Lattice Checker

Lattice Theory Project

By:
Anupam Saini - CS17BTECH11003
Vijay Tadikamalla - CS17BTECH11040

Goal

The goal of this project is to check if the given Directed graph is a Lattice or not.

So, we wrote a C++ program that takes a directed graph as input and outputs the verdict

Procedure

Definition: A lattice element **y** is said to cover another element **x**, if **y > x**, but there does not exist a **z** such that **y > z > x**.

- We take the directed graph as input and store it in the form of a adjacency list, i.e, for each node in the graph, we maintain a set of elements that cover it.

- We then check this for cycles in this graph. If no cycles are found, then we can say that this graph is a **DAG (Directed acyclic graph)**

Procedure (continued)

- Now, we arrange all the nodes in the decreasing topological order, i.e,
  - $x_1, \ldots, x_n$, where $i < j$ implies $x_i \geq x_j$
- We also check that the first element in the topological sequence is the unique supremum of the lattice and the last element is the unique infimum of the lattice.
- Then $\forall\ i = 1, \ldots, n$, we attempt to compute $x_j \vee x_i$ for all $j < i$.
- And we finally return TRUE if all attempts were successful.

Procedure (continued)

To compute $x_i \vee x_j$, we use the following procedure:

- Let $u_1,\ldots,u_m$ be the elements covering x_i.
- Set $a \leftarrow x_j \vee u_1$.
- $\forall$ i = 2,..., m, check whether $x_j \vee u_i \leq a$ and if so, set $a \leftarrow x_j \vee u_i$.
- Verify that $a \leq x_j \vee u_i$ (i.e., that $a \vee (x_j \vee u_i) = x_j \vee u_i$) $\forall$ i = 1,..., m.
- If verification was successful, return a.

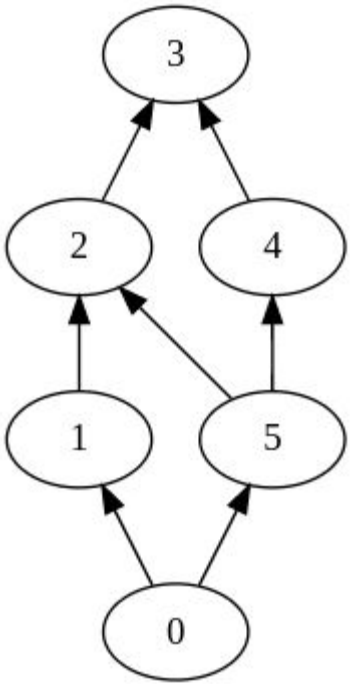
Reasoning

The procedure used to check if the given Directed graph is a Lattice is correct because we know that a join-semilattice of finite length with a universal lower bound is a lattice.

Results

Input:
6
0 1
1 2
2 3
0 5
5 4
5 2
4 3

Output:
Graph is a lattice.



Results (continued)

Input:

6
0 1
1 2
2 3
0 5
5 4
5 2
1 4
4 3

Output:

Graph is not a lattice.