

MINIMIZING CO-LOCATION POTENTIAL OF MOVING ENTITIES*

WILLIAM EVANS[†], DAVID KIRKPATRICK[†], MAARTEN LÖFFLER[‡], AND
FRANK STAALS[‡]

Abstract. We study the problem of maintaining knowledge of the locations of n entities that are moving, each with some, possibly different, upper bound on their speed. We assume a setting where we can query the current location of any one entity, but this query takes a unit of time, during which we cannot query any other entities. In this model, we can never know the exact locations of all entities at any one time. Instead, we wish to minimize uncertainty concerning the locations of all entities at some target time that is t units in the future. We measure uncertainty by the *ply* of the potential locations: the maximum over all points x of the number of entities that could potentially be at x . Since the ply could be large for every query strategy, we analyze the performance of our query strategy in a competitive framework: we consider the worst-case ratio of the ply achieved by our strategy to the *intrinsic ply* (the smallest ply achievable by any strategy, even one that knows in advance the full trajectories of all entities). We describe an efficient strategy that, knowing only an upper bound on the speed of individual entities, is $O(k)$ -competitive, provided the lead time t is at least $2n$ and the number of different entity speed classes (groups of entities whose speed bounds differ by at most a factor of two) is at most k . (This contrasts with the fact that, even given the full trajectories, the problem of computing the intrinsic ply is NP-hard.) If t is small, though at least n , and the entities move in any constant dimension d , our strategy is $O(k(\frac{\tilde{T}}{n})^{d-\frac{d}{d+1}})$ -competitive, where \tilde{T} is the median of the lengths of time since the n entity locations were last known precisely. Matching lower bounds demonstrate that our strategy, in all cases, is optimally competitive, up to constant factors.

Key words. data in motion, ply, competitive analysis, input impression

AMS subject classifications. 68Q25, 68W25, 68W27

DOI. 10.1137/15M1031217

1. Introduction. Due to the rapid growth in availability of cheap location-aware mobile devices, *data in motion* is increasingly becoming a topic of interest to researchers in various application fields, such as GIS, sensor networks, social networks, robotics, etc. [2, 8, 22, 28]. Although these applications are very different, they share some important characteristics: the motion they deal with is often *unpredictable* and data must be processed in *real-time*. These two aspects make it hard to apply traditional geometric algorithms.

The first challenge arises from the unpredictable nature of the data. In computational geometry, there is an extensive history on data in motion: kinetic data models [1, 3, 14, 15] can be used to study the complexity and structural changes of geometric objects in motion. However, they were developed to deal with moving data in a controlled environment and rely on the possibility to predict, at least locally, the trajectory that a moving point will follow. More recently, several efforts have been made to deal with data in much less restricted models of motion (though

*Received by the editors July 17, 2015; accepted for publication (in revised form) August 9, 2016; published electronically October 19, 2016. This paper is an extended version of two previous results: one required all entities to have the same speed bound [10], and the other allowed different speed bounds but only considered the case when lead time is large [11].

<http://www.siam.org/journals/sicomp/45-5/M103121.html>

Funding: The first and second authors were supported by NSERC Discovery Grants. The third and fourth authors were supported by the Netherlands Organisation for Scientific Research (NWO) under grants 639.021.123 and 612.001.022.

[†]University of British Columbia, Vancouver, BC, Canada (will@cs.ubc.ca, kirk@cs.ubc.ca).

[‡]Utrecht University, Utrecht, The Netherlands (m.loffler@uu.nl, f.staals@uu.nl).

some assumptions are still necessary, such as an upper bound on the speed of moving points) [6, 7, 9, 13, 20, 25, 27].

The second challenge is obtaining and processing the data in real-time. Traditional algorithms work on the assumption that data is first collected and stored in a central location, and then analyzed, preferably as quickly as possible (i.e., minimizing computation time). The *online algorithms* framework lifts this assumption and allows data to be processed as it arrives, but it ignores the cost of obtaining the data and does not permit the algorithm to influence the order in which the data arrives. In modern applications, a significant cost may be associated with obtaining, for example, the location of a moving entity, and computation time may not be the restricting factor. One way to model this, called the *update complexity* model, assumes one is given an incomplete or imprecise data set on which some structure (for example, the Delaunay triangulation) must be computed using a minimum number of “updates” [4, 12, 16, 17, 23, 24]. Here, an update typically returns additional information (precision) about the location of a specified data point. However, this has mostly been considered in a static context.

Model and contribution. When dealing with real-time data in motion that takes a certain amount of time to obtain, minimizing the number of required updates may not be the true goal. Instead, we should try to maximize the effect of the updates that we can make, knowing that each update of one entity takes time during which all entities may move.

There is a natural connection between data imprecision and data in unpredictable motion. When one knows the location of a moving entity at a certain time but does not continuously refresh this information, the entity will, in general, be somewhere in a region of potential locations (e.g., a ball centered at its last known location). A natural question is whether it is possible to control the collective uncertainty of a set of imprecise and moving entities using queries, that take unit time, to individual entities.

In this paper, we consider one fundamental interpretation of this question. Given a set of entities moving unpredictably, each with a, possibly different, bounded speed, we wish to maintain knowledge of their potential locations, by querying individual entities, in such a way that their co-location potential is minimized. We choose a worst-case measure of co-location potential at a fixed time as the *ply* [19] of the regions of potential locations: the maximum, over all points x , of the number of regions that contain x . Of course, the co-location potential at any fixed target time is impacted by (i) the uncertainty of the entity locations prior to any queries, (ii) the number of queries we can make before the target time, and (iii) the maximum speeds of the individual entities.

It might seem equally compelling to consider a measure that integrates co-location potential over all target times. As we note in the concluding section, our results extend directly to a recurrent version in which ply is measured at well-spaced target times. The continuous version, where the goal is to minimize ply at all times, seems to require a fundamentally different strategy. It is also interesting to consider other measures of co-location potential. Our choice of ply—basically the size of the maximum clique in the intersection graph of the entity uncertainty regions—could, for example, be replaced by the maximum or average degree in this same graph. However, unlike degree, ply reflects a real potential for congestion and relates directly, in many applications involving imprecise numbers or points, to the actual cost of resolving input uncertainty [5, 18]. For example, a set of n points in \mathbb{R} whose uncertainty regions have low ply can be considered partially sorted, making it possible, among other things, to perform approximate selection without further computation.

To set our problem more formally, let \mathcal{E} be a set $\{e_1, e_2, \dots, e_n\}$ of entities, where every entity e_i has an associated (unknown) *trajectory* f_i , which is a continuous function from the time remaining (until the target time) to a position in \mathbb{R}^d (for constant dimension d). Entity e_i has a (known) maximum speed of σ_i units of distance per unit of time. We wish to maintain knowledge of the locations of these entities at the target time. To do this, we are allowed to choose at most one entity to *query* at each integer time step, reflecting the unit cost of performing a query (multiple queries per time step could be simulated by reducing the size of the time step). From each query, we obtain the exact location of the chosen entity at that time. If our last query to entity e_i occurs with remaining-time t (until the target time), then at the target time, e_i will be somewhere within the d -dimensional ball $B_i(t)$, centered at $f_i(t)$ with diameter $2\sigma_i t$. We refer to this ball $B_i(t)$ as the *projected uncertainty region* for e_i or, since we are only ever concerned with uncertainty of locations at the target time, simply as the *uncertainty region* for e_i . Furthermore, since the uncertainty region associated with a given query, unless the entity is re Queried, persists from the time of the query until the target time, we refer to t as the *lifespan* of the uncertainty region. Subsequent queries to the entity lead to a new uncertainty region of shorter lifespan and a correspondingly smaller diameter; in fact, $B_i(t') \subseteq B_i(t)$ for remaining-times $t' \leq t$. It is worth reemphasizing that uncertainty *always* refers to the uncertainty of the entities' locations at the target time.

What constitutes a good query strategy? It is clear that worst-case analysis of the behavior of a query strategy is inadequate: it might not be possible to ensure low ply with any strategy if the entities move too close together. Therefore, we evaluate strategies by introducing a form of competitive analysis. We say that the collection $\mathcal{B} = \{B_1, B_2, \dots, B_n\}$ of balls forms an *uncertainty realization* of \mathcal{E} if there exists a sequence of *distinct, positive* integers t_1, t_2, \dots, t_n such that B_i has center $f_i(t_i)$ and diameter $2\sigma_i t_i$. We refer to the minimum, over all uncertainty realizations \mathcal{B} of \mathcal{E} , of the ply of \mathcal{B} as the *intrinsic ply* of the set \mathcal{E} . Since the set of uncertainty regions determined by any query strategy provides an uncertainty realization of the set of entities \mathcal{E} , the intrinsic ply of \mathcal{E} is the smallest ply that could be achieved by *any* query strategy, even one that has full knowledge of the trajectories of the entities involved.

What may we assume about our knowledge of the positions of the entities in \mathcal{E} prior to making any queries? We could assume that we start with exact knowledge of their current locations but this seems rather unrealistic. Instead, we assume, when the strategy starts t time steps before the target time, that the exact location of each entity e_i is known with time $t_i > t$ remaining before the target time and hence each entity comes with a corresponding initial projected uncertainty region. This initial uncertainty affects the ability of a query strategy to reduce ply. It turns out that the median of the t_i values, which we denote by T , is useful for expressing the dependency of our results on the state of initial uncertainty. That dependency can also be expressed in a more nuanced fashion, which involves the numbers of entities with different initial uncertainty region sizes. This more nuanced dependence is described in the discussion of the performance of the strategy. Figure 1 illustrates a collection of trajectories and their associated uncertainty regions for $d = 1$.

It is also necessary to express the range of maximum speeds of the entities in \mathcal{E} . Entities whose maximum speed is within a factor of two of each other can be treated similarly by a query strategy. A *speed class* is a set, S , of entities such that $\max_{e_i \in S} \sigma_i \leq 2 \min_{e_i \in S} \sigma_i$. Our measure of the range of maximum speeds is the minimum k such that there exists a partition of \mathcal{E} into speed classes $\mathcal{E}_1, \mathcal{E}_2, \dots, \mathcal{E}_k$. Note that if $a = \min_{e_i \in \mathcal{E}} \sigma_i$ and $b = \max_{e_i \in \mathcal{E}} \sigma_i$ then k is at most $1 + \log_2(b/a)$.

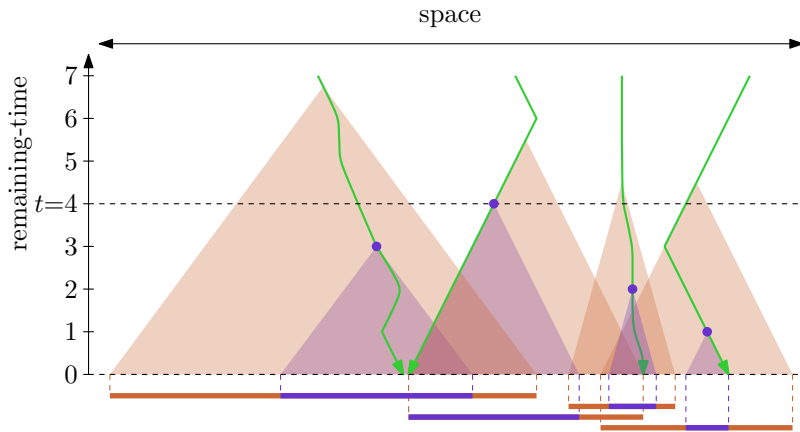


FIG. 1. The trajectories (green polylines) of a set of one-dimensional entities whose locations (the apexes of the large brown cones) were known precisely at times prior to remaining-time $t = 4$ when the query strategy starts, yielding the initial uncertainty regions (brown segments at the base of these cones) with ply 3. The entities have different maximum speeds (cone slopes). The strategy queries each entity once (obtaining the locations at the apexes of the smaller contained purple cones) yielding the final uncertainty regions (smaller contained purple segments) with ply 2.

