The database of 2-neighborly 0/1-polytopes of dimensions 6 and 7

Aleksandr N. Maksimenko

April 2, 2019

2-neighborly 0/1-polytopes. We consider only convex polytopes [2]. A convex polytope of dimension d is called a *d-polytope*. A face of P is called *k-face* if its proper dimension is equal to k. A 0-face of a d-polytope is called a *vertex*, a 1-face is called an *edge*, a $(d-1)$-face is called a *facet*. The face lattice of a polytope is the set of all its faces ordered by inclusion. Two polytopes are *combinatorially equivalent* (has the same *combinatorial type*) if they have isomorphic face lattices. It is well known that the combinatorial type of a polytope P with vertices $\{v_1, \dots, v_n\}$ and facets $\{F_1, \dots, F_k\}$ is uniquely determined by its facet-vertex incidence matrix $M = (m_{ij}) \in \{0,1\}^{k \times n}$, where $m_{ij} = 1$ if facet F_i contains vertex v_j, and $m_{ij} = 0$ otherwise. Thus, polytopes are combinatorially equivalent iff their facet-vertex incidence matrices differ only by column and row permutations.

Let f_i, $i = 0, 1, \dots, d-1$, be the number of i-faces of P. The vector $(f_0, f_1, \dots, f_{d-1})$ is *f-vector* of P. A polytope P is *2-neighborly* if any two vertices form an edge of P, i.e. $f_0(f_0 - 1) = 2f_1$. A convex polytope P is called a *0/1-polytope* if its set of vertices $X = \operatorname{ext}(P)$ is a subset of $\{0,1\}^d$. All 0/1-polytopes can be splitted into *0/1-equivalence classes* (*0/1-classes*) [3]. If two polytopes are 0/1-equivalent then they are combinatorially equivalent.

In the database, there are listed all 2-neighborly 0/1-polytopes of dimensions 6 and 7. For $d = 7$, we cann't provide the entire database explicitly, since it occupies about 1TB. Instead of this, we provide only 7-polytopes with the minimal and maximal number of facets (for every fixed number of vertices). f-vectors of these polytopes are listed in the file `7dminmax.fv`. However, all f-vectors of 2-neighborly 0/1-polytopes of dimension 7 are listed in the file `7d.fv`. If you want to work with the entire database, don't hesitate to send a request to maximenko.a.n@gmail.com

The database structure. The folder 6d has 11 subfolders with names 03v, ..., 13v. The number in subfolder's name is the number of vertices of a polytope. Every subfolder contains subsubfolders with names Nf, where N is the number of facets of a polytope. Thus, 6d\11v\023f contains descriptions of all polytopes with 11 vertices and 23 facets.

All polytopes are splitted by f-vectors. Every set of polytopes with the same f-vector occupies two files. For example, the file
6d\11v\023f\11v023f-0136-0167-0098
contains information of combinatorial types of 2-neighborly 0/1-polytopes with f-vector $(11, 55, 136, 167, 98, 23)$. The file
6d\11v\023f\11v023f-0136-0167-0098.01
contains all 0/1-classes of such polytopes.

The list of all f-vectors of 6-polytopes is in the file 6d.fv. Every line in the file is a short description of the set of polytopes with one f-vector. For example:

```
c4p1( 9 36 80 102 70 21)C8P15594
```

The parameter c4 means that the set contains a 4-simplicial polytope. (The set can also contains polytopes that are not 4-simplicial.) The parameter p1 means that it contains a pyramid. The p2 means that one of the facets (called *base*) of a polytope is adjacent with all the other facets, and 2 vertices do not belong to the facet (base). In parentheses, there is the f-vector. After C, there is the number of combinatorial types. After P, there is the number of 0/1-classes.

How to work with the database. The description of polytopes and their facet-vertex incidence matrices can be retrieved by the program read.c. The program works in command line mode. For example, the command

```
read 6d.fv -v 12 -all
```

will write into the file 6d.fv-12v.txt the list of all incidence matrices and all 0/1-classes of 2-neighborly 0/1-polytopes of dimension 6 with 12 vertices.

The first parameter must be the name of the file with f-vectors. (You can manually remove unwanted lines (f-vectors) from the file.) You can also use some additional parameters:

-v M — select only polytopes with M vertices.

-f N — select only polytopes with N facets.

`-fv` — write only f-vectors (lines from the input file).

`-all` — write all representatives for a combinatorial type (only one be written by default).

`-notinc` — don't write facet-vertex incidence matrix.

By default, for every f-vector there will be listed all combinatorial types. One combinatorial type will be represented by a facet-vertex incidence matrix and one 0/1-polytope (the list of vertices).

The format of files. For every f-vector there are two files: the file with 0/1-classes has extension `.01`, the file with combinatorial types has no extension. For example,
`6d\11v\023f\11v023f-0136-0167-0098.01`
and
`6d\11v\023f\11v023f-0136-0167-0098`

The first file has N_p records, where N_p is the number of 0/1-classes with the given f-vector. Every record is a 0/1-polytope wich is a representative of the appropriate 0/1-class. A 0/1-polytope with N_v vertices is written as N_v bytes, since every vertex is a 0/1-vector with d coordinates, $d \leq 7$. All 0/1-classes in the file ordered by the combinatorial type.

The second file has N_c records, where N_c is the number of combinatorial types with the given f-vector. One record has the following structure:

`[incidence matrix][N1][N2][polytope]`

The facet-vertex incidence matrix occupies N_f blocks, where N_f is the number of facets. One block is one row of the matrix. If the number of vertices $N_v \leq 16$ then one block occupies 2 bytes, otherwise — 4 bytes. The polytope occupies N_v bytes (the same as in `*.01`). `N1` and `N2` are the positions of the first and the last polytopes in the file `*.01` with this combinatorial type.

References

[1] O. Aichholzer. Extremal properties of 0/1-polytopes of dimension 5. In Polytopes – Combinatorics and Computation. Birkhauser, 2000, pp. 111–130.

[2] B. Grünbaum. Convex polytopes. Second edition. Springer, 2003.

[3] G.M. Ziegler. Lectures on 0/1-Polytopes. In Polytopes — Combinatorics and Computation. Birkhauser, 2000, pp. 1–41.