Vertex enumeration for polytopes defined by oracles

Vissarion Fisikopoulos

Joint work with I.Z. Emiris (UoA), B. Gärtner (ETHZ)

Dept. of Informatics & Telecommunications, University of Athens

CGL final workshop, 02.Oct.2013
Outline

Polytopes & Oracles

Vertex enumeration in the oracle model
  Total polynomial time algorithm / known edge directions
  Beneath-and-Beyond based algorithm / triangulation producing
  Face-lattice producing algorithm
Outline

Polytopes & Oracles

Vertex enumeration in the oracle model
    Total polynomial time algorithm / known edge directions
    Beneath-and-Beyond based algorithm / triangulation producing
    Face-lattice producing algorithm
A convex polytope $P \subseteq \mathbb{R}^d$ can be represented as the
1. convex hull of a pointset \( \{p_1, \ldots, p_n\} \) (V-representation)
2. intersection of halfspaces \( \{h_1, \ldots, h_m\} \) (H-representation)

▶ These problems are equivalent by polytope duality.
Algorithmic Issues

- For general dimension $d$ there is no polynomial algorithm for the convex hull (or vertex enumeration) problem since $m$ can be $O\left(n^{\lfloor d/2 \rfloor}\right)$ [McMullen’70].

- It is open whether there exist a total poly-time algorithm for the convex hull (or vertex enumeration) problem, i.e. runs in poly-time in $n, m, d$. 
Polytope Oracles

Implicit representation for a polytope $P \subseteq \mathbb{R}^d$.

$\text{OPT}_P$: Given direction $c \in \mathbb{R}^d$ return the vertex $v \in P$ that maximizes $c^T v$.

$\text{SEP}_P$: Given point $y \in \mathbb{R}^d$, return yes if $y \in P$ otherwise a hyperplane $h$ that separates $y$ from $P$. 
Well-described polytopes and oracles

Definition
A rational polytope $P \subseteq \mathbb{R}^d$ is well-described (with a parameter $\varphi$) if there exists an H-representation for $P$ in which every inequality has encoding length at most $\varphi$. The encoding length of $P$ is $\langle P \rangle = d + \varphi$.

Proposition (Grötschel et al.’93)
For a well-described polytope, we can compute $\text{OPT}_P$ from $\text{SEP}_P$ (and vice versa) in oracle polynomial-time. The runtime (polynomially) depends on $d$ and $\varphi$. 
Why oracles?

- Polynomial time algorithms for combinatorial optimization problems using the ellipsoid method [Grötschel-Lovász-Schrijver’93]

- Randomized polynomial-time algorithm for approximating the volume of convex bodies [Dyer-Frieze-Kannan ’90]
Original Motivation

(1) Secondary, Resultant polytopes

- Vertices → regular triangulations of a pointset’s convex hull
- $\text{OPT}_P$ is available via a triangulation computation
  \cite{Emiris-F-Konaxis-Penaranda12}

(2) Minkowski sums

- Applications in Computational Algebraic Geometry, Geometric Modelling, Combinatorics
Outline

Polytopes & Oracles

Vertex enumeration in the oracle model
- Total polynomial time algorithm / known edge directions
- Beneath-and-Beyond based algorithm / triangulation producing
- Face-lattice producing algorithm
Main problems

Vertex enumeration in the oracle model
Given $\text{OPT}_P$ for $P \subseteq \mathbb{R}^d$, compute the vertices of $P$.

Vertex enumeration in the oracle model with edge-directions
Given $\text{OPT}_P$ and a superset $D$ of the edge directions $D(P)$ of $P \subseteq \mathbb{R}^d$, compute the vertices of $P$. 
Outline

Polytopes & Oracles

Vertex enumeration in the oracle model
  Total polynomial time algorithm / known edge directions
  Beneath-and-Beyond based algorithm / triangulation producing
  Face-lattice producing algorithm
Vertex enumeration with edge-directions

Given $\text{OPT}_P$ and a superset $D$ of the edge directions $D(P)$ of $P \subseteq \mathbb{R}^d$, compute the vertices $P$.

