Algorithm Finder Lite

Content

Overview

Algorithm Finder Lite is an on-line tool that can be used to search for structured cube algorithms. In this context, a sequence of moves is said to be structured if the ordering of moves can be predicted in some way. For example, commutator [X, Y Z P Q] = X Y Z P Q X' Q' P' Z' Y' is structured because half of the sequence can be predicted from the other half. By imposing a structure to a sequence and searching for all possible occurrences of the structured sequence, short algorithms can generally be found.

Algorithm Finder Lite
Algorithm Finder Lite actually includes 2 distinct Finders that can be used alternately:

Algorithm Finder
A user will first enter a generator algorithm or fill in a layout mask with permuted stickers, set a number of parameters and enter his own algorithm template(s). *Algorithm Finder* will then sweep all possible cases, using optional inversion, shift or symmetry, and give a list of sorted solution algorithms.

Algorithm Finder can search for algorithms for cube sizes of 2, 3, 4, 5, 6, 7 layers and even beyond.

Seed Finder
A user will first set a number of parameters and enter his own algorithm template(s). There is no need to enter a generator algorithm or fill in a layout mask in this case, as all stickers of the selected orbit of pieces will be automatically set to –1 (don't care setting). *Seed Finder* will then sweep all possible cases, remove inversion-, shift- and symmetry-duplicates and give a list of sorted *irreducible* seed algorithms.

Seed Finder can search for algorithms for cube sizes of 2, 3, 4, 5, 6 and 7 (default case). In the latter case, algorithms can be found for the following 7 orbits of pieces: 00 (true centers), 03 (corners), 12 (midges), 06 (edges), 05 (corner-centers), 11 (midge-centers) and 08 (edge-centers).

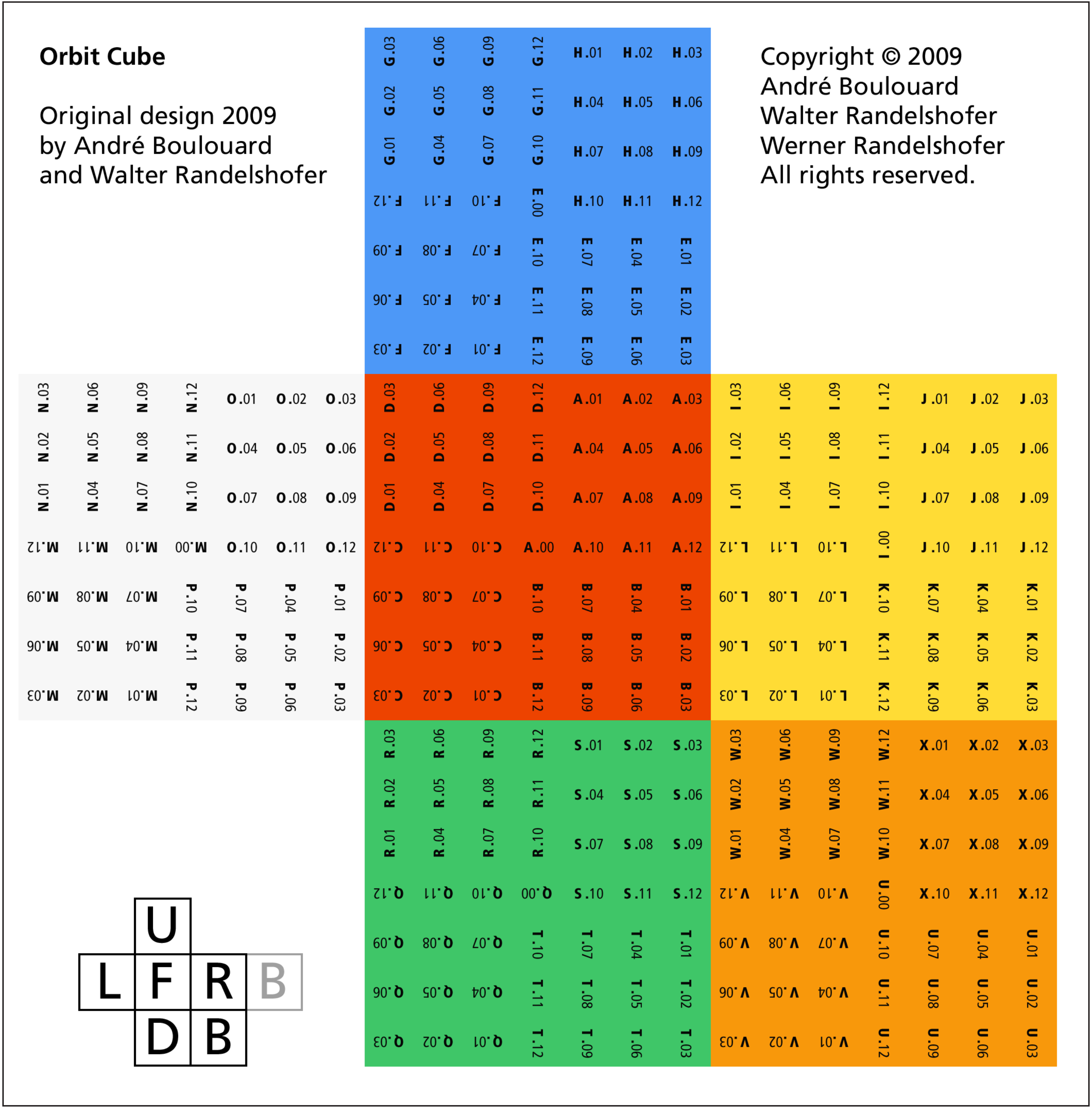
Algorithms of all other orbits can easily be derived from these by orbit scaling, e.g. by replacing NX with N3X (orbit 05 → orbit 07).

For more information on *Orbit Cube Notation*, see the 7x7x7 Orbit Cube texture:
www.randelshofer.ch/rubik/virtual_cubes/vcube7/picture_cubes/7x_orbit_cube.html

Limitations
A brute force search method is applied by the Finder to search for algorithms. The maximum number of variables that can be used in a structured sequence depends on the number of basic moves in the selected set and is limited by the combinatorial explosion phenomena. Care must then be taken to insure that the number of variables is not too high to save on computation time. For example, if we consider the case of 5 variables X, Y, Z, P, Q and edge-centers with a set of 54 basic moves, the number of checked combinations will be:

54^5 = 459’165’024 ≈ 459 M (Algorithm Finder)
$6.(54 - 3)^4$ = 40’591’206 ≈ 40 M (Seed Finder)

Notice that if shift, symmetry, inversion or conjugation were applied to these cases, the actual number of checked combinations would have been far greater than that. As a rule of thumb, it is preferable to keep the total number of checked combinations below 100 M to keep computing time within reasonable limits, although overnight computing might be of help…



7x7x7 Orbit Cube texture

Sticker notation

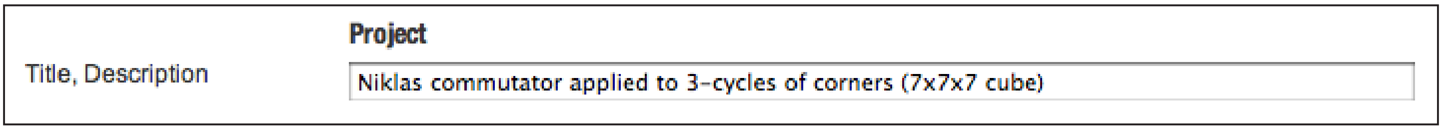
In *Super Cube Layer Notation*, stickers are numbered from 0 up to 6 × N^2 − 1, e.g. from 0 to 293 for a 7x7x7 cube and are arranged by layers, in increasing order from face F to face B (FRULDB).

In *Orbit Cube Notation*, stickers are labelled in dot notation as: *CapitalLetter.OrbitIndex*, where *CapitalLetter* is one of the first 24 letters of the alphabet (A…X) and *OrbitIndex* is the index of the orbit the sticker moves in.

