Approximate Center Points with Proofs*

Gary L. Miller
glmiller@cs.cmu.edu

Donald R. Sheehy
dsheehy@cs.cmu.edu

Computer Science Department
Carnegie Mellon University
Pittsburgh, Pennsylvania

ABSTRACT

We present the Iterated-Tverberg algorithm, the first deterministic algorithm for computing an approximate centerpoint of a set $S \in \mathbb{R}^d$ with running time sub-exponential in d. The algorithm is a derandomization of the Iterated-Radon algorithm of Clarkson et al and is guaranteed to terminate with an $O(1/d^2)$-center. Moreover, it returns a polynomial-time checkable proof of the approximation guarantee, despite the coNP-Completenes of testing centerpoints in general. We also explore the use of higher order Tverberg partitions to improve the runtime of the deterministic algorithm and improve the approximation guarantee for the randomized algorithm. In particular, we show how to improve the $O(1/d^2)$-center of the Iterated-Radon algorithm to $O(1/d^{\frac{r}{r-1}})$ for a cost of $O((rd)^d)$ in time for any integer r.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*geometrical problems and computations*

General Terms

Algorithms, Theory

Keywords

centerpoints, derandomization, approximation algorithms, Tverberg's Theorem

1. INTRODUCTION

A centerpoint of a set $S \subset \mathbb{R}^d$ is a point c such that every closed half-space containing c also contains at least $\frac{n}{d+1}$ points of S. Intuitively, every hyperplane through a c divides S into roughly equal parts. The existence of centerpoints was established by a theorem of Rado [12], which deals with general measures of which point sets are a special case. The simplified proof for the case of centerpoints of point sets is due to Danzer et al[4].

A centerpoint is a natural generalization of the median to higher dimensions. They are used as robust estimators in statistics, because they are invariant under affine transformation and robust to outliers [5]. They are also used in mesh partitioning[7].

The existence of centerpoints can be proven directly from either of two classic theorems of convexity theory, Helly's Theorem and Tverberg's Theorem. In Section 3, we discuss how these two proofs of the centerpoint theorem lead to different options for designing algorithms for computing centerpoints.

The exact complexity of computing centerpoints in higher dimensions is not known. The dual problem of testing if a given point is a centerpoint is coNP-Complete [14]. However, a simple corollary of Tverberg's Theorem guarantees the existence of a subset of centerpoints, call them Tverberg points, that admit polynomial-time checkable proofs. Moreover, testing if a point is a Tverberg point is NP-Complete[14]. In this case, the decision problem is well understood but sheds little light on the hardness of the search problem of actually finding a centerpoint.

We consider the problem of finding an approximate centerpoint. Call c a β-center if every closed half-space containing c also contains a β fraction of the points of S. So, a classical centerpoint is a $\frac{1}{d+1}$-center. The fastest known algorithm for computing a centerpoint of $S \subset \mathbb{R}^d$ is due to Chan [1] and computes a β-center in time $O(n^{d-1})$ where β is the maximum achievable for the set S. In the literature, such a β-center is also known as a Tukey median.

The Iterated-Radon algorithm of Clarkson et al was the first algorithm that computes an approximate centerpoint in time sub-exponential in d [2]. The algorithm computes a $O(1/d^2)$-center with high probability. Section 3 describes how this algorithm resembles the proof of the Centerpoint Theorem via Helly's Theorem.

The main operation in the Iterated-Radon algorithm is to replace sets of points by their Radon point, a point in the common intersection of the convex hull of two disjoint subsets. Radon's Theorem guarantees the existence of such a point. Tverberg's Theorem is a generalization of Radon's Theorem that guarantees a common intersection for a larger collection of subsets.

In this paper, we use the intuition from Tverberg's The-

*This work was supported in part by the National Science Foundation under grants CCF-0635257

SCG'09, June 8–10, 2009, Aarhus, Denmark.

orem to construct a proof for an approximate centerpoint. The result is a new approximation algorithm, Iterated-Tverberg, that derandomizes the Iterated-Radon algorithm of Clarkson et al. In Section 4, we prove that the Iterated-Tverberg algorithm produces a a $O(1/d^2)$-center in time sub-exponential in d with a polynomial-time checkable proof.

