Geometric Avatar Problems

Mario E. Consuegra[1] and Giri Narasimhan[1]

1 **School of Computing and Information Sciences, Florida International University, Miami, FL 33199, USA.** `{mcons004,giri}@fiu.edu`

Every man has the power to choose, but no power to escape the necessity of choice. – **Ayn Rand**

Abstract

We introduce the concept of *Avatar problems* that deal with situations where each entity has multiple copies or "avatars" and the solutions are constrained to use exactly one of the avatars. The resulting set of problems show a surprising range of hardness characteristics and elicit a variety of algorithmic solutions. Many avatar problems are considered. In particular, we show how to extend the concept of ϵ-kernels to find approximation algorithms for geometric avatar problems. Results for metric space graph avatar problems are also presented.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Avatar problems, choice

1 Introduction

We introduce a family of optimization problems which we call *Avatar problems*. The main feature of this family of problems is that their input entities have multiple replicas (or copies, or *avatars*), but their output is constrained to use exactly one of the copies. Avatar problems manifest themselves in many practical applications. For example, if disk storage systems have multiple copies of data items, then disk scheduling algorithms may process requests by visiting any one of the copies of each requested data to optimize the total access cost. Given any optimization (or decision) problem, its avatar version is required to achieve the same optimization (or decision) over all possible instances where each instance is created by assigning each element a_i to **exactly** one of k possible values. In this paper we investigate the complexity of *avatar* versions of classical algorithmic problems.

Avatar versions of NP-hard problems are easily shown to be NP-hard. However, designing good approximation algorithms often requires the solution of other avatar problems (e.g., avatar TSP can be well approximated by approximating avatar MST or avatar matching). This suggests the need for solving a family of avatar problems, over and beyond those with direct relevance to practical applications.

Related problems include *generalized MST* (MST spanning at least one vertex from each given set) [20], *group TSP* (minimum length tour visiting at least one vertex from each given group) [9], and *TSP with neighborhoods* (minimum length tour that visits each given neighborhood) [8, 22], all of which are NP-hard, even in Euclidean space. A closely related model is the indecisive (uncertain) points model [16], where input points have spatial uncertainty, but their true locations are known to be from a set of (possibly infinite) possibilities. Jørgensen et al. [16] suggest many applications for their model; these are applicable to the avatar model as well. For example, privacy considerations may prevent a database from storing the precise location of a person with a certain illness, but may provide a zip code; sensors may have limited accuracy and may provide approximate location data. A major difference with the avatar model is that in the uncertainty model, the power of choice lies with an adversary. Also see [5, 10–12, 14, 19, 21].

Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

The concept of switching graphs [15, 18] introduces another related model. In the area of scheduling, a related problem is the job interval selection problem (JISP) [7]. In this problem the input is a set of n jobs assigned to a worker. Each job is a set of one or more intervals on the real line, and we must select one interval for each job such that we schedule as many jobs as possible by picking non-overlapping intervals. In the k-avatar version of JISP each job is a set of at most k intervals. Avatar problems share some overlap with the area of parameterized complexity. The study of the complexity of a *k-avatar* problem as k goes from 1 to ∞ provides better understanding of the complexity landscape of the problem.

Results

The main results are summarized here, and is indicative of how "choice" affects the complexity of these problems in different ways.

1. In Section 2, we tackle two seemingly simple problems, minGap and maxGap, where the avatar versions result in natural optimization problems. We design a $\mathcal{O}(n^2 \log n)$-time algorithm for the 2-avatar maximum minGap problem for inputs in $\mathcal{R}^d$, and a 2-approximation $\mathcal{O}(n^3 \cdot k^3 \log(nk))$-time algorithm for the k-avatar minimum maxGap problem for points on a line.
2. In Section 3, we extend the concept of ϵ-kernels to the avatar world and show how to compute it efficiently for a k-avatar point set in $\mathbb{R}^d$. This enabled us to design a polynomial-time algorithm for finding an ϵ-approximate smallest convex hull for the k-avatar convex hull problem in $\mathbb{R}^d$. The ϵ-kernel result was also used to design polynomial-time $(1+\epsilon)$-approximation algorithms for the avatar versions of the following geometric problems: smallest volume axis-aligned enclosing hyperbox.
3. Reachability is a fundamental problem with linear-time algorithms for non-avatar inputs. Surprisingly, we show in Section 4 that for unweighted graphs, the k-avatar reachability problem is NP-complete. For weighted graphs, we show that the k-avatar shortest path problem is inapproximable to any constant factor unless $P = NP$.

We establish some basic notation for this paper. Let $L = \{a_1, a_2, \ldots, a_n\}$ be a set of n k-avatar entities. In other words, for each entity $a_i \in L$, one can assign a_i to one of the k avatar values from the set $Av(a_i) = \{v_i^{(1)}, v_i^{(2)}, \ldots, v_i^{(k)}\}$. An *avatar assignment* for entities in L, denoted by $A(\cdot)$, is an assignment of a single avatar value to each entity in L. Thus, $A(a_i) \in Av(a_i)$. Let $A(L)$ denote the set of values assigned to each element in L.

2 Avatar Minimum and Maximum Gaps

Given a set of points $\{x_1, \ldots, x_n\}$ on a line, we define the *minGap* (resp. *maxGap*) as the smallest (resp. largest) gap between consecutive items in the sorted order. The avatar version of the *maximum minGap* and *minimum maxGap* problems is: *Given a set of n k-avatar entities, find an avatar assignment that results in the maximum minGap (resp. minimum maxGap).* More formally, we are given a set of k-avatar entities $L = \{a_1, a_2, \ldots, a_n\}$, where each entity a_i can be assigned one of k values from the set $\{v_i^{(1)}, v_i^{(2)}, \ldots, v_i^{(k)}\}$.

