Task-1 D-seperation test

**Algorithm.** Testing whether **X** and **Y** are d-separated by **Z** in a DAG *G* is equivalent to testing whether **X**

and **Y** are disconnected in a new DAG *G*0, which is obtained by pruning DAG *G* as follows:

1. We delete any leaf node *W* from DAG *G* as long as *W* does not belong to *X* [ *Y* [ *Z*. This process is

repeated until no more nodes can be deleted.

2. We delete all edges outgoing from nodes in **Z**.

Function: d\_seperation(Graph, X, Y, Z).

Complexity: O(n^2), n is the number of nodes.

* The function takes in 4 values:
  + Graph
  + Start Node – X (A list of nodes)
  + End Node – Y (A list of nodes)
  + Observed Node – Z (A list of nodes)
* The function takes use of a recursive helper function(refine graph) to apply the first step of the algorithm.
* Function refine\_graph(G,X,Y,Z,union) returns a concentrated graph without leaf nodes and the edges outgoing from node Z deleted. Union is basically X U Y U Z.
* This concentrated graph is a directed graph, Hence a new undirected graph is created from the graph returned by refine\_graph().
* Find\_connection(dict,start\_node,end\_node) recursively finds all the paths from start\_node to end\_node. It returns a list of all the paths.
* If there is a path present, X and Y are not d-seperated(False). If find\_connection() returns an empty list then X and Y are d-seperated(True). Returns Boolean values

Function: refine\_graph(Graph, X, Y, Z)

* Recursively finds empty list values of keys in the dictionary to remove and also removes the key from other lists, Hence deleting the existence of the node
* Also removes the Outgoing edges from node Z. i.e. second step of the algorithm.
* Return a dictionary

Function: find\_connection(Graph,X,Y)

* The algorithm uses an important technique called backtracking: it tries each possibility in turn until it finds a solution. O(m+n)
* Recursively finds all paths from start node to end node in a undirected graph and returns either the path if it is able to find otherwise empty list.

Reference: <https://www.python.org/doc/essays/graphs/>