We elaborate on this intuition in Section 5, showing how solving larger sub-problems can be used to speed up the run time of the deterministic algorithm and to improve the approximation ratio of the randomized version.

2. RELATED WORK

Centerpoints are the most well known definition of a geometric median [6]. Like many such medians, it can be computed via linear programming and the problem of finding a "best" centerpoint can be written as a maximum feasible subsystem problem (see [5] for a survey of computational aspects of data depth). As might be expected, any linear programming method will require time $n^{O(d)}$, limiting their usefulness to low-dimensional instances.

In the plane, centerpoints can be computed in linear time [9]. Several algorithms are known to compute centerpoints in $\mathbb{R}^3$ in $O(n^2 \operatorname{polylog} n)$ time [3, 11]. The best known algorithm for general dimensions is due to Chan and runs in $O(n^{d-1})$ randomized time[1]. Chan's algorithm computes the deepest possible centerpoint, also known as a Tukey median. He conjectures that the $O(n^{d-1})$ runtime is optimal for this problem in the algebraic decision tree model. However, the exact complexity of computing centerpoints is not known. In particular, it is not known if it is possible to compute a centerpoint in time polynomial in n and d.

Several approximation algorithms for centerpoints exist in the literature. Several approaches using random sampling are known [10, 14, 2]. Verbarg showed that for dense points, the mean is a good approximate centerpoint[16]; it is a β-center where β depends on the density.

The only previously known algorithm to compute a centerpoint in time sub-exponential in d is the Iterated-Radon algorithm of Clarkson et al [2]. The Iterated-Radon algorithm returns a $O(1/d^2)$-center with high probability in time polynomial in n and d. Unfortunately, Iterated-Radon is a Monte Carlo algorithm and there is no way known to verify that the point returned by the algorithm is indeed a centerpoint. The inner loop of Iterated-Radon depends on the following classic theorem [13].

Theorem 2.1 (Radon's Theorem, 1921). *Given $n > d+1$ points $S \subset \mathbb{R}^d$, there exists a partition $(U, \overline{U})$ of S such that* $\operatorname{conv}(U) \cap \operatorname{conv}(\overline{U}) \neq \emptyset$.

We call a partition of $d+2$ points as described in the Theorem, a *Radon partition*, and we call a point in the intersection, a *Radon point*. The simplest version of the Iterated-Radon algorithm works as follows. Build a balanced $(d+2)$-ary tree of height h. Fill in the leaves with points from the input set S by sampling them uniformly at random. Each interior node of the tree is filled in with the Radon point of its children. A height of $h = \lg n$ is needed to compute a $O(1/d^2)$-center with high probability, resulting in a runtime that is $O(poly(d) n^{\lg d+2})$. Thus it is sub-exponential in d but not polynomial. A second version of the algorithm is also presented in [2] that pushes the running time down to $O(poly(n, d))$. The Iterated-Tverberg we present in this paper is reminiscent of the former and has similar time complexity. We also describe a modification analogous to the latter, but at this time, we are unable to analyze the running time.

The Iterated-Radon algorithm has also been modified to use sampling to achieve sub-linear running times. Because a centerpoint of a sufficiently large random sample is an approximate center of the original set, sampling may be used as a preprocess on any approximate centerpoint algorithm. Iterated-Radon has also been augmented to solve the centerpoint problem exactly (using the linear programming methods described above) on subsets of points to achieve a higher quality $O(1/d + \varepsilon)$-center. We present a new way to leverage these larger subproblems to improve centerpoint quality and analyze its impact in both the randomized and the deterministic algorithms (see Section 5).

3. TWO PROOFS OF THE CENTERPOINT THEOREM

Theorem 3.1 (The Centerpoint Theorem). *Given a set of n points $S \subset \mathbb{R}^d$, there exists a centerpoint $c \in \mathbb{R}^d$ such that every closed half-space containing c also contains at least $\left\lceil \frac{n}{d+1} \right\rceil$ points of S.*(Rado [12], 1947, Danzer et al [4], 1963)

The Centerpoint Theorem is generally presented as an easy consequence of Helly's Theorem. It is also possible to prove the existence of centerpoints via Tverberg's Theorem. The relationship between these two proofs gives insight into the relationship between the Iterated-Radon algorithm and its derandomization presented in this paper.