2.1 Avatar Maximum minGap

We present a polynomial-time algorithm for the 2-avatar version of *maximum minGap* problem. It is clear that the minGap must be between a pair of points from the set of all avatar values, $\bigcup_{a_i \in L} Av(a_i) = \{v_1^{(1)}, \ldots, v_1^{(k)}, v_2^{(1)}, \ldots, v_2^{(k)}, \ldots, v_n^{(1)}, \ldots, v_n^{(k)}\}$. We first solve the decision problem of determining if there exists an avatar assignment so that the minGap is at least B; this is achieved by giving a polynomial-time reduction to 2SAT. The construction creates two complementary boolean variables, x_i and $\neg x_i$, to represent the two avatars of entity a_i. For every pair of values that are not avatars of each other and that have a distance of at most B, a clause is created to ensure that the corresponding boolean variables are not simultaneously set to true; a conjunction of these clauses generates an instance of 2SAT. It is easily shown that the resulting 2SAT formula is satisfiable if and only if the original 2-Avatar Maximum minGap problem has a minGap that is no smaller than B. Given the linear time algorithm for 2SAT [3], it is not difficult to see that the above algorithm takes $\mathcal{O}(n^2)$ time, and that the maximum minGap can be found in $\mathcal{O}(n^2 \log n)$ time by doing a binary search on the sorted list of all interpoint distances.

The above reduction to 2SAT for the 2-avatar minGap problem readily generalizes to the case where the entity values are points in d-dimensional space. However, the k-avatar minGap problem is NP-complete, and can be proved by a trivial adaptation of the proof of NP-Completeness of the problem of finding a *System of q-Distant Representatives* proved by Fiala et al. [13].

▶ **Theorem 1.** *The* 2*-avatar maximum minGap problem for* n *points in* $\mathbb{R}^d$ *can be solved in* $\mathcal{O}(n^2 \log n)$ *time. The corresponding* k*-avatar problem for* $k > 2$ *is NP-hard.*

2.2 Avatar Minimum maxGap

The avatar minimum maxGap problem appears to be harder than the avatar maximum minGap problem. While an exact polynomial time algorithm for the minimum maxGap problem remains open, below we present an approximation algorithm for the k-avatar minimum maxGap problem for points on a line.

Let B^* be the length of the minimum MaxGap, where the minimum is over all possible avatar assignments. We will perform binary search on the sorted list of all interpoint distances in order to find good lower and upper bounds B_{low} and B_{upp} for B^* such that $B_{\text{low}} \leq B^* \leq B_{\text{upp}}$. Establishing bounds for the ratio between the lower and upper bounds gives an approximation for B^*. A sorted list of interpoint distances can be computed in $O(n^2 k^2 \log nk)$. For a given value of B during this binary search we need to solve the decision problem of determining if there exists an avatar assignment so that the maxGap is at most B. The algorithm described below will give an approximate solution to this decision problem in the following sense. If the algorithm says "NO", then maxGap is greater than B. If the algorithm says "YES", then the maxGap is at most $2B$.

Let V denote the set of kn avatar values mapped on to the real line. Any avatar assignment is a subset of n points from V. A partition of the line into infinite number of disjoint abutting cells each of size B (see Fig. 1) is called a *valid* partition if there exists an avatar assignment such that all the points in the assignment are contained in a sequence of consecutive non-empty cells. Therefore, it follows that if there exists an avatar assignment for L such that the resulting point set has maxGap at most B then every partition of the line into infinite cells of size B is *valid.* The consequence is that if there is any partition of the line into infinite cells of size B that is not valid, then we know for sure that the maxGap for every assignment is greater than B. The difficulty is that the converse need not be true. Even though the assigned values appear in a sequence of consecutive cells, the maxGap could be between two items in adjacent cells that are nearly $2B$ apart, a key observation that leads to a 2-approximate algorithm. For example, in Fig. 1, v_6^2 and v_5^2 are in adjacent cells (of the partition with vertical dotted lines) but are almost $2B$ apart.

Given B, a fixed infinite partition of the line into cells of size B, and a fixed sequence

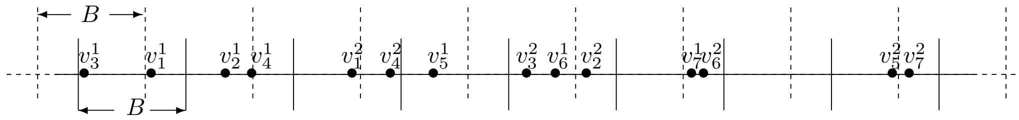

Figure 1 *Two different infinite partitions of the line are shown. The partition with dotted vertical lines is* valid*, while the partition with solid vertical lines is not* valid*. The valid partition is achieved by an avatar assignment that picks all choices with superscript 2.*

of consecutive cells, we check if that partition is valid for some avatar assignment of L by a reduction to Network Flow. Briefly, we construct a bipartite network where one partition P has vertices corresponding to entities $a_i \in L$ and the other partition Q has vertices corresponding to cells of the partition. There is an edge from a vertex $p \in P$ to a vertex $q \in Q$ if the entity corresponding to p has an avatar in the cell corresponding to q. Finally, the reduction involves showing that the network has a flow of n if and only if the partition is valid. For lack of space, details of the algorithm are omitted from this draft. As mentioned above, we perform binary search on the sorted list of interpoint distances until we find two adjacent gaps B_{i-1} and B_i in the list of gaps such that

1. $B_{i-1} \leq B_i$,
2. the algorithm returns `NO` for all partitions into cells of length B_{i-1}, and
3. Returns `YES` for at least one partition into cells of length B_i.

