Polyhedral Computation
Today's Topic: The Double Description Algorithm

Komei Fukuda

Swiss Federal Institute of Technology

Zurich

October 29, 2010

Convexity Review: Farkas-Type Alternative Theorems

Gale's Theorem.

For any $A \in \mathbb{R}^{m \times d}$ and $b \in \mathbb{R}^m$, exactly one of (a) and (b) holds:

(a) $\exists x \in \mathbb{R}^d$ such that $A\, x \leq b$;

(b) $\exists z \in \mathbb{R}^m$ such that $z \geq \mathbf{0}$, $z^T A = \mathbf{0}$ and $z^T b < 0$.

Various forms of a pair (a) and (b):

Farkas (1894)

(a) $\exists x : A\, x = b$ and $x \geq \mathbf{0}$;

(b) $\exists z : z^T A \geq \mathbf{0}$ and $z^T b < 0$.

Gordan (1973)

(a) $\exists x : A\, x = \mathbf{0}$ and $x \gneq \mathbf{0}$;

(b) $\exists z : z^T A > \mathbf{0}$.

Symmetric Form (V is a linear subspace of $\mathbb{R}^d$, $g \in [d]$ is fixed.)

(a) $\exists x \in V$ such that $x \geq \mathbf{0}$ and $x_g > 0$;

(b) $\exists y \in V^\perp$ such that $y \geq \mathbf{0}$ and $y_g > 0$.

Constructive Proofs for MW: MW in Various Forms

Theorem 2.1. [Minkowski-Weyl's Theorem for Polyhedra]
For $P \subseteq \mathbb{R}^d$, the following statements are equivalent:

(a) P is an H-polyhedron, i.e., $\exists A$ & $\exists b$ s.t. $P = \{x : Ax \geq b\}$;

(b) P is a V-polyhedron, i.e., $\exists v_i$'s & $\exists r_j$'s s.t.
$P = conv\{v_1, \ldots, v_n\} + nonneg\{r_1, \ldots, r_s\}$.

(a) $\Longrightarrow$ (b): Minkowski's Theorem

(b) $\Longrightarrow$ (a): Weyl's Theorem

Theorem 2.2. [Minkowski-Weyl's Theorem for Cones]
For $P \subseteq \mathbb{R}^d$, the following statements are equivalent:

(a) P is an H-cone, i.e., $\exists A$ s.t. $P = \{x : Ax \geq 0\}$;

(b) P is a V-cone, i.e., $\exists r_j$'s s.t. $P = nonneg\{r_1, r_2, \ldots, r_s\}$.

Constructive Proofs for MW: MW in Various Forms

Theorem 2.2. [MW for Cones in Matrix Form]
For $P \subseteq \mathbb{R}^d$, the following statements are equivalent:

(a) P is an H-cone, i.e., $\exists A$ s.t. $P = \{x : Ax \geq 0\}$;

(b) P is a V-cone, i.e., $\exists R$ s.t. $P = \{x : x = R\,\lambda, \lambda \geq 0\}$.

A pair (A, R) is a double description pair or simply a DD pair if

$$Ax \geq 0 \quad \Longleftrightarrow \quad x = R\,\lambda \text{ for some } \lambda \geq 0.$$

Theorem 2.6. [Minkowski for Cones]
$\forall A$, $\exists R$ such that (A, R) is a DD pair.

Theorem 2.7. [Weyl for Cones]
$\forall R$, $\exists A$ such that (A, R) is a DD pair.

Constructive Proofs for MW: Key Proposition

Proposition 2.8. [DD Pair Duality] For any A and R,
(A, R) is a DD pair $\Longleftrightarrow$ (R^T, A^T) is a DD pair.

Corollary. Minkowski $\Longleftrightarrow$ Weyl.
(In practical terms, one needs to implement one conversion.)

First Proof of the MW:
Let a matrix R (i.e. V-cone) is given. By the Fourier-Motzkin Elimination, we can eliminate all variables λ from the system $x = R\,\lambda, \lambda \geq 0$. The resulting system of inequalities written as $Ax \geq \mathbf{0}$ gives A. This proves the Weyl's theorem.

By Proposition 2.8, Minkowski's Theorem is true. ■

This provides neither a polynomial nor a compact algorithm.

