PROJECT DESCRIPTION: polymake

EWGENIJ GAWRILOW AND MICHAEL JOSWIG

The second author is partially supported by Deutsche Forschungsgemeinschaft, DFG Research Group "Polyhedral Surfaces."

Abstract. The mathematical software system polymake provides a wide range of functions for convex polytopes, simplicial complexes, and other objects.

1. Introduction

Polytope theory which lies in between applied fields, such as optimization, and more pure mathematics, including commutative algebra and toric algebraic geometry, invites to write software. When the polymake project started in 1996, there were already a number of systems around which could deal with polytopes in one way or another, e.g., convex hull codes such as cdd [11], lrs [3], porta [7], and qhull [5], but also visualization classics like Geomview [1]. The basic idea to polymake was – and still is – to make interfaces between any of these programs and to continue building further on top of the combined functionality. At the same time the gory technical details which help to accomplish such a thing should be entirely hidden from the user who does not want to know about it. On the outside polymake behaves somewhat similar to an expert system for polytopes: The user once describes a polytope in one of several natural ways and afterwards he or she can issue requests to the system to compute derived properties. In particular, there is no need to program in order to work with the system. On the other hand, for those who do want to program in order to extend the functionality even further, polymake offers a variety of ways to do so.

The modular design later, since version 2.0, allowed polymake to treat other mathematical objects in the same way. Mostly guided by the research of the second author the system was augmented by the TOPAZ application which deals with finite simplicial complexes. Because of the connections between polytope theory and combinatorial topology both parts of the system now benefit from each other.

Previous reports on the polymake system include the two papers [12, 13]; by now they are partially outdated. polymake is open source software which can be downloaded from http://www.math.tu-berlin.de/polymake for free. This text is an abridged version of [14].

Acknowledgments. Over the time many people made small and large contributions to the code. Most notably, Thilo Schröder and Nikolaus Witte are members of the development team since 2002.

Partial funding for the initial period of 1996–1997 of the polymake project came from the German-Israeli Foundation for Scientific Research and Development, grant I-0309-146.06/93 of Günter M. Ziegler. Later the Deutsche Forschungsgemeinschaft

Date: July 19, 2005.

(DFG) partially supported `polymake` within projects of the Sonderforschungsbereich 288 "Differentialgeometrie und Quantenphysik" and the DFG-Forschungszentrum Matheon.

2. Applications Overview

There are several different kinds of mathematical objects which `polymake` can deal with, most notably convex polytopes and finite simplicial complex.

As it was shown in the tutorial section above the system's behavior is driven by rules. In fact, the part of the system which takes care of applying rules to answer user requests is entirely independent of the mathematical objects and the algorithms. Each class of objects comes with its own set of rules. We survey the more technical aspects in Section 3. Here we focus on the mathematics.

2.1. **Convex Polytopes.** A convex polytope is the convex hull of finitely many points in $\mathbb{R}^d$; this is its *V-description*. A basic result in this area says that this notion coincides with those intersections of finitely many affine halfspaces in $\mathbb{R}^d$ which are bounded (*H-description*). This is sometimes referred to as the *Main Theorem on convex polytopes*. For an introduction to the theory, see Grünbaum [15] or Ziegler [21].

In order to deal with polytopes algorithmically often a first step is to apply an effective version of the "Main Theorem". While a polytope may naturally be given in its H-description (such as in linear programs) it is essential to also obtain a V-representation if one is interested in combinatorial properties. Algorithms which solve this problem are *convex hull algorithms*. Many such algorithms are known an implemented. The running-time that a particular algorithm/implementation requires is known to vary strongly with the class of polytopes which it is applied to; see Avis, Bremner, and Seidel [4] and also [16]. Therefore, `polymake` offers three different convex hull algorithms for the user to choose. There is one which is built into the system and two more, `cdd` [11] and `lrs` [3], respectively, which are accessible via interfaces. Actually, there also interfaces to `porta` [7] and `qhull` [5] available, but these are disabled by default. For each call it is possible to specify which algorithm to choose; additionally, the system can freely be configured to generally prefer one algorithm over another.

Convex polytopes have a metric geometry look as well as a combinatorial one. Both views are supported by `polymake`. What follows first is a list of metric properties which can be computed with the software.

- Gale transformations
- Steiner points
- projective linear transformations
- triangulations
- Voronoi diagrams and Delaunay cell decompositions in arbitrary dimension

Combinatorial properties include:

- fast face lattice construction algorithm; due to Kaibel and Pfetsch [18]
- f-vector, h-vector, flag-f-vector, and cd-index
- various graph-theoretic properties of the vertex-edge graph
- Altshuler determinant

In addition to these features there is a wide range of standard constructions and visualization functions; e.g., see Figure 1. There is also an interface to `Geomview` [1].

