UNIVERZITET U BEOGRADU
MATEMATIČKI FAKULTET

Miloš Đurić

REŠAVANJE RUBIKOVE KOCKE

master rad

Beograd, 2023.

Mentor:

dr Miodrag Živković, redovni profesor
Univerzitet u Beogradu, Matematički fakultet

Članovi komisije:

dr Predrag Janičić, redovni profesor
Univerzitet u Beogradu, Matematički fakultet

dr Stefan Mišković, docent
Univerzitet u Beogradu, Matematički fakultet

Datum odbrane: _____

Naslov master rada: Rešavanje Rubikove kocke

Rezime: Rubikova kocka je mehanička mozgalica koju je izumeo mađarski pronalazač Erne Rubik. Nastala je kao nastavno sredstvo koje bi studentima pomoglo da razumeju 3D objekte i njihovo kretanje. U ovom radu polazi se od osnovnih pojmova iz teorije grupa, idući ka tome da se permutacije koriste u igrama za jednog igrača, uz uvođenje potrebnih teorema i definicija. Dolazi se do teorije o Rubikovoj kocki i algoritama za njeno rešavanje. U poslednjoj glavi opisana je programska realizacija dva algoritma sa 3D modelom Rubikove kocke.

Ključne reči: Rubikova kocka, algoritmi, permutacije, grupe

Sadržaj

Glava 1

Neophodni pojmovi i tvrđenja iz teorije grupa

1.1 Uvod

Danas svima poznatu mozgalicu, Rubikovu kocku, izumeo je *Erne Rubik*[1] 1974. godine. Nakon što je izumeo kocku i promešao je prvi put, Rubik nije imao ideju kako bi je mogao vratiti u prvobitno stanje. Za prvo rešavanje kocke mu je trebalo čak mesec dana. Čim se pojavila na globalnoj sceni, Rubikova kocka je započela svoj rast popularnosti. Ubrzo je primećen značaj kocke u matematici i počele su se razvijati različite vrste algoritama za njeno rešavanje.

Prvobitna ideja je bila trodimenzionalna kocka sa glavnim ciljem rešavanja same kocke, dok je u današnje vreme cilj prerastao u brzo rešavanje kocke. Postoji mnoštvo takmičenja u brzom rešavanju kocke (eng. *speedcubing*) koje organizuje udruženje *World Cube Association*.

U matematici, Rubikova kocka se može opisati i rešiti pomoću teorije grupa. Različite transformacije i stanja kocke se mogu predstaviti pomoću podgrupe grupe permutacija generisane različitim vertikalnim i horizontalnim rotacijama strana kocke. Na ovaj način se mogu opisati algoritmi za rešavanje kocke različitih kompleksnosti i efikasnosti.

U prvoj glavi su uvedeni neophodni pojmovi i tvrđenja iz teorije grupa, potrebni za razumevanje druge glave, u kojoj su definisane Rubikova kocka i grupa Rubikove kocke. Nakon toga, u trećoj glavi je predstavljeno nekoliko metoda za rešavanje

[1]Erne Rubik (mađ. *Ernő Rubik*) bio je mađarski pronalazač, vajar i profesor arhitekture. Najpoznatiji je po svojim mozgalicama - Rubikovoj kocki i Rubikovoj zmiji

Rubikove kocke, dok je u četvrtoj glavi je opisana aplikacija za rešavanje Rubikove kocke, algoritmi koji su u njoj implementirani, kao i rezultati eksperimenta izvršenog pomoću aplikacije.

Uvode se neophodni pojmovi i tvrđenja iz teorije grupa, neophodni za razumevanje funkcionisanja Rubikove kocke. Tekst je zasnovan na materijalu iz projekta Lindzi Deniels (eng. *Lindsey Daniels*, [1]), sa sledećim dopunama [7]:

- Definicije: 1.2.2, 1.2.3, 1.2.4, 1.2.5, 1.2.11, 1.3.5,
- Leme: 1.2.1, 1.4.1,
- Primeri: 1.2.2, 1.2.4, 1.2.7, 1.3.2, 1.3.4.

1.2 Grupe

Definicija 1.2.1. Neka je G neprazan skup i neka je $*$ binarna operacija na njemu, za koju važi

$$*\colon G \times G \to G$$
$$(g_1, g_2)\colon \mapsto g_1 * g_2$$

Tada algebarsku strukturu $(G, *)$ zovemo *grupoid*.

Definicija 1.2.2. Asocijativni grupoid ili *semigrupa* je grupoid za koji dodatno važi

$$(a * b) * c = a * (b * c).$$

Definicija 1.2.3. Semigrupa koja ima jedinični element ili jedinicu $1 \in G$ za koju važi

$$1 * a = a * 1 = a$$

za sve $a \in G$, naziva se *monoid*.

Definicija 1.2.4. Neka je $(G, *)$ monoid. Element $a \in G$ je *invertibilan* ako postoji $b \in G$ za koje važi

$$b * a = a * b = 1.$$

Za element b kažemo da je inverz elementa a.

Lema 1.2.1. Element b iz prethodne definicije je jedinstven i označavamo ga sa a^{-1}.

Dokaz. Uzmimo element a i monoid G iz prethodne definicije. Pretpostavimo da a ima 2 inverza, b i c. Tada važi:

$$b = a * e = b * (a * c) = (b * a) * c = e * c = c.$$

□

Definicija 1.2.5. Monoid u kom je svaki element invertibilan zove se *grupa* i predstavlja se kao algebarska struktura

$$(G, *, ^{-1}, 1).$$

Obično se simbol operacije izostavlja, tj. $a * b$ se označava sa ab, na osnovu čega sledi da je $\underbrace{a * a * ... * a}_{\text{n}} = \underbrace{aa...a}_{\text{n}} = a^n$.

Primer 1.2.1. Neka je G skup celih brojeva, $G = \mathbb{Z}$. Za svako $x, y, z \in G$ važi:

1. $x + y + z = (x + y) + z = x + (y + z)$,
2. jedinični element skupa G je 0, jer važi $x + 0 = 0 + x = x$,
3. Za svako $x \in G$ postoji inverzni element $-x$ za koji važi $x + (-x) = 0$.

Sledi da je G grupa sa operacijom sabiranja.
Ukoliko bismo koristili operaciju množenja onda G ne bi bila grupa, jer ne sadrži sve inverze. Ako uzmemo $2 \in \mathbb{Z}$, njegov inverz je $\frac{1}{2}$, ali $\frac{1}{2} \notin \mathbb{Z}$.

Primer 1.2.2. Neka je G skup realnih brojeva bez nule, $G = \mathbb{R} \setminus \{0\}$. Skup G sa operacijom množenja je grupa, jer za svako $x, y, z \in G$ važi:

1. $x \cdot y \cdot z = (x \cdot y) \cdot z = x \cdot (y \cdot z)$,
2. jedinični element skupa G je 1, jer važi $x \cdot 1 = 1 \cdot x = x$,
3. Za svako $x \in G$ postoji inverzni element $\frac{1}{x}$ za koji važi $x \cdot \frac{1}{x} = 1$.

Definicija 1.2.6. *Red grupe* G definiše se kao broj elemenata u grupi i označava se sa $|G|$. Grupa može biti konačnog ili beskonačnog reda u zavisnosti od toga da li je red grupe konačan broj.

Definicija 1.2.7. Neka je H podskup skupa G. Grupu H, sa istom operacijom, koja ja definisana nad grupom G nazivamo *podgrupom* grupe G, u oznaci $H \leq G$.

Primer 1.2.3. Neka je $G = \mathbb{Z}_8 = \{0, 1, 2, 3, 4, 5, 6, 7\}$ skup celih brojeva po modulu 8. Skup G sa operacijom sabiranja je grupa čiji je red $|G| = 8$.
Primer podgrupe grupe G je $H = \{0, 2, 4, 6\}$ sa operacijom sabiranja po modulu 8.

Primer 1.2.4. Neka je $G = \mathbb{R} \setminus \{0\}$ skup realnih brojeva bez nule i neka je $H = \{\frac{a}{b} | a \neq 0, b \neq 0, a, b \in \mathbb{Z}\}$ podskup skupa G. Skup G sa operacijom množenja je grupa, a skup H je njegova podgrupa, jer za svako $a, b, c, d \in H$ važi:

- $\frac{a}{b} \cdot \frac{c}{d} = \frac{a \cdot c}{b \cdot d} \in H$,
- jedinični element je $1 = \frac{1}{1} \in H$,
- inverz elementa $\frac{a}{b} \in H$ je element $\frac{b}{a} \in H$.

Definicija 1.2.8. Neka je G grupa, $H \leq G$ i $a \in G$. Skup $aH = \{ah | h \in H\}$ naziva se *levi koset* podgrupe H. Analogno, *desni koset* grupe H je $Ha = \{ha | h \in H\}$.

Teorema 1.2.1. (*Lagranžova teorema*) Ako je G konačna grupa i H njena podgrupa, onda red grupe H deli red grupe G. Broj različitih levih, odnosno desnih, koseta grupe H u grupi G je $|G|/|H|$.

Definicija 1.2.9. Neka su G i H konačne grupe i neka je $H \leq G$. *Indeks* grupe H u G je $[G : H] = |G|/|H|$.

Primer 1.2.5. Ako je $G = \mathbb{Z}_6 = \{0, 1, 2, 3, 4, 5\}$ i $H = \{0, 2, 4\}$, onda važi $[G : H] = |G|/|H| = 6/3 = 2$. Dva različita koseta grupe H su $\{0, 2, 4\}$ i $\{1, 3, 5\}$.

Definicija 1.2.10. Neka su G i H grupe i neka je $H \leq G$. Grupa H je *normalna* podgrupa grupe G, u oznaci $H \triangleleft G$, ako i samo ako za svako $a \in G$ važi:

$$a^{-1}Ha = H, \text{ tj. } aH = Ha.$$

Primer 1.2.6. Neka je $G = \mathbb{Z}_6 = \{0, 1, 2, 3, 4, 5\}$ i neka je $H = \{0, 2, 4\}$. Zbog komutativnosti sabiranja u $\mathbb{Z}$ važi $g + H = H + g$, za svako $g \in G$. Sledi da je H normalna podgrupa grupe G, tj. $H \triangleleft G$.

Definicija 1.2.11. *Dekartov proizvod* skupova A i B definišemo kao skup svih uređenih parova kod kojih je prvi član iz skupa A, a drugi član iz skupa B:

$$A \times B = \{(a, b) | a \in A, b \in B\}.$$

Primer 1.2.7. Neka je skup $G = \{1, 2, 3\}$ i neka je skup $H = \{4, 5\}$. Dekartov proizvod skupova G i H je:

$$G \times H = \{(1, 4), (1, 5), (2, 4), (2, 5), (3, 4), (3, 5)\}.$$

Lema 1.2.2. Neka su $S_1, S_2, ..., S_n$ konačni skupovi. Tada važi:

$$|S_1 \times S_2 \times ... \times S_n| = |S_1| \cdot |S_2| \cdot ... \cdot |S_n|.$$

1.3 Tipovi grupa

Definicija 1.3.1. Grupa G je *ciklična* ako postoji neko $g \in G$ koje zadovoljava uslov $G = \{g^n | n \in \mathbb{Z}\}$. Element g se zove *generator* grupe G u oznaci $G = \langle g \rangle$. Ciklična grupa reda n označava se sa C_n.

Primer 1.3.1. Grupa $G = \mathbb{Z}_6 = \{0, 1, 2, 3, 4, 5\}$ je ciklična grupa sa operacijom sabiranja. Primer njene ciklične podgrupe je $\langle 2 \rangle = \{0, 2, 4\}$.

Primer 1.3.2. Neka je $G = \mathbb{Z}$ skup celih brojeva. Skup G sa operacijom sabiranja je ciklična grupa generisana elementom 1, jer se svaki element iz G može dobiti sabiranjem broja 1 dovoljan broj puta.

Definicija 1.3.2. *Permutacija* skupa G je *1-1* i *na* preslikavanje koje slika G u samog sebe.

Definicija 1.3.3. *Ciklus* je permutacija elemenata skupa X, $|X| = n$, za koju važi $x_1 \mapsto x_2 \mapsto ... \mapsto x_n$, gde je $x_i \in X$.

Svaka permutacija se može zapisati pomoću *ciklične notacije*, tj. može se predstaviti kao proizvod njenih ciklusa. Elementi koji se slikaju sami u sebe se preskaču, dok se jedinica predstavlja sa (1).

Primer 1.3.3. Neka je $X = \{1, 2, 3, 4, 5\}$. Permutacija $\theta : X \to X$ za koju važi $\theta(1) = 2, \theta(2) = 4, \theta(4) = 3, \theta(3) = 5$ i $\theta(5) = 1$ je ciklus. Njegova ciklična notacija je (12435).

Ciklusi se predstavljaju u onoj varijanti u kojoj je na prvom mestu najmanji element.

Definicija 1.3.4. Ciklus $(a_1, a_2, ..., a_k)$ naziva se *ciklus dužine k*, gde su elementi a_i različiti.
Parne permutacije su kompozicija parnog broja ciklusa dužine 2, dok su *neparne permutacije* kompozicija neparnog broja ciklusa dužine 2.

Primer 1.3.4. Neka je permutacija $\rho = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 1 & 6 & 4 & 5 & 3 \end{pmatrix}$.
U cikličnoj notaciji permutacija ρ se može predstaviti sa $(12)(36)(4)(5)$, tj. sa $(12)(36)$.
Permutacija ρ je parna.

Definicija 1.3.5. *N*eka je permutacija $\rho \in S_n$ takva da se može predstaviti kao kompozicija r ciklusa dužine 2. Tada se definiše znak permutacije $\text{sgn}(\rho) : S_n \to \{0, 1\}$ na sledeći način:

$$\text{sgn}(\rho) = (-1)^r \ .$$

Definicija 1.3.6. *Grupa svih permutacija* skupa S je skup svih permutacija skupa S u odnosu na kompoziciju preslikavanja i označava se sa S_x. Svaka pogrupa grupe S_x je takođe grupa permutacija, a ako je njen red konačan broj n, tada je ona *grupa permutacija stepena n.*

Primer 1.3.5. Neka je $T = \{1, 2, 3\}$ i neka je $\rho : T \to T$ permutacija takve da je $\rho(1) = 2, \rho(2) = 3$ i $\rho(3) = 1$. Permutacija ρ se može zapisati sa:
$\rho = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}$ ili u cikličnoj notaciji $\rho = (123)$.
Skup svih permutacija skupa T je:

$$S_T = \Big\{ \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix} \Big\}.$$

U cikličnoj notaciji je $S_T = (1), (23), (12), (123), (132), (13)$.
Jedinični element skupa S_T je permutacija $e = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix} = (1)$.

Definicija 1.3.7. Grupa svih parnih permutacija naziva se *alternirajuća grupa* i označava se sa A_n.

Definicija 1.3.8. Neka je G grupa i neka važi $H \triangleleft G$. *Faktor grupa* grupe G je grupa $G/H = \{aH | a \in G\}$ sa operacijom $(aH)(bH) = abH$ za $a, b \in G$.

Primer 1.3.6. Neka je $G = \mathbb{Z}_6 = \{0, 1, 2, 3, 4, 5\}$ i neka je $H = \{0, 2, 4\}$. Faktor grupa grupe G je grupa:

$$\begin{aligned} G/H &= \{a + H | a \in G\} \\ &= \{(0+\{0,2,4\}), (1+\{0,2,4\}), (2+0,2,4), (3+\{0,2,4\}), (4+\{0,2,4\}), \\ &\quad (5+\{0,2,4\})\} \\ &= \{\{0,2,4\}, \{1,3,5\}, \{2,4,0\}, \{3,5,1\}, \{4,0,2\}, \{5,1,3\}\} \\ &= \{\{0,2,4\}, \{1,3,5\}\}. \end{aligned}$$

1.4 Izomorfizmi

Definicija 1.4.1. Preslikavanje ϕ koje slika grupu $(G, *)$ u grupu $(H, \circ)$ je *homomorfizam* ako ϕ čuva operacije grupe, tj. ako važi $\phi(a * b) = \phi(a) \circ \phi(b)$ za svako $a, b \in G$.