Conversion from *Orbit Cube Notation* to *Super Cube Layer Notation* is done at the applet interface level and is completely transparent to the user.

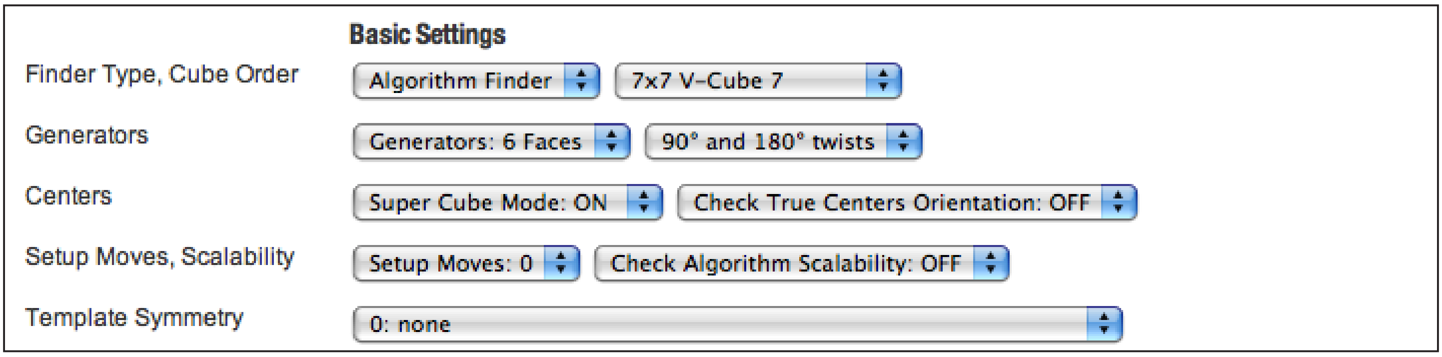
Settings common for both Finders

Project



Title, description
Add text to the box to give a brief description of your project (optional).

Basic settings



Finder type
Defines which type of Finder, either *Algorithm Finder* or *Seed Finder*, is used for the search.

Cube order
Algorithms can be found for cube sizes ranging from 2 to 127 layers, ie. for the following cube models: *2x2 Pocket Cube*, *3x3 Rubik's Cube*, *4x4 Revenge Cube*, *5x5 Professor Cube*, *6x6 V-Cube 6* and *7x7 V-Cube 7, and so on*.

In *Seed Finder* mode, the *Cube Order* range is limited to 2x2 Pocket Cube, *3x3 Rubik's Cube*, *4x4 Revenge Cube*, *5x5 Professor Cube*, *6x6 V-Cube 6* and *7x7 V-Cube 7*.

Generators
The Finder will automatically extract the *minimum* set of generators from locations of permuted stickers on the mask layout, thus freeing the user from manually entering the moves. For example, if permuted stickers are those of corners only, then the set of generators will be the set of 18 basic moves: {R, R', R2, U, U', U2, F, F', F2, L, L', L2, D, D', D2, B, B', B2}. Notice that the number of generators that will be used can be further reduced by modifying face and turn indices, to lower computing time.

Super Cube mode
Set *Super Cube Mode* to *ON* if centers must be located at their right *Super Cube* positions.

Set *Super Cube Mode* to *OFF* if colors of centers and faces must be the same and if the orientation of true centers does not matter. This will select algorithms that won't change the color of centers located on each of the 6 cube faces, but may change their respective locations. This way, centers located on a given face may not be at their right *Super Cube* positions but will still retain their original color, eg. all centers located on the blue face will be blue.

Check true centers orientation

Set *Check True Centers Orientation* to *ON* if the orientation of true centers matters.

Setup moves

Setup Moves are inserted at both ends of an algorithm to move permuted pieces to other cube locations. *Setup Moves* inserted at the end of an algorithm are just the inverses of those inserted at the beginning, but applied in the reverse direction. Any number of *Setup Moves* can be applied to an algorithm, although 1, 2 or 3 moves are most commonly used.

Examples:
U (NF D' F' D NF' D' F D) U'
NR' U (NF D' F' D NF' D' F D) U' NR

Check algorithm scalability

Set *Check Algorithm Scalability* to *ON* to find scalable algorithms, ie. algorithms that can be easily scaled up to higher order cubes or to more complex patterns.

The option is active for odd-order cubes and algorithms including *M* moves only, so that the number of letters *M* in filtered algorithms will be either 0 or 2.

Template symmetry

Knowing that a cube has a set of 48 symmetries, a given algorithm can be transformed by applying one of those symmetries to each basic move of the sequence.

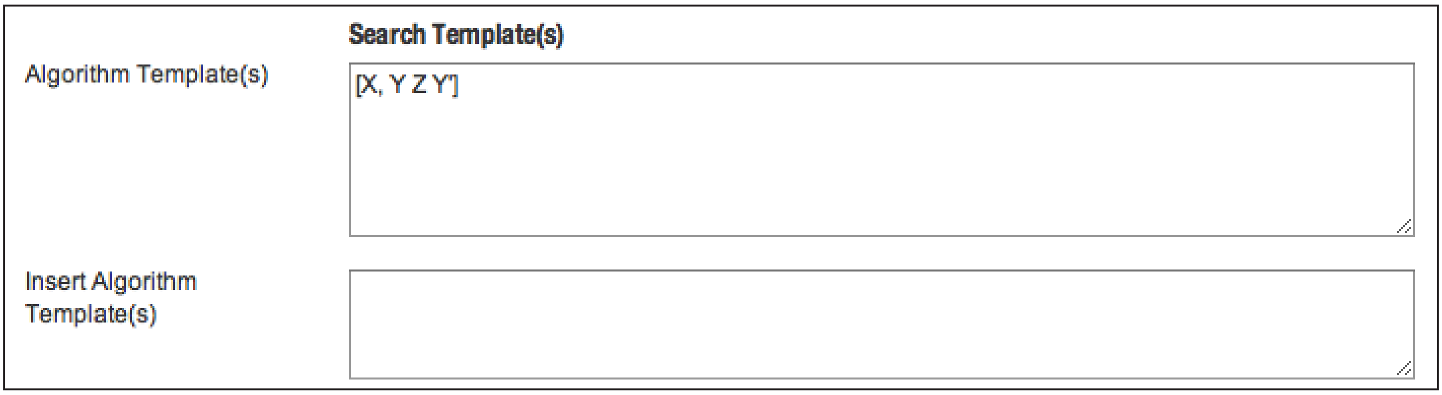
Example: in the symmetry by rotation of +180° about the F – B axis, faces are transformed as: F→F, R→L, U→D, L→R, D→U, B→B and algorithm (NF D' F' D NF' D' F D) is transformed into (NF U' F' U NF' U' F U).

A cube symmetry is defined by an index ranging from 0 to 23 for cube rotations and from 24 to 47 for cube rotations + point reflection, as shown in table «Cube symmetry – rotations».

