Orbiter User's Guide

Anton Betten

August 21, 2018

Abstract

We discuss how to use the program system Orbiter for the classification of combinatorial objects.

1 Introduction

This is a User's guide to the program package Orbiter [1] [2] for the classification of combinatorial objects. Orbiter is a library of C++ classes for algebraic combinatorial objects. It does not have a user interface, but it comes with several pre-programmed applications which can be invoked using the command line or using makefiles. This means that there are basically two ways in which Orbiter can be used. The first way is recommended for novice users. Those kinds of users would use Orbiter applications through unix command lines, possibly using makefiles to simplify the typing of commands. The second type of user are more experienced programmers. They would build their own applications using the C++ library. In both cases, the orbiter application is developed using the orbiter C++ library, which in turn uses the C++ standard library (cf. Fig. 1). The Orbiter library is layered. There are five components, one building on top of the other. This is shown in Fig. 2.

2 Installing Orbiter

Orbiter is distributed through github (`https://github.com`). Search for "abetten/orbiter" or go directly to

`https://github.com/abetten/orbiter`

Once there, find the green button called "Clone or download". The button offers two options: "Open in Desktop" and "Download ZIP" Choose one of them to download Orbiter. Orbiter is compiled with makefiles. Some system specific comments are in order:

- For Microsoft Windows users, it is recommended to install Orbiter using `cygwin` [6]. Cygwin is a Unix-like environment and command-line interface for Microsoft Windows.
- Macintosh users need to install Xcode (search for xcode in the app store). Xcode is an integrated development environment (IDE) for MacOS. It comes with command line

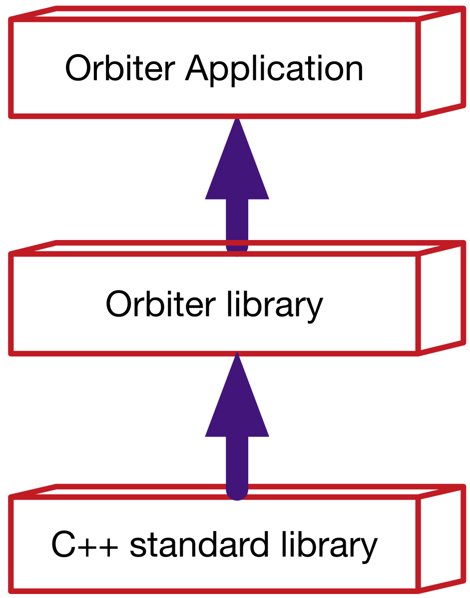


Figure 1: The orbiter model

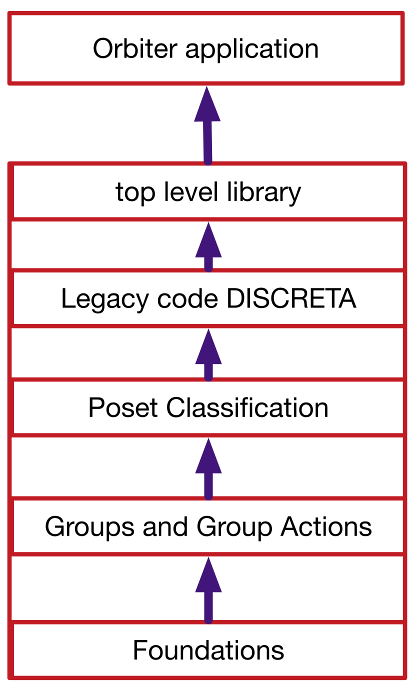


Figure 2: The layers of the orbiter library

tools for software development. In order to compile Orbiter, the command line tools are required. There is no need to use the integrated development environment though.

- For Linux users, the compiler environment (for instance Gnu C++) needs to be installed.

We will use a terminal window (console) to install Orbiter. Assuming that we have the various compiler tools avalable, the installation proceeds as follows. The following commands are typed into the terminal window.

Enter the directory `ORBITER/src` and type

```
make
```

This should create a lot of text output to the console. Assuming that the command executes without errors, orbiter is now ready. The specific purpose of this make command is to compile all C++ source code into object files, to bind the object files together into one library, and to link the orbiter executables. The C++ cource code is in files with extension .cpp. There are additional filed called header files with extension .h. The header files are needed to compile the .cpp files into .o files. This has to do that one source file needs to know a little bit what goes on in the other source files. The object files are corresponding files with extension .o.

$$\Big(\text{XXX.cpp},\ \text{orbiter.h}\Big) \mapsto \text{XXX.o} \ \mapsto\ \text{liborbiter.a}.$$

Here, XXX stands for all files in the /src/lib subtree, and the map XXX.cpp $\mapsto$ XXX.o is one-to-one. The map XXX.o $\mapsto$ liborbiter.a is many-to-one. Once the library has been compiled, the application executables are compiled:

$$\left.\begin{array}{c}\text{YYY.cpp} \ \mapsto\ \text{YYY.o} \\ \text{liborbiter.a} \\ \text{standard libraries (libc++ etc.)}\end{array}\right\} \mapsto\ \text{YYY.out}$$

This time, the source files YYY reside in the src/apps branch. The file YYY.out is the executable. Executables are programs which can be called. They are recognizable by the x flag in the directory list). The executable contains the actual program to do the work. It will be called for instance through the command line. This make command has to be executed only once. One can recognize the fact that make has been successful by verifying the presence of files with extension .o on the src subdirectories. Also, various files with the extension .out have been created in the subdirectories of src/apps. The make command descends into all subdirectories of src and performs make. This means that make has to be done only once and only in the directory src. Make takes care of all the work that needs to be done across all subdirectories in the src/ tree.

The directory structure of the Orbiter src/ tree is this:

```

    ./src/
```

```
3     ./src/apps/
4     ./src/apps/arcs/
5     ./src/apps/blt/
6     ./src/apps/codes/
7     ./src/apps/combinatorics/
8     ./src/apps/graph/
9     ./src/apps/groups/
10    ./src/apps/ovoid/
11    ./src/apps/projective_space/
12    ./src/apps/regular_ls/
13    ./src/apps/spread/
14    ./src/apps/subspace_orbits/
15    ./src/apps/surfaces/
16    ./src/apps/tdo/
17    ./src/apps/tools/
18    ./src/lib/
19    ./src/lib/DISCRETA/
20    ./src/lib/foundations/
21    ./src/lib/foundations/algebra_and_number_theory/
22    ./src/lib/foundations/coding_theory/
23    ./src/lib/foundations/combinatorics/
24    ./src/lib/foundations/data_structures/
25    ./src/lib/foundations/geometry/
26    ./src/lib/foundations/geometry/DATA/
27    ./src/lib/foundations/globals/
28    ./src/lib/foundations/graph_theory/
29    ./src/lib/foundations/graph_theory_nauty/
30    ./src/lib/foundations/graphics/
31    ./src/lib/foundations/io_and_os/
32    ./src/lib/foundations/solvers/
33    ./src/lib/foundations/statistics/
34    ./src/lib/groups_and_group_actions/
35    ./src/lib/groups_and_group_actions/data_structures/
36    ./src/lib/groups_and_group_actions/group_actions/
37    ./src/lib/groups_and_group_actions/group_theory/
38    ./src/lib/groups_and_group_actions/induced_actions/
39    ./src/lib/poset_classification/
40    ./src/lib/poset_classification/classify/
41    ./src/lib/poset_classification/other/
42    ./src/lib/poset_classification/set_stabilizer/
43    ./src/lib/poset_classification/snakes_and_ladders/
44    ./src/lib/top_level/
45    ./src/lib/top_level/algebra_and_number_theory/
46    ./src/lib/top_level/geometry/
47    ./src/lib/top_level/isomorph/
```

```
48      ./src/lib/top_level/orbits/
49      ./src/lib/top_level/solver/
50
```

In lines 3-17, the Orbiter applications are can be seen in various subdirectories of src/apps. In lines 18-49, the Orbiter library is located in subdirectories of src/lib. A more extensive list of the source tree can be found in Appendix C. The main header file is

src/lib/orbiter.h,

on line 435 (in the appendix). This header file is used by all sources which want to use the orbiter library.

In order to test orbiter, go to the subdirectory `ORBITER/examples`. There are several subdirectories containing test problems. The test problems are calls to orbiter, asking it so solve small problems. For instance, the subdirectory groups contains some problems related to groups. Once in the directory `ORBITER/examples/groups`, issue the command `make` to run the first problem. To see what kind of problems are available, open the file `makefile` and see what targets are defined. You can type `make target` where `target` is any of the targets in the makefile to run that particular problem. Most problems will create a lot of output to the console.

3 Combinatorial Objects

Orbiter is a software tool devoted to the study of combinatorial objects. So, what is a combinatorial object? It is not clear if there is an accepted definition for combinatorial object. The approach taken here is based on the notion of groups acting on sets. Let G be a group acting on a set X. The elements of X fall into orbits under G. For $x, y \in X$, the relation $x \sim_G y$ is defined whenere there exists an element $g \in G$ with $xg = y$. The equivalence classes of this relation are the orbits of G on X. We say that a combinatorial object is an orbit of a group G acting on a set X. Combinatorial objects are almost always given by means of representing elements. That is, an element $xinX$ is given to represent the orbit of x under G. If for two elements $x, y \in G$ we have $x \sim_G y$ then x and y are called isomorphic. The type of a combinatorial object is the group action (X, G). The classification problem for combinatorial objects of type (X, G) is the problem of determining the orbits of G on X. Usually, this is understood as the problem of finding a transversal of the orbits of G on X. That is, a set $T \subseteq X$ is produced such that T intersects each orbit of G on X in exactly one element.