Primer 1.4.1. Neka su $G = S_4$ i $H = \{1, -1\}$ grupe sa operacijom množenja i neka je preslikavanje $\theta : G \to H$ definisano sa:

$$\theta(a) = \begin{cases} 1 & a \text{ je parna permutacija,} \\ -1 & a \text{ je neparna permutacija,} \end{cases}$$

za svako $a \in G$. Kako bismo pokazali da je θ homomorfizam moramo proveriti 4 moguća slučaja u odnosu na parnost a i b:

1. Ako su a i b parne permutacije, onda je $\phi(ab) = \phi(a)\phi(b) = (1)(1) = 1$,
2. Ako je a parna permutacija, a b neparna permutacija, onda je $\phi(ab) = \phi(a)\phi(b) = (-1)(1) = -1$,

3. Ako je a neparna permutacija, a b parna permutacija, onda je $\phi(ab) = \phi(a)\phi(b) = (1)(-1) = -1$,

4. Ako su a i b neparne permutacije, onda je $\phi(ab) = \phi(a)\phi(b) = (-1)(-1) = 1$.

Iz ovoga jasno sledi da se parne permutacije slikaju u 1, dok se neparne permutacije slikaju u -1, stoga je θ homomorfizam.

Definicija 1.4.2. Homomorfizam grupa G i H $\phi : G \to H$ koji je *1-1* i *na* zove se *izomorfizam*. Ako postoji takav homomorfizam ϕ onda su grupe G i H izomorfne u oznaci $G \cong H$.

Primer 1.4.2. Neka su $G = \mathbb{Z}$ i $H = 2\mathbb{Z}$ grupe sa operacijom sabiranja i $\phi(a) = 2a$ za svako $a \in \mathbb{Z}$. Tada je $\phi : \mathbb{Z} \to 2\mathbb{Z}$ izomorfizam.

Definicija 1.4.3. Izomorfizam grupe G u samu sebe je *automorfizam*. Skup svih automorfizama grupe G označava se sa $\mathrm{Aut}(G)$.

Primer 1.4.3. Neka je G bilo koja grupa sa operacijom sabiranja. Tada je preslikavanje grupe G u samu sebe, koje slika svaki element grupe G u samog sebe, jedan automorfizam.

Definicija 1.4.4. Neka su G i H grupe i neka je preslikavanje $f : G \to H$ homomorfizam. *Jezgro* grupe G je skup $\ker(f) = \{g \in G | f(g) = e_H\}$, gde je e_H jedinični element grupe H.

Primer 1.4.4. Neka je preslikavanje $f : \mathbb{Z} \to \mathbb{Z}_n$ definisano sa $a \mapsto a \mod n$. Jedinični element $\mathbb{Z}_n$ je 0 pa sledi da je $\ker(f) = \{a \in \mathbb{Z} | a = bn, b \in \mathbb{Z}\} = n\mathbb{Z}$.

Lema 1.4.1. Neka je G grupa sa operacijom $*$ i neka je H neprazan podskup skupa G. Grupa H je podgrupa grupe G ako važi:

$$a * b^{-1} \in H \text{ za svako } a, b \in H.$$

Dokaz. Dovoljno je da pokažemo da je H grupa, jer $H \subset G$:

1. asocijativnost sledi iz toga što je restrikcija asocijativne operacije asocijativna,
2. $a = x$, $b = x \Rightarrow a * b^{-1} = x * x^{-1} = e_H$,
3. $a = e$, $b = x \Rightarrow a * b^{-1} = e_H * x^{-1} = x^{-1}$,
4. $x, y \in H$, $a = x$ i $b = y^{-1} \Rightarrow a * b^{-1} = x * (y^{-1})^{-1} = x * y \in H$.

□

Lema 1.4.2. Neka je $f : G \to H$ homomorfizam za bilo koje dve grupe G i H. Tada je $\ker(f) \triangleleft G$ i $G/\ker(f)$ je grupa.

Dokaz. Zbog homomorfizma je $e_G \mapsto e_H$, pa je $\ker(f) \neq \emptyset$.
Da bismo pokazali da je $\ker(f)$ podgrupa grupe G dovoljno je da pokažemo da ako $a, b \in \ker(f)$ onda i $ab^{-1} \in \ker(f)$. Ako $a, b \in \ker(f)$, tada $f(a) = e_H$ i $f(b) = e_H$. Sledi $f(ab^{-1}) = f(a)f(b^{-1}) = f(a)f(b))^{-1} = e_H e_H^{-1} = e_H$. Na osnovu toga $ab^{-1} \in \ker(f)$.
Neka su $g \in G$ i $k \in \ker(f)$, pa je podgrupa $\ker(f)$ normalna pogrupa grupe G jer važi:

$$\begin{aligned} f(gkg^{-1}) &= f(g)f(k)(fg^{-1}) \text{ , jer je } f \text{ homomorfizam} \\ &= f(g)e_H f(g^{-1}) \text{ , po definiciji jezgra i jer } k \in \ker(f) \\ &= f(g)(f(g))^{-1} \\ &= e_H \end{aligned}$$

Sledi da $gkg^{-1} \in \ker(f)$, pa je po definiciji $\ker(f)$ normalna podgrupa.
$G/\ker(f)$ je grupa na osnovu definicije faktor grupe. □

Teorema 1.4.1. (*Prva teorema o izomorfizmu*) Neka je ϕ homomorfizam iz grupe G u grupu H. Preslikavanje iz $G/\ker(\phi)$ u $\phi(G)$ definisano sa $g\ker(\phi) \to \phi(g)$ je izomorfizam, tj. $G/\ker(\phi) \cong \phi(G)$.

Dokaz. Definišimo preslikavanje

$$\rho : G/\ker(\phi) \to \phi(G)$$
$$a\ker(\phi) \mapsto \phi(a).$$

Funkcija ρ je izomorfizam ako je dobro definisana, bijekcija i homomorfizam.

Pokažimo da je ρ dobro definisana:

$a\ker(\phi) = b\ker(\phi) \iff ab^{-1} \in \ker(\phi)$, na osnovu svojstava koseta

$\iff \phi(ab^{-1}) = e_H$, gde je e_H identitet grupe H

$\iff \phi(a)\phi(b^{-1}) = e_H$, jer je ϕ homomorfizam

$\iff \phi(a)(\phi(b))^{-1} = e_H$

$\iff \phi(a) = \phi(b).$

Pokažimo da je ρ *1-1*:

$\rho(a\ker(\phi)) = \rho(b\ker(\phi))$

$\Rightarrow \phi(a) = \phi(b)$, na osnovu definicije preslikavanja ρ

$\Rightarrow \phi(ab^{-1}) = e_H$

$\Rightarrow ab^{-1} \in \ker(\phi)$

$\Rightarrow a\ker(\phi) = \ker(\phi).$

Preslikavanje ρ je *na* jer ako $b \in \phi(G)$, tada uvek postoji $a \in G$ za koje važi $\phi(a) = b$ i $\rho(a\ker(\phi)) = \phi(a) = b$.

Konačno, ρ je homomorfizam:

$\rho((a\ker(\phi))(b\ker(\phi)) = \rho(ab\ker(\phi))$

$= \phi(ab)$

$= \phi(a)\phi(b)$, jer je ϕ homomorfizam

$= \rho(a\ker(\phi))\rho(b\ker(\phi)).$

Iz toga što je ρ homomorfizam sledi da je ρ i izomorfizam. □

Primer 1.4.5. Neka je $\phi : S_4 \to \mathbb{Z}_2$ definisano sa:

$$\phi(a) = \begin{cases} 0 & a \text{ parna permutacija,} \\ 1 & a \text{ neparna permutacija.} \end{cases}$$

Tada je $\operatorname{Im}\phi = \mathbb{Z}_2$ i jedinični element $\mathbb{Z}_2$ je 0.

Prema tome, jezgro preslikavanja ϕ $\ker(\phi) = \{\text{sve parne permutacije}\} = A_4$.

Na osnovu prve teoreme o izomorfizmu važi $S_4/A_4 \cong \mathbb{Z}_2$.

1.5 Proizvodi grupa

Definicija 1.5.1. Neka su G_1 i G_2 grupe. *Direktan proizvod grupa* G_1 i G_2 je skup $G_1 \times G_2$ sa operacijom $(g_1, g_2) \cdot (g_1', g_2') = (g_1 g_1', g_2 g_2')$ za $g_1, g_1' \in G_1$ i $g_2, g_2' \in G_2$.

Primer 1.5.1. Neka je $G_1 = \mathbb{Z}_2$ i $G_2 = \mathbb{Z}_2$. Njihov direktan proizvod je:

$$\begin{aligned} A &= G_1 \times G_2 \\ &= \mathbb{Z}_2 \times \mathbb{Z}_2 \\ &= \{(0,0), (0,1), (1,0), (1,1)\} \end{aligned}$$

sa odgovarajućom operacijom nad parovima.

Definicija 1.5.2. *Dejstvo* grupe G na skup X je preslikavanje $G \times X \to X$ za koje važi:

1. $ex = x$ za svako $x \in X$,
2. $gx \in X$ za svako $g \in G$ i $x \in X$,
3. $(mn)x = m(nx)$ za svako $m, n \in G$ i $x \in X$.

Primer 1.5.2. Neka je $G = S_4$ i $X = \{1, 2, 3, 4\}$. Primeri dejstava grupe G na skup X su:

1. $[(12)(34)]2 = 1$,
2. $[(1234)]3 = 4$,
3. $[(132)(12)]2 = 3$.

Definicija 1.5.3. Neka su G_1 i G_2 podgrupe grupe G. Grupa G je *poludirektan proizvod* svojih podgrupa u oznaci $G = G_1 \rtimes G_2$, ako važi:

1. $G = G_1 G_2 = \{g_1 g_2 | g_1 \in G_1, g_2 \in G_2\}$,
2. $G_1 \cap G_2 = e_G$, gde je e_G jedinica grupe G,
3. $G_1 \triangleleft G$.

Primer 1.5.3. Grupa S_n se može predstaviti kao poludirektan proizvod $S_n = A_n \rtimes \langle(12)\rangle$. Pre svega, $A_n \cap \langle(12)\rangle = e$. Neka je $a \in S_n$ i $b \in A_n$. Tada je $aba^{-1} \in A_n$ jer je $\operatorname{sgn}(b) = 1$. Za $\operatorname{sgn}(a) = s \in \{-1, 1\}$ važi $\operatorname{sgn}(aba^{-1}) = \operatorname{sgn}(a)\operatorname{sgn}(b)\operatorname{sgn}(a^{-1}) = s^2 = 1$. Prema tome, permutacija aba^{-1} je parna, tj. pripada A_n, iz čega proizilazi $A_n \lhd S_n$ i $S_n \cong A_n \rtimes \langle(12)\rangle$.

Definicija 1.5.4. Neka su G i H grupe i neka grupa H deluje na konačan skup X. Neka je $\{x_1, x_2, ..., x_k\} \in X$ fiksirano, gde je $k = |X|$. Ako je G^k direktan proizvod grupe G sa samom sobom k puta, tada je *ukršteni proizvod* grupa G i H grupa $G^k \wr H = G^k \rtimes H$, gde H deluje na G^k svojim dejstvom na X.

Primer 1.5.4. Neka je $G = \mathbb{Z}_m$, $H = S_n$ i $X = \{1, 2, ..., n\}$. Tada je $\mathbb{Z}_m^n \wr S_n$ ukršteni proizvod grupa G i H, gde je $\rho : S_n \to \operatorname{Aut}(\mathbb{Z}_m^n)$ definisano sa $\rho(\theta)(x_1, x_2, ..., x_n) = (x_{\theta(1)}, x_{\theta(2)}, ..., x_{\theta(n)})$. Grupa $\mathbb{Z}_m^n \wr S_n$ zove se *uopštena simetrična grupa.*
Primetimo da ukršteni proizvod zapravo samo međusobno menja mesta elementima iz $\mathbb{Z}_m^n$ na osnovu dejstva iz S_n.

Glava 2

Rubikova kocka

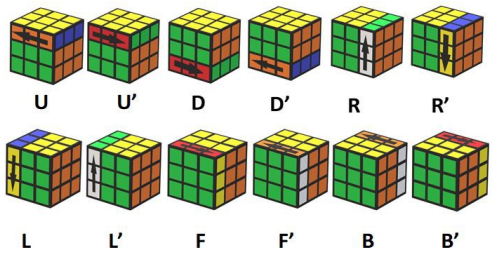
U ovoj glavi je definisana Rubikova kocka sa svojim karakteristikama i ograničenjima. Predstavljeno je obeležavanje kocke pomoću Singmasterove notacija, definisani su grupa Rubikove kocke i moguća stanja kocke. Tekst je u potpunosti zasnovan na materijalu iz projekta Lindzi Deniels [1].

2.1 Singmasterova notacija

Rubikova kocka $3 \times 3 \times 3$ ima 6 strana od po 9 kvadrata, koji će se ubuduće zvati polja. Centralno polje na svakoj strani je fiksirano. U početnom stanju su sve strane kocke jednobojne, tj. sva polja na istoj strani su obojena istom ovom bojom, i kaže se da je kocka *rešena*.
Kocka se rešava nizom poteza opisanim notacijom koju je definisao Dejvid Singmaster (eng. *David Singmaster*, [10]). Početna pretpostavka je da se kocka nalazi na ravnoj površi. Manipulacija kockom se vrši tako što se rotiraju njene strane u potezima od po 90 stepeni. Oznake svake strane kocke su:

- prednja strana – **F**,
- zadnja strana – **B**,
- desna strana – **R**,
- leva strana – **L**,
- gornja strana – **U**,
- donja strana – **D**.

Potez rotacije strane u smeru kazaljke na satu označava se sa M, dok se rotacija strane u suprotnom smeru označava sa M', gde je $M \in \{F, B, R, L, U, D\}$, videti sliku 2.1. Smer kazaljke na satu za odabranu stranu se određuje tako što se kocka okrene tako da je odabrana strana prednja strana. Transformacija nastala ponavljanjem poteza M k puta za redom, gde je k prirodan broj, označava se sa Mk.



Slika 2.1: Oznake poteza

Primer 2.1.1. Kombinacija poteza $RL2U$ rezultuje nizom rotacija strana u smeru kazaljke na satu: prvo desne, zatim dva puta leve i na kraju gornje strane kocke. Inverzna kombinacija je $U'L'2R'$.

2.2 Grupa Rubikove kocke

Svako stanje kocke se može opisati, na osnovu definicije 1.3.6, kao permutacija polja u odnosu na početno stanje. Broj polja Rubikove kocke je $6 \cdot 9 = 54$, odakle sledi da je grupa Rubikove kocke podgrupa, iz definicije 1.2.7, grupe permutacija od 54 elementa, pri tome nisu svi elementi različiti.

Definicija 2.2.1. Grupa permutacija $G = \langle F, B, R, L, U, D \rangle \leq S_{54}$ zove se *grupa Rubikove kocke*, gde je S_{54} iz definicije 1.3.6.

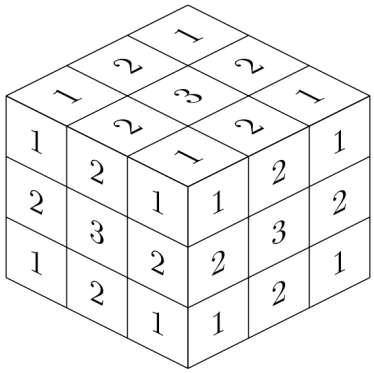
Dve glavne klasifikacije grupe Rubikove kocke su *legalna grupa Rubikove kocke* i *ilegalna grupa Rubikove kocke*. Razlikuju se u tome što je u ilegalnoj grupi dozvoljeno rastaviti kocku na delove i ponovo je sastaviti. Treba napomenuti da ni u jednoj grupi nije dozvoljena zamena boja poljima.

Rubikova kocka se sastoji od 26 manjih kockica, preciznije:

- 8 ugaonih kockica (tri vidljive strane),

- 12 ivičnih kockica (dve vidljive strane),
- 6 centralnih kockica (jedna vidljiva strana).

Uzimajući u obzir ograničenja Rubikove kocke, proizilazi da nisu moguće sve permutacije iz S_{54}.

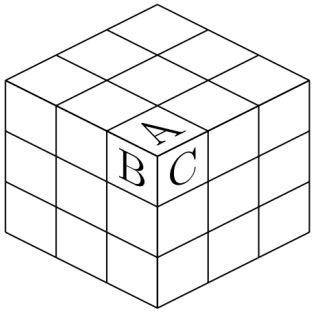


Slika 2.2: Tipovi polja na kocki

Sa slike 2.2 mogu se uočiti sledeća ograničenja:

1. centralna polja su fiksirana i nije ih moguće permutovati,
2. ivične kockice se moraju slikati u ivične kockice,
3. ugaone kockice se moraju slikati u ugaone kockice.