1.1. Overview of results. In this paper we present a strategy to query a set of entities \mathcal{E} moving in \mathbb{R}^d to minimize the ply of their uncertainty regions at a target time that is t time units in the future. We compare the ply bound guaranteed by our strategy to the intrinsic ply Δ (the smallest ply achievable by any strategy, even one that knows the trajectories of all entities). To simplify the discussion, we start by describing our strategy restricted to entities moving in one dimension. We show, in section 3.1, that if the remaining-time t when the strategy starts is large enough, at least twice as large as the number of entities n , then our strategy can achieve ply $O(k\Delta)$, where \mathcal{E} can be partitioned into k speed classes. When our strategy starts with remaining-time less than $2n$, the bound on its ply depends on the quality of knowledge of the positions of the entities prior to making any queries. We describe this dependence in section 3.2. In section 3.3, we describe the final case when the time remaining is less than the number of entities, so some entities cannot be queried at all. In section 3.5, we generalize these results to entities moving in d -dimensional space. We show that our strategy achieves ply

$$\begin{aligned} & \Delta + O(\min\{t^{d/(d+1)}(k\Delta)^{1/(d+1)}, (\tilde{T}'/t)^{d^2/(d+1)}k\Delta\}) \quad \text{if } t < n, \\ & O(\min\{\alpha^d k\Delta, n^{d/(d+1)}(k\Delta)^{1/(d+1)}, (\tilde{T}/n)^{d^2/(d+1)}k\Delta\}) \quad \text{if } n \leq t < 2n, \\ & O(\min\{n, k\Delta\}) \quad \text{if } 2n \leq t, \end{aligned}$$

where Δ is the intrinsic ply of \mathcal{E} , $\alpha = t/(t - n)$, and \tilde{T} (or \tilde{T}') is the median of all (respectively, the largest t) times remaining when the entity locations were last known precisely. See Theorem 12. Surprisingly, the ply achieved by our strategy is tight in all of these cases; in section 4 we show that there exists a set of n entities in \mathbb{R}^d with median \tilde{T} of the times remaining when their locations were last known precisely, so that every query strategy has a competitive ratio at least a constant factor of our competitive ratio. In section 5, we show that computing the intrinsic ply of the entities exactly, which requires knowledge of the full trajectories, is NP-hard, even if those trajectories are constant.

2. Crowdedness and ply. At any point in time, a query strategy for a set of entities has available an uncertainty realization formed by the uncertainty regions associated with the most recent queries to those entities. It is natural, and as it turns out essential in all of our strategies, to ask to what extent the congestion evident in this current uncertainty realization constrains the intrinsic ply, i.e., any consistent final uncertainty realization, of the entity set.

We start by developing a tool that allows us to translate a natural measure of congestion into a lower bound on the intrinsic ply. A set of uncertainty regions is *h-crowded* if each region in the set contains a point that is also contained in at least $h - 1$ other regions. We also refer to the corresponding entities as being *h-crowded*.

We first describe results for entities that move in one dimension. This serves to illustrate the fundamental techniques in their simplest form, which we later generalize to higher dimensions.

When entities are drawn from a single speed class, there is a linear relationship between crowdedness and intrinsic ply that depends on the ratio between the average and maximum lifespans of the regions in the current uncertainty realization.

LEMMA 1. *Let \mathcal{E} be any set of entities from a single speed class, with an h -crowded uncertainty realization, each region of which has lifespan at most T . Then any uncertainty realization of \mathcal{E} , whose regions have maximum lifespan at most T and average lifespan \hat{T} , must have ply at least $\frac{h\hat{T}}{24T}$.*

Proof. Let $\mathcal{E} = \{e_1, e_2, \dots, e_m\}$ be a set of entities where σ_i , the maximum speed of e_i , lies between $s/2$ and s , and let $\{I_1, I_2, \dots, I_m\}$ be an h -crowded uncertainty realization of \mathcal{E} , with maximum lifespan T .

Let J_i be a region of radius $2sT$ centered at $f_i(0)$. Note that $I_i \subseteq J_i$, since the region I_i must contain $f_i(0)$ and has length at most $2\sigma_i T \leq 2sT$. Because $\{I_1, I_2, \dots, I_m\}$ is h -crowded, every uncertainty region I_j contains a point x_j that is contained in at least h distinct regions in $\{I_1, I_2, \dots, I_m\}$. It follows that $U = \{J_i \mid i = 1, \dots, m\}$ is also h -crowded.

Consider an independent set S of regions from the set U . See Figure 2 for an illustration. Since every region $J_i \in U$ intersects at least $h - 1$ other regions in U , the number of pairs $(A, B) \in S \times (U \setminus S)$ where A and B intersect is at least $(h - 1)|S|$. However, since every region $J_i \in U$ can touch at most two regions in S (in general, the kissing number of independent balls in dimension d [21]), this number of pairs is at most $2(m - |S|)$. Thus $|S| \leq 2m/(h + 1) < 2m/h$.

If S is chosen to be a maximal independent set in U , every region in $U \setminus S$ must intersect a region in S . Hence every region in U is contained in an extended region (ball) of radius $3 \cdot 2sT$ centered at the center of a region in S . The total length (volume) of all these extended regions is at most $|S|12sT$ (in general, $|S|$ times the volume of a sphere of radius $6sT$), which is less than $24msT/h$. It follows that

$$\left| \bigcup_{i=1}^m J_i \right| < 24msT/h.$$

Now let $\{I_1^*, I_2^*, \dots, I_m^*\}$ be any uncertainty realization for \mathcal{E} with ply Δ , whose regions have maximum lifespan at most T and average lifespan \hat{T} . In particular, I_i^* contains the terminal position $f_i(0)$ of entity e_i and has length $2\sigma_i t_i^* \leq 2st_i^*$, where $t_i^* \leq T$ is the lifespan of I_i^* , and hence I_i^* , like I_i , is a subset of J_i

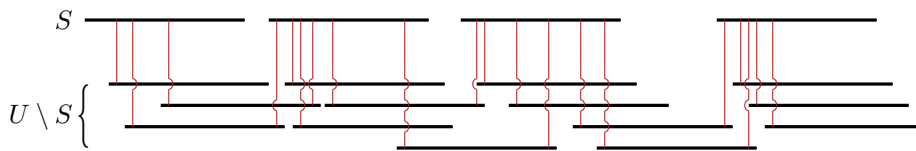


FIG. 2. The horizontal segments depict the set of regions $U = \{J_i | i = 1, \dots, m\}$ of radius $2sT$ centered at $f_i(0)$. The segments are separated vertically in the figure though the regions lie on a common line. The uppermost segments are a maximal independent set S of U (no two overlap). Every region intersects at least $h-1$ others ($h = 4$ in the figure), but only the intersections between regions in S and regions in $U \setminus S$ are shown as vertical lines. The number of these vertical lines is at least $(h-1)|S|$ (their S endpoints) and at most $2|U \setminus S| = 2(m - |S|)$ (their $U \setminus S$ endpoints).

Since, $(I_1^*, I_2^*, \dots, I_m^*)$ has ply at most Δ ,

$$\sum_{i=1}^m |I_i^*| \leq \Delta \left| \bigcup_{i=1}^m J_i \right| < 24msT\Delta/h.$$

We know that $|I_i^*| = 2\sigma_i t_i^* \geq st_i^*$ since $s/2$ is the smallest maximum speed of an entity in \mathcal{E} . Thus, $\hat{T} = \frac{1}{m} \sum_{i=1}^m t_i^* < 24T\Delta/h$ or equivalently

$$\Delta > \frac{h\hat{T}}{24T}.$$

□

When the entity set consists of entities from more than one speed class, crowdedness within individual speed classes provides a way of understanding how entities from different speed classes compete for a common resource (query slots). The next lemma considers the situation in which crowdedness is uniform across speed classes.

LEMMA 2. Let \mathcal{F} be a set of entities with intrinsic ply Δ and let $\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_k$ be a partition of \mathcal{F} into k speed classes. If \mathcal{F} has an uncertainty realization, all of whose regions have lifespan at most T , such that the regions associated with \mathcal{F}_j , for all $j = 1 \dots k$, are h -crowded, then $|\mathcal{F}| < 48T\Delta/h$.

Proof. Let $\mathcal{F} = \{e_1, e_2, \dots, e_n\}$ and let $\{I_1, I_2, \dots, I_n\}$ be an uncertainty realization of \mathcal{F} , with maximum lifespan T (note that $T \geq n$), such that the regions associated with \mathcal{F}_j , for all $j = 1 \dots k$, are h -crowded. Let $\{I_1^*, I_2^*, \dots, I_n^*\}$ be an uncertainty realization for \mathcal{F} with ply Δ . Without loss of generality we can assume that the regions $\{I_1^*, I_2^*, \dots, I_n^*\}$ have corresponding lifespans $\{t_1^*, t_2^*, \dots, t_n^*\}$ which are some permutation of $\{1, 2, \dots, n\}$.

Let \hat{T}_j denote the total lifespan of the uncertainty regions \mathcal{I}_i^* , associated with entities in \mathcal{F}_j . Then, by Lemma 1, it follows that $\Delta > \frac{h\hat{T}_j}{24|\mathcal{F}_j|T}$ for all $j = 1 \dots k$. Hence, $\sum_{i=1}^n t_i^* = \sum_{j=1}^k \hat{T}_j < \frac{24T\Delta}{h} \sum_{j=1}^k |\mathcal{F}_j| = \frac{24nT\Delta}{h}$, or

$$(1) \quad \frac{1}{n} \sum_{i=1}^n t_i^* < \frac{24T\Delta}{h}.$$

But, since $\{t_1^*, t_2^*, \dots, t_n^*\}$ are some permutation of $\{1, 2, \dots, n\}$, $\sum_{i=1}^n t_i^* = n(n+1)/2$, from which it follows immediately that

$$|\mathcal{F}| < 48T\Delta/h.$$

□

Finally we introduce a tool that allows us to address the case where each speed class, in a given entity set, is maximally crowded.

