Complexity of Approximating the Vertex Centroid of a Polyhedron[*]

Khaled Elbassioni[†]   Hans Raj Tiwary[‡]

March 5, 2011

Abstract

Let $\mathcal{P}$ be an $\mathcal{H}$-polytope in $\mathbb{R}^d$ with vertex set V. The vertex centroid is defined as the average of the vertices in V. We first prove that computing the vertex centroid of an $\mathcal{H}$-polytope, or even just checking whether it lies in a given halfspace, is #P-hard. We also consider the problem of approximating the vertex centroid by finding a point within an ϵ distance from it and prove this problem to be #P-easy in the sense that it can be solved efficiently using an oracle for some #P-complete problem. In particular we show that given an oracle for counting the number of vertices of an $\mathcal{H}$-polytope, one can approximate the vertex centroid in polynomial time. Counting the number of vertices of a polytoped defined as the intersection of halfspaces is known to be #P-complete. We also show that any algorithm approximating the vertex centroid to *any* "sufficiently" non-trivial (for example constant) distance, can be used to construct a fully polynomial-time approximation scheme for approximating the centroid and also an output-sensitive polynomial algorithm for the Vertex Enumeration problem. Finally, we show that for unbounded polyhedra the vertex centroid can not be approximated to a distance of $d^{\frac{1}{2}-\delta}$ for any fixed constant $\delta > 0$ unless $P = NP$.

1 Introduction

An intersection of a finite number of closed halfspaces in $\mathbb{R}^d$ defines a polyhedron. A polyhedron can also be represented as $conv(V) + cone(Y)$, the Minkowski sum of the convex hull of a finite set of points V and the cone of a finite set of rays. A bounded polyhedron is called a polytope. In what follows, we will discuss mostly polytopes for simplicity and refer to the unbounded case explicitly only towards the end. We call a polytope (polyhedron resp.)

---

defined by a set of inequalities an $\mathcal{H}$-polytope ($\mathcal{H}$-polyhedron resp.) and a polytope (polyhedron resp.) defined by vertices (and extreme rays) a $\mathcal{V}$-polytope (polyhedron respectively).

Let $\mathcal{P}$ be an $\mathcal{H}$-polytope in the ambient space $\mathbb{R}^d$ with vertex set V. Various notions try to capture the essence of a "center" of a polytope. Perhaps the most popular notion is that of the center of gravity of $\mathcal{P}$. Recently Rademacher proved that computing the center of gravity of a polytope is #P-hard [8]. The proof essentially relies on the fact that the center of gravity captures the volume of a polytope perfectly and that computing the volume of a polytope is #P-hard [4]. Note that, randomized polynomial algorithms exist that approximate the volume of a polytope within any arbitrary factor [5]. It is also easy to see that the center of gravity can be approximated by simply sampling random points from the polytope, the number of samples depending polynomially on the desired approximation (See Algorithm 5.8 of [5]).

In this paper we study a variant of the notion of "center" defined as the centroid (average) of the vertices of P. Despite being quite a natural feature of polytopes, this variant seems to have received very little attention both from theoretical and computational perspectives. Throughout this paper we will refer to the vertex centroid just as centroid. The reader should note that in popular literature the word centroid refers more commonly to the center of gravity. We nevertheless use the same terminology for simplicity of language. Our motivation for studying the centroid stems from the fact that the centroid encodes the number of vertices of a polytope. As we will see, this also makes computing the centroid hard.

The parallels between centroid and the center of gravity of a polytope mimic the parallels between the number of vertices and the volume of a polytope. Computing the volume is #P-complete [4] but it can be approximated quite well [5]. Accordingly, the problem of computing the corresponding centroid is hard ([8], Theorem 1) but the volume centroid can be approximated quite well [5]. On the other hand computing the number of vertices is not only #P-complete [3, 7], it can not be approximated within any factor polynomial in the number of facets and the dimension. As we will see in this paper, computing the vertex centroid of an $\mathcal{H}$-polytope exactly is #P-hard. Even approximating the vertex centroid for unbounded $\mathcal{H}$-polyhedra turns out to be NP-hard. We do not know the complexity of approximating the vertex centroid of an $\mathcal{H}$-polytope (bounded case).