4 Theoretical Background

Combinatorial objects are orbits of group actions. Hence classifying combinatorial objects is equivalent to computing the orbits of a group acting on a set. Let G be a group and let X be a finite set. For two elements x, y of X, write $x \sim y$ if x and y belong to the same G-orbit. A transversal for the orbits of G on X is a sequence $T = (t_1, \ldots, t_n)$ for some integer n such that for every $x \in X$ there is exactly one t_i with $x \sim t_i$. The classification problem for the orbits of G on X is computing a transversal for the orbits. The recognition problem for the group G acting on the set X is the following: Given an element $x \in X$, determine the element $t_i \in T$ auch that $x \sim t_i$. Here, T is the transversal that was computed previously. The constructive recognition problem for the group G acting on the set X is the following: Given an element $x \in X$, determine the element $t_i \in T$ auch that $x \sim t_i$ and in adition find a group element $g \in G$ such that $xg = t_i$. The main challenge is to find efficient algorithms for the classification problem, the recognition problem and the constructive recognition problem.

Standard orbit algorithms are related to the idea of Schreier trees. The complexity of these algorithms is linear in the size of the orbit. This is fine for small problems, but Orbiter is designed to handle cases that require faster algorithms. For many combinatorial objects, posets can be used to aid the classification task. The group induces an action on the poset and the orbits of the group on the poset are computed. Often times, the orbits on the objects at a specific layer in the poset correspond to the combinatorial objects that are desired. For the theory of group action on posets, see [16]. For a description of poset based classification, see [3]. With poset classification, the algorithm often proceeds much faster than the ordinary orbit algorithm based on Schreier trees. This is because the orbits are never fully looked at. Poset classification looks at a small fraction of the elements only in order to compute a transversal for the orbits.
Poset classification is the process by which the orbits of a group G on a poset $\mathcal{P}$ are classified. Additional information is computed also. This additional information described how the orbits are related. An early application of this can be seen in [5], where the orbits of the Mathieu group acting on the set of subsets of the set $\{1, \ldots, 24\}$ are computed and a diagram is presented which contains detailed information about how the orbits are related in the sense of [16].

In order to use Orbiter, a group G is defined. Then, an action of this group on a set X is constructed. Using the induced action on k-subsets or k-subspaces, a poset action on a lattice $\mathcal{L}$ is defined. Then, a subposet $\mathcal{P}$ of the lattice is selected, and the orbit of the group G on this poset $\mathcal{P}$ are computed. The orbit representatives and the stabilizer groups are listed and stored. We will describe these steps next.

The algorithm described in [3] is based on earlier work of Schmalz [20]. Schmalz was concerned with computing double coset representatives in certain groups. The groups that Schmalz considered turned out to have good "ladders" of subgroups. A ladder of subgroups is a sequences of subgroups $G_1, G_2, \ldots, G_r$ where two consecutive groups are related. This means that for $i = 1, \ldots, r-1$ we have $G_i \le G_{i+1}$ or $G_i \ge G_{i+1}$. A subgroup ladder is good if the indices of consecutive terms are small. The problem of computing orbits on k-subsets and orbits on k-dimensional subspaces can be formulated equivalently as a problem of computing double cosets in either the symmetric group or the full semilinear matrix groups. For these groups, good subgroup ladders exist. Because of the use of subgroup ladders, Schmalz coined the term "Leiterspiel" for his algorithm. Perhaps the closest translation into English would be "Snakes and Ladders." Much of Orbiter is based on the Schmalz algorithm.

There are other algorithms which can be used to classify combinatorial objects. Most notably there is the method of canonical ancestors proposed by McKay [14], which may be seen as part of the family of algorithms which are called "orderly generation", and which have been discovered and refined many times, see [17], [7], [12], [13], [18], [8].

The Schmalz algorithm differs from orderly generation in some important ways. First, while orderly generation proceeds depth-first, the algorithm of Schmalz proceeds in a breadth-first manner. While orderly generation only keeps the current object in memory, the algorithm of Schmalz builds up the whole tree of orbits, storing quite a lot of information. While the orderly generation algorithm relies on a backtrack procedure to compute canonical forms, the algorithm of Schmalz does not backtrack. It uses the data structure about previously computed orbits to perform recognition. The differences in the two families are sufficiently large to expect different behavior of the algorithms on different kinds of problems.

All groups in Orbiter are finite permutation groups. However, we distinguish between groups of linear or affine type and ordinary permutation groups. Groups of linear type are groups that act linearly or affinely (possibly semilinearly) on the elements of a vector space. This allows us to encode group elements efficiently using matrices. For semilinear actions, we store a field automorphism. For affine actions, we store a translation vector. All other groups need their elements stored as permutations. Memory issues are very important in Orbiter, so finding the most efficient way to encode a group element matters. At times, a representation of matrices over finite fields as bitvectors is utilized, reducing the storage requirement even further.

All groups in Orbiter are permutation group. For every group, a fixed permutation representation is chosen from the beginning. Later, new groups actions can be defined. It is important to know that any group always has a default permutation representation, independent of how the group elements are represented. For projective linear groups, a labeling of the points of $\mathrm{PG}(n-1,q)$ is used. For general linear and affine groups, a labeling of the vectors in $\mathbb{F}_q^n$ is used. For orthogonal groups, a labeling of the points of $Q^\epsilon(n-1,q)$ is used. For permutation groups, the set $\{0,\ldots,d-1\}$ is used, for some d.

5 Orbiter Applications

This section describes some of the available Orbiter applications and how they can be used. Mainly, Orbiter applications rely on the command line to being told what to do. Options are passed along using certain command line keys, which must be known. Suppose we want to create one of the orthogonal groups over a finite field. The orbiter application to do so is called `orthogonal_group.out`. Since an orthogonal group $O^\epsilon(d,q)$ involves three parameters, the Orbiter application requires us to use three options. By default, most Orbiter parameters are integers. They are `-epsilon` $\langle\epsilon\rangle$, `-d` $\langle d\rangle$ and `-q` $\langle q\rangle$. For instance, the command

```
orthogonal_group.out -v 2 -epsilon -1 -d 6 -q 2
```

creates the group $O^-(6,2)$ of order 51840. The parameter `-v` $\langle v \rangle$ specifies the verbose level. Higher values of v mean more text output. This group $O^-(6,2)$ acts on the 27 points of the $Q^-(5,2)$ quadric. The quadric is given by the quadratic form

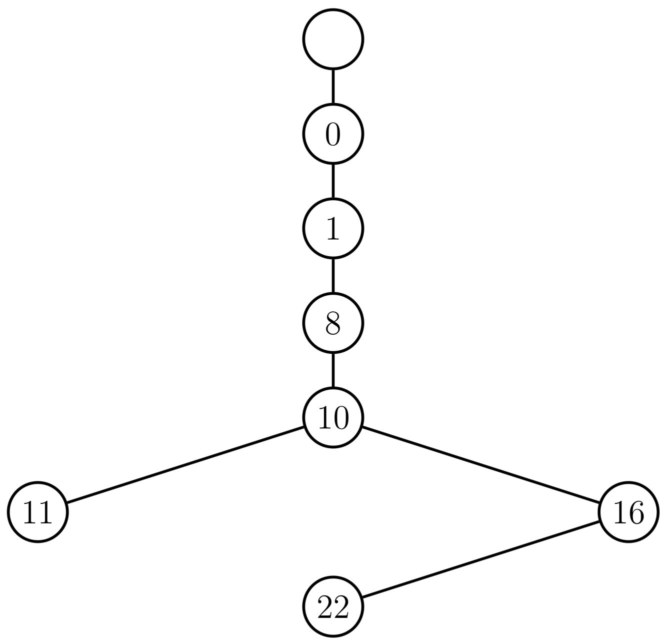
$$x_0x_1 + x_2x_3 + x_4^2 + x_4x_5 + x_5^2 = 0.$$

Generators for the group as 6×6 matrices over $\mathbb{F}_2$ are listed in the output, as are corresponding permutations of the 27 points. Finally, the generators are written in machine readable compact form.

The next problem we wish to consider is classifying ovoids in $\Omega^-(5,2)$ under the group $O^-(6,2)$ from before. A partial ovoid is a set $S = \{P_1, \ldots, P_n\}$ of points $P_i = \langle v_i \rangle$ such that $\beta(v_i, v_j) \neq 0$ for all $i \neq j$. The partial ovoids in $\Omega^-(5,2)$ can be classified using the orbiter command

```
ovoid.out -v 5 -epsilon -1 -n 5 -q 2 -draw_poset -embedded -W
```

This produces the following diagram:



which shows that the set $\{0, 1, 8, 10, 16, 22\}$ is the only partial ovoid in $\Omega^-(5,2)$ up to equivalence. The numbers stand for points in $\Omega^-(5,2)$ as listed in the output of the `orthogonal_group.out` command, mentioned earlier. They are

$$\begin{aligned}
0 &= (1,0,0,0,0,0),\\
1 &= (0,1,0,0,0,0),\\
8 &= (1,1,1,1,0,0),\\
10 &= (1,1,1,0,1,0),\\
16 &= (1,1,1,0,0,1),\\
22 &= (1,1,1,0,1,1).
\end{aligned}$$

6 Orbiter Design Principles

In order to fully utilize Orbiter, it is necessary to understand some of the design principles. There are several topics that are of great importance:

1. How groups and group actions are represented on a computer;
2. Which groups are available;
3. What kind of induced group actions are available;
4. How user-defined posets are realized;
5. How bijections are used in order to reduce the memory complexity of combinatorial objects;
6. How Orbiter trades time versus memory.

6.1 Using Bijections

Combinatorial objects are often built up from smaller objects. Hence we distinguish between atomic combinatorial objects and compound objects. Using bijections to integers can be helpful in two ways. First it reduces the the memory complexity of the object, thereby simplifying the data structures for atomic and compound objects. Second, it simplifies the handling of group actions on combinatorial objects. A group action can be realized as a function which takes a group element and a combinatorial object to yest another combinatorial object. If the combinatorial objects are mapped to a set of integers, then the group action function can take as input an integer and a group element. The function produces as output yet another integer, namely the combinatorial object that is the image. By using integers, the function to perform the mapping can have a standardized calling syntax for many different group actions. This is important for a sytem like Orbiter where there are many different group actions which all utilize the same scheme for function calling.