Thus, $B_{i-1} < B^*$. Since the smallest possible gap attainable that is larger than B_{i-1} is B_i, we have $B_i \leq B^*$. Also, since we have a partition into cells of length B_i for which we can find an avatar assignment where all the chosen points are in a set of adjacent cells such that each cell in that set contains a chosen point, we can use that avatar assignment to produce an assignment with a maximum gap that is no larger than $2 \cdot B_i$. Hence we have that $B_i \leq B^* \leq 2 \cdot B_i$. This gives us a polynomial-time 2-approximation algorithm for the 1D k-avatar minimum maxGap problem. The hardness of the avatar minimum maxGap for points in $\mathbb{R}^d$ remains open, even for $d = 1$.

▶ **Theorem 2.** *The k-avatar minimum maxGap problem for points on a line has a 2-approximate algorithm that runs in $O(n^3 \cdot k^3 \log(nk))$ time.*

3 Avatar Convex Hulls

Let L be a set of k-avatar entities where each entity can be assigned one of k different points in d-dimensional space. A k-avatar convex hull of L is a minimal convex set that contains at least one avatar for each entity $a \in L$. The aim is to minimize a specific measure such as the perimeter, surface area, or volume. The computational complexity of the problem of computing the avatar minimum convex hull remains an open problem. Related work includes results on the minimum and maximum convex hull for a set of points with imprecise locations [17, 23], and a recent paper by Abdullah et al. [1] for the model of uncertain points.

A *smallest avatar convex hull* is a convex hull that has minimum perimeter over all possible avatar assignments. Using the concept of ϵ-kernels, we present an algorithm that finds an ϵ-approximate smallest avatar convex hull for the k-avatar convex hull problem in $\mathbb{R}^d$. The results can be extended to minimum area/volume convex hulls.

3.1 Approximate Avatar Convex Hulls

For any point set $X \subset \mathbb{R}^d$, let $\omega(u, X)$ denote the *directional width* of X in direction u (see Fig. 2 (a)). A subset $Q \subseteq P$ is called an ϵ-approximation of P if for any direction $u \in S^{d-1}$ we have $(1-\epsilon)\omega(u, P) \leq \omega(u, Q) \leq (1+\epsilon)\omega(u, P)$. Our proposed algorithm finds an ϵ-approximate smallest convex hull $\mathcal{CH}(Q)$ by returning a set of avatar points $Q \subseteq A'(L)$ for some avatar assignment $A'(L)$ such that $(1-\epsilon)\omega(u, \mathcal{CH}^*(L)) \leq \omega(u, \mathcal{CH}(Q)) \leq (1+\epsilon)\omega(u, \mathcal{CH}^*(L))$, where $\mathcal{CH}^*(L)$ is the minimum avatar convex hull of L. Using the terminology of Agarwal et al. [2], one can think of the set Q as the avatar equivalent of an ϵ-kernel. This is formalized in the following definition of an *avatar ϵ-kernel* whose width along any direction is within a $1-\epsilon$ factor of the width of the optimal hull along that direction.

▶ **Definition 3.** Given a set L of n k-avatar entities in R^d, we say that a point set Q is an *avatar ϵ-kernel* of L if and only if $(1-\epsilon)\omega(u, \mathcal{CH}^*(L)) \leq \omega(u, Q) \leq (1+\epsilon)\omega(u, \mathcal{CH}^*(L)), \forall u \in \mathbb{S}^{d-1}$, where $\mathbb{S}^{d-1}$ is the unit hypersphere centered at the origin.

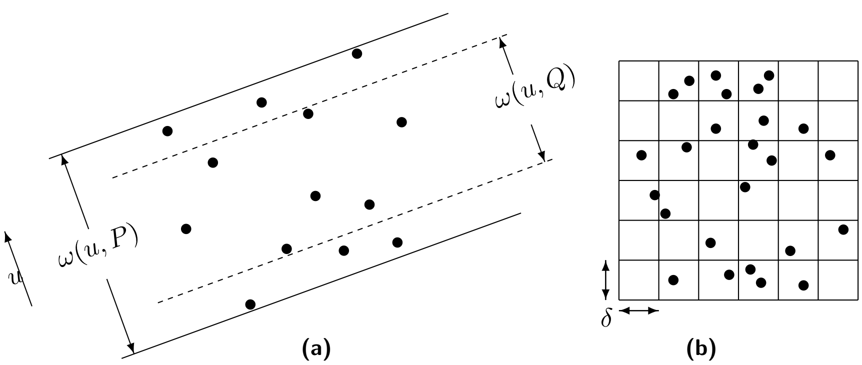

Figure 2 *(a) Directional Width (b) ϵ-grid Z*

The following procedure for finding a *diameter-oriented bounding box* $\mathcal{B}$ of a set S of points in R^d was described by Barequet and Har-Peled [4]. Let $\mathcal{D}(\mathcal{S})$ be the diameter of S and let $s_1, t_1 \in S$ s.t. $|s_1t_1| = \mathcal{D}(S)$. Let H be a hyperplane perpendicular to s_1t_1 and let Q be the orthogonal projection of S onto H. We again compute two points $s_2, t_2 \in Q$ s.t. $|s_2t_2| = \mathcal{D}(Q)$. Once again we project Q onto a hyperplane H' perpendicular to s_1t_1 and s_2t_2 and determine the diameter $\mathcal{D}(Q')$ of the projection Q onto H' and select two more points $s_3, t_3 \in Q'$ s.t. $|s_3t_3| = \mathcal{D}(Q')$. After d iterations of this process we have a diameter-oriented bounding box $\mathcal{B}(S)$ of S with the diameter in each iteration determined by the direction from s_i to t_i, for $i = 1, 2, \ldots, d-1$.

