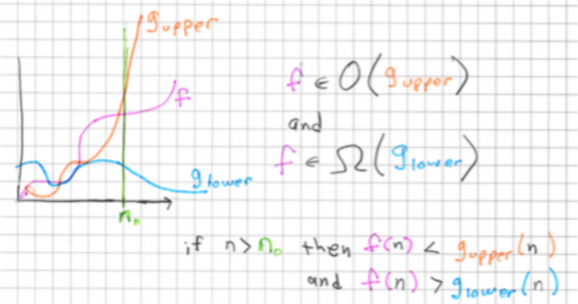
Rubik's Cube 2x2x2

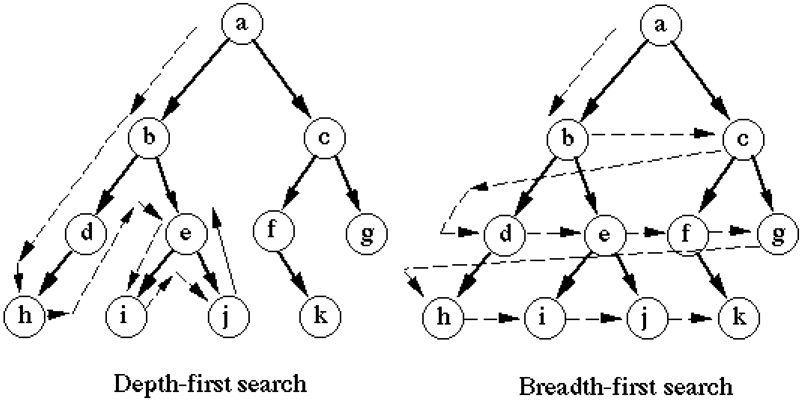
September 22, 2026

Formula to remember

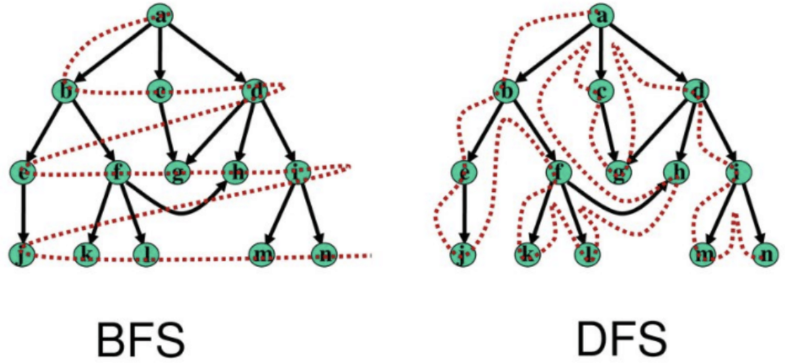
$$\sum_{v \in V} degree(v) = \Theta(|E|)$$



Intuition of DFS vs. BFS



Intuition of DFS vs. BFS



Intuition of DFS vs. BFS

- There may be no distinguished root.
- A vertex may be a neighbor of multiple vertices.
- There may be loops.
- The graph may be disconnected.
- The graph may be generated lazily.

$2 \times 2 \times 2$ Rubik's cube

Let's take a look the $2 \times 2 \times 2$ Rubik's cube.

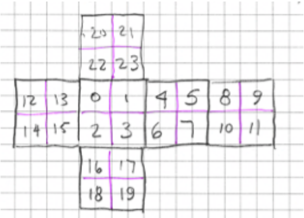
The cube has 6 faces,
each having 4 stickers.
There are 4 stickers of each color.
So $24 = 4 \times 6$ stickers, or positions.

$2 \times 2 \times 2$ Rubik's cube

Let's take a look the $2 \times 2 \times 2$ Rubik's cube.

Abbreviation	**Color**
R	red
W	white
B	blue
G	green
O	orange
Y	yellow

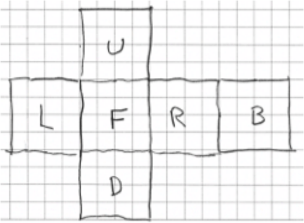
$2 \times 2 \times 2$ Rubik's cube



We can represent a state of the cube as an array of size 24.

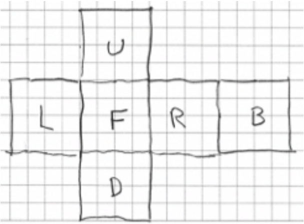
So each of the 24 positions has an index, and the value at that index is the color of the sticker at that position.

$2 \times 2 \times 2$ Rubik's cube



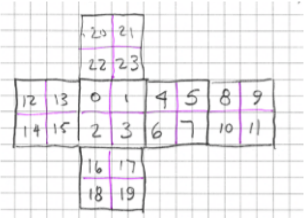
To help with notation we denote each of the 6 faces by a single letter name.

$2 \times 2 \times 2$ Rubik's cube



Abbreviation	Face
F	front
B	back
U	up
D	down
L	left
R	right

$2 \times 2 \times 2$ Rubik's cube

We can represent any state by an appropriate permutation of the string

$$WWWWGGGGYYYYBBBBOOOORRRR$$

Size of state space

How many different states can the cube be in; i.e., what is the size of the state space?

Size of state space

One way to bound the size is by asking what the largest 24 digit base 6 integer is.
I.e., if we start with any string representing a state, for example `GWRGYYRBYWBGOROWORB`, and map it to a number according to the following map,

$$
\begin{aligned}
W &\to 0 \\
G &\to 1 \\
Y &\to 2 \\
B &\to 3 \\
O &\to 4 \\
R &\to 5
\end{aligned}
$$

Size of state space

$$
\begin{aligned}
W &\to 0\\
G &\to 1\\
Y &\to 2\\
B &\to 3\\
O &\to 4\\
R &\to 5
\end{aligned}
$$

then we arrive at a 24 digit base 6 number such as

$$\text{BGWRGYYRBYWBGOROWORB} \;\to\; 31051225320314540453_6\,.$$

Size of state space

The largest 24 digit base six integer is

$$6^{24} - 1 < 4.73838 \times 10^{18}\,.$$

The largest 24 digit base 6 integer we can write with exactly four 5's, four 4's, four 3's, four 2's, four 1's, and four 0's is

$$555544443333222211110000_6\,.$$

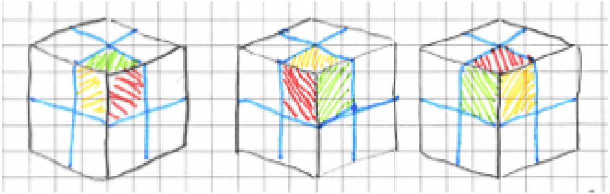
The smallest such number is

$$11112222333344445555_6\,.$$

Thus there are

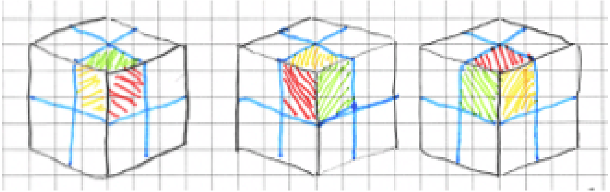
$$\begin{aligned} 555544443333222211110000_6 &- 11112222333344445555_6 \\ &< 4.737 \times 10^{18}\,. \end{aligned}$$

A better bound



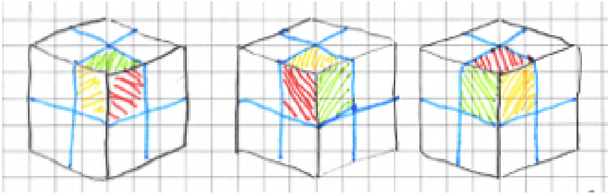
The bound is actually excessively high. How many of these 4.73765×10^{18} potential states are actually reachable by rotations (twists) of the $2 \times 2 \times 2$ Rubik's cube?

