Fewest Moves Tutorial

Solving a Rubik's Cube with as few moves as possible

by Sebastiano Tronto

Table of Contents

0. Intro

Trying to solve a Rubik's cube (or, in general, any puzzle) as fast as possible is interesting, but it is even more interesting trying to solve it using fewer moves: this the goal in “Fewest Moves” solving (or FMC, “Fewest Moves Challenge”).

Official Competitions

FMC is an official event recognized by the WCA (World Cube Association), the organization that governs competitions for the Rubik's cube and similar puzzles. Official regulations (article E) can be summed up as follows:

- The scramble (sequence of moves to scramble the cube) is given to all competitors, written on a sheet of paper.
- Time limit: 60 minutes
- Allowed material:
  - Paper and pens (provided by the judge)
  - Up to 3 Rubik's cubes (self-supplied)
  - Unlimited colored stickers (self-supplied)
- The solution
  - Has to be submitted written in the OBTM notation: allowed moves are rotations (x, y2, [u]...), single-layer moves (R, U2, L', …) and wide moves (Rw2, Fw', …) but not inner-layer moves (M, E, S, …); for the final results, rotations are not counted, while each other move counts as 1.
  - Has to be at most 80 moves long (including rotations)
  - Must not be related to the scramble in any way; in addition, the competitor must be able to explain each move in his solution.

The best result ever achieved in competition is 20 moves, by Tomoaki Okayama (Japan) and the world record for an average of 3 attempts (in one competition) is 25, by both Sébastien Auroux (Germany) and Vincent Sheu (USA).

Unofficial Competitions and Variations

Unofficial online competitions are held weekly in many forums (one of which is speedsolving.com) and on http://www.ocf.berkeley.edu/~dadams/fmc. Obviously, nobody can makes sure that the regulations are followed, therefore this kind of competitions are bases on reciprocal trust between competitors.

The most common variants are the “no time limit”, not having any time limit (generally, a time limit of one week is for practical reasons considered “no time limit”) and the “linear”, in which any move performed cannot be undone or canceled; you cannot explore more solutions, you have to write the

first one you find (usually the time limit is set to 10 minutes).

Goal of This Tutorial

The goal of this tutorial is to summarize the best known techniques that lead to good results in fewest moves solving. In some cases the explanation will detailed and enriched with examples, while in some other I will only provide a synthetic explanation and suggest other resources for further study.

Sources and Thanks

Everything you will read in this tutorial comes from the author's own experience, which has been obtained not only through practice, but also thanks to the many resources freely available online, most of which you will find in part 7 “Other Resources”, to which I will refer during all the tutorial. Therefore I want to thank the whole international speedcubing community for always openly spread techniques and methods, enabling anyone to freely learn anything that has been discovered (until now). I hope this tutorial will be helpful in the same way.

Disclaimer

As you may know, English is not my first language; if you think that a certain passage is poorly written or if you find any mistake, please contact me at:

sebastiano.tronto [at] gmail [dot] com

1. Think Outside the Box

Whatever your main method is, restricting yourself by forcing yourself to only use that method is the worst thing you can do in FMC. **You should never restrict yourself** to only one method, but try to exploit every situation.

For example, suppose you have built a 2x2x3 block; now you have many possibilities: you can place the last cross edge and finish the F2L (CFOP), you can orient edges (Petrus) or try to build more blocks freely (FreeFOP, Heise) and so on. Any of this methods may lead to a good solution, so the best thing to do is to **try to use them all** (or most of them at least).

The best way to open your mind and learn how to think outside the box is, maybe a little paradoxically, **to get many "boxes"**, that is to learn many methods. Here I will briefly describe those that are, in my opinion, the most useful methods for FMC, without talking about their development history nor their pro/cons in speedsolving. For each of these methods I will provide a link to an online tutorial, but I suggest you look for more information about them on google or speedsolving.com. This tutorial, although mainly thought for speedsolving, is a good starting point for getting more information about the 4 most commonly used methods (CFOP, Roux, ZZ and Petrus).

Petrus

Petrus' steps are the following:

1. Build a 2x2x2 block
2. Expand the 2x2x2 block to a 2x2x3 block
3. Orient Edges
4. Build the first 2 layers (F2L)
5. Solve the last layer (originally divided into 3 steps)

Having a good **blockbuilding**[1] skill is mandatory if you want to get good at FMC, and practicing Petrus is the best way to acquire it. To learn how to solve steps 1 and 2 efficiently, you need to think and try to explore different ways of building blocks every time; it is also very useful to study **reconstructions of expert cubers' solves**. For the first step it is also helpful to compare your solutions with the optimal ones, given by an optimal solver[2], since for an expert it should be (almost) always possible to find an optimal 2x2x2 block.

Step 3 teaches you how to recognize an edge's orientation regardless its position in the cube, another important skill, since flipped edges are one of the worst thing you may come across during an FMC solve. For this step, as well as for step 4, ZZ may be a better teacher than Petrus.

You don't have to learn every algorithm for solving the last layer (the same for other methods too):

1 "Blockbuilding" or "block-building" is the technique consisting of making blocks of pieces and then joining them together. It is usually opposed to the way of building the F2L used in CFOP (Fridrich), but that can be considered a kind a blockbuilding too. A more suitable contrast is given by comparing it to other techniques, such as edges orientation (Petrus, ZZ), "Corners First" solving, using algorithms or commutators.

2 For example http://laire.fi/jarcs/, which at the moment is not online.

in the next chapter I will explain in detail how you should proceed, for now it is enough to be aware that the “last layer” step usually is not included in Fewest Moves solutions.

Lars Petrus website

Roux

1. Build a 3x2x1 block
2. Build another 3x2x1 block, opposite to the first
3. CMLL (corners of last layer, without necessarily preserving the M layer)
4. LSE (last six edges, to solve only with M and U layers moves)

Petrus is an excellent method to learn blockbuilding, but forcing yourself to only use Petrus is still wrong: learning also Roux (especially the first 2 steps, even in this case) will make it in some way complete it. Also for this method it will be useful to study “the Masters'” solves (at the moment, the absolutely best is Alexander Lau, second in the world for 3x3 speedsolving average and European Champion 2014).

For step 3 it still stands what was said about last layer algorithms, while step 4 is to be avoided like plague (at least, avoid solving it in the “standard” way): remember that every inner layer move, like M, is worth 2 normal moves!

Waffle's Roux tutorial

ZZ

1. EOLine (orienting all edges and placing DF and DB, leaving the cube to be possibly solved moving only the R, L and U layers)
2. F2L
3. Last Layer

As mentioned earlier, recognizing and solving edges' orientation is a very useful skill and ZZ is surely the best way to learn it. At first sight “orienting edges” can be hard, because it is a more abstract concept than “building blocks”, but don't panic, it's not impossible!

Step 2 is more or less the same as Petrus' step 4, but you have to build the blocks on both R and L side at the same time[3]. This is also going to improve your blockbuilding experience.

Conrad Rider's tutorial

CFOP (Fridrich) / FreeFOP

In classic CFOP you follow these steps:

1. Cross (4 edges on the same side)
2. 4 corner/edge pairs insertion

3 As for execution speed, it may be better to build one block first and then the other, but not for FMC.

3. OLL (Orient Last Layer)
4. PLL (Permute Last Layer)

Classic CFOP is **not** considered a good method for FMC, but in some situation it is useful to know different ways to insert a corner/edge pair.

In FreeFOP the first two steps are replaced by a “free” blockbuilding F2L.

Anyways, I strongly advise against solving the last layer with OLL + PLL, unless you get to skip one of the two steps.

Badmephisto's Tutorial

Keyhole F2L

Keyhole is not really a method to solve the cube, but a technique to solve the first two layers. It is considerate an intermediate method between “Layer by Layer” and CFOP. The steps are the following:

1. Cross
2. Place 3 first layer corners
3. Insert 3 middle layer edges, using the “free” corner
4. Insert the last corner/edge pair, as in CFOP or as in LBL

To improve efficiency you should replace the first two steps with blockbuilding and place a few middle layer edges in the meantime. A variation consists in solving the cross and 3 middle layer edges, and then placing 3 first layer corners using the “free” edge.

Despite its simplicity, this method can be very useful in FMC.

Heise

1. Build 4 “squares” 2x2x1 (sharing one colour)
2. Match the squares and orient edges
3. Place 2 corners and the remaining 5 edges
4. Last 3 corners (commutators)

If you decided not to follow the experts' advice and use only one method for FMC, this method must be Heise. Alone it can lead to and average of fewer than 40 moves for linear[4] solves. It is an extreme method, but also extremely efficient.

First two steps are a strange but useful way of building an F2L-1[5] with all edges oriented. The “don't restrict yourself” and “exploit different situations” concepts is perfectly applied, given the freedom provided for solving blocks.

The third step is complex, but it is a more efficient alternative to finish the first two layers and then

4 AS in “linear” FMC, that is without trying different possibilities and without canceling or undoing moves.
5 With F2L-1 I mean an F2L (First 2 Layers) minus a corner/edge pair.

solving the last layer using algorithms. Practicing it will give you the ability to build and move around blocks when you are limited by the quantity of blocks already built (and this is the reason why this step is so difficult).

For the last step you will need commutators; it allows, in an FMC solve, to use insertions (both these techniques will be explained in the next chapter).

Ryan Heise's websites; here you can find not only a detailed explanation of his method, but also other useful information (for example, some fundamental techniques).