The problem of enumerating vertices of an $\mathcal{H}$-polytope has been studied for a long time. However, in spite of years of research it is neither known to be hard nor is there an output sensitive polynomial algorithm for it. Note that the problem of enumerating all vertices of an $\mathcal{H}$-polytope is different from the problem of counting the number of vertices. While the latter problem is known to be #P-complete [3, 7], the complexity status of Vertex Enumeration is open [1]. A problem that is polynomially equivalent to the Vertex Enumeration problem is to decide if a given list of vertices of an $\mathcal{H}$-polytope is complete [1]. In this paper we show that any algorithm that approximates the centroid of an arbitrary polytope to any "sufficiently" non-trivial distance can be used to

obtain an output sensitive polynomial algorithm for the Vertex Enumeration problem.

The main results of this paper are the following:

(I) Computing the centroid of an $\mathcal{H}$-polytope is #P-hard, and it remains #P-hard even just to decide whether the centroid lies in a halfspace.

(II) Approximating the centroid of an $\mathcal{H}$-polytope is #P-easy.

(III) Any algorithm approximating the centroid of an arbitrary polytope within a distance $d^{\frac{1}{2}-\delta}$ for any fixed constant $\delta > 0$ can be used to obtain a fully polynomial-time approximation scheme for the centroid approximation problem and also an output sensitive polynomial algorithm for the Vertex Enumeration problem.

(IV) There is no polynomial algorithm that approximates the vertex centroid of an arbitrary $\mathcal{H}$-polyhedron within a distance $d^{\frac{1}{2}-\delta}$ for any fixed constant $\delta > 0$, unless $P = NP$.

The first two results in (I) follow easily from the hardness of counting the number of vertices of an $\mathcal{H}$-polytope. The next result is obtained by repeatedly slicing the given polytope, in a way somewhat similar to the one used to prove that computing the center of gravity is #P-hard [8]. The bootstrapping result in (III) is obtained by taking the product of the polytope with itself sufficiently many times. Using this result, and building on a construction in [6], we prove (IV). Namely, we use a modified version of the construction in [6] to show that it is NP-hard to approximate the centroid within a distance of $1/d$, then we use the result in (III) to bootstrap the hardness threshold to $d^{\frac{1}{2}-\delta}$ for any fixed constant $\delta > 0$.

We should remark that for the approximation of centroid, we only consider polytopes (and polyhedra) whose vertices lie inside a unit hypercube. To see how this assumption can easily be satisfied, notice that a halfspace h can be added to a polyhedron P such that $P \cap h$ is bounded and the vertices of P are preserved in $P \cap h$. Suppose for simplicity that the polyhedron is defined as the set of inequalities $Ax \leq \mathbf{1}$. Then it is easy to see that a halfspace whose normal is any vector lying in cone(A) when added to the set of inequalities makes the polyhedron bounded if the defining hyperplane has a large enough distance from the origin. The distance required for this only requires a number of bits polynomial in the size of the input A. For details see [9] where the same issue is discussed in a different context.

Once we have a polytope in $\mathbb{R}^d$, solving $2d$ linear programs gives us the width along each coordinate axis. The polytope can be scaled by a factor depending on the width along each axis to obtain a polytope all whose vertices lie inside a unit hypercube. In case we started with a polyhedron P, the scaled counterpart of the halfspace h that was added can be thrown to get back a polyhedron that is a scaled version of P and all whose vertices lie inside the unit hypercube. In Subsection 2.2 we provide justification for this assumption.