Orbiter uses standard bijections to map atomic combinatorial objects to integer intervals. For instance, the set of points of a finite dimensional projective or affine geometry over a finite field can be stored as integers rather than as vectors. Likewise, the set of subspaces of a projective geometry can be coded as integers. The points and lines of an orthogonal geometry can be coded. The points on a Hermitan geometry can be coded. The elements of a finite group can be coded. The set of k-subsets of a set can be coded. Ordered and unordered pairs of a set can be coded.

7 Orbiter History and Design Issues

Orbiter is based on an earlier system called DISCRETA [4]. DISCRETA is a system to construct combinatorial objects called t-designs using an assumed symmetry group. DISCRETA in turn was influenced by two systems: SYMMETRICA and DCC. SYMMETRICA (see [11], [9],[10]), was a system devoted to the representation theory of the symmetric group (the main author, Axel Kohnert, passed away in 2013 following a tragic bicycle incident). DCC ("double coset constructor") was a program to construct double cosets in groups due to Bernd Schmalz [20],[19]. SYMMETRICA, DCC, DISCRETA and many other software projects took place over a period of perhaps 30 years at the Lehrstuhl II of Mathematics of Professor Kerber and Professor Laue from the University of Bayreuth in Germany.

SYMMETRICA had an interesting design strategy. Though written in C, it followed a clear object-oriented design philosophy. The design was developed by Axel Kohnert in collaboration with professor T. P. McDonough from the university of Aberystwyth in the UK. There was only one class, of type object, and all types of objects were realized in this class. A type value inside this class indicates which specific type of object was realized. In Fig. 3, a collaboration diagram shows the relations between the different types of objects. The type value is used to dispatch function calls to the appropriate implementation for the type at hand. This mechanism resembles virtual functions in C++. Here is a piece of C source extracted from the header file of SYMMETRICA. This piece of code is responsible for the design of objects as a single type value plus one data element of type union:

```
typedef INT OBJECTKIND;

typedef union {
   INT ob_INT;
   INT * ob_INTpointer;
   char *ob_charpointer;
   struct bruch *ob_bruch;
   struct graph *ob_graph;
   struct list *ob_list;
   struct longint *ob_longint;
   struct matrix *ob_matrix;
   struct monom *ob_monom;
   struct number *ob_number;
   struct partition *ob_partition;
   struct permutation *ob_permutation;
   struct reihe *ob_reihe;
   struct skewpartition *ob_skewpartition;
   struct symchar *ob_symchar;
   struct tableaux *ob_tableaux;
   struct vector *ob_vector;
   } OBJECTSELF;


struct object { OBJECTKIND ob_kind; OBJECTSELF ob_self; };
```

Orbiter and DISCRETA both took some inspiration from this design. Perhaps it is illustrative to take a quick look at DISCRETA, the predecessor system of Orbiter. The design of Orbiter grew out of some ideas introduced in DISCRETA around 2000. We can trace this to a specific set of classes. One of the mathematical problems that drove the development of both DISCRETA and Orbiter was the problem of classifying optimal linear codes. In DISCRETA, we find the collaboration diagram for a class `linear_codes_data` shown in Figure 4. On the left side, we see classes base and OBJECTSELF which represent the SYMMETRICA tradition of storing objects. These objects are used to store the group by

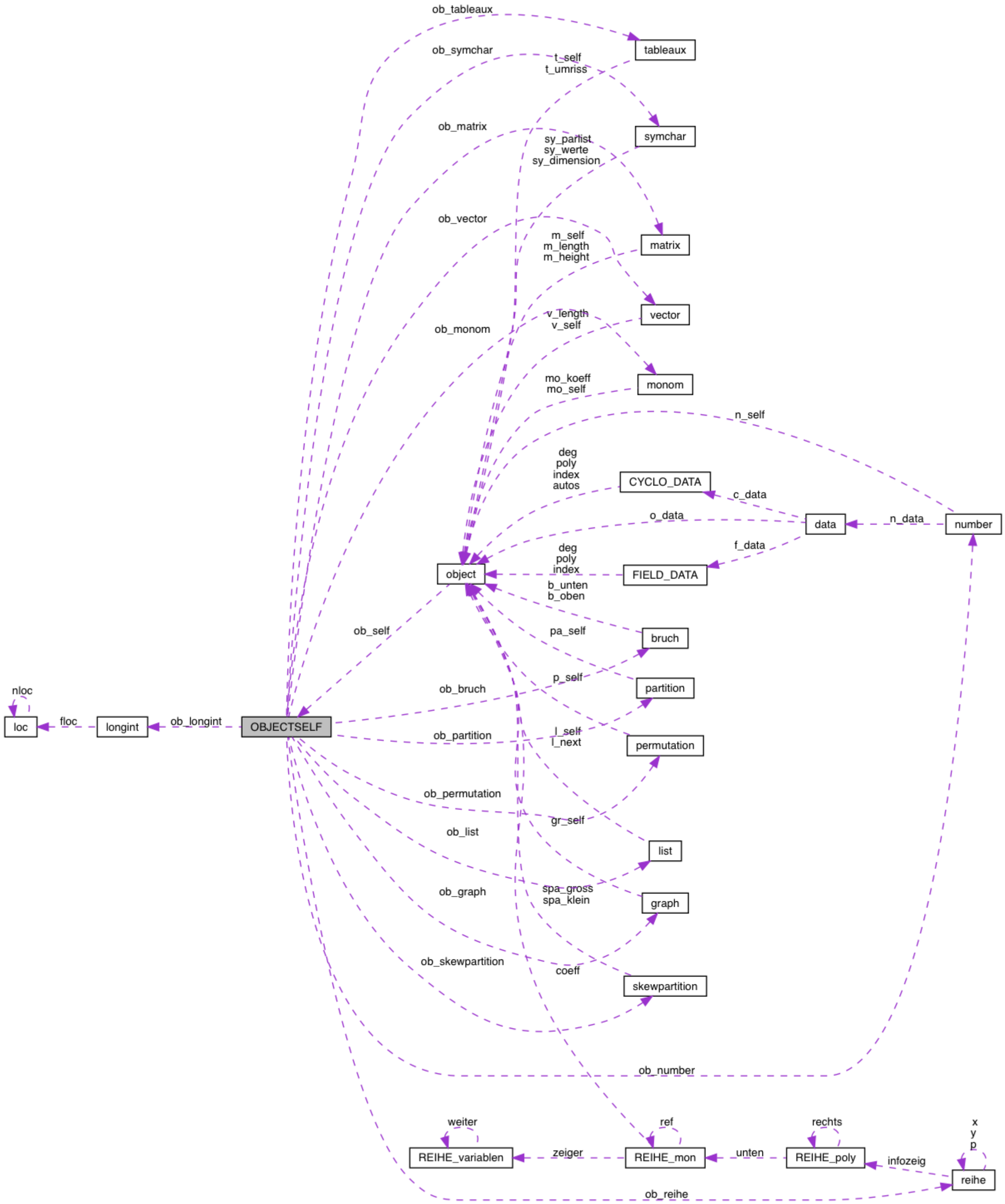


Figure 3: Collaboration graph for SYMMETRICA

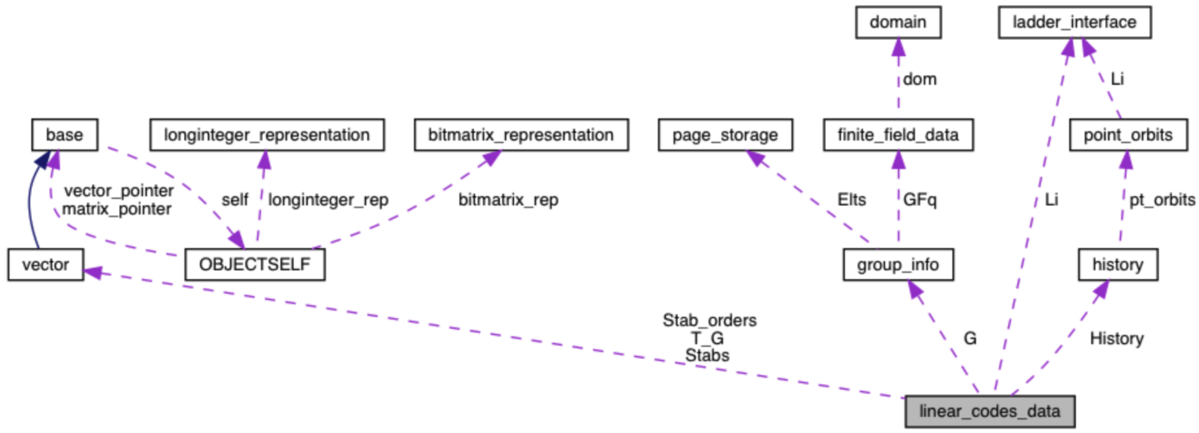


Figure 4: Collaboration graph for DISCRETA's class linear_codes_data

means of a stabilizer chain. On the right, we see several classes which were blueprints for important classes in Orbiter:

1. The class `group_info` developed into the Orbiter classes `matrix_group` and `perm_group`, implementing matrix groups and abstract permutation groups, respectively. At the time of DISCRETA, all groups were abstract permutation groups.

2. The class `page_storage` is identical to a class of the same name in Orbiter. This class provides bulk-storage for elements of a fixed group. In Orbiter, the group elements are stored in a compressed form, using a bitrepresentation of the group element. For matrix groups, this means that the matrices are encoded into bitvectors. In DISCRETA, the storage was less efficient because group elements were stored as abstract permutations.

3. The class `ladder_interface` ultimately grew into the class `action` in Orbiter. In DISCRETA, `ladder_interface` was simply a table of functions pointers which enable the abstract group action. In Orbiter, the function pointers are almost identical.

4. The class `history` was the implementation of the poset classification algorithm in DISCRETA. In orbiter, it is replaced by the classes `generator`, `poset_orbit_node` and `upstep_work`.

