subtask 1.

Observation 2





Solution Idea (for 100 points)

Observation 2 allows us to always pick an edge and replace it

To get full score, we need to calculate replacement edges "less sequential"

Solution Idea (for 100 points)

Observation 2 allows us to always pick an edge and replace it

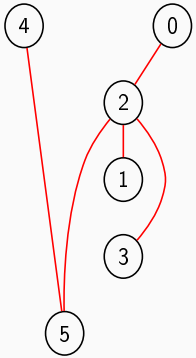
To get full score, we need to calculate replacement edges "less sequential"

Let's merge the edges appearing in both trees.

In the new tree, all edges need to be replaced

Solution Idea (for 100 points)

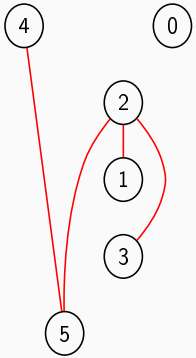
Consider a leaf



Solution Idea (for 100 points)

Consider a leaf

We can cut it (by obs. 1)

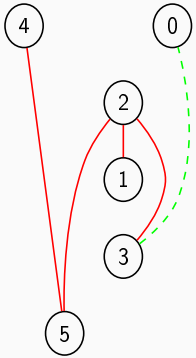


Solution Idea (for 100 points)

Consider a leaf

We can cut it (by obs. 1)

There will be a different edge incident to this leaf in B (by obs. 1)



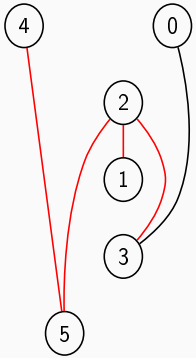
Solution Idea (for 100 points)

Consider a leaf

We can cut it (by obs. 1)

There will be a different edge incident to this leaf in B (by obs. 1)

Let's connect this vertex using this edge



Solution Idea (for 100 points)

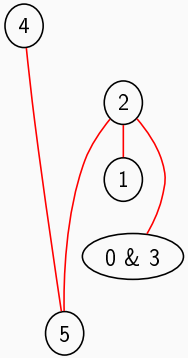
Consider a leaf

We can cut it (by obs. 1)

There will be a different edge incident to this leaf in B (by obs. 1)

Let's connect this vertex using this edge

Let's merge the newly connected vertices (the edge is in our tree and in B)



Solution Idea (for 100 points)

Consider a leaf

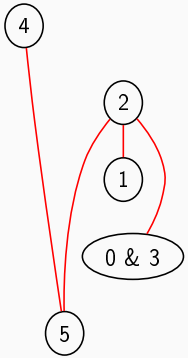
We can cut it (by obs. 1)

There will be a different edge incident to this leaf in B (by obs. 1)

Let's connect this vertex using this edge

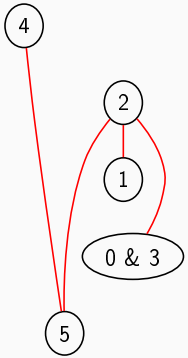
Let's merge the newly connected vertices (the edge is in our tree and in B)

Repeat this until the only leaf left is the root



Solution Idea (for 100 points)

Consider a leaf
We can cut it (by obs. 1)
There will be a different edge incident to this leaf in B (by obs. 1)
Let's connect this vertex using this edge
Let's merge the newly connected vertices (the edge is in our tree and in B)
Repeat this until the only leaf left is the root
We can implement this recursively using dfs



Solution Idea (for 100 points)

You can find all the details and a sample implementation in the solution booklet.