SOLVING THE CUBE – PT. 4

MICHAEL GEORGE

WCA ID: 2015GEOR02

PERMUTATION OF THE LAST LAYER (PLL)

Approach

- Permutation of the Last Layer (**PLL**) will be broken into in 2 sub-steps
  1. Corner Permutation of the Last Layer (**CPLL**)
  2. Edge Permutation of the Last Layer (**EPLL**)
- Manipulation of the **F2L** can be used to change the permutation of the last layer
  - e.g. Extracting an **F2L pair** (or two) then re-inserting the pair(s) using a different "**triggers**"
- This approach requires two "**algorithms**" which are essentially combinations of simple "**algorithms**"
  1. Algorithm for corner permutation (**CPLL**) – Combination of 2 simple algorithms using **R U L** moves
  2. Algorithm for edge permutation (**EPLL**) – Combination of 2 simple algorithms using **R U L** moves

CORNER PERMUTATION OF THE LAST LAYER (CPLL)

There are 3 possible cases during **CPLL** including the "**solved**" case

Cases: The cases that need to be solved are the "**diagonal corner swap**" and the "**adjacent corner swap**"

Probabilities: The "**adjacent**" corner swap is the most common **CPLL** as it occurs in **4/6** solves

Approach: A combination of 2 simple algorithms can be used to cycle through the **CPLL** cases

1/6

4/6

1/6

CPLL ALGORITHM

- To change the permutation of corners you will use a mixture of **R U L** moves
- The **CPLL** algorithm that will be used is **(R U2 R' U2') (U R U' R') (R U' L' U) (R' U' L U)**
  - It may look daunting but it is actually two shorter algorithms
  - It starts with the **Anti-Sune** algorithm that was used for **OCLL** – **(R U2 R' U2') (U R U' R')**
  - It ends with the **Niklas** algorithm which manipulates two **F2L pairs** – **(R U' L' U) (R' U' L U)**
  - The combination of these two algorithms is the **Jb-Perm** which swaps 2 LL corners and 2 LL edges
- Explanation of the algorithm(s)
  - The **Anti-Sune** algorithm disorients 3 of the LL corners and permutes 3 of the LL edges
  - The **Niklas** algorithm re-orients the LL corners and changes their permutation
  - The combined effect of these two algorithms is to maintain the **OLL** but change the **PLL**
- Setup for the algorithm
  - The key to using this algorithm is knowing the appropriate "**setup**" prior to execution

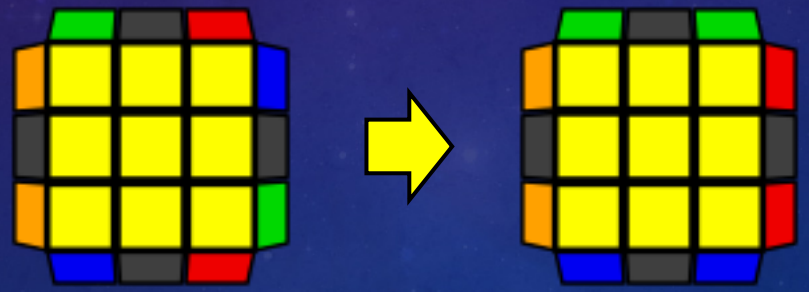
ADJACENT CORNER SWAP

The “**adjacent**” corner swap can be recognised by the “**headlights**” on one side (shown in orange below)

Approach: “**Adjacent corner swap**” -> “**solved**”

Setup: Ensure the “**Headlights**” are on the left before executing the **CPLL** algorithm

Algorithm: **(R U2 R’ U2’) (U R U’ R’) (R U’ L’ U) (R’ U’ L U)**



Adjacent Swap **Solved**

DIAGONAL CORNER SWAP

The “**diagonal**” corner swap can be recognised by the absence of “**headlights**”

Approach: “**Diagonal corner swap**” -> “**adjacent corner swap**” -> “**solved**”

Setup: Ensure the “**Headlights**” are on the left before executing the ”**adjacent corner swap**”

Algorithm: **(R U2 R’ U2’) (U R U’ R’) (R U’ L’ U) (R’ U’ L U)** – twice, remembering to **AUF** prior to execution

Diagonal Swap **Adjacent Swap** **Solved**

EDGE PERMUTATION OF THE LAST LAYER (EPLL)

There are 5 possible cases during **EPLL** including the "**solved**" case

4/12

4/12

2/12

1/12

1/12

EDGE PERMUTATION OF THE LAST LAYER (EPLL)

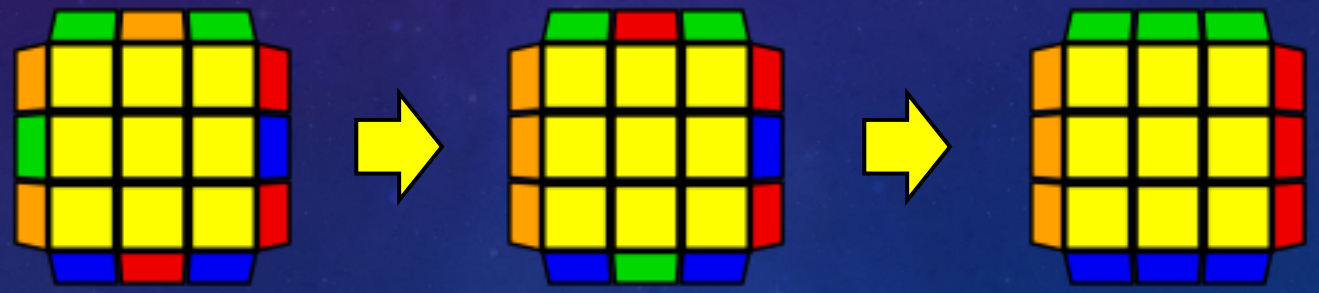
EPLL can be solved by cycling through the cases in a similar fashion to **CPLL**

