ARTICLE IN PRESS

Theoretical Computer Science ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

The directed Hausdorff distance between imprecise point sets

Christian Knauer [a], Maarten Löffler [b], Marc Scherfenberg [a,*], Thomas Wolle [c]

[a] Institut für Informatik, Universität Bayreuth, Bayreuth, Germany
[b] Computer Science Department, University of California, Irvine, USA
[c] NICTA Sydney, Sydney, Australia

ARTICLE INFO

Keywords:
Hausdorff distance
Shape matching
Impreciseness

ABSTRACT

We consider the directed Hausdorff distance between point sets in the plane, where one or both point sets consist of imprecise points. An imprecise point is modelled by a disc given by its centre and a radius. The actual position of an imprecise point may be anywhere within its disc. Due to the direction of the Hausdorff distance and whether its tight upper or lower bound is computed, there are several cases to consider. For every case we either show that the computation is NP-hard or we present an algorithm with a polynomial running time. Further we give several approximation algorithms for the hard cases and show that one of them cannot be approximated better than with factor 3, unless P = NP.

1. Introduction

The analysis and comparison of geometric shapes are essential tasks in various application areas within computer science, such as pattern recognition and computer vision. Beyond these fields also other disciplines evaluate the shape of objects such as cartography, molecular biology, medicine, or biometric signal processing. In many cases patterns and shapes are modelled as finite sets of points.

The *Hausdorff* distance is an important tool to measure the similarity between two sets of points (or, more generally, any two subsets of a metric space). It is defined as the largest distance from any point in one of the sets, to the closest point in the other set (see Section 2 for a formal definition). This distance is used extensively in pattern matching.

Data imprecision is a phenomenon that has existed as long as data is being collected. In practice, data is often sensed from the real world, and as a result has a certain error region. On the one hand, many application fields of computational geometry use algorithms that take this into account. However, these algorithms are mostly heuristics, and do not benefit from theoretical guarantees. On the other hand, algorithms from computational geometry are provably correct and efficient, often under the assumption that the input data is correct. If we want these algorithms to be used in practice, they need to take imprecision into account in the analysis. Thus not surprisingly, data imprecision in computational geometry is receiving more and more attention.

In this paper, we study several variants of the important and elementary problem of computing the Hausdorff distance under the Euclidean metric between *imprecise* point sets.

1.1. Related work

The Hausdorff distance is one of the most studied similarity measures. For a survey about similarity measures and shape matching refer to [3]. A straightforward, naive algorithm computes the Hausdorff distance between two point sets A and B

* Corresponding author. Tel.: +49 3083875159.
E-mail addresses: christian.knauer@uni-bayreuth.de (C. Knauer), mloffler@uci.edu (M. Löffler), marc.scherfenberg@uni-bayreuth.de (M. Scherfenberg), thomas.wolle@nicta.com.au (T. Wolle).

0304-3975/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2011.01.039

Table 1
P and Q are point sets and $\tilde{P}$ and $\tilde{Q}$ are imprecise point sets. Results are shown for the case when all sets have $O(n)$ elements. * can be computed exactly in $O(n^3)$ time if the discs are disjoint and the answer is smaller than $r(\sqrt{5-2\sqrt{3}}-1)/2$ where r is the radius of the smallest disc in $\tilde{Q}$.

Setting		Tight lower bound	Tight upper bound
$h(\tilde{P}, Q)$	[general]	$O(n \log n)$	$O(n \log n)$
$h(P, \tilde{Q})$	[general]	NP-hard*, 4-APX in $O(n^3 \log^3 n)$	$O(n \log n)$
	[disjoint unit discs]	3-APX-hard, 3-APX in $O(n^{10} \log n)$	$O(n \log n)$
$h(\tilde{P}, \tilde{Q})$	[general]	NP-hard	$O(n^2)$
	[const. depth in $\tilde{P}$]		$O(n \log n)$

consisting of m and n points, respectively, in $O(mn)$ time. Using Voronoi diagrams and a more sophisticated approach the running time can be reduced to $O((m+n)\log n)$ [2].

The study of imprecision within computational geometry started around twenty years ago, when Guibas et al. [7] introduced *epsilon geometry* as a way to handle computational imprecision. In this model, each point is assumed to be at most ε away from its given location.

