

Hyper-Parity

A zero-algorithm method for solving corners in Domino Reduction

Rodney Kinney, June 2020

A video explaining the contents of this document is available at

[▶ Using Hyper-Parity To Solve Corners In Domino Reduction](#)

Motivation

In the course of learning techniques for the fewest-moves challenge (FMC) event, I was often frustrated trying to find a finish to a Domino Reduction (DR) start. I would build blocks only to end up with bad parity. I would attempt a half-turn reduction (HTR) and discover that the corners were off. The most consistent advice I heard was to look for a DR with “easy” corners, but even this was not obvious to me. Ortega and VOP algorithms were usually too long or required lengthy setups. I decided to categorize all of the DR corner cases and ended up with a method that gives the shortest solution to solve corners from any DR configuration. This document describes that method.

When are corners in HTR?

When the cube is in HTR, it can be solved using only half turns. Bringing all F/B stickers to the F/B faces is necessary for HTR, but not sufficient. Let's call the distribution of stickers on the F/B faces the FB orientation. This cube configuration has the correct FB orientation, but is not in HTR:



A cube with correct UD and FB orientation, but incorrect parity.

This configuration has two diagonal corners swapped, as in a Y-perm, so it has odd parity. Is fixing parity enough to bring the cube into HTR? No. Below is a configuration that has correct FB orientation and even parity, but is still not in HTR. It has incorrect UD orientation, needing a UBL-UFR-DFL corner commutator to solve.



A cube with correct parity and FB orientation, but incorrect UD orientation.

To be in HTR, three conditions must be satisfied: Parity correct, UD orientation correct, and FB orientation correct. Half turns do not change any of these three conditions, so our strategy will be to count half turns as neutral and track quarter turns as we bring the cube into HTR. Since we are in Domino Reduction, the only quarter turns available are $U/U'/D/D'$.

Parity


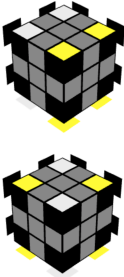



Parity is normally defined as even or odd, by the number of pairwise swaps that must be executed to solve the cube. We will think of it as the number of quarter turns needed to bring the cube into HTR, either even or odd. Each quarter turn changes the cube from one state to the other. Determine Parity by counting corner cycles using blind-solving-style tracing. An odd cycle adds zero to Parity; an even cycle adds one. $5c = \text{even}$. $6c = \text{odd}$. $5c2c = \text{odd}$. $2c2c3c = \text{even}$, etc.



	Even	Odd
Even		<any>
Odd	<any>	

Any quarter turn changes Parity from Even to Odd and from Odd to Even










UD Orientation

There are many possible U/D sticker configurations, but since we are counting half-turns as neutral we will consider any configurations that are separated by only half-turns and rotations as the same UD state. We can think of UD orientation in the same way as parity, i.e. the number of quarter turns needed to bring the cube into HTR. This is the motivation for the name “hyper-parity.” You can also identify them by how the U/D stickers are distributed between the two HTR-invariant tetrads (UFR-RBD-DFL-LBU and UFL-LBD-DFR-RBU). There are four possible UD states:

State	Reachable from	Quarter turns to solve	Tetrad Split
 Solved		0	2-2
 Bars		1	2-2
 Slashes		1	4-0

 <p>Bar/Slash</p>		2	3-1
--	---	---	-----




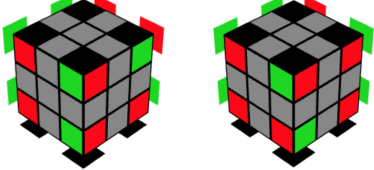
How do quarter turns affect the UD state? Any U/U'/D/D' move will change the state in the same way, depending on which stickers are showing on the U face. It is possible to stay in the Solved or Bar/Slash states after a quarter turn, but not the Slashes or Bars states. The figure below shows how the U face determines the change in the UD state. To use this diagram, find the row that corresponds to your current state. Then find the square that shows the stickers on your U face. A U turn will bring you to the state in the corresponding column.





	Slashes	Solved	Bars	Bar/Slash
Slashes				
Solved				
Bars				
Bar/Slash				 

Possible transitions of the UD state following a quarter turn,
depending on the configuration of the U face.









FB Orientation

Just as before, we treat all configurations that are separated by only half-turns as the same state, and label states by the number of quarter turns needed to bring the cube into HTR. There are four FB states. To recognize them, we treat F/B stickers as equivalent (shown as green) and R/L stickers as equivalent (shown as red).