Note that $\mathcal{CH}^*$ must cover a set S of $2 \cdot d$ avatar points of an avatar assignment such that the diameter-oriented bounding box $\mathcal{B}(\mathcal{S})$ is exactly the same as the diameter-oriented bounding box $\mathcal{B}(\mathcal{CH}^*)$. See Algorithm 1 for the pseudocode for the following procedure. We pick all possible subsets of $2 \cdot d$ avatar points of L, of which there are $\binom{n \cdot k}{2 \cdot d}$. For each such subset S_i, we first check that no two points in S_i are in the same avatar set, then we find the diameter-oriented bounding box $B_i = \mathcal{B}(S_i)$. If every entity in L has an avatar point inside B_i then it is possible that $B_i = \mathcal{B}(\mathcal{CH}^*)$, otherwise we can discard B_i. We find an ϵ-approximate minimum avatar convex hull $\mathcal{CH}_i$ of all the points inside B_i and output the smallest one, which we refer to as $\mathcal{CH}_{\min}$. Since $\mathcal{B}(\mathcal{CH}^*) = B_i$ for some i, $\mathcal{CH}_{\min}$ will ϵ-approximate $\mathcal{CH}^*$. The following lemma from [2] is useful for this proof. We say that a point set is α-*fat* if its convex hull (a) is contained in a hypercube $\mathcal{H}$ and (b) contains a copy of $\mathcal{H}$ sharing the same center as $\mathcal{H}$, but shrunk by a factor $\alpha < 1$.

▶ **Lemma 4.** [2] *For any point set P with non-zero volume in $\mathbb{R}^d$ there exists an affine transform M s.t. $M(P)$ is an α-fat point set (for some $\alpha <$) where the hypercube $\mathbb{C} = [-1, +1]^d$ is the smallest enclosing box of $M(P)$ and s.t. a subset $Q \subseteq P$ is an ϵ-kernel of P iff $M(Q)$ is an ϵ-kernel of $M(P)$.*

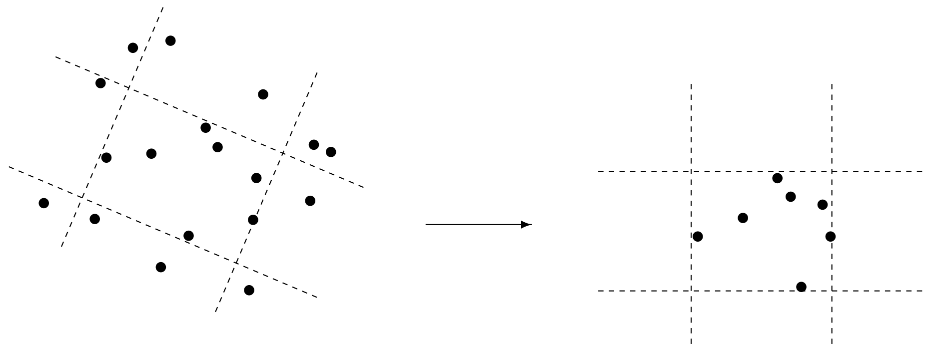

Figure 3 *Affine transform of space inside diameter-oriented bounding box of $2 \cdot d$ points.*

Algorithm 1 Computing ϵ-approximate smallest avatar convex hull

---

Require: L: set of n k-avatar entities; μ: a measure function of the size of the perimeter of a convex hull, $T(.)$ affine transform procedure

let $\mathcal{CH}_{min}$ = `null`

let S be the set of all possible sets of $2d$ avatar points of L.

for $S_i \in S$ **do**

  if no two points in S_i are avatars of each other **then**

    let $\mathcal{B}(S_i)$ be the *diameter-oriented bounding box*

    let B_i be the set of all avatar points inside $\mathcal{B}(S_i)$ such that every entity is represented in B_i

    let $\mathcal{CH}_i$ be the ϵ-approximate smallest avatar convex hull of $T(B_i)$ computed by Algorithm 2

    $\mathcal{CH}_{min} = \texttt{Min}(\mathcal{CH}_{min}, \mathcal{CH}_i)$

  end if

end for

return $\mathcal{CH}_{min}$

It is known that for a diameter-oriented bounding box B with largest side $\mathcal{D}$, if we appropriately expand or contract the box along each direction until it becomes a hypercube of side $\mathcal{D}$ and scale it to the hypercube $\mathbb{C}$, then the transformed point set is an α-fat point set (for some α <) in $\mathbb{C}$ [4]. This is illustrated by an example in Fig. 3. This transformation $T(B)$ of B as well as the transformed points can be computed in linear time, i.e., $O(n \cdot k)$ time. By Lemma 4, to compute an ϵ-approximate avatar convex hull of all the points in $\mathcal{B}$, we only need to compute an ϵ-approximate avatar convex hull of all the points in $\mathbb{C}$, which is computed as follows.

As in [4], let δ be the largest value such that $\delta \leq (\epsilon/\sqrt{d})\alpha$ and $1/\delta$ is an integer. We then partition the bounding hypercube into a uniform grid with cells of side length δ (see Fig. 2 (b)). However, applying the algorithm of Barequet and Har-Peled [4] does not help us to compute ϵ-kernels in $\mathbb{C}$ because now we must make sure not to pick two or more avatars of the same entity.

We need one other idea to compute ϵ-kernels in $\mathbb{C}$. The following procedure computes the ϵ-kernel in $T(B)$ (see Algorithm 2). Consider all possible assignments of binary values (0/1) to the cells in the grid (see Fig. 2 (b)). For the i^{th} binary assignment let Q_i be the set of cells that are assigned a value of 1. We call the set Q_i *legal* if each avatar entity has at least one element in at least one of the cells of Q_i, and it is possible to pick a representative point from each cell such that no two cells have representative points that are avatars of the same entity. Since there are $1/\delta^d$ cells, there are at most $2^{1/\delta^d}$ legal sets. In particular, if $A_{OPT}(\cdot)$ is the avatar assignment that leads to the optimal avatar convex hull, then it is easy to see that one of these legal sets must contain exactly the collection of cells with points from $A_{OPT}(\cdot)$.