LEMMA 3. *Let \mathcal{F} be a set of entities with intrinsic ply Δ and let $\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_k$ be a partition of \mathcal{F} into k speed classes. If \mathcal{F} has an uncertainty realization, all of whose regions have lifespan at most T , such that the regions associated with \mathcal{F}_j , for all $j = 1 \dots k$, are $|\mathcal{F}_j|$ -crowded, then $|\mathcal{F}| < 10\sqrt{T\Delta k}$.*

Proof. Let $n_j = |\mathcal{F}_j|$. We may assume (by reindexing if necessary) that $n_1 \geq n_2 \geq \dots \geq n_k$. Thus, for all $j = 1 \dots k$, \mathcal{F}_i is n_j -crowded for any $i \leq j$, and by Lemma 2,

$$|\mathcal{F}_1 \cup \dots \cup \mathcal{F}_j| = \sum_{i=1}^j n_i < \frac{48T\Delta}{n_j}.$$

Multiplying both sides by n_j and summing over all j yields

$$\sum_{j=1}^k \left(n_j \sum_{i=1}^j n_i \right) < 48T\Delta k.$$

Since

$$(2) \quad \frac{1}{2} \left(\sum_{j=1}^k n_j \right) \left(\sum_{j=1}^k n_j \right) \leq \sum_{j=1}^k \left(n_j \sum_{i=1}^j n_i \right),$$

we have

$$\frac{1}{2} \left(\sum_{j=1}^k n_j \right) \left(\sum_{j=1}^k n_j \right) < 48T\Delta k,$$

and thus $\sum_{j=1}^k n_j < 10\sqrt{T\Delta k}$. \square

Remark. Lemmas 2 and 3 hold even if we replace intrinsic ply with a quantity that is never greater than intrinsic ply and that depends only on the final positions and speeds of the entities. For the uncertainty realization $\{I_1^*, I_2^*, \dots, I_n^*\}$ in the proof of Lemma 2, we only use the fact that I_i^* contains the final position $f_i(0)$ and has diameter $2\sigma_i t_i$. We don't require the interval to be centered at $f_i(t_i)$. Thus the proof holds if we define Δ to be the minimum ply of a collection $\mathcal{B} = \{B_1, B_2, \dots, B_n\}$ of balls defined by a sequence of distinct, positive integer query times t_1, t_2, \dots, t_n such that B_i has diameter $2\sigma_i t_i$ and contains $f_i(0)$. In all cases, our strategy produces ply that is competitive with this potentially smaller version of intrinsic ply.

3. Query strategy. For simplicity, we describe our query strategy for entities moving in one dimension and later generalize to higher dimensions. The strategy follows different approaches depending on (i) the time remaining until the target time and (ii) the initial uncertainty of the entities. Roughly, the strategy breaks down into three parts, as follows.

When there is plenty of time remaining until the target time ($t \geq 2n$), we query all entities once in the n time steps just before there is n time remaining. This reduces the lifespan of all uncertainty regions to at most $2n$. We then set aside the $n/2$ entities with the least crowded uncertainty regions and never query them again. Since all lifespans are small, Lemma 2 guarantees that the ply of the set aside entities

regions is $O(k)$ times the intrinsic ply. With the remaining $n/2$ entities (and in the remaining n time steps), we repeat this procedure until all entities are set aside.

If the time remaining when we start is between n and $2n$, we don't necessarily have time to query so that all uncertainty regions are small (at most $2n$). In this case, we set aside a large set of entities whose uncertainty regions overlap with each other very little. We then query the remaining entities in order to return to the previous case of plenty of time remaining.

Finally, if we have $t < n$ time remaining when the strategy starts, we set aside a set of $n - t$ entities whose regions have minimum ply (among themselves) and return to the previous case when the number of entities matches the time remaining.

These approaches are described and analyzed in detail in the following subsections.

Strategy 1. PLENTYOFTIME(\mathcal{E}, t).

Input. A set of n entities \mathcal{E} , and $t \geq 2n$ time remaining.

- 1: **if** $t > 0$ **then**
 - 2: Wait (query nothing) for $t - 2n$ time steps.
 - 3: Query all entities in \mathcal{E} once. \triangleright *Now there is n time remaining.* \triangleleft
 - 4: Choose the smallest h so that the set \mathcal{E}' of all h -crowded entities has size at most $n/2$. \triangleright *Set aside the non- h -crowded entities.* \triangleleft
 - 5: Call PLENTYOFTIME(\mathcal{E}', n).
-

3.1. A competitive strategy when $t \geq 2n$. We use Lemma 2 to show that when there is plenty of time remaining until the target time ($t \geq 2n$), one part of our overall strategy, called PLENTYOFTIME, is able to obtain ply that is within a factor $O(k)$ of the intrinsic ply, where k is the number of speed classes of \mathcal{E} . The idea of this strategy is to query all entities once and then, with remaining-time at least n , to identify and set aside half of the entities, which will never again be queried. This leaves $n/2$ entities with remaining-time at least n , so we can repeat this strategy until no entities remain. The key is to show that the ply of the regions of the entities that are set aside is not more than $O(k)$ times the intrinsic ply.

THEOREM 4. *Given a set \mathcal{E} of n entities, each belonging to one of k speed classes $\mathcal{E}_1, \mathcal{E}_2, \dots, \mathcal{E}_k$, and remaining-time $t \geq 2n$, PLENTYOFTIME achieves final ply $O(\min\{n, k\Delta\})$, where Δ is the intrinsic ply of \mathcal{E} .*

Proof. After the strategy queries every entity once, all entities have been queried with remaining-time at most $2n$. By Lemma 2, the maximum size set of h -crowded entities, for $h = c\Delta$ (where constant $c = 192$ suffices), is at most $n/2$. Thus the smallest h so that the strategy sets aside at least $n/2$ non- h -crowded entities is at most $c\Delta$. This implies that whenever an entity in speed class \mathcal{E}_j is set aside that has projected uncertainty region I , all points $p \in I$ are contained in fewer than $c\Delta$ of the projected uncertainty regions for entities in speed class \mathcal{E}_j . Since uncertainty regions for entities that are not set aside are contained in their current uncertainty regions, at the end of the strategy, when all entities are set aside, every point $x \in \mathbb{R}^1$ is covered by fewer than $c\Delta$ uncertainty regions for entities in speed class \mathcal{E}_j . Since this is true for all k speed classes, the strategy's final uncertainty regions have ply less than $ck\Delta$. Since there are only n entities, ply is at most n as well. \square

3.2. A competitive strategy when $n \leq t < 2n$. When $t < 2n$, we can no longer afford to use the previous strategy. In fact, if $t = n$, we only have time to query each entity once. Using Lemma 3, we show that this strategy, called QUERYALL,

can guarantee ply that is $O(\sqrt{nk\Delta})$ for a set of n entities in k speed classes with intrinsic ply Δ .

LEMMA 5. *Given a set \mathcal{E} of n entities, each belonging to one of k speed classes $\mathcal{E}_1, \mathcal{E}_2, \dots, \mathcal{E}_k$, and remaining-time $t = n$, QUERYALL achieves ply $O(\sqrt{nk\Delta})$, where Δ is the intrinsic ply of \mathcal{E} .*

Proof. Let p be the ply of the final uncertainty regions produced by QUERYALL, and let x be a point contained in p final uncertainty regions. Let \mathcal{F}_j be the set of entities from \mathcal{E}_j whose regions contain x . When QUERYALL ends, at the target time, all entities have been queried with remaining-time at most $t = n$; thus, by Lemma 3, $p = \sum_{j=1}^k |\mathcal{F}_j| \in O(\sqrt{nk\Delta})$. \square

The ply guaranteed by this lemma for QUERYALL seems terribly large. If $\Delta = 1$ and there is one speed class, QUERYALL has ply within a factor $O(\sqrt{n})$ of the intrinsic ply. That this is the best possible guarantee (as we will see in section 4) is some consolation, but the bound implicitly assumes that the initial state of uncertainty could be arbitrarily large. If this initial uncertainty is small relative to the current time remaining, we can do better. In fact, if, initially, the location of every entity is known with remaining-time $O(n)$, PLENTYOF TIME, starting from line 4, achieves ply $O(\min\{n, k\Delta\})$.

What if our initial knowledge of the entities' locations lies between these extremes? The main idea is to set aside a fixed size set of entities whose current uncertainty regions have minimum ply, among themselves. If the intrinsic ply of the input entities is small, these entities will have small ply. After querying the remaining entities, we gain enough information to set aside more entities (with low ply) so that we can apply strategy PLENTYOF TIME. If the intrinsic ply is large, querying the remaining entities ensures a ply that is comparable to that obtained by QUERYALL and comparable to the size of the fixed size set that we initially set aside.

THEOREM 6. *Given a set \mathcal{E} of n entities, each belonging to one of k speed classes $\mathcal{E}_1, \mathcal{E}_2, \dots, \mathcal{E}_k$, and remaining-time t with $n \leq t < 2n$, NOTIME TO LOSE! achieves ply $O(\min\{\alpha k\Delta, \sqrt{nk\Delta}, \sqrt{\tilde{T}/nk\Delta}\})$, where Δ is the intrinsic ply of \mathcal{E} , $\alpha = t/(t - n)$, and \tilde{T} is the median of the remaining-times when the entity locations were last known precisely.*

Proof. First, if $\tilde{T} \leq 16n$, then after making at most $n/2$ queries (line 2), the location of all entities are known with remaining-time at most $16n$. By Lemma 2, the smallest h so that the number of h -crowded entities is at most $n/4 \leq t'/2$ is $O(\Delta)$. Following the same argument as in Theorem 4, the non- h -crowded entities set aside in line 4, have ply at most $O(\min\{n, k\Delta\})$. Since the call to PLENTYOF TIME (line 5) guarantees ply $O(\min\{n, k\Delta\})$ for the h -crowded entities, the final ply is $O(\min\{n, k\Delta\}) \in O(\min\{\alpha k\Delta, \sqrt{nk\Delta}, \sqrt{\tilde{T}/nk\Delta}\})$, which is clear after recalling that $\alpha = t/(t - n) > 1$ and $\tilde{T} \geq t \geq n$.

The analysis of the case where $\tilde{T} > 16n$ depends on which term is the minimum of $\alpha k\Delta$, $\sqrt{nk\Delta}$, and $\sqrt{\tilde{T}/nk\Delta}$. Suppose $\alpha k\Delta$ is the minimum term. In this case, $\alpha \leq \sqrt{\tilde{T}/n}$ or, equivalently, $t/(t - n) \leq \sqrt{\tilde{T}/n}$, which implies (since $t \geq n$) that $n\sqrt{n/\tilde{T}} \leq t - n$. Thus \mathcal{J} remains empty and the strategy queries all entities in \mathcal{E} once (line 11) and has time $t' = t - n$ remaining. At this point in the strategy, all entities have been queried with remaining-time at most t , so by Lemma 2, choosing $H \in O(t\Delta/t')$ suffices so that at most $t'/2$ entities are H -crowded. The non- H -crowded regions set aside in line 13 cover a space that has final ply $O(t\Delta k/t') = O(\alpha k\Delta)$. After line 13, we have t' time and $t'/2$ entities remaining. Hence, we can use strategy

Strategy 2. NoTimeToLose! (\mathcal{E}, t).

Input. A set \mathcal{E} of n entities and remaining-time $t \in [n, 2n]$; and pairs $(t_i, f_i(t_i))$ describing the time and location when entity $e_i \in \mathcal{E}$ was last known precisely. Let \tilde{T} be the median of the t_i .