Since all the vertices of the polytope (or polyhedron) lie inside a unit hypercube, any arbitrary point from inside this hypercube is at a distance of at most $d^{\frac{1}{2}}$ from the vertex centroid. Thus, our last result above should be contrasted to the fact that approximating the vertex centroid within a distance of $d^{\frac{1}{2}}$ is trivial. Also, even though we discuss only polytopes *i.e.* bounded polyhedra in Subsections 2.1 and 2.2, the results and the proofs are valid for the unbounded case as well. We discuss the unbounded case explicitly only in Subsection 2.3.

2 Results

2.1 Exact Computation of the Centroid

The most natural computational question regarding the centroid of a polytope is whether we can compute the centroid efficiently. The problem is trivial if the input polytope is presented by its vertices. So we will assume that the polytope is presented by its facets. Perhaps not surprisingly, computing the centroid of an $\mathcal{H}$-polytope turns out be #P-hard. We prove this by showing that computing the centroid of an $\mathcal{H}$-polytope amounts to counting the vertices of the same polytope, a problem known to be #P-hard [3, 7].

Proposition 1 *Given an $\mathcal{H}$-polytope $\mathcal{P} \subset \mathbb{R}^d$, it is #P-hard to compute its centroid $c(\mathcal{P})$.*

Proof Embed $\mathcal{P}$ in $\mathbb{R}^{d+1}$ by putting a copy of $\mathcal{P}$ in the hyperplane $x_{d+1} = 1$ and making a pyramid with the base $\mathcal{P}$ and apex at the origin. Call this new polytope $\mathcal{Q}$. The facets of Q can be computed efficiently from the facets of P. Treating the direction of the positive x_{d+1}-axis as up, it is easy to see that the centroid of the new polytope lies at a height $1 - \frac{1}{n+1}$ if and only if the number of vertices of $\mathcal{P}$ is n. Thus any algorithm for computing the centroid can be run on $\mathcal{Q}$ and the number of vertices of $\mathcal{P}$ can be read off the $(d+1)$-st coordinate. □

Suppose, instead, that one does not want to compute the centroid exactly but is just interested in knowing whether the centroid lies to the left or to the right of a given arbitrary hyperplane. This problem turns out to be hard too, and it is not difficult to see why.

Proposition 2 *Given an $\mathcal{H}$-polytope $\mathcal{P} \subset \mathbb{R}^d$ and a hyperplane $h = \{a \cdot x = b\}$, it is #P-hard to decide whether $a \cdot c(\mathcal{P}) \leq b$.*

Proof Consider the embedding and the direction pointing upwards as used in the proof of Proposition 1. Given an oracle answering sidedness queries for the centroid and any arbitrary hyperplane, one can perform a binary search on the height of the centroid and locate the exact height. The number of queries needed is only logarithmic in the number of vertices of $\mathcal{P}$, which is at most $O(\lfloor \frac{d}{2} \rfloor \log m)$ if $\mathcal{P}$ has m facets. □

2.2 Approximation of the Centroid

As stated before, even though computing the gravitational centroid of a polytope exactly is #P-hard, it can be approximated to any precision by random sampling. Now we consider the problem of similarly approximating the vertex centroid of an $\mathcal{H}$-polytope. Let $dist(x, y)$ denote the Euclidean distance between two points $x, y \in \mathbb{R}^d$. We are interested in the following problem:

Input: $\mathcal{H}$-polytope $P \subset \mathbb{R}^d$ and a real number $\epsilon > 0$.

Output: $p \in \mathbb{R}^d$ such that $dist(c(P), p) \leq \epsilon$.

Recall that we only consider polytopes that are contained in a unit cube. This is a natural assumption since the problem of approximating the centroid is not so interesting if we allow polytopes that contain an arbitrarily large ball, since this would allow one to use an algorithm for approximating the centroid with any *fixed* guarantee to obtain another algorithm with an arbitrary guarantee by simply scaling the input polytope appropriately, running the given algorithm and scaling back.