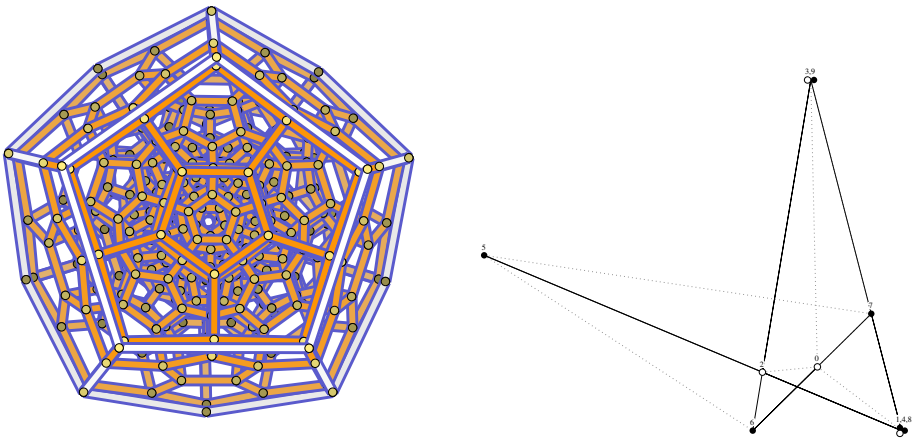

FIGURE 1. Schlegel diagram of the regular 120-cell (left) and a Gale diagram of a random 6-dimensional 01-polytope with 10 vertices (right).

2.2. **Finite Simplicial Complexes.** Given a finite *vertex set* V, a *simplicial complex* on V is a subset of 2^V which is closed with respect to taking subsets. Simplicial complexes form a basic combinatorial concept to capture properties of well-behaved topological spaces. In particular, this way certain parts of topology get within reach of effective methods.

A fundamental problem in topology is to decide whether two given spaces are *homeomorphic*, that is, indistinguishable from the topological point of view, or not. While it can be shown that this is algorithmically impossible – even for finite simplicial complexes representing 4-dimensional manifolds – it remains a key task to compute algebraic (homotopy) invariants.

`polymake` offers the following:

- simplicial homology and cohomology with integer coefficients
- cup and cap products
- Stiefel-Whitney characteristic classes
- intersection forms of 4-manifolds
- flip-heuristic by Björner and Lutz [6] for detecting spheres

In particular, in view of a celebrated result of Freedman [10], `polymake` is able to solve the homeomorphism problem for combinatorial 4-manifolds which are simply connected. See the survey [17] for some example computations.

2.3. **Extensions and Related Concepts.** The whole `polymake` system is extensible in several ways. Besides adding new functionality to the applications dealing with polytopes and simplicial complexes, it is possible to define entirely new classes of objects with an entirely new set of rules. For the more technical aspects such an extension the reader is referred to Section 3.

Here we list features which are already built into the system but which go beyond standard computations with polytopes or simplicial complexes.

2.3.1. *Tight Spans of Finite Metric Spaces.* Every tree T with non-negative weights on the edges defines a metric on the nodes of T. Conversely, it is easy to reconstruct

the tree from such a *tree-like* metric. The *phylogenetic problem* in computational biology boils down to the task to derive *a* sufficiently close tree from any given finite metric space. It is obvious that sometimes there is no tree at all which fits a given metric. Dress et al. [9, 8] devised *tight spans* as geometric objects which can be assigned to any finite metric space and which capture the deviation from a tree-like metric. Since tight spans can be described as bounded subcomplexes of unbounded polyhedra, `polymake`'s features can be exploited.

Sturmfels and Yu [20] recently used `TOPCOM` [19] and `polymake` to classify tight spans of metric spaces with at most six points.

2.3.2. *Curve Reconstruction.* If a sufficiently well distributed finite set S of points on a sufficiently smooth planar curve K is given, then it is possible to obtain a polygonal reconstruction of K. Amenta, Bern, and Eppstein [2] obtained a curve reconstruction procedure via an iterated Voronoi diagram computation. This beautiful algorithm is implemented in `polymake`; see Figure 2.

