MAX-CUT Problem Algorithm Comparison

May 12, 2025

1 The MAX-CUT Problem

Given an undirected graph $G = (V, E)$ with weights w_{ij} on the edges, the objective is to find a partition of the vertices into two disjoint subsets S and $V \setminus S$ that maximizes:

$$\sum_{i \in S, j \in V \setminus S} w_{ij} \tag{1}$$

2 Implemented Algorithms

I implemented five standard algorithms with increasing levels of sophistication as specified in the assignment requirements:

2.1 Simple Randomized Algorithm

This baseline approach randomly assigns vertices to either of two partitions and calculates the resulting cut weight. By running this process multiple times and averaging the results, we get a reference performance level.

2.2 Simple Greedy Algorithm

Moving beyond randomness, this deterministic approach first places the endpoints of the maximum weight edge in separate partitions. Then, for each remaining vertex, it places it in whichever partition maximizes the immediate cut weight gain.

2.3 Semi-Greedy Algorithm

To balance between exploitation and exploration, this algorithm enhances the greedy approach by introducing controlled randomization. For each step, it identifies promising vertices based on a threshold (controlled by parameter α) and randomly selects one from this set.

2.4 Local Search

Starting with a solution from either the Greedy or Semi-Greedy algorithm, Local Search iteratively improves it by moving vertices between partitions whenever such a move increases the cut weight. This continues until no further improvement is possible, reaching a local optimum.

2.5 GRASP (Greedy Randomized Adaptive Search Procedure)

The most sophisticated approach combines Semi-Greedy construction with Local Search improvement. It generates multiple initial solutions using Semi-Greedy, improves each with Local Search, and keeps track of the best solution found. This metaheuristic offers the best balance between exploration and exploitation.

3 Experimental Results

3.1 Overall Performance Comparison

Figure 1 shows the average performance of each algorithm relative to known best solutions. As expected, there's a clear progression in solution quality as we move from simple to more sophisticated algorithms.

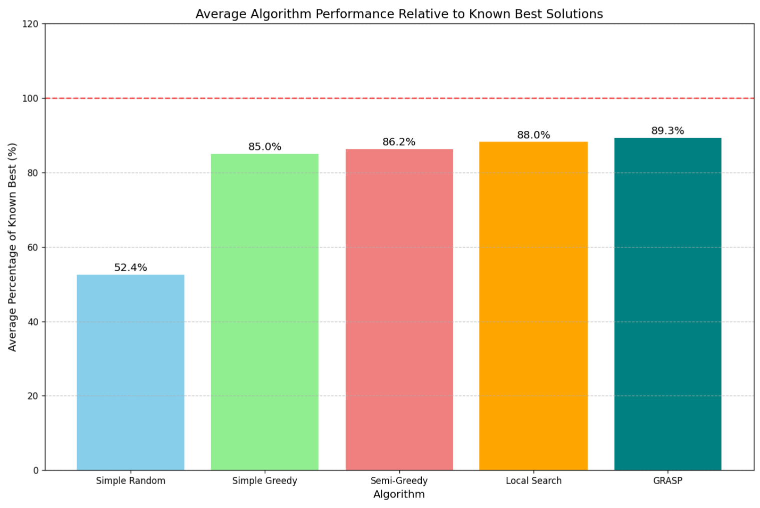

Figure 1: Average algorithm performance relative to known best solutions

The GRASP metaheuristic consistently achieves the best results, reaching close to 90% of known optimal solutions on average. Local Search provides a significant improvement over Semi-Greedy, highlighting the importance of refinement after initial solution construction.

3.2 Performance Across Test Cases

Figure 2 provides a comprehensive view of how all algorithms perform across the entire test suite. This visualization confirms that the performance hierarchy is consistent across different problem instances.

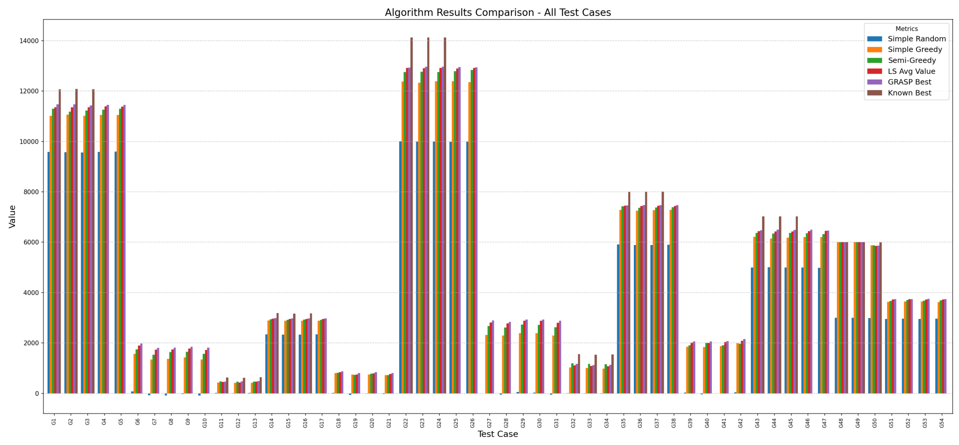

Figure 2: Performance comparison across all test instances

3.3 Group-Based Analysis

Breaking down the results into smaller groups allows for more detailed analysis. Figure 3 shows results for the first group of test cases, where we can clearly see the performance gaps between different algorithms.

