Query Strategies for Minimizing the Ply of the Potential Locations of Entities Moving with Different Speeds

William Evans* David Kirkpatrick* Maarten Löffler† Frank Staals†

Abstract

In a recent paper [2] we gave query strategies for minimizing the ply of the potential locations of a set of moving entities. We extend our results to the case where the entities have different maximum speeds.

We present a query strategy that can achieve ply $O(k\Delta)$ when we have at least $2n$ time, the *intrinsic ply* of the entities is Δ, and the speeds of the entities fall into k *speed classes*. Furthermore, we show that for every query strategy there is a set of n entities for which the strategy achieves ply $\Omega(k\Delta)$. Hence, our results are optimal up to a constant factor.

We present our query strategy for entities moving in $\mathbb{R}^1$. However, our results extend to higher dimensions.

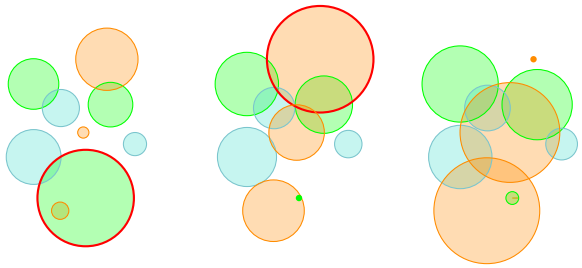

Fig. 1: Illustrating the model at three subsequent time steps, for entities of three different speeds. In each step, one disk (red border) is queried, resulting in a point at the next step. All other disks grow in diameter proportional to their speed.

1 Introduction

Movement is everywhere. Whether it is people driving to work in their car, robots assembling a product, or birds flying around in search of food, everywhere we look we see moving objects. So, it is no surprise that researchers in a wide range of application fields study movement and moving data [6]. Often, the movement of the objects involved is unpredictable, and the data must be processed in real-time. Recently, there have been several efforts from the computational geometry community to formalize unrestricted motion [1, 3, 5].

Acquiring the (exact) location of an entity in real time often has a certain cost. For example the remaining battery power of the tracking device, or the time it takes to obtain the location. Therefore, it is often still impossible to know the exact location of all entities involved at any time. Instead, the exact location of an entity may only be available at discrete times. In between two such times, the entity can be anywhere in a region of potential locations, typically a ball. The size of this *uncertainty region* increases as time progresses. See Fig. 1 for an illustration.

A natural question is now how to keep track of a set of moving entities if we can query the location of one entity at any time, and each query takes unit time. That is, can we maintain a representation of the uncertainty regions of the entities that distinguishes them as well as possible.

In a previous paper [2] we took a first step in answering this question. We presented query strategies (algorithms) that try to minimize the *ply* of the set of uncertainty regions at a given time t^* in the future, assuming that all entities have a given maximum speed s. The ply [4] of the uncertainty regions at a time t is the maximum number of uncertainty regions (at time t) that contain any given point in $\mathbb{R}^d$. If the entities move too close together it may not be possible to assure a good ply with any querying strategy. So we used a form of *competitive analysis*: we compared the ply achieved by our strategies to the ply achievable by an optimal strategy that knows the trajectories of the entities in advance, and showed that our strategies were $O(1)$-competitive to such an informed strategy.

In this paper, we extend our work to the case where the entities have different maximum speeds. More specifically, we present an $O(k)$-competitive query strategy for when we can group the entities with similar (maximum) speeds into k *speed classes* (in which maximum speeds differ by at most a constant factor). Furthermore, we show that this bound is optimal up to a constant factor; that is, there is no query strategy that achieves competitive ratio $o(k)$.

Problem Statement. Let $\mathcal{E} = \{e_1, e_2, \ldots, e_n\}$ be a set of n point entities moving unpredictably in $\mathbb{R}^1$. An entity e_i belongs to *speed class* s if its maximum speed, v_i, lies in the range $[s/2, s]$. Let $s_1, .., s_k$ be a set of speed classes covering all entities in $\mathcal{E}$. We wish

*Department of Computer Science, University of British Columbia, Canada, {will, kirk}@cs.ubc.ca

†Department of Information and Computing Sciences, Utrecht University, {m.loffler, f.staals}@uu.nl

to obtain information about the potential locations of the entities in $\mathcal{E}$ so that we can distinguish them as well as possible at some future time t^*.