What and How to Learn

Obviously, getting fast with all of the methods described is not your goal. Doing some speedsolves may help seeing some things faster and is fun, but **we don't care about speed**. Since our goal is to solve the cube with the fewest moves possible, you should try to be efficient. It is also essential to be **color neutral**[6] and it can be helpful trying to work with "**Non Matching Blocks**"[7].

But the main difference between speedsolving and fewest moves solving is that in FMC you can **try different possibilities**; if in Petrus, for example, you are left with 6 "bad" edges after a 2x2x3, you can try to build a different block or to slightly modify the building of the block you have found to improve your situation[8].

Here is some piece of advice for some of the methods described:

Petrus:

- After completing a 2x2x2 block, you can expand it in **3 different directions**.
- Try to build a 2x2x3 directly, instead of going through the 2x2x2 step.
- Try using Non Matching Blocks in step 4.
- In step 4 again, try influencing the last layer to get an easier case (even "Heise style").

Roux

- Try to build the blocks at the same time.
- Try Non Matching Blocks.
- Influence CMLL while finishing the second block, and the LSE during the CMLL.

ZZ

- After edges orientation, building the "Line" isn't always a good idea: try blockbuilding

6 Being able to solve the cube by starting with any "color"; for example, starting from any cross in CFOP or from any of the 8 possible 2x2x2s in Petrus.

7 Sometimes, especially in FMC, also called "Pseudo Blocks"; it is a useful technique in Roux, ZZ and Heise, but it can be used in other methods as well; it consists in building blocks different than the ones you should build if you are following the method, but that can be placed in the same "slots". For example, in Roux, the second 3x2x1 block can be any of the 4 that can be built on the side opposed to the first one. This technique is very powerful if combined with premoves, that will be explained in chapter 3.

8 Trying to influence a later step while solving another one is a good habit, which will be talked about again later.

directly (without breaking the EO!)

- Once you have oriented the edges, you have reduced the cube to be solved moving only the R, L ,U and D layers: you can build the F2L on any of these sides.
- As in Petrus and Roux, try using Non Matching Blocks and influencing the last layer while building the F2L.

CFOP / FreeFOP

- **FreeFOP is obviously better than CFOP**; try at least to build an **Xcross**[9].
- Try to influence the last layer edges' orientation, avoid the “4 edges flipped” cases; some ZBF2L algorithms can be useful, but instead of learning them by heart try to **study them to understand how they work**.
- Some optimal pair insertion algorithms are not well known (for example **F2 U R U' R' F2**): study them, again trying to understand them rather than learning them.
- Try “**multislotting**”, that is inserting more pairs at the same time. The easiest case is when you use a D-layer move as setup, for example D R U R' D'. You can find some algorithms here, here and here, but I suggest trying in a “free” way: for example, look at how I have inserted the second, third and fourth pair in this solve.

9 Cross and first pair, built at the same time. It can also be seen as a 2x2x2 block + 2 edges.

2. How to Proceed During a Solve

To quote Per Kristen Fredlund, the general way to proceed is:

“Think of it more like so: solve the cube in 2 stages where stage 1 solves as many pieces as possible as efficiently as possible (i.e.: make a good skeleton[10]). The second step is fixing the unsolved pieces, typically by inserting algorithms into the skeleton[11]”[12].

This the general approach, but you don't need to use it every time: sometimes you can find a very short solution by, for example, solving the F2L with blockbuilding and then finishing the last layer with an algorithm. There are also different ways to proceed, two of which will be explained in chapter 4.

If this description seems too generic, it is because it can't be differently: there isn't a standard method that allows you to always get good results, you should **always try as many strategies as you can**, so that you don't miss any chance.

Here I will describe some basic techniques used in FMC solves. You have already learnt some of them by practicing the methods described in the previous chapter, while you will need to study some other. Some will be explained in detail, some other will be only mentioned and other tutorials will be linked for a more complete study.

Blockbuilding

Blockbuilding is probably the most important technique in FMC. It is a simple concept, but it requires a lot of practice to be mastered. Practicing blockbuilding-based methods (Petrus, Roux, Heise and ZZ), in the ways previously described, is the most direct way to improve at blockbuilding.

Here I will list some fundamental techniques that will come in handy; the first ones are taken from Ryan Heise's website, which is full of examples: look for them!

Align Then Join

Basic technique: to build a corner/edge pair, the simplest kind of block, you have to align them first, making them joinable by a single move, and then join them with that move.

This concept can be applied, in a more general meaning, to the case of building bigger blocks, for example when you want to match a pair and 2x2x1 square to get a 3x2x1

All of this may look trivial, and in some way it is; the important thing to learn is to **recognize when two pieces are aligned** and can be therefore joined with one move. This way you will be able to realize in advance whether a certain move will join 2 pieces.

Example:

---

10 A partial solution, where only a few pieces (typically 2 to 6) are left to be solved.

11 Techniques that allows to solve a few pieces by inserting moves somewhere earlier in the solve. It will be soon explained, be patient!

12 http://www.speedsolving.com/forum/showthread.php?1566-Fewest-Moves-Tips-and-Techniques&p=16209&viewfull=1#post16209

Scramble[13]: F U' D2 L D L2 D F R2 B U2 R2 D L2 D L2 D R2 U' L2 F2

Two pairs are already built. If you want to build the one formed by the blue-red edge and the blue-red-yellow corner you can do it this way:

Align: L2
Join: U2

Move it out of the Way

It can happen that we want to build a block, but the direct move to build it breaks other blocks, or moves aways pieces that we don't want to move. One of the ways to bypass this problem is to move aways such pieces first, saving them from the breaking move, and then, if necessary, putting them back together once the move has been performed.

Example:
Scramble: F R2 B D2 F D2 L2 B2 F' D2 F' L B U' F2 U2 L2 U B R2 F

After U2 R2 D R2 we have a 2x2x1 square that can be expanded to a 2x2x2 with F' D'; but F' breaks the red-white edge and blue-red-white pair, which we would like to save. We can do U' (move it out of the way) F' D' (solve) to save the pair.

Destroy and Restore

Another approach to solve this problem is to temporarily break some pieces, to join them afterwards by using the "move it out of the way" technique.

Example:
Scramble R U R' U'

Let's ignore the last layer and pretend not to know that U R U' R' obviously solves the cube. Looking at the first two layers only, we see that R' places the ready pair next to the other pieces in F, but breaks part of the F2L already built. We can use "Destroy and Restore" this way:

R' (Destroy)
F (Move it out of the way)
R (Restore)
F' (Puts back the pieces moved away with F)

Keyhole

We talked about it earlier as a standalone method, but keyhole can be considered a particular strategy to build blocks. This technique consists in exploiting unsolved parts of the cube to solve some pieces. After some good Keyhole F2L practice you shouldn't need any example to understand what I mean, but I put here a Keyhole solve anyways:

Example[14]:

---

13 From now on, when a scramble is given, in explaining the solution (complete or partial) I may mention the color of the pieces, When this happens, I will assume you are a cube with the "standard" color scheme (BOY) and that the scramble is applied with white in U and green in F.

14 This is a slightly modified version of this solve by Edoardo Disarò adapted so that it doesn't require the knowledge of techniques that will be explained only later.

Scramble: U' R' L F' B U2 R2 B2 L' B R D F2 D2 L2 F2 D' R2 F2
Solution:

F' L' *//Face -1 corner*
F2 L' B' L *//Keyhole*
F U' B U *//Keyhole, but it actually inserts also the last corner*
F' R' B2 R *//Keyhole*
F B L B L' *//F2L*
B' *//LL*

One Move, Two Goals

It is often possible to use only one move to build two blocks or, in general, to "get two things done". An example will make this clearer.

Example:
Scramble: D U' F2 U' R' F R2 B D' B R F B' U R' D2 L' R2 F2 B' U' B D B2 F2 U L F U' B2

Taken from this solve by Mirek Goljan and Guus Razoux Schultz (found independently by both, not in cooperation). It is possible to build a 2x2x2 block in 4 moves: L U' **F2** D'; that F2 is what we are talking about: notice how that single move builds the 2x2x1 block in DF and places the orange-green edge at the same time, which allows to build the 2x2x2 with the next move.

Situations like this may arise without forcing them, but it useful to learn to recognize them, in case the don't.

Influence Later Steps

We have already (quickly) talked about influencing the LL step while finishing the F2L[15]. This idea also applies to blockbuilding: it is often better to build a block sub-optimally[16], or to add some unnecessary moves to have a better continuation, blockbuilding or else (i.e., edges orientation).

Example:
Scramble: L2 D2 U R2 F2 D2 B2 U' R2 B2 U B U F D B2 U L D' R' F

You can easily build a 2x2x1 square with L2 R B, but adding only one move the squares become 2: L2 **B** R B.

Pay Attention to EO[17]!

Someone may have noticed, while studying different methods, that "Edges Orientation" is a recurring step. As said before, **the more bad edges there are, the harder things will be**. Orienting edges at the end is usually not efficient. Orienting them first, as in ZZ, is convenient, but it can limit the blockbuilding phase. The best thing to do, according to many (including myself) is trying to solve, at least partially, edges orientation **during** the blockbuilding step. Experience with methods such as ZZ and Petrus can lead to being able to easily spot if an edge will be correctly oriented after

15 Notable examples are ZBLS (sometimes less properly called ZBF2L) and Winter Variation, but there are many more: look them up!
16 "Sub-optimal" (with or without the -) refers to a solution (complete or partial) that uses more moves than the best possible one.
17 Edges Orientation

some moves; if you don't have this ability yet, remember that in FMC solves you can go back and modify your solution every time you wish: if you are having troubles with EO, try going back and modifying some move to see if things get better.