State	Equivalent to	Quarter turns to solve	Recognition
 Solved		0	No bad corners
 One Face		1	4 bad corners Can bring all bad corners to the same face using half turns
		2	4 bad corners Impossible to bring all bad corners to the same face using half turns

 Bars			
 One Bar	 	3	2 bad corners

Like before, any U/U'/D/D' quarter turn will change the FB state in the same way, depending on which stickers are showing on the U face. A quarter turn can leave the One Face and One Bar states unchanged, but not the Solved or Bars states. The figure below shows how the U face determines the change in the FB state.

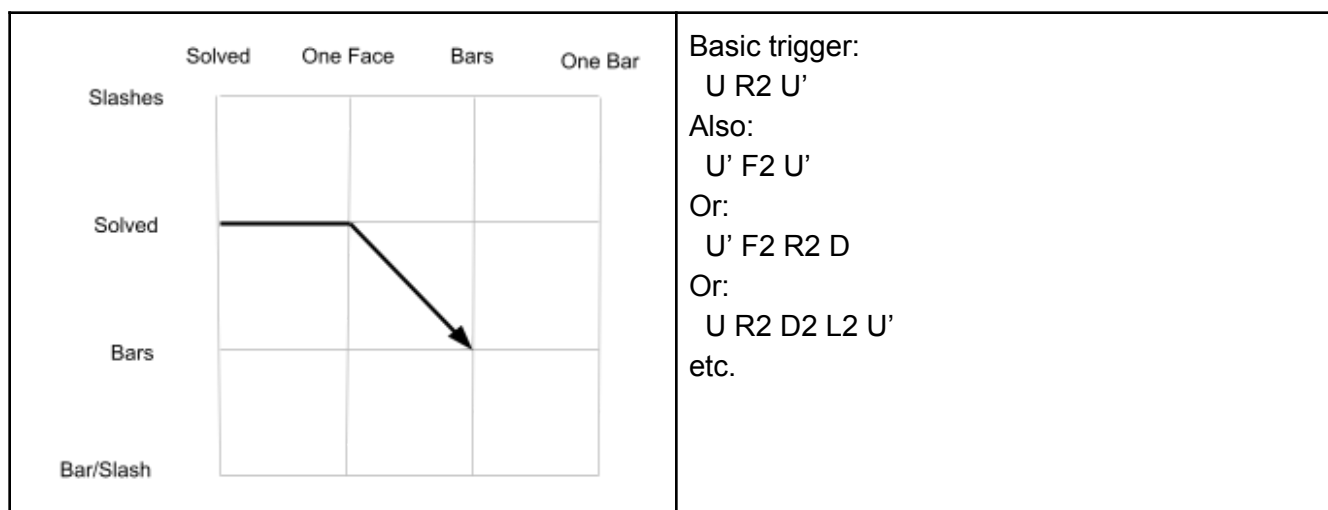
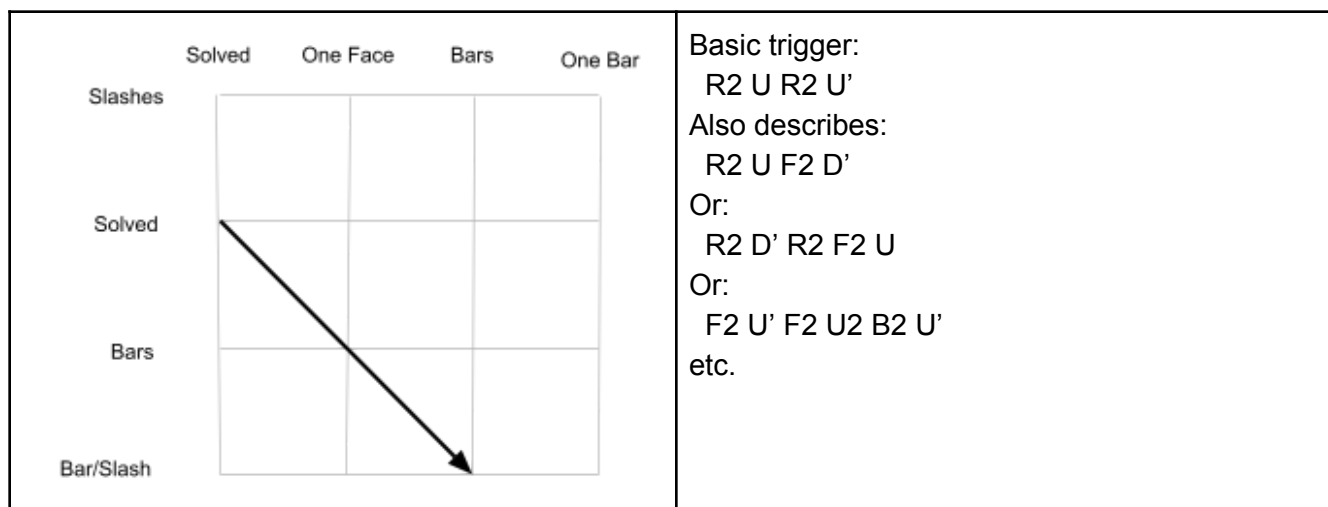
	Solved	One Face	Bars	One Bar
Solved				
One Face				
Bars				
One Bar				