Svaka permutacija koja ne zadovoljava neki od ovih uslova je fizički nemoguća.



Slika 2.3: Orijentacija ugaone kockice

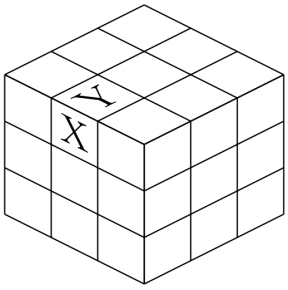
Sa slike 2.3 vidi se da se polje A nalazi na gornjoj strani kocke, polje B na levoj i polje C na prednjoj. Nizom poteza je moguće zarotirati ugaonu kockicu na kojoj se nalaze pomenuta polja tako da ona međusobno zamene mesta, npr. da se polje A nalazi na mestu polja B, polje B na mestu polja C i polje C na mestu polja A. Iz ovoga se može zaključiti da polja svake ugaone kockice pripadaju cikličnoj grupi C_3, iz definicije 1.3.1.
Kocka ima ukupno 8 ugaonih kockica, pa se orijentacija svake od njih može opisati cikličnom grupom reda 3, a orijentacija svih njih zajedno grupom:

$$C_3 \times C_3 \times C_3 \times C_3 \times C_3 \times C_3 \times C_3 \times C_3 = C_3^8.$$

S obzirom da svaka ugaona kockica može doći na mesto bilo koje druge ugaone kockice, a ima ih ukupno 8, ukupan broj njihovih kombinacija može se predstaviti grupom S_8.

Lema 2.2.1. Pozicija ugaonih polja na Rubikovoj kocki može se opisati pomoću grupe $C_3^8 \wr S_8$.

Dokaz. Lema sledi iz definicije ukrštenog proizvoda, definicija 1.5.4, i toga da se ugaona kockica može opisati pomoću njene pozicije na kocki i orijentacije njenih polja u C_3. □



Slika 2.4: Orijentacija ivične kockice

Sa slike 2.4 se vidi da se polje X nalazi na levoj, a polje Y na gornjoj strani kocke. Ivična kockica na kojoj se nalaze ova dva polja može se zarotirati tako da polja međusobno zamene mesta, pa u terminologiji grupa ona pripadaju grupi C_2. S obzirom da kocka ima ukupno 12 ivičnih kockica, orijentacija njihovih polja se može

predstaviti grupom:

$$C_2 \times C_2 \times C_2 \times C_2 \times C_2 \times C_2 \times C_2 \times C_2 \times C_2 \times C_2 \times C_2 \times C_2 = C_2^{12}.$$

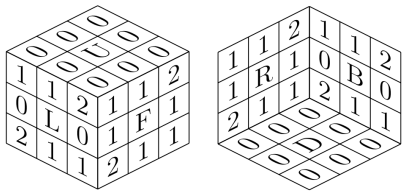
Svaka ivična kockica može doći na mesto bilo koje druge ivične kockice, pa se ukupan broj kombinacija svih ivičnih kockica može predstaviti grupom S_{12}.

Lema 2.2.2. Pozicija ivičnih polja na Rubikovoj kocki može se opisati pomoću grupe $C_2^{12} \wr S_{12}$.

Dokaz. Lema sledi iz definicije ukrštenog proizvoda i toga da se ivična kockica može opisati pomoću njene pozicije na kocki i orijentacije njenih polja u C_2. □

2.3 Fundamentalne teoreme teorije kocki

Iz prethodnih lema proističe da se pozicije ivičnih i ugaonih kockica mogu opisati odgovarajućim k-torkama, gde je k respektivno 12 i 8. Za određivanje elemenata k-torki potrebno je fiksirano obeležavanje svih polja na kocki, takvo da promenu orijentacije kockica oslikava promenom vrednosti njihovih polja.



Slika 2.5: Fiksirana obeležja polja odgovarajućim orijentacionim brojevima

Za obeležavanje polja sa slike 2.5 prilikom svakog poteza važi:

- orijentacioni broj ivičnih kockica je promenjen za 0 mod 2 ili 1 mod 2,
- orijentacioni broj ugaonih kockica je promenjen za 1 mod 3 ili 2 mod 3,
- orijentacioni broj ugaonih kockica se ne menja prilikom poteza U i D.

U nastavku ćemo povremeno ivične kockice označavati sa $x - y$, a ugaone sa $x - y - z$, gde $x, y, z \in \{F, B, R, L, U, D\}$.

Lema 2.3.1. Ciklusi parnog reda su neparne permutacije, dok su ciklusi neparnog reda parne permutacije.

Dokaz. Ciklus reda n, gde je n prirodan broj, se može zapisati na sledeći način:

$$(12 \cdots n) = (1n) \cdots (13)(12).$$

Sledi da je on proizvod $n-1$ permutacije, pa ako je n paran broj, tj. pomenuti ciklus je parnog reda, onda je on neparna permutacija po definiciji. Analogno važi za parne permutacije kada je n neparan broj. □

Teorema 2.3.1. (*Prva Fundamentalna teorema o teoriji kocki* [6]) Neka je $v \in C_3^8$, $r \in S_8$, $w \in C_2^{12}$ i $s \in S_{12}$. Četvorka (v, r, w, s) predstavlja moguće stanje kocke ako i samo ako:

1. $\text{sgn}(r) = \text{sgn}(s)$,
2. $v_1 + v_2 + ... + v_8 = 0 \mod 3$,
3. $w_1 + w_2 + ... + w_{12} = 0 \mod 2$.

Dokaz. ($\Rightarrow$) Neka je $v \in C_3^8$, $r \in S_8$, $w \in C_2^{12}$, $s \in S_{12}$ i $g \in G$, gde je g stanje kocke različito od početnog predstavljeno sa (v, r, w, s). Stanje g se može opisati pomoću niza poteza $g = M_1 M_2 ... M_n$, gde $M_i \in \{F, B, R, L, U, D\}$.

- Svaki potez kocke pomeri tačno 4 ivične i tačno 4 ugaone kockice, tj. isti broj ivičnih i ugaonih kockica biva pomeren pri svakom potezu. Sledi da se svaki potez može predstaviti ciklusom dužine 4, koji je neparna permutacija, tj. permutacija znaka -1. Dakle, za svako g važi:
$$\text{sgn}(r) = \prod_{k=1}^{n} \text{sgn}(M_i) = \text{sgn}(s).$$
- U slučaju da je $M_i \in \{U, D\}$, v ostaje nepromenjeno. Ukoliko je $M_i \in \{R, L, F, B\}$, tada se dve ugaone kockice pomeraju. Jedna je pomerena sa strane U, dok je druga pomerena na stranu U. Iz ovoga proističe da su komponente v ili umanjene za 1 $\mod 3$ ili uvećane za 1 $\mod 3$, respektivno, pa za svako R, L, F, B važi $v_1 + v_2 + ... + v_8 = 1 - 1 = 0 \mod 3$. Odatle sledi da je $v_1 + v_2 + ... + v_8 = 0 \mod 3$ za svaki potez g.
- U svakom potezu g uvek se rotiraju 4 ugaone kockice, tj.
$$w_1 + w_2 + ... + w_{12} = 4 \mod 2.$$

($\Leftarrow$) Neka je $A = (v, r, w, s)$ četvorka koja zadovoljava uslove (1), (2) i (3).
Uslov (1) kaže da je $\text{sgn}(s) = \text{sgn}(r)$, tj. da su permutacije ivičnih i ugaonih kockica uvek iste parnosti. Bez umanjena opštosti važi da je $\text{sgn}(s) = \text{sgn}(r) = 1$, tj. da su permutacije parne. Ukoliko su one neparne dodavanjem bilo kog poteza iz skupa $\{F, B, R, L, U, D\}$ postaju parne.

Posmatrajmo cikluse ugaonih kockica dužine 3. Transformacija $M = RB'RF2R'BRF2R2$ redom permutuje samo kockice U-F-L, U-F-R i U-B-R, dok ostale kockice ostaju na svom mestu. Označimo kockicu U-F-L sa a_1, kockicu U-F-R sa a_2 i kockicu U-B-R sa a_3,a ostale ugaone kockice sa a_4, a_5, a_6, a_7, a_8. Postoji transformacija kocke x koja se sastoji od najviše 2 poteza iz $\{F, B, R, L, U, D\}$ i koja pomera neko a_i na mesto a_3, dok a_1 i a_2 ostaju na svojim mestima. Neka je xMx^{-1} transformacija koja kreira ciklus dužine 3 (a_1, a_2, a_i), za $i \in \{4, 5, 6, 7, 8\}$. Na ovaj način se generišu sve parne permutacije ugaonih kockica, pa sledi da postoji odgovarajuća transformacija x koja će vratiti sve ugaone kockice u njihovo početno stanje.

Dalje, posmatrajmo cikluse ivičnih kockica dužine 3. Neka je transformacija $M^* = R2UFB'R2F'BUR2$. Označimo kockicu U-F sa b_1, kockicu U-B sa b_2, kockicu U-R sa b_3, dok su ostale kockice iz skupa $\{a_i | i \in \{4, 5, ..., 12\}\}$. Transformacija M^* permutuje kockice b_1, b_2 i b_3 ostavljajući ostale kockice na njihovim mestima. Kao i u slučaju ugaonih kockica, postoji transformacija kocke y koja se sastoji od najviše 2 poteza iz $\{F, B, R, L, U, D\}$ i koja pomera neko b_i na mesto b_3, dok b_1 i b_2 ostaju na svojim mestima. Neka je yM^*y^{-1} transformacija koja kreira ciklus dužine 3 (b_1, b_2, b_i), za $i \in \{4, 5, 6, 7, 8, 9, 10, 11, 12\}$. Na ovaj način se generišu sve parne permutacije ivičnih kockica, pa sledi da postoji odgovarajuća transformacija z koja će vratiti sve ivične kockice u njihovo početno stanje.
Da bismo dokazali ovaj smer ostalo je da još rotiramo kockice tako da kocka bude rešena.
Iz uslova (2) sledi da broj rotacija ugaonih kockica u smeru kazaljke mora biti isti broju rotacija u suprotnom smeru.
Dalje, postoji transformacija $M_1 = (R'D2RB'U2B)2$ koja će zarotirati samo dve ugaone kockice, dok će ostale kockice ostati na svojim mestima. Ona će zarotirati kockicu U-F-R za 120 stepeni i kockicu B-D-F za -120 stepeni. Transformacija M_1 se može prilagoditi tako da rotira bilo koje dve ugaone kockice.

Prvi korak u rešavanju kocke je rotiranje parova ugaonih kockica dok ne postignu početne orijentacije svojih polja. Broj preostalih ugaonih kockica koje nemaju početnu orijentaciju je deljiv sa 3, jer $\sum_{i=1}^{8} v_i = 0 \mod 3$, tj. za njihovo početno orijentisanje biće potrebne 3 rotacije u smeru kazaljke na satu ili u obnutom smeru. Obeležimo te 3 kockice sa c_1, c_2 i c_3. Transformacija $M_1^* = L'D2LBD2B'UBD2B'L'D2LU'$ će pravilno orijentisati jednu od preostale 3 kockice, recimo c_1, dok će drugu, recimo c_2, rotirati u suprotnom smeru od smera trenutne rotacije preostale kockice, u ovom slučaju c_3. Sledi da se preostale kockice c_2 i c_3 mogu orijentisati pomoću transformacije M_1, pa su sve ugaone kockice orijentisane kao u početnom stanju kocke.
Iz uslova (3) sledi da broj okretanja ugaonih kockica u odnosu na početno stanje kocke mora biti paran, tj. broj okrenutih ivičnih kockica će uvek biti paran. Transformacija $M_2 = LFR'F'L'U2RURU'R2U2R$ okreće tačno 2 ivične kockice, dok ostale ostaju u svom trenutnom stanju. Neka te dve kockice budu U-F i U-R, iako isto važi za bilo koje dve ivične kockice. S obzirom da kocka ima paran broj ivičnih kockica, ovom transformacijom se sve kockice mogu vratiti u početno stanje.

Sledi da je A rešivo stanje, pa je (v, r, w, s) legalno stanje Rubikove kocke. □

Teorema 2.3.2. (*Druga Fundamentalna teorema o teoriji kocki* [6]) Transformacija Rubikove kocke je moguća ako i samo ako:

1. ukupan broj ivičnih i ugaonih ciklusa istih dužina je paran
2. broj ciklusa koji rotiraju ugaone kockice u smeru kazljke i u suprotnom smeru je jednak po modulu 3
3. broj reorijentisanih ivičnih kocki u odnosu na njihovu početnu poziciju je paran

Dokaz. ($\Rightarrow$) Neka je M transformacija koja kocku dovodi od početnog stanja do stanja $g = (v, s, w, r)$, gde je $v \in C_3^8$, $r \in S_8$, $w \in C_2^{12}$ i $s \in S_{12}$.

- Uslov (1) teoreme 2.3.1 kaže da $\text{sgn}(r) = \text{sgn}(s)$, što znači da je M parna permutacija. Sledi da je ukupan broj ivičnih i ugaonih ciklusa paran.
- Za bilo koju transformaciju kocke važi da svaka ugaona kockica može ostati iste orijentacije, biti zarotirana u smeru kazaljke ili suprotnom smeru. Uslov

(2) teoreme 2.3.1 kaže da je $\sum_{i=1}^{8} v_i = 0 \mod 3$, što je moguće ako i samo ako je broj ciklusa koji rotiraju ugaone kockice u smeru kazaljke i u suprotnom smeru jednak po modulu 3.

- Za bilo koju transformaciju kocke važi da svaka ivična kockica može ostati iste orijentacije ili biti zarotirana. Uslov (3) teoreme 2.3.1 kaže da je $\sum_{i=1}^{12} w_i = 0 \mod 2$, što znači da ako je zarotirana jedna ivična kockica onda mora biti zarotirana još jedna kockica kako bi suma njihovih orijentacija bila 0.

($\Leftarrow$) Neka važe uslovi (1), (2) i (3) za stanje kocke g. Na osnovu teoreme 2.3.1 postoji transformacija M koja dovodi kocku iz početnog stanja u stanje g, kao i inverzna transformacija M^{-1} koja dovodi kocku iz stanja g u početno stanje. Po pretpostavci M i M^{-1} zadovoljavaju uslove (1), (2) i (3), sledi da su one validne transformacije. □

Glava 3

Algoritmi za rešavanje Rubikove kocke

U ovoj glavi je predstavljeno nekoliko metoda za rešavanja Rubikove kocke baziranih na metodi podgrupa, kao i detaljan opis Kociembina metode. Predstavljene metode su pogodne za programsku implementaciju jer se oslanjaju na obradu velike količine slučajeva. Opis prve tri metode je u potpunosti zasnovan na radu profesora Dejvida Džojnera (eng. *David Joyner*, [6]).

3.1 Metoda podgrupa

Jedan od načina da se reši Rubikova kocka pomoću računara je kreiranje niza podgrupa [6]:

$$G_n \subset G_{n-1} \subset ... \subset G_1 \subset G_0 = G,$$

gde je $G = \langle F, B, R, L, U, D \rangle$ grupa Rubikove kocke. Dalje se može implementirati sledeća strategija:

1. Zadato stanje Rubikove kocke predstaviti elementom $g_0 \in G$, gde je stanje definisano teoremom 2.3.1,

2. Odrediti skup svih predstavnika koseta, iz definicije 1.2.8, grupe G_{k+1}/G_k:
   $$G_{k+1}/G_k = \cup_{i=1}^{r_k} g_{k+1,i} G_{k+1}, \text{ za } r_k > 1, 0 \leq k < n.$$
   Primetiti da važi $m_{n-1} = 1, g_{n,1} = 1$.