It is clear that for any box B_i, with largest side of length $\mathcal{D}_i$, if we expand the box along each direction until it becomes a hypercube of side $\mathcal{D}_i$ and scale it to the unit hypercube $\mathbb{C}$, we are left with an α-fat point set in $\mathbb{C}$ (for some $\alpha <$) since $\mathcal{CH}(B_i)$ must cover all the points in S_i and hence it must touch each face of $\mathbb{C}$. This transformation $T(B_i)$ of B_i can be found in time linear in the number of points in B_i, which is equal to $O(nk)$. By Lemma 4, we know that finding an ϵ-approximate avatar convex hull of all the points in $\mathbb{C}$ gives us directly an ϵ-approximate avatar convex hull of all the points in $\mathcal{B}_i$.

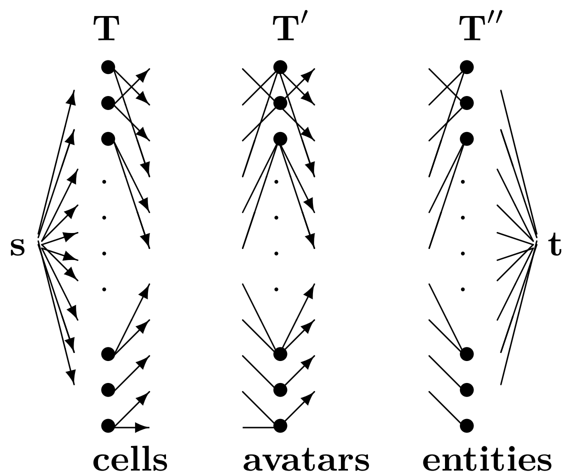

Figure 4 Reduction to network flow used to determine if a set of cells is legal.

We can determine if a given set of grid cells Q_i is legal by solving a network flow problem as follows. (See Fig. 4.) Create a set of vertices T such that each vertex in T represents a different cell in Q_i. Create a source vertex s with directed edges to each vertex in T. Create a set of vertices T' such that each vertex in T' represents a distinct point in some cell in Q_i. Add an edge from $u \in T$ to $u' \in T'$ if the cell in Q_i corresponding to u contains the point corresponding to u'. Create another set of vertices T'' such that each vertex in T'' corresponds to an avatar entity. Add an edge from $u' \in T'$ to $u'' \in T''$ if u' is a possible assignment for the avatar entity u''. Finally add a sink vertex t and connect all vertices in T'' to t by an edge. All edges have capacity 1. A maximum flow of size $|T|$ from s to t will identify a representative point in each cell such that no two points are avatars of the same entity. It is easy to see that such a flow exists if and only if the corresponding set of cells Q_i is legal. The following theorem formalizes the result.

▶ **Theorem 5.** *There is an algorithm that finds an ϵ-approximate min-perimeter k-avatar convex hull in time $O((nk)^{(2d+3)} \cdot \frac{n}{\delta^d} \cdot (2d)^2 \cdot 2^{\frac{1}{\delta^d}} (\frac{2}{\delta^{d-1}})^{\lfloor \frac{d}{2} \rfloor})$ by finding an avatar ϵ-kernel Q*

Algorithm 2 ϵ-APPROXIMATION OF SMALLEST AVATAR CONVEX HULL FOR α-FAT AVATAR POINT SET

Require: P: an α-fat set (for some $\alpha <$) of k-avatar points inside the unit hypercube $\mathbb{C}$
  μ: measure function of the size of the perimeter
  let δ be the largest integer s.t. $\delta \leq (\epsilon/\sqrt{d})\alpha$
  let Z be a d-dimensional grid of cell size δ
  for each assignment of binary values (0/1) to the cells in the grid Z **do**
    `let` Q_i be the set of cells assigned with a 1 in the i^{th} binary assignment
    if Q_i is *legal* **then**
      `let` $Q'_i \subseteq Q_i$ be the collection of highest and lowest cells in every hypercolumn containing at least one cell of Q_i
      `let` Q' be the set of representative points of cells in Q'_i
      `let` $\mathcal{CH}_i = \mathcal{CH}(Q')$
      if $\mu(\mathcal{CH}_i) < \mu(\mathcal{CH}_{min})$ **then**
        $\mathcal{CH}_{min} = \mathcal{CH}_i$
      end if
    end if
  end for
  return $\mathcal{CH}_{min}$

of L, which by Definition 3 satisfies $(1-\epsilon)\omega(u, \mathcal{CH}^*(L)) \leq \omega(u, \mathcal{CH}(Q)), \quad \forall u \in S^{d-1}$. *Note that the choice of constant* δ *depends on* k, ϵ, *and* α.

The proof is sketched as follows. Given a legal set, Q_i, let $Q'_i \subseteq Q_i$ be the collection of highest and lowest cells in every hypercolumn containing at least one cell of Q_i. Let Q (resp., Q') be the set of representative points of cells in Q_i (resp., Q'_i). It is easy to see that Q is an ϵ-kernel of Q'. We argue that $A_{OPT}(\cdot)$, the avatar assignment that leads to the optimal avatar convex hull, occupies a collection of cells (call this set of cells Q_{OPT}), which would have been considered by our algorithm. While the algorithm may not have picked the points in the optimal avatar assignment, it is sure to pick one representative point from each of the cells in Q_{OPT}. Since for each point in the legal set, there is at least one representative point that is within distance $\epsilon \cdot \alpha$ for every point in the optimal avatar assignment, we immediately have an avatar ϵ-kernel of the original input.

3.2 Approximate Smallest Volume Enclosing Hyperbox

Using ϵ-kernels we prove the following theorem.

▶ **Theorem 6.** *Given an exact algorithm for finding the min-volume axis-aligned enclosing hyperbox that runs in time* $O(n^a)$, *there exists an algorithm that finds a* $(1+\epsilon)$*-approximate smallest volume axis-aligned avatar enclosing hyperbox in time* $O((nk)^{(2d+3)} \cdot \frac{n}{\delta^d} \cdot (2d)^2 \cdot 2^{\frac{1}{\delta^d}}(\frac{2}{\delta^{d-1}})^{\lfloor \frac{d}{2} \rfloor} + (\frac{2}{\delta^{d-1}})^a)$. *Note that the choice of constant* δ *depends on* k, ϵ, *and* α.

