AMERICAN JOURNAL OF AI CYBER COMPUTING MANAGEMENT
E-ISSN:3069-0102
VOL.4, NO. 4(2024)

THE ROLE OF ALGORITHMS IN SOLVING RUBIK'S CUBE

Author: Vanshika (Researcher)
Email: vanshika08bhatti@gmail.com

Abstract

The Rubik's Cube, a widely recognized combinational puzzle which provides a rich mathematical structure that has become an important object of study in discrete mathematics. Algorithms developed for solving it reveals how systematic sequences of moves can efficiently guide an immense state space of over 4.3×10^{19} possible configurations. Within the mathematical field, these algorithms describe core concepts such as permutation, symmetry, computation, optimization, and the study of algorithmic complexity. Thus, the Rubik's Cube serves as a unique and comprehensible model for exploring how mathematical reasoning can be applied to solve high-dimensional, contrived problems. Regardless of significant progress in deriving optimal solutions such as God's Number, heuristic algorithms, and group-theoretic approaches gaps remain in connecting theoretical optimality with practical algorithmic performance. Existing research basically focuses either on pure mathematical analysis of the cube's structure or on speed-solving techniques, leaving limited work that combines these perspectives. Furthermore, there remains a need for deeper exploration of how algorithms help with variations of the cube, including higher-order cubes and non-standard configurations.

The aim of this research is to analyze and evaluate the mathematical foundations that support algorithms for solving the Rubik's Cube, and to investigate improved algorithmic strategies that justify theoretical optimality with practical usability. This study will adopt a mixed approach of mathematical and computational methods, involving group-theoretic modelling, analysis of permutation structures, algorithmic model, and comparative assessment of existing solving methods. Through this, the research seeks to contribute to a deeper understanding of the mathematical principles commanding Rubik's Cube algorithms.

Key words:

Learning, quantum, algorithms, variants, mechanics, cubes, problem, structure, mathematical, solutions.

Introduction

The Rubik's Cube, since its invention in the 1970s by *Ernő Rubik*, has become a remarkable symbol of combinatorial complexity and brain teaser. With over 4.3×10^{19} possible layouts in its standard $3 \times 3 \times 3$ form, the cube demonstrates the complex relationship between mathematical theory and algorithmic design. While the puzzle is prominent for its deep connections to group theory and combinatorics, the practical act of solving the cube efficiently whether by human or machine depends substantially on mathematical algorithms.[1] These algorithms conclude a spectrum of disciplines, including combinatorial optimization, artificial intelligence (AI), reinforcement learning, and even quantum mechanics, and are central to the exploration, understanding, and solution of the cube in both theoretical and applied contexts.[2] This research paper examines the role of mathematical algorithms in solving the Rubik's Cube, explaining the evolution and diverseness of algorithmic approaches, their theoretical support, computational complexity, and the impact of emerging patterns such as deep learning and quantum computation. By integrating foundational and recent research,

[1] Korf RE.
[2] Felner, A., Korf, R. E. & Hanan, S.

this paper aims to provide a through account of how algorithmic thinking has shaped, and continues to shape, our involvement with the Rubik's Cube as a mathematical object and as a computational challenge.

Mathematical Foundations: Group Theory and Configuration Space

The Rubik's Cube is not simply a toy but a rich mathematical object whose possible states and moves can be exactly described using group theory. Each legal move is equivalent to a permutation of the cube's smaller pieces or "cubies", and the set of all possible configurations forms a group under the operation of move composition. This group-theoretic structure validates both a theoretical analysis of the cube's configuration space and a base for practical solution algorithms.