Theorem 3.2 (Helly's Theorem [8], 1913). *Given a collection of compact, convex sets $X_1, \ldots, X_n \subset \mathbb{R}^d$. If every $d+1$ of these sets have a common intersection, then the whole collection has a common intersection.*

The Centerpoint Theorem follows from Helly's Theorem as follows. Consider the set H of all open half-spaces that contain more than $\frac{dn}{d+1}$ points of S. For each such half-space $h \in H$ let P_h denote $conv(S \cap h)$. Clearly, any $d+1$ of the half-spaces have a common intersection at one of the points of S, and thus every $d+1$ of the P_h's also have a common intersection. We apply Helly's Theorem to the sets P_h. The common intersection guaranteed by Helly's Theorem is exactly the set of all centerpoints.

The most common elementary proof of Helly's Theorem makes extensive use of Radon's Theorem, despite that Helly's Theorem technically came first (though published second). The proof first considers the case where there are only d+2 sets. The hypothesis of the Theorem implies the existence of $d+2$ points, each taken from the common intersection of $d+1$ of the sets. The Radon point of these $d+2$ points satisfies the conclusion of the Theorem. The proof for $n > d+2$ sets uses induction. The inductive step again considers a set of points taken from each of the common intersections of $n-1$ sets, and shows the Radon point of this set satisfies the Theorem. Unraveling this induction into an algorithm, we arrive at something very much like the Iterated-Radon algorithm of Clarkson et al. The difference is that the run time is prohibitive because we would have to consider far too many sets.

The Centerpoint Theorem can also be proven via Tverberg's generalization of Radon's Theorem.

THEOREM 3.3 (TVERBERG'S THEOREM [15], 1966). *Given $(d+1)(r-1)+1$ points $S \subset \mathbb{R}^d$, there exists a partition of S into $S_1, \ldots, S_r$, such that $\bigcap_{i=1}^{r} \operatorname{conv}(S_i) \neq \emptyset$.*

Observe that Radon's Theorem is a special case of Tverberg's Theorem when $r = 2$.

Say that a point c is a Tverberg point if it is in the common intersection of the convex hulls in the Tverberg partition. Then, setting $r = \lceil n/d+1 \rceil$ yields a Tverberg point contained in the convex hull of $\lceil n/d+1 \rceil$ pairwise disjoint subsets of S. Any half-space containing c must also contain at least one point from each of the subsets and therefore, c is a centerpoint.

Observe that the centerpoints guaranteed by Tverberg's Theorem come equipped with a polynomial-time checkable proof. Given the partition, we need only verify that the point is in the convex hull of each part. If any part in the partition has more than $d+1$ points then by Carathéodory's Theorem, there is a subset of size $d+1$ that contains the Tverberg point is its convex hull. We may therefore assume the convex hulls are simplices of dimension at most d, so checking can be done quickly. The key insight in derandomizing the Iterated-Radon algorithm is to actively construct these Tverberg partitions for the intermediate points used in the algorithm.

4. DERANDOMIZING THE ITERATED-RADON ALGORITHM

The Iterated-Tverberg algorithm looks very similar to the Iterated-Radon algorithm. The key difference is that each successive approximation computed along the way carries with it a proof of its quality as a centerpoint. The proof is in the form of a Tverberg partition of a subset of the inputs. Define the *depth* of a Tverberg point to be the number of parts in the corresponding Tverberg partition.

When we combine $d+2$ points of depth r into a Radon point c, we can rearrange the proofs to get a new proof that c has depth $2r$ as shown in the following Lemma.

LEMMA 4.1. *Given a set P of $d+2$ Tverberg points of depth r with disjoint partitions, the Radon point of P has depth $2r$.*

PROOF. Let (P_1, P_2) be the Radon partition for P, and let c be the Radon point. For each $p_i \in P$, order the partitions in the proof of p_i and call the jth partition $U_{i,j}$. We build a proof that c has depth at least $2r$. The partitions in the new proof are of the form $\bigcup_{p_i \in P_k} U_{i,j}$ for some choice of $k \in \{1, 2\}$ and $j \in \{1, \ldots, r\}$.

To show that the new proof is correct, it suffices to show that for any choice of j and k, the new approximation c is contained in $conv(\bigcup_{p_i \in P_k} U_{i,j})$. What follows is the long proof of the intuitive statement that a convex combination of convex combinations is itself a convex combination of the base set.