Proof. We can compute a $(1+\epsilon)$-approximate smallest volume axis-aligned enclosing hyperbox $B(L)$ containing an avatar of each entity in L after finding an ϵ'-kernel of L, for some constant ϵ'. Let $\mathcal{CH}(L)$ be the smallest avatar convex hull of a set L of k-avatar points. If

Q is a k-avatar ϵ'-kernel of L such that $Q \subset \mathcal{CH}(L)$, then we have:

$$\begin{aligned}
(1-\epsilon') \cdot \omega(u, L) &\leq \omega(u, Q), \quad \forall u \in S^{d-1} \\
(1-\epsilon') \cdot \omega(u, L) &\leq \omega(u, Q), \quad \forall u \in [d] = \{e_1, e_2, \ldots, e_d\} \\
(1-\epsilon')^d \prod_{u \in [d]} \omega(u, L) &\leq \prod_{u \in [d]} \omega(u, Q)
\end{aligned}$$

There exists a constant c (function of ϵ' and d), such that $(1 - c\epsilon') \leq (1-\epsilon')^d$, thus implying the following:

$$\begin{aligned}
(1-c\epsilon') \prod_{u \in [d]} \omega(u, L) &\leq \prod_{u \in [d]} \omega(u, Q) \\
(1-c\epsilon') \cdot Volume(B(L)) &\leq Volume(B(Q))
\end{aligned}$$

By choosing $\epsilon = \frac{1}{1-c\epsilon'}$, we obtain a $(1+\epsilon)$-approximation of the smallest volume axis-aligned enclosing rectangle, since

$$1 \leq \frac{(1+\epsilon) \cdot Volume(B(Q))}{Volume(B(L))} \leq (1+\epsilon)$$

◀

Similar results can be achieved for an approximate min-diameter (see Section 3.3) and min-perimeter axis-aligned avatar enclosing box.

3.3 $(1+\epsilon)$-Approximate Avatar Diameter

This section gives yet another result using ϵ-kernels.

▶ **Definition 7.** Define the minimum *avatar* diameter $diam(L)$ of a set L of avatar points as the diameter of the *avatar assignment* $A(L)$ with the smallest diameter.

▶ **Theorem 8.** *Given an exact algorithm for finding the diameter of a convex hull that runs in time $O(n^a)$, there exists an algorithm that computes a $(1+\epsilon)$-approximate smallest k-avatar diameter in time $O((nk)^{(2d+3)} \cdot \frac{n}{\delta^d} \cdot (2d)^2 \cdot 2^{\frac{1}{\delta^d}} (\frac{2}{\delta^{d-1}})^{\lfloor \frac{d}{2} \rfloor} + (\frac{2}{\delta^{d-1}})^a)$. Note that the choice of constant δ depends on k, ϵ, and α.*

Proof. We can use the procedure described in Section 3 to find an *avatar* ϵ'-kernel Q. (We find Q by finding an ϵ'-approximate smallest convex hull $\mathcal{CH}(Q)$ of L.) Our measure function is $\mu(.) = diam(.)$. This measure function $diam(.)$ has the property that $diam(L) = diam(\mathcal{CH}^*(L))$. Let $\bar{u} \in S^{d-1}$ be the direction of $diam(L)$. Then we have that:

$$\begin{aligned}
(1-\epsilon')\omega(u, L) &\leq \omega(u, Q), \quad \forall u \in S^{d-1} \\
(1-\epsilon') \cdot diam(L) &\leq \omega(\bar{u}, Q) \\
&\leq diam(Q) \\
diam(L) &\leq (1+\epsilon) \cdot diam(Q), \quad \text{where } \epsilon = \frac{1}{1-\epsilon'}
\end{aligned}$$

Thus we can just return $(1+\epsilon) \cdot diam(Q)$, which gives us an $(1+\epsilon)$-approximate minimum avatar diameter knowing that:

$$\begin{aligned}
diam(Q) \leq diam(L) &\leq (1+\epsilon) \cdot diam(Q) \\
1 \leq \frac{(1+\epsilon) \cdot diam(Q)}{diam(L)} &\leq (1+\epsilon)
\end{aligned}$$

◀

4 Avatar Problems in Graphs and Metric Spaces

In this section we consider the hardness of the avatar versions of vertex reachability and shortest paths in unweighted graphs. The results easily generalize to weighted graphs and metric spaces. Vertex reachability has ties to *rainbow connectivity* problems from the graph theory literature [6]. As before, in order to set the stage, we provide some formal definitions.

4.1 Avatar graph reachability

A k-avatar graph $G(V, E, L, \mathcal{A})$ (or simply an "avatar" graph) consists of the following: a set of vertices V; a set of edges E connecting pairs of vertices in V; a set of entities $L = \{a_1, \ldots, a_m\}$; and a collection of disjoint avatar sets $\mathcal{A} = \{A_1, \ldots, A_m\}$ such that $\forall i$, $A_i \subseteq V$ is the avatar set for entity i, $|A_i| \leq k$, and $A_i \cap A_j = \phi$ if $i \neq j$. An *avatar path* in G is a path p such that no two vertices on the path p are avatars of the same entity.

The *k-avatar reachability problem* is stated as follows: *Given an avatar graph G and two vertices s and t in G determine wether there is an avatar path p from s to t.* Reachability is a fundamental graph problem and can be solved in linear time using simple techniques such as DFS or BFS. Surprisingly enough, in the avatar setting it turns out to be NP-complete, even for $k = 2$.