Cube symmetry – rotations

Symmetry index	Rotation axis	Rotation angle	Cube rotation	
0	none	none	none	
1	F – B	+90°	CF	
2	F – B	+180°	CF2	
3	F – B	–90°	CF'	
4	R – L	+90°	CR	
5	R – L	+180°	CR2	
6	R – L	–90°	CR'	
7	U – D	+90°	CU	
8	U – D	+180°	CU2	
9	U – D	–90°	CU'	
10	FRD – ULB	–120°	CF' CR'	
11	FRD – ULB	+120°	CF CU'	
12	FLD – RUB	–120°	CF' CU	
13	FLD – RUB	+120°	CF CR'	
14	FRU – LDB	–120°	CF' CU'	
15	FRU – LDB	+120°	CF CR	
16	FUL – RDB	–120°	CF' CR	
17	FUL – RDB	+120°	CF CU	
18	FD – UB	+180°	CF2 CR'	
19	FU – DB	+180°	CR CU2	
20	RD – UL	+180°	CF CU2	
21	RU – LD	+180°	CF' CU2	
22	FR – LB	+180°	CU' CR2	
23	FL – RB	+180°	CF2 CU	
24	none	none	none	+ point reflection
25	F – B	+90°	CF	+ point reflection
26	F – B	+180°	CF2	+ point reflection
27	F – B	–90°	CF'	+ point reflection
28	R – L	+90°	CR	+ point reflection
29	R – L	+180°	CR2	+ point reflection
30	R – L	–90°	CR'	+ point reflection
31	U – D	+90°	CU	+ point reflection
32	U – D	+180°	CU2	+ point reflection
33	U – D	–90°	CU'	+ point reflection
34	FRD – ULB	–120°	CF' CR'	+ point reflection
35	FRD – ULB	+120°	CF CU'	+ point reflection
36	FLD – RUB	–120°	CF' CU	+ point reflection
37	FLD – RUB	+120°	CF CR'	+ point reflection
38	FRU – LDB	–120°	CF' CU'	+ point reflection
39	FRU – LDB	+120°	CF CR	+ point reflection
40	FUL – RDB	–120°	CF' CR	+ point reflection
41	FUL – RDB	+120°	CF CU	+ point reflection
42	FD – UB	+180°	CF2 CR'	+ point reflection
43	FU – DB	+180°	CR CU2	+ point reflection
44	RD – UL	+180°	CF CU2	+ point reflection
45	RU – LD	+180°	CF' CU2	+ point reflection
46	FR – LB	+180°	CU' CR2	+ point reflection
47	FL – RB	+180°	CF2 CU	+ point reflection

Search template(s)



Algorithm template(s)

Enter a single template or a list of templates, separated with CR/LF. Characters and moves in each template are first checked by the built-in parser and must have been recognised before launching search.

Templates can vary from completely unstructured sequences of unordered moves and variables to highly structured sequences such as (semi-)commutators or (semi-)conjugators.

[X Y, Z P Q A] = (X Y Z P Q A Y' X' A' Q' P' Z') is a commutator whereas]X Y, Z P Q A] = (X Y Z P Q A X' Y' A' Q' P' Z') is an example of a semi-commutator. Semi-commutators may be useful for finding short sequences, especially for cycles more complex than 3-cycles, e.g.. 2 2-cycles, 5-cycles or even beyond.

Example: Here are some additional examples of semi-commutators:

[X, Y Z P Q[= (X Y Z P Q X' Y' Z' P' Q')
[X Y, Z P Q[= (X Y Z P Q Y' X' Z' P' Q')
]X Y, Z P Q] = (X Y Z P Q X' Y' Q' P' Z')
]X Y, Z P Q[= (X Y Z P Q X' Y' Z' P' Q')
[X, Y] [Z, P Q[= (X Y X' Y') (Z P Q Z' P' Q')

Variable names

Variable names have been chosen not to interfere with SSE or WCA move notation, to give an ordered sequence of up to 11 letters: X, Y, Z, P, Q, A, G, H, I, J, K. If variable X takes values in {NR, NR', NR2}, for example, then X' will take values in {NR', NR, NR2} in that precise order. Any number of variables from set {X, Y, Z, P, Q, A, G, H, I, J, K} can be used in a template, provided the following rules are met:

1 variable: must be selected from subset {X, X', X2, Xo, Xo', Xo2, Xs, Xs', Xs2},
2 variables: must be selected from subset {X, X', X2, Xo, Xo', Xo2, Xs, Xs', Xs2, Y, Y', Y2, Yo, Yo', Yo2, Ys, Ys', Ys2}, etc …

For example, variable P can be used *if* and *only if* variables X, Y and Z are already present in the template.

For more information on *Superset ENG Notation (SSE)*, see the notation description:
www.randelshofer.ch/rubik/vcube7/doc/supersetENG_7x7.html

Using variables

To decrease computing time, it's good practice not to use more than 5 variables {X, Y, Z, P, Q}. Notice that up to 3 additional variables {A, G, H} could be used in some specific cases, but at the expense of an increase in computing time, though. When using additional variables, it is highly recommended to set face and turn indices in *Basic Settings* to lower values, e.g. to set *Generators* to *3 Faces* and turn indices to *90° twists only*. In some particular cases, it may also help to use *opposed* or *symmetric* variables instead of additional variables, e.g. use [X, Y Z P Zo] instead of [X, Y Z P Q].

Any reduction in search space will greatly decrease computing time but also decrease the odds of finding algorithms, so the right tradeoff has to be found. Reducing the number of simultaneously used search options may also give faster results.

The use of a fast web browser such as Google Chrome is highly recommended, as it may give more than a 2x decrease in computing time, compared to similar browsers.

<u>Conjugating seed algorithms</u>

Seed algorithms can be conjugated by *Algorithm Finder* by applying a specified number of setup moves. Notice that optional inversion, shift or symmetry can also be simultaneously used to further increase the odds of finding a short solution algorithm. If no algorithm can be found this way, try another seed or set 1 or 2 stickers to –1 and run *Algorithm Finder* again. This may give a cube state that could be fairly close to the solved state. Try to solve this last state afterwards. If solved, then concatenating the 2 separate solution algorithms may still give a short end algorithm.

<u>Deselecting templates</u>

Templates can be individually deselected by inserting an exclamation mark at the beginning of the string.

Example: !NR U NU' NB' U' NR' U NB NU U'

<u>Commenting templates</u>

Templates can be commented by inserting lines of text beginning with double backslash marks.

Example: // Comment

<u>Symmetric templates</u>

The usual commutator/conjugator notation has been extended to include symmetric templates, defined by a *Template Symmetry* index. Examples of 5-variable symmetric templates are given below:

Symmetric commutator:	[X Y Z, P Q] = X Y Z P Q Zs' Ys' Xs' Qs' Ps'
Symmetric semi-commutator:	]X Y Z, P Q[= X Y Z P Q Xs' Ys' Zs' Ps' Qs'
Symmetric semi-commutator:	]X Y Z, P Q] = X Y Z P Q Xs' Ys' Zs' Qs' Ps'
Symmetric semi-commutator:	[X Y Z, P Q[= X Y Z P Q Zs' Ys' Xs' Ps' Qs'
Symmetric conjugator:	[X Y Z : P Q] = X Y Z P Q Zs' Ys' Xs'
Symmetric semi-conjugator:	]X Y Z : P Q] = X Y Z P Q Xs' Ys' Zs'
Symmetric sequence:	[X Y Z P Q] = X Y Z P Q Qs' Ps' Zs' Ys' Xs'
Symmetric semi-sequence:	]X Y Z P Q] = X Y Z P Q Xs' Ys' Zs' Ps' Qs' (same as]X Y Z, P Q[)
Symmetric repetition:	X Y Z P Q Xs Ys Zs Ps Qs

In these expressions, variables represent individual moves and subscripted variables are variables transformed by symmetry.

For patterns showing symmetry, searching for algorithms using symmetric templates may give short solutions.

<u>Algorithm Finder</u>

When using *Algorithm Finder*, variables and moves can be mixed up in any order inside a template, e.g. [Z B NU, Y X F2] would be a perfectly acceptable algorithm template.

Seed Finder

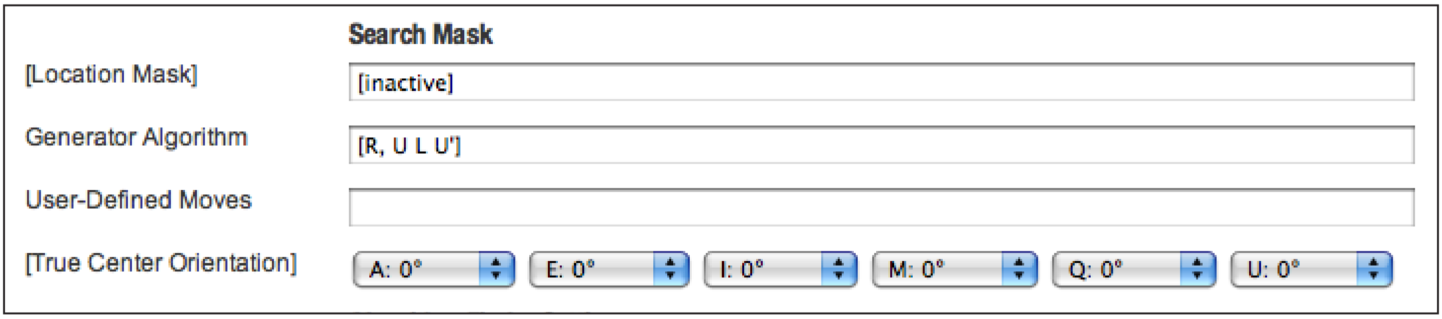
When using *Seed Finder*, however, it is recommended to use variables first in their ordered sequence, beginning with variable 'X', and not to mix up variables and moves, e.g.. [X Y, Z P Q NF R'] would be an acceptable seed template.