A better bound



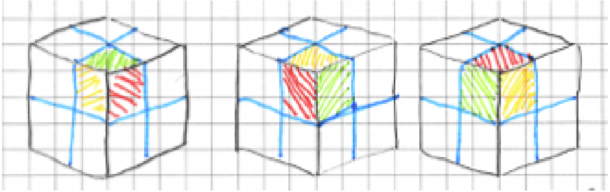
The cube consists of 8 *cubelets*, each with 3 or stickers or *facelets*.

A better bound



One might imagine any of the cubelets in one of 8 positions, and with any of 3 orientations, $|S| < 8! \times 3^8 = 265,539,520$.

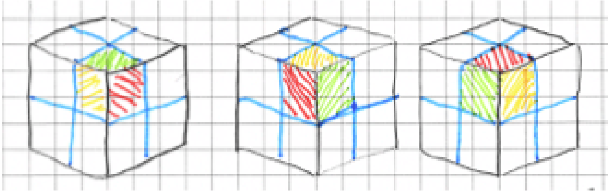
A better bound



One might imagine any of the cubelets in one of 8 positions, and with any of 3 orientations, $|S| < 8! \times 3^8 = 265,539,520$.

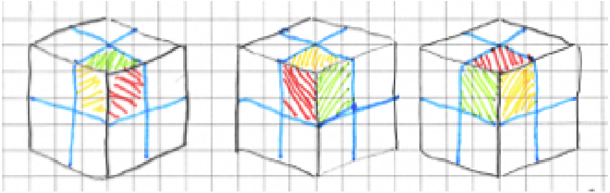
We are careful not to claim that all of these states are reachable by rotating the cube–we are merely trying to bound the size of the state space.

A better bound



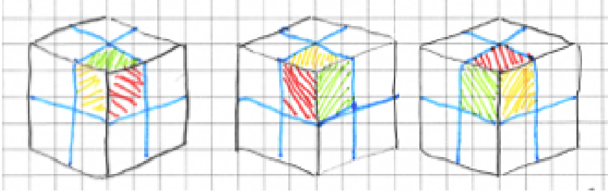
Finally, if we realize that we can rotate the entire cube in our hand to look at any face, without actually twisting the cube, we see that we can look at any of 6 faces,

A better bound



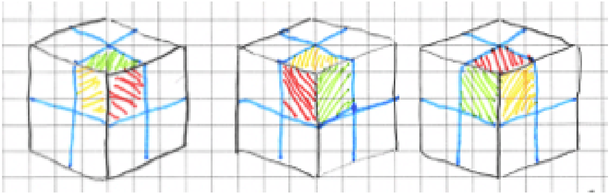
and while looking at the face we can turn the entire cube so that any of the adjacent faces is on top.

A better bound



and while looking at the face we can turn the entire cube so that any of the adjacent faces is on top.
That means there are $6 \times 4 = 24$ orientations of the cube which we can consider isomorphic.

A better bound

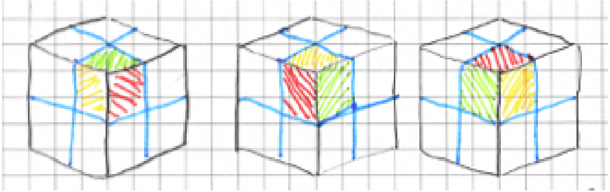


and while looking at the face we can turn the entire cube so that any of the adjacent faces is on top.
That means there are $6 \times 4 = 24$ orientations of the cube which we can consider isomorphic. So we can further divide $8! \times 3^8$ by 24 to arrive at

$$\frac{8! \times 3^8}{6 \times 4} = 11,022,480$$

I.e., much less than the upper bounds we derived above.

A better bound



We will show experimentally, not theoretically, that the actual size of the state space is exactly $\frac{1}{3}$ of this number;

$$\frac{1}{3} \times \frac{8! \times 3^8}{24} = 3,674,160\,.$$

Walking the State Space

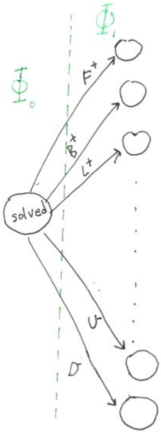
The possible rotations (twists) of the cube can be characterized as follows. Each face can be turned either of two directions, 90°

F+	F-	front
B+	B-	back
L+	L-	left
R+	R-	right
U+	U-	up
D+	D-	down

We will consider 180° twists (F2, B2, L2, R2, U2, and D2) later.

Frontier

Let's define a concept called *frontier*.

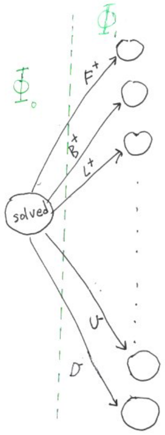


Definition (frontier 0)

The set containing the singleton state of the solved cube, we call this ground state *frontier 0*, denoted Φ_0.

Frontier

Let's define a concept called *frontier*.



Definition (frontier 0)

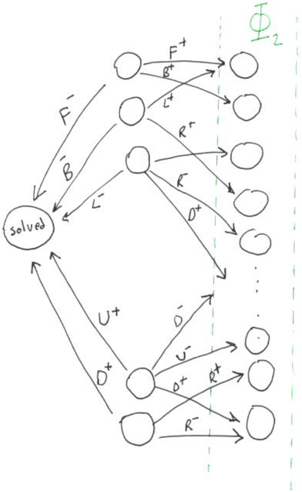
The set containing the singleton state of the solved cube, we call this ground state *frontier 0*, denoted Φ_0.

Definition (frontier 1)

Start from the solved state. The set of states reachable from the solved state by exactly one twist, we call *frontier 1*, and denote it Φ_1.

Frontier

Let's define a concept called *frontier*.



Definition (frontier 2)

The set of states reachable from some state in Φ_1 by exactly one twist, excluding states already encountered, we call this set *frontier 2*, and denote it Φ_2.

If we arrive at *frontier 1* by F$+$, we return to *frontier 0* by F-. Every twist uniquely reversible.

Frontier n

Definition (frontier n)

If $n > 0$, Φ_n (*frontier n*) is the set of states reachable from Φ_{n-1} with a single twist, but are not in $\Phi_0 \cup \Phi_1 \cup \ldots \cup \Phi_{n-1}$.

Size of Frontier n

We can bound the sizes of each frontier.