The cube's configuration space is huge: for the $3 \times 3 \times 3$ cube, there are $43{,}252{,}003{,}274{,}489{,}856{,}000$ possible states, each attainable via a sequence of slice rotations. The study of the cube's *"God's Number"*—the minimal number of moves required to solve the cube from any starting position , has been a central mathematical and algorithmic question. Remarkably, research established that God's Number for the $3 \times 3 \times 3$ cube is exactly 20 moves, a result achieved through extensive computational enumeration and algorithmic search. For larger generalizations, such as the $n \times n \times n$ or $n \times n \times 1$ cubes, the configuration spaces increase dramatically in size and complexity. *Demaine et al.* demonstrated that the diameter of the configuration space (i.e., the generalized God's Number) for these cubes scales as $\Theta(\frac{n^2}{\log n})$, both upper and lower bounds being established through algorithmic parallelization and combinatorial counting arguments. These foundational mathematical properties both motivate and constrain the design of efficient solving algorithms.

Classical Algorithmic Approaches to Solving Rubik's Cube

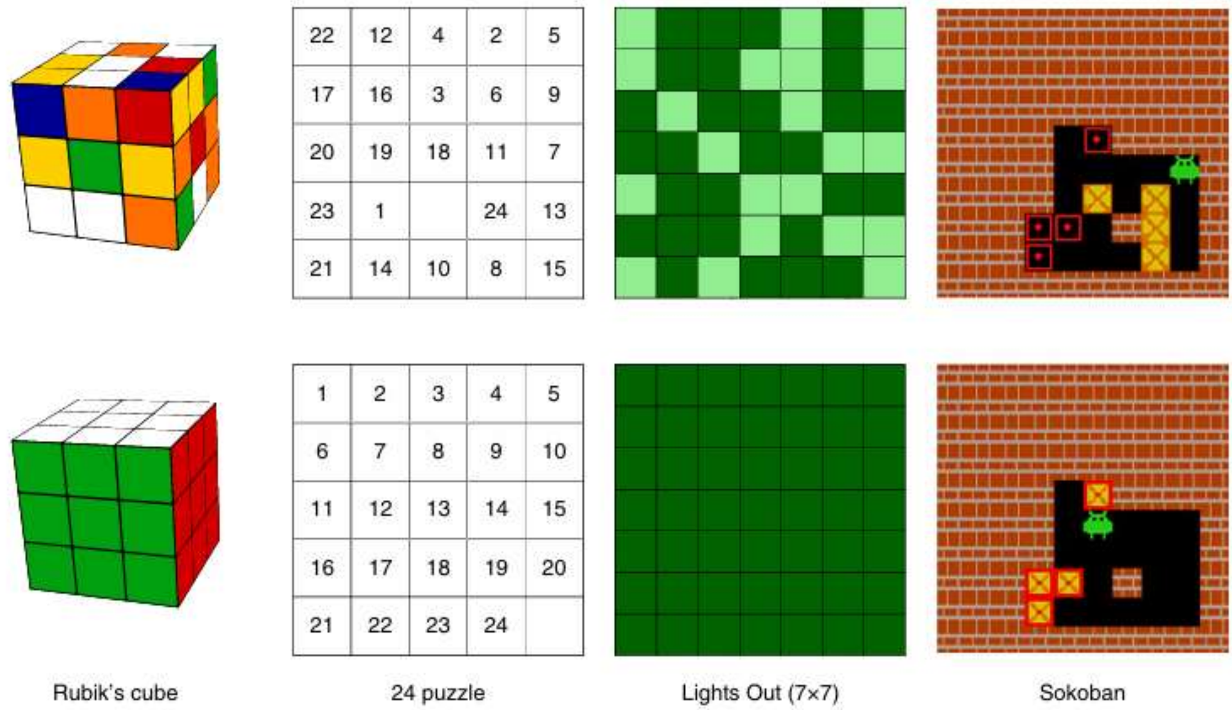

Fig. 1: Visualization of scrambled states (top) and goal states (bottom) for four different puzzles.