Which Blocks Should I Build?

The golden rule is to exploit different situations: a 2x2x2 block, a 3x2x1, two 2x2x1 squares and many other kind of blocks can be a good start. Try out every possibility.

Two possible approaches are:

1. Try completing big blocks, like a 2x2x3 or a F2L-1.
2. Proceeding with small steps, building many small blocks and joining them at the end.

Erik Jernqvist proposes the following number of moves as good starts[18]:

2x2x1 square + corner/edge pair: 3
2x2x2 block: 4
Two 2x2x1 squares: 5
2x2x3 block: 9
F2L-1: 14
F2L: 17

Personally, I think these are good estimations, especially for the first 3 cases, but always have to remember that **how good a start is depends on the possible continuations**. If you build an F2L-1 in 12 moves, but you have 4 bad edges, one of which is the last F2L edge (flipped in place), you may have to throw it away. On the other hand, a 2x2x3 in 12 moves with all edges oriented can be a better alternative. Obviously EO is not the only thing you have to consider, but it's one of the most important.

Another rule is: **never[19] complete the F2L without influencing the last layer**. The reason is simple: a bad last layer can require many moves, no matter how many algorithms you know, and a completely built F2L gives a little freedom to manipulate the cube; on the contrary, **an F2L-1 is a good partial goal**, because it leaves you more with more freedom, despite the many pieces already placed.

Ready-Made Blocks: How to Deal with Them?

It can happen that you find some blocks already built after scrambling the cube, or that with the first moves you unintentionally build some more blocks (mostly corner/edge pairs). In this cases it is preferable to exploit those blocks rather than giving them and up and going on your own way. How? There are three ways:

1. Expand / match the ready-made blocks first; the obvious thing to do.
2. Expand / match the ready-made blocks, but **watching other pieces too** and trying to build

18 Taken from this post on speedsolving.com. Obviously, a what is “good start” depends on your level; the given move counts are to be considered a good goal for who wants to become an expert, but if you are not this good yet you can get on with longer blocks: don't waste your time looking for a good start.

19 There are, as always, exceptions.

other blocks at the same time; this is usually the best thing to do.

3. Don't expand the ready-made blocks, but try to make new ones, possibly saving the others.

Moreover, it is very important, but terribly difficult, to **understand when it is not worth to save a block**, and you should break it to make new ones. Personally, I find it very hard to “give up” and “throw away” a block, but I realize this is often cause of troubles, especially in time management.

Tricks to Get Fast and Advanced Techniques

“Getting fast” is not to be intended as in speedsolving, but as “finding faster a good blockbuilding start[20]”. Being fast at finding a good start is very important, because it saves time (one hour isn't that long!) and you should try to explore, or at least to notice, every promising start.

The simplest kind of block is the corner/edge pair, or 2x1x1 block. There will probably be one already made in a scramble, without making any move. If there is/are, you can deal with it in one of the ways described in the previous section “Ready-Made Blocks: How to Deal with Them”. If there isn't any, you should learn how to recognize at the sight pairs that can be made with one move (see “Align Then Join”). If you are not fast at recognizing them yet, or if you want to avoid some effort[21], you can use the “Brute Force” techniques: try all possible moves, starting with U, U2, U', then R, R2, R', and so on, checking after each move if you have got to build one pair.

A more advanced technique, more useful but that require a bigger effort, consists in checking every possible 2x2x2 (there are 8 of them). You can do it this way: **for each corner[22], look for its three matching edges** and try to see which way you can join them to make that 2x2x2 block; try not to make any “test” move, so that you can do the same for the other corners without re-scrambling. Usually, if I see a very bad 2x2x2 block(that requires too many moves) I just ignore it and go on. This technique, beside enabling you to find, most of times, an optimal 2x2x2 block (which is usually a good starting point), gives you an idea of where every piece is on the cube.

Another reason why I suggest trying to “see” every move for making the 2x2x2 without performing is that to get faster and better at blockbuilding it is useful to be able to “calculate”[23] pieces' movements without seeing them. Alexander Lau, whom we have already talked about, is able, in the 15 seconds given to inspect the cube before a speedsolve, to plan a whole 3x2x1 block (Roux' first step). His planning is so accurate that while solving this block he can look-ahead[24] and (partially) plan the second block.

You can train your planning ability with this game: ask a friend to scramble your cube with 3 moves and then try to find those 3 moves[25] (it should be easy). Then try with 4 moves, and so on. Depending on your level, you may find it difficult once you reach 6, 7 or 8 moves. If you get without problem to 9 or 10 moves, congratulations!

---

20 A “start” can be a 2x2x2, a 3x2x1, some smaller blocks or, in some cases, a 2x2x3; in general, anything from the first 2 to the first 7 moves.

21 Since you need to keep thinking for an hour, avoid efforts is a good habit, not only if you are lazy.

22 I start from ULB, proceed clockwise, do x2 and repeat.

23 The word “calculate” should be intended as in chess: a player is said to “calculate” 6, 7 or 7 moves if he can think of the possible moves and counter-moves for a total of 6, 7 or 8 moves.

24 In speedsolving, “look-ahead” (with or without the -) is the ability to think about later step while you are solving another one.

25 You can choose between HTM, QTM or STM, but agree on which metric to use first!

Find a Good Skeleton

Once you've got to something good (i.e.: and F2L-1) with blockbuilding, it is hard to go on with the techniques just described. Your goal now is to find a so-called “skeleton”, that is a partial solution that only leaves a few pieces unsolved. The best situation is the one with 3 corners unsolved, where the pieces make a 3-cycle[26], but also 4 or 5 corners, or 3 edges can be good, depending on how many moves you need to build the skeleton.

What should we do then, if the blockbuilding techniques we have seen are difficult to use without destroying what we have just built up? Heise is an excellent starting point. Both the method and the website: step 3 is accurately described, in particular the “Two Pairs Approach”.

Heise's step 3 requires not only an F2L-1, but also all edges to be oriented. If they are not, you can:

1. Orient them now; usually not recommended, unless it takes very few moves.
2. Modify the steps you have already solved, to get all edges oriented at the end.
3. Orient them while finishing your skeleton.

To sum it up, the possible approaches to get a skeleton are many and the best way to train in this case, always a good habit, is to look online (for example on http://fmcsolves.cubing.net) for expert FMCers' solves; most of times, they build a skeleton.

It can also be helpful some deliberate practice: on qqtimer you can choose the scramble type 3x3x3 subsets → last slot + last layer.

You obviously don't have to build an F2L-1 before completing your skeleton, but is often the easiest way[27]. Anyways, try to save small blocks (pairs or 2x2x1 squares) made by LL pieces, if some of them happen to get built.

Commutators

According to speedsolving's definition, in cubing[28], a commutator is a sequence of moves like:

A B A' B'

where A and B are move sequences and X' is the inverse sequence of X[29]. Such commutator is written, in compact notation, as:

[A, B]

Often, in practice, “commutator” actually means “commutator that solves a 3-cycle”. We will use this meaning too.

In contrast with blockbuilding, that solves a lot pieces but heavily influencing the rest of the cube, commutator solve fewer pieces (usually 3) leaving all the others where they were. Therefore, together with blockbuilding and insertions (which we will see in the next paragraph), commutators are basic for a good FMC solve.

---

26 A 3-cycle is a cycle of 3 pieces; for example, A and U perm are 3-cycle, as well as the (L F R F' L' F R' F') OLL.
27 In case you don't, there are many possibilities, requiring ad-hoc solutions.
28 The given definition comes from the mathematical one, given in Group Theory.
29 For example, the inverse of (U R) is (R' U'), not (U' R') nor (R U)!

Corner Commutators

Corner commutators are **the most useful** in FMC. “Niklas” (R U' L' U R' U' L U) A perm (R2 B2 R F R' B2 R F' R) and other algorithms that you probably already know can be seen as commutators.

To learn commutators (which I will not explain in this tutorial), I advise you Brian Yu's tutorial, which is both written and video, and very well made.

For FMC you only need to know “pure” 8 moves commutators. Take a look at A9s and other cases if you want, but as we will see when talking about insertions, you will almost never[30] need them in classic FMC.

Edge Commutators

Once you have learned corner commutators, it should be hard to understand how edge commutators work, especially the one like:

[U R U', M'] = U R U' M' U R' U' M

Sadly, commutators like this aren't usually convenient (although sometimes useful) because they use M layer moves, which are actually two moves in our metric.

The commutator **everyone should know** is:

[M', U2] = M' U2 M U2

Which is also very useful with some setup move, like:

[U: [M', U2]][31] = U M' U2 M U2 U' = U M' U2 M U

Remember that in official competitions you can't write a inner layer moves, so the [M', U2] commutator becomes:

M' U2 M U2 = R' L x U2 R L' x' U2 = R' L F2 R L' U2

Notice how the first two moves (R' L) can be swapped. This is true for every pair of parallel (opposite) moves.

Another thing you need to notice is that the first two moves don't affect any of the 3 edges we want to cycle: R' L F2 R L' U2 is therefore equivalent to L F2 R L' U2 R', to F2 R L' U2 R' L and, since we can invert R' and L, to R' F2 R L' U2 L.

