Approximation Algorithms for Maximum Cliques in 3D Unit-Disk Graphs

Peyman Afshani and Timothy M. Chan
School of Computer Science
University of Waterloo
Waterloo, Ontario, N2L 3G1, Canada
{ pafshani, tmchan }@uwaterloo.ca

Abstract

We study two problems for a given n-point set in 3-space: finding a largest subset with diameter at most one, and finding a subset of k points with minimum diameter. For the former problem we suggest several polynomial-time algorithms with constant approximation factors, the best of which has factor $\pi/\arccos(1/3) < 2.553$. For the latter problem we observe that there is a polynomial-time approximation scheme.

1 Introduction

Clustering is an important and well-studied area of computational geometry [4]. In this paper, we study two basic problems in this area:

Problem 1 *Given a set S of n points in $\mathbb{R}^3$ and a value r, find a subset $A \subseteq S$ of diameter at most r having the maximum number k^* of points.*

Problem 2 *Given a set S of n points in $\mathbb{R}^3$ and an integer k, find a subset $A \subseteq S$ of k points having the minimum diameter r^*.*

These two problems are closely related: it is not difficult to see that a polynomial-time solution for one implies a polynomial-time solution for the other one. The first problem can be reformulated as finding a maximum clique in a unit-disk graph, as we may assume $r = 1$ without loss of generality. Although the general clique problem is NP-complete (and hard to approximate even to within factor of $n^{1-\epsilon}$ [6]), better results may be possible in geometric settings.

In two dimensions, efficient algorithms are known for both problems; for example, for Problem 2, Aggarwal et al. [1] gave an $O(n \log n + k^{2.5} n \log k)$-time solution; subsequently, Eppstein and Erickson [5] improved the time bound to $O(n \log n + nk^2 \log^2 k)$ (which was further improved to $O(n \log n + nk^2 \log k)$ [2, 3]). Also, if diameters are measured under the L_∞ metric instead of Euclidean, then polynomial-time methods exist in any fixed dimension [1, 4, 5]. However, no polynomial-time algorithms are known in the 3D Euclidean version; the best result reported [4, 5] has $O(n \log n + 2^{O(k)} n)$ complexity, which is exponential in k (though demonstrates fixed-parameter tractability).

In this paper, we present some approximation results for these two problems in 3D. Section 2 gives a straightforward $(1+\epsilon)$-factor approximation algorithm (PTAS) for Problem 2, while Section 3 presents several constant-factor approximation algorithms for Problem 1.

2 Problem 2: Approximating r^*

We first observe that approximating the minimum diameter r^* of a k-point subset is easy.

Theorem 3 *There is a PTAS for Problem 2 that computes a $(1+\epsilon)$-factor approximation in $O(n \log n + 2^{O(1/\epsilon^3)} n)$ time for any fixed $\epsilon > 0$.*

Proof. First compute a constant-factor approximation. As explained by Datta et al. [4], this can be accomplished in $O(n \log n)$ time by a simple grid method. Say the optimal diameter is between r and cr.

Form a uniform grid of side length ϵr. Round each input point to its nearest grid point. As a result, we obtain a multiset of grid points. For each grid point p, store a count of the input points mapped to p. Note that the value of the optimal diameter changes by an additive error of at most $O(\epsilon r)$. Thus a $(1 + O(\epsilon))$-factor approximate solution for the original point set can be obtained from an (exact) solution for the rounded multiset. The rounded problem can be solved as follows:

For each rounded point p, exhaustively generate all subsets of grid points (disregarding multiplicities) at distance at most $2cr$ from p. Since each such subset has size at most $O(1/\epsilon^3)$, there are at most $2^{O(1/\epsilon^3)}$ such subsets. For each such subset, if the total count of its points is at least k, compute the diameter of the subset in $O(1/\epsilon^{O(1)})$ time. We just return the minimum of all diameters found. □

3 Problem 1: Approximating k^*

We now study the problem of approximating the maximum size k^* of a clique in a unit-disk graph in $\mathbb{R}^3$. The rounding approach in the previous section no longer works, because small perturbations of the input points can cause drastic changes to k^*. We give several approaches yielding constant-factor approximations.