Layered and Human-Inspired Algorithms

The earliest algorithms for solving the Rubik's Cube were developed by enthusiasts and mathematicians who sought systematic, repeatable solution methods accessible to human solvers. These "layered" algorithms, such as the beginner's method, solve the cube in stages typically by first solving one face, then its adjacent layers, and finally orienting and permuting the remaining cubies. These methods rely heavily on the decomposition of the cube's complexity into manageable subgoals, each addressed by a repertoire of move sequences or "algorithms" (in the colloquial sense). While layered methods are suboptimal in terms of move count, they are highly effective for human learning, illustration of group-theoretic concepts (such as commutators and conjugation), and serve as the conceptual foundation for more advanced algorithmic strategies.[3]

Optimal and Computer-Assisted Algorithms

The pursuit of optimal solutions, i.e., those requiring the minimal number of moves, gave rise to computer-assisted algorithms that can efficiently pass over the enormous configuration space. *Kociemba's* two-phase algorithm is a landmark in this domain. The algorithm splits the solution process into two phases: the initial reduces the cube to a restricted subgroup of configurations using a subset of moves, and the following solves the cube within this subgroup, often using pre-computed lookup tables.[4] This two-phase approach supports group-theoretic properties to reduce search space and, when merged with powerful heuristics and computational resources, can find near-optimal or optimal solutions for arbitrary configurations.

The computational verification that God's Number is 20 moves was obtained via distributed computing, utilizing algorithms that segregated the vast configuration space and comprehensively searched for the worst-case scenario. These efforts emphasize the deep interplay between mathematical structure, algorithmic originality, and computational power in the search for optimal solutions.

Algorithmic Complexity and Hardness

While the standard $3 \times 3 \times 3$ cube is manageable with the aid of algorithms and data-rich lookup tables, the algorithmic complexity of solving generalized cubes expand quickly with size. *Demaine*, *Eisenstat*, and *Rudoy* proved that the decision problem of whether a $n \times n \times n$ Rubik's Cube can be solved in a given number of moves is NP-complete.[5] This result, proved via reduction from the Hamiltonian Cycle problem in square grid graphs, closes a prevailing open question about the computational hardness of the cube when optimality is required. The proof extends to variants such as the $n \times n \times 1$ Rubik's Square, further demonstrating the essential computational complexity of the puzzle in its generalized forms.[6] Notably, while polynomial-time algorithms exist for finding solutions (albeit not necessarily optimal ones) for large cubes, the requirement to find the shortest possible solution sequence thrusts the problem into the domain of intractability, barring breakthroughs in computational complexity theory.

[3] Felner, A., Korf, R. E. & Hanan, S.
[4] Corli S, Moro L, Galli DE, Prati E.
[5] Demaine ED, Demaine ML, Eisenstat S, Lubiw A, Winslow A.
[6] Smith, R. J., Kelly, S. & Heywood, M. I.

Standardizing of Solving as a Search and Planning Problem

State-Space Search and SAT Encoding

The Rubik's Cube can be compressed as a finite-state machine, where each state illustrates a particular configuration and each move corresponds to a state transition. Solving the cube, then, is equivalent to finding a path in the state graph from a scrambled state to the goal state. This interpretation aligns the cube with classical search and planning problems in artificial intelligence.

Jingchao Chen's work formalizes this abstraction by translating the Rubik's Cube into a *Boolean satisfiability* (SAT) problem. States are encoded as *Boolean variables* representing the color of each facelet, and moves are represented as state transition constraints.[7] The challenge is to find a sequence of moves (a plan) that fulfils the goal constraint: reaching the goal state within a given number of moves.

Efficient SAT encoding is critical due to the combinatorial explosion of variables and clauses for non-trivial cube sizes. *Chen* introduced optimizations such as encoding colors with fewer variables and utilizing the two-phase algorithm structure to reduce the size and complexity of the SAT example. Moreover, advanced SAT solving techniques such as at-least-one (ALO) constraints, two-product AMO encoding, and problem decomposition allow practical solution of the cube via SAT solvers, although with greater computational resource demands compared to heuristic or lookup-based algorithms.[8] This approach exhibits the utility of viewing the Rubik's Cube as a generic planning problem, manageable to formal logical encoding and solution all through general-purpose combinatorial optimization tools.