These observations are particularly useful when you want to **cancel** moves, that is making the first moves of our commutator (or, in general, any sequence of moves we want to apply) correspond to the inverse of the preceding moves (or that the last moves correspond to the inverse of the following ones)[32]

There are also edge 3-cycles that don't use inner layer moves. The bet known are:

R2 B2 L2 U B2 R2 F2 D

---

30 You should know that “never” means footnote. Even in this case there are exceptions.

31 The [A: B] notation stays for a *conjugate*, that is A B A'; A is usualle called “setup moves”.

32 For example, if our F2L ends with …U R2 F', and we want to use the algorithm F R U R' U' F', 3 moves cancel: ...U *R2 F F' R* U R' U' F' = U R' U R' U' F'

R2 B2 L2 U L2 B2 R2 D

Notice how the first two moves of the first one don't affect our 3 edges: we can therefore “shift” it, as we have done before to R' L F2 R L' U2.

One last note for edges: even if they are not 3-cycles, it is useful to know the sequences M2 U2 M2 U2 and R2 U2 R2 U2 R2 U2, both swapping two pairs of edges.

Block Commutators

If you have read Ryan Heise's website carefully, you already know something about the so-called “pair 3-cycles”, or block commutators. If you know corner commutators, it will not be hard to find them intuitively. For example:

[L Dw L', U'] = L Dw' L' U' L Dw L' U

They are very useful in Heise's third step, but also to solve a corner 3-cycle and an edge 3-cycle at the same time. For example, the last layer algorithm M F U F' U' F' L F R' can be seen as:

[R: [L' Dw L, U']] = R L' Dw L U' L' Dw' L U R'

That is a pair commutator with a setup move.

J-perm can also be seen as a pair 3-cycle:

Scramble: R U2 R' U' R U2 L' U R' U' L U
Solution: [R2: [Fw2, D B2 D']] = R2 Fw2 D B2 D' Fw2 D B2 D' R2

Insertions

After mentioning them multiple times throughout this tutorial, it is (finally) insertions time.

We have somehow found a good skeleton; let's suppose we need to solve a corners 3-cycle to finish. What do we do now? We can obviously directly solve the cycle with a commutator; but, if you have studied all the cases, you know that a corner commutator can need up to 12 moves. Knowing that the best case only needs 8, those 4 more moves are not really nice. But there is a way to solve a corners 3-cycle almost certainly with *fewer* than 8 moves: insertions.

Simple Insertions

The idea behind insertions is not too difficult: if there are only 3 corners left to solve, I can go through my whole skeleton move by move and solve them when the corresponding commutator only requires 8 moves; since that commutator doesn't affect anything but the pieces that need to be affected, the rest of the solution will solve every other piece, as it did before, and those 3 corners will also be solved, since I have cycled them with the inserted commutator.

This is how to solve almost always a 3-cycle of corner with 8 moves. How can we improve this? Among all possible inserted commutators that solve our 3-cycle, we simply choose the one that **cancels the most moves**; usually, it is enough to just check the cases where our 3 corners can be solved with a “pure” commutator, and then choose the bet one. It happens extremely rarely that the best insertion is given by a 9 move commutator (or longer), but situations like this are so unlikely that it is not worth to check every possible type of commutator.

In order to make it easier to follow the movements of the 3 corners while going through your skeleton, many suggest stickering the cube with white stickers[33] on which writing numbers 1, 2 and 3 (or letters A, B and C if you prefer)[34]. Personally, I suggest taking a cheap cube with non-bright stickers and writing on it with a pencil.

An example solve will make everything clearer:

Scramble: D B2 U' F2 L2 D2 R2 U F2 U2 L2 R' D2 B L' U' R2 F2 R B F2

Skeleton:

B' U' D L' F' //EO + blocks
D2 L2 D' L //Pseudo[35] 2x2x3
U2 R2 U' R' //Pseudo 2x2x1
U L' U R' U' L U2 R' L' //All but 3 corners

At this point, grab a pencil and write “1” on the red sticker of the blue-red-yellow corner, “2” on the orange sticker of the blue-yellow-orange corner and “3” on the white sticker of the orange-blue-white corner.[36] Solve the cube (L B2 L F L' B2 L F' L2) and re-scramble it.

We could solve the three corners at this point, but we would need 9 moves (R2 F R B2 R' F' R B2 R). So lets perform the first move of the skeleton (B') and see if we have a better case: no, still 9 moves. Perform the following move (U')[37]: we can now solve our three corners with an 8 moves commutator (L2 F R F' L2 F R' F')! If we wanted to use it, the final solution to submit would be:

B' U' **L2 F R F' L2 F R' F'** D L' F' D2 L2 D' L U2 R2 U' R' U L' U R' U' L U2 R' L'

Try it out and convince yourself it works.

Anyways, we can do better. Perform the next move (D) and notice that this case requires 9 moves[38]. Go on this way for some more move, until you reach ...L' F' D2. We can again solve our three corners with 8 moves (D' F2 D B2 D' F2 D B2)[39]. But there is more: the last moved performed and the first of our commutator cancel! If we wanted to solve the three corners now, we should write:

B' U' D L' F' *D2* ***D'*** **F2 D B2 D' F2 D B2** L2 D' L U2 R2 U' R' U L' U R' U' L U2 R' L'

Which is equivalent to:

B' U' D L' F' ***D*** **F2 D B2 D' F2 D B2** L2 D' L U2 R2 U' R' U L' U R' U' L U2 R' L'

One move less, yay! So we got to solve 3 corners with 7 moves.

For the sake of completeness, we should continue until the end of the skeleton looking for better insertions. Actually, the best insertion is towards the end:

---

33 This is why you are allowed to bring with you “unlimited colored stickers” at a competition.
34 Sticker need to be attached so that the 3-cycle that moves 1 to 2, 2 to 3 and 3 to 1 is the one that solve the cube.
35 We will soon talk about pseudo blocks.
36 There are many equivalent enumerations: you can start from the corner and from the sticker you wish, you just have to be *coherent*.
37 Watch out: when in a skeleton there are two consecutive parallel layers moves try swapping them too see if this gives better insertions. This is not the case, but you never know.
38 The last 2 moves (U' D) are equivalent to an inner-layer move (E'), so they don't affect corners: it is reasonable that, before and after these moves, our 3-cycle is the same, modulo a rotation (y'/y in this case).
39 Also B2 U' F' U B2 U' F U

B' U' D L' F' D2 L2 D' L U2 R2 U' R' U L' * U R' U' L U2 R' L'

* = L F' L' B L F L' B'

Final solution:

B' U' D L' F' D2 L2 D' L U2 R2 U' R' U *L' L* F' L' B L F L' B' U R' U' L U2 R' L'

Where L and L' obviously cancel, so:

B' U' D L' F' D2 L2 D' L U2 R2 U' R' U F' L' B L F L' B' U R' U' L U2 R' L'

Without further explanation you should understand how to find insertions for edges 3-cycles. You do the same also for double edges 2-cycles, solvable with M2 U2 M2 or R2 U2 R2 U2 R2 U2 or variations[40].

Multiple Insertions: Separated Cycles (3 edges and 3 corners)

A skeleton doesn't have to always leave one 3-cycle: insertions can also be used to solve more (or longer) cycles.

As we have already seen, two 3-cycles, a corners and an edges one, can be solved with a **pairs commutator** (with setup moves, if necessary). Another way is to solve the edges with a **"Sune"**[41] or a variation of it, so that the only corners affected are the ones we need to cycle[42]. Both these ways should be kept in mind, but they are rarely easy to use[43]. The "standard" solution is to **insert two commutators**.

After having numbered both the corners and the edges[44], proceed move by move as you would do with simple insertions, but at every spot you look both for a corners and an edges solution, besides checking for pairs commutators and Sunes. Once you are done, you can write down the final solution with two insertions, but you can also **do another pass**: if you want to keep, for example, the corner commutator, but you want to look for a better edge cycle, you can write down your solution with only the corner commutator inserted; what you have now is a **new skeleton**, a few moves longer, which only leaves 3 edges. At this point you can solve them with one simple insertion. Why should we do this, when we could get anything to work with only one pass? Because a good place to insert the edge commutator can be **inside the other commutator**. You can also insert the edge cycle first and then look for a corner insertion. This solve is a good example, although a little overcomplicated for just understanding this.

You can use exactly this approach for other cases that require you to solve two or more cycles (3 cycles or double 2-cycles) that are separated. In all other cases, things get a bit more complicated.

Multiple Insertions: 2 or 3 Twisted Corners

When you need to twist 2 corners, you can try to insert [F L' D2 L F', U2] (12 HTM) somewhere; for three corners, you can use U' B U' F U2 B2 D' R2 U D F' U' B (13 HTM). But this is usually not the best way, especially in the second case.

40 You can also solve this case with a double insertion, as with corners double 2-cycles.
41 R U R' U R U2 R'
42 Which still need to be solved somehow, possibly with another insertion.
43 Unless you used a lot of setup moves, which will make them longer and inefficient.
44 I prefer using numbers for corners and letters for edges, so that I can't mistake one for another.

The classic way to solve 2 or 3 twisted corners is to use **two inserted corner commutators**. The first one only needs to cycle in any way the three twisted corners (or the two twisted corners + one random corner) and will usually lead to may cancellations. Then you only need to insert a simple corner commutator in the new skeleton you have got.

To do this, you only need to draw an X or any other symbol on the twisted corners; if you want to try to use one of the “pure flip” algorithms written above, I suggest drawing an arrow, so that you can tell which way you need to twist your corner (clockwise or counterclockwise).