Possible transitions of the FB state following a quarter turn,

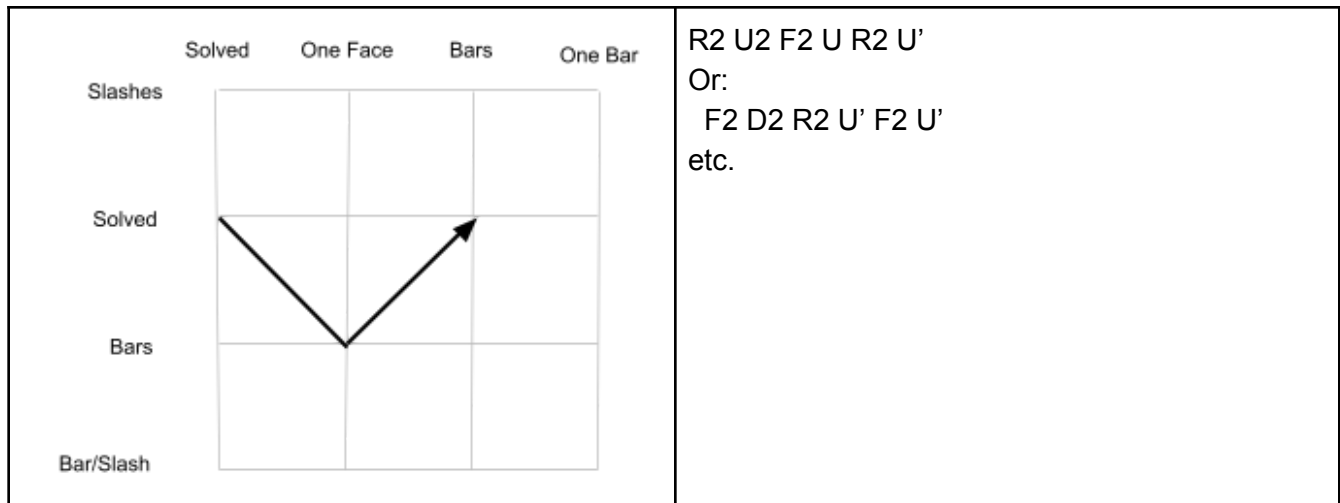
depending on the configuration of the U face.

Visualizing Corner Solutions

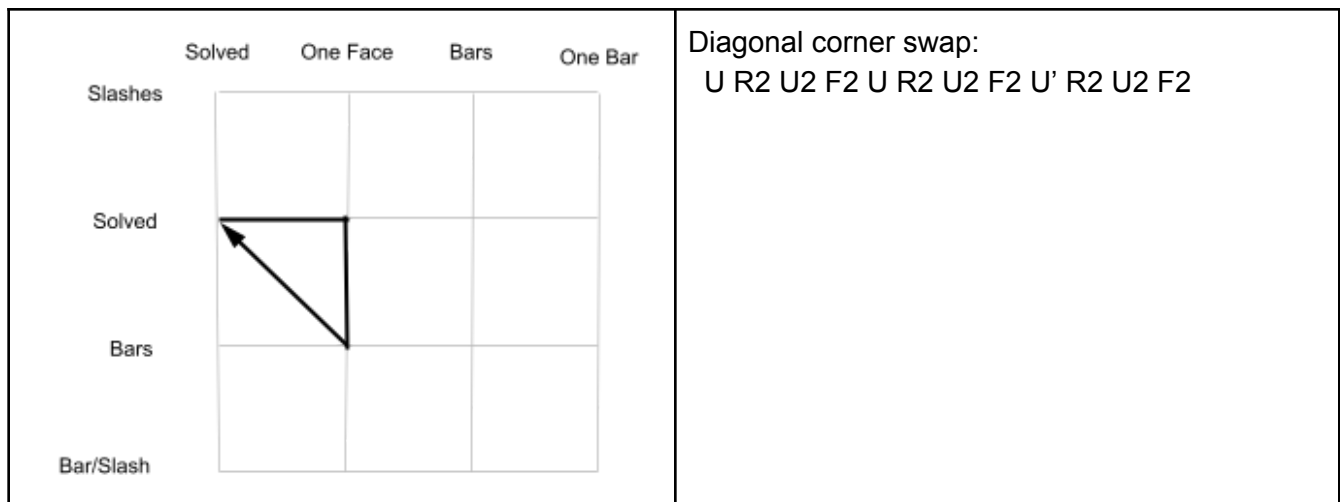
Although it may seem like the number of DR corner configurations is large, we have reduced them to just 32 cases: 2 parity states X 4 UD states X 4 FB states. We can visualize the state of the cube corners by plotting points on a 4x4 grid, with the UD state on the vertical axis and the FB state on the horizontal axis. Half turns do not change our location on the grid, and every quarter turn moves us to an adjacent location (and swaps parity). Let's visualize some familiar algorithms (starting from a solved cube).