5. The class `point_orbits` was the implementation of Schreier trees for orbits of groups on sets. The implementation relied on a bijection between the set and an interval of integers. This class was the predecessor class of the Orbiter class `schreier`.

Let us take a closer look at Orbiter. The main concept of a symmetry group is realized in a way that resembles the object structure in SYMMETRICA. Different classes implement different kinds of groups. All classes are accessible through an object of type `symmetry group`, which is simply a union of pointers to objects of the various classes (cf. Fig. 5). The class `symmetry_group` is closely related to the class `action` which provides a framework for group

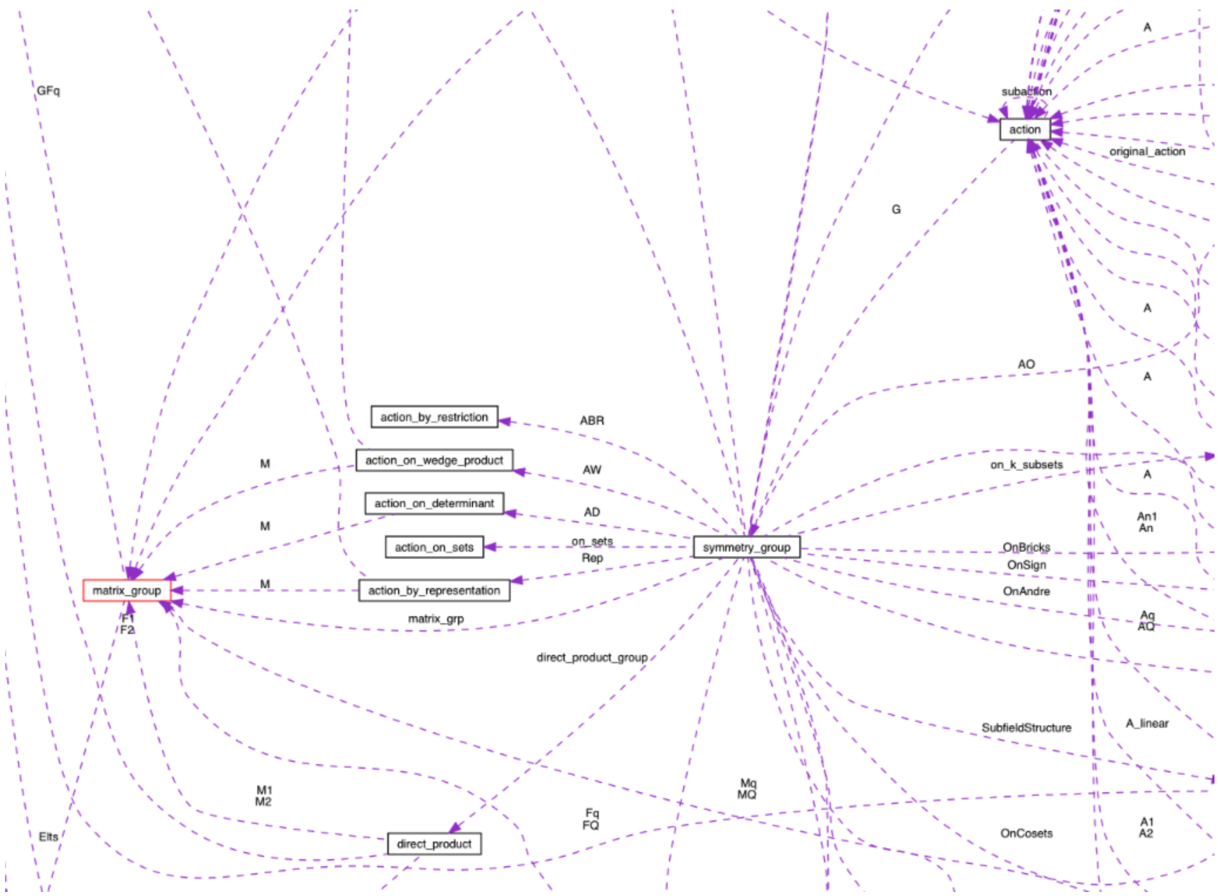


Figure 5: Collaboration graph for Orbiter's symmetry_group (cutout)

actions. A function pointer table in action realizes the dispatch of functions based on a type value. As mentioned earlier, the table of function pointers in action is almost identical to the table of function pointers in the DISCRETA class ladder_interface.

A Ovoid search in $O^{-}(6, 2)$

Here is the makefile for the ovoid search in $O^{-}(6, 2)$:

```
MY_PATH=~/DEV.18
SRC=$(MY_PATH)/GITHUB/orbiter/ORBITER/SRC
SRC2=$(MY_PATH)/ORBITER2/SRC2

OVOID_PATH=$(SRC)/APPS/OVOID
TOOLS_PATH=$(SRC)/APPS/TOOLS


Op42:
   $(OVOID_PATH)/ovoid.out -v 2 -epsilon 1 -d 4 -q 2

Om42:
   $(OVOID_PATH)/ovoid.out -v 2 -epsilon -1 -d 4 -q 2

Op62:
   $(OVOID_PATH)/ovoid.out -v 2 -epsilon 1 -d 6 -q 2

Om62:
   $(OVOID_PATH)/ovoid.out -v 5 -epsilon -1 -n 5 -q 2 -draw_poset -embedded -W

# order 25920, degree 27 = U_4(2) = Weyl group of type E_6 = Sp_4(2) = O_5(3)


draw:
   $(TOOLS_PATH)/layered_graph_main.out -v 4 \
      -file ovoid_-1_6_2_poset_lvl_9.layered_graph \
      -draw test \
      -rad 25000 \
      -xin 1000000 \
      -yin 1000000 \
      -xout 1000000 \
      -yout 1000000 \
      -embedded \
      -scale .44 \
      -line_width 1.0 \
      -y_stretch 0.8
   pdflatex ovoid_-1_6_2_poset_lvl_9_draw.tex
   open ovoid_-1_6_2_poset_lvl_9_draw.pdf
```

B The class action

In Figur 6, the collaboration graph for the class action is shown.