Because c is a Radon point, we know that $c \in conv(P_k)$. Also, the Tverberg points $p_i \in P_k$ are each contained in $conv(U_{i,j})$. So, we can write $c = \sum_{p_i \in P_k} \lambda_i p_i$ and $p_i = \sum_{u_m \in U_{i,j}} \alpha_m u_m$, where $\sum \lambda_i = \sum \alpha_m = 1$ and $\lambda_i, \alpha_m \geq 0$.

```
ITERATED-TVERBERG(S ∈ ℝ^d : |S| = n)
  IF |S| ≤ 2(d+1)^2
     RETURN any point of S (it is a proof of depth 1)
  ENDIF
  FOR i = 1 to d+2
     Select ⌈n/2⌉ points S' ⊂ S.
     P_i ← Iterated − Tvererg(S').
     S ← S \ proof(P_i)
  ENDFOR
  center, U, Ū ← Radon(center(P_1), ..., center(P_{d+2})).
  Combine proofs from P_1, ..., P_{d+2}.
  Prune the proof until it is minimal for depth ⌈n/(2(d+1)^2)⌉.
  RETURN the center and the proof.
```

Combining these two convex combinations, we see that

$$c = \sum_{p_i \in P_k} \lambda_i \sum_{u_m \in U_{i,j}} \alpha_m u_m \tag{1}$$

$$= \sum_{u_m \in \bigcup_{p_i \in P_k} U_{i,j}} \lambda_i \alpha_m u_m. \tag{2}$$

To show that this is indeed a convex combination, we note that $\sum_{i,m} \lambda_i \alpha_m = \sum_i \lambda_i (\sum_m \alpha_m) = \sum_i \lambda_i(1) = 1$.

□

The preceding Lemma implies a simple linear time deterministic algorithm for computing an approximate centerpoint. Construct a $(d+2)$-ary tree with n leaves. Fill the leaves with the points of S. Fill in each interior node of the tree by the Radon point of its children. The height of the tree is $\log_{d+2} n$, so Lemma 4.1 implies that the depth of the root is $2^{\log_{d+2} n} = O(n^{1/\lg(d+2)})$. Not too shabby for such a simple algorithm, but the depth of the output is only sublinear in n. To get a constant-factor approximate center, we need to find a way to build this tree higher, and in order to do that, we need more leaves. The following Lemma gives a hint as to where we can look to find some more points to stick in the leaves.

LEMMA 4.2. *If there is a proof that a point p has depth r, there exists such a proof that contains at most $r(d+1)$ points of S.*

PROOF. Let $P_1, \ldots, P_r$ be the sets in the proof for p. This means that $p \in conv(P_i)$ for each $i = 1 \ldots r$. By Carathéodory's Theorem, there exists a subset $P'_i \subset P_i$ of at most $d+1$ points such that $p \in conv(P'_i)$. So, the sets $P'_1, \ldots, P'_r$ is the desired proof of the correct size. □

We refer to this economizing of proofs as *pruning*. In the algorithm, pruning is applied to the proofs generated by combining smaller proofs as in Lemma 4.1. In such instances, the convex combination is known. Moreover, if the combined proofs were each pruned, then the total number of points in the combined sets is at most $2(d+1)$, and the pruned set can be found by computing $O(d)$ projections.

It can easily be done in poly(d) time.

4.1 Analysis of the deterministic algorithm

THEOREM 4.3. *The Iterated-Tverberg algorithm (see Figure 4) always returns a β-center of depth at least $\left\lceil \frac{n}{2(d+1)^2} \right\rceil$.*