After finding the first cycle, erase the symbols you have drawn and number the stickers 1,2 and 3.

You can do the same with two flipped edges, but I suggest avoiding such situations, because edge commutators usually require more moves.

Multiple Insertions: 4 corners

If have 4 corners left, the only bad case, requiring three inserted commutators, is when the corner are all placed but twisted. Try to avoid it.[45]

There are three other cases:

1. One corner is placed but twisted, the other 3 form a “twisted 3-cycle”[46].
2. Two pair of corners that need to be swapped, correctly oriented (double swap or double 2-cycle)[47].
3. A “twisted double swap”.

All these cases **can be solved with two commutators**: the first one has to solve one of the 4 corners, freely affecting[48] the other 3, while the second one, which should be inserted in the new skeleton obtained after the first insertion, has to solve the 3 corners left.

To do so, I mark the corners as follows, depending on the case:

1. I mark the twisted corner with and X (I don't care which way it needs to be twisted), then I number the other three from 1 to 3; at this point, since the cycle is “twisted”, 1 belongs to 2, 2 to 3 while 3 doesn't belong to 1, but to another sticker on that corner[49]. No problem: just mark that sticker with a 4.
2. I mark the first pair of corner with two Xs and the second one with two As.
3. Since we needed 4 numbers for a twisted 3-cycle, for a twisted 2-cycle we will need 3:

45 The discussion started by this post in the FMC thread on speedsolving.com is a good reading. Sébastien Auroux' (more than authoritative) point of view is slightly different. The “4 twisted” is still the worst 4 corners case and, beside sheer move count, you have to take in account that finding three insertions takes longer than finding only 2.

46 A 3-cycle as regards to permutation, without considering orientation. A twisted cycle always depends on some other twisted piece (or cycle).

47 This case can be solved, besides using the method that will soon be described, also by transforming the corner double swap in an edge double swap: remember that an H perm (M2 U M2 U2 M2 U M2) can be transformed in a corner permutation with only one move (U2).
Another way it so use algorithm such as (R U R' U')3 (triple sexy) or (R' F R F')3 (triple sledge), that swap two corner pairs.
These alternative methods can be useful once in a while, but the general method is almost always better.

48 Except for the first case, where you need the placed but twisted corner to be affected by the commutator.

49 Always keep in mind that when talking about commutators there is a difference between “sticker” and “piece”.

reasoning as in the first case, number one of the 2-cycles 1 to 3 and the other A to C.

In the first case, for example, a possible “first cycle” is the one that sends X to 3, 3 to 4 and 4 to X.

The cycle 1 → 2 → 3 → 1 is not good, because it leaves 2 twisted corners instead of a 3-cycle.

Also in this case, I advise against dealing with the similar situation with 4 edges.

An excellent example is this modified version[50] of the South American record by João Pedro Batista Ribeiro Costa:

Scramble: L U2 D' L' U2 B' D B' L U2 F2 R2 F' R2 L2 F' U2 D2 F

Solution:

F D' # U2 * R *//EO*
D' F' L2 *//2x2x2*
F2 D F2 *//Pseudo F2L-2*
B' D B *//F2L-1*
F' D' F' *//AB4C*

* = U R D' R' U' R D R' *//First commutator, 5 moves cancel*
# = L' U' R' U L U' R U *//Second commutator, 3 moves cancel*

Final solution: F D' L' U' R' U L U' R2 D' R' U' R F' L2 F2 D F2 B' D B F' D' F'

Multiple Insertions: 5 Corners

Among all the cases with 5 corners left, the only one that requires 2 commutators is the one where the pieces make a 5-cycle. All other cases require 3 commutators, except when you have 5 corners placed but twisted: in that case you need 4. Here I will ignore any case requiring 3 or more commutators (although in some cases you may want to go for 3 insertions) and only look at the first one.

The easiest and probably the most used way to deal with this situation is a **two-pass** way, as for 4 corners. After having numbered the corners 1 to 5, you go through the skeleton move by move and look for any commutator that solves a three consecutive corners cycle. These cycles are:

1 → 2 → 3 → 1
2 → 3 → 4 → 2
3 → 4 → 5 → 3
4 → 5 → 1 → 4
5 → 1 → 2 → 5

This will obviously take longer than looking for just one corner cycle. It is still enough to just look for “pure” commutators. Every time you find a good commutator write it down somewhere.

Once you are done with this first pass, choose the best commutator you have found (the one canceling the most moves)[51]. Insert that commutator; now you have a new skeleton that only leaves

50 The solution is adapted (and is one move shorter) so that it doesn't require, as the original one does, a premove, a technique that will be explained later.

51 The best choice may not be unique (for example, if you have found two different commutators canceling 3 moves); which should we choose? If you are short on time, just randomly choose any of them. If you have some time left,

a corner 3-cycle: you know what to do.

With this method **you can't be sure** you have found the optimal solution: to be safer, you should check all (or at lest most) of the commutators you have found in the first pass an do many other (probably useless) passes. This usually leads to a huge waste of time and rarely to a better solution.

To know if you don't know if you can improve your solution, keep in mind that 10 or 11 moves total are a good goal for a corner 5-cycle.

There is also a faster, but slightly more complex, way that only requires **one pass**. It is almost identical to the first pass of the two-pass way, but besides taking note of any commutator you have found, you should also write down which corner cycle it solves. At this point, without even touching the cube anymore, you are already able to choose a pair of commutators that will solve the corner 5-cycle.

To understand how, we need some "**cycle theory**"[52]; don't worry, I will cut it down to the essential.

First of all, written with the usual notation, our cycle is:

(1 2 3 4 5)

which means that the corner 1 belongs to 2, 2 to 3, … and 5 to 1. With this notation, this cycle is equivalent to (2 3 4 5 1), (3 4 5 1 2) and so on. What we want to do is to break this 5-cycle into two 3-cycles, for example:

(1 2 3) (3 4 5)

But this decomposition doesn't work. If you want to know why you can read some of the posts I have linked in the footnote; or, if you prefer a more practical approach... try it out! The important thing is that the correct possible decompositions are:

(1 2 3) (4 5 1)
(2 3 4) (5 1 2)
(3 4 5) (1 2 3)
(4 5 1) (2 3 4)
(5 1 2) (3 4 5)

So, **the first number of the first cycle must be the same as the last number of the second cycle**.

Notice that **the order is important**: as you can see, (1 2 3) can be followed by (4 5 1) or preceded by (3 4 5). This means that once you have found a good commutator that solves, for example, (1 2 3), to complete the solve you can choose between a commutator that solves (4 5 1) somewhere *after* it or for a commutator that solves (3 4 5) somewhere *before* it. You can do the same with any of the listed pairs of cycles.

This method, although faster than the other one, doesn't allow you to check for "insertions inside insertions"; therefore, I advise using it only if you are short on time, or as "preliminary analysis":

---

you can think of trying both of them.

52 A further interesting reading is the discussion started by this post in the FMC thread on speedsolving.com. In particular this post, a mathematical proof of the "rule" for finding a 3-cycle combination to solve a 5-cycle; this other one, which is the same explanation I give here (it is where I have learned this technique); and this is an example solve using this method.
Math geeks may be familiar with the topic: it is more or less permutation theory.

try to see how many moves you can get this way, and in the second pass only check for commutators that may lead to something better.

Multiple Insertions: 5 Edges

As usual, even for 5-cycles the edge case is preferably to avoid. If you decided to go that way, you can use the same method described for corners.

But there is one case that can be solve in very few moves: when you can use the **6 moves 5 cycle**:

M' U M U'

even with some setup move, it can be really nice. Try to learn which cases it solves and possibly also some variations, like the “shifted” L F L' R U' R'. If you get a skeleton that leaves an edge 5-cycle, it is a good idea to number the stickers and quickly go through the solve to see if you can insert one of these algorithms. But don't waste to much time looking for it.

Other Insertions: 2 Corners and 2 Edges

Sometimes you may find a skeleton leaving 2 corners and 2 edges in a double swap (i.e.: a PLL such as J, T, V and so on).

In these cases, it is very useful to know a few **10 moves** algorithms:

Fw2 R D R' Fw2 R D' R D R2 (J perm)
Rw' U Rw' U2 R B' R' U2 Rw2 B' (T-perm + corner twist)

There are also many 11 moves[53] ones, some of them are:

R U2 R' U' R U2 L' U R' U' L (J perm)
R2 D B2 D' Fw2 D B2 D' Fw2 R2 U' (J perm)
R2 Uw R2 Uw' R2 F2 Uw' F2 Uw F2 U' (T perm)
R' U R U2 L' R' U R U' L U2 (J perm + corner twist)

Besides all of these algorithms (this is not a complete list anyway), also their inverses and “shifted” versions solve a 2 corners – 2 edges double swap.

Note that **the inverses of these algorithms solve exactly the same case**. Just by noting this you double your chances of cancellations without the need to learn more algorithms.

Don't expect lots of cancellations, but the more algorithms you know, the better.

This solve is a good example, although it uses NISS, which will be explained later.

Other Insertions: Conjugate and Solve

A particular situation you can solve with only one insertion is when you have 4 edges and 4 corners left and both sets make a 4-cycle.