'Insert' Algorithm Template(s)

An *'Insert' Algorithm Template* Ti is inserted at different move positions into a regular algorithm template Tr, until all positions have been visited. Using variables together with other options, cancellation/combination of some consecutive moves may thus be obtained. Inserting Ti into Tr may then give a shorter algorithm than simply concatenating Ti and Tr.

Algorithm Finder settings

Search Mask



Location mask
In works.

Generator algorithm
A *Generator Algorithm* sets the cube to a user-defined goal state. Algorithms are then searched that will give that precise goal state. This is an alternate and quick method of setting the *Location Mask*.

If the *Generator Algorithm* is set to a non-void string, then the goal state will be set from that algorithm.
If the *Generator Algorithm* is set to a void string, then the goal state will be set from the location mask.

User-defined moves
The Finder will automatically extract the minimum set of generators from locations of permuted stickers. The minimum set depends on cube size and consists only of single-slice moves like R, MR, NR, N3R.

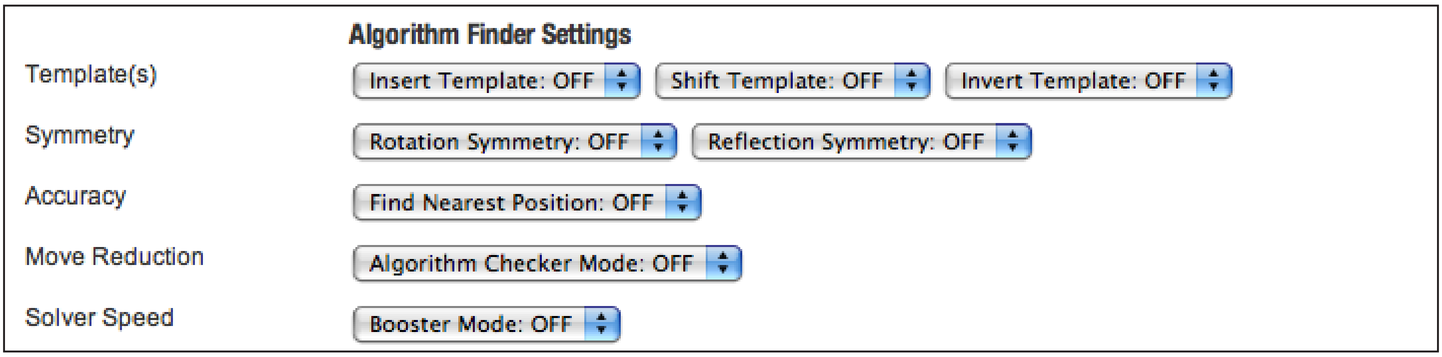
It may however be of interest to manually enter *User-Defined Block Moves*, as an option. Available *User-Defined Block Moves* depend on cube size as follows:

2x2x2: R, CR
3x3x3: R, MR, NR, SR, TR, WR, CR
4x4x4: R, MR, M2R, NR, N3R, SR, S2R, TR, T3R, VR, WR, CR
5x5x5: R, MR, M2R, M3R, NR, N3R, N4R, SR, S2R, TR, T3R, T4R, VR, V3R, WR, CR
6x6x6: R, MR, M2R, M3R, M4R, NR, N3R, N4R, N5R, SR, S2R, S3R, TR, T3R, T4R, T5R, VR, V3R, V4R, WR, CR
7x7x7: R, MR, M2R, M3R, M4R, M5R, NR, N3R, N4R, N5R, N6R, SR, S2R, S3R, TR, T3R, T4R, T5R, T6R, VR, V3R, V4R, V5R, WR, CR

It is sufficient to enter moves in *R* only, because moves in *U*, *F*, *L*, *D*, *B* will be automatically generated from *R* moves. Consecutive moves must be separated by commas. It is recommended not to enter more than 4 distinct types of moves. Notice that under *Generators*, face and turn indices are still active for user-defined moves.

True center orientation
In works.

Algorithm Finder settings



<u>Insert template</u>
Set *Insert Template* to *ON* to include inserted algorithms in the search.

<u>Shift template</u>
Set *Shift Template* to *ON* to remove the last move of a template and add it at the beginning of the template. This is done recursively until all moves have been shifted and all shifted templates executed. The shift operation is similar to conjugating a template. If, for example, a block of k moves is removed at the end of a template and added at its beginning, sequence X Y Z P Q A G H will then become *A G H X Y Z P Q*, when shifted with k = *3*. This is the same as conjugating the initial sequence (X Y Z P Q A G H) by (A G H), because A G H X Y Z P Q = A G H (X Y Z P Q A G H) H' G' A'. Shifting a sequence is a way of obtaining conjugates at a low computing cost: for a sequence of n moves, only n runs are needed. *Shift Template* is generally used together with *Invert Template*, *Rotation Symmetry*, *Reflection Symmetry* and conjugation to increase search space.

<u>Invert template</u>
Set *Invert Template* to *ON* to include inverted templates in the search.

<u>Symmetry</u>
When using *Symmetry* to widen search space, only the *non-variable* part of a template is transformed by *Symmetry*, e.g. in (X Y) [NR NL', NF R' NB' ND] (Y' X'), only [NR NL', NF R' NB' ND] will be transformed.

<u>Find nearest position</u>
Set *Find Nearest Position* to *ON* if the nearest position to the goal position is searched for, instead of the goal position itself. Set *Find Nearest Position* to *OFF* if it is the exact goal position that is searched for.

<u>Move reduction</u>
In *Algorithm Checker Mode*, each algorithm *n* in a list is set as a Generator, whereas all algorithms of the list are conjugated and executed in a row. The shortest algorithm found is then selected and compared to the Generator algorithm. If it is shorter, then algorithm *n* is replaced with the shorter version.

<u>Solver Speed</u>
Set *Booster Mode* to *ON* to decrease computing time by using canonical sequences. A *Generator Algorithm* plus a set of user-defined moves must be entered. Use this option for templates with many variables, eg. [X Y Z P Q A].