Reinforcement Learning and Sparse Reward Structures

Modern improvements in machine learning, particularly deep reinforcement learning (RL) have allowed the automatic discovery of solving strategies without direct algorithmic programming. The Rubik's Cube presents a particularly challenging workshop for RL due to its vast state space and slight reward structure only the solved state yields a positive reward.

Lin and *Liang* introduced an RL framework that employs policy gradient methods and neural networks to determine the cost (number of moves) between arbitrary cube states, consequently permit learning from fully scrambled cubes rather than depending on illustration near the solved state. Their framework, *"ChaseNet"* anticipate the minimal move distance between pairs of states and the policy network is practiced via Proximal Policy Optimization (PPO) that uses rewards shaped by these learned cost predictions.[9] Significantly, their approach avoids the need for solved-state sampling, aligning more closely with human problem solving strategies and achieving high success rates on the $2 \times 2 \times 2$ cube. This work illustrates the potential of RL and neural cost-to-go evaluation for cross-examining combinatorial puzzles with limited rewards and complex state transitions.

Quantum Formalism and Hamiltonian-Based Algorithms

The Rubik's Cube's group-theoretic and combinatorial structure encourages analogies with concepts from statistical and quantum mechanics. *Corli et al*. elaborated this perspective by developing quantum formalism for the cube, constructing a unitary representation of the cube's group and mapping cubie positions and orientations to quantum states. Cubies are modeled as single-particle

---

[7] Chen J.
[8] Arfaee, S. J., Zilles, S. & Holte, R. C.
[9] Lin Y, Liang S.

states demonstrating bosonic or fermionic behavior. The cube's symmetries are linked with Hamiltonian Operators. When the cube is in its solved configuration, all symmetries are conserved (Hamiltonians in their ground states).[10]

The solution process is formulated as an energy minimization problem: a deep reinforcement learning agent is rewarded based on the system's Hamiltonian spectrum, inspired by the Ising model. The solution develops in phases, each characterized by a specific Hamiltonian reward leading the agent toward symmetry restoration (i.e., solution).[11] While this approach has not yet been executed on quantum hardware, it demonstrates the potential for embedding combinatorial problems like the Rubik's Cube within quantum mechanical structures and for leveraging quantum-inspired algorithms in classical and quantum computation settings.

Computational Complexity and the Limits of Algorithmic Solvability

Upper and Lower Bounds for Solution Length

The disclosure of God's Number for the $3 \times 3 \times 3$ cube—20 moves was a turning point, enabled by algorithmic search, symmetry reductions, and immense computational resources. For generalized cubes, *Demaine et al*. established both upper and lower bounds for the worst-case solution length, showing that the number of required moves scales as $\Theta(\frac{n^2}{\log n})$ for both $n \times n \times n$ and $n \times n \times 1$ cubes. The upper bound is achieved by utilizing the parallelism fundamental in large moves (each affecting many cubies), while the lower bound is established by counting arguments based on the number of configurations and possible moves.[12] These results provide not only theoretical understanding into the versatility of solving algorithms but also practical guidelines for designing algorithms that undertake the optimal move count as cube size increases.

NP-Completeness and Intractability of Optimal Solutions

While polynomial-time algorithms exist for finding some solution to large cubes, finding the shortest solution (i.e., the minimal-move solution) is provably hard. *Demaine*, *Eisenstat*, and *Rudoy* established that the problem of deciding whether an $n \times n \times n$ cube can be solved in k or fewer moves is NP-complete. Their reduction from the *Hamiltonian Cycle* problem demonstrates that, under widely held complexity assumptions, no efficient algorithm exists for optimal solving in the general case.[13]

This intractability arises from the exponential growth of the configuration space and the combinatorial complexity of move sequences. For practical purposes, this means that algorithms must balance between optimality and tractability, often employing heuristic, approximate, or probabilistic strategies for large instances.

