Plan Moving Forward

The primary objective of our research is to design and implement an efficient algorithm that can discover graphs for which one of the following two equalities holds:

- $dim(G) = edim(G) + mdim(G)$
- $mdim(G) = dim(G) + edim(G)$

Challenge lies not only in identifying graphs that satisfy these conditions but also in understanding the underlying relationships between the metric dimensions and the graph's topology. By focusing on metric dimensions—such as the vertex metric dimension (dim(G)), edge metric dimension (edim(G)), and mixed metric dimension (mdim(G))—we aim to pinpoint structural properties that allow these inequalities to hold.

Once our algorithm successfully identifies such graphs, the next step will involve a thorough analysis of the characteristics of these graphs. This stage is essential for uncovering any recurring patterns or structural motifs that might explain why certain graphs satisfy the equalities. The insights gained from this analysis will guide the development of a refined and optimized search space, targeting graphs that are more likely to meet the required conditions. Our ultimate goal is to build a more refined and efficient search algorithm that leverages these patterns to optimize search times while exploring a broader set of graphs.

Systematic Search : Goal is to use a **complete search** to find graphs that satisfy the equalities. Objective is to use this approach on smaller graphs to verify the correctness of the equalities:

1. **ILP set up:** Implement integer linear programs (ILPs) to determine $dim(G)$, $edim(G)$, and $mdim(G)$ for a given graph.
2. **Search space:** Define a search space of graphs with a fixed number of vertices and edges.
3. **Search algorithm:** Use a complete search algorithm to explore the search space and identify graphs that satisfy the equalities.

Stocastic Search : Goal is to use genetic algorithm to search for graphs that satisfy the equalities. We will use the following steps to implement the genetic algorithm:

1. **Initialization:** Generate random graphs, where *hyperparameters* are the number of vertices and edges.

2. **Evaluation:** AKA fitness function, which will evaluate the graphs based on the equalities. The closer the graph is to satisfying the equalities, the higher the fitness score.

3. **Selection:** Select the best graphs based on their fitness scores. It will be optional to set elitism to keep the best graphs from one generation to the next.

4. **Crossover:** Combine the best graphs from the previous generation to create new graphs. This will involve swapping edges between two graphs to create a new graph. Again it will be optional to set the crossover type. (e.g. roulette wheel, tournament, etc.)

5. **Mutation:** Randomly change the edges of the graphs to introduce new genetic material. This will help to explore new areas of the search space. Optimal to set the mutation rate.

6. **Termination:** Algorithm will stop after a certain number of generations or when the best graph satisfies the equalities. It will be able to set the termination criteria.

Additionally, if we might try to set up **symulated annealing** to search for graphs that satisfy the equalities. (depends on time and willpower)