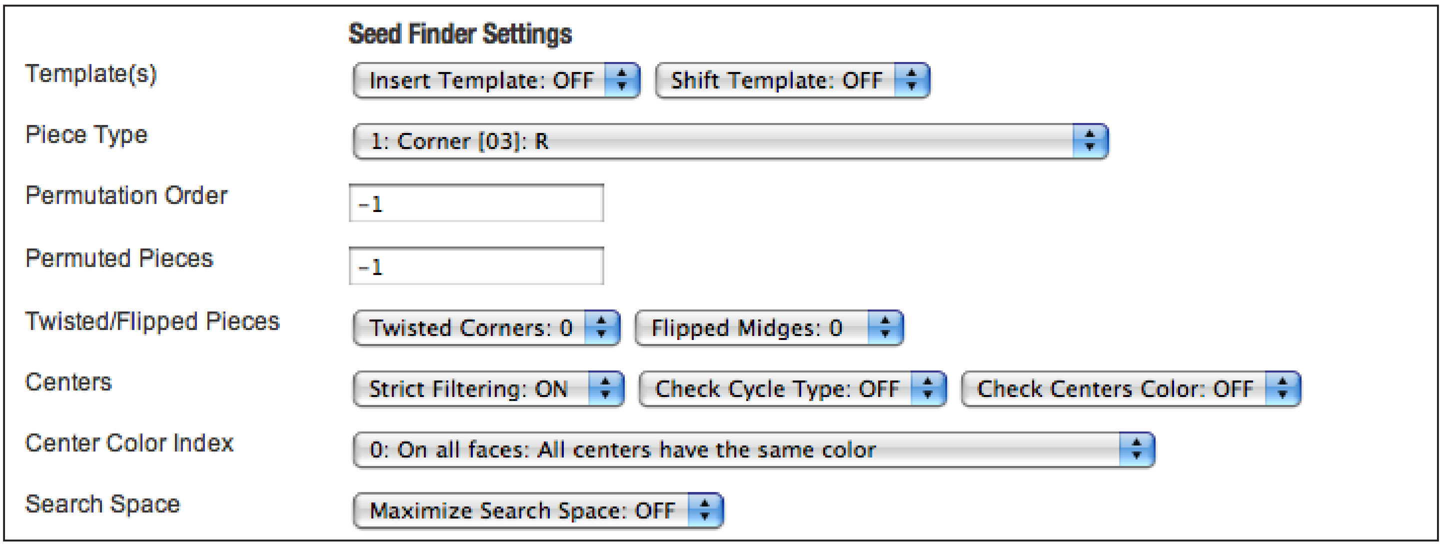
<u>Two-phase search</u>
The *Algorithm Finder* is basically a single-phase solver, where all templates are executed to reach either the exact goal position or the nearest position to goal. If no template can be found that gives a solution to the exact goal position, at least in a reasonable period of time, then the algorithm of the nearest position can be considered as a Phase 1 solution. In Phase 2, this already found algorithm can be placed just in front of new templates that will permute pieces

from nearest position to goal position. By executing the two combined template(s), a solution may eventually be found.

Seed Finder settings

Seed Finder settings

A seed search is defined from 3 parameters: *piece type*, *permutation order* and *number of permuted pieces*. Notice that there are cases where a same permutation order and a same number of permuted pieces may give distinct cycle types, though. This is the case for example of cycle types (2c) (3c) (6c) and 4(2c) (3c), both of order 6 for 11 permuted pieces each. Such cases can only show up for a number of complex cycle types composed of multiple cycles. On the other hand, simpler cycle types such as (17c) do not show this phenomena.



<u>Insert template</u>

Set *Insert Template* to *ON* to include inserted algorithms in the search.

<u>Shift template</u>

Set *Shift Template* to *ON* to remove the last move of a template and add it at the beginning of the template. This is done recursively until all moves have been shifted and all shifted templates executed. This will increase both search space and computing time.

<u>Piece type</u>

Seeds can be found for pieces permuted either in a single orbit or in paired orbits. Orbit numbers are given in the table below for a 7x7x7 cube only:

Piece type index	Piece type	Orbit	Moves
0	True Center	00	R, MR
1	Corner	03	R
2	Midge	12	R, MR
3	Edge	02, 06	R, NR
4	Corner-Center	05	R, NR
5	Midge-Center	11	R, NR, MR
6	Edge-Center	08	R, NR, N3R
7	Corner + Midge	03 + 12	R, MR
8	Corner + Edge	03 + 02, 06	R, NR
9	Corner + Edge	03 + 02, 06	R, TR
10	Midge + Edge	12 + 01, 09	R, N3R, MR
11	Midge + Edge	12 + 01, 09	R, M2R
12	Edge + Edge	01, 09 + 02, 06	R, NR, N3R
13	Edge + Edge	01, 09 + 02, 06	R, VR
14	Corner-Center + Edge	05 + 02, 06	R, NR
15	Corner-Center + Edge	05 + 02, 06	R, NR, TR
16	2 Corner-Centers + Edge	05 + 02, 06	R, NR
17	2 Corner-Centers + Edge	05 + 02, 06	R, NR, TR
18	Midge-Center + Midge	11 + 12	R, NR, MR
19	Midge-Center + Midge	11 + 12	R, TR, MR
20	2 Midge-Centers + Midge	11 + 12	R, NR, MR
21	2 Midge-Centers + Midge	11 + 12	R, TR, MR
22	Edge-Center + Edge	08 + 01, 09	R, NR, N3R
23	Edge-Center + Edge	08 + 01, 09	R, TR, N3R
24	2 Edge-Centers + Edge	04, 08 + 01, 09	R, NR, N3R
25	2 Edge-Centers + Edge	04, 08 + 01, 09	R, TR, N3R
26	Corner-Center + Corner-Center	05 + 07	R, NR, N3R
27	Midge-Center + Midge-Center	10 + 11	R, NR, N3R, MR
28	Midge-Center + Midge-Center	10 + 11	R, VR, MR
29	Edge-Center + Edge-Center	04 + 08	R, NR, N3R
30	Corner-Center + Edge-Center	05 + 08	R, NR, N3R
31	Corner-Center + Edge-Center	05 + 08	R, NR, VR
32	Corner-Center + Midge-Center	05 + 08	R, NR, MR
33	Corner-Center + Midge-Center	05 + 11	R, NR, MR, WR
34	2 Corner-Centers + Midge-Center	05 + 11	R, NR, MR
35	2 Corner-Centers + Midge-Center	05 + 11	R, NR, MR, WR
36	2 Midge-Centers + Corner-Center	11 + 05	R, NR, MR
37	2 Midge-Centers + Corner-Center	11 + 05	R, NR, MR, WR

The number of permuted pieces in each orbit of a selected pair is automatically set by the Finder according to the selected piece ratio (1 : 1 or 2 : 1). For the 1 : 1 piece ratio, the total number of *Permuted Pieces* must be an integer multiple of 2. For the 2 : 1 piece ratio, the total number of *Permuted Pieces* must be an integer multiple of 3.

For example, if *Piece Type* index is set to *22* and the number of *Permuted Pieces* to *10*, then there will be 5 permuted edge-centers plus 5 permuted edges in the paired orbits. Similarly, if *Piece Type* index is set to *17* and the number of *Permuted Pieces* to *12*, then there will be 8 permuted corner-centers plus 4 permuted edges in the paired orbits. Sets of moves including *M2*, *T* or *V* moves should be preferably used if pieces of a pair have to be of the same color.

Additional sets of moves may be used by *Seed Finder*, depending both on *Cube Order* and *Piece Type*:

Piece type index	Piece type	Orbit	Moves
4	Corner-Center	4x4x4: 03	4x4x4: R, MR, WR
		5x5x5: 03	5x5x5: R, NR, WR
		6x6x6: 05	6x6x6: R, NR, WR
		7x7x7: 05	7x7x7: R, NR
5	Midge-Center	5x5x5: 05	5x5x5: R, NR, MR, WR
		7x7x7: 11	7x7x7: R, NR, MR
6	Edge-Center	6x6x6: 08	6x6x6: R, NR, MR, WR
		7x7x7: 08	7x7x7: R, NR, N3R
29	Edge-Center + Edge-Center	6x6x6: 04 + 08	6x6x6: R, NR, MR, WR
		7x7x7: 04 + 08	7x7x7: R, NR, N3R
32	Corner-Center + Midge-Center	5x5x5: 03 + 05	5x5x5: R, NR, MR
		7x7x7: 05 + 08	7x7x7: R, NR, MR
33	Corner-Center + Midge-Center	5x5x5: 03 + 05	5x5x5: R, NR, MR, WR
		7x7x7: 05 + 11	7x7x7: R, NR, MR, WR
34	2 Corner-Centers + Midge-Center	5x5x5: 03 + 05	5x5x5: R, NR, MR
		7x7x7: 05 + 11	7x7x7: R, NR, MR
35	2 Corner-Centers + Midge-Center	5x5x5: 03 + 05	5x5x5: R, NR, MR, WR
		7x7x7: 05 + 11	7x7x7: R, NR, MR, WR
36	2 Midge-Centers + Corner-Center	5x5x5: 05 + 03	5x5x5: R, NR, MR
		7x7x7: 11 + 05	7x7x7: R, NR, MR
37	2 Midge-Centers + Corner-Center	5x5x5: 05 + 03	5x5x5: R, NR, MR, WR
		7x7x7: 11 + 05	7x7x7: R, NR, MR, WR

Sets of moves including *M2*, *T*, *V* or *W* moves should be used preferably if pieces of a pair should have the same color.