3.1 A Fast Factor-8 Algorithm

We begin with the simplest method:

Theorem 4 *There is a factor-8 approximation algorithm for Problem 1 that runs in $O(n \log n)$ time.*

Proof. Let c be a fixed small positive constant to be determined later. Build a grid of side length $1+c$. For each grid cell, divide it into 8 sub-cubes of side length $\frac{1+c}{2}$. Note that each sub-cube has diameter $\frac{\sqrt{3}}{2}(1+c) < 1$ by making c sufficiently small. Take the number of points in each of these sub-cubes and report the largest number over all grid cells. Now, repeat the same procedure $O(\frac{1}{c^3})$ times, but with the grid shifted by a vector (ci, cj, ck), for all combinations of $i, j, k \in \{0, 1, \ldots, \lfloor 1/c \rfloor\}$. Since c is a constant, this algorithm can be clearly implemented in $O(n \log n)$ time. We claim that the largest number reported is an 8-factor approximation.

Let A^* be the optimal subset. Since A^* has diameter at most one, the extent of A^* along each axis is at most one, so A^* fits inside a cube C^* of side length one. Among the various shifts of the grid, there must a grid cell C such that C^* completely fits inside C. Thus $A^* \subset C$. One of the 8 sub-cubes of C must contain at least $|A^*|/8$ points. □

3.2 A Fast Factor-7 Algorithm

We now refine the factor-8 algorithm by employing a more clever strategy to cover A^* by only 7 rectangular sub-boxes (not necessarily cubes) each of diameter at most one.

Theorem 5 *There is a factor-7 approximation algorithm for Problem 1 that runs in $O(n \log n)$ time.*

Proof. We follow the same approach as in the factor-8 algorithm, except we divide each $(1+c)\times(1+c)\times(1+c)$ grid cell C into sub-boxes in a different way. First divide C into a $(1+c) \times (1+c) \times 0.7(1+c)$ box A and a $(1+c) \times (1+c) \times 0.3(1+c)$ box B. Then divide A into four $0.5(1+c) \times 0.5(1+c) \times 0.7(1+c)$ sub-boxes, each with diameter $< 0.995(1+c)$. Divide B into two $0.8(1+c) \times 0.5(1+c) \times 0.3(1+c)$ sub-boxes, each with diameter $< 0.990(1+c)$, and one $0.2(1+c) \times 0.93(1+c) \times 0.3(1+c)$ sub-box, with diameter $< 0.998(1+c)$. By making c sufficiently small, the diameters of these 7 sub-boxes are all less than one.

The above strategy still leaves a small $0.2(1+c) \times 0.07(1+c) \times 0.3(1+c)$ sub-box R uncovered. We repeat the same strategy, this time leaving a different copy R' of R uncovered at the opposite corner of the cube. Again, we consider the number of points inside each of these sub-boxes generated and return the maximum.

To prove that the approximation factor is bounded by 7, we know that as before, the optimal subset A^* must be contained inside some grid cell C. Inside C, if both uncovered sub-boxes R and R' contain points of A^*, then A^* would contain two points of distance at least $\sqrt{(1-0.4)^2 + (1-0.14)^2 + (1-0.6)^2}(1+c) > 1.122(1+c) > 1$: a contradiction. So, one of R or R' is empty of points of A^*; w.l.o.g., say it is R. Then one of the 7 sub-boxes of $C - R$ must contain at least $|A^*|/7$ points. □

3.3 Factor 3.5

We show that with a larger running time, we can cut the approximation factor by half, by borrowing a technique of Aggarwal et al. [1]. (They originally used this technique to obtain an exact algorithm for the 2D version of Problem 1.)