We consider *query strategies* that can obtain the current location of an entity by querying it, but can only query one entity per unit time. Knowing the current location of an entity restricts its future positions because it has bounded speed. Let f_i be the (unknown) *trajectory* of entity e_i, which is a continuous function from time ($\mathbb{R}$) to position ($\mathbb{R}^1$).

A query strategy's t^*-*uncertainty interval* with time τ remaining for an entity e_i is the interval centered at $f_i(t_i)$ with length $2v_i(t^* - t_i)$, where $t_i < t^* - \tau$ is the last time the strategy queries e_i with at least τ time remaining. We will omit t^*, the time τ remaining, and the entity e_i when these are implied by context. When the time remaining is $\tau = 0$, we will call this the *final uncertainty interval*.

We measure how well a strategy distinguishes entities by the *(final) ply* of its final uncertainty intervals. The *ply* of a set of intervals is the maximum number of intervals in the set that contain a common point.

For some entities, it may be impossible for a query strategy to achieve low final ply. We say that the collection $\mathcal{I} = \{I_1, I_2, \ldots I_n\}$ of intervals forms a t^*-*uncertainty realisation* of $\mathcal{E}$ if there exists a sequence of distinct integer query times $t_1, t_2, \ldots t_n$, all less than t^*, such that I_i has center $f_i(t_i)$ and length $2v_i(t^* - t_i)$. The minimum, over all t^*-uncertainty realisations $\mathcal{I}$ of $\mathcal{E}$, of the ply of I is the t^*-*intrinsic ply* of $\mathcal{E}$. The intrinsic ply of $\mathcal{E}$ is a lower bound on the final ply that *any* query strategy could achieve, even one that has full knowledge of the trajectories of the entities involved. We will call an optimal informed strategy that achieves final ply equal to the intrinsic ply OPT.

2 A $O(k)$-Competitive Query Strategy

In this section we describe a query strategy that with $\tau \geq 2n$ time remaining and a set $\mathcal{E}$ of n entities, each belonging to one of k different speed classes $s_1, s_2, \ldots, s_k$, achieves final ply at most some fixed constant c times $k\Delta$, where Δ is the intrinsic ply of $\mathcal{E}$. Like our strategy PlentyOfTime [2], which worked for the case $k = 1$, the new strategy PlentyOfTimeWithSpeeds proceeds in rounds, each of which begins by querying all entities once in arbitrary order, and then sets aside half of the entities (never to be queried again). The key questions are how to decide which entities to set aside, and how to argue that the final ply of all uncertainty intervals does not exceed $ck\Delta$. Since our strategy eventually sets aside all entities, this means we have to argue that the ply of all uncertainty intervals that we set aside does not exceed $ck\Delta$.

Our previous strategy PlentyOfTime [2] set aside entities whose uncertainty intervals did not intersect h-*heavy windows*: regions containing at least h uncertainty intervals (for an appropriate value of h). However, it seems that this argument is no longer applicable, since the size of the windows now also depends on the speed of the entities.[1] Therefore, we use a more fine-grained approach. In each round, our strategy will set aside the set of entities that is not h-*crowded* (for an appropriate value of h). An entity of speed class s is h-crowded (with remaining time τ), if its current uncertainty interval, that is, the uncertainty interval when the remaining time is τ, contains a point that is also contained in at least $h - 1$ other uncertainty intervals of entities in speed class s.

To reason that the ply of the uncertainty intervals that we set aside is not too large, we introduce the concept of *delay units*. Given a set $\mathcal{E}$ of entities, the *delay units with time τ remaining* assigned to entity $e_i \in \mathcal{E}$ is $\lceil \lg \frac{2\tau}{x_i} \rceil$ where x_i is the time remaining (before t^*) when entity e_i was last queried by OPT. If OPT does not query e_i in the last 2τ time steps, we set $x_i = 2\tau$. The number of delay units assigned to entity e_i measures the difference in scale between the length of the uncertainty interval with time 2τ remaining for e_i (assuming that the exact position of e_i at this time is known), and the size of OPT's final uncertainty interval for e_i. It is easy to argue that the total number of delay units, over all entities, is bounded:

Lemma 1 *With time τ remaining, the total number of delay units over all entities is at most 3τ.*

Proof. Since OPT can query only once per time step, the total number of delay units is at most $\sum_{i=1}^{\tau} \lceil \lg \frac{2\tau}{i} \rceil$ which, bounding the sum by the corresponding integral, is at most 3τ. □

The following lemma, which is the key to our approach, asserts that if some point p is contained in more than $c\Delta$ uncertainty intervals for entities from any one speed class s then those ($c\Delta$-crowded) entities must be assigned a total of at least $\lg c - 2$ delay units on average.

Lemma 2 *In any set of entities with intrinsic ply Δ, the average number of delay units assigned to each h-crowded entity with time τ remaining is at least $\lg \frac{h}{4\Delta}$.*

Proof. Assume, by renumbering if necessary, that $e_1, e_2, \ldots, e_m$ are the h-crowded entities in speed class s. Let I_i be the uncertainty interval for e_i and let I_i^* be OPT's final uncertainty interval for e_i. We will show that OPT, in order to achieve ply Δ, must assign at least $\lg \frac{h}{4\Delta}$ delay units on average to each of these h-crowded entities.

[1] When speeds are the same, we can avoid this problem by scaling [2].

Let $u_i = \lceil \lg \frac{2\tau}{x_i} \rceil$ be the delay units assigned to entity e_i, where x_i is the time remaining (before t^*) when entity e_i was last queried by OPT. If OPT does not query e_i in the last 2τ time steps, we set $x_i = 2\tau$. In effect, this permits OPT to query multiple entities at time $t^* - 2\tau$ without using any delay units. It also means that we may assume that the intervals I_i^* are contained in intervals of length $2\tau \cdot 2s$.

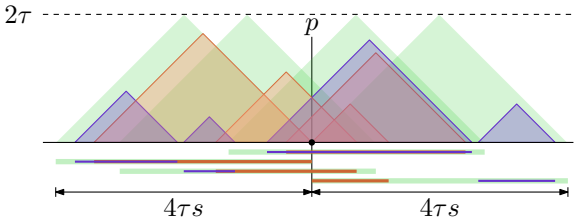

Fig. 2: The uncertainty intervals (brown) covering p, and their corresponding final uncertainty intervals in OPT (purple) are clustered around p.

Every uncertainty interval I_i, with $i \le m$, contains a point p that is contained in at least h distinct uncertainty intervals. Each of these intervals is contained in an interval of length $2\tau \cdot 2s$ that also contains OPT's corresponding final uncertainty interval. Thus OPT's final uncertainty intervals are clustered within distance $4\tau s$ of such points p, in groups of size at least h (see Fig. 2). The union of all of OPT's final uncertainty intervals thus covers at most 1-dimensional volume $8\tau s(m/h)$, since there are at most m/h groups and each group lies within an interval of length $8\tau s$ centered at such a point p. The intrinsic ply is Δ, so

$$\sum_{i=1}^{m} |I_i^*| \le 8\tau s(m/h)\Delta. \tag{1}$$

We know $|I_i^*| \ge x_i s$ since $s/2$ is the smallest maximum speed of an entity in speed class s. Furthermore, by the definition of u_i, $x_i \ge \frac{2\tau}{2^{u_i}}$. Thus, from inequality (1), we have

$$\sum_{i=1}^{m} \frac{2\tau}{2^{u_i}} s \le 8\tau(m/h)\Delta.$$

By the arithmetic-geometric mean inequality,

$$\left(\prod_{i=1}^{m} 2^{-u_i} \right)^{1/m} \le \frac{4\Delta}{h},$$

which implies $\frac{1}{m}\sum_{i=1}^{m} u_i \ge \lg \frac{h}{4\Delta}$. □

We use Lemma 2 to show:

Theorem 3 *Given a set $\mathcal{E}$ of n entities, each belonging to one of k speed classes $s_1, s_2, \ldots, s_k$, and time $\tau \ge 2n$ remaining,* PlentyOfTimeWithSpeeds *achieves final ply at most $2^8 k\Delta$ where Δ is the intrinsic ply of $\mathcal{E}$.*

Algorithm PlentyOfTimeWithSpeeds$(\mathcal{E}, \tau)$
Input. A set of n entities $\mathcal{E}$, and the remaining time τ, with $\tau \ge 2n$.