- $|\Phi_0| = 12^0 = 1$, exactly 1 state, the solved state.
- $|\Phi_1| = 12^1 = 12$, one for each of the 90° twists.
- $|\Phi_2| < 12^2 = 144$.
- Less, because some of the states were seen in $\Phi_0 \cup \Phi_1$, and because some of the 144 are duplicates.
- For example, we start in Φ_0 and arrive in Φ_2 by F$+$ F$+$, or we can arrive at the exact same state by F- F-.

The Final Frontier

Since there are finitely many states of the cube, eventually $|\Phi_k| = 0$.

What is the value of $k - 1$?

I.e., what is the worst scrambled state of the cube? How many twists is it from the solved state?

God's Number

- We will call each state a vertex, and each twist an edge.

God's Number

- We will call each state a vertex, and each twist an edge.
- There is a (not necessarily unique) shortest path from any vertex v to v_{solved}.

God's Number

- We will call each state a vertex, and each twist an edge.
- There is a (not necessarily unique) shortest path from any vertex v to v_{solved}.
- This path represents a minimum sequence of twists to solve the cube from that shuffled state.

God's Number

- We will call each state a vertex, and each twist an edge.
- There is a (not necessarily unique) shortest path from any vertex v to v_{solved}.
- This path represents a minimum sequence of twists to solve the cube from that shuffled state.
- What is the maximum shortest path length?

God's Number

- What is the maximum shortest path length?

God's Number

- What is the maximum shortest path length?
- For the 2x2x2, without 180° twists the answer is 14, and if 180° are allowed, the number is 11.

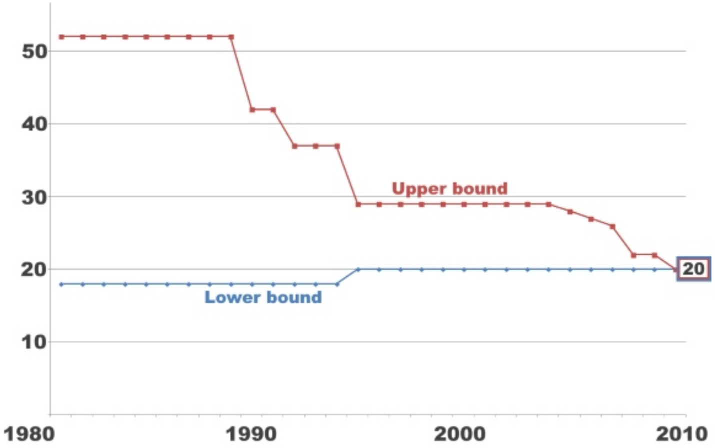
God's Number

- What is the maximum shortest path length?
- For the 2x2x2, without 180° twists the answer is 14, and if 180° are allowed, the number is 11.
- This number is called *God's Number*.

God's Number

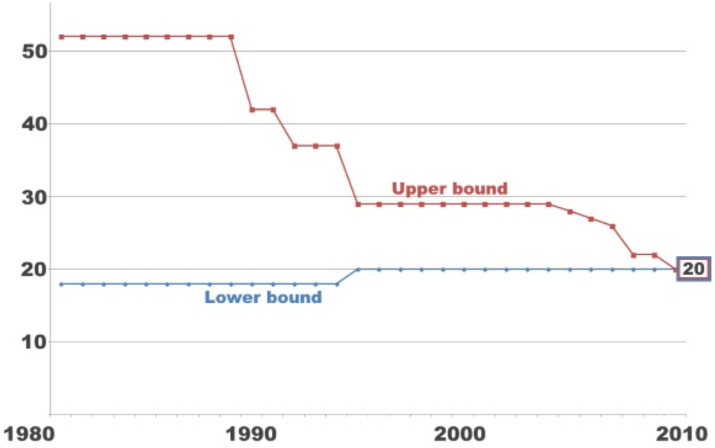
- What is the maximum shortest path length?
- For the 2x2x2, without 180° twists the answer is 14, and if 180° are allowed, the number is 11.
- This number is called *God's Number*.
- It is the number of twists God would need to solve the cube, assuming he always knows the best tactic.

God's Number



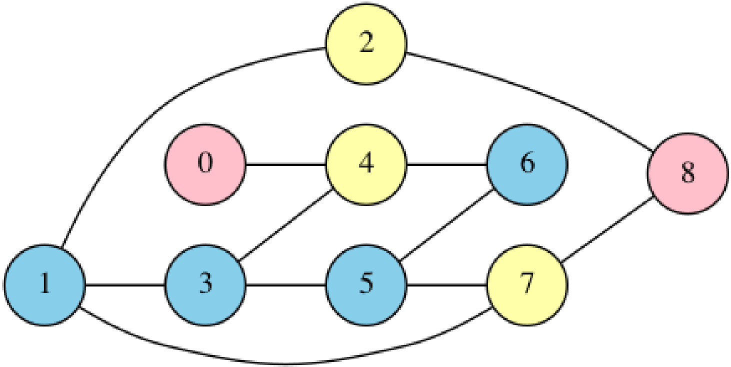
- For the $3 \times 3 \times 3$ cube was proven in 2010 to be 20.

God's Number



- For the $3 \times 3 \times 3$ cube was proven in 2010 to be 20.
- Unknown for any size larger than $3 \times 3 \times 3$.

Bounds on a Graph



Bounds on a Graph

Definition (path)

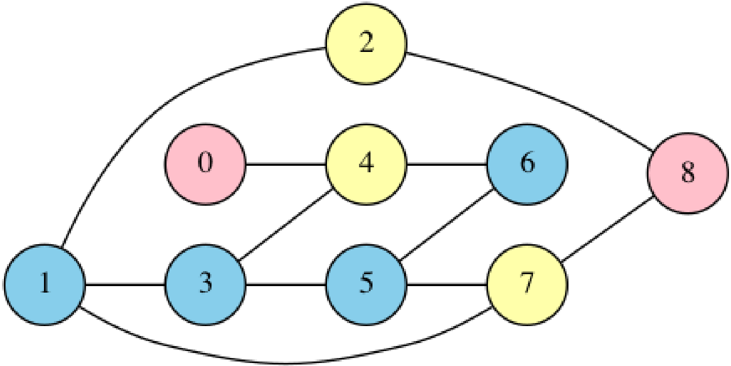
Given a graph, $G = (V, E)$,

let P be a sequence of *distinct* edges $(e_1, e_2, \ldots, e_n)$, with $e_i \in E$ and $|e_i \cap e_{i+1}| = 1$ for $1 < i < n - 1$.

Denote $u \in e_1 \setminus e_2$ and $v \in e_n \setminus e_{n-1}$.

Then P is called a *path* from u to v. The number of elements, n, of P is called the *path length*.

Bounds on a Graph



What are some examples of paths?

Distance

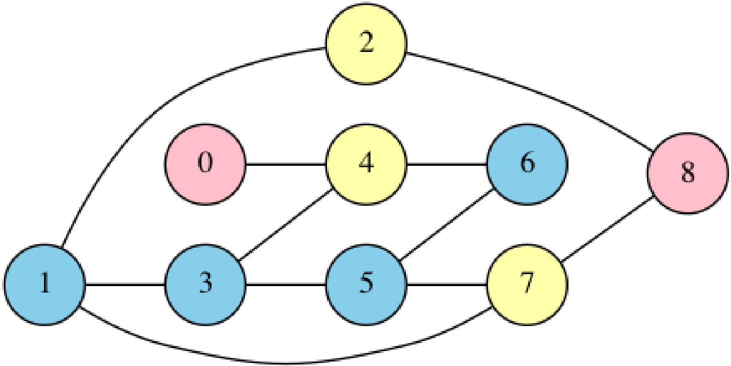
There are finitely many paths from u to v (if edges are distinct).