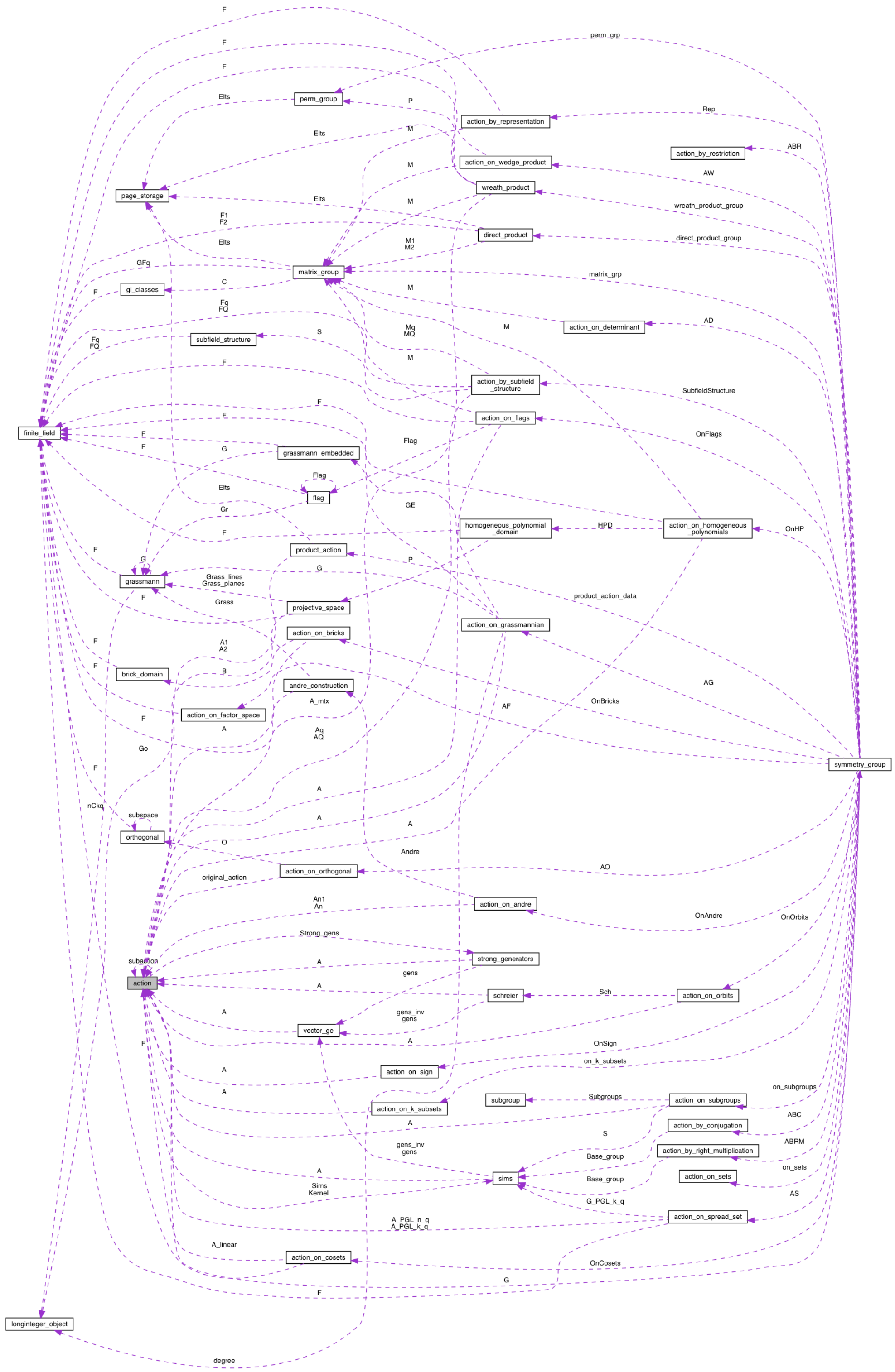


Figure 6: The collaboration graph for the action class

C The orbiter source tree

The following is a list of all files in Orbiter's source tree before make. At the time of writing (August 2018), these files together account for about 560,000 lines of code. Of these about 210,000 lines are combinatorial data that has been computed using Orbiter. These lines are machine-generated. The remaining 350,000 lines are coded manually. Of these, about 6,400 lines are from the nauty software package [15]. I thank Abdullah Alazemi for helping with the Nauty interface. Nauty is a graph isomorphism and canonization package written by Brendan McKay from the Australian National University at Canberra.

```

./src/.cproject
./src/.project
./src/.settings/language.settings.xml
./src/apps/arcs/arcs_main.cpp
./src/apps/arcs/arcs_orderly.cpp
./src/apps/arcs/create_group.cpp
./src/apps/arcs/k_arc_generator_main.cpp
./src/apps/arcs/k_arc_lifting.cpp
./src/apps/arcs/makefile
./src/apps/arcs/test_arc.cpp
./src/apps/arcs/test_hyperoval.cpp
./src/apps/blt/blt.h
./src/apps/blt/blt_main.cpp
./src/apps/blt/blt_set.cpp
./src/apps/blt/blt_set2.cpp
./src/apps/blt/create_BLT_set_main.cpp
./src/apps/blt/makefile
./src/apps/blt/plot_stats_on_graphs.cpp
./src/apps/codes/code_generator.cpp
./src/apps/codes/codes.cpp
./src/apps/codes/codes.h
./src/apps/codes/makefile
./src/apps/combinatorics/all_k_subsets.cpp
./src/apps/combinatorics/alphabet.cpp
./src/apps/combinatorics/alphabet2.cpp
./src/apps/combinatorics/burnside.cpp
./src/apps/combinatorics/cayley.cpp
./src/apps/combinatorics/cayley_sym_n.cpp
./src/apps/combinatorics/change_1_dollar.cpp
./src/apps/combinatorics/conic.cpp
./src/apps/combinatorics/conjugacy_classes_sym_n.cpp
./src/apps/combinatorics/costas.cpp
./src/apps/combinatorics/design.cpp
./src/apps/combinatorics/ferdinand.cpp
./src/apps/combinatorics/fibonacci.cpp
./src/apps/combinatorics/field.cpp
./src/apps/combinatorics/grassmann.cpp
./src/apps/combinatorics/group_ring.cpp
./src/apps/combinatorics/had.cpp
./src/apps/combinatorics/hadamard.cpp
./src/apps/combinatorics/hamming.cpp
./src/apps/combinatorics/johnson.cpp
./src/apps/combinatorics/johnson_table.cpp
./src/apps/combinatorics/lehmer.c
./src/apps/combinatorics/makefile
./src/apps/combinatorics/matrix_rank.cpp
./src/apps/combinatorics/nauty.cpp
./src/apps/combinatorics/nondecreasing.cpp
./src/apps/combinatorics/orthogonal.cpp
./src/apps/combinatorics/output.txt
./src/apps/combinatorics/paley.cpp
./src/apps/combinatorics/partitions.cpp
./src/apps/combinatorics/pentomino_5x5.cpp
```

55 ./src/apps/combinatorics/poset_of_subsets.cpp
56 ./src/apps/combinatorics/problem11.cpp
57 ./src/apps/combinatorics/puzzle.cpp
58 ./src/apps/combinatorics/random.cpp
59 ./src/apps/combinatorics/random_permutation.cpp
60 ./src/apps/combinatorics/rank_anything.cpp
61 ./src/apps/combinatorics/rank_subsets_lex.cpp
62 ./src/apps/combinatorics/sarnak.cpp
63 ./src/apps/combinatorics/schlaefli.cpp
64 ./src/apps/combinatorics/sequences.cpp
65 ./src/apps/combinatorics/shrikhande.cpp
66 ./src/apps/combinatorics/srg.cpp
67 ./src/apps/combinatorics/subsets.cpp
68 ./src/apps/combinatorics/tao.cpp
69 ./src/apps/combinatorics/test.cpp
70 ./src/apps/combinatorics/test2.cpp
71 ./src/apps/combinatorics/unrank.cpp
72 ./src/apps/combinatorics/winnie_li.cpp
73 ./src/apps/graph/graph.cpp
74 ./src/apps/graph/graph.h
75 ./src/apps/graph/graph_generator.cpp
76 ./src/apps/graph/makefile
77 ./src/apps/groups/direct_product.cpp
78 ./src/apps/groups/makefile
79 ./src/apps/groups/orthogonal_group.cpp
80 ./src/apps/groups/wreath_product.cpp
81 ./src/apps/makefile
82 ./src/apps/ovoid/make_design.cpp
83 ./src/apps/ovoid/makefile
84 ./src/apps/ovoid/ovoid.cpp
85 ./src/apps/ovoid/ovoid.h
86 ./src/apps/ovoid/ovoid_generator.cpp
87 ./src/apps/projective_space/analyze.cpp
88 ./src/apps/projective_space/analyze_projective_code.cpp
89 ./src/apps/projective_space/analyze_q_designs.cpp
90 ./src/apps/projective_space/baer_subplane.cpp
91 ./src/apps/projective_space/cheat_sheet_PG.cpp
92 ./src/apps/projective_space/determine_conic.cpp
93 ./src/apps/projective_space/determine_quadric.cpp
94 ./src/apps/projective_space/example_fano_plane.cpp
95 ./src/apps/projective_space/export_data.cpp
96 ./src/apps/projective_space/factor_space.cpp
97 ./src/apps/projective_space/hill.cpp
98 ./src/apps/projective_space/make_BLT_set.cpp
99 ./src/apps/projective_space/make_grassmannian.cpp
100 ./src/apps/projective_space/make_group.cpp
101 ./src/apps/projective_space/make_something.cpp
102 ./src/apps/projective_space/makefile
103 ./src/apps/projective_space/orthogonal_points.cpp
104 ./src/apps/projective_space/ovoid.cpp
105 ./src/apps/projective_space/plane_search.cpp
106 ./src/apps/projective_space/points.cpp
107 ./src/apps/projective_space/polar.cpp
108 ./src/apps/projective_space/process.cpp
109 ./src/apps/projective_space/process_PG.cpp
110 ./src/apps/projective_space/projective_space_main.cpp
111 ./src/apps/projective_space/simeon.cpp
112 ./src/apps/projective_space/test_PG.cpp
113 ./src/apps/regular_ls/makefile
114 ./src/apps/regular_ls/regular_ls.cpp
115 ./src/apps/regular_ls/regular_ls.h
116 ./src/apps/regular_ls/regular_ls_generator.cpp
117 ./src/apps/regular_ls/transpose.cpp
118 ./src/apps/spread/makefile
119 ./src/apps/spread/spread_classify.cpp
120 ./src/apps/spread/spread_create.cpp
121 ./src/apps/subspace_orbits/desarguesian_spread.cpp
122 ./src/apps/subspace_orbits/linear_set.cpp

123 ./src/apps/subspace_orbits/linear_set.h
124 ./src/apps/subspace_orbits/linear_set2.cpp
125 ./src/apps/subspace_orbits/linear_set_main.cpp
126 ./src/apps/subspace_orbits/makefile
127 ./src/apps/subspace_orbits/subspace_orbits_main.cpp
128 ./src/apps/surfaces/arc_lifting_main.cpp
129 ./src/apps/surfaces/create_surface_main.cpp
130 ./src/apps/surfaces/makefile
131 ./src/apps/surfaces/surface_classify.cpp
132 ./src/apps/tdo/makefile
133 ./src/apps/tdo/maxfit.cpp
134 ./src/apps/tdo/packing.cpp
135 ./src/apps/tdo/tdo_can.cpp
136 ./src/apps/tdo/tdo_print.cpp
137 ./src/apps/tdo/tdo_refine.cpp
138 ./src/apps/tdo/tdo_start.cpp
139 ./src/apps/tools/all_cliques.cpp
140 ./src/apps/tools/all_cycles.cpp
141 ./src/apps/tools/all_rainbow_cliques.cpp
142 ./src/apps/tools/cc2widor.cpp
143 ./src/apps/tools/collect.cpp
144 ./src/apps/tools/colored_graph.cpp
145 ./src/apps/tools/concatenate_files.cpp
146 ./src/apps/tools/create_file.cpp
147 ./src/apps/tools/create_graph.cpp
148 ./src/apps/tools/create_layered_graph_file.cpp
149 ./src/apps/tools/distribution.cpp
150 ./src/apps/tools/dlx.cpp
151 ./src/apps/tools/draw_colored_graph.cpp
152 ./src/apps/tools/draw_graph.cpp
153 ./src/apps/tools/graph.cpp
154 ./src/apps/tools/join_sets.cpp
155 ./src/apps/tools/latex_table.cpp
156 ./src/apps/tools/layered_graph_main.cpp
157 ./src/apps/tools/loop.cpp
158 ./src/apps/tools/makefile
159 ./src/apps/tools/missing_files.cpp
160 ./src/apps/tools/plot_decomposition_matrix.cpp
161 ./src/apps/tools/plot_xy.cpp
162 ./src/apps/tools/prepare_frames.cpp
163 ./src/apps/tools/read_orbiter_file.cpp
164 ./src/apps/tools/read_solutions.cpp
165 ./src/apps/tools/run_blt.cpp
166 ./src/apps/tools/sajeeb.cpp
167 ./src/apps/tools/sajeeb.h
168 ./src/apps/tools/scheduler.cpp
169 ./src/apps/tools/solve_diophant.cpp
170 ./src/apps/tools/split.cpp
171 ./src/apps/tools/treedraw.cpp
172 ./src/apps/tools/widor.cpp
173 ./src/lib/DISCRETA/base.cpp
174 ./src/lib/DISCRETA/bitmatrix.cpp
175 ./src/lib/DISCRETA/bt_key.cpp
176 ./src/lib/DISCRETA/btree.cpp
177 ./src/lib/DISCRETA/database.cpp
178 ./src/lib/DISCRETA/design.cpp
179 ./src/lib/DISCRETA/design_parameter.cpp
180 ./src/lib/DISCRETA/design_parameter_source.cpp
181 ./src/lib/DISCRETA/discreta.h
182 ./src/lib/DISCRETA/discreta_global.cpp
183 ./src/lib/DISCRETA/domain.cpp
184 ./src/lib/DISCRETA/geometry.cpp
185 ./src/lib/DISCRETA/global.cpp
186 ./src/lib/DISCRETA/group_selection.cpp
187 ./src/lib/DISCRETA/hollerith.cpp
188 ./src/lib/DISCRETA/integer.cpp
189 ./src/lib/DISCRETA/longinteger.cpp
190 ./src/lib/DISCRETA/makefile