Minkowski via the Double Description Algorithm (2.1.3)

Suppose an $m \times d$ matrix A is given and let

$$P(A) = \{x : Ax \geq 0\}.$$

Let $K \subset \{1, 2, \ldots, m\}$ be a subset of the row indices of A and let A_K denote the submatrix of A.

Suppose we already found a generating matrix R for $P(A_K)$, or equivalently (A_K, R) is a DD pair.

Key Incremental Step: Select any row index i not in K and construct a DD pair (A_{K+i}, R') using the computed DD pair (A_K, R).

Remark: The double description algorithm is due to Motzkin-Raiffa-Thompson-Thrall (1953).

Minkowski via the Double Description Algorithm (2.1.3)

Partition the column index set J of R into three parts:

$$\begin{aligned} J^+ &= \{j \in J : \ A_i \ r_j > 0\} \\ J^0 &= \{j \in J : \ A_i \ r_j = 0\} \\ J^- &= \{j \in J : \ A_i \ r_j < 0\}. \end{aligned}$$

Lemma 2.9. [Main Lemma for Double Description Method]
The pair (A_{K+i}, R') is a DD pair, where R' is the $d \times |J'|$ matrix with columns r_j ($j \in J'$) defined by

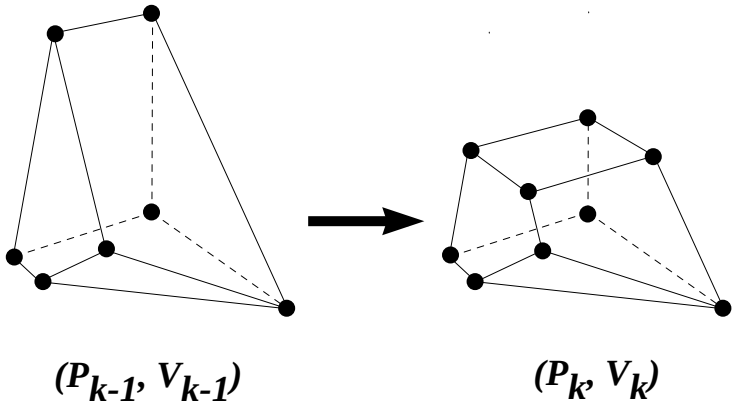
$$\begin{aligned} J' &= J^+ \cup J^0 \cup (J^+ \times J^-), \text{ and} \\ r_{jj'} &= (A_i \ r_j) r_{j'} - (A_i \ r_{j'}) r_j \ \text{ for each } (j, j') \in J^+ \times J^-. \end{aligned}$$

The Double Description Method: Complexity?

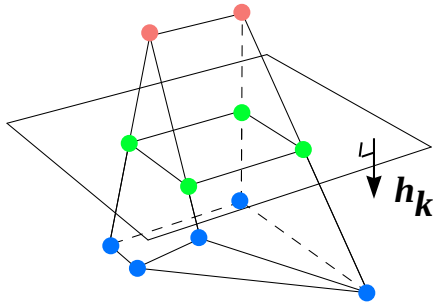
P : an H-polytope represented by m halfspaces $h_1, \ldots, h_m$ in $\mathbb{R}^d$.

$P_k = \cap_{i=1}^{k} h_i$: kth polytope ($P = P_m$).

$V_k = V(P_k)$: the vertex set computed at kth step.



