Rolf Vidar Mazunki Hoksaas
Institutt for Informatikk
Universitetet i Oslo

October 9, 2020

OBLIG 2 — Graphs and dependencies

What is the time complexity of your solution? You must explain the stated complexity. If you have a different complexity for each problem, you need to discuss them separately.

To check if the graph is cyclic we have a linear complexity, since we only check each link once, traversing until we find the first loop, and as such it's limited by the amount of directed links we have in the graph.

We can use this method to see if any node is part of a dependency, so we can spot where the project would fail.

For all the timing problems, a similar solution has been used, and their complexities are all the same. They also follow a linear complexity, even though we are using recursion to calculate their timings. The recursion just goes through all the nodes until it hits an end, so it is bounded by the highest number of predecessors/sucessors.

□

What requirements are there for the input graph in order for the project to be successfully planned? Please put the discussion in the light of graf theoretical properties.

The graph can't have any circular dependencies, since this would create an issue like the chicken and the egg.

□

What kind of the graph algorithms did you use in your implementation?

We're using a Depth-First Algorithm to traverse through the nodes. This, compared to a Breadth-First algorithm doesn't make any real difference in the worst-case complexity, although one could discuss whether one or the other is better in average-case and best-case scenarios.

□

Submitted by Rolf Vidar Mazunki Hoksaas *on October 9, 2020.*