Permutation order

The *Permutation Order* can be any (reasonable) integer value greater than 1. Setting this parameter to *-1* will force the Finder to search for algorithms of *any* permutation order.

Permuted pieces

The number of *Permuted Pieces* can be any (reasonable) integer value greater than or equal to zero. Setting this parameter to *-1* will force the Finder to search for algorithms of *any* number of permuted pieces.

Twisted/flipped pieces

The number of *Twisted Corners* can be any integer values in set {0,2,4,6,8}. The selection is applied only if the number of permuted corners is zero.

The number of *Flipped Midges* can be any integer values in set {0,2,4,6,8,10,12}. The selection is applied only if the number of permuted edges is zero.

Strict filtering

Set *Strict Filtering* to *ON* to delete algorithms of twisted sub-patterns on each face and thus decrease the total number of seeds. Set this option to *OFF* to display a maximum of seeds.

Check cycle type

Set this option to *ON* to select algorithms of main and paired orbits of centers that have the same cycle type.

Check centers color

Set *Check Centers Color* to *ON* to apply a selective filter, based on the color of centers, to solutions found by *Seed Finder*.

Center color index

The filter width depends on parameter *Center Color Index*, which can take 5 values (0, 1, 2, 3, 4), as follows:

0: On any face:
• All centers have the same color as the initial face color, ie. centers can be twisted, provided their color doesn't change.

1: On a number of faces:
• 4 centers of the main orbit have the same color, distinct from the initial face color.
• 4 centers of the paired orbit have the same color, distinct from the initial face color.
For paired orbits, the 4 centers of the main orbit and the 4 centers of the paired orbit may not be on a same face.

2: On a number of faces:
• 4 centers of the main orbit have the same color, distinct from the initial face color.
• 2 centers of the paired orbit have the same color, distinct from the initial face color.
For paired orbits, the 4 centers of the main orbit and the 2 centers of the paired orbit may not be on a same face.

3: On a number of faces:
• 2 centers of the main orbit have the same color, distinct from the initial face color.
• 2 centers of the paired orbit have the same color, distinct from the initial face color.
For paired orbits, the 2 centers of the main orbit and the 2 centers of the paired orbit may not be on a same face.

4: On a number of faces:
• 2 opposed centers of the main orbit have the same color, distinct from the color of the 2 other opposed centers.
Corners, midges and edges are not checked for color match, only centers.

Search space

When using Seed Finder, the set of moves in which variable X takes values is usually restricted to R moves only and does not include inverses. For example, the restricted set of X moves of corner-centers is simply {R, R2, NR, NR2}. This actually minimizes the search space size and greatly reduce computing time. Because seeds found this way can be further inverted, shifted, transformed by symmetry and conjugated in Algorithm Finder, the restricted set of X moves is generally sufficient to cover all usual cases.

As there may be cases where the search space size must be maximized, the *Maximize Search Space* selector can be set to ON in order to expand the set of X moves. Expanding the set of X moves to its maximum size will indeed maximize the search space size, at the expense of a sharp increase in computing time though. In our example, the set of X moves of corner-centers is now {R, R', R2, NR, NR', NR2, … B, B', B2, NB, NB', NB2}, which is 9x larger than the restricted set.

For example, by checking this option and setting *Permutation Order* to *-1* and *Permuted Pieces* to *12*, the Finder will search for algorithms of any permutation order and 12 permuted pieces. Conversely, setting *Permutation Order* to *6* and *Permuted Pieces* to *-1*, will force the Finder to search for algorithms of permutation order 6 and any number of permuted pieces.

Algorithm Finder example 1

Niklas commutator applied to 3-cycles of corners (7x7x7 cube)

	Project
Title, Description	Niklas commutator applied to 3-cycles of corners (7x7x7 cube)
	Basic Settings
Finder Type, Cube Order	Algorithm Finder; 7x7 V-Cube 7
Generators	Generators: 6 Faces; 90° and 180° twists
Centers	Super Cube Mode: ON; Check True Centers Orientation: OFF
Setup Moves, Scalability	Setup Moves: 0; Check Algorithm Scalability: OFF
Template Symmetry	0: none
	Search Template(s)
Algorithm Template(s)	[X, Y Z Y']
Insert Algorithm Template(s)	
	Search Mask
[Location Mask]	[inactive]
Generator Algorithm	[R, U L U']
User-Defined Moves	
[True Center Orientation]	A: 0°; E: 0°; I: 0°; M: 0°; Q: 0°; U: 0°
	Algorithm Finder Settings
Template(s)	Insert Template: OFF; Shift Template: OFF; Invert Template: OFF
Symmetry	Rotation Symmetry: OFF; Reflection Symmetry: OFF
Accuracy	Find Nearest Position: OFF
Move Reduction	Algorithm Checker Mode: OFF
Solver Speed	Booster Mode: OFF

Algorithm Finder example 1
Niklas commutator applied to 3-cycles of corners (7x7x7 cube)

Search template: [X, Y Z Y']
Search mask: [R, U L U']

<u>1 algorithm found:</u>
R U L U' R' U L' U' (8 btm; Solved Stickers: 9; Order: 3)

Algorithm Finder example 2

Hardest distance-20 position (3x3x3 cube)

	Project
Title, Description	Hardest distance-20 position (3x3x3 cube)
	Basic Settings
Finder Type, Cube Order	Algorithm Finder; 3x3 Rubik's Cube
Generators	Generators: 6 Faces; 90° and 180° twists
Centers	Super Cube Mode: OFF; Check True Centers Orientation: OFF
Setup Moves, Scalability	Setup Moves: 1; Check Algorithm Scalability: OFF
Template Symmetry	2: F-B axis +180°, Cube Rotation CF2
	Search Template(s)
Algorithm Template(s)	[R' L, D2 U' F' L D U2 F']
Insert Algorithm Template(s)	
	Search Mask
[Location Mask]	[inactive]
Generator Algorithm	F U' F2 D' B U R' F' L D' R' U' L U B' D2 R' F U2 D2
User-Defined Moves	
[True Center Orientation]	A: 0°; E: 0°; I: 0°; M: 0°; Q: 0°; U: 0°
	Algorithm Finder Settings
Template(s)	Insert Template: OFF; Shift Template: ON; Invert Template: ON
Symmetry	Rotation Symmetry: ON; Reflection Symmetry: ON
Accuracy	Find Nearest Position: OFF
Move Reduction	Algorithm Checker Mode: OFF
Solver Speed	Booster Mode: OFF

Algorithm Finder example 2
Hardest distance-20 position (3x3x3 cube)

Search template: [R' L, D2 U' F' L D U2 F']
Search mask: F U' F2 D' B U R' F' L D' R' U' L U B' D2 R' F U2 D2

8 algorithms found:
F' L D2 U' F' L D U2 F' SR' F D2 U' R' F D U2 R' F (19 btm; Solved Stickers: 48; Order: 6)
B' U L2 R' F' U L R2 F' SU F L2 R' D' F L R2 D' B (19 btm; Solved Stickers: 48; Order: 6)
B' R D2 U' B' R D U2 B' SR B D2 U' L' B D U2 L' B (19 btm; Solved Stickers: 48; Order: 6)
F' U R2 L' B' U R L2 B' SU B R2 L' D' B R L2 D' F (19 btm; Solved Stickers: 48; Order: 6)
B R' U2 D B R' U' D2 B SR' B' U2 D L B' U' D2 L B' (19 btm; Solved Stickers: 48; Order: 6)
F D' R2 L B D' R' L2 B SU B' R2 L U B' R' L2 U F' (19 btm; Solved Stickers: 48; Order: 6)
F L' U2 D F L' U' D2 F SR F' U2 D R F' U' D2 R F' (19 btm; Solved Stickers: 48; Order: 6)
B D' L2 R F D' L' R2 F SU F' L2 R U F' L' R2 U B' (19 btm; Solved Stickers: 48; Order: 6)