Definition (distance)

The length of a shortest path from u to v is called the *distance* from u to v, and denoted

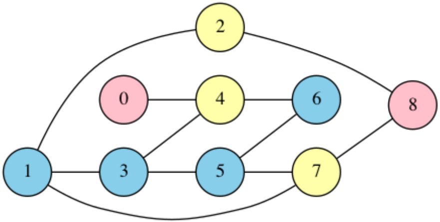
$$\delta(u, v) .$$

Distance



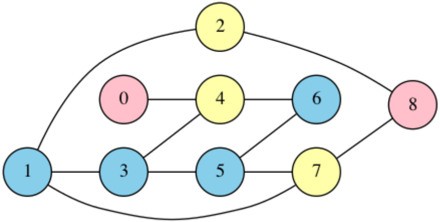
What are some examples of distances?

Distance



	0	1	2	3	4	5	6	7	8
0									
1									
2									
3									
4									
5									
6									
7									
8									

Distance



	0	1	2	3	4	5	6	7	8
0	0								
1	3	0							
2	4	1	0						
3	2	1	2	0					
4	1	2	3	1	0				
5	3	2	3	1	2	0			
6	2	3	4	2	1	1	0		
7	4	1	2	2	3	1	2	0	
8	5	2	1	3	4	2	3	1	0

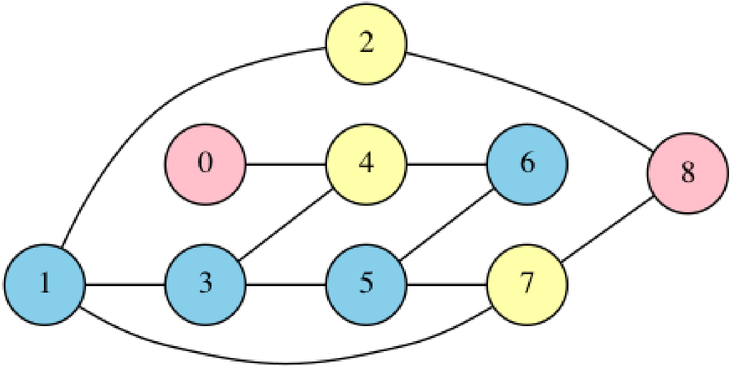
Eccentricity

Definition

Given a graph, $G = (V, E)$. If $v \in V$, the *eccentricity* of v is the maximum distance from v. I.e.,

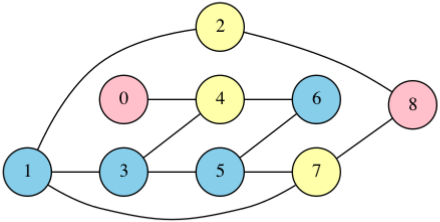
$$\mathit{eccentricity}(v) = \max_{u \in V} \delta(v, u)\,.$$

Eccentricity



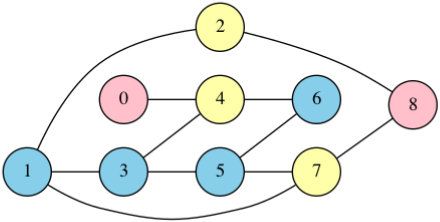
What are some examples of eccentricities?

Eccentricity



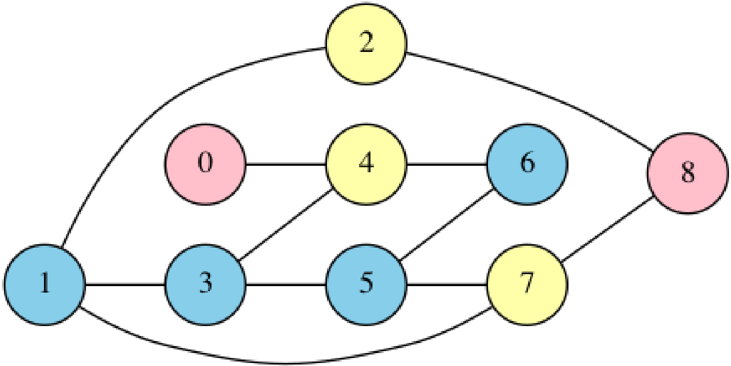
	0	1	2	3	4	5	6	7	8	eccentricity
0	0									
1	3	0								
2	4	1	0							
3	2	1	2	0						
4	1	2	3	1	0					
5	3	2	3	1	2	0				
6	2	3	4	2	1	1	0			
7	4	1	2	2	3	1	2	0		
8	5	2	1	3	4	2	3	1	0	

Eccentricity



	0	1	2	3	4	5	6	7	8	eccentricity
0	0									5
1	3	0								3
2	4	1	0							4
3	2	1	2	0						3
4	1	2	3	1	0					4
5	3	2	3	1	2	0				3
6	2	3	4	2	1	1	0			3
7	4	1	2	2	3	1	2	0		4
8	5	2	1	3	4	2	3	1	0	5

Eccentricity



God's number is the eccentricity of the solved state, v_{solved}.

Diameter

Definition (diameter)

The *diameter* of graph $G = (V, E)$ is

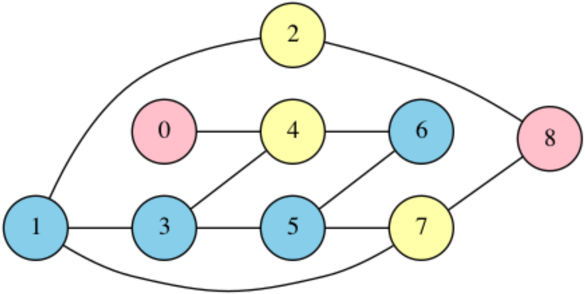
$$d = diameter(G) = \max_{v \in V} eccentricity(v) \, .$$

Diameter

Definition (diameter)

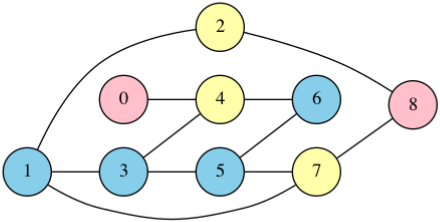
The *diameter* of graph $G = (V, E)$ is

$$d = diameter(G) = \max_{v \in V} eccentricity(v) .$$



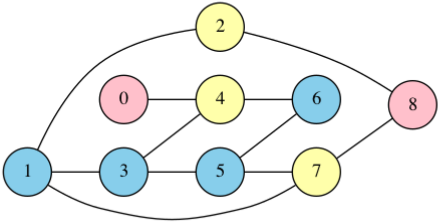
What is the diameter?

