A collection of edges is easiest to implement, since only one additional class is needed that stores 3 data points for each edge: its parent node, child node, and label. An adjacency matrix has most efficient runtime when adding, deleting, and iterating through its nodes and edges. An adjacency matrix can rapidly determine the existence of an edge between two given nodes in constant time. I chose to use an adjacency list because it increases the efficiency of path-finding algorithms. For example, an important part of BFS is to find what nodes are children to a given node. For this operation, collection of edges and adjacency matrix requires iterating over all edges and/or nodes, while adjacency list requires only iterating over the spedific node’s outgoing edges, since they are already listed.