(P_{k-1}, V_{k-1}) ***(P_k, V_k)***

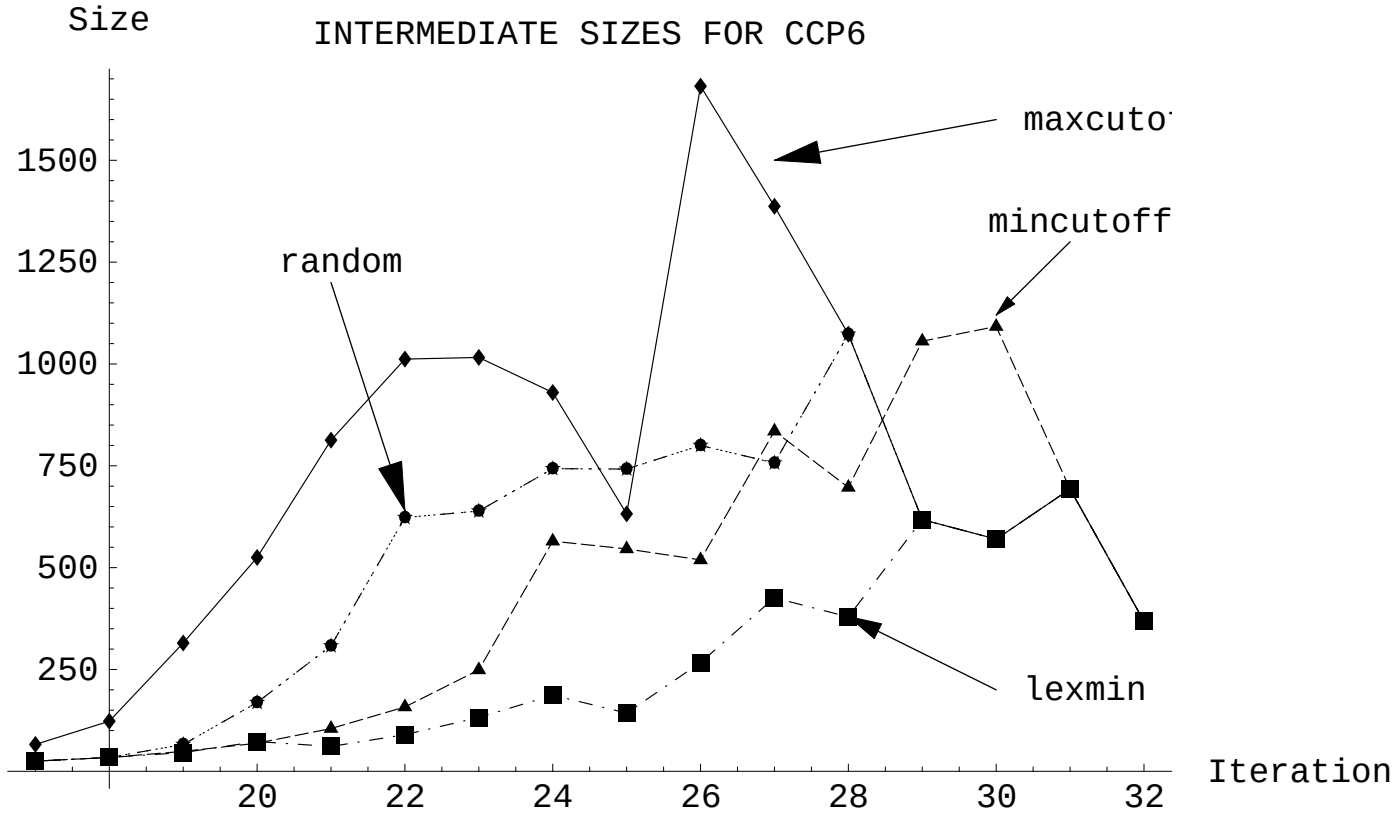


newly generated for each adj pair (,)

The Double Description Method: Complexity? (cont.)