1. **if** $\tau > 0$ **then**
2. Wait (query nothing) for $\tau - 2n$ time steps.
3. Query all entities in $\mathcal{E}$ once.
   ▷ *Now there is time n remaining.* ◁
4. Choose the smallest h so that the set $\mathcal{E}'$ of h-crowded entities has size at most $n/2$.
   ▷ *Set aside the non-h-crowded entities.* ◁
5. Call PlentyOfTimeWithSpeeds$(\mathcal{E}', n)$.

Proof. For $h = 2^8\Delta$, the number of h-crowded entities with time $\tau = n$ remaining is at most $n/2$; otherwise, by Lemma 2, the total delay units assigned to h-crowded entities is more than $\frac{\tau}{2} \lg \frac{2^8\Delta}{4\Delta} = 3\tau$, which, by Lemma 1, exceeds the total number of delay units available.

Thus the entities the strategy sets aside are not h-crowded for $h = 2^8\Delta$. This implies that whenever an entity in speed class s_i is set aside that has current (i.e., with τ time remaining) uncertainty interval I, all points $p \in I$ are contained in less than $2^8\Delta$ current uncertainty intervals for entities in speed class s_i. The strategy's final uncertainty intervals for entities that are not set aside are contained in their current uncertainty intervals. Thus at the end of the strategy, when all entities are set aside, every point $p \in \mathbb{R}^1$ is covered by at most $2^8\Delta$ uncertainty intervals for entities in speed class s_i. Since this is true for all k speed classes, the strategy's final uncertainty intervals have ply at most $k2^8\Delta$. □

3 A Lower bound

Next, we describe a construction involving a collection of $2k$ entities moving in $\mathbb{R}^1$. The entities belong to k different speed classes: two entities per class. The construction is designed to force ply at least $k/4-1$ for any strategy that knows only the initial uncertainty intervals and maximum speeds of the various entities. An informed strategy, one that knows the actual trajectories followed by the various entities, is capable of choosing query times for the various entities in such a way that the final ply is one.

Note that the $2k$ entities can form part of a larger collection of $n > 2k$ entities by simply adding $n - 2k$ entities whose initial uncertainty intervals are all disjoint from those of all other entities, and hence require no further queries. Furthermore, a modest generalization of the construction involves $2k\Delta$ entities, and forces ply $\Omega(k\Delta)$, where ply Δ is realizable by an informed strategy.

For each i, $1 \le i \le k$, the pair of entities in speed class s_i have maximum speed of $s_i = (3k)^{i-1}$. The construction fixes the position of both entities up to

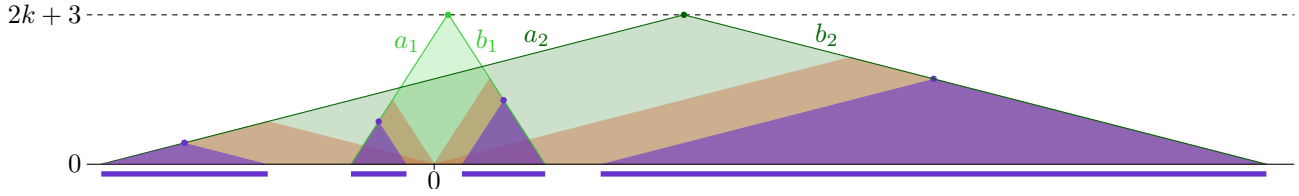

Fig. 3: The lower-bound construction for $k = 2$. The vertical axis represents time. The purple intervals correspond to the ideal query times. For all entities it holds that if we were to query them even one time unit before their ideal query times (the brown intervals), then their uncertainty intervals contain the origin.

time $2k+3$ remaining (at times $t \leq t^* - (2k+3)$) at the same point, $(2i-1)(3k)^{i-1}$, on the real number line. Thereafter both move at their maximum permissible speed in opposite directions. The only information that is denied to the uninformed strategy is which of the two entities (copy a) moves left and which (copy b) moves right. See Fig. 3 for an illustration.