Algorithmic Model: From Enumeration to Machine Learning

Exhaustive Search and Lookup Tables

Early computer-assisted algorithms depend on detailed search of the configuration space, often assisted by massive lookup tables that store optimal or near-optimal solutions for large subsets of states. While useful for the $3 \times 3 \times 3$ cube, this approach becomes impractical for larger cubes due to memory and computational constraints.

---

[10] Rokicki T, Kociemba H, Davidson M, Dethridge J
[11] Arfaee, S. J., Zilles, S. & Holte, R. C.
[12] Brunetto, R. & Trunda, O.
[13] Korf, R. E.

SAT and Constraint Programming

Encoding the cube as a SAT or constraint satisfaction problem enables powerful, general-purpose solvers developed in the fields of logic and AI. Efficient encoding of moves, states, and constraints, as well as problem decomposition and variable reduction, are crucial for the practical solvability of large examples. While SAT-based methods are generally slower than specialized cube-solving algorithms, they offer flexibility and the ability to address variants of the puzzle or incorporate additional constraints.[14]

Reinforcement Learning and Neural Policy Search

Deep RL algorithms, exemplified by the *DeepCube* and *ChaseNet* frameworks, have demonstrated the ability to learn solving strategies from scratch, given only the definition of the puzzle and access to state transitions. These methods, especially when combined with neural cost-to-go estimators, can generalize to previously unseen cube configurations and offer a flexible, model-free alternative to hand-crafted algorithms.[15]

The primary challenges for RL-based methods are the size of the state space, the sparsity of rewards, and the need for efficient exploration. Innovations such as cost prediction networks, reward shaping, and sample-efficient policy optimization have made significant progress toward overcoming these obstacles, at least for smaller cube variants.

Quantum-Inspired and Hamiltonian-Based Algorithms

The mapping of the Rubik's Cube to quantum formalism and energy minimization opens new vistas for algorithmic design, inspired by techniques from quantum computing and statistical mechanics. While still in the exploratory stage, these approaches suggest that quantum hardware and algorithms may one day offer new tools for tackling combinatorial optimization problems like the Rubik's Cube, either by direct computation or by inspiring new classical algorithms.

Case Studies and Empirical Results

SAT-Based Solution of the $3 \times 3 \times 3$ Cube

Chen's SAT-based approach successfully finds solutions of length 20 for the standard cube, demonstrating the practical potential of SAT encoding and optimized solving strategies. While slower than domain-specific algorithms like *Kociemba's*, SAT solvers do not require enormous lookup tables and can be adapted to different cube variants and constraints.[16]

Reinforcement Learning on the $2 \times 2 \times 2$ Cube

Lin and Liang's RL approach achieved over 99.4% success rate in solving the $2 \times 2 \times 2$ cube without search or solved-state sampling, relying solely on a neural policy guided by cost-to-go predictions.[17] This result highlights the feasibility of learning effective policies in sparse-reward environments and points toward future scalability to larger cubes as computational and algorithmic advances permit.

Quantum Hamiltonian Approach

Corli et al. illustrated the theoretical formulation and classical simulation of cube solving via quantum Hamiltonian minimization, using DRL agents and Ising-model-inspired reward structures. As yet realized on quantum hardware, this configuration offers a novel outlook and suggests

---

[14] Felner, A., Korf, R. E. & Hanan, S.
[15] arXiv:1105.1436v1. 2011.
[16] Bonet, B. & Geffner, H.
[17] arXiv:2411.19583v1. 2024

direction for further research at the junction of combinatorics, quantum mechanics, and machine learning.

Discussion

The advancement of algorithms for solving the Rubik's Cube indicates the broader approach of algorithmic research in combinatorial optimization, artificial intelligence, and computational complexity. From human-devised layered methods to optimal computer-assisted algorithms, from SAT encoding to deep reinforcement learning and quantum-inspired approaches, each prototype addresses the core challenge of navigating an immense, richly structured configuration space.