3. Dalje se primenjuje metoda matematičke indukcije:

- (Baza indukcije) Ako $g_0 \in g_{1,i}G_1$, $i \in \{1, ..., n_1\}$, onda važi $g_1 = g_{1,i}$ i $g_1^{'} = g_1^{-1}g_0$, gde $g_1^{'} \in G_1$.
- (Indukcijska hipoteza) Ako je $g_k^{'} \in G_k$ definisano i ako $g_k^{'} \in g_{k+1,i}G_k$, $j \in \{1, ..., n_1\}$, neka je tada $g_{k+1} = g_{k+1,j}$ i $g_{k+1}^{'} = g_{k+1}^{-1}g_k^{'}$, gde $g_{k+1}^{'} \in G_{k+1}$.
- (Indukcijski korak) Iz prethodnog sledi $1 = g_n^{-1}g_{n-1}^{-1}g_{n-2}^{-1}...g_1^{-1}g_0$, pa važi:
$$g_0 = g_1g_2...g_{n-1}g_n.$$

Što su predstavnici koseta kraći i prostiji potezi na Rubikovoj kocki, to je rešenje $g_0 = g_1g_2...g_{n-1}g_n$ kraće, pa je metoda brža ako prvobitno izabran niz podgrupa G_i obezbeđuje baš te uslove.

3.2 Metoda „uglovi-ivice”

Metoda „uglovi-ivice” spada u metode podgrupa. Neka važi sledeći izbor podgrupa:

- G_1 – podgrupa koja ne permutuje nijedan ugao,
- G_2 – podgrupa koja ne permutuje ni ivice ni uglove,
- G_3 – podgrupa koja ne permutuje ni ivice ni uglove i pritom ne menja orijentaciju uglovima,
- $G_4 = \{1\}$.

Tada sledi:
$$G_4 = \{1\} \subset G_3 \subset G_2 \subset G_1 \subset G_0 = G.$$

Ideja je sledeća:

1. Zadato stanje Rubikove kocke predstaviti elementom $g_0 \in G$.
2. Označimo sa g_1 transformaciju kocke iz teoreme 2.3.2 koja permutuje sve ugaone kockice na početne pozicije, i pritom ih verovatno rotira. Tada je $g_1^{'} = g_1^{-1}g_0 \in G_1$.
3. Označimo sa g_2 transformaciju kocke koja permutuje sve ivične kockice na početne pozicije, i pritom verovatno rotira ostale ugaone i ivične kockice, dok ostale kockice ostaju na svojim mestima. Tada je $g_2^{'} = g_2^{-1}g_1^{'} \in G_2$.

4. Označimo sa g_3 transformaciju kocke koja rotira sve ugaone kockice u njihove početne orijentacije, dok ostale kockice ostaju na svojim mestima. Tada je $g_3' = g_3^{-1} g_2' \in G_3$.

5. Označimo sa g_4 transformaciju kocke koja rotira sve ivične kockice u njihove početne orijentacije, dok ostala polja ostaju na svojim mestima.

6. Rešenje kocke je transformacija $g_0 = g_1 g_2 g_3 g_4$.

3.3 Tistelvejtova metoda

Ovu metodu je osmislio *Morvin Tistelvejt*[1] i ona važi za jednu od najboljih metoda koje spadaju u grupu metoda podgrupa [5].
Neka važi sledeći izbor podgrupa:

- $G_1 = \langle R, L, F, B, U2, D2 \rangle$,
- $G_2 = \langle R, L, F2, B2, U2, D2 \rangle$,
- $G_3 = \langle R2, L2, F2, B2, U2, D2 \rangle$,
- $G_4 = \{1\}$.

Rubikova $3 \times 3 \times 2$ „domino" kocka je verzija Rubikove $3 \times 3 \times 3$ kocke, sa tim da je jedan horizontalni red izbačen. Svako njeno stanje se može opisati Rubikovom $3 \times 3 \times 2$ „domino" grupom, čiji je red $(8!)^2/4$, zato što je samo $1/4$ stanja moguće razlikovati duž njene U-D ose.
Grupa G_2 je izomorfna sa Rubikovom $3 \times 3 \times 2$ „domino" grupom, i važi da je $|G_2| = (8!)^2 \cdot 12$.
G_3 je grupa „kvadrata", prema Singmasteru, i njen red je $2^{13} \cdot 3^4$.
Tistelvejt je pokazao da važi sledeće:

1. Skup $\{g_{1,i} | 1 \leq i \leq n_1\}$ je skup svih predstavnika koseta grupe G/G_1, gde $g_{1,i}$ sadrži najviše 7 poteza za $n_1 = 2048$.

2. Skup $\{g_{2,i} | 1 \leq i \leq n_2\}$ je skup svih predstavnika koseta grupe G_1/G_2, gde $g_{2,i}$ sadrži najviše 13 poteza za $n_2 = 1.082.565$.

[1] Morvin Tistelvejt (eng. *Morwen Thistlethwaite*) je američki profesor matematike. Najveći doprinos dao je teoriji čvorova i grupi Rubikove kocke.

3. Skup $\{g_{3,i} | 1 \leq i \leq n_3\}$ je skup svih predstavnika koseta grupe G_2/G_3, gde $g_{3,i}$ sadrži najviše 15 poteza za $n_3 = 29.400$.

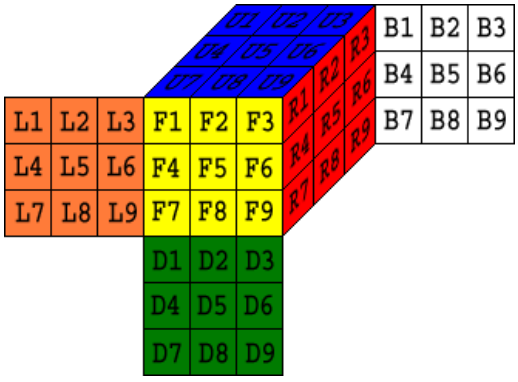
4. Skup $\{g_{4,i} | 1 \leq i \leq n_4\}$ je skup svih predstavnika koseta grupe G_3/G_4, gde $g_{4,i}$ sadrži najviše 17 poteza za $n_4 = 663.552$.

Sledi da se Rubikova kocka može rešiti u najviše $7 + 13 + 15 + 17 = 52$ poteza.

3.4 Kociembina metoda

Početkom 1990. godine *Herbert Kociemba*[2] je unapredio Tistelvejtov algoritam i nazvao ga „dvofazni algoritam", poznat i pod nazivom „Kociembina metoda" [6]. Detaljno objašnjenje profesora Kociembe, na osnovu kog je u potpunosti napisana ova glava, nalazi se na njegovom sajtu: `http://kociemba.org/cube.htm`.

Predstavljanje polja



Slika 3.1: Označavanje kocke u metodi Kociemba

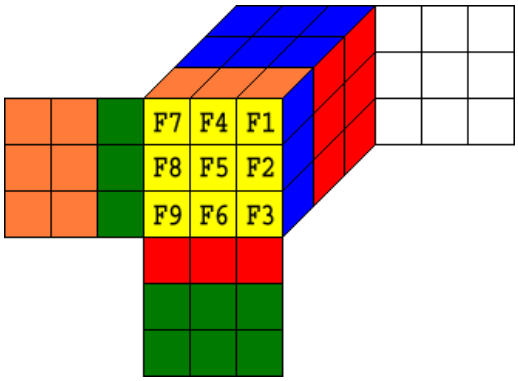
Kada se na kocku prikazanu na slici 3.1 primeni potez F, dobija se stanje prikazano na slici 3.2.

Neka je $G = \langle F, B, R, L, U, D \rangle$ grupa Rubikove kocke. Postoje dva načina za prikaz poteza F, permutacijama $\rho : G \to G$ i $\theta : G \to G$:

$$\rho = \begin{pmatrix} \text{F1} & \text{F2} & \text{F3} & \text{F4} & \text{F5} & \text{F6} & \text{F7} & \text{F8} & \text{F9} \\ \text{F3} & \text{F6} & \text{F9} & \text{F2} & \text{F5} & \text{F8} & \text{F1} & \text{F4} & \text{F7} \end{pmatrix},$$

$$\theta = \begin{pmatrix} \text{F1} & \text{F2} & \text{F3} & \text{F4} & \text{F5} & \text{F6} & \text{F7} & \text{F8} & \text{F9} \\ \text{F7} & \text{F4} & \text{F1} & \text{F8} & \text{F5} & \text{F2} & \text{F9} & \text{F6} & \text{F3} \end{pmatrix}.$$

[2]Herbert Kociemba je profesor matematike i fizike u penziji. Poznat je po svom dvofaznom algoritmu za rešavanje Rubikove kocke i radu na projektu „Božjeg broja".

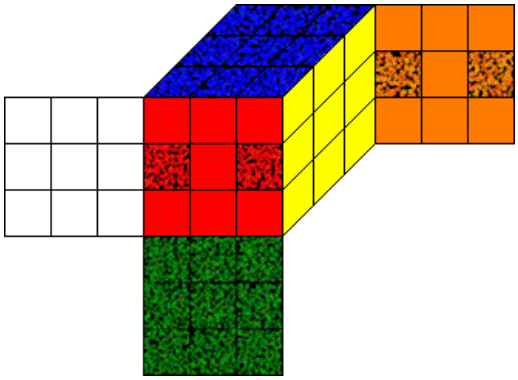


Slika 3.2: Stanje kocke nakon primene poteza F

Permutacija ρ slika polje u one polje čije je mesto zauzelo, dok permutacija θ slika polje u ono polje koje je zauzelo njegovo prvobitno mesto.
Obe permutacije se pišu u skraćenom obliku tako što se izuzme njihov domen koji je uvek (F1, F2, F3, F4, F5, F6, F7, F8):
ρ = (F3,F6,F9,F2,F5,F8,F1,F4,F7),
θ = (F7,F4,F1,F8,F5,F2,F9,F6,F3).
Predstavljanje pomoću permutacije ρ zove se „zamenjeno sa", dok se predstavljanje pomoću permutacije θ zove „zamenilo je".

Predstavljanje kockica

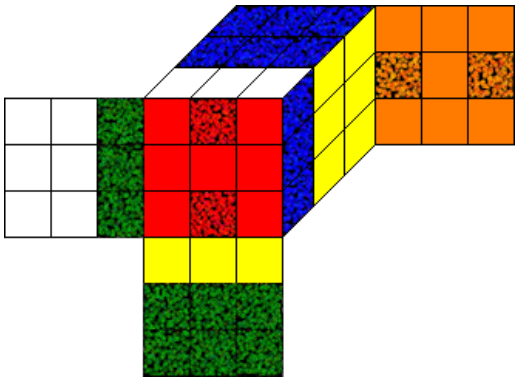
Ugaone kockice se skraćeno označavaju sa XYZ, gde su X, Y, Z strane kojima kockica pripada.



Slika 3.3: Referentne pozicije polja za orijentaciju

Označena polja na rešenoj kocki sa slike 3.3 su reference za orijentaciju.

Ako se ivična kockica pomeri na drugu poziciju, na njenu trenutnu orijentaciju se dodaje 0 ako se pozicija njenog označenog polja poklapa sa nekom od pozicija označenih polja u rešenom stanju kocke, u suprotnom se dodaje 1 $\mod 2$.
Ako se ugaona kockica pomeri na drugu poziciju, na njenu trenutnu orijentaciju se dodaje 0 ako se pozicija njenog označenog polja poklapa sa nekom od pozicija označenih polja u rešenom stanju kocke. U slučaju da je zarotirana u smeru kazaljke na satu dodaje se 1 $\mod 3$, a u suprotnom smeru 2 $\mod 3$.



Slika 3.4: Referentna polja nakon primene poteza F

Permutacija kockica sa slike 3.4 u predstavljanju „zamenilo je" glasi:
$\theta = \begin{pmatrix} \text{URF} & \text{UFL} & \text{ULB} & \text{UBR} & \text{DFR} & \text{DLF} & \text{DBL} & \text{DRB} \\ \text{UFL} & \text{DLF} & \text{ULB} & \text{UBR} & \text{URF} & \text{DFR} & \text{DBL} & \text{DRB} \end{pmatrix}$.
Da bi se pamtila i orijentacija svake kockice, potrebno je izmeniti permutaciju θ na sledeći način:
$\theta : G \to G \times O$,
gde je $O = \{0, 1, 2\}$ skup svih mogućih vrednosti orijentacija kockice.
Predstavljanje kockica sa slike je tada:
$\theta = \begin{pmatrix} \text{URF} & \text{UFL} & \text{ULB} & \text{UBR} & \text{DFR} & \text{DLF} & \text{DBL} & \text{DRB} \\ \text{UFL,1} & \text{DLF,2} & \text{ULB,0} & \text{UBR,0} & \text{URF,2} & \text{DFR,1} & \text{DBL,0} & \text{DRB,0} \end{pmatrix}$.
Zbog jednostavnosti za prikaz slikanja θ koje predstavlja potez $M \in \{F, B, R, L, U, D\}$ i kockicu ABC koja se slika u DEF, gde su A,B,C,D,E,F oznake strana Rubikove kocke, koristi se sledeća notacija:

- sa $X(\text{ABC}).c$ označava se kockica u koju se slika kockica ABC, tj. DEF,
- sa $X(\text{ABC}).o$ označava se orijentacija kockice DEF.

Ubuduće će potez M predstavljati i slikanje θ, zbog jednostavnosti.
Primer ove notacije u odnosu na sliku 3.4 je:

$$F(\text{URF}).c = \text{UFL}$$
$$F(\text{URF}).o = 1$$
$$F(\text{UFL}).c = \text{DLF}$$
$$F(\text{UFL}).o = 2$$

Neka su $A : G \to G \times O$ i $B : G \to G \times O$ permutacije sa orijentacijama. U ovoj notaciji njihov proizvod se zapisuje na sledeći način:

$$(A * B)(x).c = A(B(x).c).c \ ,$$
$$(A * B)(x).o = A(B(x).c).o + B(x).o \ .$$

Predstavljanje koordinata

Na nivou koordinata, permutacije sa orijentacijama se predstavljaju pomoću prirodnih brojeva.
Neka je potez $M \in \{F, B, R, L, U, D\}$. Koordinata orijentacije ugaonih kockica može se predstaviti sa brojem iz skupa $\{0, 1, ..., 2186\}$, gde je $2186 = 3^7 - 1$:

$$\sum_{i=0}^{6} 3^i M(C_i).o \ ,$$

gde su C_i ugaone kockice. U sumi se koristi 7 ugaonih kockica, jer je orijentacija osme definisana na osnovu orijentacije ostalih 7.
Ako se primeni potez R na rešenu kocku dobija se:

$$R = \begin{pmatrix} \text{URF} & \text{UFL} & \text{ULB} & \text{UBR} & \text{DFR} & \text{DLF} & \text{DBL} & \text{DRB} \\ \text{DFR,2} & \text{UFL,0} & \text{ULB,0} & \text{URF,1} & \text{DRB,1} & \text{DLF,0} & \text{DBL,0} & \text{UBR,2} \end{pmatrix}.$$

Za ovaj potez orijentacija kockica je:

$$2 \cdot 3^6 + 0 \cdot 3^5 + 0 \cdot 3^4 + 1 \cdot 3^3 + 1 \cdot 3^2 + 0 \cdot 3^1 + 0 \cdot 3^0 = 1494$$

Koordinata orijentacije ivičnih kockica se predstavlja analogno sa prirodnim brojem iz skupa $\{0, ..., 2047\}$, gde je $2047 = 2^{11} - 1$:

$$\sum_{i=0}^{10} 2^i M(C_i).o \ .$$

Jedna ivična kockica je izostavljena iz sume jer je njena orijentacija određena orijentacijom ostalih 11 ivičnih kockica.
Koordinata permutacije ugaonih kockica može se predstaviti sa brojem iz skupa $\{0, 1, ..., 40.319\}$, gde je $40.319 = 8! - 1$:

$$\sum_{i=1}^{7} i!X(C_i) \ ,$$

gde je preslikavanje $X : \{F, B, R, L, U, D\} \to \mathbb{N}$ i gde su C_i ugaone kockice. Neka je $A = \{URF, UFL, ULB, UBR, DFR, DLF, DBL, DRB\}$ skup svih ugaonih kockica iz domena poteza M. Neka je $B = \{M(URF).c, M(UFL).c, M(ULB).c, M(UBR).c, M(DFR).c, M(DLF).c, M(DBL).c, M(DRB).c\}$ skup svih ugaonih kockica iz kodomena poteza M. Slikanje X slika ugaonu kockicu C u broj ugaonih kockica čija je slika na nižoj poziciji u skupu B, a na višoj u skupu A u odnosu na sliku $M(C).c$ kockice C.

Za potez R primenjen na rešenoj kocki kooridnata permutacije je:
$1 \cdot 1! + 1 \cdot 2! + 3 \cdot 3! + 0 \cdot 4! + 1 \cdot 5! + 1 \cdot 6! + 4 \cdot 7! = 21021$.