We would like an algorithm for this problem that runs in time polynomial in the number of facets of P, the dimension d and $\frac{1}{\epsilon}$. Clearly, such an algorithm would be very useful because if such an algorithm is found then it can be used to test whether a polytope described by m facets has more than n vertices, in time polynomial in m, n and the dimension d of the polytope by setting $\epsilon < \frac{1}{2}\left(\frac{1}{n} - \frac{1}{n+1}\right)$ in the construction used in the proof of Proposition 1 This in turn would yield an algorithm that computes the number of vertices n of a d-dimensional polytope with m facets, in time polynomial in m, n and d. As stated before, a problem that is polynomially equivalent to the Vertex Enumeration problem is to decide if a given list of vertices of an $\mathcal{H}$-polytope is complete [1]. Clearly then, a polynomial-time approximation scheme for the centroid problem would yield an output-sensitive polynomial algorithm for the Vertex Enumeration problem.

Now we prove that the problem of approximating the centroid is #P-easy. We do this by showing that given an algorithm that computes the number of vertices of an arbitrary polytope (a #P-complete problem), one can compute the centroid to any desired precision by making a polynomial (in $\frac{1}{\epsilon}$, the number of facets and the dimension of the polytope) number of calls to this oracle. Notice that in the approximation problem at hand, we are required to find a point within a d-ball centered at the centroid of the polytope and of radius ϵ. We first modify the problem slightly by requiring to report a point that lies inside a hypercube, of side length 2ϵ, centered at the centroid of the polytope. (The hypercube has a clearly defined center of symmetry, namely its own vertex centroid.) To see why this does not essentially change the problem, note that the unit hypercube fits completely inside a d-ball with the same center and radius $\frac{\sqrt{d}}{2}$. Therefore the value of ϵ changes exactly by a factor of $\frac{\sqrt{d}}{2}$. We will call any point that is a valid output to this approximation problem, an ϵ-approximation of the centroid $c(P)$.

Given an $\mathcal{H}$-polytope P and a hyperplane $\{a \cdot x = b\}$ that intersects P in

the relative interior and does not contain any vertex of P, define P_1 and P_2 as follows:
$$P_1 = P \cap \{x | a \cdot x \leq b\}, \quad P_2 = P \cap \{x | a \cdot x \geq b\}.$$

Let V_1 be the common vertices of P_1 and P, and V_2 be common vertices of P_2 and P. Note that V_1, V_2 are just a partition of the vertex set of P and not the complete vertex sets of P_1, P_2. The hyperplane $\{a \cdot x = b\}$ creates some extra vertices in P_1 and P_2. The following lemma gives a way to obtain the ϵ-approximation of the centroid of P from the ϵ-approximations of the centroids of V_1 and V_2.

Lemma 1 *Given P, V_1, V_2 defined as above, let n_1 and n_2 be the number of vertices in V_1 and V_2 respectively. If c_1 and c_2 are ϵ-approximations of the centroids of V_1 and V_2 respectively, then $c = \frac{n_1 c_1 + n_2 c_2}{n_1 + n_2}$ is an ϵ-approximation of the centroid c^* of P.*

Proof Let c_{ij} be the j-th coordinate of c_i for $i \in \{1, 2\}$. Also, let c_i^* be the actual centroid of V_i with c_{ij}^* denoting the j-th coordinate of c_i^*. Since c_i approximates c_i^* within a hypercube of side-length 2ϵ, for each $j \in \{1, \cdots, d\}$ we have

$$c_{ij}^* - \epsilon \leq \quad c_{ij} \quad \leq c_{ij}^* + \epsilon.$$

Also, since c^* is the centroid of P,

$$c^* = \frac{n_1 c_1^* + n_2 c_2^*}{n_1 + n_2}.$$

Hence, for each coordinate c_j^* of c^* we have

$$\frac{n_1(c_{1j} - \epsilon) + n_2(c_{2j} - \epsilon)}{n_1 + n_2} \quad \leq c_j^* \leq \frac{n_1(c_{1j} + \epsilon) + n_2(c_{2j} + \epsilon)}{n_1 + n_2}$$
$$\Rightarrow \quad \frac{n_1 c_{1j} + n_2 c_{2j}}{n_1 + n_2} - \epsilon \quad \leq c_j^* \leq \frac{n_1 c_{1j} + n_2 c_{2j}}{n_1 + n_2} + \epsilon$$
$$\Rightarrow \quad c_j - \epsilon \quad \leq c_j^* \leq c_j + \epsilon$$
$$\Rightarrow \quad c_j^* - \epsilon \quad \leq c_j \leq c_j^* + \epsilon.$$