You can solve this case as follows: place all your 8 unsolved pieces on one layer (setup), so that a single move of that layer solves both the 4-cycles, perform that move (solve) and then put the 8 pieces back (anti-setup). The same technique works when the 8 pieces make four 2-cycles in total:

53 Sometimes it's 10 moves + AUF

in this case, the “interchange” will be a 180° one (for example U2).[54]

This solve is a good example. I have found the skeleton and solved the rest with three insertions, while Mirek Goljan suggest a more interesting insertion, using this method:

Scramble: R2 L2 D2 F2 D' R2 U' B2 D' F2 U2 F' D2 L' F U B F2 U2 F2 L
Solution: U2 F B' L2 D2 F' U F2 L' B2 L F2 L' B' L' D L B' D L U L' D' L U2 R F2 R' L2 B
Skeleton: U2 F B' L2 D2 F' * R F2 R' L2 B
Insertion: * = (B D R2 B R' B2 D U2 F') U (F U2 D' B2 R B' R2 D' B')

There are also some LL algorithm that work this way: one is (R B2 R2 U2 R) B (R' U2 R2 B2 R'), and you can find some other here.

Move Count (an Estimate)

Here I will give an estimate of how many moves are usually needed for the most common types of insertions. It is a heuristic estimate, not a mathematical proven one, and keep in mind that these number also depend on:

- How many commutators/algorithms you know and your ability to recognize them.
- The skeleton's length: longer skeletons give more spots where you to insert an algorithm, and so better chance of cancellations (but you shouldn't choose a long skeleton instead of a short one because of this!).

These estimates are useful if you want to know whether it is worth or not to spend some time looking for insertions: if you goal is to achieve a solution shorter than 30 moves, if you have a skeleton leaving 3 corners in 23 moves you will probably get it, if your skeleton is 25 moves long you will need some luck.

You can also compare different kind of skeletons: a skeleton leaving 4 corner in 18 moves is probably better than one leaving 3 corners in 25.

So here are the numbers:[55]

- Corner 3-cycle: 6 moves.
- Edge 3-cycle: 7 moves (very changeable).
- 2 twisted corners, 2 commutators: 8 moves.
- 3 twisted corners, 2 commutators: 9 moves.
- 4 corners, 2 commutators: 10 moves.
- Corner 5-cycle, 2 commutators: 11 moves.
- 2 corners and 2 edges, double swap: 10 moves

54 If the pieces left are not exactly 8, or do not make two 4-cycles, you can use “Reverse NISS”, later explained.
55 Mostly taken from here, slightly adjusted to match my personal opinion.

Insertion Finder

Insertion Finder, developed by Baiqiang Dong, is a useful tool to find insertions and check if you have failed to see something in your solutions: it finds up to 4 insertions to solve a skeleton.

It is especially useful for easy cases (3 corners or 3 edges) but in complex situation it may find solution that are not possible to find for humans: use it responsibly!

Go Back and Pimp your Solve

If you get stuck after a good start, you can try this: go through your solve move by move, looking for a spot where there is at least one layer containing only pieces that you have not solved yet; when you find it, you can move that layer (this is not going to affect any of the blocks you have already built). You have 3 choices (for example U, U2, U') and this way you will get 3 different starts that are just slightly (one move) longer than the previous. One move can be a good price to pay for a better continuation!

Get Lucky!

Luck is not a skill to be learned obviously, but remember that in FMC you have to **go for it**: an "simple" solve ending with an LL skip is not less worthy than a complex unlucky one, if they have the same length. This is one of the reasons why you need to try as many different alternatives as you can: you are more likely to get a skip if you try 100 solutions than if you try 10 or 20.

First Example: Insert Last Pair

After completing an F2L-1, you can finish the F2L by inserting the last pair. This isn't usually a good way to continue, unless you get lucky. To improve your chances to get lucky, try to **insert the last pair in all the ways** you can think of.[56]

Second Example: How to Use Algorithms

First of all, you need to be able to recognize **symmetries** in algorithms: the OLL that is solved by F R U R' U' F' is symmetrical about the S plane, so the same case can be solved by B' R' U' R B U[57]. If you want to use it, try also its symmetrical to double your chances of getting a skip (or a good case).

An extreme example is the OLL solvable with R U2 R' U' R U R' U' R U' R': with this algorithm and its symmetrical L' U2 L U L' U' L U L' U L you can solve the same case from 4 different angles, and from other 4 if you use their inverses.[58]

Secondly, you don't need to use an algorithm for the purpose you have learnt it for. Sticking to F R U R' U' F', you probably know it as an OLL; but you can decide to use it **ignoring corners**: if you complete the F2L and are left with 2 bad edges, you can try this algorithm from 4 different angles to see if you get a skip or at least an easy case (i.e.: a Sune).

---

56 With an already made pair, there are 3 easy ones: R U R', R U2 R' and R' F R F'.
57 This case is also symmetrical for corners permutation.
58 Even in this case corners permutation remains the same, since it isn't affected by the algorithm.

3. Advanced Tools

In the previous chapter we have looked at the basic techniques, needed to find a good solution. In this one more advanced tool will be provided. They are not necessary, but they help getting unstuck and give you more possibilities to explore.

Inverse Scramble

If you can't find any good start, you can try with inverse scramble: if you find a solution for the inverse sequence[59], you just need to invert it to get a solution for the normal scramble. It looks complicated but it is actually very simple.

As an example, here is the former North American record by Tim Reynolds:

Scramble: D2 L2 B R2 U2 F' L2 U2 B2 L2 F' D L2 B U L' U2 L' F' R'
Inverse Scramble: R F L U2 L U' B' L2 D' F L2 B2 U2 L2 F U2 R2 B' L2 D2

Apply inverse scramble and solve with:

R' U F' L2
F2 D' B' * D2 B
R2 F R2 F' R2
F D' F' D

Insert B' U2 B D B' U2 B D' at *, 2 moves cancel.

The solution you have found is:

R' U F' L2 F2 D' B2 U2 B D B' U2 B D B R2 F R2 F' R2 F D' F' D

But this solves the inverse scramble; the solution for the original scramble is:

D' F D F' R2 F R2 F' R2 B' D' B' U2 B D' B' U2 B' D F2 L2 F U' R

It is a common mistake thinking that normal and inverse scramble are complete unrelated. They are actually very similar: for example, if you use ZZ, you will notice that, for any orientation, the two scrambles have the same number of “bad” edges, but in different positions. You will also notice that any block found in one of the two is also in the other one, but sometimes it is somewhere else and made of pieces of different colors. The general rule is this: **if in the normal scramble piece X occupies the place of piece Y, in the inverse scramble piece Y occupies the place of piece X**. Therefore, solved and flipped-in-place pieces will be respectively solved and flipped-in-place[60] in the inverse scramble. “Fixed” blocks will stay the same, while “moving”[61] blocks will be made of different pieces and placed somewhere else.

Besides being a technique useful as it is, in case you get stuck right at the beginning of a solve or simply to get more possibilities to explore, the ideas explained here are fundamental to the techniques introduced in the next paragraph.

---

59 I prefer repeating myself: the inverse sequence of, for example, F R U' is U R' F', not F' R' U nor U' R F!
60 Corner will be twisted the other way round.
61 “Fixed” blocks are those than can't more around centers, such as a 2x2x2 or a 2x2x3; “moving” blocks are, for example, 3x2x1s, 2x2x1s and corner/edge pairs.

Pseudo Blocks, Premoves and NISS

To make what I mean by “pseudo blocks” clear it is better to start with an example:

Scramble: F' L2 F2 U2 R2 B R2 F' R2 D2 U2 L' U' B' U R U L2 F2 L'

R2 F makes a 2x2x1 block. It would be nice to expand it to a 2x2x2 in 2 or 3 moves, but the 4 moves required (L' U B' D) are maybe too many. But try with L2 D': what you get is not an actual 2x2x2 block, but a *pseudo* 2x2x2 block. We can think of the D layer as it was temporarily off by a D2 move, that we can do at the end to solve everything back. For example, we can continue with a CFOP style:

B' U2 R' U2 R2 U R
U2 F' U F U' F' U' F
L U2 L'
B L B' U' B U L U' B'
F2 D' L2 D F2 R2 D B2 D' R2
D2

In this case the D2 could have been done before the OLL, or between OLL and PLL, but if you don't finish the F2L as an intermediate step you have to do it at the end.

But this “pseudoness[62]” makes it harder to go on with the solve (anyone who is not an expert will find it difficult, for example, to recognize the F2L pairs). Someone advises to try solving with pseudo blocks, but there is a trick that makes everything easier: you just need to perform the move that should be done at the end (in this case, D2) **before the scramble**[63]. Try it out!

Scramble: *D2* F' L2 F2 U2 R2 B R2 F' R2 D2 U2 L' U' B' U R U L2 F2 L'
Solution:
R2 F L2 D'
B' U2 R' U2 R2 U R
U2 F' U F U' F' U' F
L U2 L'
B L B' U' B U L U' B'
F2 D' L2 D F2 R2 D B2 D' R2

Once you have found such a solution, remember that the premove (or premoves, if there is more than one) has to be **added at the end** of the solution, to solve the given scramble.