**Proposition (Rothblum-Onn ’07)**

Let $P \subseteq \mathbb{R}^d$ given by $\text{OPT}_P$, and $D \supseteq D(P)$. All vertices of $P$ can be computed in

$$O(|D|^{d-1}) \text{ calls to } \text{OPT}_P + O(|D|^{d-1}) \text{ arithmetic operations.}$$
The edge-skeleton algorithm

Input:
- OPT_\text{P}
- Edge vec. P (dir. & len.): D

Output:
- Edge-skeleton of P

Sketch of Algorithm:
- Compute a vertex of P (x = OPT_\text{P}(c) for arbitrary c^T \in \mathbb{R}^d)
The edge-skeleton algorithm

Input:
- OPT<sub>P</sub>
- Edge vec. P (dir. & len.): D

Output:
- Edge-skeleton of P

Sketch of Algorithm:
- Compute a vertex of P (x = OPT<sub>P</sub>(c) for arbitrary c<sup>T</sup> ∈ ℝ<sup>d</sup>)
- Compute segments S = {(x, x + d), for all d ∈ D}
The edge-skeleton algorithm

Input:
- \( \text{OPT}_P \)
- Edge vec. \( P \) (dir. & len.): \( D \)

Output:
- Edge-skeleton of \( P \)

Sketch of **Algorithm**:
- Compute a vertex of \( P \) \( \langle \mathbf{x} = \text{OPT}_P(\mathbf{c}) \rangle \) for arbitrary \( \mathbf{c}^T \in \mathbb{R}^d \)
- Compute segments \( S = \{(\mathbf{x}, \mathbf{x} + \mathbf{d}), \text{for all } \mathbf{d} \in D\} \)
- Remove from \( S \) all segments \( (\mathbf{x}, \mathbf{y}) \) s.t. \( \mathbf{y} \notin P \) \( \langle \text{OPT}_P \rightarrow \text{SEP}_P \rangle \)
The edge-skeleton algorithm

Input:
- OPT\(P\)
- Edge vec. \(P\) (dir. & len.): \(D\)

Output:
- Edge-skeleton of \(P\)

Sketch of **Algorithm**:
- Compute a vertex of \(P\) \((x = \text{OPT}_P(c)\) for arbitrary \(c^T \in \mathbb{R}^d\))
- Compute segments \(S = \{(x, x + d), \text{ for all } d \in D\}\)
- Remove from \(S\) all segments \((x, y)\) s.t. \(y \notin P\) \((\text{OPT}_P \rightarrow \text{SEP}_P)\)
- Remove from \(S\) the segments that are not extreme
The edge-skeleton algorithm

Input:
- OPT\(_P\)
- Edge vec. \(P\) (dir. & len.): \(D\)

Output:
- Edge-skeleton of \(P\)

Sketch of **Algorithm**:
- Compute a vertex of \(P\) \((x = OPT_{P}(c)\) for arbitrary \(c^T \in \mathbb{R}^d\))
- Compute segments \(S = \{(x, x + d), \text{ for all } d \in D\}\)
- Remove from \(S\) all segments \((x, y)\) s.t. \(y \notin P\) (\(OPT_{P} \rightarrow SEP_{P}\))
- Remove from \(S\) the segments that are not extreme

Can be altered to work with edge directions only
Complexity

Theorem
Given $OPT_P$ and a superset of edge directions $D$ of a well-described polytope $P \subseteq \mathbb{R}^d$, the edge skeleton of $P$ can be computed in oracle total polynomial-time

$$O \left( n \ |D| \ T + n \ \mathbb{LP}(d^3|D|\langle B \rangle) \right),$$

- $n$ the number of vertices of $P$,
- $T$: runtime of oracle conversion algorithm for $P$ and $D$,
- $\langle B \rangle$ is the binary encoding length of the vector set $D$ and $P$,
- $\mathbb{LP}(\langle A \rangle + \langle b \rangle + \langle c \rangle)$ runtime of max $c^T x$ over $\{x : Ax \leq b\}$. 
Applications

Corollary

The edge skeleton of resultant, secondary polytopes can be computed in oracle total polynomial-time.

