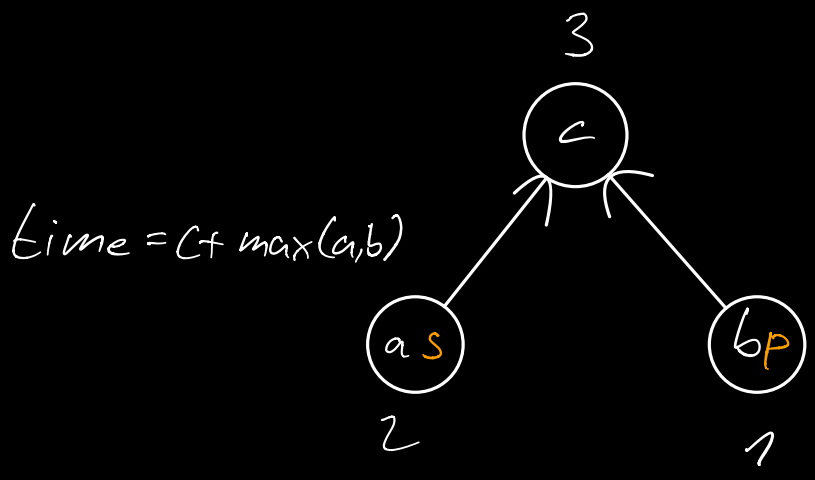
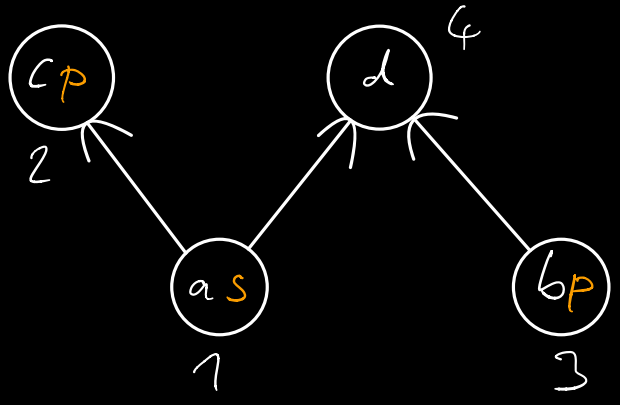
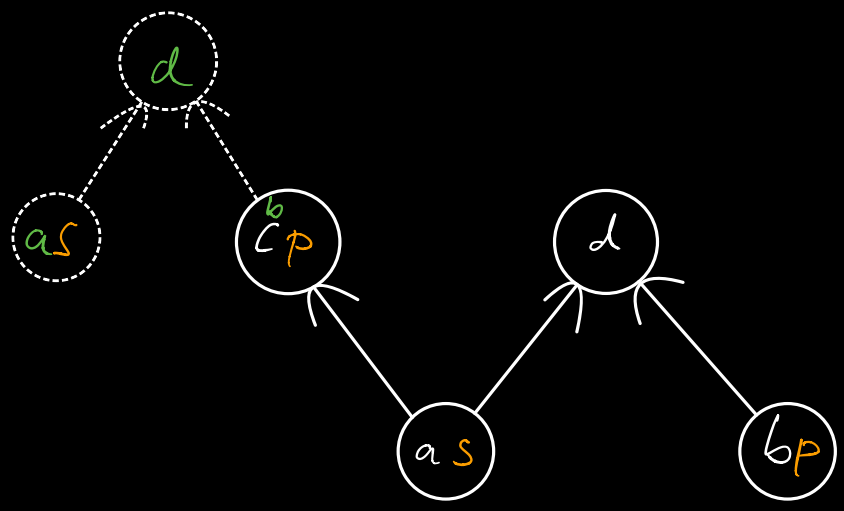




$c > b + d$

time = a + max(b+d, c)



time = d + max(a, b)

time = d + max(a + time, b)

Graph has to be fully explored before you can decide in which order the nodes have to be visited for the most efficient way.

It is possible to make it undecidable which node to pick next, because you can tell if some other unexplored nodes would lead to a situation, where it is not better to go for the parallel task first, but go for the sequential one.

So the only solution is to efficiently generate possible sequences how to traverse the graph and then simualte it‘s execution and pick the execution with the lowest execution time.