Each transition from one point of the grid to another corresponds to a U/U'/D/D' quarter turn. In between, any number of half turns moves may be inserted. The half turns change the U face configuration, and therefore affect where the next transition will end up, but given a particular U face configuration, all quarter turns result in the same transition. The variants in these diagrams all affect the corners in the same way, but affect the edges differently.



Parity is determined by whether the number of transitions is even or odd. Returning to Solved/Solved will not solve the corners if done in an odd number of transitions.



Here are some other familiar algorithms:

	Solved	One Face	Bars	One Bar
Slashes				
Solved				
Bars				
Bar/Slash				

Standard JJ perm:

$$R2\ U'\ R2\ U\ D\ R2\ D'\ R2$$

Alternative (2c2c2e2e3e):

$$R2\ U'\ R2\ U2\ F2\ U'\ R2$$

	Solved	One Face	Bars	One Bar
Slashes				
Solved				
Bars				
Bar/Slash				

Standard J perm:

$$R2\ D\ B2\ D'\ B2\ U\ B2\ U'\ B2\ R2\ U'$$

Alternative (2c4e):

$$F2\ U\ R2\ U'\ L2\ U\ R2\ U'\ L2\ F2\ U'$$

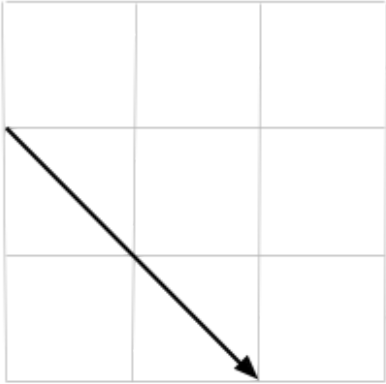
	Solved	One Face	Bars	One Bar
Slashes				
Solved				
Bars				
Bar/Slash				

UBL-UFR-DFL corner commutator:

$$U'\ L2\ D\ L2\ D'\ L2\ U2\ L2\ D\ L2\ D'\ L2\ U'$$

Here are some [VOP](#) algorithms:

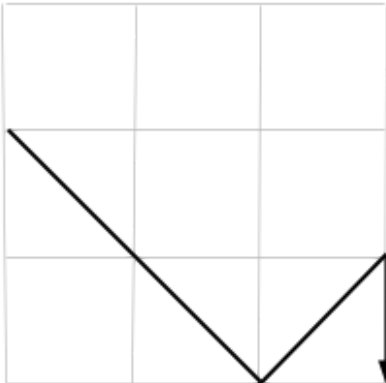
	Solved	One Face	Bars	One Bar
Slashes				
Solved				
Bars				
Bar/Slash				



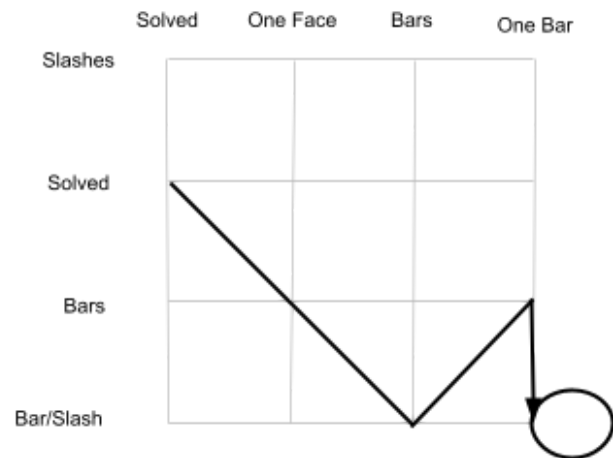
Opposite/Adjacent:
 $R2\ U\ R2\ U'\ R2\ U2$
 Or:
 $R2\ U\ R2\ U\ R2\ U2\ R2$

Adjacent/Opposite:
 $F2\ U'\ F2\ U'\ F2\ U2\ F2$
 Or:
 $F2\ U'\ F2\ U\ F2\ U2$

	Solved	One Face	Bars	One Bar
Slashes				
Solved				
Bars				
Bar/Slash				





Adjacent/Same:
 $F2\ U\ R2\ U'\ F2\ U\ R2\ U'$
 Same/Adjacent:
 $R2\ U'\ F2\ U\ F2\ U'\ F2\ U$




Same/Same:

$R^2 U' R^2 U R^2 D' R^2 U R^2 U' R^2$

The square grid is a simplified representation, because the UD and FB states cannot be combined arbitrarily. The actual number of different states is 34, not 32. Some combinations are not achievable without disassembling the cube. In 3 cases, states can be combined in multiple ways that are not HTR equivalent. Nevertheless, the grid is an intuitive way of visualizing the state of the cube and how it changes. The table below shows the state combinations that have multiple configurations, and how to recognize them.