- 1: **if** $\tilde{T} \leq 16n$ **then**
- 2: Query the (at most) $n/2$ entities with $t_i > \tilde{T}$.
- 3: Let t' be the time remaining.
- 4: Choose the smallest h so that the set \mathcal{E}' of all h -crowded entities from \mathcal{E} has size at most $t'/2$. Set aside the other entities.
- 5: Call PLENTYOFTime(\mathcal{E}', t').
- 6: **else**
- 7: Let $\mathcal{J} = \emptyset$.
- 8: **if** $n\sqrt{n/\tilde{T}} > t - n$ **then**
- 9: Let \mathcal{J} be a minimum ply (within \mathcal{E}) subset of \mathcal{E} of size $n\sqrt{n/\tilde{T}}$.
 ▷ For dimension d , replace $n\sqrt{n/\tilde{T}}$ with $n(n/\tilde{T})^{d/(d+1)}$. ◁
- 10: Set aside the entities in \mathcal{J} .
- 11: Query each of the remaining $n - |\mathcal{J}|$ entities once, in any order.
- 12: Let t' be the time remaining.
- 13: Choose the smallest H so that the set \mathcal{E}' of all H -crowded entities from $\mathcal{E} \setminus \mathcal{J}$ has size at most $t'/2$. Set aside the other remaining entities.
- 14: Call PLENTYOFTime(\mathcal{E}', t').

PLENTYOFTime. The intrinsic ply of the remaining entities is at most Δ (since they form a subset of \mathcal{E}), so PLENTYOFTime yields a set of uncertainty regions with ply at most $O(k\Delta) \in O(\alpha k\Delta)$.

Otherwise, the main idea of NoTimeToLose! is to set aside, without querying, some number of entities so that we have fewer entities to handle in the remaining time. We set aside a set \mathcal{J} of $2x$ entities, where $x = \frac{n}{2}\sqrt{n/\tilde{T}}$ is chosen to minimize the ply of the strategy. Clearly, we want the uncertainty regions of entities in \mathcal{J} to have low ply since the strategy never queries these entities. Then the strategy queries the remaining $n - |\mathcal{J}|$ entities in any order, reducing their uncertainty regions. The strategy then sets aside, never to be queried again, all but $|\mathcal{J}|/2$ entities. Again, we want the entities set aside by the strategy to have low ply. Finally, with the $|\mathcal{J}|/2$ remaining entities and remaining-time $|\mathcal{J}|$, the strategy invokes PLENTYOFTime to perform its final queries.

Suppose $\sqrt{nk\Delta}$ is the minimum term. In this case, $\sqrt{nk\Delta} \leq \sqrt{\tilde{T}/nk}\Delta$ or, equivalently, $n \leq \sqrt{\tilde{T}\Delta}k$. The ply contributed by \mathcal{J} is at most its size, $n\sqrt{n/\tilde{T}}$, which by our supposition is at most $\sqrt{nk\Delta}$. After the strategy queries the remaining entities at line 11, the ply of these remaining entities (among themselves) is $O(\sqrt{nk\Delta})$. This follows from Lemma 5. We may imagine QUERYALL on input \mathcal{E} first querying the entities in $\mathcal{E} \setminus \mathcal{J}$, mimicking line 11 of NoTimeToLose!. Since QUERYALL on input \mathcal{E} achieves ply $O(\sqrt{nk\Delta})$ regardless of the query order, the ply of these entities (among themselves) is at most $O(\sqrt{nk\Delta})$. Thus the final ply is at most $\sqrt{nk\Delta} + O(\sqrt{nk\Delta}) = O(\sqrt{nk\Delta})$.

Finally, suppose $\sqrt{\tilde{T}/nk}\Delta$ is the minimum term. In this case, $\sqrt{\tilde{T}/nk}\Delta < \sqrt{nk\Delta}$. To understand the role of the median \tilde{T} in these calculations, it will help to imagine

T taking different values and let $n_T = |\mathcal{E}_T|$, where $\mathcal{E}_T = \{e_i \in \mathcal{E} | t_i \leq T\}$. Of course, $n_T = n/2$, for $T = \tilde{T}$, but it can be larger or smaller if T is, respectively, larger or smaller than the median.

We describe how to find a set \mathcal{J} of $2x$ entities from \mathcal{E}_T that has ply $O(xkT\Delta/n_T^2)$ for any T such that $n_T \geq 4x$. (Note that $n_{\tilde{T}} = n/2 \geq 4\frac{n}{2}\sqrt{n/\tilde{T}} = 4x$, since $\tilde{T} \geq 16n$.) Therefore the set \mathcal{J} that NOTIMELOSE! chooses also has ply $O(xkT\Delta/n_T^2)$. The idea is to add to \mathcal{J} a sequence of maximal “independent” sets of entities from the set \mathcal{E}_T , those whose uncertainty regions are independent of the regions of other entities in the same speed class. Each independent set is large so the number of independent sets we select for \mathcal{J} , whose total size is $2x$, is small and hence the ply of the uncertainty regions of entities in \mathcal{J} is small.

The procedure is as follows. Initially, let $\mathcal{J} = \emptyset$. Choose the smallest h so that the number of h -crowded entities in $\mathcal{E}_T \setminus \mathcal{J}$ is at most $n_T/4$. By Lemma 2, $h \in O(T\Delta/n_T)$ suffices. Let \mathcal{E}^* be the non- h -crowded entities in $\mathcal{E}_T \setminus \mathcal{J}$. Since $|\mathcal{E}_T \setminus \mathcal{J}| \geq n_T - |\mathcal{J}|$ and at most $n_T/4$ are h -crowded, \mathcal{E}^* contains at least $n_T - |\mathcal{J}| - n_T/4 \geq n_T/4$ entities (recall $|\mathcal{J}| = 2x \leq n_T/2$). Add to \mathcal{J} the entity in \mathcal{E}^* with the leftmost uncertainty region. That region intersects fewer than h other uncertainty regions of entities in \mathcal{E}^* within the same speed class. Remove these entities and the one with the leftmost uncertainty region from \mathcal{E}^* and repeat until \mathcal{E}^* is empty. At the end of this round, we have added to \mathcal{J} at least $\frac{n_T/4}{h}$ entities that have ply at most k , since the chosen entities within each speed class have independent regions. Repeat this process by choosing the set \mathcal{E}^* again, from $\mathcal{E}_T \setminus \mathcal{J}$. After at most $\frac{2x}{n_T/(4h)}$ rounds, we have a set \mathcal{J} of $2x$ entities. Since each round added at most ply k to \mathcal{J} , the total ply in \mathcal{J} is

$$(3) \quad O\left(\frac{xk}{n_T/h}\right) = O\left(\frac{xkT\Delta}{n_T^2}\right).$$

Note that line 10 sets aside entities whose ply must be *added* to the ply we obtain by querying the remaining entities. This is in contrast to the entities we set aside in line 13, whose current uncertainty regions cover a space that has ply less than H , no matter how the other entities are queried.

After line 11, all remaining entities have been queried with remaining-time at most n . By Lemma 2, with $H \in O(n\Delta/x)$, at most x entities are H -crowded.

The non- H -crowded regions set aside in line 13 cover a space that has final ply at most

$$(4) \quad O(n\Delta k/x)$$

no matter how the remaining entities are queried. After line 13, we still have $2x$ time and x entities remaining. Hence, we can use strategy PLENTYOF TIME. The intrinsic ply of the remaining entities is at most Δ (since they form a subset of \mathcal{E}), so PLENTYOF TIME yields a set of uncertainty regions with ply at most $O(k\Delta)$.

It remains to choose x to minimize the ply guaranteed by NOTIMELOSE!. We do this by balancing the ply contributed by \mathcal{J} (equation (3)) with the ply contributed by the non- H -crowded entities (equation (4)). The outcome is that x should be $n_T\sqrt{n/\tilde{T}}$, and the strategy's ply is $O(n\Delta k/x) = O(\sqrt{n\tilde{T}}\Delta k/n_T)$. Of course, to minimize its ply guarantee, the strategy could choose T to maximize x , i.e., to maximize $n_T/\sqrt{\tilde{T}}$. As written, NOTIMELOSE! chooses $T = \tilde{T}$, the median of t_1, t_2, \dots, t_n . This means the strategy's ply is $O(\sqrt{\tilde{T}/n}\Delta k)$. \square

3.3. A competitive strategy for all cases. We have described our strategy for the case when the remaining-time t is greater than twice the number of entities,

and when t lies between n and $2n$. In this section, we handle the remaining case when $t < n$.

When $t < n$, our strategy chooses a subset $\mathcal{X} \subset \mathcal{E}$ of t entities and invokes NOTIME TO LOSE! on \mathcal{X} . The strategy sets aside (and never queries) the remaining $n - t$ entities $\mathcal{E} \setminus \mathcal{X}$.

By Theorem 6, the strategy achieves ply

$$\text{ply}(\mathcal{E} \setminus \mathcal{X}) + O\left(\min\left\{\sqrt{tk_{\mathcal{X}}\Delta_{\mathcal{X}}}, \sqrt{\tilde{T}_{\mathcal{X}}/tk_{\mathcal{X}}\Delta_{\mathcal{X}}}\right\}\right),$$

where $\text{ply}(\mathcal{E} \setminus \mathcal{X})$ is the ply of the initial uncertainty regions of $\mathcal{E} \setminus \mathcal{X}$, $k_{\mathcal{X}}$ is the number of speed classes of \mathcal{X} , $\tilde{T}_{\mathcal{X}}$ is the median of the remaining-times when the entity locations for entities in \mathcal{X} were last known precisely, and $\Delta_{\mathcal{X}}$ is the intrinsic ply of \mathcal{X} . Notice that our choice of \mathcal{X} may affect several parameters in this bound. We should choose \mathcal{X} to minimize the bound, but we don't know $\Delta_{\mathcal{X}}$ and, for arbitrary \mathcal{X} , we don't know the relation of $\text{ply}(\mathcal{E} \setminus \mathcal{X})$ to the rest of the bound. However, since $\mathcal{X} \subset \mathcal{E}$, we know $\Delta_{\mathcal{X}} < \Delta$. Also, if we choose \mathcal{X} so that $\mathcal{E} \setminus \mathcal{X}$ is the lowest ply subset of $n - t$ entities from \mathcal{E} , then $\text{ply}(\mathcal{E} \setminus \mathcal{X}) \leq \Delta$ since every uncertainty realization of \mathcal{E} must include at least $n - t$ regions that coincide with initial uncertainty regions and their ply contributes to the final ply Δ . This choice of \mathcal{X} achieves final ply $\Delta + O(\min\{\sqrt{tk\Delta}, \sqrt{\tilde{T}'/tk\Delta}\})$, where \tilde{T}' is the median of the t largest times from t_1, t_2, \dots, t_n , since $\text{ply}(\mathcal{E} \setminus \mathcal{X}) \leq \Delta$, $k_{\mathcal{X}} \leq k$, $\Delta_{\mathcal{X}} \leq \Delta$, and $\tilde{T}_{\mathcal{X}} \leq \tilde{T}'$.

Combining this case with the previous cases, we obtain a combined strategy that has the following property.

THEOREM 7. *Given a set \mathcal{E} of n entities, each belonging to one of k speed classes $\mathcal{E}_1, \mathcal{E}_2, \dots, \mathcal{E}_k$, and remaining-time t with $n \leq t < 2n$, our combined strategy achieves ply*

$$\begin{aligned} &\Delta + O\left(\min\left\{\sqrt{tk\Delta}, \sqrt{\tilde{T}'/tk\Delta}\right\}\right) \quad \text{if } t < n, \\ &O\left(\min\left\{\alpha k\Delta, \sqrt{nk\Delta}, \sqrt{\tilde{T}/nk\Delta}\right\}\right) \quad \text{if } n \leq t < 2n, \\ &O(\min\{n, k\Delta\}) \quad \text{if } 2n \leq t, \end{aligned}$$

where Δ is the intrinsic ply of \mathcal{E} , $\alpha = t/(t - n)$, and \tilde{T} (or \tilde{T}') is the median of all (respectively, the largest t) remaining-times when the entity locations were last known precisely.