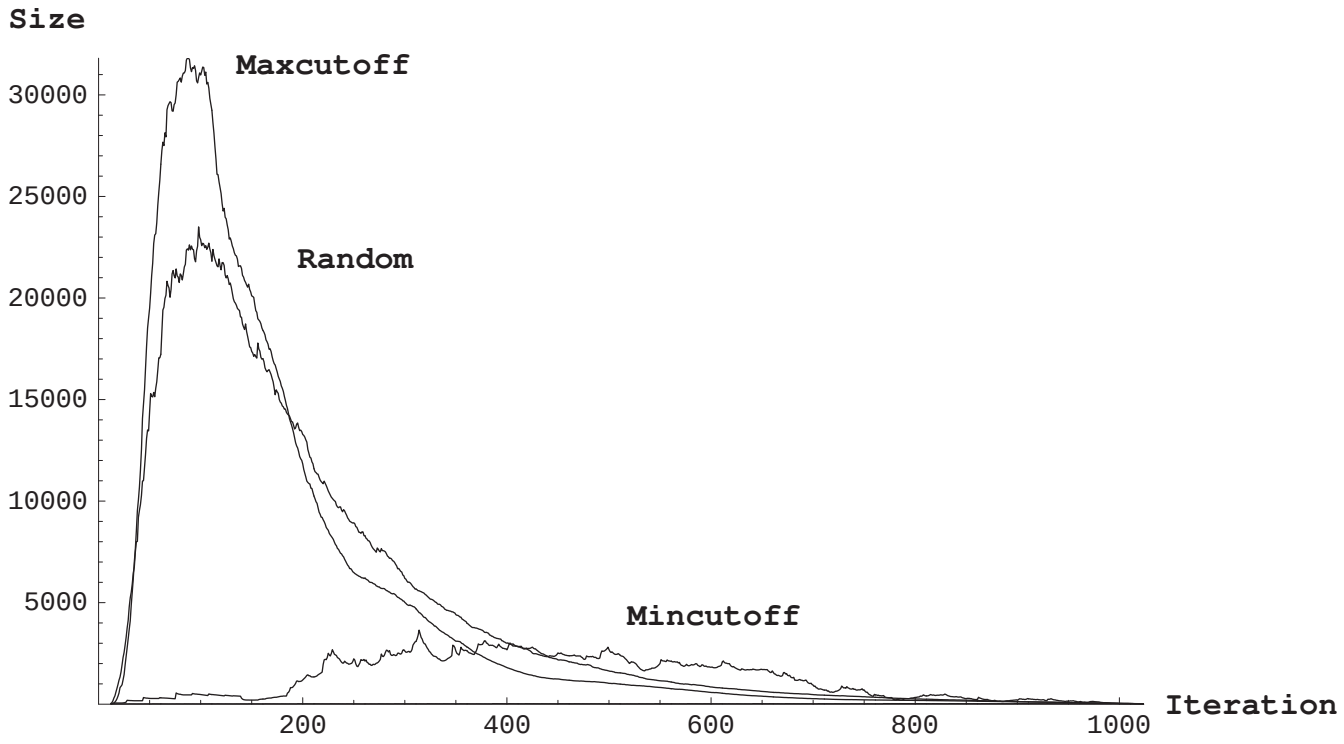
- It is an incremental method, dual to the Beneath-Beyond Method.
- Practical for low dimensions and highly degenerate inputs.
- For highly degenerate inputs, the sizes of intermediate polytopes are very sensitive to the ordering of halfspaces. For example, the maxcutoff ordering ("the deepest cut") may provoke extremely high intermediate sizes.
- It is hard to estimate its complexity in terms of and the sizes of input and output. The main reason is that the intermediate polytopes P_k can become very complex relative to the original polytope $P = P_m$.
- D. Bremner (1999) proved that there is a class of polytopes for which the double description method (and the beneath-beyond) method is exponential.

How Intermediate Sizes Fluctuate with Different Orderings



The input is a 15-dimensional polytope with 32 facets. The output is a list of 368 vertices. The lexmin is a sort of shelling ordering.

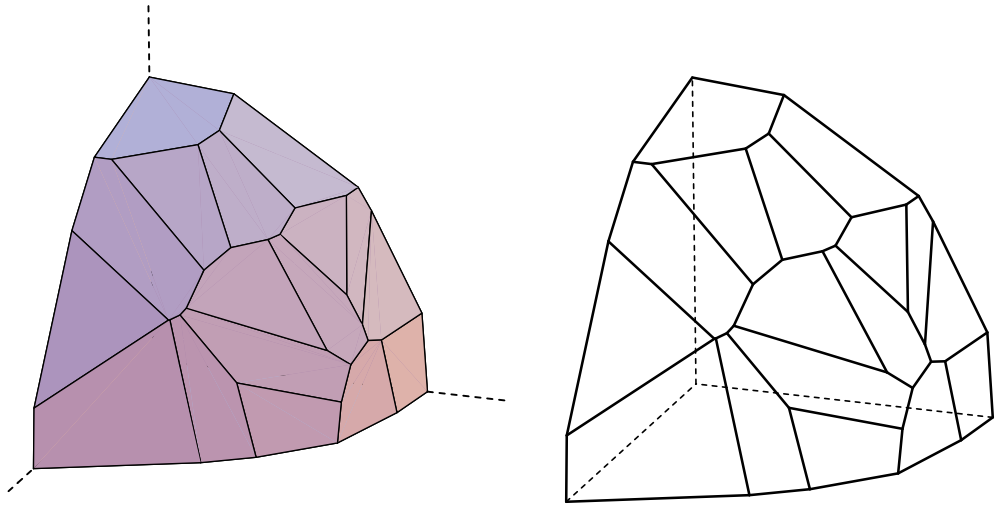
How Intermediate Sizes Fluctuate with Different Orderings



The input is a 10-dimensional cross polytope with 2^{10} facets. The output is a list of 20 vertices. The highest peak is attained by maxcutoff ordering, following by random and mincutoff. Lexmin is the best among all and the peak intermediate size is less than 30. (Too small too see it above.)