Koordinata permutacije ivičnih kockica se predstavlja analogno sa prirodnim brojem iz skupa $\{0, ..., 479.001.600\}$, gde je $479.001.600 = 12! - 1$:

$$\sum_{i=1}^{11} i!X(C_i) \ ,$$

gde je preslikavanje X malopre definisano i gde su C_i ivične kockice.

Na osnovu prethodno definisanih koordinata svaka kockica se može predstaviti četvorkom (x_1, x_2, x_3, x_4) i važi:

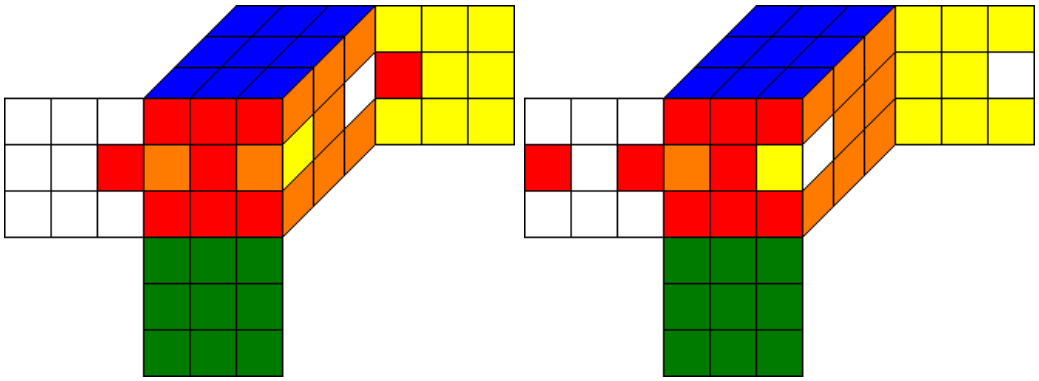
- $0 \leq x_1 < 3^7$,
- $0 \leq x_2 < 2^{11}$,
- $0 \leq x_3 < 8!$,
- $0 \leq x_4 < 12!$.

Ekvivalentne kocke i simetrija

Kocke sa slike 3.5 zapravo imaju ista stanja, jer kada se desna kocka rotira za 90 stepeni oko ose definisane sa centralnim poljima strana U i D, dobija se isto stanje kao na kocki sa leve strane, s tim što su strane obojene drugim bojama, tj. koristi se drugo bojenje kocke. S obzirom da je potreban isti broj koraka da se reše kocke istih stanja, a različitih bojenja, one se smatraju ekvivalentnim.

Neka je S_{U4} pomenuta rotacija kocke za 90 stepeni. Neka je A permutacija koja definiše stanje leve kocke i neka je B permutacija koja definiše stanje desne kocke. Tada je $B = S_{U4}^{-1} A S_{U4}$.

Permutacije kocke A i B su ekvivalentne ako postoji simetrija kocke S takva da je:



Slika 3.5: Primer ekvivalentnih kocki

$$B = S^{-1}AS \ .$$

Sve kocke koje su ekvivalentne pripadaju istoj klasi ekvivalencije koordinata.
Za svaku kocku postoji 48 ekvivalentnih kocki, zato što Rubikova kocka ima 48 simetrija, uključujući i refleksije. Sve su generisane proizvodom:
$S_{URF3}^{x_1} * S_{F2}^{x_2} * S_{U4}^{x_3} * S_{LR2}^{x_4}$,
gde:

- $x_1 \in \{0, 1, 2\}$,
- $x_2 \in \{0, 1\}$,
- $x_3 \in \{0, 1, 2, 3\}$,
- $x_4 \in \{0, 1\}$,
- S_{URF3} je rotacija kocke za 120 stepeni oko ose definsane ugaonim kockicama URF i DBL,
- S_{F2} je rotacija kocke za 180 stepeni oko ose definisane centralnim poljima strana F i B,
- S_{U4} je rotacija kocke za 90 stepeni oko ose definsane centralnim poljima strana U i D,
- S_{LR2} je refleksija kocke u odnosu na ravan definisanu stranama R i L.

Svaka simetrija kocke može se predstaviti prirodnim brojem iz skupa $\{0, 1, ..., 47\}$ na sledeći način:

$$2^4x_1 + 2^3x_2 + 2^2x_3 + 2^1x_4.$$

Koseti

Neka je G grupa Rubikove kocke, $H \leq G$ i $g \in G$. Svaka prethodno definisana koordinata kocke može se preslikati u desni koset grupe H definisan sa $Hg = \{hg | h \in H\}$, gde je podgrupa H definisana tipom koordinate.
Neka je C_0 podgrupa grupe Rubikove kocke definisana sa:

$$C_0 = \{g \in G | g(x).o = 0\} \ ,$$

za svako x iz skupa ugaonih kockica. Desni koset ove podgrupe je:

$$C_0 g = \{ag | a \in C_0\} \ .$$

Dakle, sve kockice iz koseta $C_0 g$ imaju iste orijentacije, definisane permutacijom g, i koordinatu orijentacije ugaonih kocki. Takođe, ako dve permutacije imaju istu koordinatu orijentacije ugaonih kocki, onda pripadaju istom kosetu.

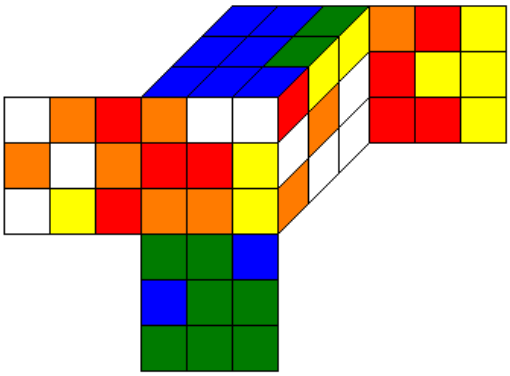
Koordinate i simetrija

Koordinate predstavljaju kosete, a svaki koset većinom sadrži mnogo permutacija. Neka je S simetrija kocke, P i Q dve različite permutacije kocke koje pripadaju istom kosetu i neka su permutacije $S^{-1}PS$ i $S^{-1}QS$ takve da pripadaju istom kosetu. Koordinate kocke se mogu preslikati u sim koordinate pomoću simetrije S ako za grupu $H \leq G$ važi $S^{-1}HS = H$. Stare koordinate će se ubuduće zvati grube koordinate.
Ubuduće će se koristiti simetrije kocke koje slikaju kockicke koje pripadaju stranama U i D u iste te strane. Tada svakoj klasi ekvivalencije sim koordinata pripada do 16 sim koordinata umesto 48.
Neka je indeks simetrije broj koji označava poziciju simetrije kocke u skupu svih simetrija kocke. Gruba koordinata x može biti zamenjena sim koordinatom $y = 16j+i$, gde je i indeks simetrije kojom je y nastala i j indeks klase ekvivalencije kojoj pripada y.

Koordinate algoritma „dve-faze”

Ukoliko se na rešenoj kocki koriste samo potezi iz skupa $\{U, U', D, D'\}$ dobije se podgrupa $G_1 = \langle U, D, R2, L2, F2, B2 \rangle$ grupe G.

U ovoj podgrupi orijentacije svih ugaonih i ivičnih kockica su 0. Takođe, sve 4 ivične kockice iz srednjeg sloja kocke ostaju u istom sloju, što se vidi i na slici 3.6.



Slika 3.6: Primer permutacije iz grupe G_1

U prvoj fazi algoritam traži transformaciju koja će dovesti kocku iz bilo kog stanja u stanje koje odgovara nekoj permutaciji iz grupe G_1. U ovoj fazi je svaka kocka definisana sa 3 koordinate:

- koordinatom orijentacije ugaonih kockica,
- koordinatom orijentacije ivičnih kockica,
- „UDSlice koordinatom".

„UDSlice koordinata" je broj iz skupa $\{0, 1, ..., 494\}$, gde je $494 = \frac{12 \cdot 11 \cdot 10 \cdot 9}{4!} - 1$, određen pozicijama 4 ivične kockice koje pripadaju srednjem sloju kocke. U ovom tipu koordinata se pamti samo položaj ivičnih kockica, boje su nebitne.

Sledi da se svako stanje kocke u prvoj fazi može predstaviti pomoću trojke (x_1, x_2, x_3), gde su x_1, x_2, x_3 prethodno definisane koordinate redom. Trojka ima vrednost $(0, 0, 0)$ ako i samo ako je stanje kocke permutacija iz grupe G_1. Ova faza ima prostornu složenost od $2187 \cdot 2048 \cdot 495 = 2.217.093.120$ različitih stanja.
U drugoj fazi algoritam traži rešenje kocke. U ovoj fazi je svaka kockica je definisana sa 3 koordinate:

- koordinatom permutacije ugaonih kockica,
- koordinatom permutacije ivičnih kockica,
- „UDSlice koordinatom".

Sledi da se svako stanje kocke u drugoj fazi može predstaviti pomoću trojke (x_1, x_2, x_3), gde su x_1, x_2, x_3 prethodno definisane koordinate redom. Trojka ima vrednost $(0, 0, 0)$ ako i samo ako je kocka u rešenom stanju.Ova faza ima prostornu složenost od $40.320 \cdot 40.320 \cdot 24 \cdot \frac{1}{2} = 19.508.428.800$ različitih stanja.

Tabele poteza

Neka su A i B dve permutacije kocke koje imaju jednu istu koordinatu x i neka je potez $M \in \{U, D, R, L, F, B\}$. Ako primenimo potez M na obe permutacije, rezultujuće permutacije će slikati koordinatu x u koordinatu x'. Neka grupa $H \leq G$ definiše kosete za koordinatu x, tada postoji permutacija $g \in G$ takva da $A, B \in Hg$. Prema tome permutacije AM i BM pripadaju $(Hg)M = H(gM)$, tj. pripadaju istim kosetima i imaju istu koordinatu x'.
Sledi da se svaki potez može čuvati u tabeli poteza, tj. dvodimenzionalnom nizu. To omogućava da ako se generišu tabele za sve poteze, prilikom rešavanja kocke vreme određivanja ishoda se znatno smanjuje.
Prethodno opisana tabela poteza koristi se za grube koordinate. Kod sim koordinata se generišu tabele poteza za predstavnike klasa ekvivalencije koordinata. Neka je $R(j)$ permutacija koja pripada klasi ekvivalencije koordinata sa indeksom j. Kada primenimo potez $M \in \{U, D, R, L, F, B\}$ na $R(j)$, rezultujuća permutacija će biti u nekoj drugoj klasi ekvivalencije indeksa k, pa postoji simetrija kocke S takva da je $R(j)M = S^{-1}R(k)S$. Prema tome, u tabelu poteza se kao ključ unosi par (j, M), a kao vrednost odgovarajuća sim koordinata $16 \cdot k + i$, gde je i indeks simetrije S.

Neka je P permutacije kocke i $S(i)$ simetrija kocke sa indeksom i, gde $i \in \{0, 1, ..., 15\}$. Da bi se pronašla sim koordinata za permutaciju P potrebno je primenjivati konjugaciju $S(i)PS(i)^{-1}$ i računati grube koordinate dok se one ne poklope sa nekom vrednošću iz tabele poteza grubih koordinata na poziciji k. Neka je to gruba koordinata $R(k)$. Tada je $S(i)PS(i)^{-1} = R(k)$ ekvivalentno sa $S(i)^{-1}R(k)S(i) = P$, što znači da permutacija P ima sim koordinatu $16 \cdot k + i$.
Neka je x sim koordinata i neka je potez $M \in \{U, D, R, L, F, B\}$. Na osnovu sim koordinate x poznat je indeks klase ekvivalencije koordinate j i indeks simetrije i. Koristeći asocijativnost grupe permutacija, za predstavnika klase ekvivalencije $R(j)$ važi:

$$(S(i)^{-1}R(j)S(i))M = (S(i)^{-1}R(j)S(i))M(S(i)^{-1}S(i)) =$$
$$= (S(i)^{-1}R(j))(S(i)MS(i)^{-1})S(i).$$

Za primenu poteza na osnovu sim koordinata potrebne su 3 tabele:

1. tabela sim poteza,
2. tabela poteza,
3. tabela sim množenja.

Neka je potez $M_1 = (S(i)MS(i)^{-1})$ i on se zapisuje u tabelu sim poteza. Kao ključ se unosi par (i, M), dok se kao vrednost unosi permutacija M_1. Sledi:

$$(S(i)^{-1}R(j))M_1S(i) = S(i)^{-1}(R(j)M_1)S(i).$$

Sim koordinata y za permutaciju $R(j)M_1$ se može pročitati iz tabele poteza. Na osnovu y poznat je indeks klase ekvivalencije koordinate j_1 i indeks simetrije i_1. Prema tome:

$$R(j)M_1 = S(i_1)^{-1}R(j_1)S(i_1)\ ,$$
$$S(i)^{-1}(R(j)M_1)S(i) = S(i)^{-1}(S(i_1)^{-1}R(j_1)S(i_1))S(i) =$$
$$= (S(i)^{-1}S(i_1)^{-1})R(j_1)(S(i_1)S_i)\ .$$

Zato što važi $(S(i)^{-1}S(i_1)^{-1}) = (S(i_1)S(i))^{-1}$ sledi:

$$(S(i_1)S(i))^{-1}R(j_1)(S(i_1)S(i))\ .$$

Neka je simetrija $S(i_2) = S(i_1)S(i)$ jer je proizvod dve simetrije simetrija i ona se zapisuje u tabelu sim množenja. Kao ključ se unosi par (i_1, i), dok se kao vrednost unosi permutacija $S(i_2)$.

Rezultat primene poteza na osnovu sim koordinata je $S(i_2)^{-1}R(j_1)S(i_2)$ i odgovarajuća sim koordinata je $16 \cdot j_1 + i_2$.

Tabele odsecanja

Brzina „algoritma dve faze" zavisi od procene donje granice broja potrebnih poteza za dovođenje kocke u početno stanje iz zadatog stanja. Neka je trojka (x_1, x_2, x_3) koordinata koja je predstavnik koseta odgovarajuće grupe $H \leq G$. Grupa H je zapravo presek podgrupa grupe G definisanih pojedinačnim koordinatama trojke. Tabele odsecanja se baziraju na koordinatama. Sama koordinata definiše poziciju unosa u tebeli odsecanja, dok je vrednost broj poteza potrebnih da dovedemo kocku u stanje koje odgovara permutaciji iz podgrupe H.

S obzirom da je rešeno stanje kocke permutacija iz H, brojevi zapisani u tabeli odsecanja su uvek donja granica za broj poteza koji dovodi kocku u rešeno stanje. Potrebna je po jedna tabela odescanja za svaku fazu algoritma.

Princip računanja indeksa u tabelama odsecanja je sledeći:

- Izvući indeks simetrije i iz sim koordinata.
- Transformisati kocku konjugacijom simetrije $S(i)$ u kocku iz iste klase ekvivalencije indeksa y, čiji je indeks simetrije 0.
- Transformisati grube koordinate x u $x' = ({x_1}', {x_2}')$.
- Indeks u tabeli odsecanja u prvoj fazi je $2187 \cdot y + x'$, jer je 2187 broj različitih stanja u prvoj fazi.
- Indeks u tabeli odsecanja u drugoj fazi je $40.320 \cdot x' + y$, jer je 40.320 broj različitih stanja u drugoj fazi.

Tabele odsecanja se generišu na sledeći način:

- Na poziciji indeksa rešenog stanja kocke upisuje se dubina 0. Primenjuju se svi mogući potezi na rešeno stanje kocke. Za svako od dobijenih stanja na poziciji indeksa stanja u tabelu odsecanja se upisuje vrednost dubine 1.
- Primenjuju se svi mogući potezi na svako od prethodno dobijenih stanja. Za svako od dobijenih stanja na poziciji indeksa stanja u tabelu odsecanja se upisuje vrednost dubine 2.
- U prvoj fazi algoritma postupak se ponavlja do dubine 12, dok se u drugoj fazi algoritma postupak ponavlja do dubine 18.

Detalji „algoritma dve faze”

Da bi prva faza bila gotova potrebno je najviše 12 poteza, dok je za drugu fazu potrebno najviše 18. Uvek će prvo dobijeno rešenje ovog algoritma imati najviše 30 poteza. Na početku svake faze generišu se odgovarajuće tabele poteza i tabele odsecanja. Algoritam neće stati kada naiđe na prvo rešenje, već će nastaviti da traži optimalnije rešenje.