$\square$

Now to obtain an approximation of the centroid, we first slice the input polytope P from left to right (say, x_1 coordinate) into $\frac{1}{\epsilon}$ slices each of thickness at most ϵ. Using standard perturbation techniques we can ensure that no vertex of the input polytope lies on the left or right boundary of any slice. Any point in the interior of a slice gives us an ϵ-approximation of the x_1 coordinates of vertices of P that are contained in that slice. We can compute the number of vertices of P lying in this slice - which itself is a polytope - by subtracting the number of vertices on the boundary of the slice from the total number of

vertices of the slice. This can be done using the oracle for vertex counting and then using the previous Lemma along with a slicing along each of the coordinate axes, we can obtain an approximation to the centroid of P. Note that for slicing along one axis we only approximate that particular coordinate of the centroid and hence slicing along each of the axes is necessary and sufficient. Thus we have the following theorem:

Theorem 1 *Given an $\mathcal{H}$-polytope P contained in the unit hypercube, an ϵ-approximation of the centroid of P can be computed by making a polynomial number of calls to an oracle for computing the number of vertices of a polytope.*

Now we present a bootstrapping theorem indicating that any "sufficiently" non-trivial approximation of the centroid can be used to obtain arbitrary approximations. For the notion of approximation let us revert back to the Euclidean distance function. Thus, any point x approximating the centroid c within a parameter ϵ satisfies $dist(x, c) \leq \epsilon$. As before we assume that the polytope $\mathcal{P}$ is contained in the unit hypercube. Since the polytope is thus contained in a ball with the origin as its center and radius at most $\frac{\sqrt{d}}{2}$, any point inside $\mathcal{P}$ approximates the centroid within a distance $\sqrt{d}$. Before we make precise our notion of "sufficiently" non-trivial and present the bootstrapping theorem, some preliminaries are in order.

Lemma 2 *Suppose $(x, y), (u, u) \in \mathbb{R}^{2d}$, then*

$$||u - \frac{x + y}{2}|| \leq \frac{||(u, u) - (x, y)||}{\sqrt{2}},$$

where $|| \cdot ||$ is the Euclidean norm in $\mathbb{R}^d$ and $\mathbb{R}^{2d}$ respectively.

The proof of the above lemma is easy and elementary, and hence we omit it here. Next, consider the product of two polytopes. Given d-dimensional polytopes $\mathcal{P}, \mathcal{Q}$ the product $\mathcal{P} \times \mathcal{Q}$ is defined as the set $\{(x, y)|x \in \mathcal{P}, y \in \mathcal{Q}\}$. The facet defining inequalities of the product of P, Q can be computed easily from the inequalities defining P and Q.

$P = \{x|A_1 x \leq b_1\}$ and $Q = \{y|A_2 y \leq b_2\} \Rightarrow P \times Q = \{(x, y)|A_1 x \leq b_1, A_2 y \leq b_2\},$

where $A_1 \in \mathbb{R}^{m_1 \times d_1}, A_2 \in \mathbb{R}^{m_2 \times d_2}, x \in \mathbb{R}^{d_1}, y \in \mathbb{R}^{d_2}, b_1 \in \mathbb{R}^{m_1 \times 1}, b_2 \in \mathbb{R}^{m_2 \times 1}$.