Pivoting Algorithms for Vertex Enumeration

Basic Idea: Search the connected graph of an H-polytope P by pivoting operations to list all vertices.



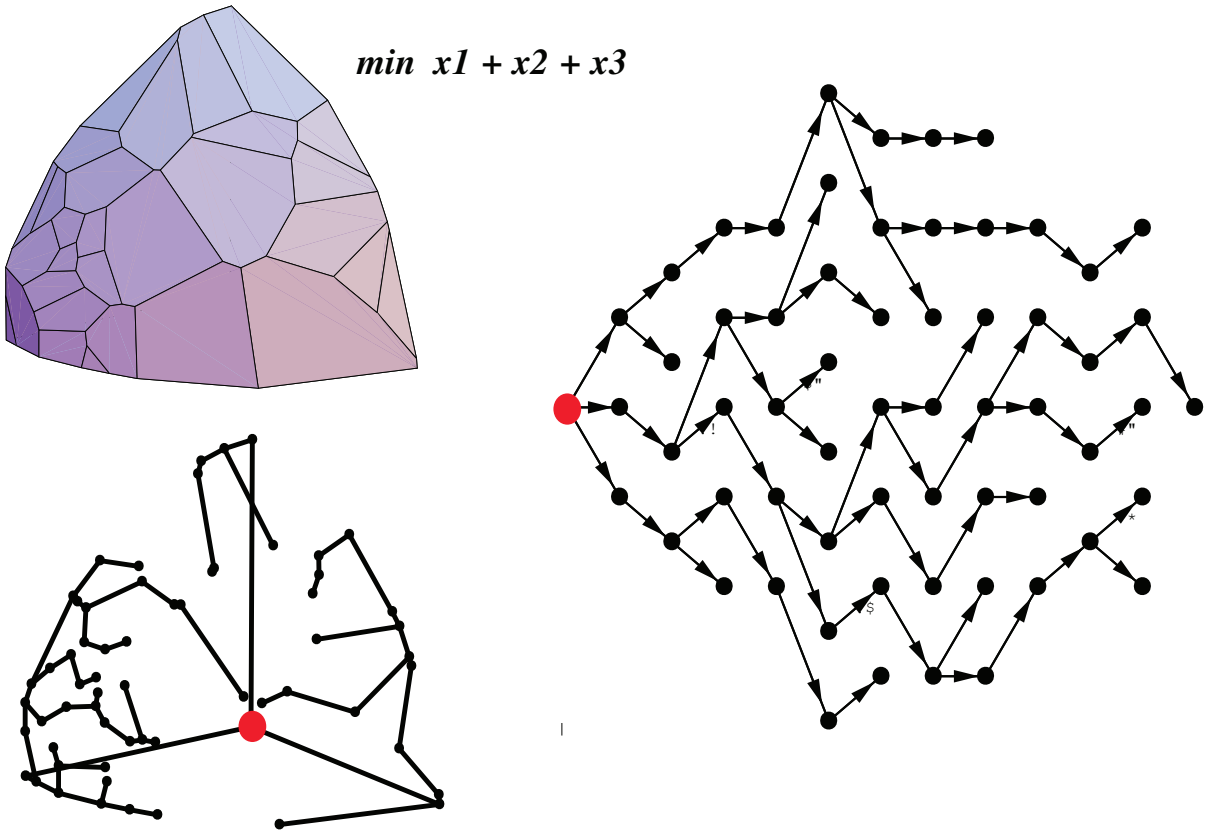
A polytope P and its graph (1-skeleton)

Advantage: Under the usual nondegeneracy (i.e. no points in P lie on more than d facets), it is polynomial in the input size and the output size.