3.4. Computational efficiency. Our strategy performs some computation in order to choose which entities to query. Even with unbounded computation, strategies that obtain information only by querying cannot hope to perform as well as an optimal strategy that knows the entities' true trajectories (see section 4). Exactly how well such information-limited strategies compare to an all-knowing strategy is our primary concern; however, it is desirable that the computation performed by our strategy be efficient. In this section we show that in all but the extreme case when $t < n$, our strategy takes time polynomial in n (for constant dimension). Note that our intent is not to describe the most efficient implementation of our strategy, but only to demonstrate that it takes polynomial time in most cases.

The only step of PLENTY OF TIME that requires some computation is the choice of h in step 4. The uncertainty region for each entity is an interval of the real line

(or a sphere in dimension d) that shrinks to some contained interval when the entity is queried. Maximum ply occurs at vertices in the arrangement of the regions. All vertices and their ply can be found in $O(n^{d+1})$ time. The strategy can then determine those regions that contain a point within at least $h-1$ other regions of the same speed class, i.e., the set of h -crowded regions. A simple linear search can find the smallest h so the set of h -crowded entities has at most size $n/2$.

Strategy NOTIME TO LOSE! also chooses h -crowded sets in a similar fashion. In addition, it chooses a set \mathcal{J} of a particular size with minimum ply. This problem is NP-hard for dimension $d \geq 2$ since maximum independent set for unit disk graphs is NP-hard [26], and appears to be NP-hard even for dimension $d = 1$. Fortunately, as we outline in the proof of Theorem 6, we may use a greedy algorithm to find a set \mathcal{J} with ply that is small enough for our purposes in $O(n)$ rounds each taking time that is dominated by the time to find the h -crowded regions.

If $t < n$, our combined strategy starts by choosing a minimum ply subset, $\mathcal{E} \setminus \mathcal{X}$, of $n-t$ entities to set aside and never query. This is the same task as choosing the set \mathcal{J} in NOTIME TO LOSE!; however, we cannot guarantee that the same greedy algorithm will produce a set $\mathcal{E} \setminus \mathcal{X}$ with ply $O(\Delta)$ even if there is only one speed class. The problem of finding such a set, even for dimension $d = 1$, appears to be intractable.

3.5. Higher dimensions. The previous results extend to dimension $d > 1$. The proofs are essentially the same as in the one-dimensional case. The only challenge is to extend the h -crowded lemma (Lemma 2) and the all-overlapping lemma (Lemma 3) to dimension d , which we accomplish using Jensen's inequality.

Let $\nu_d = \frac{\pi^{d/2}}{\Gamma(d/2+1)}$ be the volume of a d -dimensional ball of radius 1. Let κ_d be the maximum number of independent d -dimensional unit spheres that can touch a given d -dimensional sphere.

LEMMA 8. *Let \mathcal{F} be a set of entities in dimension d with intrinsic ply Δ and let $\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_k$ be a partition of \mathcal{F} into k speed classes. If \mathcal{F} has an uncertainty realization, all of whose regions have lifespan at most T , such that the regions associated with \mathcal{F}_j , for all $j = 1 \dots k$, are h -crowded, then $|\mathcal{F}| < 12T(\nu_d \kappa_d \Delta / h)^{1/d}$.*

Proof. The proof follows the outline of the proof when $d = 1$ (Lemma 2) but replaces intervals with d -dimensional balls. We get as the d -dimensional generalization of (1),

$$\frac{1}{n} \sum_{i=1}^n (t_i^*)^d \leq \nu_d (6T)^d \kappa_d \Delta / h.$$

Jensen's inequality implies that

$$\left(\frac{1}{n} \sum_{i=1}^n t_i^* \right)^d \leq \frac{1}{n} \sum_{i=1}^n (t_i^*)^d.$$

Together, after taking d th roots, these imply that

$$\frac{1}{n} \sum_{i=1}^n t_i^* \leq 6T(\nu_d \kappa_d \Delta / h)^{1/d}.$$

The remainder of the proof follows Lemma 2 and we obtain

$$|\mathcal{F}| \leq 12T(\nu_d \kappa_d \Delta / h)^{1/d}. \quad \square$$

LEMMA 9. Let \mathcal{F} be a set of entities in dimension d with intrinsic ply Δ and let $\mathcal{F}_1, \mathcal{F}_2, \dots, \mathcal{F}_k$ be a partition of \mathcal{F} into k speed classes. If \mathcal{F} has an uncertainty realization, all of whose regions have lifespan at most T , such that the regions associated with \mathcal{F}_j , for all $j = 1 \dots k$, are $|\mathcal{F}_j|$ -crowded, then

$$|\mathcal{F}| < (24T)^{d/(d+1)} (\nu_d \kappa_d \Delta k)^{1/(d+1)}.$$

Proof. Let $n_j = |\mathcal{F}_j|$ and $S_j = \sum_{i=1}^j n_i$. Following the proof of Lemma 3 but using d -dimensional balls and Lemma 8 (the d -dimensional version of Lemma 2), we obtain

$$\sum_{j=1}^k n_j S_j^d \leq (12T)^d \nu_d \kappa_d \Delta k.$$

By Jensen's inequality,

$$\left(\frac{\sum_{j=1}^k n_j S_j}{\sum_{j=1}^k n_j} \right)^d \leq \frac{\sum_{j=1}^k n_j S_j^d}{\sum_{j=1}^k n_j}.$$

Combining the two preceding inequalities with (2) from the proof of Lemma 3, reproduced here,

$$\frac{1}{2} \left(\sum_{j=1}^k n_j \right) \left(\sum_{j=1}^k n_j \right) \leq \sum_{j=1}^k n_j S_j,$$

gives

$$(1/2)^d \left(\sum_{j=1}^k n_j \right)^{d+1} \leq (12T)^d \nu_d \kappa_d \Delta k.$$

The lemma follows by taking $(d+1)$ th roots. \square

LEMMA 10. Given a set \mathcal{E} of n entities in dimension d , each belonging to one of k speed classes $\mathcal{E}_1, \mathcal{E}_2, \dots, \mathcal{E}_k$, and remaining-time $t = n$, QUERYALL achieves ply $O(n^{d/(d+1)} (\Delta k)^{1/(d+1)})$, where Δ is the intrinsic ply of \mathcal{E} .

Proof. The proof is the same as for Lemma 5 but using Lemma 9 in place of Lemma 3. \square

THEOREM 11. Given a set \mathcal{E} of n entities in dimension d , each belonging to one of k speed classes $\mathcal{E}_1, \mathcal{E}_2, \dots, \mathcal{E}_k$, and remaining-time t with $n \leq t < 2n$, NOTIME-TOLOSE! achieves ply $O(\min\{\alpha^d k \Delta, n^{d/(d+1)} (k \Delta)^{1/(d+1)}, (\tilde{T}/n)^{d^2/(d+1)} k \Delta\})$, where Δ is the intrinsic ply of \mathcal{E} , $\alpha = t/(t-n)$, and \tilde{T} is the median of the remaining-times when the entity locations were last known precisely.

Proof. The proof follows the structure of the proof of Theorem 6 and uses the same algorithm except that we replace $n\sqrt{n/\tilde{T}}$ in line 9 with $n(n/\tilde{T})^{d/(d+1)}$.

If $\tilde{T} \leq 16n$, then, by Lemma 8, the non- h -crowded entities set aside in line 4 have ply at most $O(k\Delta)$. Since the call to PLENTYOF TIME (line 5) guarantees ply $O(k\Delta)$ for the h -crowded entities, the final ply is

$$O(k\Delta) \in O(\min\{\alpha^d k \Delta, n^{d/(d+1)} (k \Delta)^{1/(d+1)}, (\tilde{T}/n)^{d^2/(d+1)} k \Delta\}).$$

Again, we are really claiming that

$$O(\min\{n, k\Delta\}) \in O(\min\{\alpha^d k\Delta, n^{d/(d+1)}(k\Delta)^{1/(d+1)}, (\tilde{T}/n)^{d^2/(d+1)}k\Delta\}),$$

which is clear after recalling that $\alpha = t/(t-n) > 1$ and $\tilde{T} \geq t \geq n$.

The analysis of the case where $\tilde{T} > 16n$ depends on which term is the minimum of $\alpha^d k\Delta$, $n^{d/(d+1)}(k\Delta)^{1/(d+1)}$, and $(\tilde{T}/n)^{d^2/(d+1)}k\Delta$. Suppose $\alpha^d k\Delta$ is the minimum term. In this case, $\alpha^d \leq (\tilde{T}/n)^{d^2/(d+1)}$ or, equivalently, $t/(t-n) \leq (\tilde{T}/n)^{d/(d+1)}$, which implies (since $t \geq n$) that $n(n/\tilde{T})^{d/(d+1)} \leq t-n$. Thus \mathcal{J} remains empty and the strategy queries *all* entities in \mathcal{E} once (line 11) and has time $t' = t-n$ remaining. At this point in the strategy, all entities have been queried with remaining-time at most t , so by Lemma 8, choosing $H \in O(t^d \Delta / (t')^d)$ suffices so that at most $t'/2$ entities are H -crowded. The non- H -crowded regions set aside in line 13 cover a space that has final ply $O(t^d \Delta k / (t')^d) = O(\alpha^d k\Delta)$. After line 13, we have t' time and $t'/2$ entities remaining, so PLENTYOF TIME yields a set of uncertainty regions with ply at most $O(k\Delta) \in O(\alpha^d k\Delta)$.

Suppose $n^{d/(d+1)}(k\Delta)^{1/(d+1)}$ is the minimum term. In this case, $n^{d/(d+1)}(k\Delta)^{1/(d+1)} \leq (\tilde{T}/n)^{d^2/(d+1)}k\Delta$ or, equivalently, $n \leq \tilde{T}^{d/(d+1)}(k\Delta)^{1/(d+1)}$. The ply contributed by \mathcal{J} is at most its size, $n(n/\tilde{T})^{d/(d+1)}$, which by our supposition is at most $n^{d/(d+1)}(k\Delta)^{1/(d+1)}$. After the strategy queries the remaining entities at line 11, the ply of these remaining entities (among themselves) is $O(n^{d/(d+1)}(k\Delta)^{1/(d+1)})$. This follows from Lemma 10 as in the proof in dimension one. Thus the final ply is at most $n^{d/(d+1)}(k\Delta)^{1/(d+1)} + O(n^{d/(d+1)}(k\Delta)^{1/(d+1)}) = O(n^{d/(d+1)}(k\Delta)^{1/(d+1)})$.

Finally, suppose $(\tilde{T}/n)^{d^2/(d+1)}k\Delta$ is the minimum term. In this case, $(\tilde{T}/n)^{d^2/(d+1)}k\Delta < n^{d/(d+1)}(k\Delta)^{1/(d+1)}$. Let $x = n_T(n/T)^{d/(d+1)}$, where $n_T = |\mathcal{E}_T|$ and $\mathcal{E}_T = \{e_i \in \mathcal{E} | t_i \leq T\}$. We describe how to find a set \mathcal{J} of $2x$ entities from \mathcal{E}_T that has ply $O(xkT^d \Delta / n_T^{d+1})$. The set \mathcal{J} that NOTIME TO LOSE! chooses also has ply $O(xkT^d \Delta / n_T^{d+1})$.