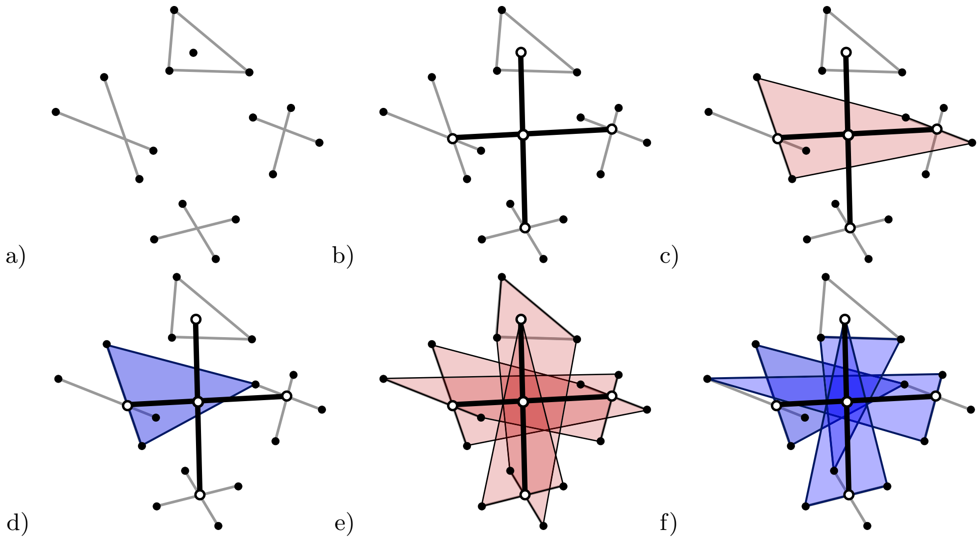

Figure 1: The Iterated-Tverberg algorithm: (a) Sets of $d+2$ points are divided into Radon partitions. (b) $d+2$ Radon points are combined into a second-order Radon partition. (c) A proof polygon is formed by taking the convex hull of two subpartitions, one from each of the Radon points in the second order partition. (d) The proof polygon is reduced to a simplex. (e) All of the proof polygons before the reduction phase. (f) The proof simplices after the reduction.

Proof. Observe first that proof returned by the algorithm always has at most $\left\lceil \frac{n}{2(d+1)^2} \right\rceil$ sets. It has at least this value in the base case, where $n \le 2(d+1)^2$ because the point returned is itself a proof of depth 1. We only need $d+2$ points by Radon's Theorem to get depth 2 so the algorithm also succeeds when $n \le 4(d+1)^2$, and we may assume that $n > 4(d+1)^2$.

Suppose for contradiction that for some minimal set S of size n', the proof output by the algorithm has fewer than $\lceil \frac{n'}{2(d+1)^2} \rceil$ sets. Let g_k denote the size of the proof (in sets) returned by the algorithm when the input is an arbitrary k element subset of S'. Because S' is a minimal contradiction, we have $g_k = \lceil \frac{k}{2(d+1)^2} \rceil$ for all $k < n'$. In particular,

$$g_{\lceil \frac{n'}{2} \rceil} = \left\lceil \frac{\lceil \frac{n'}{2} \rceil}{2(d+1)^2} \right\rceil \tag{3}$$

$$\le \frac{n'+1}{4(d+1)^2} + 1. \tag{4}$$

The number of points of S contained in $d+1$ proofs of size $g_{\lceil \frac{n'}{2} \rceil}$ is

$$(d+1)^2 g_{\lceil \frac{n'}{2} \rceil} \le (d+1)^2 \left(\frac{n'+1}{4(d+1)^2} \right) + (d+1)^2 \tag{5}$$

$$= \frac{n'+1}{4} + (d+1)^2 \tag{6}$$

$$< \frac{n'+1}{4} + \frac{n}{4} \tag{7}$$

$$\le \left\lfloor \frac{n'}{2} \right\rfloor. \tag{8}$$

This means that after $d+1$ recursive calls, we still have $\left\lceil \frac{n'}{2} \right\rceil$ points left in S, enough to make one more. So, we are able to run the combining operation on the $d+2$ (point,proof) pairs to get a new point with proof of depth $2g_{\lceil \frac{n'}{2} \rceil}$ by Lemma 4.1. We can now derive a contradiction by showing that $g_{n'}$ is larger than we supposed.

$$g_{n'} = 2g_{\lceil \frac{n'}{2} \rceil} \tag{9}$$

$$= 2 \left\lceil \frac{\left\lceil \frac{n'}{2} \right\rceil}{2(d+1)^2} \right\rceil \tag{10}$$

$$\ge \left\lceil \frac{n'}{2(d+1)^2} \right\rceil. \tag{11}$$

□

4.2 Running Time

The algorithm of the previous section can be analyzed as follows. Let t_n represent the running time for n nodes. We see that t_n is as follows.

