Week 07 Tutorial

Reachability Queries

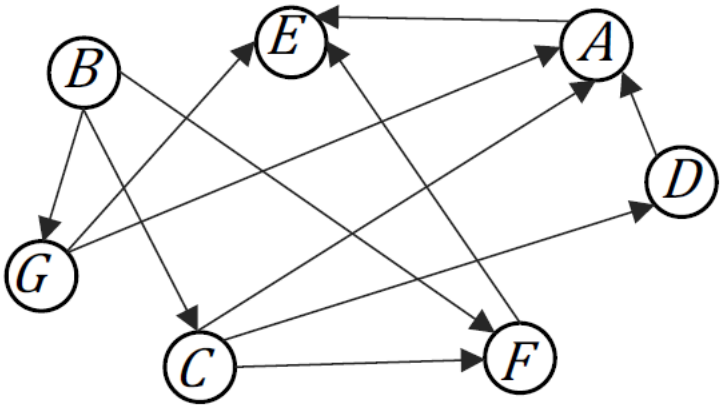
Aims

This exercise aims to get you to:

- Review key concepts on reachability
- Understand how to construct and use a transitive closure
- Understand the key features of optimal tree cover

Exercise 1: Reachability

1. Concept review: what are reachability queries?
2. Refresh your memory on graph traversal algorithms. Use BFS to answer the following queries. Can G reach B? Can C reach A? Can E reach F in the example graph?
3. What is the complexity for query processing in this case?



Example graph G

Exercise 2: Transitive closure

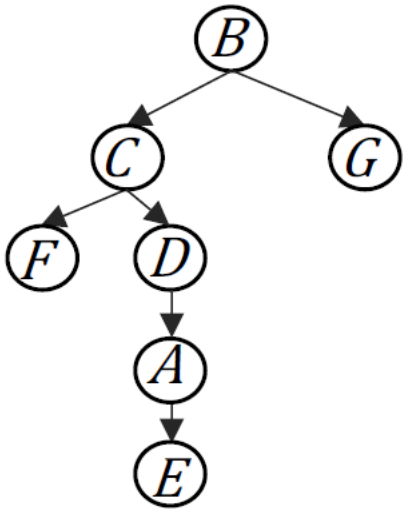
1. Concept review: the definition and motivation for a transitive closture.
2. Construct the transitive closure for the example graph G
3. Use the transitive closure to answer the reachability queries in Exercise 1.2
4. What is the **time complexity** for answering queries in this case? What is the **space complexity** of a transitive closure? (hint: the size of the matrix)

Exercise 3: Tree cover

1. Construct a spanning tree T_1 by yourself for G.
2. Call Algorithm 1 to construct a compression scheme from T1. Use the compression scheme to answer the reachability queries in Exercise 1.
3. Concept review: what is an optimal tree cover?
4. Repeat the above process for the given optimal tree cover T_2
5. Count the number of intervals in both compression schemes (corresponding to T_1 and T_2). What do you observe?

Algorithm1:

1. Find a spanning tree (tree cover) T of G.
2. Assign post-order numbers and indices as intervals to the nodes of T.
3. Go through vertices in **reverse topological order**. For each processed vertex q, consider all its in-edges (p, q). Add the intervals of q to the intervals of p. If any interval is subsumed, discard it.



An optimal tree cover

Exercise 4: Implementation

1. Load the example graph G via the class 'DirectedGraph' in tutorial_5.py.
2. Implement the class 'Reachability' which inputs the example graph G and answer the queries in Exercise 1.2.

3. You can choose one of the algorithms mentioned in lecture.