Diameter



	0	1	2	3	4	5	6	7	8	eccentricity
0	0									5
1	3	0								3
2	4	1	0							4
3	2	1	2	0						3
4	1	2	3	1	0					4
5	3	2	3	1	2	0				3
6	2	3	4	2	1	1	0			3
7	4	1	2	2	3	1	2	0		4
8	5	2	1	3	4	2	3	1	0	5

Diameter



	0	1	2	3	4	5	6	7	8	eccentricity	
0	0									5	$d = 5$
1	3	0								3	
2	4	1	0							4	
3	2	1	2	0						3	
4	1	2	3	1	0					4	
5	3	2	3	1	2	0				3	
6	2	3	4	2	1	1	0			3	
7	4	1	2	2	3	1	2	0		4	
8	5	2	1	3	4	2	3	1	0	5	$d = 5$

Radius

Definition (radius)

The *radius* of graph $G = (V, E)$ is

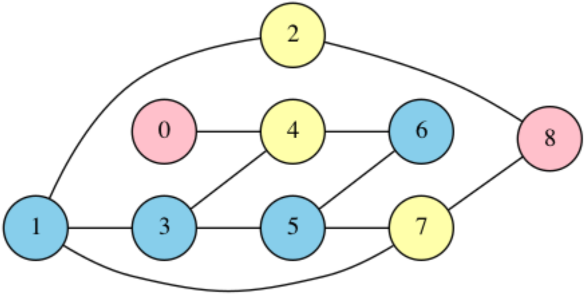
$$r = radius(G) = \min_{v \in V} eccentricity(v) .$$

Radius

Definition (radius)

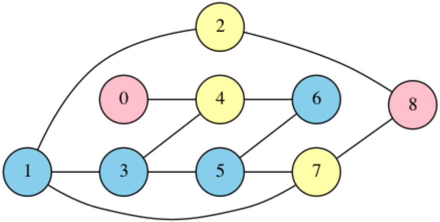
The *radius* of graph $G = (V, E)$ is

$$r = radius(G) = \min_{v \in V} eccentricity(v)\,.$$



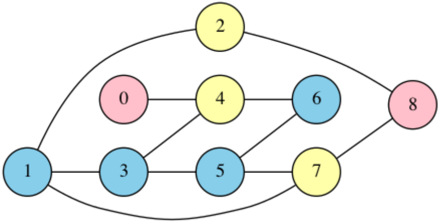
What is the radius?

Radius



	0	1	2	3	4	5	6	7	8	eccentricity	
0	0									5	
1	3	0								3	$r = 3$
2	4	1	0							4	
3	2	1	2	0						3	$r = 3$
4	1	2	3	1	0					4	
5	3	2	3	1	2	0				3	$r = 3$
6	2	3	4	2	1	1	0			3	$r = 3$
7	4	1	2	2	3	1	2	0		4	
8	5	2	1	3	4	2	3	1	0	5	

Relation of Diameter and Radius



	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**eccentricity**	
0	0									5	$d = 5$
1	3	0								3	$r = 3$
2	4	1	0							4	
3	2	1	2	0						3	$r = 3$
4	1	2	3	1	0					4	
5	3	2	3	1	2	0				3	$r = 3$
6	2	3	4	2	1	1	0			3	$r = 3$
7	4	1	2	2	3	1	2	0		4	
8	5	2	1	3	4	2	3	1	0	5	$d = 5$

Relation of Diameter and Radius

$$
\begin{aligned}
radius &= 3 \\
diameter &= 5 \\
diameter &\leq 2 \times radius
\end{aligned}
$$

Radius

Claim:

$$diameter(G) \leq 2 \times radius(G)$$

Proof:
Let D be $diameter(G)$, and $R = radius(G)$.
Let $v_1, v_2 \in V$ such that $\delta(v_1, v_2) = D$.
And take $u \in V$ such that $eccentricity(u) = R$.

$$D \leq \delta(v_1, u) + \delta(u, v_2)$$

Radius

Claim:

$$diameter(G) \leq 2 \times radius(G)$$

Proof:
Let D be $diameter(G)$, and $R = radius(G)$.
Let $v_1, v_2 \in V$ such that $\delta(v_1, v_2) = D$.
And take $u \in V$ such that $eccentricity(u) = R$.

$$D \leq \delta(v_1, u) + \delta(u, v_2)$$

Why?

Radius

Claim:

$$diameter(G) \leq 2 \times radius(G)$$

Proof:
Let D be $diameter(G)$, and $R = radius(G)$.
Let $v_1, v_2 \in V$ such that $\delta(v_1, v_2) = D$.
And take $u \in V$ such that $eccentricity(u) = R$.

$$D \leq \delta(v_1, u) + \delta(u, v_2)$$

Why? Because, if $\delta(v_1, v_3) + \delta(v_3, v_2)$ were less than D we would have a shorter path from v_1 to v_2, and by assumption the shortest path has length D.

Radius

Claim:

$$diameter(G) \leq 2 \times radius(G)$$

Proof:
Let D be $diameter(G)$, and $R = radius(G)$.
Let $v_1, v_2 \in V$ such that $\delta(v_1, v_2) = D$.
And take $u \in V$ such that $eccentricity(u) = R$.

$$D \leq \delta(v_1, u) + \delta(u, v_2)$$

We know $\delta(v_1, u) \leq R$ and $\delta(u, v_2) \leq R$.
Why?

Radius

Claim:

$$diameter(G) \leq 2 \times radius(G)$$

Proof:
Let D be $diameter(G)$, and $R = radius(G)$.
Let $v_1, v_2 \in V$ such that $\delta(v_1, v_2) = D$.
And take $u \in V$ such that $eccentricity(u) = R$.

$$D \leq \delta(v_1, u) + \delta(u, v_2)$$

We know $\delta(v_1, u) \leq R$ and $\delta(u, v_2) \leq R$.
Why? Because R is the eccentricity of u, every vertex is within a distance R of u.

Radius

Claim:

$$diameter(G) \leq 2 \times radius(G)$$

Proof:
Let D be $diameter(G)$, and $R = radius(G)$.
Let $v_1, v_2 \in V$ such that $\delta(v_1, v_2) = D$.
And take $u \in V$ such that $eccentricity(u) = R$.

$$D \leq \delta(v_1, u) + \delta(u, v_2)$$

We know $\delta(v_1, u) \leq R$ and $\delta(u, v_2) \leq R$.
Thus

$$D \leq \delta(v_1, v_3) + \delta(v_3, v_2) \leq R + R = 2R\,.$$

Periphery

Definition (periphery, peripheral vertex)

Given a graph, $G = (V, E)$, the set $P_G \subset V$ of vertices whose eccentricity equal the diameter is called the *periphery*.

$$P_G = \{v \in V \mid \mathit{eccentricity}(v) = \mathit{diameter}(G)\}$$

A vertex in the periphery of a graph is called a *peripheral vertex*.

Center

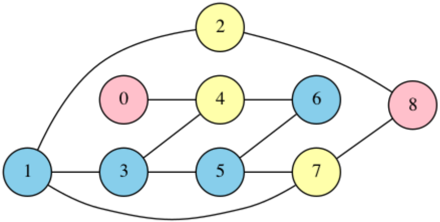
Definition (center, central vertex)

Given a graph, $G = (V, E)$, the set $C_G \subset V$ of vertices whose eccentricity equal the radius is called the *center*.

$$C_G = \{v \in V \mid \mathit{eccentricity}(v) = \mathit{radius}(G)\}$$

A vertex in the center of a graph is called a *central vertex*.