▶ **Theorem 9.** *The k-avatar reachability problem is NP-complete, for $k \geq 2$.*

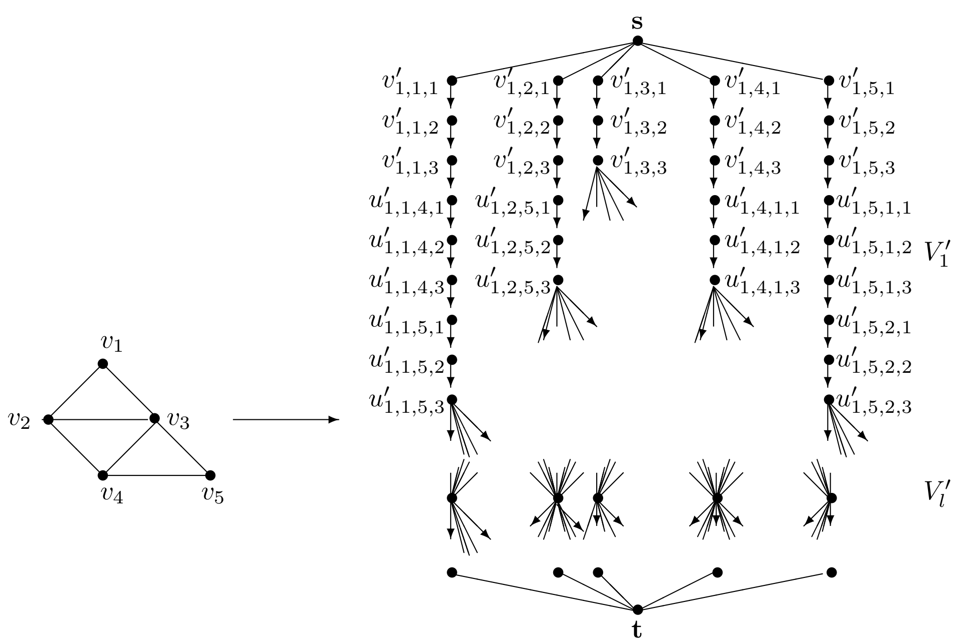

Figure 5 *Sketch of reduction from CLIQUE to avatar vertex reachability.*

Proof. The reduction is from the CLIQUE problem. Let graph $G_C(V, E)$ and integer k denote an instance of the CLIQUE problem. (G_C, k) is a YES instance if and only if G_C contains a clique of size k. We construct graph $G_A(V', E')$ as follows (see Fig. 5): create $k + 2$ layers of vertex sets, $V'_0, V'_1, \ldots, V'_{k+1}$. Let $V'_0 = \{s\}$ and $V'_{k+1} = \{t\}$. For $l = 1, \ldots, k$, let $V'_l = \{v'_{l,i,j} : 1 \leq i \leq |V|, 1 \leq j \leq k\}$. Vertices $v'_{x,i,y}$ and $v'_{y,i,x}$ correspond to avatars of the same entity A_i (prevents picking same vertex from 2 different layers). Add edges $(v'_{l,i,j}, v'_{l,i,j+1})$ for all l, i, and j; for $0 \le l < k$, add edges $(v'_{l,i,k}, v'_{l+1,j,1})$ for all i, j. Note that the vertices $v'_{l,i,1}, \dots, v'_{l,i,k}$ in layer l form k connected subpaths. Denote the subpath from $v'_{l,i,1}$ to $v'_{l,i,k}$ by $S_{l,i}$. It is important to note that for each vertex $v_i \in V$, there is exactly one corresponding subpath in each layer V'_l. Now for each pair of non-adjacent vertices $v_i, v_j \in V$ from G_C, add vertices $u'_{l,i,j,x}$ and $u'_{l,j,i,x}$, for all $1 \le x \le k$ and $1 \le l \le k$ to G_A. Add edges $(u'_{l,i,j,x}, u'_{l,i,j,x+1})$ and $(u'_{l,j,i,x}, u'_{l,j,i,x+1})$, $1 \le x < k$. Update subpaths $S_{l,i}$ by taking each outgoing edge from it and make it outgoing from $u'_{l,i,j,k}$ instead, and then adding an edge from it to $u'_{l,i,j,1}$, effectively making $u'_{l,i,j,1}$ the new last vertex of subpath $S_{l,i}$. Finally, let each pair of vertices $u'_{x,i,j,y}, u'_{y,j,i,x}$ be avatars of each other, for all $1 \le x, y \le k$.

It is not hard to see that the size of the graph G_A is polynomial in n. More importantly, we claim that if the set $\{v_{i_1}, \dots, v_{i_k}\}$ is a clique in G_C, then a 2-avatar path can be found from s to t in G_A by starting at s (layer V'_0), moving from each layer to the next, selecting in each layer a subpath corresponding to a distinct vertex from the clique. Intuitively, if the path in level l goes through vertices of the form $v'_{l,i,j}$, then vertex i is chosen as the l-th vertex in the clique. Furthermore, the vertices of the form $u'_{l,i,j,x}$ that are required to be visited by the path (and its avatars) ensure that the other vertices picked for the clique are indeed adjacent to i. The converse is proved by starting from a 2-avatar path and selecting the clique vertices based on the subpaths traversed in each layer. Hence there is a clique of size k in G_C if and only if there is a 2-avatar $s \rightsquigarrow t$ path in G_A. It is readily shown that 2-avatar reachability is in NP, thus completing the proof that it is NP-complete. ◀

4.2 Avatar Shortest Paths

Given an unweighted (or unit-weighted) k-avatar graph and two vertices s and t in the graph, find the shortest length avatar path from s to t. Given that reachability is hard in the avatar setting, the shortest path would be expected to be at least as hard. The following theorem highlights its inapproximability.

▶ **Theorem 10.** *The k-avatar shortest path problem is APX-Hard for $k \ge 2$.*

The key to the proof is a gap-preserving reduction from the maximum clique problem, which is known to be APX-hard, implying immediately that it cannot be approximated to within any constant factor in polynomial time. The proof is omitted here because of limited space.