You can use **more than one premove**: for example in this solution I have found them one at the time. It is a little harder to recognize pseudo blocks that require more than one premove: with our scramble, start with the same 2x2x1 (R2 F); even and expert will fin it difficult to see that adding D F' as premoves you get a 2x2x2:

Scramble: *D F'* F' L2 F2 U2 R2 B R2 F' R2 D2 U2 L' U' B' U R U L2 F2 L'
2x2x2: R2 F

But such premoves are not hard to find, if you know **NISS** (Normal-Inverse Scramble Switch). To

62 I hope the meaning is clear.
63 That is why they are called “premoves”.

understand this technique, we need some of theory that there is behind it.[64]

The scramble and the solution can be thought of as a single move sequence loop, that doesn't affect the cube in any way. For example:

Scramble: A B C D
Solution: p q r s

The sequence A B C D p q r s brings the cube back to its solved state. In the same way, every "shifted" version of this sequence:

s (A B C D p q r s) s' = s A B C D p q r
r s (A B C D p q r s) s' r' = r s A B C D
q r s (A B C D p q r s) s' r' q' = q r s A B C D
p q r s (A B C D p q r s) s' r' q' p' = p q r s A B C D
D p q r s (A B C D p q r s) s' r' q' p' D' = D p q r s A B C
…

doesn't affect the cube.[65]

In our first example with premove D2, the loop would have been:

(Scramble) R2 F L2 D' (Other moves) D2

And preforming D2 at the beginning only means taking one of the shifted variations:

D2 (Scramble) R2 F L2 D' (Other moves)

In other words, we can consider "R2 F L2 D' (Other moves) D2" as a solution for "(Scramble)" or "R2 F L2 D' (Other moves)" as a solution for "D2 (Scramble)".[66]

This should be enough to understand how premoves work. Knowing this, we can also find a solution partly on the normal scramble scramble and partly on the inverse one. How? Consider the same scramble and the same start R2 F. We know that, starting from here, our solution will look like R2 F (W), where (W) stay form some sequence of moves. Our loop sequence is:

(Scramble) R2 F (W)

As we said before, the inverse of this is also a "magic" loop (It is equivalent to finding a solution using the inverse scramble):

(W)' F' R2 (Inverse Scramble)

We can therefore consider "(W)' F' R2" as a solution for "(Inverse Scramble)" but also, for example, "(W)'" as a solution for "F' R2 (Inverse Scramble)". This way, you can use **the inverse of the moves found on normal scramble as premoves for the inverse scramble**. Find a solution (let's call it (K), which in our example corresponds to (W)') for inverse scramble with premoves and you are done: your final solution would be R2 F (K)'.

You can repeat this process: suppose you have found the moves F D' on inverse scramble with

64 Taken from this post by Tomoaki Okayama.
65 Inverse sequences don't affect the cube either.
66 You can also see the scramble as a solution to the solution.

premoves (which make a 2x2x2 block), but no good continuation. At this point we can go back to normal scramble, using D F' as premoves. In fact, the loop sequence is:

F' R2 (Inverse Scramble) F D' (Moves yet to be found)

Inverting it gives another magic loop:

(Inverse of moves yet to be found) D F' (Scramble) R2 F

So we can consider D F' as premoves for the original scramble and start with R2 F.

An example (taken from here) will make everything clearer:

Scramble: (fmc.mustcube.net #265) F2 B' R' B D B F' R B F L' D2 U2 B2 D U' B' F D2 L' F2 B' R2 B2 R L' F2 U' F' L
Solution (by Guus Razoux Schultz): D' B' U B2 D' L F2 L' D R U2 R U2 B' R' U2 RU' R B' R' F2 U' L B' (25 HTM)

Explanation:
Nice start on inverse scramble: B L' U F2
Apply on normal scramble: [pre-moves F2 U' L B'] nice continuation: D' B' U B2 R2
Apply on inverse scramble: [pre-moves R2 B2 U' B D]: B L' U F2
Easy continaution on inverse scramble:
F2L: R B R' U R' U2 R B
LL: U2 R' U2 R' D' L F2 L' D R2
Premoves correction: R2 B2 U' B D, 2 moves cancel (25)

To make writing such solutions shorter and easier, I have proposed the following notation[67]: write the parts of the solution found on inverse scramble in brackets. So Guus' solve becomes:

Nice start: (B L' U F2)
Nice continuation: D' B' U B2 R2
F2L: (R B R' U R' U2 R B)
LL: (U2 R' U2 R' D' L F2 L' D R2)

Simple and compact notation, that avoids repeating “On normal/inverse scramble with premoves...”.

Reverse NISS

It is not a widely used technique, but it can occasionally be useful: it can be considered an improvement over both “Conjugate and Solve” and “Go Back and Pimp your Solve”. Suppose you have found a good skeleton solving everything but a few pieces (from 4 to 8); you would like to insert an algorithm to solve them, but if these pieces don't have at least one color in common (they don't belong to the same layer) it can be hard to recognize which algorithm to use. The trick is this: if you find a spot in the solve where all your unsolved pieces are conjugated to one layer, you can **“split” the solve** there, using all the following moves as premoves (that is why I called it “Reverse NISS”); at this point, you have only a “Last Layer” left to solve. If needed, you can use some setup

67 It has not yet gained a good popularity,so if you use it give some short explanation like “move in brackets are done on inverse scramble”.

moves. Let's take this solve as an example:[68]

Scramble: L2 D2 F2 U' L2 D L2 D2 B2 U' F' U' R' D B2 L D B' F' R'
Solution: F2 U R' U' F D2 F R F D F2 R F R2 D R D2 R' F2 U2 R D R' D2 B2 R2 (26 HTM)

Explanation:

All but 5 pieces: F2 U R' U' F D2 * F' U2 R D R' D2 B2 R2 (easy to follow if you use R2 as premove). Inserting F R at dot conjugates unsolved pieces to one layer. Let's take this skeleton:

F2 U R' U' F D2 F R + R' F' F' U2 R D R' D2 B2 R2

And “split it” at +: use R' F' F' U2 R D R' D2 B2 R2 as premoves for normal scramble; now:

Premoves: R' F2 U2 R D R' D2 B2 R2
F2L: F2 U R' U' F D2 F R
LL: F D F2 R F R2 D R D2

Useful Algorithms

As you can see, in the last example solve I have use a maybe not that well known OLL: modulo rotations, R U R2 F R F2 U F (U2). This in particular is very useful, because it is the shortest algorithm that affects the orientation but not the permutation of pieces.

It is in general useful to know some of the shortest last layer algorithms, **up to 9 or 10 moves**; you can find a complete list (modulo rotations, symmetries and inverses) here.[69]

If you decide to break the “never build an F2L without influencing the last layer” rule (sometimes it is worth trying!) you can hope the last layer algorithm is very short: in this case, the more you know, the better![70]

There are two other reasons why it is worth learning some algorithm, at least the ones from 6 to 9 moves: the first is that even if you don't know the exact algorithm for the last layer, or if you haven't completed the F2L, one of these algorithms may leave you with a good skeleton (for example, a corner 3-cycle), hopefully canceling some move.

The other is that by studying algorithms you can learn to match blocks or complete the F2L. Let's take a look at the optimal T perm:

R2 Uw R2 Uw' R2 y L2 Uw' L2 Uw L2 (U')

It is just a useful F2L algorithm repeated twice.

Besides just learning algorithms, you can also **learn *from* the algorithms**.

Pair Analysis

A really obscure technique, based on intuition, not proven to actually give you some advantage

68 Actually, in this solve, the algorithm was easy to recognize simply by marking the pieces with arrows and Xs, so I didn't use this technique, but this solution is a good example anyway.
69 Some of them have a name or some comment, that I have given them for personal use: you can safely ignore them. I may publish a well commented and sorted list in the future.
70 When using a last layer algorithm, remember that you can try performing the AUF (“Adjust Upper Face”) both before and after it, hoping to cancel some move with the ones before it of with premoves.

during the solve. What is it about? “Analyzing pairs” means taking into account some things about the scramble, which are:

- Ready made pairs. There isn't much to say.
- Pairs that can be made **with only one move**. To find them you can use the “brute force” technique described earlier or try to recognize them at first sight. Moreover, it can be useful knowing how to recognize **pseudo pairs**, that pairs that can be solved with only one move on inverse scramble. Moves that make a pair, both on normal and on inverse scramble, can be taken as first move, or as a premove for the inverse of the scramble you are considering.
- **Bad Pairs**: those corner/edge pairs that are wrongly matched, because one of the pieces is misoriented. Intuitively, such pairs are bad and you want to break most of them as soon as you can.

There isn't much documentation about this technique, especially for bad pairs. Guus Razoux Schultz did a good analysis for this scramble.

4. How to Practice

Many people say that to get better you need to “practice, practice, practice”. It is true, but you also need to know *how* to practice: here is some advice on how to practice to get better at FMC.

No Time Limit and Simulating Competitions

Always trying to simulate a competition and forcing yourself to complete your solution in one hour is not the best thing to do: on the contrary, I suggest **not limiting your time** and trying the same scramble again and again until your are satisfied with your result.

This doesn't mean doing one hour solves is bad: it tells you what your level is and it helps you finding a good time management strategy[71]. To train like this I suggest taking part in the online weekly competition on David Adams' website.

Trying for one hour as it was a competition and then keep trying until you reach a good result is a balanced compromise.

Compare yourself to Masters (and Study their Solves)

When you want to practice, I suggest trying a scramble that has already been solved by some expert FMCer, so that you can compare your solution to their and find out whether you have missed a good start or anything else.

Moreover, to train blockbuilding and other methods it is mandatory to study the masters' solutions: you can find many of them on http://fmcsolves.cubing.net/.

Hard Scrambles

To see what you can do in the “worst case scenario”, I suggest trying out some scrambles that are considered by expert to be really hard. You can find a list of hard scrambles here.

Deliberate Practice

If you think you have troubles in finding a good start, practice that: take a scramble, find a 2x2x2, 2x2x3 or something else until you are satisfied, than change scramble. You can apply this idea to F2L-1 or any other substep.