Corollary

The edge skeletons of polytopes appearing in convex combinatorial optimization [Rothblum-Onn ’04] and convex integer programming [De Loera et al. ’09] problems can be computed in oracle total polynomial-time.
Outline

Polytopes & Oracles

Vertex enumeration in the oracle model
  Total polynomial time algorithm / known edge directions
  Beneath-and-Beyond based algorithm / triangulation producing
  Face-lattice producing algorithm
Vertex enumeration in the oracle model

Beneath-and-Beyond based Algorithm (triangulation producing)

- [Emiris-F-Konaxis-Peñaranda ’12] for resultant polytopes
- first: compute conv.hull of $d + 1$ aff. indep. vertices of $P$
- step: call $\text{OPT}_P$ with outer normal vector of a halfspace
  → either validate this halfspace
  → or add a new vertex to the convex hull

\[ Q \]
\[ N(R) \]
Vertex enumeration in the oracle model

Beneath-and-Beyond based Algorithm (triangulation producing)

- [Emiris-F-Konaxis-Peñaranda '12] for resultant polytopes
- first: compute conv.hull of $d + 1$ aff. indep. vertices of $P$
- step: call $\text{OPT}_P$ with outer normal vector of a halfspace
  $\rightarrow$ either validate this halfspace
  $\rightarrow$ or add a new vertex to the convex hull

Complexity

Given $P \subseteq \mathbb{R}^d$, H-, V-repr. & triang. $T$ of $P$ can be computed in

$$O(d^5ns^2)$$ arithmetic operations $+$ $O(n + m)$ calls to $\text{OPT}_P$

$s$ is the number of cells of $T$. 
Vertex enumeration in the oracle model

Beneath-and-Beyond based Algorithm (triangulation producing)

- [Emiris-F-Konaxis-Peñaranda '12] for resultant polytopes
- first: compute conv.hull of $d + 1$ aff. indep. vertices of $P$
- step: call $\text{OPT}_P$ with outer normal vector of a halfspace
  $\rightarrow$ either validate this halfspace
  $\rightarrow$ or add a new vertex to the convex hull

Complexity

Given $P \subseteq \mathbb{R}^d$, H-, V-repr. & triang. $T$ of $P$ can be computed in

$O(d^5 ns^2)$ arithmetic operations $+ O(n + m)$ calls to $\text{OPT}_P$

$s$ is the number of cells of $T$.

**BUT:** $s$ can be $O\left(n^{\lfloor d/2 \rfloor}\right)$
Outline

Polytopes & Oracles

Vertex enumeration in the oracle model
  Total polynomial time algorithm / known edge directions
  Beneath-and-Beyond based algorithm / triangulation producing
  Face-lattice producing algorithm
Vertex enumeration in the oracle model

Recursive algorithm (face-lattice producing)

- For $P \subseteq \mathbb{R}^d$ define $K_v$ the intersection of normal cone of vertex $v$ with hyperplane $H$ s.t. $\dim(K_v) = d - 1$.
- Given vertex $v \in P \subseteq \mathbb{R}^d$ and $\text{OPT}_P$ compute in oracle polynomial time $\text{OPT}_{K_v}$ for $K_v \subseteq \mathbb{R}^{d-1}$. 
Polytope Volume Approximation

- Volume approximation of $P$ reduces to uniform sampling from $P$ [Dyer et.al '91]
- Random walks on convex bodies (e.g. hit-&-run)
- Assume membership (weak case of SEP) oracle

Ongoing work

- Study case of OPT oracle
- Implementation of both cases (MEM and OPT oracles)
Thank you!