$$t_n = (d+2)t_{\lceil \frac{n}{2} \rceil} + O(n \operatorname{poly}(d)) \tag{12}$$

$$\le (d+2)^{\lg n} + O(\operatorname{poly}(d) n \lg n) \tag{13}$$

$$= O(n^{\lg(d+2)} + \operatorname{poly}(d) n \lg n). \tag{14}$$

4.3 Reusing Work

The Iterated-Tverberg Algorithm as presented could benefit from a very simple optimization. At each phase, when the combined proofs are pruned and some are thrown back, we can attempt to reuse the computation of the now extraneous Tverberg points. The hope is that by reusing work, the algorithm will run faster. Unfortunately, this dynamic programming variant does not achieve an asymptotic speedup of the algorithm over the straightforward version presented.

5. LEVERAGING LARGER SUBPROBLEMS

Both the Iterated-Radon algorithm and the Iterated-Tverberg algorithm combine sets of $d+2$ points by partitioning them into two sets by Radon's Theorem. In this section we address the result on these algorithms if we instead solve larger subproblems. That is, rather than combining points in sets of $d+2$, we look at sets of size $(d+1)(r-1)+1$ for some fixed r. It is not known how to solve these larger problems in time sub-exponential in d. However, if n is large and d is not too large, it may be feasible to solve subproblems in $O(d^d)$ time even though $O(n^d)$ is prohibitive.

5.1 Improving the approximation for the Iterated-Radon Algorithm

The Radon point of $d+2$ points is a centerpoint of the subset. Consider the following modified version of the Iterated-Radon Algorithm.

We can run the same iterative algorithm as before except using r-partitions instead of 2-partitions. In fact, it is not necessary to keep around the partition, it actually suffices just to find any centerpoint of $(d+1)(r-1)+1$ points at each round. We can go through the analysis from the Clarkson et al and see the impact of r in the quality of the centerpoint achieved.

The analysis works by looking at any projection of the point set to the line. We compute the probability $f_h(x)$ that the tree of iterations with height h returns a center of depth at most x. Without loss of generality, the projections of the points of S land on $\frac{1}{n}, \frac{2}{n}, \ldots, 1$. It follows that $f_0(x) \leq x$. The quality of the center will be nx where x is such that $f_h(x)$ is very small.

At each iteration, the centerpoint is at least r deep in the projection. There are $\binom{(d+1)(r-1)+1}{r}$ choices for the r points less than the centerpoint in the projection. By the union bound,

$$f_h(x) \geq \binom{(d+1)(r-1)+1}{r} f_{h-1}(x)^r. \tag{15}$$

Say, $\beta = \binom{(d+1)(r-1)+1}{r}^{-1}$.

$$f_h(x) \geq \beta^{-1} f_{h-1}(x)^r \tag{16}$$

$$\geq \beta^{-1}(\beta^{-r} f_{h-2}(x)^{r^2}) \tag{17}$$

$$\geq \beta^{-1} \cdot \beta^{-r} \cdots \beta^{-r^h} f_0(x)^{r^h} \tag{18}$$

$$\geq \beta^{\frac{1-r^h}{r-1}} x^{r^h} \tag{19}$$

$$\geq \beta^{\frac{1}{r-1}} \left(\frac{x}{\beta^{\frac{1}{r-1}}} \right)^{r^h} \tag{20}$$

Now, since $\beta = O((rd)^{-r})$, we can choose x smaller than $O((rd)^{\frac{-r}{r-1}})$ and the probability $f_h(x)$ vanishes as desired. So, even for a choice of $r = 3$, we can improve the quality of the resulting centerpoint by $O\left(\frac{n}{d^{\frac{1}{2}}}\right)$.

5.2 Speeding up the Iterated-Tverberg Algorithm

In this section, we show how the same trick of solving larger subproblems can speed up the run time of the deterministic algorithm. Tverberg's Theorem guarantees the existence of a partition of S into r sets whose convex hulls have a common intersection as long as $|S| > (d+1)(r-1)+1$. Say $T(r)$ is the time required to compute a Tverberg partition into r parts. To the best of our knowledge, nothing better than brute force is known for computing Tverberg partitions for $r > 2$.

We will show that a slight modification to the Iterated-Tverberg algorithm to use Tverberg r-partitions instead of Radon partitions results in a $n^{\lg r}/T(r)$ speedup. Thus, for n large enough, we get a substantial speedup.