191 ./src/lib/DISCRETA/matrix.cpp
192 ./src/lib/DISCRETA/memory.cpp
193 ./src/lib/DISCRETA/number_partition.cpp
194 ./src/lib/DISCRETA/page_table.cpp
195 ./src/lib/DISCRETA/perm_group_gens.cpp
196 ./src/lib/DISCRETA/permutation.cpp
197 ./src/lib/DISCRETA/solid.cpp
198 ./src/lib/DISCRETA/unipoly.cpp
199 ./src/lib/DISCRETA/vector.cpp
200 ./src/lib/foundations/algebra_and_number_theory/a_domain.cpp
201 ./src/lib/foundations/algebra_and_number_theory/algebra_and_number_theory.h
202 ./src/lib/foundations/algebra_and_number_theory/finite_field.cpp
203 ./src/lib/foundations/algebra_and_number_theory/finite_field_linear_algebra.cpp
204 ./src/lib/foundations/algebra_and_number_theory/finite_field_representations.cpp
205 ./src/lib/foundations/algebra_and_number_theory/finite_field_tables.cpp
206 ./src/lib/foundations/algebra_and_number_theory/finite_ring.cpp
207 ./src/lib/foundations/algebra_and_number_theory/generators_symplectic_group.cpp
208 ./src/lib/foundations/algebra_and_number_theory/gl_classes.cpp
209 ./src/lib/foundations/algebra_and_number_theory/group_generators.cpp
210 ./src/lib/foundations/algebra_and_number_theory/heisenberg.cpp
211 ./src/lib/foundations/algebra_and_number_theory/homogeneous_polynomial_domain.cpp
212 ./src/lib/foundations/algebra_and_number_theory/longinteger_domain.cpp
213 ./src/lib/foundations/algebra_and_number_theory/longinteger_object.cpp
214 ./src/lib/foundations/algebra_and_number_theory/makefile
215 ./src/lib/foundations/algebra_and_number_theory/norm_tables.cpp
216 ./src/lib/foundations/algebra_and_number_theory/null_polarity_generator.cpp
217 ./src/lib/foundations/algebra_and_number_theory/number_theory.cpp
218 ./src/lib/foundations/algebra_and_number_theory/rank_checker.cpp
219 ./src/lib/foundations/algebra_and_number_theory/subfield_structure.cpp
220 ./src/lib/foundations/algebra_and_number_theory/unipoly_domain.cpp
221 ./src/lib/foundations/coding_theory/coding_theory.h
222 ./src/lib/foundations/coding_theory/makefile
223 ./src/lib/foundations/coding_theory/mindist.cpp
224 ./src/lib/foundations/coding_theory/tensor.cpp
225 ./src/lib/foundations/combinatorics/brick_domain.cpp
226 ./src/lib/foundations/combinatorics/classify_bitvectors.cpp
227 ./src/lib/foundations/combinatorics/combinatorics.cpp
228 ./src/lib/foundations/combinatorics/combinatorics.h
229 ./src/lib/foundations/combinatorics/makefile
230 ./src/lib/foundations/data_structures/INT_matrix.cpp
231 ./src/lib/foundations/data_structures/INT_vector.cpp
232 ./src/lib/foundations/data_structures/data_file.cpp
233 ./src/lib/foundations/data_structures/data_structures.h
234 ./src/lib/foundations/data_structures/fancy_set.cpp
235 ./src/lib/foundations/data_structures/makefile
236 ./src/lib/foundations/data_structures/partitionstack.cpp
237 ./src/lib/foundations/data_structures/set_of_sets.cpp
238 ./src/lib/foundations/data_structures/sorting.cpp
239 ./src/lib/foundations/data_structures/spreadsheet.cpp
240 ./src/lib/foundations/data_structures/super_fast_hash.cpp
241 ./src/lib/foundations/data_structures/vector_hashing.cpp
242 ./src/lib/foundations/foundations.h
243 ./src/lib/foundations/geometry/DATA/data_BLT.cpp
244 ./src/lib/foundations/geometry/DATA/data_DH.cpp
245 ./src/lib/foundations/geometry/DATA/data_hyperovals.cpp
246 ./src/lib/foundations/geometry/DATA/data_packings_PG_3_3.cpp
247 ./src/lib/foundations/geometry/DATA/data_spreads.cpp
248 ./src/lib/foundations/geometry/DATA/surface_11.cpp
249 ./src/lib/foundations/geometry/DATA/surface_13.cpp
250 ./src/lib/foundations/geometry/DATA/surface_16.cpp
251 ./src/lib/foundations/geometry/DATA/surface_17.cpp
252 ./src/lib/foundations/geometry/DATA/surface_19.cpp
253 ./src/lib/foundations/geometry/DATA/surface_23.cpp
254 ./src/lib/foundations/geometry/DATA/surface_25.cpp
255 ./src/lib/foundations/geometry/DATA/surface_27.cpp
256 ./src/lib/foundations/geometry/DATA/surface_29.cpp
257 ./src/lib/foundations/geometry/DATA/surface_31.cpp
258 ./src/lib/foundations/geometry/DATA/surface_32.cpp

259 ./src/lib/foundations/geometry/DATA/surface_37.cpp
260 ./src/lib/foundations/geometry/DATA/surface_4.cpp
261 ./src/lib/foundations/geometry/DATA/surface_41.cpp
262 ./src/lib/foundations/geometry/DATA/surface_43.cpp
263 ./src/lib/foundations/geometry/DATA/surface_47.cpp
264 ./src/lib/foundations/geometry/DATA/surface_49.cpp
265 ./src/lib/foundations/geometry/DATA/surface_53.cpp
266 ./src/lib/foundations/geometry/DATA/surface_59.cpp
267 ./src/lib/foundations/geometry/DATA/surface_61.cpp
268 ./src/lib/foundations/geometry/DATA/surface_64.cpp
269 ./src/lib/foundations/geometry/DATA/surface_67.cpp
270 ./src/lib/foundations/geometry/DATA/surface_7.cpp
271 ./src/lib/foundations/geometry/DATA/surface_71.cpp
272 ./src/lib/foundations/geometry/DATA/surface_73.cpp
273 ./src/lib/foundations/geometry/DATA/surface_79.cpp
274 ./src/lib/foundations/geometry/DATA/surface_8.cpp
275 ./src/lib/foundations/geometry/DATA/surface_81.cpp
276 ./src/lib/foundations/geometry/DATA/surface_83.cpp
277 ./src/lib/foundations/geometry/DATA/surface_89.cpp
278 ./src/lib/foundations/geometry/DATA/surface_9.cpp
279 ./src/lib/foundations/geometry/DATA/surface_97.cpp
280 ./src/lib/foundations/geometry/andre_construction.cpp
281 ./src/lib/foundations/geometry/andre_construction_line_element.cpp
282 ./src/lib/foundations/geometry/andre_construction_point_element.cpp
283 ./src/lib/foundations/geometry/buekenhout_metz.cpp
284 ./src/lib/foundations/geometry/data.cpp
285 ./src/lib/foundations/geometry/decomposition.cpp
286 ./src/lib/foundations/geometry/desarguesian_spread.cpp
287 ./src/lib/foundations/geometry/eckardt_point.cpp
288 ./src/lib/foundations/geometry/flag.cpp
289 ./src/lib/foundations/geometry/geo_parameter.cpp
290 ./src/lib/foundations/geometry/geometric_object.cpp
291 ./src/lib/foundations/geometry/geometric_operations.cpp
292 ./src/lib/foundations/geometry/geometry.h
293 ./src/lib/foundations/geometry/grassmann.cpp
294 ./src/lib/foundations/geometry/grassmann_embedded.cpp
295 ./src/lib/foundations/geometry/hermitian.cpp
296 ./src/lib/foundations/geometry/hjelmslev.cpp
297 ./src/lib/foundations/geometry/inc_gen_global.cpp
298 ./src/lib/foundations/geometry/incidence_structure.cpp
299 ./src/lib/foundations/geometry/klein_correspondence.cpp
300 ./src/lib/foundations/geometry/knarr.cpp
301 ./src/lib/foundations/geometry/makefile
302 ./src/lib/foundations/geometry/object_in_projective_space.cpp
303 ./src/lib/foundations/geometry/orthogonal.cpp
304 ./src/lib/foundations/geometry/orthogonal_points.cpp
305 ./src/lib/foundations/geometry/packing.cpp
306 ./src/lib/foundations/geometry/point_line.cpp
307 ./src/lib/foundations/geometry/projective.cpp
308 ./src/lib/foundations/geometry/projective_space.cpp
309 ./src/lib/foundations/geometry/surface.cpp
310 ./src/lib/foundations/geometry/surface_object.cpp
311 ./src/lib/foundations/geometry/tdo_data.cpp
312 ./src/lib/foundations/geometry/tdo_scheme.cpp
313 ./src/lib/foundations/geometry/unusual.cpp
314 ./src/lib/foundations/geometry/w3q.cpp
315 ./src/lib/foundations/globals/galois_global.cpp
316 ./src/lib/foundations/globals/globals.h
317 ./src/lib/foundations/globals/magma_interface.cpp
318 ./src/lib/foundations/globals/makefile
319 ./src/lib/foundations/globals/numerics.cpp
320 ./src/lib/foundations/graph_theory/clique_finder.cpp
321 ./src/lib/foundations/graph_theory/colored_graph.cpp
322 ./src/lib/foundations/graph_theory/graph_layer.cpp
323 ./src/lib/foundations/graph_theory/graph_node.cpp
324 ./src/lib/foundations/graph_theory/graph_theory.h
325 ./src/lib/foundations/graph_theory/layered_graph.cpp
326 ./src/lib/foundations/graph_theory/layered_graph_draw_options.cpp