▶ **Lemma 11.** *There is a gap-preserving reduction from the max-clique problem to the 2-avatar shortest path problem that transforms a graph $G_{\text{clique}}(V, E)$ to a graph $G_{\text{avatar}}(V', E')$ such that:*

1. *if $OPT_{\text{clique}}(G) \ge k \cdot |V|$, $OPT_{\text{avatar}}(s - t) \le m$, and*
2. *if $OPT_{clique}(G) < \alpha \cdot (k \cdot |V|)$, $OPT_{\text{avatar}}(s - t) > (2 - \alpha) \cdot m$,*

where $m = (k^2 \cdot |V|^3) + 1$, and α and k are any constants such that $0 \le \alpha, k \le 1$. Here $OPT_{\text{clique}}(G)$ is the size of the maximum clique in $G_{\text{clique}}(V, E)$, and $OPT_{\text{avatar}}(s - t)$ is the length of the shortest 2-avatar path from a vertex s to a vertex t in $G_{\text{avatar}}(V', E')$.

Open Problems: Open problems from this work include determining the time complexity of the k-avatar versions of minimum MaxGap problem and convex hull.

ACKNOWLEDGMENTS: This work was partly supported by NSF Grant (CNS-1018262) and the NSF Graduate Research Fellowship (DGE-1038321). The authors thank Shin-ichi Tanigawa, Daniel Rodriguez, Joshua Kirstein, and Ning Xie for useful discussions on earlier drafts. The detailed reviews by anonymous FSTTCS reviewers is gratefully acknowledged.

References

1. A. Abdullah, S. Daruki, and J. M. Phillips. Range counting coresets for uncertain data. In *Proceedings of the twenty-ninth annual symposium on Computational geometry*, SoCG '13, pages 223–232, New York, NY, USA, 2013. ACM.
2. P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Approximating extent measures of points. *J. ACM*, 51(4):606–635, July 2004.
3. B. Aspvall, M. F. Plass, and R. E. Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Information Processing Letters*, 8(3):121–123, 1979.
4. G. Barequet and S. Har-Peled. Efficiently approximating the minimum-volume bounding box of a point set in three dimensions. *Journal of Algorithms*, 38(1):91 – 109, 2001.
5. R. Bruce, M. Hoffmann, D. Krizanc, and R. Raman. Efficient update strategies for geometric computing with uncertainty. *Theory of Computing Systems*, 38(4):411–423, 2005.
6. G. Chartrand, G. L. Johns, K. A. McKeon, and P. Zhang. The rainbow connectivity of a graph. *Networks*, 54(2):75–81, 2009.
7. J. Chuzhoy, R. Ostrovsky, and Y. Rabani. Approximation algorithms for the job interval selection problem and related scheduling problems. *Mathematics of Operations Research*, 31(4):730–738, 2006.
8. A. Dumitrescu and J. S. B. Mitchell. Approximation algorithms for TSP with neighborhoods in the plane. *Journal of Algorithms*, 48(1):135 – 159, 2003.
9. K. Elbassioni, A.V. Fishkin, N.H. Mustafa, and R. Sitters. Approximation algorithms for euclidean group TSP. In *Proc. ICALP*, pages 1115–1126. Springer, 2005.
10. T. Erlebach, M. Hoffmann, D. Krizanc, M. Mihal'ák, and R. Raman. Computing minimum spanning trees with uncertainty. In *Proc. of the 25th Annual STACS*, pages 277–288, 2008.
11. T. Feder, R. Motwani, L. O'Callaghan, C. Olston, and R. Panigrahy. Computing shortest paths with uncertainty. *Journal of Algorithms*, 62(1):1–18, 2007.
12. T. Feder, R. Motwani, R. Panigrahy, C. Olston, and J. Widom. Computing the median with uncertainty. In *Proc. of the 32nd annual ACM STOC*, pages 602–607. ACM, 2000.
13. J. Fiala, J. Kratochvíl, and A. Proskurowski. Systems of distant representatives. *Discrete Appl. Math.*, 145(2):306–316, January 2005.
14. A. Goel, S. Guha, and K. Munagala. Asking the right questions: Model-driven optimization using probes. In *Proc. of the 25th ACM Symposium on PODS*, pages 203–212. ACM, 2006.
15. J. Groote and B. Ploeger. Switching graphs. *Intl J Found Comp Sci*, 20(5):869–886, 2009.
16. A. Jørgensen, M. Löffler, and J. M. Phillips. Geometric computations on indecisive points. In F. Dehne, J. Iacono, and J.-R. Sack, editors, *Algorithms and Data Structures*, volume 6844 of *LNCS*, pages 536–547. Springer, 2011.
17. W. Ju and J. Luo. New algorithms for computing maximum perimeter and maximum area of the convex hull of imprecise inputs based on the parallel line segment model. In *Proceedings of the CCCG*, 2009.
18. B. Katz, I. Rutter, and G. J. Woeginger. An algorithmic study of switch graphs. *Acta Inf.*, 49(5):295–312, 2012.
19. S. Khanna and W.-C. Tan. On computing functions with uncertainty. In *Proc. of the 20th ACM Symposium on PODS*, pages 171–182. ACM, 2001.
20. Y.-S. Myung, C.-H. Lee, and D.-W. Tcha. On the generalized minimum spanning tree problem. *Networks*, 26(4):231–241, 1995.
21. C. Olston and J. Widom. Offering a precision-performance tradeoff for aggregation queries. In *Proc. of VLDB*, pages 144–155. Morgan Kaufmann, 2000.
22. S. Safra and O. Schwartz. On the complexity of approximating TSP with neighborhoods and related problems. *Comput. Complex.*, 14:281–307, March 2006.
23. M. van Kreveld and M. Löffler. Largest and smallest convex hulls for imprecise points. *Algorithmica*, 56(2):235–269, February 2010.