Practical 5: LCA Skeleton Tree

Ximena Moure Eliya Tiram

17/10/2023, submission deadline 23/10/2023

The following code is improved from the last delivery in order to get the restricted taxonomy to be able to do this practical. Now we do get 2111999 nodes.

```python
import networkx as nx


def load_graph_from_file(file_path):
    G = nx.DiGraph()  # Initialize directed graph
    with open(file_path, 'r') as file:
        for line in file:
            child, parent, rank = map(str.strip, line.split('|')[:3])
            child, parent = int(child), int(parent)

            # Skip self-loops
            if child != parent:
                # Add edge
                G.add_edge(parent, child)
                # Assign rank attribute to the child node
                G.nodes[child]['rank'] = rank
    return G


file_path = "nodes.dmp"

initial_tree = load_graph_from_file(file_path)

valid_ranks = ["superkingdom", "phylum", "class", "order", "family", "genus", "species"]
# Filter nodes based on rank attribute
nodes_to_remove = [node for node, attrs in initial_tree.nodes(data=True) if attrs.get('rank') not in valid_ranks]
# Create a copy of the graph
restricted_taxonomy = initial_tree.copy()


def create_restricted_tax(graph, nodes_to_delete):
    for node in nodes_to_delete:
        successors = list(graph.successors(node))
        predecessors = list(graph.predecessors(node))
        if node != 1:
            graph.remove_node(node)
        for source in predecessors:
            for target in successors:
                if source != target and not graph.has_edge(source, target):
                    graph.add_edge(source, target)


create_restricted_tax(restricted_taxonomy, nodes_to_remove)
print('Tree: ', nx.is_tree(restricted_taxonomy))
print('Rooted:', nx.is_arborescence(restricted_taxonomy))  # to check if it is rooted
print('Number of nodes in the restricted taxonomy: ', restricted_taxonomy.number_of_nodes())
print('Number of edges in the restricted version: ', restricted_taxonomy.number_of_edges())

```

Listing 1: Python code to extract restricted taxonomy

1. **Given a file nodes.dmp for the NCBI taxonomy and a file mapping.txt of mappings of sequence reads to**

NCBI taxonomic identifiers, write a Python script to find the lineage of the mapped nodes for each sequence read. Give the code of your Python script as your answer to this question, using the LATEX package listings.

```
def find_lineage(taxonomy_graph, node, cache=None):
    if cache and node in cache:
        return cache[node]

    lineage = []
    current = node
    while current in taxonomy_graph:
        lineage.append(current)
        parents = list(taxonomy_graph.predecessors(current))
        if not parents:
            break
        current = parents[0]

    lineage.reverse()

    if cache is not None:
        cache[node] = lineage

    return lineage


# Reading the mapping file
mapping = {}
with open('mapping.txt', 'r') as file:
    for line in file:
        li = line.strip().split()
        sq_read_id = li[0]
        nodes = [int(node) for node in li[1:]]
        mapping[sq_read_id] = nodes

lineages = {}
lineage_cache = {}
for read_id, taxonomic_nodes in mapping.items():
    lineage_for_read = [find_lineage
   (restricted_taxonomy, node, lineage_cache) for node in taxonomic_nodes]
    lineages[read_id] = lineage_for_read

print("lineages", lineages)

```

Listing 2: Python code ex1

2. **What is the taxonomic rank of the LCA of the mapped nodes for each of the sequence reads?**
   The taxonomic rank of the LCA of the mapped nodes for each of the sequence reads is the root.
3. **Do the sequence reads come from archaea, bacteria, eukaryota, or viruses?**
   The answer is:
   R00010: eukaryota: 8, bacteria: 2
   R00020: eukaryota: 16, viruses: 1, bacteria: 3
   R00030: eukaryota: 21, bacteria: 8, archaea: 1
   R00040: eukaryota: 26, bacteria: 14
   R00050: bacteria: 7, eukaryota: 42, viruses: 1
   R00060: viruses: 3, eukaryota: 43, bacteria: 14
   R00070: bacteria: 21, eukaryota: 49
   R00080: others: 1, eukaryota: 57, bacteria: 20, viruses: 2
   R00090: others: 1, eukaryota: 59, archaea: 2, bacteria: 21, viruses: 7
   R00100: others: 1, eukaryota: 79, bacteria: 19, archaea: 1

Thus, not all come from archaea, bacteria, eukaryota, or viruses. Most of them come from eukaryota.

```
from collections import defaultdict

sk_matching
    = { 2: 'bacteria', 10239: 'viruses', 2157: 'archaea', 2759: 'eukaryota'}
superkingdoms = {}

for read_id, seq in lineages.items():
    sk_read = defaultdict(int)
    sk_read['others'] = 0
    for l in seq:
        sk_read[sk_matching.get(l[1], 'others')] += 1

    superkingdoms[read_id] = sk_read

for read_id, sk_counts in superkingdoms.items():
    formatted_output = ', '.join([f"{k}: {
   v}" for k, v in dict(sk_counts).items() if not (k == 'others' and v == 0)])
    print(f"{read_id}: {formatted_output}")

```

Listing 3: Python code ex3

4. **Given a file nodes.dmp for the NCBI taxonomy and a file mapping.txt of mappings of sequence reads to NCBI taxonomic identifiers, write a Python script to build the LCA skeleton tree for each sequence read. Give the code of your Python script as your answer to this question, using the LATEX package listings.**

The following code uses the lineages from exercise 1

```
skeleton_trees = {}

for read_id, seq_lineages in lineages.items():
    all_nodes = {node for lineage in seq_lineages for node in lineage}
    skeleton_graph = nx.DiGraph()
    skeleton_graph.add_nodes_from(all_nodes)

    for lineage in seq_lineages:
        edges_from_lineage = zip(lineage[:-1], lineage[1:])
        skeleton_graph.add_edges_from(edges_from_lineage)

    skeleton_trees[read_id] = skeleton_graph

    print(f"{read_id} has {skeleton_trees[read_id].number_of_nodes()} nodes")

```

Listing 4: Python code ex4

5. **How many nodes are there in the LCA skeleton tree for each of the sequence reads?**
The answer to this question is in the code provided for question number 4 in the following line (which is inside the loop, so it is printed for every read)

```

     print(f"{read_id} has {skeleton_trees[read_id].number_of_nodes()} nodes")

```

Listing 5: Python code ex5

R00010 has 47 nodes
R00020 has 85 nodes
R00030 has 130 nodes
R00040 has 154 nodes
R00050 has 201 nodes
R00060 has 214 nodes

R00070 has 266 nodes
R00080 has 283 nodes
R00090 has 305 nodes
R00100 has 344 nodes

6. **What is the taxonomic rank of the root of the LCA skeleton tree for each of the sequence reads?**

   The taxonomic rank of the root of the LCA skeleton tree for each of the sequence reads is the root