Space Complexity: Depends on the search technique. The standard depth-first search requires to store all vertices found.

Memory Free: Reverse Search for Vertex Enumeration

Key idea: Reverse the simplex method from the optimal vertex in all possible ways:



Complexity: $O(mdf_0)$-time and $O(md)$-space (under nondegeneracy).

Reverse Search: General Description

Two functions f and adj define the search:

A finite local search f for a graph $G = (V, E)$ with a special node $s \in V$ is a function: $V \setminus \{s\} \to V$ satisfying

(L1) $\{v, f(v)\} \in E$ for each $v \in V \setminus \{s\}$, and

(L2) for each $v \in V \setminus \{s\}$, $\exists k > 0$ such that $f^k(v) = s$.

Example:

- Let $P = \{x \in \mathbb{R}^d : A\ x \leq b\}$ be a simple polytope, and $c^T x$ be any generic linear objective function. Let V be the set of all vertices of P, s the unique optimal, and $f(v)$ be the vertex adjacent to v selected by the (deterministic) simplex method.

Reverse Search: General Description

A adjacency oracle adj for a graph $G = (V, E)$ is a function (where δ a upper bound for the maximum degree of G) satisfying:

(i) for each vertex v and each number k with $1 \leq k \leq \delta$ the oracle returns $adj(v, k)$, a vertex adjacent to v or extraneous 0 (zero),

(ii) if $adj(v, k) = adj(v, k') \neq 0$ for some $v \in V$, k and k', then $k = k'$,

(iii) for each vertex v, $\{adj(v, k) : adj(v, k) \neq 0, 1 \leq k \leq \delta\}$ is exactly the set of vertices adjacent to v.

Example:

- Let $P = \{x \in \mathbb{R}^d : A\ x \leq b\}$ be a simple polytope. Let V be the set of all vertices of P, δ be the number of nonbasic variables and $adj(v, k)$ be the vertex adjacent to v obtained by pivoting on the kth nonbasic variable at v.

Reverse Search: General Description

```
    procedure ReverseSearch(adj,δ,s,f);
        v := s; j := 0; (* j: neighbor counter *)
        repeat
            while j < δ do
                j := j + 1;
(r1)            next := adj(v, j);
                if next ≠ 0 then
(r2)                if f(next) = v then  (* reverse traverse *)
                    v := next; j := 0
                    endif
                endif
            endwhile;
            if v ≠ s then (* forward traverse *)
(f1)            u := v;    v := f(v);
(f2)            j := 0;    repeat j := j + 1 until adj(v, j) = u (* restore j *)
            endif
        until v = s and j = δ
```

Pivoting Algorithm vs Incremental Algorithm

- Pivoting algorithms, in particular the reverse search algorithm (lrs, lrslib), work well for high dimensional cases.
- Incremental algorithms work well for low (up to 10) dimensional cases and highly degenerate cases. For example, the codes cdd/cddlib and porta are implemented for highly degenerate cases and the code qhull for low (up to 10) dimensional cases.
- The reverse search algorithm seems to be the only method that scales very efficiently in massively parallel environment.
- Various comparisons of representation conversion algorithms and implementations can be found in the excellent article:

  D. Avis, D. Bremner, and R. Seidel. How good are convex hull algorithms. Computational Geometry: Theory and Applications, 7:265–302, 1997.

References

[1] D. Avis and K. Fukuda. A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra. Discrete Comput. Geom., 8:295–313, 1992.

[2] D. Bremner. Incremental convex hull algorithms are not output sensitive. Discrete Comput. Geom., 21:57–68, 1999.

[3] K. Fukuda. cdd, cddplus and cddlib homepage. ETH Zurich. http://www.ifor.math.ethz.ch/~fukuda/cdd_home/index.html.

[4] K. Fukuda and A. Prodon. Double description method revisited. In M. Deza, R. Euler, and I. Manoussakis, editors, Combinatorics and Computer Science, volume 1120 of Lecture Notes in Computer Science, pages 91–111. Springer-Verlag, 1996.

[5] T.S. Motzkin, H. Raiffa, GL. Thompson, and R.M. Thrall. The double description method. In H.W. Kuhn and A.W.Tucker, editors, Contributions to Theory of Games, Vol. 2. Princeton University Press, Princeton, RI, 1953.