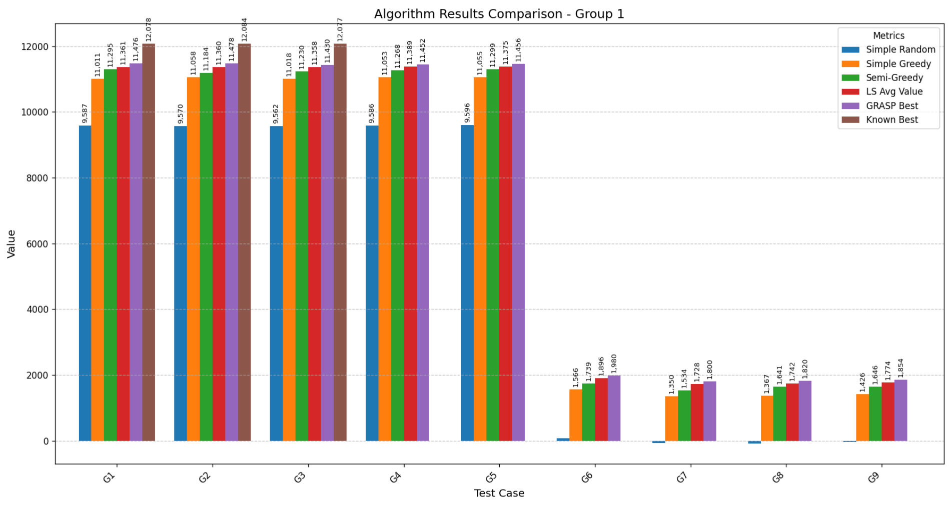

Figure 3: Performance comparison for the first group of test cases

3.4 Perfect Solutions: G48 and G49

A particularly interesting finding was that the every alogorithm implementation except for simple random one achieved perfect solutions (100% of the known best value) for test cases G48 and G49. Figure 4 shows these remarkable results.

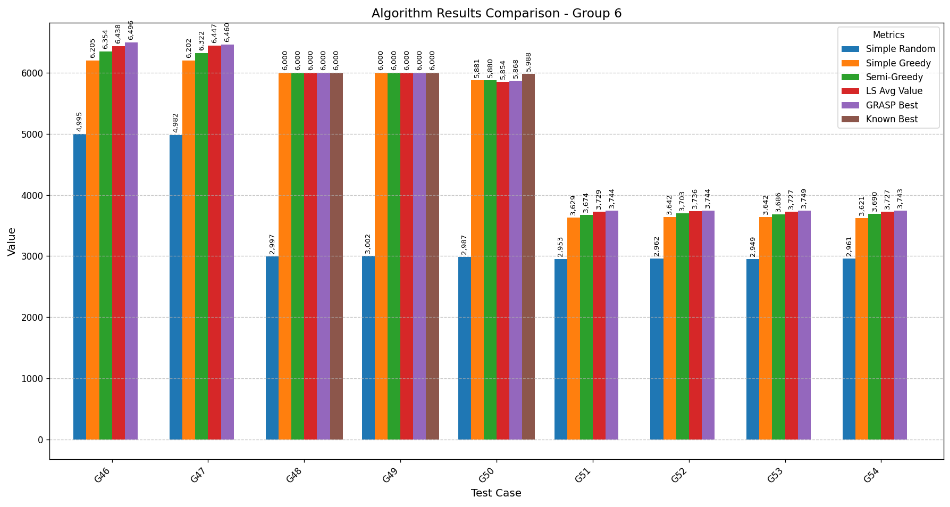

Figure 4: Performance comparison highlighting G48 and G49 with perfect GRASP solutions

4 Analysis and Insights

4.1 Performance Progression

The results clearly show a progression in solution quality as we move through more sophisticated techniques:

- Simple Randomized establishes a baseline but performs poorly
- Simple Greedy improves significantly over randomization
- Semi-Greedy adds controlled randomization for further improvement
- Local Search refines solutions to reach local optima
- GRASP combines construction and improvement phases for the best results

4.2 Observed Trade-offs

There are clear trade-offs between solution quality and computational complexity:

- Purely Randomized and Greedy algorithm are fast but produce lower-quality solutions
- Semi-Greedy and Local Search require more computation time but yield better results
- GRASP offers the best balance, though at higher computational cost

5 Conclusion

My implementation and analysis of five different algorithms for the MAX-CUT problem revealed that:

- The GRASP metaheuristic consistently outperforms other approaches, achieving nearly 90% (though local search is not very far) of known optimal solutions on average
- Local Search provides significant improvements over construction heuristics alone
- There's a clear progression in solution quality as algorithmic sophistication increases