327 ./src/lib/foundations/graph_theory/makefile
328 ./src/lib/foundations/graph_theory/rainbow_cliques.cpp
329 ./src/lib/foundations/graph_theory_nauty/graph_theory_nauty.h
330 ./src/lib/foundations/graph_theory_nauty/makefile
331 ./src/lib/foundations/graph_theory_nauty/naugraph.c
332 ./src/lib/foundations/graph_theory_nauty/naurng.c
333 ./src/lib/foundations/graph_theory_nauty/naurng.h
334 ./src/lib/foundations/graph_theory_nauty/nautil.c
335 ./src/lib/foundations/graph_theory_nauty/nauty.c
336 ./src/lib/foundations/graph_theory_nauty/nauty.h
337 ./src/lib/foundations/graph_theory_nauty/nauty_interface.cpp
338 ./src/lib/foundations/graph_theory_nauty/schreier.c
339 ./src/lib/foundations/graph_theory_nauty/schreier.h
340 ./src/lib/foundations/graph_theory_nauty/sorttemplates.c
341 ./src/lib/foundations/graphics/draw.cpp
342 ./src/lib/foundations/graphics/graphics.h
343 ./src/lib/foundations/graphics/makefile
344 ./src/lib/foundations/graphics/mp_graphics.cpp
345 ./src/lib/foundations/graphics/plot.cpp
346 ./src/lib/foundations/graphics/scene.cpp
347 ./src/lib/foundations/graphics/tree.cpp
348 ./src/lib/foundations/graphics/tree_node.cpp
349 ./src/lib/foundations/io_and_os/dynamic_memory.cpp
350 ./src/lib/foundations/io_and_os/file_output.cpp
351 ./src/lib/foundations/io_and_os/io_and_os.h
352 ./src/lib/foundations/io_and_os/makefile
353 ./src/lib/foundations/io_and_os/memory.cpp
354 ./src/lib/foundations/io_and_os/memory_object.cpp
355 ./src/lib/foundations/io_and_os/orbiter_data_file.cpp
356 ./src/lib/foundations/io_and_os/util.cpp
357 ./src/lib/foundations/makefile
358 ./src/lib/foundations/solvers/diophant.cpp
359 ./src/lib/foundations/solvers/dlx.cpp
360 ./src/lib/foundations/solvers/makefile
361 ./src/lib/foundations/solvers/mckay.cpp
362 ./src/lib/foundations/solvers/solvers.h
363 ./src/lib/foundations/statistics/classify.cpp
364 ./src/lib/foundations/statistics/makefile
365 ./src/lib/foundations/statistics/statistics.h
366 ./src/lib/groups_and_group_actions/data_structures/data_structures.h
367 ./src/lib/groups_and_group_actions/data_structures/group.cpp
368 ./src/lib/groups_and_group_actions/data_structures/makefile
369 ./src/lib/groups_and_group_actions/data_structures/orbit_transversal.cpp
370 ./src/lib/groups_and_group_actions/data_structures/page_storage.cpp
371 ./src/lib/groups_and_group_actions/data_structures/projective_space_with_action.cpp
372 ./src/lib/groups_and_group_actions/data_structures/schreier_vector.cpp
373 ./src/lib/groups_and_group_actions/data_structures/set_and_stabilizer.cpp
374 ./src/lib/groups_and_group_actions/data_structures/union_find.cpp
375 ./src/lib/groups_and_group_actions/data_structures/union_find_on_k_subsets.cpp
376 ./src/lib/groups_and_group_actions/data_structures/vector_ge.cpp
377 ./src/lib/groups_and_group_actions/group_actions/action.cpp
378 ./src/lib/groups_and_group_actions/group_actions/action_cb.cpp
379 ./src/lib/groups_and_group_actions/group_actions/action_global.cpp
380 ./src/lib/groups_and_group_actions/group_actions/action_indexing_cosets.cpp
381 ./src/lib/groups_and_group_actions/group_actions/action_induce.cpp
382 ./src/lib/groups_and_group_actions/group_actions/action_init.cpp
383 ./src/lib/groups_and_group_actions/group_actions/backtrack.cpp
384 ./src/lib/groups_and_group_actions/group_actions/group_actions.h
385 ./src/lib/groups_and_group_actions/group_actions/interface.cpp
386 ./src/lib/groups_and_group_actions/group_actions/interface_direct_product.cpp
387 ./src/lib/groups_and_group_actions/group_actions/interface_matrix_group.cpp
388 ./src/lib/groups_and_group_actions/group_actions/interface_perm_group.cpp
389 ./src/lib/groups_and_group_actions/group_actions/interface_wreath_product.cpp
390 ./src/lib/groups_and_group_actions/group_actions/makefile
391 ./src/lib/groups_and_group_actions/group_theory/direct_product.cpp
392 ./src/lib/groups_and_group_actions/group_theory/group_theory.h
393 ./src/lib/groups_and_group_actions/group_theory/linear_group.cpp
394 ./src/lib/groups_and_group_actions/group_theory/linear_group_description.cpp

395 ./src/lib/groups_and_group_actions/group_theory/makefile
396 ./src/lib/groups_and_group_actions/group_theory/matrix_group.cpp
397 ./src/lib/groups_and_group_actions/group_theory/perm_group.cpp
398 ./src/lib/groups_and_group_actions/group_theory/schreier.cpp
399 ./src/lib/groups_and_group_actions/group_theory/schreier_sims.cpp
400 ./src/lib/groups_and_group_actions/group_theory/sims.cpp
401 ./src/lib/groups_and_group_actions/group_theory/sims2.cpp
402 ./src/lib/groups_and_group_actions/group_theory/sims3.cpp
403 ./src/lib/groups_and_group_actions/group_theory/sims_global.cpp
404 ./src/lib/groups_and_group_actions/group_theory/strong_generators.cpp
405 ./src/lib/groups_and_group_actions/group_theory/strong_generators_groups.cpp
406 ./src/lib/groups_and_group_actions/group_theory/subgroup.cpp
407 ./src/lib/groups_and_group_actions/group_theory/wreath_product.cpp
408 ./src/lib/groups_and_group_actions/groups_and_group_actions.h
409 ./src/lib/groups_and_group_actions/induced_actions/action_by_conjugation.cpp
410 ./src/lib/groups_and_group_actions/induced_actions/action_by_representation.cpp
411 ./src/lib/groups_and_group_actions/induced_actions/action_by_restriction.cpp
412 ./src/lib/groups_and_group_actions/induced_actions/action_by_right_multiplication.cpp
413 ./src/lib/groups_and_group_actions/induced_actions/action_by_subfield_structure.cpp
414 ./src/lib/groups_and_group_actions/induced_actions/action_on_andre.cpp
415 ./src/lib/groups_and_group_actions/induced_actions/action_on_bricks.cpp
416 ./src/lib/groups_and_group_actions/induced_actions/action_on_cosets.cpp
417 ./src/lib/groups_and_group_actions/induced_actions/action_on_determinant.cpp
418 ./src/lib/groups_and_group_actions/induced_actions/action_on_factor_space.cpp
419 ./src/lib/groups_and_group_actions/induced_actions/action_on_flags.cpp
420 ./src/lib/groups_and_group_actions/induced_actions/action_on_grassmannian.cpp
421 ./src/lib/groups_and_group_actions/induced_actions/action_on_homogeneous_polynomials.cpp
422 ./src/lib/groups_and_group_actions/induced_actions/action_on_k_subsets.cpp
423 ./src/lib/groups_and_group_actions/induced_actions/action_on_orbits.cpp
424 ./src/lib/groups_and_group_actions/induced_actions/action_on_orthogonal.cpp
425 ./src/lib/groups_and_group_actions/induced_actions/action_on_sets.cpp
426 ./src/lib/groups_and_group_actions/induced_actions/action_on_sign.cpp
427 ./src/lib/groups_and_group_actions/induced_actions/action_on_spread_set.cpp
428 ./src/lib/groups_and_group_actions/induced_actions/action_on_subgroups.cpp
429 ./src/lib/groups_and_group_actions/induced_actions/action_on_wedge_product.cpp
430 ./src/lib/groups_and_group_actions/induced_actions/induced_actions.h
431 ./src/lib/groups_and_group_actions/induced_actions/makefile
432 ./src/lib/groups_and_group_actions/induced_actions/product_action.cpp
433 ./src/lib/groups_and_group_actions/makefile
434 ./src/lib/makefile
435 ./src/lib/orbiter.h
436 ./src/lib/poset_classification/classify/classification.cpp
437 ./src/lib/poset_classification/classify/classify.h
438 ./src/lib/poset_classification/classify/flag_orbit_node.cpp
439 ./src/lib/poset_classification/classify/flag_orbits.cpp
440 ./src/lib/poset_classification/classify/makefile
441 ./src/lib/poset_classification/classify/orbit_node.cpp
442 ./src/lib/poset_classification/makefile
443 ./src/lib/poset_classification/other/makefile
444 ./src/lib/poset_classification/other/other.h
445 ./src/lib/poset_classification/other/snakes_and_ladders_global.cpp
446 ./src/lib/poset_classification/poset_classification.h
447 ./src/lib/poset_classification/set_stabilizer/compute_stabilizer.cpp
448 ./src/lib/poset_classification/set_stabilizer/makefile
449 ./src/lib/poset_classification/set_stabilizer/set_stabilizer.h
450 ./src/lib/poset_classification/set_stabilizer/set_stabilizer_compute.cpp
451 ./src/lib/poset_classification/snakes_and_ladders/extension.cpp
452 ./src/lib/poset_classification/snakes_and_ladders/generator.cpp
453 ./src/lib/poset_classification/snakes_and_ladders/generator_classify.cpp
454 ./src/lib/poset_classification/snakes_and_ladders/generator_combinatorics.cpp
455 ./src/lib/poset_classification/snakes_and_ladders/generator_draw.cpp
456 ./src/lib/poset_classification/snakes_and_ladders/generator_init.cpp
457 ./src/lib/poset_classification/snakes_and_ladders/generator_io.cpp
458 ./src/lib/poset_classification/snakes_and_ladders/generator_recognize.cpp
459 ./src/lib/poset_classification/snakes_and_ladders/generator_trace.cpp
460 ./src/lib/poset_classification/snakes_and_ladders/makefile
461 ./src/lib/poset_classification/snakes_and_ladders/oracle.cpp
462 ./src/lib/poset_classification/snakes_and_ladders/oracle_downstep.cpp