It is easy to see that the number of vertices of $\mathcal{P} \times \mathcal{Q}$ is the product of the number of vertices of $\mathcal{P}$ and that of $\mathcal{Q}$, and the number of facets of $\mathcal{P} \times \mathcal{Q}$ is the sum of the number of facets of $\mathcal{P}$ and that of $\mathcal{Q}$. Moreover, the dimension of $\mathcal{P} \times \mathcal{Q}$ is the sum of the dimensions of $\mathcal{P}$ and $\mathcal{Q}$. Moreover, the vertices of $\mathcal{P} \times \mathcal{Q}$ are exactly all pairs of vertices with one vertex from $\mathcal{P}$ and the other from $\mathcal{Q}$ [10]

Observation 1 *If c is the centroid of a polytope P then (c, c) is the centroid of $P \times P$.*

Suppose we are given an algorithm for finding an ϵ-approximation of an arbitrary polytope contained in the unit hypercube. For example, for the simple algorithm that returns an arbitrary point inside the polytope, the approximation guarantee is $\sqrt{d}$. We consider similar algorithms whose approximation guarantee is a function of the ambient dimension of the polytope. Now suppose that for the given algorithm the approximation guarantee is $f(d)$. For some parameter k consider the k-fold product of P with itself $\overbrace{P \times \cdots \times P}^{k \; times}$, denoted by P^k. Using the given algorithm one can find the $f(kd)$ approximation of P^k and using Lemma 2 repeatedly one can then find the $\frac{f(kd)}{\sqrt{k}}$-approximation of P because if (x, y) is an ϵ-approximate centroid of $P \times P$ then $\frac{x+y}{2}$ is an $\frac{\epsilon}{\sqrt{2}}$-approximate centroid of P. This gives us the following bootstrapping theorem:

Theorem 2 *Suppose we are given an algorithm that computes a $\frac{\sqrt{d}}{g(d)}$-approximation for any polytope contained in the unit hypercube in polynomial time, where $g(.)$ is an unbounded monotonically increasing function. Then, one can compute an ϵ-approximation in time polynomial in the size of the polytope and $g^{-1}(\frac{\sqrt{d}}{\epsilon})$.*

In particular, if we have an algorithm with $d^{\frac{1}{2}-\delta}$ approximation guarantee for finding the centroid of any polytope for some fixed constant $\delta > 0$, then this algorithm can be used to construct a fully polynomial-time approximation scheme for the general problem.

2.3   Approximating the centroid of a polyhedron is hard

The reader should note that the analysis of Subsections 2.1 and 2.2 remains valid even for the unbounded case (polyhedra). Also recall that in our setup vertices of the given polyhedron lie inside a unit cube. See introduction for a discussion on how to satisfy this assumption.

Even though we do not have any idea about the complexity of approximating the centroid of a polytope, now we show that for an arbitrary unbounded polyhedron the vertex centroid cannot be $d^{\frac{1}{2}-\delta}$-approximated for any fixed constant $\delta > 0$ unless $P = NP$. To show this we first prove that for an $\mathcal{H}$-polyhedron $P \subset \mathbb{R}^d$ the vertex centroid of P cannot be $\frac{1}{d}$-approximated in polynomial time unless $P = NP$. This together with Theorem 2 implies that if the centroid of a polyhedron could be $d^{\frac{1}{2}-\delta}$-approximated then one could compute a $\frac{1}{d}$-approximation of the centroid in time polynomial in $d^{\frac{3}{2\delta}}$ (set $g(d) = d^\delta$), which is bounded from above by a polynomial for any fixed δ. This would complete the proof for hardness of $d^{\frac{1}{2}-\delta}$-approximation of the centroid of an $\mathcal{H}$-polyhedron.

Our proof uses the construction from [6] and its slight modification in [2]. We give a sketch below. For completeness, we also give the complete construction in the appendix.

The proof goes as follows: Given a Boolean CNF formula ϕ, we construct a graph $G(\phi)$ such that $G(\phi)$ has a "long" negative cycle if and only if ϕ is satisfiable. For a given graph G we define a polyhedron $P(G)$ such that every

negative cycle in G is a vertex of $P(G)$ and vice-versa. From the properties of the vertex centroid of this class of polyhedra, we then prove that for any formula ϕ, $\frac{1}{d}$-approximating the vertex centroid of $P(G(\phi))$ would reveal whether ϕ is satisfiable or not.

