Hessisches Kompetenzzentrum
für Hochleistungsrechnen

A Megaminx Solver

Project Manager
Alexander Botz

Principal Investigator
Prof. Dr. Pascal Schweitzer

Project Term
2025 - 2025

Clusters
Lichtenberg II Cluster Darmstadt

Additional Software
MobaXterm, FileZilla

Institute
Department of Mathematics

University
Technische Universität Darmstadt

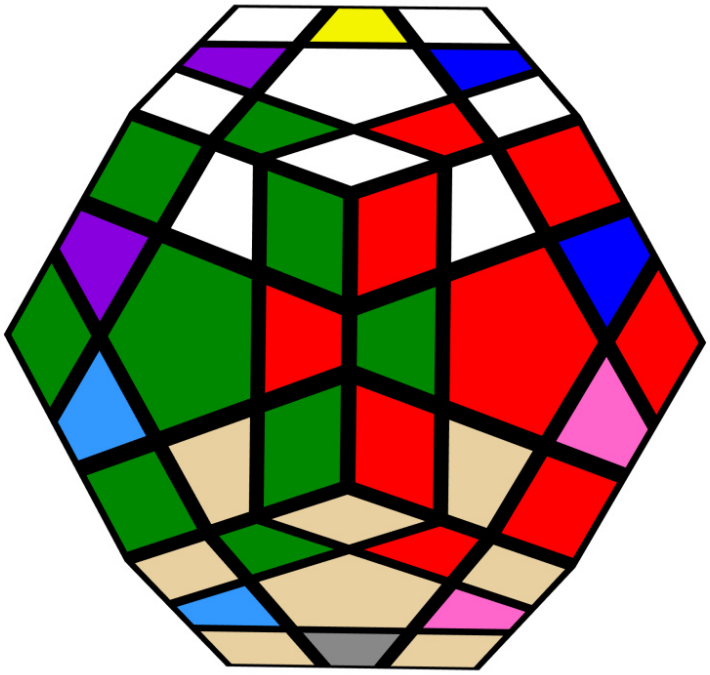

Figure 1: The Megaminx's Superflip-Configuration.

Introduction

Twisty puzzles, such as the Rubik’s Cube, are among the most popular types of puzzles and continue to fascinate people of all ages even today. An interesting task that arises with twisty puzzles is to solve them with as few twists as possible, for which there are currently no known efficient algorithms. While brute force can be a feasible approach to some puzzles, we often have to introduce a divide-and-conquer approach to sufficiently reduce the required computational resources. Unfortunately, decomposing the problem generally increases the number of twists in found solutions, so we want to avoid it as much as possible. The goal of this project was to develop an efficient solving algorithm for the Megaminx, a dodecahedral variant of the Rubik’s Cube, and to establish an upper bound on its God’s Number, which is the maximum number of moves required to solve any possible state.

Methods

To solve a given state of the Megaminx, we can consider an undirected graph whose nodes are given by the states of the Megaminx and for which two states are connected by an edge if they can be transformed into one another by a single twist. The

problem of finding a solution now reduces to finding a path from the given state to the solved state. Since this graph has around 10^{68} nodes, we cannot directly apply a breadth-first search algorithm, so we divide the solving process into multiple steps. Mathematical group theory allows us to multiplicatively decompose the number of nodes of the original graph, which eventually leaves us with a sequence of feasible shortest path problems that can be solved with breadth-first searches. Pregenerating all of these depths allows for very fast generation of solutions in practice, and by adding up the worst-case depths of each step, we obtain an upper bound on the Megaminx's God's Number.

Results

The most efficient variant of the solving algorithm provided an upper bound of 133 for the God's Number of the Megaminx and required the generation of over 22 GB of data, which would not have been feasible without the HPC Lichtenberg. The most basic variant of the solving algorithm was able to find solutions averaging 102 twists in fractions of milliseconds, with only a couple gigabytes of memory usage. With the HPC Lichtenberg it was easily possible to test with up to 80 GB of memory, which allowed the solving algorithm to achieve solutions averaging 82 twists, with a searching time of around a second.

Discussion

This solving algorithm serves as a foundation for future attempts at computational Megaminx solving, and we can expect further progress to be made. Based on our results, an improved upper bound of 114 on the Megaminx's God's number has already been conjectured, and several more ideas for modifications or entirely new substep selections have been suggested. We hope for these ideas to be implemented in the near future and for more complex optimizations to be introduced.

Last Update: 2026-02-06 09:29