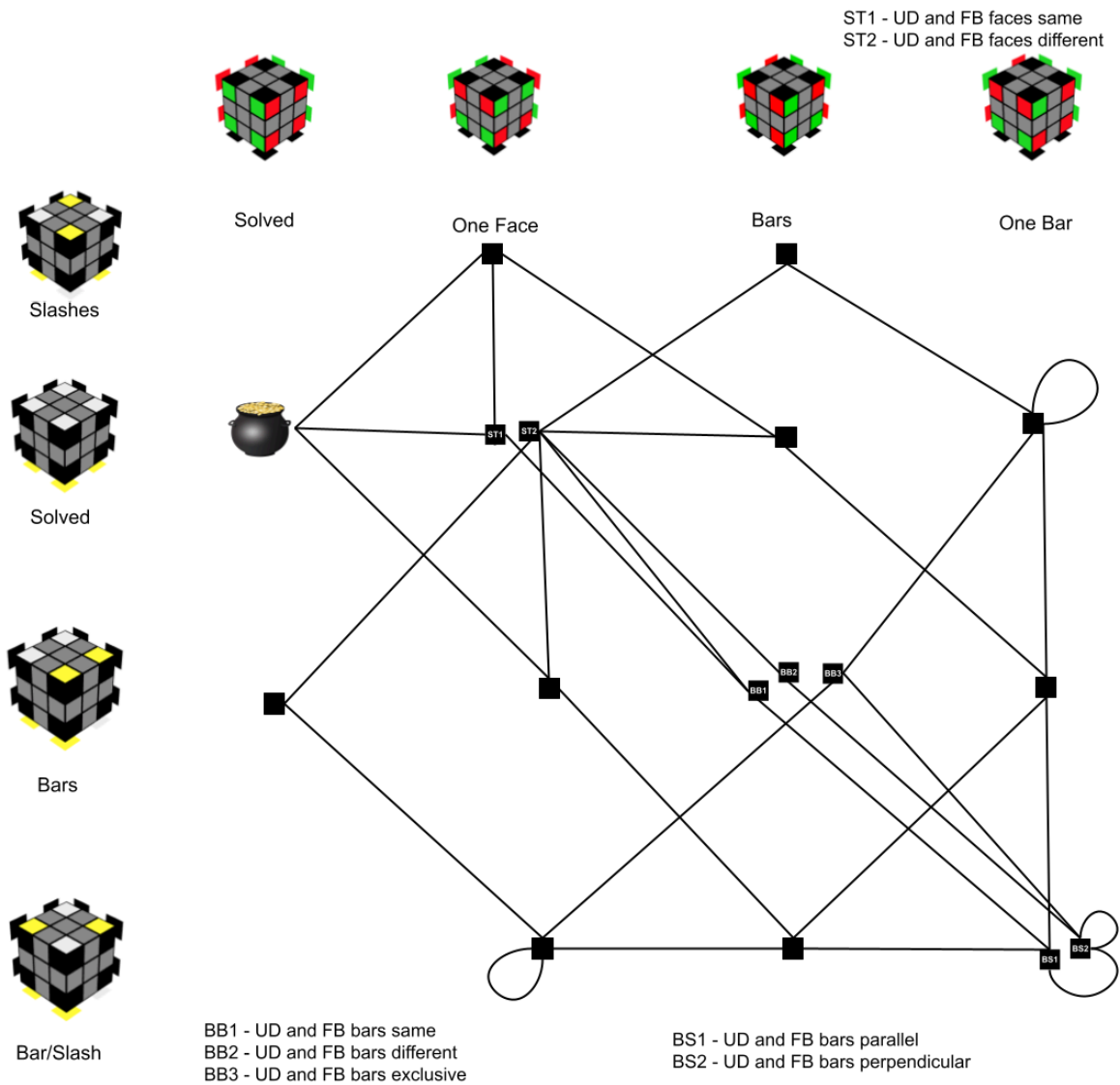
			Generator
Solved + One Face	 ST1	UD and FB faces same <i>U solves both U/D and F/B</i>	U
	 ST2	UD and FB faces different <i>U solves either U/D or F/B but not both</i>	R2 U R2 U2 F2 U F2
Bars + Bars	 BB1	UD and FB bars same <i>U R2 U solves both U/D and F/B</i>	U R2 U
	 BB2	UD and FB bars different <i>U R2 U solves either U/D or F/B but not both</i>	R2 U R2 U2 F2 U R2 U
	 BB3	UD and FB bars exclusive <i>Cannot form UD and FB bars at the same time</i>	R2 U R2 U R2 U2 F2 U' R2 U
Bar/Slash + One Bar	 BS1	UD and FB bars parallel <i>When formed, UD and FB bars point in the same direction</i>	U R2 U R2 U