Algorithm Finder example 3

Semi-commutator applied to 2 3-cycles of corner-centers (7x7x7 cube)

	Project
Title, Description	Semi-commutator applied to 2 3-cycles of corner-centers (7x7x7 cube)
	Basic Settings
Finder Type, Cube Order	Algorithm Finder; 7x7 V-Cube 7
Generators	Generators: 6 Faces; 90° and 180° twists
Centers	Super Cube Mode: ON; Check True Centers Orientation: OFF
Setup Moves, Scalability	Setup Moves: 1; Check Algorithm Scalability: OFF
Template Symmetry	0: none
	Search Template(s)
Algorithm Template(s)	]NR NU, L NU' L NU[
Insert Algorithm Template(s)	
	Search Mask
[Location Mask]	[inactive]
Generator Algorithm	[NR, NU' L2 NU] [NR' D NR, NU]
User-Defined Moves	
[True Center Orientation]	A: 0°; E: 0°; I: 0°; M: 0°; Q: 0°; U: 0°
	Algorithm Finder Settings
Template(s)	Insert Template: OFF; Shift Template: OFF; Invert Template: OFF
Symmetry	Rotation Symmetry: OFF; Reflection Symmetry: OFF
Accuracy	Find Nearest Position: OFF
Move Reduction	Algorithm Checker Mode: OFF
Solver Speed	Booster Mode: OFF

Algorithm Finder example 3
Semi-commutator applied to 2 3-cycles of corner-centers (7x7x7 cube)

Search template:]NR NU, L NU' L NU[
Search mask: [NR, NU' L2 NU] [NR' D NR, NU]

1 algorithm found:
D NR NU L NU' L NU NR' NU' L' NU L' NU' D' (14 btm; Solved Stickers: 6; Order: 3)

Seed Finder example 1

Corner-centers 17-cycle – 5 variables – 4 faces – 90° twists (7x7x7 cube)

	Project
Title, Description	Corner-centers 17-cycle – 5 variables – 4 faces – 90° twists (7x7x7 cube)
	Basic Settings
Finder Type, Cube Order	Seed Finder; 7x7 V-Cube 7
Generators	Generators: 4 Faces; 90° twists only
Centers	Super Cube Mode: ON; Check True Centers Orientation: OFF
Setup Moves, Scalability	Setup Moves: 0; Check Algorithm Scalability: OFF
Template Symmetry	0: none
	Search Template(s)
Algorithm Template(s)	[X Y, Z P Q]
Insert Algorithm Template(s)	
	Seed Finder Settings
Template(s)	Insert Template: OFF; Shift Template: OFF
Piece Type	4: Corner-Center [05]: R, NR
Permutation Order	17
Permuted Pieces	17
Twisted/Flipped Pieces	Twisted Corners: 0; Flipped Midges: 0
Centers	Strict Filtering: ON; Check Cycle Type: OFF; Check Centers Color: OFF
Center Color Index	0: On all faces: All centers have the same color
Search Space	Maximize Search Space: OFF

Seed Finder example 1
Corner-centers 17-cycle – 5 variables – 4 faces – 90° twists (7x7x7 cube)

Search template: [X Y, Z P Q]

16 algorithms found:
NR NL NU R' NF NL' NR' NF' R NU' (10 btm; Solved Stickers: 17; Order: 17)
NR NL NU L NF NL' NR' NF' L' NU' (10 btm; Solved Stickers: 17; Order: 17)
NR NL NU' R' NF' NL' NR' NF R NU (10 btm; Solved Stickers: 17; Order: 17)
NR NL NU' L NF' NL' NR' NF L' NU (10 btm; Solved Stickers: 17; Order: 17)
NR NL NF R NU NL' NR' NU' R' NF' (10 btm; Solved Stickers: 17; Order: 17)
NR NL NF L' NU NL' NR' NU' L NF' (10 btm; Solved Stickers: 17; Order: 17)
NR NL NF' R NU' NL' NR' NU R' NF (10 btm; Solved Stickers: 17; Order: 17)
NR NL NF' L' NU' NL' NR' NU L NF (10 btm; Solved Stickers: 17; Order: 17)
NR NL' NU R' NF NL NR' NF' R NU' (10 btm; Solved Stickers: 17; Order: 17)
NR NL' NU L NF NL NR' NF' L' NU' (10 btm; Solved Stickers: 17; Order: 17)
NR NL' NU' R' NF' NL NR' NF R NU (10 btm; Solved Stickers: 17; Order: 17)
NR NL' NU' L NF' NL NR' NF L' NU (10 btm; Solved Stickers: 17; Order: 17)
NR NL' NF R NU NL NR' NU' R' NF' (10 btm; Solved Stickers: 17; Order: 17)
NR NL' NF L' NU NL NR' NU' L NF' (10 btm; Solved Stickers: 17; Order: 17)
NR NL' NF' R NU' NL NR' NU R' NF (10 btm; Solved Stickers: 17; Order: 17)
NR NL' NF' L' NU' NL NR' NU L NF (10 btm; Solved Stickers: 17; Order: 17)

Seed Finder example 2

True centers – batch mode – 2 3-cycles – 4 variables (7x7x7 cube)

	Project
Title, Description	True centers – batch mode – 2 3-cycles – 4 variables (7x7x7 cube)
	Basic Settings
Finder Type, Cube Order	Seed Finder; 7x7 V-Cube 7
Generators	Generators: 6 Faces; 90° and 180° twists
Centers	Super Cube Mode: ON; Check True Centers Orientation: OFF
Setup Moves, Scalability	Setup Moves: 0; Check Algorithm Scalability: OFF
Template Symmetry	0: none
	Search Template(s)
Algorithm Template(s)	[X, Y] X Y X Y X Y X' Z P Z' X Y Z P X' Y' [X, Y Z[
Insert Algorithm Template(s)	
	Seed Finder Settings
Template(s)	Insert Template: OFF; Shift Template: OFF
Piece Type	0: True Center [00]: R, MR
Permutation Order	3
Permuted Pieces	6
Twisted/Flipped Pieces	Twisted Corners: 0; Flipped Midges: 0
Centers	Strict Filtering: ON; Check Cycle Type: OFF; Check Centers Color: OFF
Center Color Index	0: On all faces: All centers have the same color
Search Space	Maximize Search Space: OFF

Seed Finder example 2
True centers – batch mode – 2 3-cycles – 4 variables (7x7x7 cube)

Search templates:
[X, Y]
X Y X Y
X Y X' Z P Z'
X Y Z P X' Y'
[X, Y Z[
[X, Y Z]
X Y Z X Y Z
X Y X Y X Y
[X, Y Z P[
[X, Y Z P]
[X Y, Z P[
[X Y, Z P]
[X, Y] [Z, P]
X Y Z P X Y Z P
X Y X Y X Y X Y