Cases: There can only be 3 or 4 misplaced edges, corresponding to the images on the previous slide

Probabilities: The most common cases are the "**3-cycles**" as they occur in **8/12** solves

Approach: Cycle from **4** misplaced edges -> **3** misplaced edges -> "**solved**"

e.g.



Z-Perm **Ub-Perm** **Solved**

EPLL ALGORITHM

- To change the permutation of edges you can use a mixture of **R U L** moves
- The **EPLL** algorithm that will be used is **(R U2 R' U2') (U R U' R') U (L' U2 L U2') (U' L' U L)**
  - It may look daunting but it is actually two **Anti-Sune** algorithms
  - It starts with the **Anti-Sune** algorithm that was used for **OCLL** – **(R U2 R' U2') (U R U' R')**
  - It ends with the left-handed **"mirror"** – **(L' U2 L U2') (U' L' U L)**
  - The combination of these two algorithms is the **Ub-Perm** which is a clockwise "**3-cycle**" of LL edges
- Explanation of the algorithm(s)
  - The first **Anti-Sune** algorithm disorients 3 of the LL corners and permutes 3 of the LL edges
  - The second **Anti-Sune** algorithm re-orients all of the LL corners and permutes 3 of the LL edges
  - The combined effect of the two algorithms is to maintain the **OLL** but change the **EPLL**

3 MISPLACED EDGES

There are 2 **EPLL** cases with 3 misplaced edges and they can be recognised by the “**bar**” (orange below)

Approach: “**Ua-Perm**” -> “**Ub-Perm**” -> “**Solved**”

Setup: Ensure the ”**Bar**” is on the left before executing the **EPLL** algorithm

Algorithm: **(R U2 R’ U2’) (U R U’ R’) U (L’ U2 L U2’) (U’ L’ U L)** – once or twice, remembering to **AUF**

Ua-Perm **Ub-Perm** **Solved**

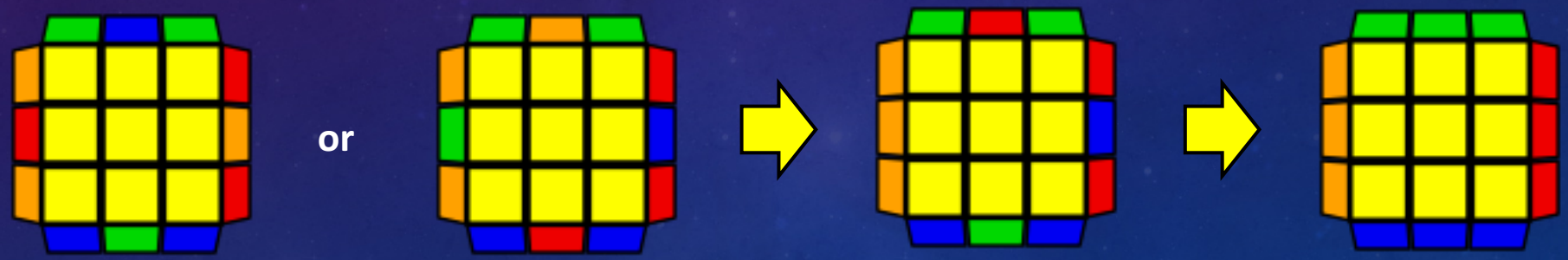
4 MISPLACED EDGES

There are 2 **EPLL** cases with 4 misplaced edges and they take the most effort to solve

Approach: "**Z-Perm**" or "**H-Perm**" -> "**Ub-Perm**" -> "**Solved**"

Setup: Avoid the "**Ua-Perm**" by setting up the Z-Perm correctly before executing the **EPLL** algorithm

Algorithm: **(R U2 R' U2') (U R U' R') U (L' U2 L U2') (U' L' U L)** – twice, remembering to **AUF**



H-Perm **Z-Perm** **Ub-Perm** **Solved**

FINGER TRICKS

(R U2 R’ U2’) (U R U’ R’)

1. Use your index finger(s) to turn the U-layer for the “**setup**” instead of rotating the cube around the y-axis
2. Re-grip so that your right thumb is underneath the R-face and fingers are on top before executing the algorithm
3. Execute the whole algorithm without re-gripping
4. The **U2** can be executed as a “**double flick**” – i.e. index finger followed by middle finger

(L’ U2 L U2’) (U’ L’ U L)

1. Use your index finger(s) to turn the U-layer for the “**setup**”
2. Execute the whole algorithm without re-gripping
3. The **U2** can be executed as a “**double flick**” – i.e. index finger followed by middle finger

CONGRATULATIONS!