	 BS2	UD and FB bars perpendicular <i>When formed, the UD and FB bars are perpendicular</i>	R2 U R2 U R2 U' B2 U
--	--	---	----------------------

The Hyper-Parity Maze

Finally, we have a procedure for solving the corners from an arbitrary DR configuration. Analyze the position to determine UD orientation, FB orientation, and Parity. Find the corresponding point in the grid, and plot a path to Solved/Solved with an even number of transitions if Parity is even, or an odd number of transitions if Parity is odd.

Below is the “Hyper-Parity Maze”, which shows all possible DR corner states and the allowed transitions between them:



You may prefer to memorize this simplified version. It shows only the cases that you're likely to want to try in an actual FMC attempt: all of the 3 quarter-turn cases and the 4 quarter-turn cases that can be potentially simplified.



Slashes



Solved



Bars



Bar/Slash



Solved



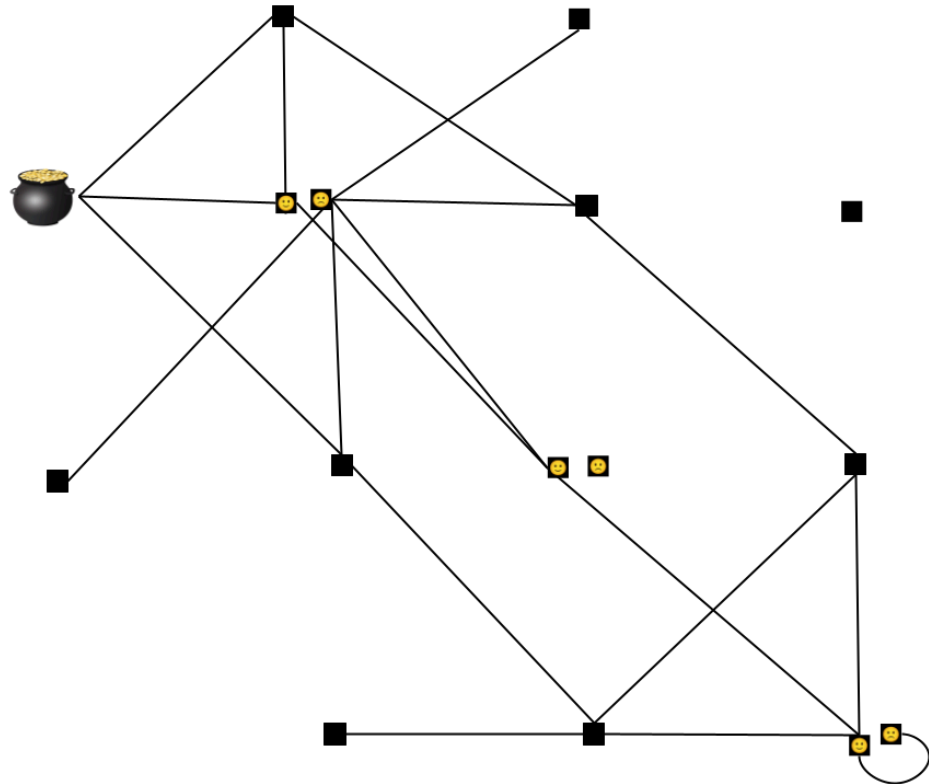
One Face



Bars



One Bar

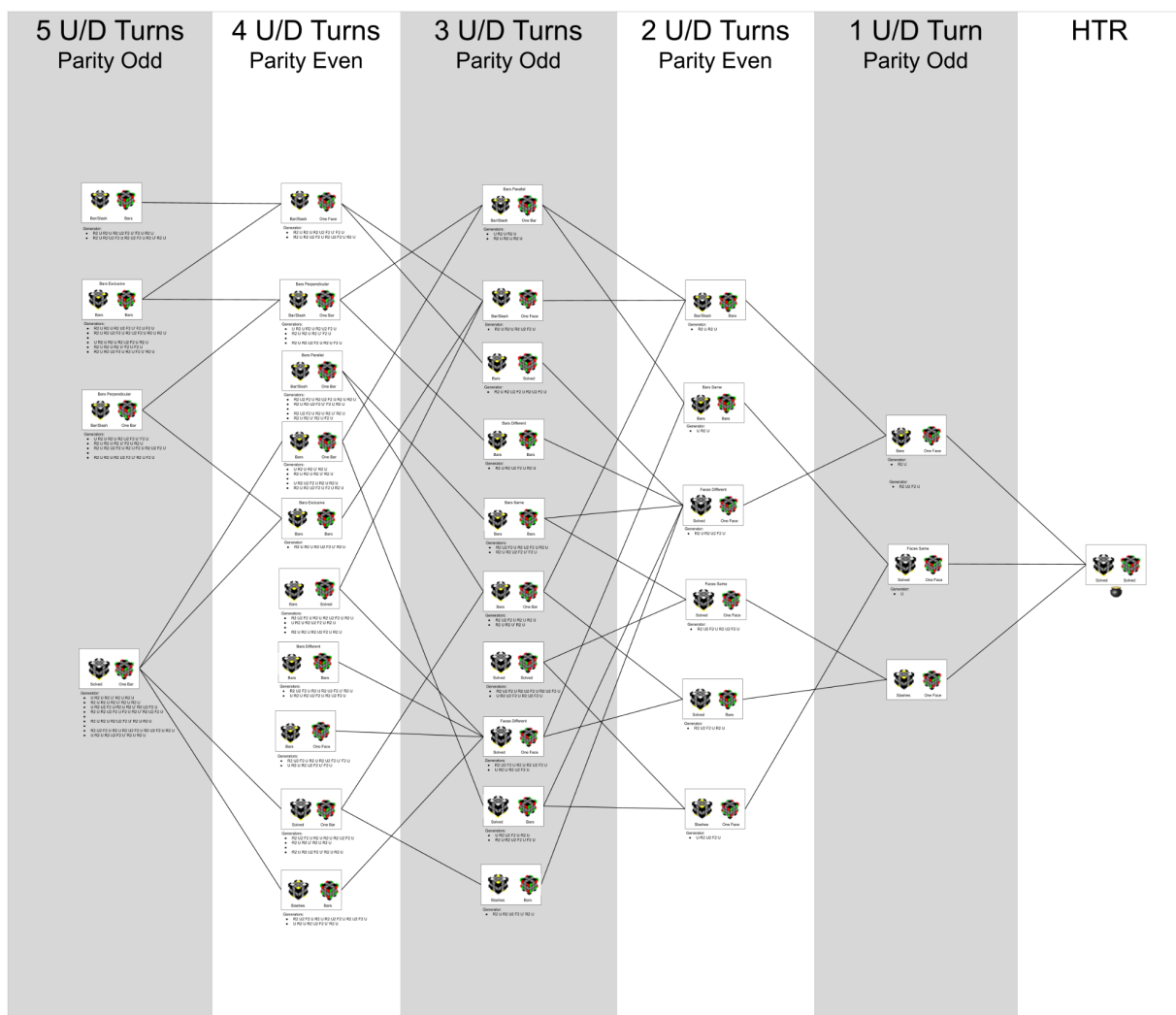


☺: On the U R2 U R2 U happy path
 ☹: Not on the happy path