Lemma 6 *If we can cover the input set with two given shapes each of diameter at most one then we can solve Problem 1 in $O(T_{\mathrm{match}}(n))$ time, where $T_{\mathrm{match}}(n)$ is the time required to solve the maximum matching problem for an n-vertex bipartite graph.*

Proof. A maximum clique in the unit-disk graph corresponds to a maximum independent set in the complement graph. Since points in each of the two shapes form a clique, the complement graph is bipartite. It is well known that for bipartite graphs the maximum independent set problem reduces to the maximum matching problem. □

The classical matching algorithm by Hopcroft and Karp [7] yielded $T_{\mathrm{match}}(n) = O(n^{2.5})$. More recently, Mucha and Sankowski [8] gave a randomized algorithm with $T_{\text{match}}(n) = O(n^\omega)$, where $\omega < 2.376$ is the matrix multiplication exponent.

Theorem 7 *There is a factor-3.5 approximation algorithm for Problem 1 that runs in $O(T_{\text{match}}(n))$ time.*

Proof. We modify the factor-7 algorithm: for each grid cell, instead of taking the number of points over each sub-box, we take the optimal answer for the subset of points from each pair of sub-boxes and report the largest answer.

In the correctness proof, since some pair of the 7 sub-boxes of $C - R$ must contain at least $2|A^*|/7$ points, the approximation factor is bounded by $7/2$. $\square$

3.4 Factor 2.553

We now give our best approximation-factor result. The key subproblem, as we have discovered already, is how to cover an unknown set A^* of diameter at most one with a small number of simple known shapes of diameter at most one. With rectangular boxes, we have demonstrated that 7 suffices. We now show how to reduce the number, to about 5.106 in an "average" sense, by using a more complicated shape which we call a "rounded diamond." Together with the earlier matching idea, the factor is further halved to 2.553.

Definition 8 Consider two points u and v with $||u - v|| = 1$ and two spheres C_u and C_v of radius one centered respectively at u and v. Let r be a point on the intersection of these two spheres. Let T be the curved triangle formed by the segment uv and the arcs ur and vr. Consider a rotation of T around the axis uv by an angle α which maps r to r' and T to T'. The resulting pyramid-like volume D is called an *rounded diamond of angle* α. (See Figure 1.)

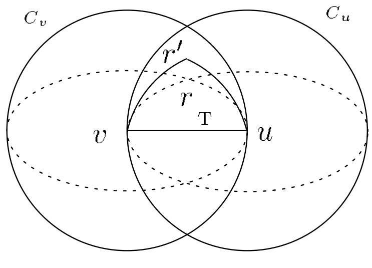

Figure 1: The rounded diamond D. Notice that the arcs ru and $r'u$ (resp. rv and $r'v$) are part of circles centered at v (resp. u).

In order to compute the diameter of this rounded diamond, we need the following technical lemma:

Lemma 9 *Given a point q and a circular arc st with center at o, if the ray $\overrightarrow{qo}$ does not hit the arc, then the farthest point on the arc from q is either s or t.*

Proof. W.l.o.g., say $q = (1, 0)$ and the arc is $\{(r\cos\theta, r\sin\theta) \mid \theta \in [\alpha_1, \alpha_2]\}$. Then we want to maximize $(r\cos\theta - 1)^2 + r^2\sin^2\theta = r^2 - 2r\cos\theta + 1$ over $\theta \in [\alpha_1, \alpha_2]$. Now, $\cos\theta$ over this range is minimized either at α_1 or α_2, assuming that $\pi \notin [\alpha_1, \alpha_2]$. $\square$

Lemma 10 *If $0 \le \alpha \le \frac{\pi}{2}$ then the diameter of D is equal to* $\max\{1, ||r - r'||\}$.

Proof. Consider two points p, q with maximum distance inside D. We can assume both p, q are on the surface of D or otherwise we can extend the segment pq and thus increase their distance. In particular, q is either on a flat portion T or T' or on a sphere C_u or C_v. We only address the cases when q is on T or C_v; the other cases are symmetric.

If q is on T, then by convexity q must be one of the points u, v, r or on one of the arcs ru, rv. W.l.o.g., we may assume that q is on ru.