U prvoj fazi algoritam traži permutacije koje će dovesti kocku u stanje koje odgovara permutaciji iz grupe G_1. Za pretragu u tabelama se koristi algoritam *Iterative deepening A* with a lowerbound heuristic function(IDA*)*[6]. Algoritam iterira kroz

sve moguće permutacije povećavajući im dužinu. Heuristična funkcija za svako stanje kocke procenjuje potreban broj poteza do stanja koje odgovara permutaciji iz grupe G_1. Ona omogućava odsecanje delova tabele prilikom generisanja permutacija, što značajno ubrzava rad algoritma. Bazirana je na tabelama u kojima se pamte obrađena stanja i dozvoljava procenu do 12 poteza unapred.

U drugoj fazi algoritam rešava kocku koristeći samo poteze iz grupe G_1. Heuristična funkcija za svako stanje kocke procenjuje potreban broj poteza do rešenog stanja.

Glava 4

Programska realizacija dva algoritma

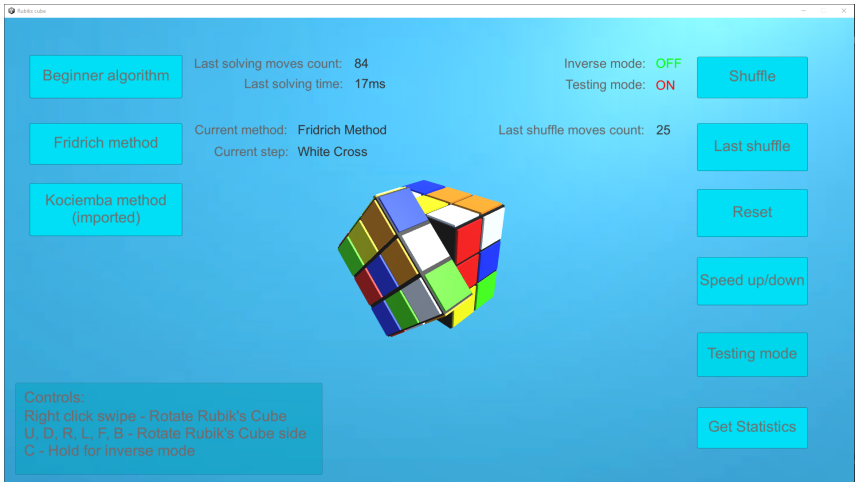
4.1 Aplikacija

Aplikacije ja realizovana pomoću platforme *Unity Game Engine 2022.3.8f1* i pisana je u jeziku *C#*. Celokupan projekat aplikacije se nalazi na adresi:
`https://github.com/mr16146/Rubikova-kocka-master-rad`.
Glavni objekat u prikazu aplikacije je 3D model Rubikove kocke, koja je u početnom stanju rešena. Aplikacija omogućava sledeće:

- Rotiranje kocke,
- Rotiranje strana kocke,
- Generisanje nasumičnog niza poteza, sa ravnomernom raspodelom verovatnoće poteza, i primenu poteza iz tog niza na kocku, tj. mešanje kocke,
- Ponovnu primenu poslednjeg generisanog niza za mešanje kocke,
- Rešavanje kocke pomoću početničke metode, koja je implementirana u aplikaciji,
- Rešavanje kocke pomoću Fridrihine metode, koja je implementirana u aplikaciji,
- Rešavanje kocke pomoću Kociembina metode, za čiju implementaciju je preuzeta biblioteka,
- Pokretanje režima za testiranje koji služi za generisanje podataka pogodnih za eksperimente i čuvanje rezultata testiranja.

- Statistička obrada rezultata testiranja i njeno čuvanje.

3D model kocke sa slike 4.1 je sastavljen od 26 manjih kocki grupisanih u *GameObject* sa imenom *Cube*. Manje kockice imaju jednu, dve ili tri obojene strane, u odnosu na to da li predstavljaju centralnu, ivičnu ili ugaonu kockicu Rubikove kocke. Geometrijski centar Rubikove kocke je na sredini ekrana.



Slika 4.1: Intefrejs aplikacije

Podaci koji se unose u aplikaciju registruju se na jedan od sledećih načina:

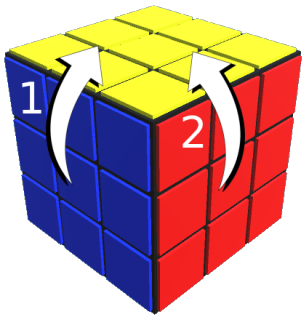
- Unutar *Update()* metode koja se nalazi u svakoj klasi i koja se pokreće prilikom svakog osvežavanja slike,
- Pomoću tastera ili dugmića u interfejsu aplikacije, na čiji klik se okidaju odgovarajuće metode.

Implementacija stanja kocke

Podaci koji određuju stanje kocke implementirani su u klasi *CubeStateData.cs*, i to pomoću narednih atributa:

- *cubeState* – predstavlja stanje kocke u kom se pamti slikanje svake strane kocke u niz od 9 boja kocke, koje predstavlja boje polja date strane,
- *sideToRotationMapping* – čuva slikanje svake strane kocke u stranu koja se trenutno nalazi na njenom mestu u modelu kocke,

- *rotationToSideMapping* – čuva slikanje svake rotacije kocke u rotaciju trenutnog modela kocke.



Slika 4.2: Promena atributa stanja kocke nakon rotacije kocke

Primer 4.1.1. Neka su strane kocke u početnom stanju, odnosno položaju, obojene tako da odgovaraju kocki sa slike 4.2, tj. na sledeći način:

- gornja strana – žutom bojom,
- donja strana – belom bojom,
- prednja strana – plavom bojom,
- zadnja strana – zelenom bojom,
- desna strana – crvenom bojom,
- leva strana – narandžastom bojom.

U slučaju rotacije 3D modela kocke tako da trenutna prednja strana postane gornja strana kocke, vrednosti atributa *sideToRotationMapping* se menjaju na sledeći način:

- gornja strana se slika u prednju,
- donja strana se slika u zadnju,
- prednja strana se slika u donju,

- zadnja strana se slika u gornju,
- desna strana se slika u desnu,
- leva strana se slika u levu.

Ukoliko bi se model kocke nakon toga zarotiro tako da desna strana dođe na mesto gornje strane, vrednosti atributa *sideToRotationMapping* bi bile sledeće:

- gornja strana se slika u desnu,
- donja strana se slika u levu,
- prednja strana se slika u donju,
- zadnja strana se slika u gornju,
- desna strana se slika u zadnju,
- leva strana se slika u prednju.

Tokom izvršavanja algoritama kocku posmatramo u prethodno definisanom početnom položaju. Atribut *rotationToSideMapping* preslikava rotaciju kocke u početnom položaju u rotaciju kocke u trenutnom položaju. Nakon prethodnih rotacija, vrednost ovog atributa je sledeća:

- gornja strana se slika u zadnju,
- donja strana se slika u prednju,
- prednja strana se slika u levu,
- zadnja strana se slika u desnu,
- desna strana se slika u gornju,
- leva strana se slika u donju.

Atributi *sideToRotationMapping* i *rotationToSideMapping* služe kako bi se rotacije kocke i njenih strana mogle preneti na model kocke koji je rotiran na bilo koji način.
Kako bi se svaka rotacija modela kocke ili neke njegove strane preslikala u stanje kocke, postoji klasa *CubeStateWrapper.cs*, koja ima atribut *cubeStateData*. Ovaj atribut

je tipa *CubeStateData* i u njemu se čuva stanje kocke. U klasi *CubeStateWrapper.cs* se nalaze metode koje primenjuju rotacije kocke i njenih strana na stanje kocke iz atributa *cubeStateData*.

U klasi *StateReader.cs* se nalaze sledeći atributi:

- *cubeStateWrapper* – predstavlja trenutno stanje modela kocke i njegov tip je *CubeStateWrapper*. Sve promene nad njim se dešavaju isključivo kao rešenje nekog od algoritama ili kao posledica rotacije modela kocke ili neke njegove strane. Bilo koja promena nad modelom kocke mora biti primenjena i nad stanjem ovog atributa, kako bi on uvek oslikavao trenutno stanje kocke,

- *solvingCubeStateWrapper* – predstavlja stanje kocke koje koristi neki od algoritama tokom svog izvršavanja i njegov tip je takođe *CubeStateWrapper*. Na samom početku nekog od agloritama se napravi duboka kopija atributa *cubeStateWrapper* u atribut *solvingCubeStateWrapper*. Algoritam se preko metoda klase *CubeStateWrapper.cs* primenjuje na stanje atributa *solvingCubeStateWrapper* sve dok ne pronađe niz poteza koji će ga dovesti do rešenog stanja kocke. Nakon toga se dobijeni niz poteza primenjuje na stanje atributa *cubeStateWrapper*, kao i na sam model kocke.

Implementacija rotiranja kocke

Rotiranje kocke se vrši prevlačenjem kocke pomoću desnog klika miša u metodi *RotateCube()* klase *CubeRotation.cs*. Pritiskom desnog klika miša ispaljuje se zrak (eng. *Ray*) iz pozicije kamere, tj. iz trenutne perspektive, i proverava se da li zrak udara u bilo koju prepreku (eng. *collider*). Prepreke su definisane samo na kockicama koje čine 3D model Rubikove kocke, odakle sledi da je pozicija klika na kocki ukoliko zrak udari u bilo koju prepreku.

Ukoliko je zrak udario u bilo koju prepreku, tada se pamte trenutne koordinate pozicije miša u atributu klase *this.startPosition*. Prilikom puštanja desnog klika miša proverava se da li je razlika trenutnih koordinata pozicije miša i atributa klase *this.startPosition* dovoljno velika da bi se 3D model kocke rotirarao. Ova provera služi da se spreči sutacija u kojoj svaki desni klik na model kocke pokreće rotaciju kocke. U odnosu na pomenutu razliku kocka se rotira u pravcima definisanim sa 3 ose. Smer rotacije se određuje na osnovu rezultata pomenute razlike.

Animacija rotiranja kocke implementirana je tako što su model kocke i objekat pod nazivom *RotationObject* grupisani u objektu *MainElement*. Nakon što se

odrede pravac i smer rotacije kocke, prvo se rotira objekat *RotationObject*, a zatim se model kocke animacijski rotira ka njemu, tj. rotira se dok se vrednosti rotacija ova dva objekta ne poklope. Brzina animacije se može promeniti klikom na odgovarajuće dugme interfejsa, čime se menja vrednost atributa *rotationSpeed* klase *CubeRotation.cs*, koji direktno utiče na brzinu animacije.

Implementacija rotiranja strana kocke

Pomoću tastera U, D, R, L, F, B, koji predstavljaju strane kocke, pokreću se rotacije odgovarajućih strana kocke. U odnosu na to koji je od pomenutih tastera pritisnut, u metodi *RotateFace* klase *CubeRotation.cs* proverava se koje sve kockice pripadaju odgovarajućoj strani, i koja je centralna kockica te strane. Nakon toga se kockice koje pripadaju datoj strani grupišu pod centralnom kockicom iste strane i pokreće se njeno animacijsko rotiranje. Animacijsko rotiranje je realizovano po istom principu kao i rotiranje kocke, tj. pravi pravi se kopija trenutne centralne kockice, pod kojom su grupisane ostale kockice odgovarajuće strane, u vidu privremenog objekta. Nakon toga se privremeni objekat rotira u željenom pravcu i smeru, posle čega počinje animacijsko rotiranje centralne kockice strane, odnosno rotiranje cele strane. Rotiranje traje sve dok se rotacije centralne kockice i privremenog objekta ne poklope.
Rotiranje strana u suprotnom smeru omogućeno je pomoću atributa *clockwiseMoves* koji je tipa *bool*. Njegova početna vrednost je *true* i tada se odgovarajući uglovi rotacija strana množe sa 1, pa se strane rotiraju u smeru kazaljke na satu. Njegova vrednost je *false* samo ako je dugme C na tastaturi pritisnuto i odgovarajući uglovi rotacija strana množe sa -1, pa se strane rotiraju u obrnutom smeru.

Implementacija primene algoritama

Bitniji atributi klase *CubeSolver.cs* su:

- *beginnerAlgorithm* i *fridrichMethod* – reference na klase *FridrichMethod.cs* i *BeginnerAlgorithm.cs* u kojima su implementirani algoritmi,
- *isSolving* – promenljiva u kojoj se pamti da li se neki algoritam trenutno izvršava ili se vrše promene nad modelom kocke,
- *testingMode* – promenljiva na osnovu koje se pokreće ili zaustavlja režim testiranja,

- *minimumShuffleMoves* – minimalni broj poteza koje će generisati metoda za mešanje kocke,
- *maximumShuffleMoves* – maksimalni broj poteza koje će generisati metoda za mešanje kocke uvećan za 1,
- *shuffleHistory* – lista u kojoj se pamte potezi svakog mešanja kocke,
- *testingResultsFilePath* – putanja do tekstualnog dokumenta u kome će se čuvati rezultati režima testiranja.
- *statisticsFilePath* – putanja do tekstualnog dokumenta u kome će se čuvati statistika rezultata.

U ovoj klasi se pozivaju metode za rešavanje kocke *SolveBeginnerAlgorithm()*, *SolveFridrichMethod()* i *SolveKociembaMethod()* pomoću klika na odgovarajuće dugme interfejsa, koje rešenje kocke, tj. niz poteza koji rešava kocku, primenjuju na model kocke. Ukoliko je kocka već rešena nije moguće pokrenuti nijedan algoritam za rešavanje kocke.
Mešanje kocke pokreće se klikom na odgovarajućee dugme intefejsa, što okida metodu *ShuffleCubeEvenly()* koja generiše niz poteza sa ravnomernom raspodelom verovatnoće svakog poteza, i meša kocku primenjujući svaki od poteza iz niza na model kocke. Klikom na odgovarajuće dugme interfejsa okida se metoda *LastShuffle()* koja iz atributa *shuffleHistory* uzima poslednji niz poteza i primenjuje ga na model kocke.
Vraćanje kocke u početno stanje omogućeno je klikom na odgovarajuće dugme interfejsa, čime se brzina animacije privremeno povećava na maksimalnu definisanu, a zatim se kocka rešava Kociembinom metodom.
Prilikom klika na odgovarajuće dugme interfejsa atribut *testingMode* se menja i pokreće se režim za testiranje. U svakoj iteraciji metode *Update()* poziva se metoda *DoTesting()* koja se sastoji od narednih koraka:

1. Primenjuje se metoda *DoShuffleEvenly()* koja meša kocku,
2. Primenjuje se Fridrihina metoda za rešavanje kocke i dopisuju se njeni rezultati izvršavanja u tekstualni dokument na lokaciji *testingResultsFilePath*,
3. Primenjuje se metoda *DoLastShuffle()* koja primenjuje poslednje zapamćeno mešanje kocke iz atributa *shuffleHistory*,

4. Primenjuje se početna metoda za rešavanje kockei dopisuju se njeni rezultati izvršavanja u tekstualni dokument na lokaciji *testingResultsFilePath*,

5. Primenjuje se metoda *DoLastShuffle()* koja primenjuje poslednje zapamćeno mešanje kocke iz atributa *shuffleHistory*,

6. Primenjuje se Kociembina metoda za rešavanje kocke i dopisuju se njeni rezultati izvršavanja u tekstualni dokument na lokaciji *testingResultsFilePath*.

Rezultati izvršavanja metoda za rešavanje kocke su sledeći parametri:

- Broj poteza koji je potreban da bi se kocka rešila,
- Vreme izvršavanja metode.

Rezultati se mogu statistički obraditi klikom na odgovarajući dugmić interfejsa. Na osnovu podataka iz fajla sačuvanog u režimu testiranja računaju za svaku metodu prosek, minimum, maksimum i standardna devijacija. Ti podaci se upisuju u fajl na putanji *testingResultsFilePath*.

