An Evolutionary Approach for Solving the Rubik's Cube Incorporating Exact Methods

Anurag Misra
Dept. of Computer Science and Engineering, IIT Kanpur

Abstract

Solutions calculated by Evolutionary Algorithms have come to surpass exact methods for solving various problems. The Rubik's Cube multiobjective optimization problem is one such area. The work presented is an evolutionary approach to solve the Rubik's Cube with a low number of moves by building upon the classic Thistlethwaite's approach. We provide a group theoretic analysis of the subproblem complexity induced by Thistlethwaite's group transitions and design an Evolutionary
Algorithm from the ground up including detailed derivation of our custom fitness functions. The implementation resulting from these observations is thoroughly tested for integrity and random scrambles, revealing performance that is competitive with exact methods without the need for pre-calculated lookup-tables

Introduction

- Classic 3*3*3 Rubik's Cube invented in 1974 by Erno Rubik
- Highly complex puzzle
- 4.3 * 10^19 unique configurations
- Only 1 of these is "solved state"
- Smallest number of moves to solve ("God's Number") yet unknown
- Only few exact approaches exist
- Most (promising) based on group theory
- No valid evolutionary approach incorporating group theory until now
- Each face of the Rubik's cube is referred to by its position (relative to users viewpoint)
- Common notation is F, R, U, B, L, D
- These also stand for a 90 degree clockwise turn
- Correspondingly Fi, Ri, Ui, Bi, Li, Di denote counter-clockwise 90-degree turn.
- Moreover, F2, R2, U2, B2, L2, D2, correspond to clockwise half turns

Motivation

- Idea
  - Take human strategies and incorporate them into an evolutionary approach
  - Use group theoretical background to reduce complexity
- Result
  - A more powerful evolutionary algorithm adapting human strategies and incorporating exact methods
  - Symbiotic Intelligence
- Advantage
  - No need of terabytes of pre-calculated lookup tables

Existing Exact Algorithm: Thistlewaiste's Algorithm

- Developed by Morgan Thistlewaite in 1984
- Divides the problem of solving the cube into 4-subproblems

Definition

$G_0 =< F, R, U, B, L, D > \quad \supset$
$G_1 =< F, U, B, D, R2, L2 > \quad \supset$
$G_2 =< U, D, R2, L2, F2, B2 > \quad \supset$
$G_3 =< F2, R2, U2, B2, L2, D2 > \quad \supset$
$G_4 = I$
with $|G_0| > |G_1| > |G_2| > |G_3| > |G_4|$.

- Transition cube from Gi → Gi+1 only using moves from Gi
- Pre-calculated lookup-tables, solves in max. 52 moves

Definition

A subset $S \subseteq G$, is called a *generator* of G if any element of G can be written of a product of elements of S and their inverses. This is denoted by $G =< S >$.

- All the sequences can generated using the generator of G0, G1, G2, G3 when in that particular sub-group