463 ./src/lib/poset_classification/snakes_and_ladders/oracle_downstep_subspace_action.cpp
464 ./src/lib/poset_classification/snakes_and_ladders/oracle_io.cpp
465 ./src/lib/poset_classification/snakes_and_ladders/oracle_upstep.cpp
466 ./src/lib/poset_classification/snakes_and_ladders/oracle_upstep_subspace_action.cpp
467 ./src/lib/poset_classification/snakes_and_ladders/snakes_and_ladders.h
468 ./src/lib/poset_classification/snakes_and_ladders/upstep_work.cpp
469 ./src/lib/poset_classification/snakes_and_ladders/upstep_work_subspace_action.cpp
470 ./src/lib/poset_classification/snakes_and_ladders/upstep_work_trace.cpp
471 ./src/lib/top_level/algebra_and_number_theory/analyze_group.cpp
472 ./src/lib/top_level/algebra_and_number_theory/elliptic_curve.cpp
473 ./src/lib/top_level/algebra_and_number_theory/extra.cpp
474 ./src/lib/top_level/algebra_and_number_theory/factor_group.cpp
475 ./src/lib/top_level/algebra_and_number_theory/makefile
476 ./src/lib/top_level/algebra_and_number_theory/tl_algebra_and_number_theory.h
477 ./src/lib/top_level/algebra_and_number_theory/top_level_global.cpp
478 ./src/lib/top_level/algebra_and_number_theory/young.cpp
479 ./src/lib/top_level/geometry/BLT_set_create.cpp
480 ./src/lib/top_level/geometry/BLT_set_create_description.cpp
481 ./src/lib/top_level/geometry/arc_generator.cpp
482 ./src/lib/top_level/geometry/arc_lifting.cpp
483 ./src/lib/top_level/geometry/choose_points_or_lines.cpp
484 ./src/lib/top_level/geometry/classify_double_sixes.cpp
485 ./src/lib/top_level/geometry/classify_trihedral_pairs.cpp
486 ./src/lib/top_level/geometry/decomposition.cpp
487 ./src/lib/top_level/geometry/incidence_structure.cpp
488 ./src/lib/top_level/geometry/k_arc_generator.cpp
489 ./src/lib/top_level/geometry/makefile
490 ./src/lib/top_level/geometry/object_in_projective_space_with_action.cpp
491 ./src/lib/top_level/geometry/polar.cpp
492 ./src/lib/top_level/geometry/projective_space.cpp
493 ./src/lib/top_level/geometry/recoordinatize.cpp
494 ./src/lib/top_level/geometry/search_blocking_set.cpp
495 ./src/lib/top_level/geometry/singer_cycle.cpp
496 ./src/lib/top_level/geometry/six_arcs_not_on_a_conic.cpp
497 ./src/lib/top_level/geometry/spread.cpp
498 ./src/lib/top_level/geometry/spread2.cpp
499 ./src/lib/top_level/geometry/spread_create.cpp
500 ./src/lib/top_level/geometry/spread_create_description.cpp
501 ./src/lib/top_level/geometry/spread_lifting.cpp
502 ./src/lib/top_level/geometry/surface_classify_wedge.cpp
503 ./src/lib/top_level/geometry/surface_create.cpp
504 ./src/lib/top_level/geometry/surface_create_description.cpp
505 ./src/lib/top_level/geometry/surface_object_with_action.cpp
506 ./src/lib/top_level/geometry/surface_with_action.cpp
507 ./src/lib/top_level/geometry/tl_geometry.h
508 ./src/lib/top_level/geometry/translation_plane_via_andre_model.cpp
509 ./src/lib/top_level/isomorph/isomorph.cpp
510 ./src/lib/top_level/isomorph/isomorph.h
511 ./src/lib/top_level/isomorph/isomorph_arguments.cpp
512 ./src/lib/top_level/isomorph/isomorph_database.cpp
513 ./src/lib/top_level/isomorph/isomorph_files.cpp
514 ./src/lib/top_level/isomorph/isomorph_global.cpp
515 ./src/lib/top_level/isomorph/isomorph_testing.cpp
516 ./src/lib/top_level/isomorph/isomorph_trace.cpp
517 ./src/lib/top_level/isomorph/makefile
518 ./src/lib/top_level/makefile
519 ./src/lib/top_level/orbits/makefile
520 ./src/lib/top_level/orbits/orbit_of_equations.cpp
521 ./src/lib/top_level/orbits/orbit_of_sets.cpp
522 ./src/lib/top_level/orbits/orbit_of_subspaces.cpp
523 ./src/lib/top_level/orbits/orbit_rep.cpp
524 ./src/lib/top_level/orbits/orbits.h
525 ./src/lib/top_level/orbits/representatives.cpp
526 ./src/lib/top_level/orbits/subspace_orbits.cpp
527 ./src/lib/top_level/solver/exact_cover.cpp
528 ./src/lib/top_level/solver/exact_cover_arguments.cpp
529 ./src/lib/top_level/solver/makefile
530 ./src/lib/top_level/solver/solver.h

```
./src/lib/top_level/top_level.h
./src/makefile

```

References

[1] Anton Betten. Classifying discrete objects with Orbiter. ACM Communications in Computer Algebra, Vol. 47, No. 4, Issue 186, December 2013.

[2] Anton Betten. Orbiter – a program to classify discrete objects. `https://github.com/abetten/orbiter`, 2013-2018.

[3] Anton Betten. How fast can we compute orbits of groups? In *ICMS 2018—Proceedings of the International Congress on Mathematical Software; James H. Davenport, Manuel Kauers, George Labahn, Josef Urban (ed.)*, pages 62–70. Springer, 2018.

[4] Anton Betten, Evi Haberberger, Reinhard Laue, and Alfred Wassermann. Discerta – a system to construct t-designs with prescibed automorphism group. `http://www.algorithm.uni-bayreuth.de/en/research/discreta/index.html`, circa 1996-2004.

[5] J. H. Conway and N. J. A. Sloane. *Sphere packings, lattices and groups*, volume 290 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, New York, third edition, 1999. With additional contributions by E. Bannai, R. E. Borcherds, J. Leech, S. P. Norton, A. M. Odlyzko, R. A. Parker, L. Queen and B. B. Venkov.

[6] Cygwin. A Unix-like environment and command-line interface for Microsoft Windows. `https://www.cygwin.com`.

[7] I. A. Faradžev. Constructive enumeration of combinatorial objects. In *Problèmes combinatoires et théorie des graphes (Colloq. Internat. CNRS, Univ. Orsay, Orsay, 1976)*, volume 260 of *Colloq. Internat. CNRS*, pages 131–135. CNRS, Paris, 1978.

[8] P. Kaski and P. Östergård. *Classification algorithms for codes and designs*, volume 15 of *Algorithms and Computation in Mathematics*. Springer-Verlag, Berlin, 2006.

[9] A. Kerber and A. Kohnert. SYMMETRICA, an object oriented computer algebra system for representations, combinatorics and applications of symmetric groups. In *Symmetry and structural properties of condensed matter (Zajaczkowo, 1994)*, pages 494–503. World Sci. Publ., River Edge, NJ, 1995.

[10] Adalbert Kerber and Axel Kohnert. Modular irreducible representations of the symmetric group as linear codes. *European J. Combin.*, 25(8):1285–1299, 2004.

[11] Adalbert Kerber, Axel Kohnert, and Alain Lascoux. SYMMETRICA, an object oriented computer-algebra system for the symmetric group. *J. Symbolic Comput.*, 14(2-3):195–203, 1992.

[12] Jeffrey S. Leon. Permutation group algorithms based on partitions. I. Theory and algorithms. *J. Symbolic Comput.*, 12(4-5):533–583, 1991. Computational group theory, Part 2.

[13] Jeffrey S. Leon. Partitions, refinements, and permutation group computation. In *Groups and computation, II (New Brunswick, NJ, 1995)*, volume 28 of *DIMACS Ser. Discrete Math. Theoret. Comput. Sci.*, pages 123–158. Amer. Math. Soc., Providence, RI, 1997.

[14] Brendan D. McKay. Isomorph-free exhaustive generation. *J. Algorithms*, 26(2):306–324, 1998.

[15] Nauty User's Guide (Version 2.4), Brendan McKay, Australian National University, Nov 4, 2009.

[16] Wilhelm Plesken. Counting with groups and rings. *J. Reine Angew. Math.*, 334:40–68, 1982.

[17] Ronald C. Read. Every one a winner or how to avoid isomorphism search when cataloguing combinatorial configurations. *Ann. Discrete Math.*, 2:107–120, 1978. Algorithmic aspects of combinatorics (Conf., Vancouver Island, B.C., 1976).

[18] Gordon F. Royle. An orderly algorithm and some applications in finite geometry. *Discrete Math.*, 185(1-3):105–115, 1998.

[19] B. Schmalz. t-Designs zu vorgegebener Automorphismengruppe. *Bayreuth. Math. Schr.*, 41:1–164, 1992. Dissertation, Universität Bayreuth, Bayreuth, 1992.

[20] Bernd Schmalz. Verwendung von Untergruppenleitern zur Bestimmung von Doppelnebenklassen. *Bayreuth. Math. Schr.*, (31):109–143, 1990.