Biblioteka za Kociembinu metodu

Kociembina metoda je realizovana pomoću biblioteke Meta Kolborna (eng. *Matt Colbourne*), preuzete sa naredne adrese:
`https://github.com/Megalomatt/Kociemba/tree/Unity`.
Metoda se izvršava u metodi *SolveKociemba()* klase *CubeSolver.cs* sledećim koracima:

1. Pomoću metode *GetKociembaInputState()* od trenutnog stanje kocke dobija se ulaz pogodan za poziv metode preuzete biblioteke koja rešava kocku Kociembinom metodom,

2. Poziva se metoda *Search.solution()* preuzete biblioteke koja kao povratnu vrednost ima niz poteza koji rešavaju trenutno stanje kocke,

3. Transformiše se rešenje dobijeno metodom *Search.solution()* iz preuzete biblioteke u niz poteza tipa *CubeMove* koji se primenjuju na model kocke, gde je *CubeMove.cs* klasa koja sadrži sve neophodne atribute da bi se određeni potez primenio na model kocke.

4.2 Početnička metoda

Pod uslovima da se kocka posmatra tako da je strana sa centralnim belim pooljem na donjoj, a strana sa centralnim žutim poljem na gornjoj strani kocke, početnička metoda za rešavanje Rubikove kocke sastoji se iz narednih koraka [2]:

1. Pravljenje belog krsta na žutoj strani kocke,
2. Spuštanje belog krsta na belu stranu kocke,
3. Istovremeno pozicioniranje i orijentisanje kockica prvog sloja kocke,
4. Istovremeno pozicioniranje i orijentisanje kockica drugog sloja kocke,
5. Pravljenje krsta na gornjoj strani kocke,
6. Pravljenje krsta na svim preostalim stranama kocke,
7. Pozicioniranje ugaonih kockica gornjeg sloja,
8. Orijentisanje ugaonih kockica gornjeg sloja.

Slojevi kocke se broje odozdo na gore u odnosu na prethodno definisani položaj posmatranja kocke. Pod bočnim stranama kocke se ubuduće misli na sve strane kocke sem gornje i donje.

Implementacija ovog algoritma se nalazi u klasi *BeginnerAlgorithm.cs*, a ceo algoritam se izvršava pozivom metode *SolveBeginner()*.

Prvi korak je implementiran u metodi *WhiteCrossOnYellowSide()*. Prolazi se kroz sve strane kocke dok se ne napravi beli krst na žutoj strani, tj. dok sva ivična polja gornje strane ne budu bele boje. Prilikom prolaska se proverava da li je na trenutnoj strani neko ivično polje bele boje. Ukoliko postoji ivično polje bele boje, proverava se da li je odgovarajuće ciljano polje na gornjoj strani bele boje. Ako nije, onda se premešta ugaona kockica sa belim poljem na gornju stranu odgovarajućim nizom poteza. Ako jeste, gornja strana se rotira dok se ovaj uslov ne promeni. Zatim se prelazi na sledeću stranu kocke dok se ne prođu sve bočne strane kocke. Gornja strana se preskače, a ukoliko je trenutna strana donja strana proverava se da li postoji belo polje na donjoj strani kocke. Ukoliko postoji, rotira se gornji sloj kocke dok se na naspramnom polju na žutoj strani ne pojavi belo polje. Premešta se belo polje

sa donje strane na gornju odgovarajućim nizom poteza.

Drugi korak je implementiran u metodi *WhiteCross()*. Prolazi se kroz sve bočne strane kocke i proverava se da li je prednja strana gornje ivične kockice iste boje kao boja te strane, odnosno kao boja centralnog polja trenutne strane. Rotira se gornja strana dok se ovaj uslov ne ispuni, a zatim se odgovarajućim nizom poteza zamenjuju polja gornje i donje strane kocke date strane, tj. prebacuje se bela ivična kockica na donju stranu.

Treći korak je implementiran u metodi *FirstLayer()*. Proverava se da li je prvi sloj već pozicioniran i orijentisan; ukoliko jeste ovaj korak se prekida. Prolazi se kroz ugaone kockice gornjeg sloja i proverava se da li postoji ugaona kockica sa belim poljem. Ukoliko ne postoji, prolazi se kroz strane i traži se prva ugaona kockica donjeg sloja koja nije pozicionirana i orijentisana; ona se prebacuje u gornji sloj određenim nizom poteza. Rotira se gornji sloj kocke dok ugaona kockica sa belim poljem ne bude iznad svoje odgovarajuće pozicije u donjem sloju, zatim se ona odgovarajućim nizom poteza prebacuje u donji sloj, orijentisana i pozicionirana. Postupak se ponavlja dok se ne orijentišu i ne pozicioniraju sve ugaone kockice donjeg sloja.

Četvrti korak je implementiran u metodi *SecondLayer()*. Proverava se da li je drugi sloj već pozicioniran i orijentisan; ukoliko jeste ovaj korak se prekida. Prolazi se kroz ivične kockice gornjeg sloja i proverava se da li postoji ivična kockica bez žutog polja. Ukoliko ne postoji, prolazi se kroz strane i traži se prva ivična kockica srednjeg sloja koja nije pozicionirana i orijentisana; ona se prebacuje u gornji sloj određenim nizom poteza. Rotira se gornji sloj kocke dok prednja strana ivična kockice bez žutog polja ne bude iste boje kao centralno polje trenutne strane, zatim se ona odgovarajućim nizom poteza prebacuje u srednji sloj, orijentisana i pozicionirana. Postupak se ponavlja dok se ne orijentisu i ne pozicioniraju sve ivične kockice srednjeg sloja.

Peti korak je implementiran u metodi *SideCrosses()*. Prolazi se kroz bočne strane kocke i traže se dve strane koje imaju već podešen krst na svojoj strani, tj. da su sva ivična polja jedne strane iste boje. Ukoliko ne postoje takve dve strane za redom, primenjuje se odgovarajući niz poteza koji rotira ivične kockice gornjeg sloja, nakon čega će sigurno postojati takve dve strane. Primenjuje se odgovarajući niz poteza koji rotira ivične kockice gornjeg sloja, nakon čega su sve ivične kockice

gornjeg sloja pozicionirane i orijentisane.

Šesti korak je implementiran u metodi *RepositionTopCorners()*. Proverava se da li su sve ugaone kockice gornjeg sloja pozicionirane; ukoliko jesu ovaj korak se prekida. Prolazi se kroz bočne strane kocke i traži se strana koja ima jednu pozicioniranu gornju ugaonu kockicu. Ukoliko takva strana ne postoji, primenjuje se odgovarajući niz poteza koji rotira ugaone kockice gornjeg sloja, nakon čega će sigurno postojati takva strana. Dok je ta strana trenutna prednja strana, primenjuje se odgovarajući niz poteza koji rotira ugaone kockice gornjeg sloja dok sve ugaone kockice gornjeg sloja ne budu pozicionirane.

Sedmi korak je implementiran u metodi *ReorientTopCorners()*. Proverava se da li su sve ugaone kockice gornjeg sloja orijentisane; ukoliko jesu ovaj korak se prekida. Prolazi se kroz sve bočne strane i traži se prva pogrešno orijentisana gornja ugaona kockica. Zatim se primenjuje odgovarajući niz poteza kako bi se ona pravilno orijentisala. Rotira se gornji sloj kocke dok na njeno mesto ne dođe druga pogrešno orijentisana ugaona kockica, ponavlja se isti niz poteza dok se ona ne orijentiše pravilno. Postupak se ponavlja dok sve ugaone kockice gornjeg sloja ne budu orijentisane, nakon čega je potrebno rotirati gornji sloj kocke dok kocka ne bude rešena.

4.3 Fridrihina metoda

Ovu metodu je popularizovala i dokumentovala *Džesika Fridrih*[1], a poznata je i pod nazivom metoda *CFOP*, po njenim koracima [4]. Pod uslovima da se kocka posmatra tako da je strana sa centralnim belim pooljem na donjoj, a strana sa centralnim žutim poljem na gornjoj strani kocke, koraci ove metode su sledeći:

1. Pravljenje belog krsta na beloj strani kocke (eng. *Cross*) [3],
2. Istovremeno pozicioniranje i orijentisanje kockica prvog i drugog sloja kocke (eng. *First 2 layers*) [8],

[1]Džesika Fridrih je češki profesor matematike. Smatra se da je među prvima počela da se bavi brzim rešavanjem Rubikove kocke (eng. *speedcubing*). Najpoznatija je po popularizaciji i dokumentovanju *CFOP* metode, koja je kasnije po njoj dobila ime Fridrihina metoda.

3. Orijentacija gornjeg sloja, tj. prebacivanje svih žutih polja na gornji sloj (eng. *Orientation of the last layer*) [9],

4. Permutacija gornjeg sloja, tj. orijentacija kockica gornjeg sloja (eng. *Permutation of the last layer*).

Implementacija ovog algoritma se nalazi u klasi *FridrichMethod.cs*, a ceo algoritam se izvršava pozivom metode *SolveFridrich()*.

Prvi korak je implementiran koristeći metode *WhiteCrossOnYellowSide()* i *WhiteCross()* iz početničke metode. Efikasnije rešenje ovog koraka bi podrazumevalo implementaciju mnogo različitih slučaja u odnosu na poziciju belih polja na kocki, kao i tabele poteza koje bi zamenile intuiciju prilikom samog odabira redosleda poteza.

Drugi korak je implementiran u metodi *FirstTwoLayers()*. Postoji 41 slučaj za rešavanje ovog koraka u odnosu na raspored ugaonih kockica donjeg sloja i ivičnih kockica srednjeg sloja kocke. Oni se po sličnosti rasporeda kockica mogu podeliti na 6 grupa. U prvoj grupi su slučajevi sa najjednostavnijim rešenjima, dok su u šestoj slučajevi sa najkomplikovanijim rešenjima, tj. kompleksnost rešenja raste sa rednim brojem grupe. Ukoliko su kockice prva dva sloja već orijentisane i pozicionirane, ovaj korak se prekida. Korak je implementiran tako što se prolazi kroz bočne strane kocke i proverava da li se kocka rotacijom gornjeg sloja može dovesti u neki od poznatih slučajeva. Provera da li stanje kocke odgovara nekom od slučajeva ide od jednostavnijih grupa slučajeva, do komplikovanijih. Nakon prolaska kroz svih 6 grupa slučajeva i primene rešenja ukoliko stanje kocke odgovara nekom od slučajeva, proverava se da li su kockice prva dva sloja pozicionirane i orijentisane. Ako jesu korak je završen, a u suprotnom se poziva metoda *UnstuckFirstTwoLayers()*. Ova metoda prolazi kroz bočne strane kocke i traži prvi par ugaone kockice prvog sloja i ivične kockice srednjeg sloja koji nije orijentisan i pozicioniran. Taj par se zamenjuje sa trenutnim parom iz gornjeg sloja i ponovo poziva metodu *FirstTwoLayers()*. Zna se da se ovakva situacija može desiti najviše 4 puta, u slučaju da se desi za sva 4 para ugaonih kockica donjeg sloja i ivičnih kockica srednjeg sloja.

Treći korak je implementiran koristeći metodu *YellowCross()* iz početničkog algo-

ritma i metodu *SecondLookOll()*. Najefikasnija verzija ovog koraka ima ukupno 57 slučaja, dok je u aplikaciji implementirana *Second Look OLL* verzija koja ima 10 slučajeva. Ona se sastoji iz dva dela:

1. Pravljenje žutog krsta na gornjoj strani kocke (eng. *first look*),
2. Prebacivanje svih žutih polja na gornju stranu kocke (eng. *second look*), na osnovu kog je ovaj korak i dobio ime.

Prvi deo je isti kao i istoimeni korak u početničkoj metodi i ima tri različita slučaja, u odnosu na raspored žutih polja na gornjoj strani kocke, koji se rešavaju odgovarajućim nizom poteza. Drugi deo ima ukupno 7 slučaja, u odnosu na orijentaciju žutih polja na ugaonim kockicama gornjeg sloja, koji se rešavaju odgovarajućim nizom poteza. Oni se mogu podeliti u odnosu na broj žutih ugaonih polja na gornjem sloju radi lakše implementacije. Primenom rešenja prvog, a zatim i drugog dela ovog koraka na gornjem sloju se nalaze samo žuta polja.

Četvrti korak je implementiran u metodama *FirstLookPll()* i *SecondLookPll()*. Najefikasnija verzija ovog koraka ima ukupno 21 slučaj, dok je u aplikaciji implementirana *Second Look PLL* verzija koja ima 6 slučajeva. Ona se sastoji iz dva dela:

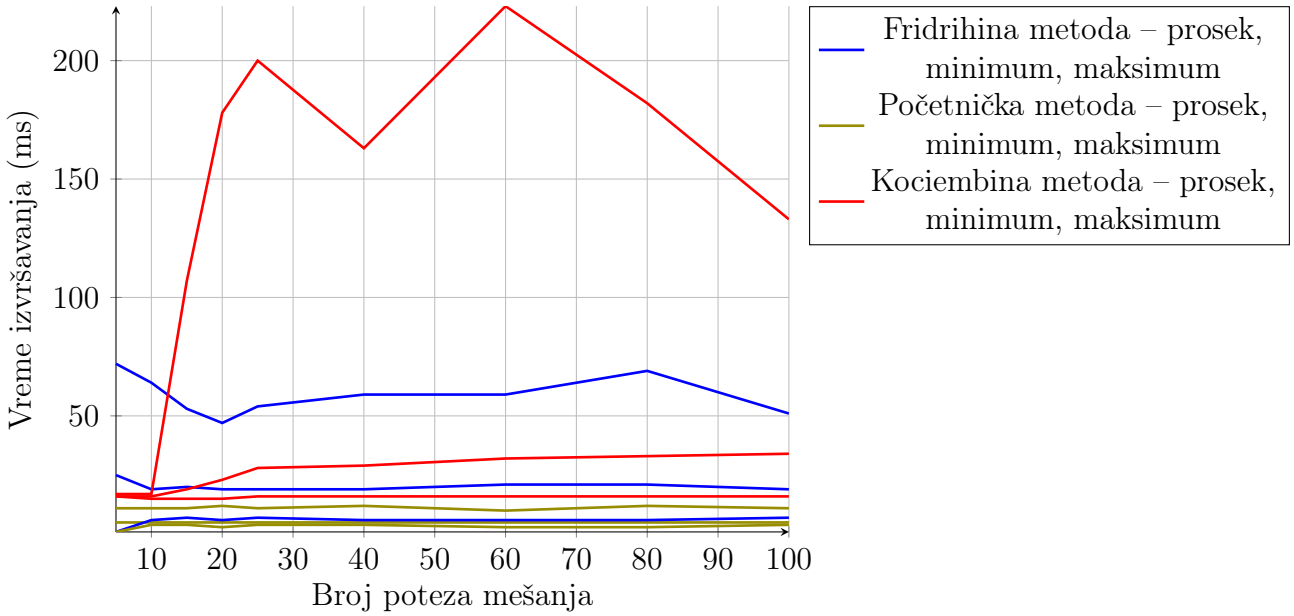
1. Pozicioniranje i orijentisanje ugaonih kockica gornjeg sloja (eng. *first look*),
2. Pozicioniranje i orijentisanje ivičnih kockica gornjeg sloja (eng. *second look*), na osnovu kog je ovaj korak i dobio ime.

Prvi deo ima samo dva slučaja u odnosu na raspored polja ugaonih kockica gornjeg sloja i rešava se primenom odgovarajućeg niza poteza sa svaki od slučaja. Drugi deo ima četiri slučaja u odnosu na raspored polja ivičnih kockica gornjeg sloja. Gornji sloj kocke se rotira dok ugaone kockice gornjeg sloja ne budu pozicionirane i orijentisane, zatim se primenjuje niz poteza rešenja za odgovarajući slučaj. Nakon toga je Rubikova kocka rešena.