If q is on C_v, then consider the rotation of q around the axis uv. We can apply Lemma 9 (to the 2-dimensional points obtained by projection along this axis) to conclude that in this case, the maximum distance is attained when q is either on the arc ru or $r'u$. W.l.o.g., we may again assume that q is on ru.

Now if q is on the arc ru, then consider the rotation of q around v in an axis perpendicular to T. We can apply Lemma 9 a second time to conclude that the maximum distance is attained when q is either r or u.

Thus the maximum distance occurs when $q \in \{u, v, r, r'\}$, and similarly when $p \in \{u, v, r, r'\}$. So indeed the diameter is $\max\{1, ||r - r'||\}$. $\square$

Corollary 11 *If $\alpha = \arccos(1/3)$ then the diameter of D is one.*

Proof. The distance of r or r' to the segment uv is $\frac{\sqrt{3}}{2}$. So, $||r - r'|| = \frac{\sqrt{3}}{2} \cdot 2\sin(\alpha/2)$, which is equal to 1 when $\alpha = 2\arcsin(1/\sqrt{3}) = \arccos(1/3)$. $\square$

Theorem 12 *There is an approximation algorithm for Problem 1 with factor $\pi/\arccos(1/3) < 2.553$ that runs in $O(n^3 T_{\text{match}}(n))$ time.*

Proof. Our algorithm tries all pairs of points $u, v \in S$ with $||u-v|| \leq 1$. Let v' be the point of distance 1 from u on $\overrightarrow{uv}$. Consider the *lune* L created by the intersection of the interior of the spheres C_u and $C_{v'}$. Consider a rounded diamond $D \subset L$ of angle $2\arccos(1/3)$ around uv. By Lemma 6 and Corollary 11 (since D is the union of two rounded diamonds of angle $\arccos(1/3)$), we can compute the optimal answer for the subset $S \cap D$. Do this for all possible placements of D rotated around uv. Return the largest answer found. Note that although technically there are infinitely many placements of D for each pair u, v, there are only $O(n)$ combinatorially different subsets $S \cap D$, so the running time is bounded by $O(n^3 T_{\text{match}}(n))$.

To prove correctness, suppose that $u, v \in S$ are in the optimal solution A^* with $||u-v||$ being the diameter of A^*. Then A^* must be contained inside the lune L. A complete 2π rotation of D around uv sweeps all of L. So, there must be a placement of D which contains at least $\frac{2\pi}{2\arccos(1/3)}|A^*|$ points. $\square$

4 Remarks

The algorithms in Sections 2 and 3.1 can be extended to any fixed dimensions, although it is unclear what the best approximation results are in higher dimensions using the approaches in Sections 3.2–3.4.

Besides improving the approximation factor for Problem 1, an even more basic and interesting question is to determine whether the 3D problem can actually be solved exactly in polynomial time, or whether one can prove NP-hardness in 3D or in other low dimensions.

References

[1] A. Aggarwal, H. Imai, N. Katoh, and S. Suri. Finding k points with minimum diameter and related problems. *Journal of Algorithms*, 12(1):38–56, 1991.

[2] K. Bhattacharya and H. ElGindy. Biased search and k-point clustering. In *Proceedings of the 9th Canadian Conference on Computational Geometry*, pages 141–146, 1997.

[3] T. M. Chan. Geometric applications of a randomized optimization technique. *Discrete and Computational Geometry*, 22(4):547–567, 1999.

[4] A. Datta, H.-P. Lenhof, C. Schwarz, and M. Smid. Static and dynamic algorithms for k-point clustering problems. *Journal of Algorithms*, 19:474–503, 1995.

[5] D. Eppstein and J. Erickson. Iterated nearest neighbors and finding minimal polytopes. *Discrete and Computational Geometry*, 11:321–350, 1994.

[6] J. Hastad. Clique is hard to approximate within $n^{1-\varepsilon}$. *Acta Mathematica*, 182:105–142, 1999.

[7] J. Hopcroft and R. M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973.

[8] M. Mucha and P. Sankowski. Maximum matchings via Gaussian elimination. In *Proceedings of the 45th Annual Symposium on Foundations of Computer Science*, pages 248–255, 2004.