Periphery and Center



	0	1	2	3	4	5	6	7	8	eccentricity	
0	0									5	periphery
1	3	0								3	center
2	4	1	0							4	
3	2	1	2	0						3	center
4	1	2	3	1	0					4	
5	3	2	3	1	2	0				3	center
6	2	3	4	2	1	1	0			3	center
7	4	1	2	2	3	1	2	0		4	
8	5	2	1	3	4	2	3	1	0	5	periphery

Regularity of the Rubik's Graph

In the Rubik's state graph, every vertex is central and every vertex is peripheral.

The eccentricity of every vertex is the same: God's Number.

Implementing the Rubik's cube

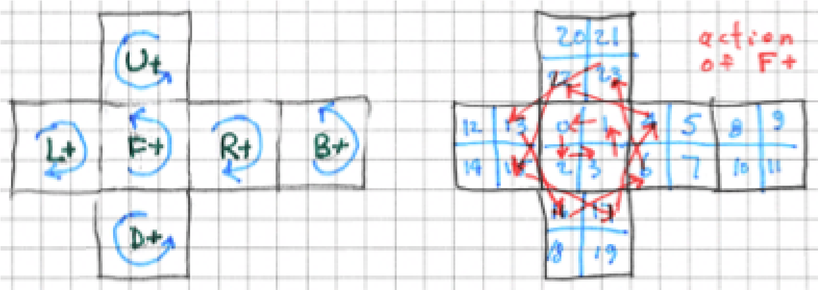
We would like to implement the Rubik's cube to find God's Number.

Cycle Notation

- Use *cycle notation* to specify a permutation.
- A permutation that transforms (123456) to (421365) can be denoted as (143)(56),
- meaning $1 \mapsto 4 \mapsto 3 \mapsto 1$ and $5 \mapsto 6 \mapsto 5$.

Implementing the Rubik's cube

Represent a twist in terms of the cycles of its permutation of the facelets.



F+ is represented as (0 2 3 1)(4 22 15 17)(6 23 13 16).

Implementing the Rubik's cube

Represent each forward 90° twist as cycles.

F+	=	(0 2 3 1)	(4 22 15 17)	(6 23 13 16)
B+	=	(5 19 14 20)	(7 18 12 21)	(8 10 11 9)
L+	=	(0 16 11 20)	(2 18 9 22)	(12 13 15 14)
R+	=	(1 21 10 17)	(3 23 8 19)	(4 5 7 6)
U+	=	(0 4 8 12)	(1 5 9 13)	(20 22 23 21)
D+	=	(2 14 10 6)	(3 15 11 7)	(16 18 19 17)

Implementing the Rubik's cube

The inverse rotations, can be formed by reversing the lists.

$$
\begin{array}{lcl l l}
\text{F-} & = & (1\ 3\ 2\ 0) & (17\ 15\ 22\ 4) & (16\ 13\ 23\ 6) \\
\text{B-} & = & (20\ 14\ 19\ 5) & (21\ 12\ 18\ 7) & (9\ 11\ 10\ 8) \\
\text{L-} & = & (20\ 11\ 16\ 0) & (22\ 9\ 18\ 2) & (14\ 15\ 13\ 12) \\
\text{R-} & = & (17\ 10\ 21\ 1) & (19\ 8\ 23\ 3) & (6\ 7\ 5\ 4) \\
\text{U-} & = & (12\ 8\ 4\ 0) & (13\ 9\ 5\ 1) & (21\ 23\ 22\ 20) \\
\text{D-} & = & (6\ 10\ 14\ 2) & (7\ 11\ 15\ 3) & (17\ 19\ 18\ 16)
\end{array}
$$

Implementing the Rubik's cube

The 180° rotations are double applications of 90° twists.

$$
\begin{aligned}
F2 &= (1\ 3\ 2\ 0) \quad (17\ 15\ 22\ 4) \quad (16\ 13\ 23\ 6) \\
&\ (1\ 3\ 2\ 0) \quad (17\ 15\ 22\ 4) \quad (16\ 13\ 23\ 6) \\
&= (1\ 2)\ (3\ 0) \quad (17\ 22)\ (15\ 4) \quad (16\ 23)\ (13\ 6) \\
B2 &= (20\ 14\ 19\ 5) \quad (21\ 12\ 18\ 7) \quad (9\ 11\ 10\ 8) \\
&\ (20\ 14\ 19\ 5) \quad (21\ 12\ 18\ 7) \quad (9\ 11\ 10\ 8) \\
L2 &= (20\ 11\ 16\ 0) \quad (22\ 9\ 18\ 2) \quad (14\ 15\ 13\ 12) \\
&\ (20\ 11\ 16\ 0) \quad (22\ 9\ 18\ 2) \quad (14\ 15\ 13\ 12) \\
R2 &= (17\ 10\ 21\ 1) \quad (19\ 8\ 23\ 3) \quad (6\ 7\ 5\ 4) \\
&\ (17\ 10\ 21\ 1) \quad (19\ 8\ 23\ 3) \quad (6\ 7\ 5\ 4) \\
U2 &= (12\ 8\ 4\ 0) \quad (13\ 9\ 5\ 1) \quad (21\ 23\ 22\ 20) \\
&\ (12\ 8\ 4\ 0) \quad (13\ 9\ 5\ 1) \quad (21\ 23\ 22\ 20) \\
D2 &= (6\ 10\ 14\ 2) \quad (7\ 11\ 15\ 3) \quad (17\ 19\ 18\ 16) \\
&\ (6\ 10\ 14\ 2) \quad (7\ 11\ 15\ 3) \quad (17\ 19\ 18\ 16)
\end{aligned}
$$

Optimization

To reduce memory consumption, don't use all 12 90° twists, but rather 6 of them:

D-, D+, R-, R+, B-, and B+.

Or incorporate the 180° degree twists, use 9 twists:

D-, D+, D2, R-, R+, R2, B-, B+, and B2.

Why does this work?

Optimization

For example, we can effectuate the L+ twist by performing R- and then rolling the entire cube toward myself.

R+ L- simply rotates the entire cube.

Similar for the other two pairs of opposite sides.

Optimization

We could represent a state of the cube as a 24 character string consisting of the letters `'W'`, `'G'`, `'Y'`, `'B'`, `'O'`, and `'R'`, such as `"BGWRGYYRBYWBGOROWORB"`. However, this would require 24 bytes per state.

Instead, we can represent `"BGWRGYYRBYWBGOROWORB"` as a 24 digit base 6 number, if we map the characters as follows:
$W \to 0$, $G \to 1$, $Y \to 2$, $B \to 3$, $O \to 4$, and $R \to 5$.

The maximum such number is

$$
\begin{aligned}
555544443333222211110000_6 &= 4{,}737{,}649{,}541{,}975{,}930{,}160_{10} \\
&< 2^{62.04} \\
&< 2^{63} - 1\,.
\end{aligned}
$$

The cube is encoded as a positive 64 bit integer: 8 bytes.

