# Analytical Report on Minimum Spanning Tree Algorithms

## 1. Summary of Input Data and Algorithm Results

The analysis was performed on 8 graphs using Prim’s and Kruskal’s algorithms. Each graph was evaluated for total cost, execution time, and number of operations performed.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Graph ID | Vertices | Edges | Algorithm | Total Cost | Operations Count |
| 1 | 5 | 7 | Prim | 16 | 48 |
| 1 | 5 | 7 | Kruskal | 16 | 44 |
| 2 | 4 | 5 | Prim | 6 | 29 |
| 2 | 4 | 5 | Kruskal | 6 | 32 |
| 3 | 6 | 7 | Prim | 21 | 57 |
| 3 | 6 | 7 | Kruskal | 21 | 57 |
| 4 | 5 | 7 | Prim | 13 | 48 |
| 4 | 5 | 7 | Kruskal | 13 | 44 |
| 5 | 6 | 6 | Prim | 20 | 50 |
| 5 | 6 | 6 | Kruskal | 20 | 41 |
| 6 | 10 | 15 | Prim | 30 | 181 |
| 6 | 10 | 15 | Kruskal | 30 | 109 |
| 7 | 12 | 20 | Prim | 37 | 279 |
| 7 | 12 | 20 | Kruskal | 37 | 163 |
| 8 | 11 | 14 | Prim | 35 | 186 |
| 8  9  9 | 11  13  13 | 14  14  14 | Kruskal  Prim  Kruskal | 35  48  48 | 105  218  109 |

## 2. Comparison Between Prim’s and Kruskal’s Algorithms

Theoretically, Prim’s algorithm performs better on dense graphs where the number of edges is high. It uses a priority queue to expand the tree efficiently from a starting vertex. Kruskal’s algorithm, on the other hand, excels in sparse graphs because it sorts all edges and then adds the smallest edges without forming cycles.

In practice, based on the experimental data, Kruskal consistently required fewer operations than Prim’s algorithm across all tested graphs, despite producing identical total costs for the MST. This indicates Kruskal’s superior performance efficiency on the provided datasets.

## 3. Conclusions

Both algorithms yield the same minimum spanning tree cost, confirming their correctness. However, the choice between them depends on the graph structure:  
- For dense graphs (many edges), Prim’s algorithm can be more efficient when implemented with adjacency lists and heaps.  
- For sparse graphs (few edges), Kruskal’s algorithm is simpler and faster.  
- Kruskal’s approach also benefits from easier parallelization and clearer implementation.  
Overall, Kruskal demonstrated lower operational complexity in this analysis.