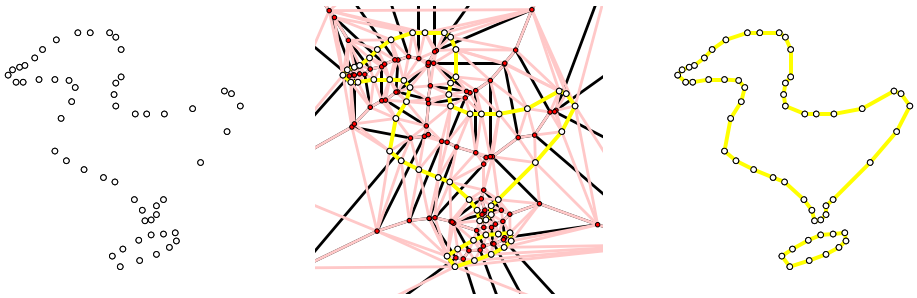

FIGURE 2. Planar curve(s) reconstructed from given points via the crust method of Amenta, Bern, and Eppstein [2].

3. Software Design

Since its first version from 1997 `polymake` was –at least partially– re-written several times. In spite of the many changes on the way, the core ideas always remained the same. The first goal was to have a flexible interface structure such that it is possible to interface to as many existing polytope processing software components (developed by other people) as possible. The second goal was scalability in the sense that the system should be useful both for programmers and mere users, and also both for students and expert scientists.

Feeling that one language is not enough for this, this resulted in an object-oriented hybrid design based on the two programming languages `C++` and `Perl`. The borderline is roughly defined as follows: The `Perl` side takes care of all the object management and their interfaces, while the `C++` half harbors the mathematical algorithms. Information exchange between both worlds is subject to a client-server scheme, `Perl` being the language of the server.

3.1. **Open Objects.** Convex polytopes are represented in the system as a class of objects which are defined by an extendible list of properties. In the current distributed version there are already more than one hundred of these properties defined; they range from the vertices (`VERTICES`) and facets (`FACETS`) of a polytope

to the list of Steiner points on all the faces (STEINER_POINTS) and the information whether or not the polytope is SIMPLICIAL or CUBICAL.

The client perspective (on the C++ side) is very restricted: The client asks the server for properties of some polytope object and leaves it entirely to the server to decide how these should be obtained. The Perl-written server has a list of rules which specify how to compute properties from the already known ones. For instance, there is a rule which explains how the facets can be computed for a polytope which was initially specified as the convex hull of finitely many points; that is, there is a convex-hull algorithm rule which computes FACETS from POINTS. Actually, as in this case it is the fact, there may be several competing rules computing the same. It is the task of the server to compile admissible sequences of rules (via a Dijkstra type algorithm for determining shortest weighted paths) to fulfill the user's (or the client's) requests from the information initially given.

It is fundamental to the design that the set of rules as well as the list of properties known to the system can be expanded and modified. Moreover, the object management is abstract, too; this way it is possible to define entirely new classes of objects and rule bases for them. For instance, simplicial complexes are objects different from polytopes (which actually includes pointed unbounded polyhedra), while tight spans are specializations of polytope objects since they can be described as the bounded subcomplexes of certain unbounded polyhedra; see Section 2.3.1.

3.2. **Scripting.** One way of using polymake is to generate large sets of polytopes and to filter them for individual members with specific properties. Such tasks are easily accomplished by often small Perl scripts which make use of polymake's object model.

As an example, the code below iterates through all the facets of a given polytope and offers a visualization of the vertex-edge graphs of all the facets; on the way combinatorially equivalent facets are detected and only one representative of each class is shown.

```
application 'polytope';

die "usage: polymake --script show_facets FILE\n" unless @ARGV;

my $p=load($ARGV[0]);
my @list=();

FACETS:
for (my $i=0; $i<$p->N_FACETS; ++$i) {
   my $facet=new Apps::polytope::RationalPolytope("facet #$i");
   Modules::client("facet", $facet, $p, $i, "-relabel");
   foreach my $other_facet (@list) {
      next FACETS if (check_iso($facet, $other_facet));
   }
   push @list, $facet;
}

static_javaview;
$_->VISUAL_GRAPH for @list;
```

This script `show_facets` is part of the distribution.

4. Technical Requirements

`polymake` can be used on UNIX systems only. It has been successfully tested on Linux, Sun Solaris, FreeBSD, MacOS X, IBM AIX and Tru64 Unix. Depending on the size of your objects `polymake` can run on small machines with, say, 128 MB of RAM. Only to compile the system from the source code at least 1 GB of RAM is required.

Our website at `http://www.math.tu-berlin.de/polymake` offers the full source code as well as several precompiled versions for download.