The informed strategy queries copy a of the entities in speed class s_i at time $k-i+1$ remaining and copy b at time $k+i$ remaining. We refer to these times as the *ideal* query times. The resulting final uncertainty intervals are $[-2(k-i+2)(3k)^{i-1}, -2(3k)^{i-1}]$, for copy a, and $[2(3k)^{i-1}, 2(k+i+1)(3k)^{i-1}]$ for copy b. It is straightforward to confirm that these intervals (from all speed classes) are disjoint. Note, however, that if the query to any entity takes place even one time unit earlier than these ideal query times then the resulting uncertainty interval must include the origin (see the brown intervals in Fig. 3). In fact, we will argue that any uninformed strategy can be forced, by making a suitable choice for the identity of copy a and b within each speed class, to have the last query to at least $k/4-1$ of the entities take place earlier than its ideal query time. As a consequence, the resulting ply at the origin must be at least $k/4-1$.

In speed class s_i, in order to ensure that copy a is queried at or after its ideal query time it must be queried with time less than $k-i \leq k$ remaining. However, an adversary is free to make copy b be the first of the pair to be queried. Hence, for each speed class where both entities are queried at or after their ideal time, either (i) both entities are queried with time less than k remaining, or (ii) both entities are queried between time $2k+2$ and k remaining and one is queried with time less than k remaining. Let x and y be the number of speed classes that satisfy condition (i) and (ii), respectively. Since there are k query times with time less than k remaining, $2x + y \leq k$. Since there are $k+3$ query times between time $2k+2$ and k remaining, $2y \leq k+3$. Thus $x + y \leq 3(k+1)/4$. It follows then that the entity pairs from at least $k/4-1$ speed classes satisfy neither condition. Thus the final uncertainty intervals of at least $k/4-1$ entities must intersect the origin. We conclude:

Theorem 4 *For every query strategy, there exist k speed classes $s_1, s_2, \ldots, s_k$ and a set of $n \geq 2k$ entities from those speed classes with intrinsic ply Δ, such that the strategy can only achieve final ply $\Omega(k\Delta)$.*

4 Concluding Remarks

We briefly discuss some further consequences of our results. Our query strategy makes no assumptions about the relative values of the s_i's. So when the maximum speeds lie in the range $[1, s]$, our strategy guarantees a final ply that is $O(\Delta \log s)$. Similarly, we obtain a lower bound of $\Omega(\frac{\log s}{\log\log s})$.

Our results extend to entities moving in $\mathbb{R}^d$, with $d > 1$. Each h-crowded entity is then assigned at least $\frac{1}{d}\lg\frac{h}{4\Delta}$ delay-units. Choosing $h = 2^{9d-1}\Delta$ guarantees that our query strategy achieves ply at most $k2^{9d-1}\Delta$.

Acknowledgments. This work has been partially supported by the Netherlands Organisation for Scientific Research (NWO) under grants 639.021.123 and 612.001.022, and by NSERC of Canada.

References

[1] M. de Berg, M. Roeloffzen, and B. Speckmann. Kinetic convex hulls and Delaunay triangulations in the black-box model. In *Proc. 27th ACM Symp. on Comput. Geom.*, pages 244–253, 2011.

[2] W. Evans, D. Kirkpatrick, M. Löffler, and F. Staals. Competitive query strategies for minimising the ply of the potential locations of moving points. In *Proc. Symp. on Comp. Geom.*, pages 155–164. ACM, 2013.

[3] J. Gao, L. Guibas, and A. Nguyen. Deformable spanners and their applications. *Computational Geometry: Theory and Applications*, 35:2–19, 2006.

[4] G. Miller, S. Teng, W. Thurston, and S. Vavasis. Separators for sphere-packings and nearest neighbor graphs. *Journal of the ACM*, 44(1):1–29, 1992.

[5] D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. A computational framework for incremental motion. In *Proc. 20th ACM Symp. on Comput. Geom.*, pages 200–209, 2004.

[6] M. Schneider. Moving Objects in Databases and GIS: State-of-the-Art and Open Problems. *Research Trends in Geographic Information Science*, pages 169–187, 2009.