Optimization

To rotate apply $D^+ = ((2\,14\,10\,6)(3\,15\,1\,7)(16\,18\,19\,17))$ to cube $=$ `RRRROOOOBBBBYYYYGGGGWWWW`$=$ $555544443333222211110000_6 = 4737649541975930160_{10}$:

- $555544443333222211110000_6 \times (2\,14\,10\,6) \to 555544443033232212110100_6$
- Computing the digits

$$d_2 = \left\lfloor \frac{cube}{6^2} \right\rfloor \ \% \ 6 = 0$$

$$d_6 = \left\lfloor \frac{cube}{6^6} \right\rfloor \ \% \ 6 = 1$$

$$d_{10} = \left\lfloor \frac{cube}{6^{10}} \right\rfloor \ \% \ 6 = 2$$

$$d_{14} = \left\lfloor \frac{cube}{6^{14}} \right\rfloor \ \% \ 6 = 3$$

Optimization

To rotate apply $D^+ = ((2\,14\,10\,6)(3\,15\,1\,7)(16\,18\,19\,17))$ to cube $=$ `RRRROOOOBBBBYYYYGGGGWWWW`$=$ $555544443333222211110000_6 = 4737649541975930160_{10}$:

- $555544443333222211110000_6 \times (2\,14\,10\,6) \rightarrow 555544443033232212110100_6$
- $cube + (d_6 \times 6^2 - d_2 \times 6^2) + (d_2 \times 6^{14} - d_{14} \times 6^{14}) + (d_{14} \times 6^{10} - d_{10} \times 6^{10}) + (d_{10} \times 6^6 - d_6 \times 6^6)$

  $= cube + (d_6 - d_2) \times 6^2 + (d_2 - d_{14}) \times 6^{14} + (d_{14} - d_{10}) \times 6^{10} + (d_{10} - d_6) \times 6^6$

  $= cube + 1 \times 6^2 - 3 \times 6^{14} + 1 \times 6^{10} + 1 \times 6^6$

  $= 4737641078706720660_{10}$

Homework `solverubik.py`

Implement `solve_cube(cube, twists)`.

Given a shuffled cube (`str`) and a list of legal twists (*e.g.*, `["D-", "D+", "R-", "R+", "B-", "B+"]`, return a list such as `["D-", "R+", "D+" "R+"]` which solves the given cube using only legal twists.

API:

- `solved(cube:int) -> bool`
- `cubeToInt(cube:str) -> int`
- `rotateCube(cube: int, twist: str) -> int`

Breadth first search BFS

Use the BFS to examine the Rubik's cube graph.

Algorithm 1: BFS(s,Adj)

Input : s a starting vertex
Input : Adj graph adjacency list, as a mapping

```
level[s] ← 0
Π[s] ← Top
fifo.push(s)
while fifo not empty do
    v ← fifo.pop()
    for u ∈ Adj[v] do
        if not Π[u] then
            Π[u] ← v
            level[u] ← 1 + level[v]
            fifo.push(u)
```

Complexity of BFS

Every reachable edge goes into the fifo exactly once (push), and comes out (pop) exactly once.
The `while` loop repeats $|V|$ times.
The complexity is

$$O(|V| + |E|)$$

The key insight is that *for* $u \in Adj[v]$ occurs once per edge (or twice for an undirected graph). Recall

$$\sum_{v \in V} degree(v) = 2|E| = \Theta(|E|)$$

Alternate BFS Algorithm

A 2nd approach to BFS: use two stacks,

- one for pushing
- and one for popping.

Alternate BFS Algorithm

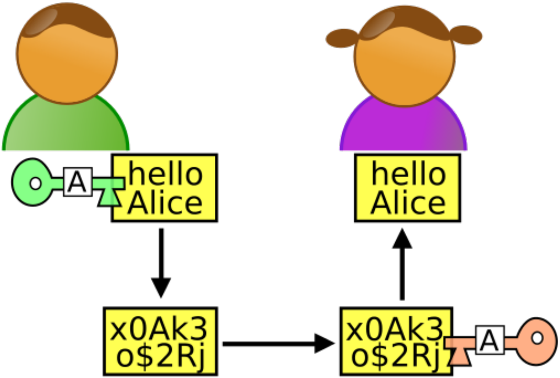
s is the starting vertex. *Adj* is the adjacency list.

Algorithm 2: BFS(s,Adj)

```
level[s] ← 0
ecc ← 0
Π[s] ← Top
frontier ← empty stack
frontier.push(s)
while frontier not empty do
    next ← empty stack
    for v ∈ frontier do
        for u ∈ Adj[v] do
            if not Π[u] then
                next.push(u)
                Π[u] ← v
                level[u] ← ecc
    frontier ← next
    ecc = 1 + ecc
```

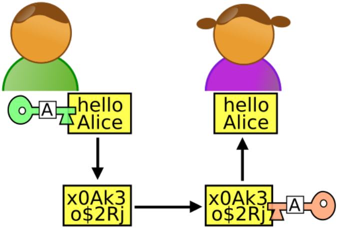
Zero Knowledge Proofs and Graph Isomorphism

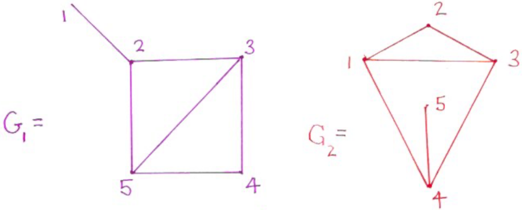
Zero Knowledge Proofs and Graph Isomorphism



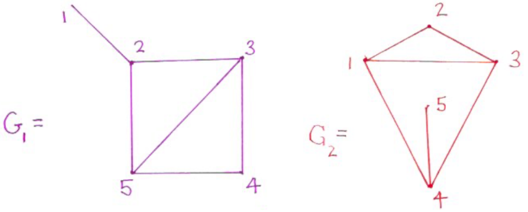
https://commons.wikimedia.org/wiki/File:Asymmetric_cryptography_-_step_2.svg

Zero Knowledge Proofs and Graph Isomorphism





Zero Knowledge Proofs and Graph Isomorphism



- Given two *public* graphs G_1 and G_2.
- Alice knows the isomorphism $h = (5\ 3\ 1)(2\ 4)$,
- but Bob does not.
- Recall that it is difficult for Bob to compute h,
- but easy to verify.

Zero Knowledge Proofs and Graph Isomorphism

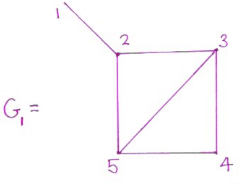
Question: Can Alice convince Bob that she knows h without revealing it to Bob nor allowing Bob to reconstruct it?

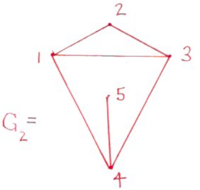
Zero Knowledge Proofs and Graph Isomorphism

Question: Can Alice convince Bob that she knows h without revealing it to Bob nor allowing Bob to reconstruct it?

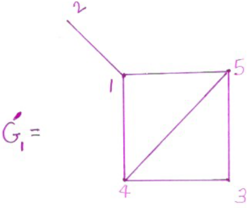
Answer: Yes! Here is how.