Necessarily, the teamwork between mathematical theory (group theory, combinatorics, complexity) and practical algorithm design supports every advance. The cube serves as a representative of larger challenges in computer science: the exchange between optimality and tractability, the role of heuristics and approximation, and the impact of arising computational models. The NP-completeness of optimal solving for generalized cubes sets a theoretical limit on what can be accomplished by algorithms, yet ongoing research continues to push the boundaries of practical solvability through creative encoding, machine learning, and hybrid quantum-classical techniques.

Conclusion

Mathematical algorithms are essential for solving the Rubik's Cube, both as a theoretical sideline and a practical attempt. They include a synthesis of group-theoretic insight, combinatorial reasoning, and computational strategy. As the cube has evolved from a didactic tool to a benchmark for algorithmic innovation, so too have the algorithms for its solution evolved, from layered human strategies, through lookup-table-driven search and SAT encoding, to the vanguard of machine learning and quantum formalism. The study of these algorithms not only elucidates the structure and complexity of the cube itself but also contributes to our understanding of broader algorithmic principles and computational limits. As new models emerge and computational resources expand, the Rubik's Cube will unquestionably remain a hotbed for exploring the intersection of mathematics, computation, and human creativity.

References

1. Demaine ED, Demaine ML, Eisenstat S, Lubiw A, Winslow A. Algorithms for Solving Rubik's Cubes. arXiv preprint arXiv:1106.5736v1. 2011.
2. Demaine ED, Eisenstat S, Rudoy M. Solving the Rubik's Cube Optimally is NP-complete. arXiv preprint arXiv:1706.06708v2. 2018.
3. Chen J. Solving Rubik's Cube Using SAT Solvers. arXiv preprint arXiv:1105.1436v1. 2011.
4. Lin Y, Liang S. Solving Rubik's Cube Without Tricky Sampling. arXiv preprint arXiv:2411.19583v1. 2024.
5. Prodduturi, S.M.K. (2024). 'Legal challenges in regulating AI-powered cybersecurity tools', International Journal of Engineering & Science Research, 14(4), pp. 316–323.
6. M. V. Sruthi, "High-performance ternary designs using graphene nanoribbon transistors," Materials Today: Proceedings, Jul. 2023, doi: 10.1016/j.matpr.2023.07.170.
7. Rokicki T, Kociemba H, Davidson M, Dethridge J. God's Number is 20. In: Proceedings of the 19th Annual European Symposium on Algorithms (ESA 2011). 2011.
8. S. Gajula, "A Review of Anomaly Identification in Finance Frauds using Machine Learning System," International Journal of Current Engineering and Technology, vol. 13, no. 06, Jun. 2023, doi: 10.14741/ijcet/v.13.6.9.
9. Smith, R. J., Kelly, S. & Heywood, M. I. Discovering Rubik's cube subgroups using coevolutionary GP: a five twist experiment. In Proceedings of the Genetic and Evolutionary Computation Conference 2016 789–796 (ACM, 2016).

10. Sankar Das, S. (2024). Harnessing data lineage: making artificial intelligence smarter using data governance Frameworks. International Journal of Research and Analytical Reviews, 11(1). https://doi.org/10.56975/ijrar.v11i1.322571.
11. Johnson, C. G. Solving the Rubik's cube with learned guidance functions. In Proceedings of 2018 IEEE Symposium Series on Computational Intelligence (SSCI) 2082–2089 (IEEE, 2018).
12. Korf, R. E. Macro-operators: a weak method for learning. Artif. Intell. 26, 35–77 (1985).
13. Arfaee, S. J., Zilles, S. & Holte, R. C. Learning heuristic functions for large state spaces. Artif. Intell. 175, 2075–2098 (2011).
14. Korf, R. E. & Felner, A. Disjoint pattern database heuristics. Artif. Intell. 134, 9–22 (2002).
15. Felner, A., Korf, R. E. & Hanan, S. Additive pattern database heuristics. J. Artif. Intell. Res. 22, 279–318 (2004).
16. Bonet, B. & Geffner, H. Planning as heuristic search. Artif. Intell. 129, 5–33 (2001).