We follow the same procedure as in dimension one. Initially, let $\mathcal{J} = \emptyset$. Choose the smallest h so that the number of h -crowded entities in $\mathcal{E}_T \setminus \mathcal{J}$ is at most $n_T/4$. By Lemma 8, $h \in O((T/n_T)^d \Delta)$ suffices. Add to \mathcal{J} a maximal independent set from each speed class in the non- h -crowded entities in $\mathcal{E}_T \setminus \mathcal{J}$. As in dimension one, we add at least $\frac{n_T/4}{h}$ entities that have ply at most k . Repeating this process for at most $\frac{2x}{n_T/(4h)}$ rounds, we obtain a set \mathcal{J} of $2x$ entities. Since each round added at most ply k to \mathcal{J} , the total ply in \mathcal{J} is

$$(5) \quad O\left(\frac{xk}{n_T/h}\right) = O\left(\frac{xkT^d \Delta}{n_T^{d+1}}\right).$$

After line 11, all remaining entities have been queried with remaining-time at most n . By Lemma 8, with $H \in O((n/x)^d \Delta)$, at most x entities are H -crowded.

The non- H -crowded regions set aside in line 13 cover a space that has final ply at most

$$(6) \quad O((n/x)^d k\Delta)$$

no matter how the remaining entities are queried. After line 13, we still have $2x$ time and x entities remaining, so PLENTYOF TIME yields a set of uncertainty regions with ply at most $O(k\Delta)$. Combining this with the ply contributions of the set aside entities ((5) and (6)) and letting $x = n_{\tilde{T}}(n/\tilde{T})^{d/(d+1)}$ gives the strategy's final ply of $O(n^{d/(d+1)}\tilde{T}^{d^2/(d+1)}\Delta k/n_{\tilde{T}}^d) = O((\tilde{T}/n)^{d^2/(d+1)}\Delta k)$. \square

THEOREM 12. Given a set \mathcal{E} of n entities in dimension d , each belonging to one of k speed classes $\mathcal{E}_1, \mathcal{E}_2, \dots, \mathcal{E}_k$, and remaining-time t , our combined strategy achieves ply

$$\begin{aligned} & \Delta + O\left(\min\{t^{d/(d+1)}(k\Delta)^{1/(d+1)}, (\tilde{T}'/t)^{d^2/(d+1)}k\Delta\}\right) \quad \text{if } t < n, \\ & O\left(\min\{\alpha^d k\Delta, n^{d/(d+1)}(k\Delta)^{1/(d+1)}, (\tilde{T}/n)^{d^2/(d+1)}k\Delta\}\right) \quad \text{if } n \leq t < 2n, \\ & O(\min\{n, k\Delta\}) \quad \text{if } 2n \leq t, \end{aligned}$$

where Δ is the intrinsic ply of \mathcal{E} , $\alpha = t/(t-n)$, and \tilde{T} (or \tilde{T}') is the median of all (respectively, the largest t) remaining-times when the entity locations were last known precisely.

4. A lower bound on the competitive ratio. We now prove that the competitive ratios obtained by our strategies are within a constant factor of the optimum. We describe a set of n entities, \mathcal{E} , with remaining-time t , where $n \leq t$ such that any query strategy can be forced, by relabeling entities with identical uncertainty regions, to achieve ply $\Omega(\zeta^{d/(d+1)}k\Delta)$, where

$$\zeta = \min\{\alpha^{d+1}/(16b)^d, 2n/(k\Delta), (T/n)^d/2\},$$

T is the remaining-time when the entity locations were last known precisely, $\alpha = t/(t-n)$, k is the number of speed classes, and $b = 4\sqrt{d}$ is a constant. We will only construct sets of entities when these parameters result in $\zeta \geq 1$. We assume that n is a multiple of $k\zeta\Delta$.

Let $\Delta' = \Delta/2$. We construct a collection of $z = n/(k\zeta\Delta')$ sets $\mathcal{A}_1, \dots, \mathcal{A}_z$ (since $k\zeta\Delta' \leq k(2n/(k\Delta))\Delta' = n$, there is at least one), each consisting of $k\zeta\Delta'$ entities. Each set \mathcal{A}_i has k speed classes, $\mathcal{A}_{i,j}$ (for $j = 1, \dots, k$), each with $\zeta\Delta'$ entities. The speed of entities in $\mathcal{A}_{i,j}$ is 2^{j-1} .

Let \mathbf{o}_i be the origin for the entities in \mathcal{A}_i . The z origins are spaced more than $T2^k$ apart so no uncertainty region from entities in \mathcal{A}_i intersects one from entities in \mathcal{A}_j for $i \neq j$. All entities in $\mathcal{A}_{i,j}$ have last known position $\mathbf{o}_i + T2^{j-1}\mathbf{e}_1$ with T time remaining, where \mathbf{e}_1 is a unit basis vector in dimension d . Each set $\mathcal{A}_{i,j}$ has $m = \Delta'(\zeta/b)^{d/(d+1)}$ special entities that move toward origin \mathbf{o}_i at their maximum speed until remaining-time $n/2$. Thus, a query of a special entity in \mathcal{A}_i before remaining-time $n/2$ produces an uncertainty region that contains the origin \mathbf{o}_i .

With $n/2$ time remaining, the special entities move at maximum speed, in groups of size Δ' , to m/Δ' positions that are separated by distance $(mkz)2^j$ from each other and stop. Any query after remaining-time mkz to two special entities at different stationary positions will result in disjoint uncertainty regions. Furthermore, the stationary positions are chosen so that these uncertainty regions lie within the half of the d -dimensional ball of radius $(n/2)2^{j-1}$ and center $\mathbf{o}_i + (n/2)2^{j-1}\mathbf{e}_1$ that is farthest from the origin. Being in this d -dimensional ball means that the entities can reach their stationary position with mkz time remaining. Being in the half of this d -dimensional ball that is farthest from the origin means that the uncertainty regions of special entities in $\mathcal{A}_{i,j}$ are disjoint from those of special entities in $\mathcal{A}_{i,j'}$ for $j \neq j'$, provided the special entities are queried with remaining-time at most mkz . This is so because the d -dimensional ball of radius $(n/2)2^{j-1}$ and center $\mathbf{o}_i + (n/2)2^{j-1}\mathbf{e}_1$ lies entirely inside the half of the d -dimensional ball of radius $(n/2)2^{j'-1}$ and center $\mathbf{o}_i + (n/2)2^{j'-1}\mathbf{e}_1$ that is closest to the origin for $j < j'$.

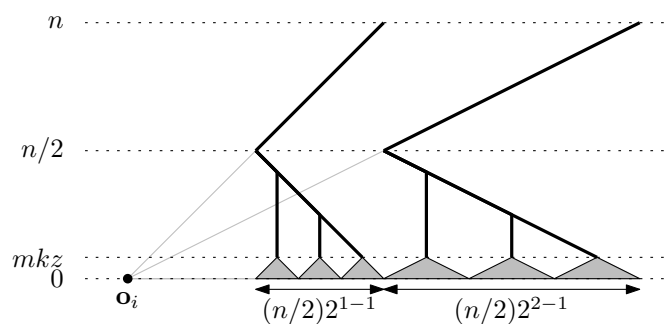


FIG. 3. The trajectories of the special entities in speed classes $\mathcal{A}_{i,1}$ and $\mathcal{A}_{i,2}$, shown with remaining-time between n and mkz for dimension $d = 1$ and $m/\Delta' = 3$.

So that the m/Δ' uncertainty regions for the groups of special entities in $\mathcal{A}_{i,j}$ lie within the appropriate half-ball, the volume of the uncertainty regions must be less than the volume of the half-ball. This is not quite sufficient since the packing of disjoint uncertainty regions (d -dimensional balls of radius $(mkz)2^{j-1}$) into a half-ball of radius $(n/2)2^{j-1}$ does not fill the half-ball. What is sufficient is if the volume of m/Δ' , d -dimensional uncertainty region bounding boxes (with side lengths $2(mkz)2^{j-1}$) is at most half the volume of the inscribed d -dimensional box with side length $2(n/2)2^{j-1}/\sqrt{d}$. That is,

$$(m/\Delta')(2(mkz)2^{j-1})^d \leq (1/2)(n2^{j-1}/\sqrt{d})^d.$$

This is equivalent to

$$\begin{aligned} (m/\Delta')(2(mkz))^d &\leq (1/2)(n/\sqrt{d})^d, \\ (\Delta'(\zeta/b)^{d/(d+1)}/\Delta')(2n/(\zeta b^d)^{1/(d+1)})^d &\leq (1/2)(n/\sqrt{d})^d, \\ (\zeta/b)^{1/(d+1)}2n/(\zeta b^d)^{1/(d+1)} &\leq (1/2)^{1/d}n/\sqrt{d}, \\ 2/b &\leq (1/2)^{1/d}/\sqrt{d}, \\ b &\geq 2^{1+1/d}\sqrt{d}, \end{aligned}$$

which is true for $b = 4\sqrt{d}$.

At most Δ' special entities stop at the same position. Since the stationary positions are separated by $(mkz)2^j$, we achieve ply at most Δ' among the special entities by querying them with remaining-time at most mkz , in any order. This is possible since the number of special entities is mkz . See Figure 3.

The $\zeta\Delta' - m$ nonspecial entities in speed class $\mathcal{A}_{i,j}$ also have last known position $\mathbf{o}_i + T2^{j-1}\mathbf{e}_1$ with T time remaining. From that position, the nonspecial entities move at maximum speed, in groups of size Δ' , to $(\zeta\Delta' - m)/\Delta'$ positions that are separated by distance $n2^j$ from each other and stop. Any query, after remaining-time n , to two nonspecial entities at different stationary positions will result in disjoint uncertainty regions. Furthermore, the stationary positions are chosen so that these uncertainty regions lie within the half of the d -dimensional ball of radius $T2^{j-1}$ and center $\mathbf{o}_i + T2^{j-1}\mathbf{e}_1$ that is farthest from the origin. Being in this d -dimensional ball means that the entities can reach their stationary position with n time remaining. Being in the half of this d -dimensional ball that is farthest from the origin means that the uncertainty regions of nonspecial entities in $\mathcal{A}_{i,j}$ are disjoint from those of

nonspecial entities in $\mathcal{A}_{i,j'}$ for $j \neq j'$, provided the nonspecial entities are queried with remaining-time at most n . Again, this is so because the d -dimensional ball of radius $T2^{j-1}$ and center $\mathbf{o}_i + T2^{j-1}\mathbf{e}_1$ lies entirely inside the half of the d -dimensional ball of radius $T2^{j'-1}$ and center $\mathbf{o}_i + T2^{j'-1}\mathbf{e}_1$ that is closest to the origin for $j < j'$.

So that the $(\zeta\Delta' - m)/\Delta'$ uncertainty regions for the groups of nonspecial entities in $\mathcal{A}_{i,j}$ lie within the appropriate half-ball, it suffices that the volume of $\zeta > (\zeta\Delta' - m)/\Delta'$, d -dimensional uncertainty region bounding boxes (with side lengths $2n2^{j-1}$) is at most half the volume of the inscribed d -dimensional box with side length $2T2^{j-1}/\sqrt{d}$. That is,

$$\zeta(2n2^{j-1})^d \leq (1/2)(2T2^{j-1}/\sqrt{d})^d,$$

which is true since $\zeta \leq (1/2)(T/n)^d$.