References

1. *GeomView, Version 1.8.1*, 2002, `http://www.geomview.org`.
2. Nina Amenta, Marshall Bern, and David Eppstein, *The crust and the beta-skeleton: combinatorial curve reconstruction*, Graphical Models and Image Processing **60** (1998), no. 2:2, 125–135.
3. David Avis, *lrslib, Version 4.2*, 2005, `http://cgm.cs.mcgill.ca/~avis/C/lrs.html`.
4. David Avis, David Bremner, and Raimund Seidel, *How good are convex hull algorithms?*, Comput. Geom. **7** (1997), no. 5-6, 265–301, 11th ACM Symposium on Computational Geometry (Vancouver, BC, 1995). MR MR1447243 (98c:52017)
5. C.B. Barber, D.P. Dobkin, and H.T Huhdanpaa, *qhull, Version 2003.1*, 2003, `http://www.qhull.org`.
6. Anders Björner and Frank H. Lutz, *Simplicial manifolds, bistellar flips and a 16-vertex triangulation of the Poincaré homology 3-sphere*, Experiment. Math. **9** (2000), no. 2, 275–289. MR 2001h:57026
7. Thomas Christof and Andreas Löbel, *PORTA - POlyhedron Representation Transformation Algorithm, Version 1.4.0*, 2004, `http://www.zib.de/Optimization/Software/Porta/`.
8. Andreas Dress, Katharina T. Huber, and Vincent Moulton, *An explicit computation of the injective hull of certain finite metric spaces in terms of their associated Buneman complex*, Adv. Math. **168** (2002), no. 1, 1–28. MR 2003g:54077
9. Andreas Dress, Vincent Moulton, and Werner Terhalle, *T-theory. An overview*, Sém. Lothar. Combin. **34** (1995), Art. B34b, approx. 23 pp. (electronic). MR 97i:57002
10. Michael Hartley Freedman, *The topology of four-dimensional manifolds*, J. Differential Geom. **17** (1982), no. 3, 357–453. MR 84b:57006
11. Komei Fukuda, *cddlib, Version 0.93d*, 2005, `http://www.ifor.math.ethz.ch/~fukuda/cdd_home/cdd.html`.
12. Ewgenij Gawrilow and Michael Joswig, *polymake: a framework for analyzing convex polytopes*, Polytopes—combinatorics and computation (Oberwolfach, 1997), DMV Sem., vol. 29, Birkhäuser, Basel, 2000, pp. 43–73. MR MR1785292 (2001f:52033)
13. ———, *polymake: an approach to modular software design in computational geometry*, Proceedings of the 17th Annual Symposium on Computational Geometry, ACM, 2001, June 3-5, 2001, Medford, MA, pp. 222–231.
14. ———, *Geometric reasoning with `polymake`*, 2005, `arXiv:math.CO/0507273`.
15. Branko Grünbaum, *Convex polytopes*, Pure and Applied Mathematics, vol. 16, Interscience Publishers, London, 1967, Second edition (Volker Kaibel, Victor Klee, and Günter M. Ziegler, eds.), Graduate Texts in Mathematics **221**. Springer-Verlag, New York, NY, 2003.
16. Michael Joswig, *Beneath-and-beyond revisited*, Algebra, geometry, and software systems, Springer, Berlin, 2003, pp. 1–21. MR MR2011751 (2004k:68169)
17. Michael Joswig, *Computing Invariants of Simplicial Manifolds*, 2004, `arXiv:math.AT/0401176`.
18. Volker Kaibel and Marc E. Pfetsch, *Computing the face lattice of a polytope from its vertex-facet incidences*, Comput. Geom. **23** (2002), no. 3, 281–290. MR MR1927137 (2003h:52019)
19. Jörg Rambau, *TOPCOM, Version 0.13.2*, 2004, `http://www.uni-bayreuth.de/departments/wirtschaftsmathematik/rambau/TOPCOM/`.

20. Bernd Sturmfels and Josephine Yu, *Classification of six-point metrics*, Electron. J. Combin. **11** (2004), Research Paper 44, 16 pp. (electronic). MR MR2097310
21. Günter M. Ziegler, *Lectures on polytopes*, Graduate Texts in Mathematics, vol. 152, Springer-Verlag, New York, NY, 1995, Revised edition, 1998.

E-mail address: gawrilow@math.tu-berlin.de, joswig@mathematik.tu-darmstadt.de

Ewgenij Gawrilow, Institut für Mathematik, MA 6-1, TU Berlin, 10623 Berlin, Germany, Michael Joswig, Fachbereich Mathematik, AG 7, TU Darmstadt, 64289 Darmstadt, Germany