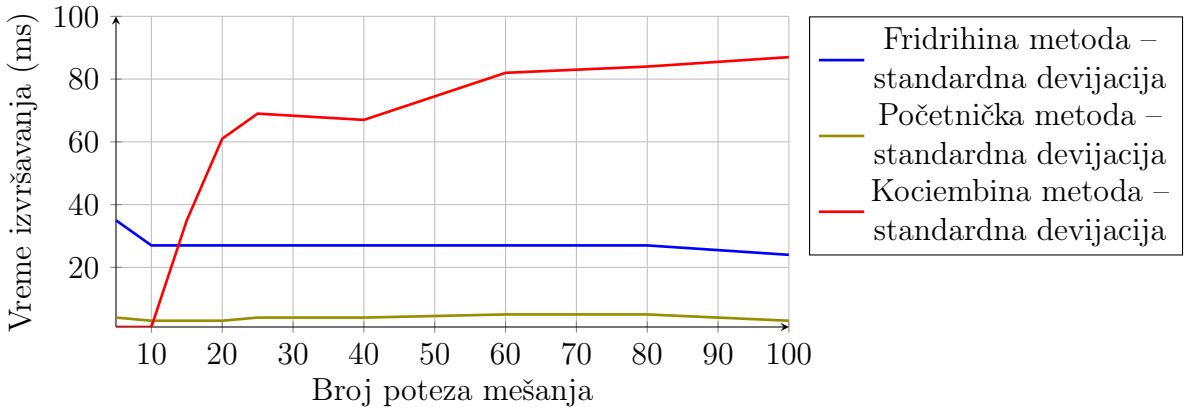
4.4 Eksperimentalno poređenje tri algoritma

Testiranje programa realizovano je pomoću režima za testiranje. Izvršeno je 9 iteracija testiranja u trajanju od po 1h. Korišćen je računar sa procesorom koji ima 6

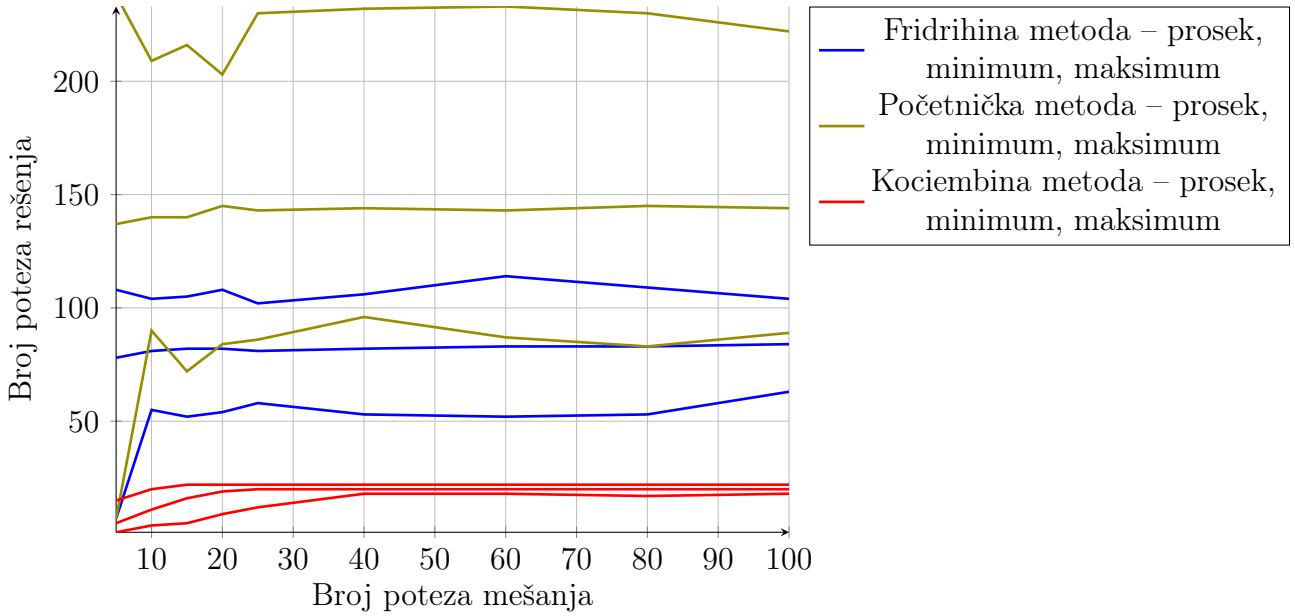
jezgara na frekvenciji 4.7GHz i 32GB radne memorije. U svakoj narednoj iteraciji povećava se broj poteza mešanja kocke, koji pripada skupu $\{5, 10, 15, 20, 25, 40, 60, 80, 100\}$. Broj instanci u svakoj grupi redom pripada skupu $\{788, 653, 583, 470, 447, 436, 419, 356, 212\}$. Cilj eksperimenta je ispitivanje zavisnosti broja poteza rešenja i vremena izvršavanja u odnosu na broj poteza korišćenih za mešanje kocke.



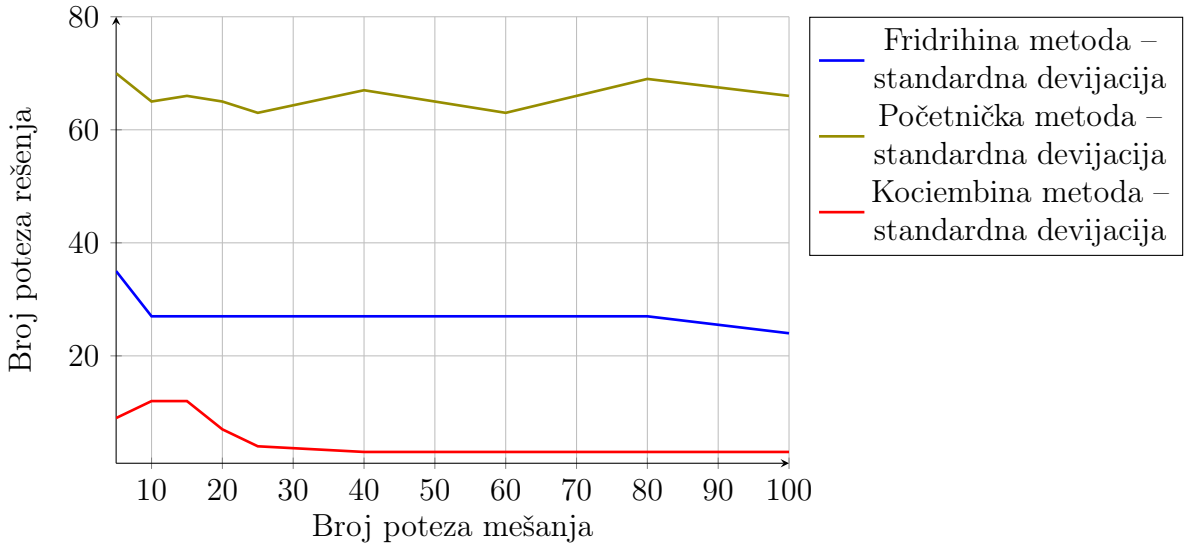
Slika 4.3: Dijagram zavisnosti vremena izvršavanja od broja poteza mešanja



Slika 4.4: Dijagram standardne devijacije vremena izvršavanja u odnosu na broj poteza mešanja



Slika 4.5: Dijagram zavisnosti broja poteza rešenja od broja poteza mešanja



Slika 4.6: Dijagram standardne devijacije broja poteza rešenja u odnosu na broj poteza mešanja

Vreme izvršavanja (ms)		Broj poteza mešanja kocke								
		5	10	15	20	25	40	60	80	100
Fridrihina metoda	prosek	25	19	20	19	19	19	21	21	19
	minimum	1	6	7	6	7	6	6	6	7
	maksimum	72	64	53	47	54	59	59	69	51
	stand. dev.	12.3	9.8	10.1	9.6	10.2	10.7	10.9	11.3	10.6
Početnička metoda	prosek	5	5	5	5	5	5	5	5	5
	minimum	1	4	4	3	4	4	3	3	4
	maksimum	11	11	11	12	11	12	10	12	11
	stand. dev.	1.4	1	1.2	1.1	1.3	1.2	1.6	1.6	1.1
Kociembina metoda	prosek	16	16	19	23	28	29	32	33	34
	minimum	16	15	15	15	16	16	16	16	16
	maksimum	17	17	107	178	207	163	273	182	133
	stand. dev.	0.2	0.2	11.8	20.5	23.1	22.3	27.3	28.3	28.9

Tabela 4.1: Vremena izvršavanja u odnosu broj poteza mešanja

Broj poteza rešenja		Broj poteza mešanja kocke								
		5	10	15	20	25	40	60	80	100
Fridrihina metoda	prosek	78	81	82	82	81	82	83	83	84
	minimum	7	55	52	54	58	53	52	53	63
	maksimum	108	104	105	108	102	106	114	109	104
	stand. dev.	11.6	9	9.1	9.1	9.1	9.1	9.1	8.9	8.2
Početnička metoda	prosek	137	140	140	145	143	144	143	145	144
	minimum	7	90	72	84	86	96	87	83	89
	maksimum	238	209	216	203	230	232	233	230	222
	stand. dev.	23.4	21.8	22	21.8	21.2	22.4	21.6	23	22.6
Kociembina metoda	prosek	5	11	16	19	20	20	20	20	20
	minimum	1	4	5	9	12	18	18	17	18
	maksimum	15	20	22	22	22	22	22	22	22
	stand. dev.	2.8	4.1	4.2	2.4	1.3	1	1.1	1.1	1.2

Tabela 4.2: Broj poteza rešenja u odnosu broj poteza mešanja

Na dijagramima sa slika 4.3, 4.4, 4.5 i 4.6 vrednosti standardnih devijacija pomnožene su sa faktorom 3, radi lakšeg prikaza na grafu. Prikazana je zavisnost vremena izvršavanja, odnosno broja poteza rešenja, u odnosu na broj poteza mešanja, i to pomoću narednih parametara:

- prosek,
- minimum
- maksimum,
- standardna devijacija, koja se računa po formuli $\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})^2}$, gde je x_i element skupa za koji se standardna devijacija računa, $\overline{x}$ je aritmetička sredina skupa, a N je broj elemenata skupa.

Na osnovu dijagrama sa slike 4.5 i tabele 4.2 se vidi da je najefikasnija metoda po broju poteza rešenja Kociembina metoda, zatim Fridrihina metoda, dok je početnička metoda najlošija. Posledica toga je da početnička metoda u svakoj iteraciji ima najveće razlike prosečnih, minimalnih i maksimalnih vrednosti broja poteza rešenja, nasuprot Kociembinoj metodi čije su razlike istih parametara minimalne.

U svakoj metodi broj poteza mešanja kocke utiče na broj poteza rešenja samo u slučaju najmanje vrednosti broja poteza mešanja kocke. To se vidi i na osnovu standardne devijacije koja ima veće oscilacije samo u slučaju najmanje vrednosti broja poteza mešanja.

Sa rastom broja poteza mešanja se može primetiti da razlike između proseka, minimuma i maksimuma broja poteza rešenja, u odnosu na rezultate iz prethodne iteracije testiranja, postaju sve manje, a vrednost standardne devijacije teži konstanti. Iz toga sledi da svaka od metoda podjednako efikasno pronalazi rešenje bez obzira na broj poteza mešanja, sem u slučaju najmanje vrednosti kada je broj poteza rešenja znatno manji.

Posledica najmanjeg broja poteza rešenja Kociembine metode, u odnosu na druge dve metode, je da Kociembina metoda ima najveće vreme izvršavanja, što se može videti u tabeli 4.1 i na dijagramu sa slike 4.3. U slučaju početničke metode broj poteza mešanja kocke nema skoro nikakav uticaj na vreme izvršavanja algoritma, što je posledica toga da algoritam uvek primenjuje isti niz koraka bez obzira na zadato stanje kocke. Slično važi i za Fridrihinu metodu, koja ima male oscilacije u vrednostima parametara. U njenom slučaju je vreme izvršavanja nešto veće, što je

posledica mnogo većeg broja slučaja koje metoda treba da obradi, ali i dela implementacije koji proverava koji od poteza dovodi do najefikasnijeg rešenja.

U slučaju Kociembine metode vrednosti prosečnog i minimalnog vremena izvršavanja nemaju oscilacije u odnosu na broj poteza mešanja kocke, ali postoje specifični slučajevi za koje su maksimalne vrednosti vremena izvršavanja znatno veće od prosečnih. To se vidi i na osnovu porasta vrednosti standardne devijacije vremena izvršavanja sa porastom broja poteza mešanja kocke. U slučajevima sa manjim brojem poteza mešanja Kociembina metoda pronalazi niz poteza rešenja kocke pomoću tabela odsecanja i poteza, i na taj način u tim slučajevima ima standardu devijaciju približnu nuli. To se vidi i na osnovu minimalnih vrednosti broja poteza rešenja u tim slučajevima, u najboljem slučaju ova metoda je rešila kocku u samo jednom potezu. Kako raste broj poteza mešanja kocke, tako u Kociembinoj metodi raste i broj specifičnih slučajeva koji zahtevaju više vremena izvršavanja.

Krive proseka vremena broja poteza rešenja i Fridrihine metode u svim iteracijama imaju približno iste vrednosti u odnosu na broj poteza mešanja kocke. Za razliku od njih, Kociembina metoda za manji broj poteza mešanja ima znatno manje vrednosti proseka broja poteza rešenja. To govori da Kociembina metoda u jednostavnijim slučajevima koristi prednost tabela poteza i tabela odsecanja kako bi zapravo ponovila invertovan niz poteza mešanja kocke, dok početnička i Fridrihina metoda idu redom definisanim koracima bez obzira na zadato stanje kocke.

Glava 5

Zaključak

Zanimljivo je da se jednoj igrački iz detinjstva može pristupiti na jedan ni malo naivan način. Rubikovu kocku je moguće posmatrati kao izazov dok je rešavamo iz dosade ili na takmičenju, ali i kao ozbiljan projekat pravljenja algoritama koji često uključuju i računar kako bi se došlo do najefikasnijeg rešenja. Postoji mnogo različitih algoritama za rešavanje kocke i samo istraživanje svih tih algoritama nema kraja, što i privlači ljude. Iza svakog od tih algoritama stoji neretko kompleksna matematika koja je većinom i pokretač ideje o algoritmu. Sama činjenica da su bile potrebne godine usavršavanja algoritama za rešavanje kocke da bi se dobilo najefikasnije moguće rešenje govori o njenoj kompleksnosti i zanimljivosti.

U ovoj tezi, nakon definisanja potrebnih pojmova iz teorije grupa, predstavljeni su pojmovi vezani za teoriju kocki. Fundamentalne teoreme teorije kocki definišu uslove koji tokom konstruisanja nekog algoritma moraju biti ispunjeni kako bi kocka uopšte mogla biti rešena. Da bi se odabrani algoritam za rešavanje kocke što jasnije predstavio, potrebno je pre svega definisati najpogodnije obeležavanje kocke kao i način manipulacije njome. Opisano je nekoliko algoritama koji se baziraju na metodi podgrupa, dok je detaljno predstavljena Kociembina metoda.

U okviru aplikacije su implementirane dve metode, dok je za testiranje treće metode iskorišćena spoljna biblioteka. Aplikacija nudi čuvanje rezultata eksperimenata kao i njihovu obradu. Sproveden je eksperiment koje je uporedio rezultate sve tri metode za rešavanje kocke. Pokazano je da Kociembina metoda ima najduže vreme izvršavanja i rešenje sa najmanjim brojem poteza, dok za početničku metodu važi obrnuto. Sama aplikacija može služiti kao alat za poređenje pomenuta tri algoritma, ali s obzirom da sadrži i 3D model kocke kojim je moguće manipulisati, može služiti i kao alat za proizvoljno slaganje Rubikove kocke.

Literatura

[1] L. Daniels, Group Theory and the Rubik's Cube, 2014.

[2] D. Ferenc (2023), How to solve the Rubik's Cube - Beginners Method, Ruwix, Preuzeto sa adrese: https://ruwix.com/the-rubiks-cube/how-to-solve-the-rubiks-cube-beginners-method.

[3] D. Ferenc (2023), First two layers - F2L, Ruwix, Preuzeto sa adrese: https://ruwix.com/the-rubiks-cube/advanced-cfop-fridrich/first-two-layers-f2l.

[4] D. Ferenc (2023), Rubik's Cube solution with advanced Fridrich (CFOP) method, Ruwix, Preuzeto sa adrese: https://ruwix.com/the-rubiks-cube/advanced-cfop-fridrich.

[5] A. Frey, D. Singmaster, Handbook of Cubik Math, 1982.

[6] D. Joyner, Adventures in Group Theory: Rubik's Cube, Merlin's Machine, and Other Mathematical Toys, 2005.

[7] G. Kalajdžić, Algebra, 2011.

[8] D. Racine (2023), 3x3 2LOLL, Dan's Cubing Cheat Sheet App, Preuzeto sa adrese: https://cubingcheatsheet.com/algs3x_2loll.html.

[9] D. Racine (2023), 3x3 2LPLL, Dan's Cubing Cheat Sheet App, Preuzeto sa adrese: https://cubingcheatsheet.com/algs3x_2lpll.html.

[10] D. Singmaster, Notes on Rubik's Magic Cube, 1980.

[11] W. D. Joyner, Mathematics of the Rubik's cube, 1996.