28 algorithms found:
MR MU MR' MU' (4 btm; Solved Stickers: 6; Order: 3)
MR MU' MR' MU (4 btm; Solved Stickers: 6; Order: 3)
MR MU MR MU' MR2 (5 btm; Solved Stickers: 6; Order: 3)
MR MU' MR MU MR2 (5 btm; Solved Stickers: 6; Order: 3)
MR MU MR' MF MU' MF' (6 btm; Solved Stickers: 6; Order: 3)
MR MU MR' MF' MU' MF (6 btm; Solved Stickers: 6; Order: 3)
MR MF' MR' MU' MF MU (6 btm; Solved Stickers: 6; Order: 3)
MR MU' MR' MF MU MF' (6 btm; Solved Stickers: 6; Order: 3)
MR MF MR' MU MF' MU' (6 btm; Solved Stickers: 6; Order: 3)
MR MF MR' MU' MF' MU (6 btm; Solved Stickers: 6; Order: 3)
MR R MU' MR' MU R' (6 btm; Solved Stickers: 6; Order: 3)
MR R MF MR' MF' R' (6 btm; Solved Stickers: 6; Order: 3)
MR MU MF MR2 MU' MF' MR (7 btm; Solved Stickers: 6; Order: 3)
MR MU MF' MR2 MU' MF MR (7 btm; Solved Stickers: 6; Order: 3)
MR MF MU MR2 MF' MU' MR (7 btm; Solved Stickers: 6; Order: 3)
MR MU' MF MR2 MU MF' MR (7 btm; Solved Stickers: 6; Order: 3)
MR MF' MU' MR2 MF MU MR (7 btm; Solved Stickers: 6; Order: 3)
MR MF MU' MR2 MF' MU MR (7 btm; Solved Stickers: 6; Order: 3)
MR MF' MU MR2 MF MU' MR (7 btm; Solved Stickers: 6; Order: 3)
MR MU MF MU2 MR' MF' MU (7 btm; Solved Stickers: 6; Order: 3)
MR MF MR MF MR' MF' MR' MF' (8 btm; Solved Stickers: 6; Order: 3)
MR MF MR MF' MR' MF' MR' MF (8 btm; Solved Stickers: 6; Order: 3)
MR MF MR' MF' MR' MF' MR MF (8 btm; Solved Stickers: 6; Order: 3)
MR MF MR' MF MR' MF' MR MF' (8 btm; Solved Stickers: 6; Order: 3)
MR MU MR MU MR' MU' MR' MU' (8 btm; Solved Stickers: 6; Order: 3)
MR MU MR MU' MR' MU' MR' MU (8 btm; Solved Stickers: 6; Order: 3)
MR MU MR' MU MR' MU' MR MU' (8 btm; Solved Stickers: 6; Order: 3)
MR MU MR' MU' MR' MU' MR MU (8 btm; Solved Stickers: 6; Order: 3)

Seed Finder example 3

Corner-centers – generalized commutator – 2 3-cycles – 5 variables (7x7x7 cube)

	Project
Title, Description	Corner-centers - generalized commutator - 2 3-cycles - 5 variables (7x7x7 cube)
	Basic Settings
Finder Type, Cube Order	Seed Finder; 7x7 V-Cube 7
Generators	Generators: 6 Faces; 90° and 180° twists
Centers	Super Cube Mode: ON; Check True Centers Orientation: OFF
Setup Moves, Scalability	Setup Moves: 0; Check Algorithm Scalability: OFF
Template Symmetry	0: none
	Search Template(s)
Algorithm Template(s)	[X,Y] [Y,Z] [Z,P] [P,Q]
Insert Algorithm Template(s)	
	Seed Finder Settings
Template(s)	Insert Template: OFF; Shift Template: OFF
Piece Type	4: Corner-Center [05]: R, NR
Permutation Order	3
Permuted Pieces	6
Twisted/Flipped Pieces	Twisted Corners: 0; Flipped Midges: 0
Centers	Strict Filtering: ON; Check Cycle Type: OFF; Check Centers Color: OFF
Center Color Index	0: On all faces: All centers have the same color
Search Space	Maximize Search Space: OFF

Seed Finder example 3
Corner-centers – generalized commutator – 2 3-cycles – 5 variables (7x7x7 cube)

Search template: [X,Y] [Y,Z] [Z,P] [P,Q]

<u>40 algorithms found:</u>
NR NU NR' NU' (4 btm; Solved Stickers: 6; Order: 3)
NR NU' NR' NU (4 btm; Solved Stickers: 6; Order: 3)
NR NU2 NR' NU2 (4 btm; Solved Stickers: 6; Order: 3)
NR2 NU2 NR2 NU2 (4 btm; Solved Stickers: 6; Order: 3)
NR NU NR2 NU' NR (5 btm; Solved Stickers: 6; Order: 3)
NR NU NR NU' NR2 (5 btm; Solved Stickers: 6; Order: 3)
NR NU' NR NU NR2 (5 btm; Solved Stickers: 6; Order: 3)
NR NU2 NR2 NU2 NR (5 btm; Solved Stickers: 6; Order: 3)
NR NU2 NR NU2 NR2 (5 btm; Solved Stickers: 6; Order: 3)
NR NU NR' NF' NU NF NU2 (7 btm; Solved Stickers: 6; Order: 3)
NR NU NR NU2 NR' NU NR' (7 btm; Solved Stickers: 6; Order: 3)
NR NU NR NU NR' NU2 NR' (7 btm; Solved Stickers: 6; Order: 3)
NR NU NR' NU2 NR NU NR' (7 btm; Solved Stickers: 6; Order: 3)
NR NU NR' NU NR NU2 NR' (7 btm; Solved Stickers: 6; Order: 3)
NR NU' NR NU' NR' NU2 NR' (7 btm; Solved Stickers: 6; Order: 3)
NR NU' NR' NU2 NF' NU' NF (7 btm; Solved Stickers: 6; Order: 3)
NR NU' NR' NU' NR NU2 NR' (7 btm; Solved Stickers: 6; Order: 3)
NR NU NR2 NU2 NR2 NU NR' (7 btm; Solved Stickers: 6; Order: 3)
NR NU NR2 NU NR2 NU2 NR' (7 btm; Solved Stickers: 6; Order: 3)
NR NU' NR2 NU' NR2 NU2 NR' (7 btm; Solved Stickers: 6; Order: 3)
NR2 NU NR' NU2 NR NU NR2 (7 btm; Solved Stickers: 6; Order: 3)
NR2 NU NR' NU NR NU2 NR2 (7 btm; Solved Stickers: 6; Order: 3)
NR2 NU' NR' NU' NR NU2 NR2 (7 btm; Solved Stickers: 6; Order: 3)
NR2 NU' NR2 NU2 NR2 NU' NR2 (7 btm; Solved Stickers: 6; Order: 3)
NR2 NU2 NR2 NU' NR2 NU' NR2 (7 btm; Solved Stickers: 6; Order: 3)
NR NU NR NU' NB' NR NB NR (8 btm; Solved Stickers: 6; Order: 3)
NR2 NU2 NR2 NL2 NU2 ND2 NL2 ND2 (8 btm; Solved Stickers: 6; Order: 3)
NR NU' NR' NF' NU NL NF' NL' NF2 (9 btm; Solved Stickers: 6; Order: 3)
NR NU2 NR2 NU2 NR NL NF2 NL' NF2 (9 btm; Solved Stickers: 6; Order: 3)
NR NU2 NR NU2 NR2 NL NF2 NL' NF2 (9 btm; Solved Stickers: 6; Order: 3)
NR NU2 NR' NL NU2 NF2 NL NF2 NL2 (9 btm; Solved Stickers: 6; Order: 3)
NR2 NU2 NR' NU2 NR' NL2 ND2 NL2 ND2 (9 btm; Solved Stickers: 6; Order: 3)
NR NU2 NR' NL NU2 NF2 NL' NR' NF2 NR (10 btm; Solved Stickers: 6; Order: 3)
NR2 NU NR2 NU' ND NR2 ND' NU NR2 NU' (10 btm; Solved Stickers: 6; Order: 3)
NR2 NU NR2 NU' ND NB2 ND' NU' NB2 NU (10 btm; Solved Stickers: 6; Order: 3)
NR2 NF NR2 NF' NB' NR2 NB NF NR2 NF' (10 btm; Solved Stickers: 6; Order: 3)
NR2 NU NR2 NU' ND2 NR2 ND2 NU NR2 NU' (10 btm; Solved Stickers: 6; Order: 3)
NR NU2 NR' NL2 NU2 ND2 NL2 NR2 ND2 NR2 (10 btm; Solved Stickers: 6; Order: 3)
NR2 NF2 NR2 NF2 NB NR2 NB' NF2 NR2 NF2 (10 btm; Solved Stickers: 6; Order: 3)
NR2 NF2 NR2 NF2 NB2 NR2 NB2 NF2 NR2 NF2 (10 btm; Solved Stickers: 6; Order: 3)