Since the stationary positions are separated by $n2^j$, we achieve ply at most Δ' among the nonspecial entities by querying them with remaining-time between n and $n - mkz$, in any order. Together with the special entities, the total ply we achieve is at most $2\Delta' = \Delta$.

Note that we achieve ply Δ only by assuming we know at the start which entities are special. Query strategies, which only know the uncertainty regions and speed limits of the entities, do not have this information and must rely on their queries to reveal which entities are special.

Let $\mathcal{E}_j = \bigcup_{i=1}^z \mathcal{A}_{i,j}$ and $\mathcal{E} = \bigcup_{j=1}^k \mathcal{E}_j$.

LEMMA 13. *Let Q be any query strategy, and let p be the maximum ply, over relabeling of entities with identical speed limits and uncertainty regions, achieved by executing Q on \mathcal{E} . Then $p \in \Omega(\zeta^{d/(d+1)}k\Delta)$.*

Proof. Since the uncertainty regions associated with entities in set $\mathcal{A}_{i,j}$ are initially identical, we can force, by suitable relabeling of entities, the first m queries made by any query strategy Q (that doesn't know the trajectories of the entities) to distinct entities in $\mathcal{A}_{i,j}$ to be to special entities. These early queries to special entities are not productive. A query of an entity in set \mathcal{A}_i is *productive* if it reduces the uncertainty region of the entity so that the region no longer contains \mathbf{o}_i . Let $\Delta' = \Delta/2$. There are $\zeta\Delta'$ entities in set $\mathcal{A}_{i,j}$ of which m are special, so at most $\zeta\Delta' - m$ queries to entities in $\mathcal{A}_{i,j}$ made before $n/2$ time remaining are productive. This means at most a fraction $\frac{\zeta\Delta' - m}{\zeta\Delta'} = 1 - \frac{1}{\zeta^{1/(d+1)}b^{d/(d+1)}}$ of all $t - n/2$ queries before $n/2$ time remaining are productive, recalling that $m = \Delta'(\zeta/b)^{d/(d+1)}$. Let γ be this fraction. Any query made with $n/2$ time remaining or less may be productive but there are only $n/2$ of these. So for any query strategy, the average ply measured at the z origins is at least

$$\frac{n - \gamma(t - n/2) - n/2}{z},$$

which, after replacing z and t , equals

$$\begin{aligned} & \frac{n - \gamma(n + n/\alpha - n/2) - n/2}{\frac{n}{k\zeta\Delta'}} \\ &= (1/2 - \gamma(1/2 + 1/\alpha))k\zeta\Delta' \\ &= \zeta \left(\frac{1-\gamma}{2} - \frac{\gamma}{\alpha} \right) k\Delta' \geq \zeta \left(\frac{1-\gamma}{2} - \frac{1}{\alpha} \right) k\Delta'. \end{aligned}$$

Since $1 - \gamma = \zeta^{-1/(d+1)}b^{-d/(d+1)}$ and $\alpha \geq 4\zeta^{1/(d+1)}b^{d/(d+1)}$, the average ply, and thus the ply of at least one set \mathcal{A}_i , is at least $\Omega(\zeta^{d/(d+1)}k\Delta') = \Omega(\zeta^{d/(d+1)}k\Delta)$. \square



FIG. 4. (a) An example input for GRACEFUL SEGMENT COVER. (b) A possible solution. (The lengths of the intervals and gaps are scaled by a factor 2 to avoid fractions; e.g., the blue interval labeled “4” has length 8, 4 on each side of the point.)

We use this lemma to show the following.

THEOREM 14. *There exists a set \mathcal{E} of n entities in dimension d , each belonging to one of k speed classes $\mathcal{E}_1, \mathcal{E}_2, \dots, \mathcal{E}_k$, and remaining-time t , such that any query strategy can be forced, by relabeling of entities with identical speed limits and initial uncertainty regions, to achieve ply*

$$\begin{aligned} \Delta + \Omega(\min\{t^{d/(d+1)}(k\Delta)^{1/(d+1)}, (T'/t)^{d^2/(d+1)}k\Delta\}) & \text{ if } t < n, \\ \Omega(\min\{\alpha^d k\Delta, n^{d/(d+1)}(k\Delta)^{1/(d+1)}, (T/n)^{d^2/(d+1)}k\Delta\}) & \text{ if } n \leq t < 2n, \\ \Omega(\min\{n, k\Delta\}) & \text{ if } 2n \leq t, \end{aligned}$$

where $\Delta \geq 2$ is the intrinsic ply of \mathcal{E} , $\alpha = t/(t - n)$, and T (or T') is the median of all (respectively, the largest t) remaining-times when the entity locations were last known precisely.

Proof. If $t < n$, we use the same construction as used for Lemma 13 with $n = t$ entities; the other $n - t$ entities play no role in the lower bound.

If $n \leq t < 2n$, the lower bound follows from Lemma 13.

If $t \geq 2n$, we first consider the case when $k\Delta \leq 2n$. In this case, $2n/(k\Delta) \geq 1$. Also $(T/n)^{d/2} \geq (t/n)^{d/2} \geq 2^{d-1}$, and $\alpha = t/(t - n) \geq 1$. Thus, $\zeta = \min\{\alpha^{d+1}, 2n/(k\Delta), (T/n)^{d/2}\}$ is at least a constant, and Lemma 13 implies the $\Omega(k\Delta)$ bound. If $k\Delta > 2n$, we use the same construction as used for Lemma 13, but we only use $2n/\Delta < k$ different speed classes to obtain the $\Omega(n)$ bound. \square

5. Hardness of computing the intrinsic ply. Recall that the intrinsic ply of a collection of entities at a fixed target time is a function of the full trajectories of the entities, the knowledge of which is not available to a query strategy. Nevertheless, as we have shown in the previous sections, our strategy is, up to constant factors, minimally competitive with the intrinsic ply.

In this section, we show that it is NP-hard to compute the intrinsic ply of a collection of entities at a fixed target time, even if the entities are points set in one dimension whose location does not change over time, a situation in which our strategy is guaranteed to be $O(1)$ -competitive.

We first show that the following, simpler, problem is NP-hard. We refer to this problem as GRACEFUL SEGMENT COVER.

PROBLEM 15. *Given a set X of n numbers (points in \mathbb{R}^1), compute n disjoint intervals of lengths 1 to n , such that the midpoint of each interval is a point in X .*

Figure 4 shows an instance of this problem. We now prove the following.

THEOREM 16. *The decision version of GRACEFUL SEGMENT COVER is NP-complete.*

Proof. We give a reduction from CNF-SAT. That is, given a CNF formula ϕ , we construct an instance of GRACEFUL SEGMENT COVER that has a solution if and only

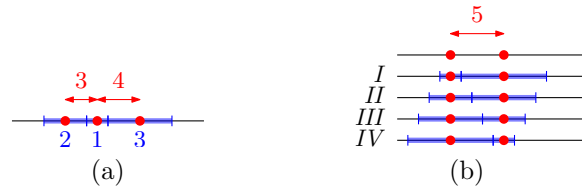


FIG. 5. (a) A wall. (b) Half of a jump gadget.

if ϕ is satisfiable. To avoid fractions, we build a GRACEFUL SEGMENT COVER that is scaled by a factor of 2, that is, the intervals have radii $1, \dots, n$, and diameters/lengths $2, 4, 6, \dots, 2n$. We refer to the intervals by their radius, so interval i is the interval of radius i .

The main idea in the reduction is to make gadgets that force any solution to use the 3 or 4 smallest intervals left available. That is, the first gadget we build consists of, say, 3 points and can be satisfied only by mapping the intervals 1, 2, and 3 to those three points, in some permutation. Then the next gadget might consist of 4 points and can be satisfied only by using intervals 4 to 7. In general, we let S_i be the total number of intervals used by the first i gadgets, and then the $i + 1$ th gadget will use intervals $S_i + 1$, $S_i + 2$, $S_i + 3$, and possibly $S_i + 4$. Since the S_i part will be present in all points of the $i + 1$ th gadget, we can conceptually ignore it, and we will build all gadgets using only intervals of radii 1, 2, 3, and possibly 4.

Basic gadgets. The first gadget we need is the *wall gadget*. It consists of three consecutive points, with gaps of size 3 and 4. The only way to satisfy this gadget is by mapping interval 2 to the first point, interval 1 to the second point, and interval 3 to the third point. Figure 5(a) illustrates this. The wall gadget can be satisfied in only one way, therefore, it creates a fixed region in \mathbb{R} that is filled, and needs to be avoided by all other gadgets.

The second gadget we need is the *jump gadget*. It consists of two pairs of consecutive points. In both pairs (p, q) the distance (gap) between p and q is 5, but the pairs themselves can be arbitrary far away from each other. There are several ways to satisfy this gadget, but since $1 + 2 + 3 + 4$ equals the sum of the gaps (two times 5), any solution must use intervals 1 to 4, and furthermore, intervals 1 and 4 must be used together at one of the pairs and 2 and 3 at the other pair. Figure 5(b) illustrates this. Note that the intervals used in one pair together use 10 space.

Composite gadgets. Let ϕ be a CNF-SAT instance, consisting of V , a set of variables; C , a set of clauses; and $E \subset V \times C \times \{\text{true}, \text{false}\}$, a set of occurrences of variables in clauses, possibly negated. We also call E the edges of the instance. We will create a jump gadget for each element of E , and one more for each variable of V . We construct variable and clause gadgets on consecutive segments of \mathbb{R} , which are separated by wall gadgets. Hence, we use $|V| + |E|$ jump gadgets and $|V| + |C| + 1$ wall gadgets in total.

Let $v \in V$ be a variable of the SAT instance, and suppose there are k edges in E that use v . We start by making a gap of length $10k + 21$ between two walls for the variable gadget. Inside, we place a special jump gadget; one pair at distance 2 from the left wall and one pair at distance 2 from the right wall. Then for each $(v, c_i, b_i) \in E$, we place one pair of the jump gadget (the other pair will be in the clause gadget) at distance $10i + 2$ from the left wall if $b = \text{true}$ or at distance $10i + 4$ if $b = \text{false}$. Figure 6(a) illustrates the construction.

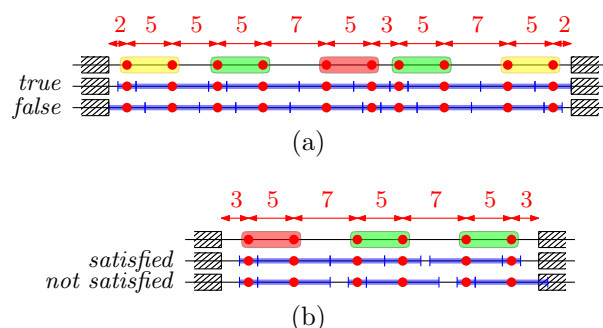


FIG. 6. (a) A variable. The first and last (yellow) jumpers are linked to each other to ensure there are only two possible states. Green jumpers correspond to positive literals of the variable and are linked to clauses where the variable appears positively. Red jumpers correspond to negative literals and link to clauses where the variable appears negatively. (b) A clause.