Fast Scrambling

Even if it is not necessary, when using techniques like NISS, or simply if like me you tend to solve and scramble the cube many times during a solve, you should try to be at least “not too slow”[72] at scrambling, and most important to be **accurate** (not making mistakes).

---

71 I will give you some advice in chapter 6.

72 10 seconds or less for a 20 moves scramble is fine.

Study!

Last but not least. Study this guide, study from other sources, study algorithms and techniques, new or known. I have read “The FMC Thread” on speedsolving.com *twice*, from top to bottom.

Study algorithms: there are dozens of different sets, to mention some of them LLEF (Last Layers Edges First) or Summer Variation. Remember that you shouldn't just *memorize* them, but also try to understand *how they work*.

5. Some Other Ways

The “standard” method is building a skeleton with blockbuilding and then solving the last few pieces with insertions. But there are some other approaches worth mentioning.

Starting with EO

You should always remember this possibility: **starting by orienting all edges**, as in a ZZ solve. From here you have two ways to continue.

EO + Blockbuilding

After having oriented all edges, the most common way to go on is blockbuilding. The pro is that we don't have any “bad” edge, but this forces[73] us not to use moves that break the EO, and this is a (relatively small) limit.

Grzegorz Łuczyna usually started with EO; here is the solve that made European Champion 2010:

Scramble: L D2 B' D2 B R' B' U B L B L2 B2 U2 F2 U R2 D' B2 D' B2
Solution:
EO: x y2 L2 D F'
EOLine: R L2 D #
2x2x3: R' U2 B2 R2 B2
All but 3 corners: L2 U' R' U L U' L' R U2 L' U'
3 corners: [D2, R' U' R] at # (4 moves cancel)

“Domino” Reduction (and HTA)

Edges orientation can be considered, modulo rotations, a reduction to the *subgroup* generated by the moves <R, L, U, D, F2, B2>. Another step in this direction leads to reducing the cube to subgroup generated by <U, D, R2, L2, F2, B2>; to do so you have to:

- Placing E layer edges on the E layer.
- Orient corners.

This reduction is also called “Domino”, because it makes a Rubik's Cube solvable as a 3x3x2 “cube” (called “Domino Cube”). Moreover, these are the first two steps of Human Thistlethwaite Algorithm (HTA), a modified version of Thistlethwaite Algorithm. If you are interested in this method for FMC, I suggest this tutorial.

Here is an example solve by Per Kristen Fredlund:

Scramble: D U' R' F B2 R B2 R' U2 R B2 R' U2 R B2 R' U2 R B2 R' U2 B2 F2 U D F D' B2 D F D' B2 D F D' B2 D F D' B2 F' D B' F R'
Solution:
R' B U' D F //EO (5/5)
L' F2 L //Domino Reduction(3/8)

73 Obviously, no one is forcing you to do anything, but orienting edges and then destroying what you have just done doesn't look like a smart thing to do. You can also start with a *partial* EO if you wish.

D2 L2 F2 D F2 D L2 U' R2 D2 R2 //Finish (11/19)

Corners First

“Corners First” (sometimes shortened to CF) is not really a method, but a class of methods that, as the name says, solve **the corners first**, and then the edges. Roux can be considered a CF method.

Among the ones who have figured out how to solve the cube on their own, many had a corners first approach:[74] Thinking separately about corners and edges makes it somehow easier to solve the cube intuitively. Moreover, by solving the corners first you can more freely solve edges: inner layers can be moved without affecting corners.

But this is also a disadvantage in FMC: inner layer moves count as two moves! Despite this, there at least two expert FMCer that use this technique: **Attila Horváth[75],** who has only participated in two official competitions and **Javier Cabezuelo Sánchez[76]**, Spanish national record holder. Both say that Corner First methods are excellent for FMC, but not very suitable for the one hour time limit.[77]

Attila Horváth mostly solves corners using a method similar to Guimond (orient first); centers are not cared about in this step; for this step he sometimes uses premoves or NISS. After this, he goes through the solution he has found and modifies it slightly, inserting inner layer moves, to get at least 2 or 3 edges solved. At the end he solves the edges left, not in a specific order. He sometimes doesn't solve the center until the end, and solves them with an insertion[78], usually canceling many move. Example solve.

Javier Cabezuelo Sánchez solves the corners in a different way: first layer corners first, then the other. He then tries to solve the edges inserting moves (or algorithms) in the solution he has found. He doesn't use techniques such as inverse scramble, premoves or NISS. Differently from Attila, he cares about centers while solving corners.[79]

Both Attila and Javier only use their CF method and are not wiling to learn more “standard” ones, which break the “never restrict yourself” rule; but they still get excellent results.

74 For example, Valery Morozov, who has made a tutorial to learn his method.
75 WCA profile: 2012HORV01; speedsolving.com profile: Attila.
76 WCA profile: 2007SANC01.
77 Notice how many of Javier's results are DNFs.
78 Algorithms like M E M' E' or M E2 M' E2
79 Source: this post.

6. In Competitions

How to Write a Solution

Both in competition and while practicing, write it down **without rotations**. There are many good reasons to do so:

- Using rotations, it is easier to make mistakes.
- Rotations can hide cancellations.[80]
- While solving, if you rotate the cube, you always need to keep in mind which side is where.

How to write a solution without rotations? A PLL in B is obviously awkward. There is an easy way: keeping the standard orientation, with a BOY color scheme (the “standard” one), you just need to remember that the white-centered layer is always U, the green-centered one is F, the yellow-centered one is D and so on.[81] Every time you move a layer, for example the white-centered one, you don't need to care about how you are looking at the cube at that moment: just write U.

Backup Solution

It is good habit, in time-limited competitions, to write a “backup solution”. It is usually a not so good solution, but still better than a DNF, found before the last moments, when haste may lead to making a mistake or not finding a solution at all. If, for example, you average about 35 moves, but after 20 minutes you have found and written down somewhere[82] a 40 moves solution, you will be calmer for the remaining 40 minutes. There are many possible approaches to finding a backup solution:

- Force yourself to have found and written a solution, no matter how bad, in a fixed time limit (i.e.: 35 minutes); I don't do this, but it can be useful if you often find yourself at the end of the hour without anything written down.
- If you casually come across some solution (i.e.: you have found a good start and solving the cube to re-scramble it you get a PLL skip), take note of it somewhere.
- What I do: I don't really find backup solutions, but many backup *skeletons*. For example, my goal is usually sub-30; in this case, a skeleton leaving 3 corners in 26 moves is not good, but if I find one I keep it somewhere. If I have, for example, 10 minutes left and I don't have anything better, I look for an insertion in that skeleton.[83]

Which can be a good backup solution? Any solution! Anything is better than a DNF, especially now that the preferred format for FMC (in official competitions) is “Mean of 3”: a single DNF gives you a DNF mean.

---

80 For example, if you write something like ... R z' U' … you deserve being made fun of.

81 To help memorizing the scheme (not that it is hard), remember that Blue and Red begin with the same letter as their layer.

82 You can even write it down on the official sheet: if you later want to change your solution, you can delete the backup solution and write down the new one.

83 A single 3 corners insertion usually takes me 5 minutes.

Time Management

“How to manage your time” is a complex topic, and I don't want to say that my advice is absolutely good in any case: follow it carefully! The best teacher, in this case, is personal experience.

Don't Get Stuck!

It can happen to anyone: during a competition you get stuck on a certain start and don't seem to find any better continuation. The ability to quickly understand if a start can lead to a good continuation would as useful as being able to read other people's mind. My advice, maybe trivial, is: don't get stuck. If you have tried every method and technique you know and found nothing, don't stare at the cube hoping it solves itself: go back and try something else.

Explore Every Possibility

If you are computer scientist, mathematicians or so I can tell you that exploring different possible solution is actually a tree *search*: you can choose if you prefer a DFS (depth-first search, trying to complete a solution before moving to another one) or a BFS (breadth-first search, “advance” every solution in parallel) approach, or a mixed one. Keep in mind that form a single partial solution can derive a lot of nice branches as well a none; for this reason I don't use a fixed method. It is also important to know when to prune a branch (that is, discarding unpromising partial solutions).

I hope also non-nerds could understand what I wrote.

7. Other Resources

To conclude, here is a list of the sources where I got all this information from and other useful links:

- David Adams' website, hosting a weekly competition, in which you can compete against cubers of different levels. Very useful to study different solutions for the same scramble.
- fmc.mustcube.net by Per Kristen Fredlund, offline at the moment, used to host a similar weekly competition.
- Insertion Finder, useful to find insertions.
- JARCS, a tool to find (and study) optimal solutions to many common substeps. Offline at the moment.
- fmcsolves.cubing.net, a Blog/Database for FMC solves, both official and unofficial.
- menas.com.br, a collection of official solves by Brazilian cubers, each with an explanation video (in Portuguese).
- Speedsolving.com, a good reference for cubing in general; in particular:
  - The FMC thread, dedicated to Fewest Moves.
  - Fewest Moves: Tips and Techniques, another thread for FMC.
  - Commutators Tutorial by Brian Yu
- Ryan Heise' website, explaining both his method and some “fundamental techniques”.
- Lars Petrus' website, with useful blockbuilding examples
- A video by Daniel Sheppard with come advice on how to get better
- 5-video tutorial by Ranzha: [1] [2] [3] [4] [5]
- A presentation by Pranav Maneriker.