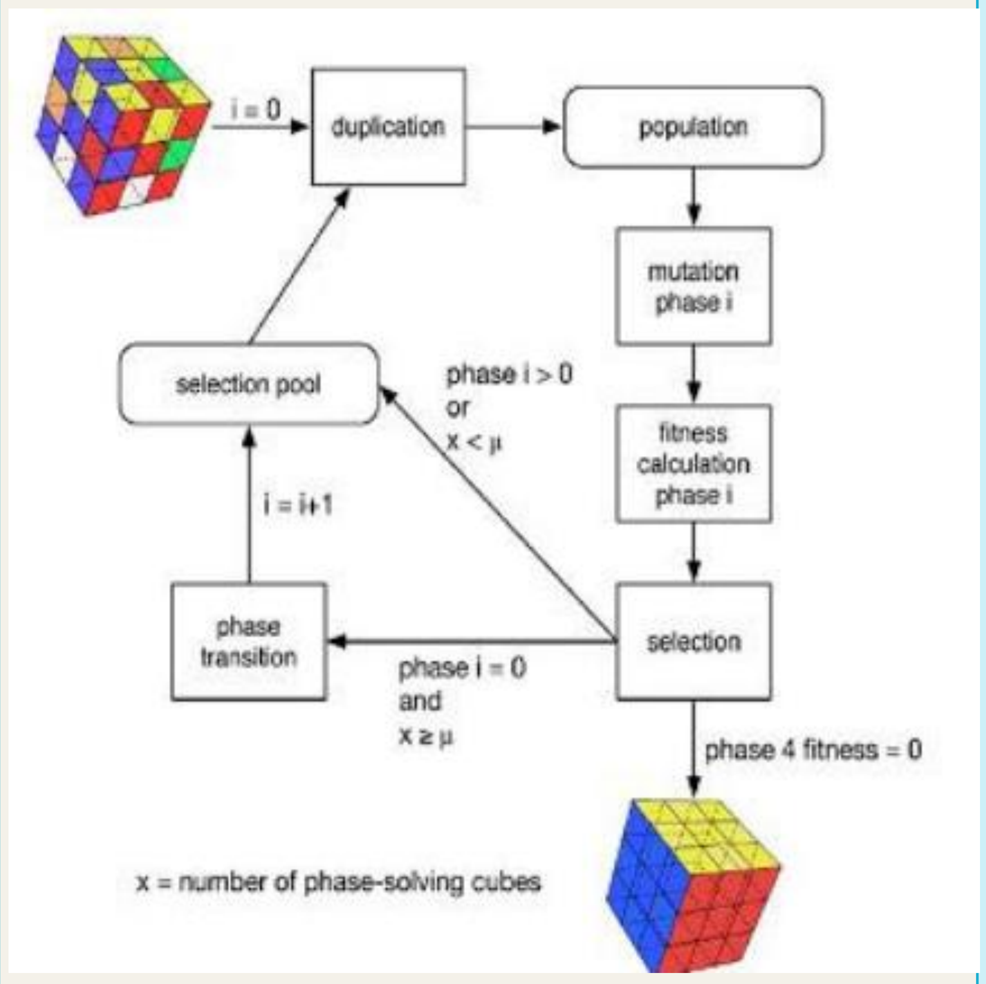
Size of the Groups:
- G(0), |G(0)| = 4.3 * 10^19 * no constraint
- G(1), |G(1)| = 2.11 * 10^16 * orientation of edge cubies
- G(2), |G(2)| = 1.95 * 10^10* orientation of corner cubies transport of edge cubies to/from middle layer
- G(3), |G(3)| = 6.63 * 10^5 * cubies from one layer to another layer and edge cubies in correct orbits
- G(4) is the final solved states

The transitions take place using pre-calculated lookup tables only using moves from generator of the particular sub-group

Thistlewaiste's Evolutionary Strategy

We present a 4-phase ES with each phase calculating one group transition
(will be referred to as Thistlethwaite Evolution Strategy, TWES). These
phases share the same basic selection method but differ in mutation operators
and fitness functions. Effectively, the presented ES can best be described as four
consecutive ES, each using the solution of the
previous one as starting individual
to be duplicated (the first using the scrambled input Cube)

TWES- Workflow



RUBIK's cube
- Represented using 6 2D matrices
- Can be mutated only by applying move sequences
- Remembers all mutations undergone as a sequence list
- Automatically removes abundant moves after each mutation
- Remembers optimized sequence only

WORKFLOW
- Scrambled cube is duplicated λ times
- Yields first population after the phase transition
- Process is repeated until phase-4 is solved
- Selection pool generated by choosing best μ individuals from current population

FITNESS FUNCTION
- Each phase has it's own fitness function, counting
  - Wrong oriented/positioned cubiesaccording to group constraints
  - Length of the remembered sequence list
- Weights adjustable
- Example G(0) → G(1):
  - phase(o) fitness = weight.(w) + c
  - w: = number of wrong oriented edges
  - c: = length of the sequence list
  - G(i) constraints satisfied if phase(i) fitness = c

RESULT

	run 1	run 2	run 3	run 4	run 5
avg. Generations	95.72	100.63	92.71	99.66	92.22
avg. Moves	50.67	50.32	50.87	50.23	49.46
avg. Time(s)	321.78	381.68	393.99	312.98	287.93

References

N El-Sourani, S. Hauke, M. Borschbach, "An Evolutionary Approach for Solving Rubik's Cube Incorporating Exact Methods", EvoApplications2010
▪N El-Sourani, M. Borshbach, "Design and Comparison of two Evolutionary Approaches for Solving Rubik's Cube"
▪M Borshbach, C. Grelle, "Empirical Benchmarks of a Genetic Algorithm incorporating Human Strategies", University of Applied Sciences 2010