The modified algorithm simply makes recursive calls on sets of $\lceil n/r \rceil$ points and combines them in sets of $(r-1)(d+1)+1$. The analysis is virtually identical to the original version except we give up a factor of $r/2$ in the depth of the output. As for the running time, the new algorithm now has a recursion tree with higher fan out and the resulting run time is $O((d+2)^{\log_r n} T(r)) = O(n^{\lg(d+2)/\lg r} T(r))$.

6. CONCLUSION

We have presented the Iterated-Tverberg algorithm, the first algorithm that deterministically computes an approximate centerpoint in time sub-exponential in d. By combining intuition from both Helly's Theorem and Tverberg's Theorem, our method sheds an interesting new light on the problem of computing centerpoints. It still remains open whether it is possible to compute approximate centerpoints deterministically in time polynomial in n and d. We conjecture that it is.

We also extended both our algorithm and the Iterated-Radon algorithm by looking at the impact of solving larger subproblems. One consequence of this work is that any new results on quickly computing centerpoints for small point sets can be used to improve these algorithms. Currently, it is not known how to compute centerpoints of more than $2d+2$ points in time polynomial in d. However, we conjecture that computing the centerpoint of $2d+3$ points in $\mathbb{R}^d$ is NP-hard.

The computation of centerpoints draws a compelling correspondence between fundamental theorems in convexity theory, Helly's Theorem and Tverberg's Theorem, and fundamental complexity classes of NP and coNP. It is our hope that future work will further elucidate this correspondence.

7. REFERENCES

[1] T. Chan. An optimal randomized algorithm for maximum Tukey depth. In *SODA: ACM-SIAM Symposium on Discrete Algorithms*, 2004.

[2] K. Clarkson, D. Eppstein, G. Miller, C. Sturtivant, and S.-H. Teng. Approximating center points with iterated Radon points. *International Journal of Computational Geometry and Applications*, 6(3):357–377, Sep 1996. invited submission.

[3] R. Cole, M. Sharir, and C. K. Yap. On k-hulls and related problems. *SIAM J. Comput.*, 16(1):61–77, 1987.

[4] L. Danzer, B. Grünbaum, and V. Klee. Helly's theorem and its relatives. In *Proc. Symp. Pure Math*, volume VII, 1963.

[5] K. Fukuda and V. Rosta. Data depth and maximum feasible subsystems. In Avis, Hertz, and Marcotte, editors, *Graph Theory and Combinatorial Optimization, Springer, 2005.* 2005.

[6] J. Gil, W. L. Steiger, and A. Wigderson. Geometric medians. *Discrete Mathematics*, 108(1-3):37–51, 1992.

[7] J. Gilbert, G. Miller, and S. Teng. Geometric mesh partitioning: Implementation and experiments. *SIAM J. Scientific Computing*, 19(6):2091–2110, 1998.

[8] E. Helly. Über Mengen konvexer Körper mit gemeinschaftlichen Punkten. *Jber. Deutsch. Math*, 32:175–176, 1923.

[9] S. Jadhav and A. Mukhopadhyay. Computing a centerpoint of a finite planar set of points in linear time. *Discrete & Computational Geometry*, 12(3):291–312, 1994.

[10] J. Matoušek. Approximations and optimal geometric divide-and-conquer. In *23rd ACM Symp. Theory of Computing*, pages 512–522, 1991.

[11] N. Naor and M. Sharir. Computing a point in the center of a point set in three dimensions. In *CCCG: Canadian Conference in Computational Geometry*, 1990.

[12] R. Rado. A theorem on general measure. *J. Lond. Math. Soc.*, 21:291–300, 1947.

[13] J. Radon. Mengen Konvexer Körper, die Einen Gemeinschaftlichen Punkt Enthalten. *Mathematische Annalen*, 83:113–115, 1921.

[14] S.-H. Teng. *Points, Spheres, and Separators: A Unified Geometric Approach to Graph Partitioning*. PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, Aug. 1991. Available as Technical Report CMU-CS-91-184.

[15] H. Tverberg. A generalization of Radon's theorem. *J. Lond. Math. Soc.*, 41:123–128, 1966.

[16] K. Verbarg. Approximate center points in dense point sets. *Information Processing Letters*, 61(5):271–278, 1997.