The Hyper-Parity Funnel

Another way of visualizing DR corner state transitions is shown below. The funnel groups all DR corner states by how many quarter turns are needed to get to HR, showing all possible paths to get from any starting position to HTR.

The funnel gives generators for all 34 corner cases. They are listed in the [DR Corner Case Generators](#) spreadsheet. The generators are the basis of the [Domino Reduction Corner Case Catalog](#), a flow-chart that gives algorithms for solving any DR corner case.




Using Hyper-Parity in FMC

After reaching DR, the generator move count tells us on inspection what the minimum number of moves will be to solve corners. Useful knowledge! The actual solution will require 0-3 HTR setup moves to reach the state at which you can apply the inverse generator. Widening moves gives you an opportunity to solve edges, and you can try different paths through the maze to see if they lead to better edge insertions.

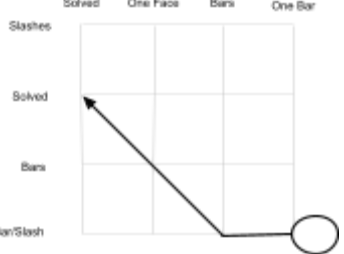
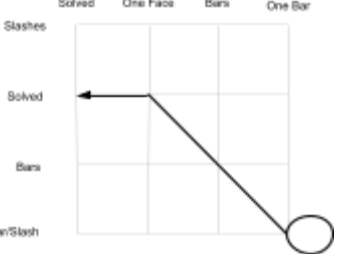
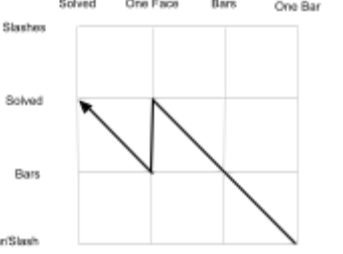
Example Solve