There are $k + 2$ half jump gadgets in the construction, each of which will take up 10 space. The total space is $10k + 21$, which leaves one unit empty. The outer two jump gadgets that are linked to each other ensure that we cannot build all the way to the left and right walls simultaneously; therefore, there are only two possible states that satisfy the variable gadget: either the leftmost unit is empty or the rightmost unit is empty. We use these to encode the states *true* and *false* of the variable. Once this choice is made, the way to fill in all the jump gadgets is fixed. Note that for each edge (v, c, b) , we use intervals 1 and 4 if the variable is set to true and $b = \text{true}$ or if the variable is set to false and $b = \text{false}$, and intervals 2 and 3 otherwise.

Let $c \in C$ be a clause of the SAT instance, and suppose there are k edges in E that use c . We make a gap of length $12k - 1$ between two walls for the clause gadget. Inside, we place the k half jump gadgets, each gadget i at distance $12i - 9$ from the left wall. Figure 6(b) illustrates the construction. Recall that each jump gadget in a variable uses intervals 1 and 4 if its sign matches the state of the variable, and 2 and 3 otherwise. This means that the other halves of the gadget, in the clause, must use intervals 2 and 3 if the literal is true and 1 and 4 if it is false. If all literals in the clause are not satisfied, then there is no solution, since somewhere two intervals 4 would need to overlap. If at least one of the literals is true, then there is enough space for any assignment of the other literals.

Finally, to complete the proof, we need to consider the actual lengths again, including the S_i factors that we ignored so far. Recall we used $2|V| + |C| + |E| + 1$ gadgets. We order and number them arbitrarily. Now, for any gap in the construction between two points belonging to gadgets i and j , we increase the gap size by $S_i + S_j$. \square

Now, we can prove hardness of the original problem.

THEOREM 17. *Computing the intrinsic ply of a set \mathcal{E} of n entities at time $t^* = t^0 + t$ for $t \geq n$ is NP-hard.*

Proof. We reduce from GRACEFUL SEGMENT COVER. Consider an input set X of n points in \mathbb{R} . Now simply place an entity on position x for every input number x in X , and keep the entities at their starting position. An optimal solution must query the entities in some order and will result in a set of intervals of lengths 1 to n . Since the entities are stationary, these intervals will be centered at the corresponding

points. To test whether a solution of ply 1 is possible it is necessary to decide if there exists a placement of disjoint intervals of lengths 1 to n , centered at the points in X . The theorem follows. \square

6. Concluding remarks. We studied a new model for dealing with unpredictable motion in which obtaining the location of a moving entity costs a certain amount of time. We developed strategies to keep track of the positions of moving entities in this model by minimizing the ply of their uncertainty regions at a fixed time t time steps in the future. We described strategies that obtain asymptotically tight competitive ratios with respect to an optimal strategy that knows the trajectories of the entities; though we show that such a strategy would need to solve an NP-hard problem to produce the optimal sequence of queries from the trajectories.

We can also use our techniques to address the more challenging *recurrent* version of ply minimization in which the goal is to minimize the ply at regular checkpoints, spaced t units apart. As before, we will address this in a competitive framework, by trying to minimize, at every checkpoint, the competitive ratio of the ply attained by our strategy with the intrinsic ply of the trajectories at that point. Since the intrinsic ply corresponds to the minimum ply achievable at one single point in time by a query strategy that has complete knowledge of the trajectories, we are in effect contrasting the results achievable by our strategy with those achievable by a family of fully informed strategies designed to optimize the ply at each checkpoint separately.

As it turns out a simple modification of our single target strategy works remarkably well: it is $O(1)$ -competitive, when the checkpoint spacing is any constant fraction of the number of entities. To be precise, suppose that we have n entities and we want to minimize the ply of their associated uncertainty regions at regular checkpoints, spaced t time units apart. Our strategy interleaves two processes, ROUNDROBIN and SINGLETARGET, running in odd and even time steps, respectively.

ROUNDROBIN: Query all entities in a round-robin fashion.

SINGLETARGET: After each checkpoint, follow our single target ply minimization strategy (section 3.3) with the goal of minimizing the ply at the next checkpoint.

Since ROUNDROBIN queries every entity once every $2n$ time steps, we know that, after at most $2n$ time steps (or right from the start, if the initial uncertainty regions are of diameter at most $2n$), the diameter of all uncertainty regions, projected to the next checkpoint, never exceeds $2n + t$. Furthermore, since SINGLETARGET executes on even time steps, its behavior is indistinguishable from that of our single target strategy when entities are moving at two times their normal speed. Thus our results in Theorem 12 still hold in this case, with $t/2$ replacing t , since SINGLETARGET executes only every other time step, and with \tilde{T} (and \tilde{T}') upper bounded by $2n + t$. In particular,

THEOREM 18. *If the checkpoint spacing t is at least n/c for positive constant c , interleaved ROUNDROBIN–SINGLETARGET guarantees ply $O((4c + 1)^{d^2/(d+1)} k \Delta)$ at all checkpoints.*

Interesting directions for future work include extending this work to deal with uncertainty in time (e.g., what if we want to minimize the ply of the uncertainty regions at an unknown time in $[t^-, t^+]$), and to handle varying query time costs. In particular, what happens if the time cost of a query depends on the running time of the querying strategy? Another possible extension would be to allow multiple queries to be executed simultaneously.

REFERENCES

- [1] P. K. AGARWAL, J. ERICKSON, AND L. J. GUIBAS, *Kinetic BSPs for intersecting segments and disjoint triangles*, in Proceedings of the 9th ACM-SIAM Symposium Discrete Algorithms, 1998, pp. 107–116.
- [2] J. ALMEIDA AND R. ARAUJO, *Tracking multiple moving objects in a dynamic environment for autonomous navigation*, in Proceedings of the 10th IEEE International Workshop on Advanced Motion Control, 2008, pp. 21–26.
- [3] J. BASCH, L. J. GUIBAS, C. SILVERSTEIN, AND L. ZHANG, *A practical evaluation of kinetic data structures*, in Proceedings of the 13th Annual Symposium on Computational Geometry, ACM, 1997, pp. 388–390.
- [4] R. BRUCE, M. HOFFMANN, D. KRIZANC, AND R. RAMAN, *Efficient update strategies for geometric computing with uncertainty*, Theory Comput. Syst., 38 (2005), pp. 411–423.
- [5] K. BUCHIN, M. LÖFFLER, P. MORIN, AND W. MULZER, *Delaunay triangulation of imprecise points simplified and extended*, Algorithmica, 61 (2011), pp. 674–693.
- [6] M. CHO, D. M. MOUNT, AND E. PARK, *Maintaining nets and net trees under incremental motion*, in Proceedings of the 20th International Symposium on Algorithms and Computation, Springer, New York, 2009, pp. 1134–1143.
- [7] M. DE BERG, M. ROELOFFZEN, AND B. SPECKMANN, *Kinetic convex hulls and Delaunay triangulations in the black-box model*, in Proceedings of the 27th Annual Symposium on Computational Geometry, ACM, (2011), pp. 244–253.
- [8] S. B. EISENMAN, *People-Centric Mobile Sensing Networks*, Ph.D. thesis, Columbia University, New York, 2008.
- [9] D. EPPSTEIN, M. T. GOODRICH, AND M. LÖFFLER, *Tracking moving objects with few handovers*, in Proceedings of the 12th Algorithms and Data Structures Symposium, 2011, pp. 362–373.
- [10] W. EVANS, D. KIRKPATRICK, M. LÖFFLER, AND F. STAALS, *Competitive query strategies for minimising the ply of the potential locations of moving points*, in Proceedings of the 29th Annual Symposium on Computational Geometry, SoCG '13, ACM, New York, 2013, pp. 155–164.
- [11] W. EVANS, D. KIRKPATRICK, M. LÖFFLER, AND F. STAALS, *Query strategies for minimizing the ply of the potential locations of entities moving with different speeds*, in Abstracts of the 30th European Workshop on Computational Geometry (EuroCG), 2014.
- [12] P. G. FRANCIOSA, C. GAIBISSO, G. GAMBOSI, AND M. TALAMO, *A convex hull algorithm for points with approximately known positions*, Internat. J. Comput. Geom. Appl., 4 (1994), pp. 153–163.
- [13] J. GAO, L. GUIBAS, AND A. NGUYEN, *Deformable spanners and their applications*, Comput. Geom., 35 (2006), pp. 2–19.
- [14] L. GUIBAS, J. HERSHBERGER, S. SURI, AND L. ZHANG, *Kinetic connectivity for unit disks*, in Proceedings of the 16th Annual Symposium on Computational Geometry, ACM, 2000, pp. 331–340.
- [15] L. J. GUIBAS, *Kinetic data structures—a state of the art report*, in Proceedings of the Workshop Algorithmic Foundations of Robotics, A. K. Peters, Wellesley, MA, 1998, pp. 191–209.
- [16] M. HOFFMANN, T. ERLEBACH, D. KRIZANC, M. MIHALÁK, AND R. RAMAN, *Computing minimum spanning trees with uncertainty*, in Proceedings of the STACS, 2008, pp. 277–288.
- [17] S. H. KAHAN, *Real-Time Processing of Moving Data*, Ph.D. thesis, University of Washington, 1991.
- [18] M. LÖFFLER AND M. VAN KREVELD, *Largest and smallest convex hulls for imprecise points*, Algorithmica, 56 (2010), pp. 235–269.
- [19] G. MILLER, S. TENG, W. THURSTON, AND S. VAVASIS, *Separators for sphere-packings and nearest neighbor graphs*, J. ACM, 44 (1997), pp. 1–29.
- [20] D. M. MOUNT, N. S. NETANYAHU, C. D. PIATKO, R. SILVERMAN, AND A. Y. WU, *A computational framework for incremental motion*, in Proceedings of the 20th Annual Symposium on Computational Geometry, ACM, 2004, pp. 200–209.
- [21] F. PFENDER AND G. ZIEGLER, *Kissing numbers, sphere packings, and some unexpected proofs*, Notices Amer. Math. Soc., 51 (2004).
- [22] M. SCHNEIDER, *Moving objects in databases and GIS: State-of-the-art and open problems*, in Research Trends in Geographic Information Science, Lecture Notes in Geoinform. Sci., Springer, New York, 2009, pp. 169–187.
- [23] K. SREENATH, F. L. LEWIS, AND D. O. POPA, *Simultaneous adaptive localization of a wireless sensor network*, ACM, SIGMOBILE Mob. Comput. Commun. Rev., 11 (2007), pp. 14–28.
- [24] K.-C. R. TSENG AND D. G. KIRKPATRICK, *Input-thrifty extrema testing*, in Proceedings of ISAAC, 2011, pp. 554–563.

- [25] J. VAN DEN BERG AND M. OVERMARS, *Planning time-minimal safe paths amidst unpredictably moving obstacles*, Internat. J. Robotics Res., 27 (2008), pp. 1274–1294.
- [26] D. WANG AND Y.-S. KUO, *A study on two geometric location problems*, Inform. Process. Lett., 28 (1988), pp. 281–286.
- [27] K. YI AND Q. ZHANG, *Multi-dimensional online tracking*, in Proceedings of the 20th ACM-SIAM Symposium on Discrete Algorithms, 2009, pp. 1098–1107.
- [28] C. ZHU, L. SHU, T. HARA, L. WANG, AND S. NISHIO, *Research issues on mobile sensor networks*, in Proceedings of the 5th International ICST Conference on Communications and Networking in China (CHINACOM), 2010, pp. 1–6.