2.3.1   Graph of a CNF formula.

Recall that the 3SAT problem is the following decision problem: Given a CNF Boolean formula $\phi = C_1 \wedge \cdots \wedge C_M$ on N literals $x_1, \cdots, x_N$ such that every clause C_i is a disjunction of exactly 3 literals, is ϕ satisfiable?

Given a directed graph $G = (V, E)$ and a weight function $w : E \to \mathbb{R}$ on its arcs, a directed cycle will be called *short* if it has only two nodes and *long* otherwise. A cycle is *negative* if the total weight on its arcs is negative. The following result was established in [6] (see also [2]).

Lemma 3 *For any 3-CNF ϕ with m clauses we can obtain an arc weighted directed graph $G(\phi)$ with the following properties:*

(P1) $G(\phi)$ has $18m + 1$ edges;

(P2) $G(\phi)$ has $3m$ short negative cycles;

(P3) every negative cycle in $G(\phi)$ has total weight -1;

(P4) there is a distinguished arc e, such that every long negative cycle but no short cycle in $G(\phi)$ contains e; and

(P5) $G(\phi)$ has a long negative cycle if and only if ϕ is satisfiable.

2.3.2   The polyhedron of negative-weight flows of a graph.

Given a directed graph $G = (V, E)$ and a weight function $w : E \to \mathbb{R}$ on its arcs, consider the following polyhedron:

$$
P(G, w) = \left\{ y \in \mathbb{R}^E \; \middle| \; \begin{array}{ll} (F) & \displaystyle\sum_{v:(u,v)\in E} y_{uv} - \sum_{v:(v,u)\in E} y_{vu} = 0 \quad \forall \; u \in V \\[2ex] (N) & \displaystyle\sum_{(u,v)\in E} w_{uv} y_{uv} = -1 \\[1ex] & y_{uv} \geq 0 \hspace{2.5cm} \forall \; (u,v) \in E \end{array} \right\}.
$$

If we think of $w_{u,v}$ as the cost/profit paid for edge (u, v) per unit of flow, then each point of $P(G, w)$ represents a *negative-weight circulation* in G, i.e., assigns a non-negative flow on the arcs, obeying the *conservation of flow* at each node of G, and such that total weight of the flow is strictly negative.

For a subset $X \subseteq E$, and a weight function $w : E \mapsto \mathbb{R}$, we denote by $w(X) = \sum_{e \in X} w_e$, the total weight of X. For $X \subseteq E$, we denote by $\chi(X) \in \{0, 1\}^E$ the characteristic vector of X: $\chi_e(X) = 1$ if and only if $e \in X$, for $e \in E$. The following theorem states that the vertex set $\mathcal{V}(P(G, w))$ of $P(G, w)$ is in one-to-one correspondence with the negative cycles of the graph G.

Theorem 3 ([2]) *Let $G = (V, E)$ be a directed graph and $w : E \to \mathbb{R}$ be a real weight on the arcs. Also, let $\mathcal{C}^-(G, w)$ be the set of negative cycles in G. Then*

$$\mathcal{V}(P(G, w)) \;=\; \left\{ \frac{-1}{w(C)} \chi(C) : \; C \in \mathcal{C}^-(G, w) \right\}. \tag{1}$$

Since by $(P3)$, in the graph G arising from a 3-CNF formula, every negative cycle has weight exactly -1, Theorem 3 implies that the vertices of $P(G, w)$ are exactly the characteristic vectors of the negative cycles of G. By $(P5)$, finding whether G has any long negative cycle, *i.e.*, a negative cycle containing the distinguished arc e (cf. $(P4)$) is NP-complete. By $(P1)$ and $(P2)$, for a 3-CNF formula with m clauses the constructed graph G has $18m + 1$ arcs and $3m$ trivial short negative cycles. Consequently, the polyhedron $P(G, w)$ that is finally obtained has dimension $18m + 1$ and $3m$ trivial vertices corresponding to the short negative cycles of G.