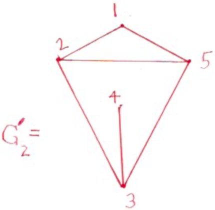
Zero Knowledge Proofs and Graph Isomorphism



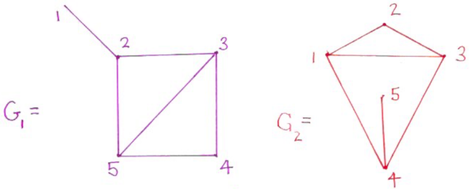


- Alice chooses a secret random permutation of $[1, 2, 3, 4, 5]$. E.g., $\mu = (1\ 2)(5\ 3\ 4)$
- Alice randomly chooses G_1 or G_2. I.e., $a \in \{1, 2\}$.
- Alice computes $H = \mu G_a$.
- I.e., either $H = \mu G_1 = G_1'$ or $H = \mu G_2 = G_2'$.

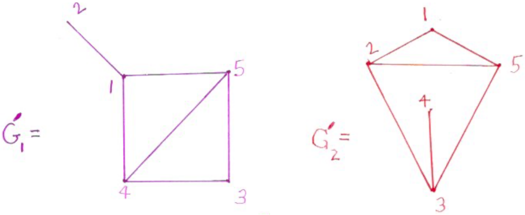




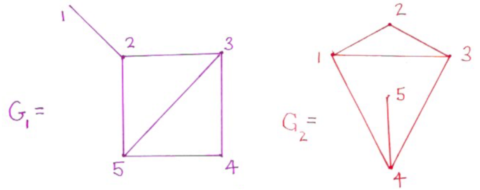
Zero Knowledge Proofs and Graph Isomorphism



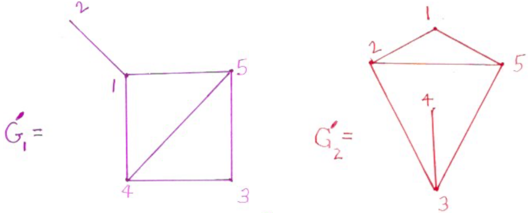
- Alice sends H to Bob, but neither μ nor a.
- Bob does not know whether he has $H = G_1'$ or $H = G_2'$.



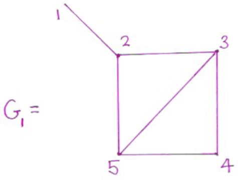
Zero Knowledge Proofs and Graph Isomorphism

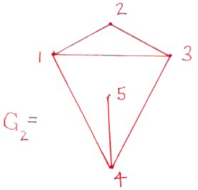


- Bob has $H \in \{\mu G_1, \mu G_2\} = \{G_1', G_2'\}$,
- but he doesn't know which.
- Bob randomly chooses G_1 or G_2. I.e., $b \in \{1, 2\}$.
- Bob sends b to Alice.
- Bob challenges Alice to produce λ such that $\lambda H = G_b$



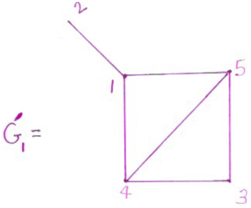
Zero Knowledge Proofs and Graph Isomorphism

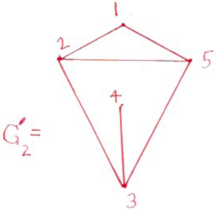




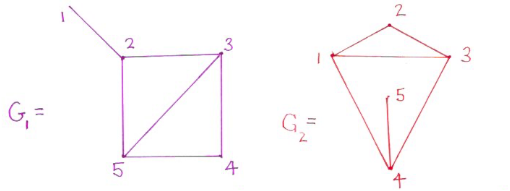
I.e., Bob asks for the isomorphism from H to G_1 or G_2.

He will easily be able to verify the isomorphism, as that is easy.

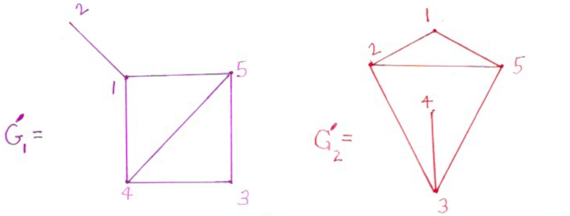




Zero Knowledge Proofs and Graph Isomorphism



- Alice computes λ and sends it to Bob.
- Bob verifies that $\lambda H = G_b$

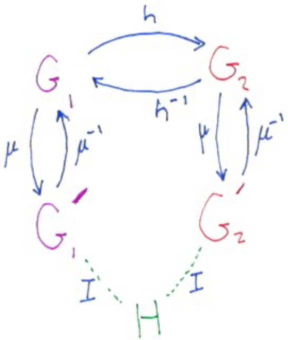


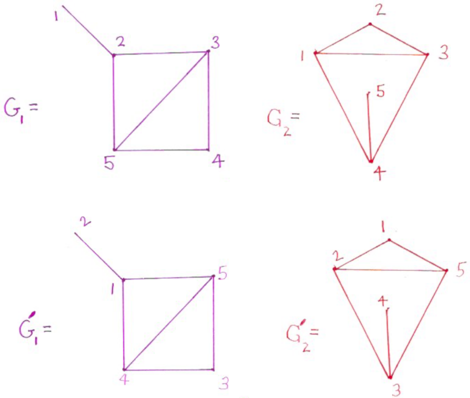
Zero Knowledge Proofs and Graph Isomorphism

- Alice computes λ and sends it to Bob.
- Bob verifies that $\lambda H = G_b$

Question: Should Bob be convinced that Alice knows h ?

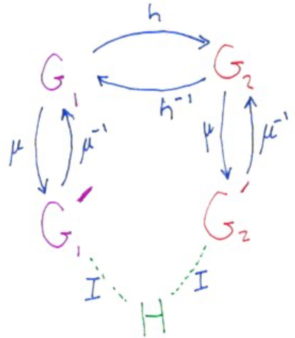
Zero Knowledge Proofs and Graph Isomorphism





Zero Knowledge Proofs and Graph Isomorphism

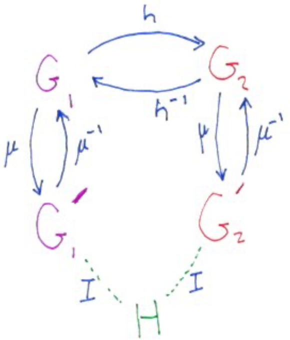
Find λ such that $\lambda H = \lambda G'_a = G_b$.



a	G'_a (b: G_b)	1: G_1	2: G_2
1	$H = G'_1$	μ^{-1}	$h \circ \mu^{-1}$
2	$H = G'_2$	$h^{-1} \circ \mu^{-1}$	μ^{-1}

Zero Knowledge Proofs and Graph Isomorphism

Bob is not 100% convinced.



- Bob knows b but not a.
- There is a 50% chance that $a = b$,
- Alice might have sent μ^{-1} without knowing h.
- So Bob must repeat the experiment n times,
- to have a certainty of $1 - \frac{1}{2^n}$.