Consider this scramble:

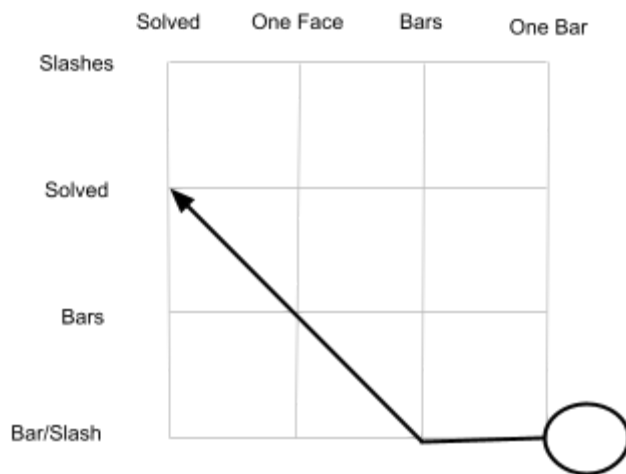
	<p>Scramble:</p> <p>U' R2 L2 F2 B2 R2 F2 B2 U' B2 R2 L2 U' B2 R2 F2 U2 R2 L2 F2 U2 B2 L2 U F2 B2</p> <p>Solution:</p> <p>D B2 R2 U2 B2 R2 B2 D' R2 D U2 L2 D' B2 (14)</p>
--	---

This case has Parity=Even, UD=Bar/Slash, FB=One Bar, with UD and FB bars perpendicular. Most people would not consider this an easy corner case, but we can use the funnel to find corner solutions and then see which yield good edges.

Start by searching the funnel for the Bar/Slash+One Bar (Perpendicular) state in a column with Parity=Even. We find it in the “4 U/D Turns” column. There are 3 different ways to solve corners, corresponding to these paths through the maze:

		
<p>U F2 U R2 U' B2 U' B2 // +8 to 2e2e + E-slice</p>	<p>U L2 U2 B2 U L2 U' F2 U' B2 L2 B2 // +12 to 5e2e + E-slice</p>	<p>L2 U F2 U F2 U L2 U2 B2 U' D2 L2 B2 // + 13 to 4e + E-slice</p>

Of these, the first option is obviously the most promising. Let's look at it in more detail.



1. UD and FB bars Perpendicular → Parallel
2. → Bar/Slash + Bars
3. → Bars + One Face
4. → Solved

Here is a breakdown of the corner solution. At each step, we are checking the state-transition diagrams for UD and FB orientation. We need to execute HTR setup moves to bring the right stickers to the U face so that a U/U' turn will bring us to the next desired state

Move	Explanation
U	Neither UD nor FB orientation change UD/FB bars change from parallel to perpendicular
F2	Setup: FB orientation will change, UD will not
U	→ Bar/Slash + Bars
R2	Setup: Both UD and FB orientation will change
U'	→ Bars + Face
B2	Setup: Both UD and FB orientation will change
U' B2	2e2e + E slice

For each of the setup moves, we had a choice between a move and its wide counterpart (equivalent to inserting a slice). We chose the option that formed or preserved the most pairs. We solve the E layer with slice insertions and are left with an edge commutator to arrive at a 14-move solution.

D // Bar/Slash + Bars, perpendicular
R2 (*) D' // Bar/Slash + Bars
U2 R2 D // Bars + Top Face
U2 L2 D' B2 // 2e2e (10)

(*) - R2 B2 R2 U2 B2 R2 B2 U2 (8-4)

Discussion

Getting the cube to a Domino Reduction state is relatively well understood, and explained in an excellent [guide](#). After reaching DR, most FMC-ers only consider DRs with easy corner cases. However, using the computer to find optimal DR solutions, I was intrigued to see that almost all cases had 12-14-move solutions, whether the corners were easy or not. At the same time, I was frustrated trying to go from DR to HTR and finding myself in a state that looked like HTR but wasn't. Those two mysteries sent me down a long rabbit hole that lead to this document.

34 cases is a very manageable number in cubing, but memorizing the generators is not the right approach for FMC. Many cases have different paths through the maze of the same or almost-same length that are not listed in the spreadsheet, and there are many ways of executing a path through the maze that have different edge outcomes. Instead, I've internalized the geometry of the Hyper-Parity maze to find corner solutions quickly. With that as a skeleton, I'll often find a DR finish that is within two moves of optimal in a couple of minutes. Depending on how many DRs I find, and their move count, I'll check every DR that has a corner solution requiring ≤ 3 or ≤ 4 hops. Happy solving!