Now, if there are no long negative cycles then the vertex centroid of $P(G, w)$ has value 0 in the coordinate corresponding to the edge e. For simplicity, we will refer to this coordinate axis as x_e. On the other hand, if there are $K \geq 1$ long negative cycles in G then in the centroid $x_e = \frac{K}{K+3m} \geq \frac{1}{3m+1}$. This implies that having an ϵ-approximation for the centroid of $P(G, w)$ for $\epsilon < \frac{1}{2(3m+1)}$ would reveal whether or not $P(G, w)$ has a non-trivial vertex and hence whether or not G has a long negative cycle. Thus we have the following theorem:

Theorem 4 *There is no polynomial algorithm that computes a $\frac{1}{d}$-approximation of the vertex centroid of an arbitrary $\mathcal{H}$-polyhedron $P \subset \mathbb{R}^d$, unless $P = NP$.*

An immediate consequence of Theorem 2 and Theorem 4 is that there is no polynomial algorithm that computes any "sufficiently non-trivial" approximation of the vertex centroid of an arbitrary $\mathcal{H}$-polyhedron unless $P = NP$. More formally,

Corollary 1 *There is no polynomial algorithm that $d^{\frac{1}{2}-\delta}$-approximates the centroid of an arbitrary d-dimensional $\mathcal{H}$-polyhedron for any fixed constant $\delta > 0$ unless $P = NP$.*

3   Open Problems

Although we can show that for unbounded polyhedra almost any non-trivial approximation of the vertex centroid is hard, we can not make a similar statement for the bounded case (*i.e. polytopes*). One interesting variant of Theorem 2 would be to consider a ball of radius r instead of a halfspace. If containment of vertex centroid in a ball of radius r can be decided in time polynomial in the number of inequalities defining the polytope, the dimension and r then one can perform a sort of random walk inside the polytope and approximate the centroid in polynomial time. We leave out the details of this random walk since we do not have a method to check containment inside a ball.

References

[1] D. Avis, D. Bremner, and R. Seidel. How good are convex hull algorithms? *Comput. Geom.*, 7:265–301, 1997.

[2] E. Boros, K. Elbassioni, V. Gurvich, and H. R. Tiwary. The negative cycles polyhedron and hardness of checking some polyhedral properties. *Annals of Operations Research*, Feb. 2010.

[3] M. E. Dyer. The complexity of vertex enumeration methods. *Mathematics of Operations Research*, 8(3):381–402, 1983.

[4] M. E. Dyer and A. M. Frieze. On the complexity of computing the volume of a polyhedron. *SIAM J. Comput.*, 17(5):967–974, 1988.

[5] R. Kannan, L. Lovász, and M. Simonovits. Random walks and an $o^*(n^5)$ volume algorithm for convex bodies. *Random Structures and Algorithms*, 11(1):1–50, December 1998.

[6] L. Khachiyan, E. Boros, K. Borys, K. M. Elbassioni, and V. Gurvich. Generating all vertices of a polyhedron is hard. *Discrete & Computational Geometry*, 39(1-3):174–190, 2008.

[7] N. Linial. Hard enumeration problems in geometry and combinatorics. *SIAM J. Algebraic Discrete Methods*, 7(2):331–335, 1986.

[8] L. Rademacher. Approximating the centroid is hard. In *Symposium on Computational Geometry*, pages 302–305, 2007.

[9] H. R. Tiwary. On the hardness of computing intersection, union and minkowski sum of polytopes. *Discrete Comput. Geom.*, 40(3):469–479, 2008.

[10] G. M. Ziegler. *Lectures on Polytopes*, volume 152 of *Graduate Texts in Mathematics*. Springer-Verlag, Berlin, 1995.