For a given measure on a set of imprecise points, one of the simplest questions to ask in this model is what are the possible output values? Each input point can be anywhere in a given region, and depending on where each point is, the output will have a different value. This leads to the problem of placing the points in their regions such that this value is minimised or maximised. One of the first results of this kind is due to Goodrich and Snoeyink [6], who show how to place a set of points on a set of vertical line segments such that the points are in convex position and the area or perimeter of the convex hull is minimised in $O(n^2)$ time. A similar problem is studied by Mukhopadhyay et al. [12], and their result was later generalised to isothetic line segments [11].

Nagai and Tokura [13] thoroughly study the computation of lower and upper bounds for a variety of region shapes and measures; in particular they study the diameter, the width, and the convex hull, and all their algorithms run in $O(n \log n)$ time. However, not all of their bounds are tight. Van Kreveld and Löffler [10] study the same problems and give algorithms to compute tight bounds, though the running times of the algorithms can be much higher and some variants are proven to be NP-hard.

Ahn et al. [1] study the efficient computation of the discrete Fréchet distance when the input is imprecise. The discrete Fréchet distance is a distance function which measures the similarity between two point sequences. The authors give polynomial time algorithms for computing the lower bound and approximation algorithms for computing the upper bound and the lower and upper bound of the distance under translation.

Despite the fact that both the Hausdorff distance and data imprecision are well-studied, we could not find any previous work on the combination. Since the application areas that use the Hausdorff distance (as well as other comparison measures) have to deal with imprecise data in practice, a better understanding of the implications is needed.

1.2. Contribution

In this paper, we assume that an imprecise point is modelled by a disc with a given centre and radius, that is, the real location of the point can be anywhere inside the specified disc. In general, it is possible that the discs intersect. We assume we have two sets of points, P and Q, and that at least one of them is imprecise. We want to compute the directed Hausdorff distance from P to Q. If P or Q is imprecise, then this does not lead to a unique number as answer anymore; instead, we want to compute the smallest (lower bound) and largest (upper bound) value this number can attain. Since there are three combinations of P, Q or both being imprecise, and two bounds to compute for each combination, we have different problems to solve. Additionally, some of these problems become easier if we restrict the model of imprecision in some way, for example by requiring the imprecision discs to be disjoint, or at least that they have a limited intersection depth, or by requiring that all discs have the same radius. Our results are summarised in Table 1.

In the next section, we review some definitions and structures that we use to obtain our results. After that, we present our three main results. In Section 3, we give a general algorithm for computing the upper bound, which works in all settings in the table, though it can be simplified (conceptually) in some settings. In Section 4, we prove the hardness of computing the lower bound in most settings. Finally, in Section 5, we give algorithmic results for computing the lower bound, exactly in some cases and approximately in others.

2. Preliminaries

In this section, we will review a number of known concepts and structures from computational geometry, as well as define variations of them that we will need later.

Imprecise points. An *imprecise point* is a closed disc in the plane. Let $\tilde{P} = \{\tilde{p}_1, \ldots, \tilde{p}_m\}$ and $\tilde{Q} = \{\tilde{q}_1, \ldots, \tilde{q}_n\}$ denote two imprecise point sets consisting of m and n closed discs, respectively.

Please cite this article in press as: C. Knauer, et al., The directed Hausdorff distance between imprecise point sets, Theoretical Computer Science (2011), doi:10.1016/j.tcs.2011.01.039

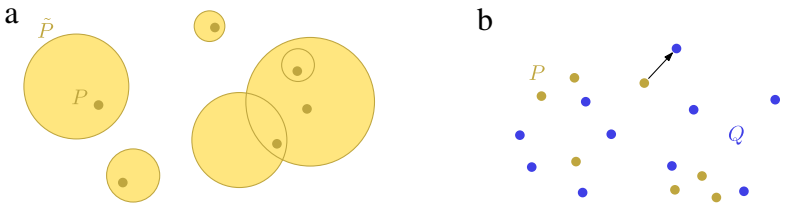

Fig. 1. (a) An example of imprecise point sets. (b) $h(P, Q)$ is realised by the pair of points indicated by the arrow.

We call a point set $P = \{p_1, \ldots, p_m\}$ a *precise realisation* of $\tilde{P} = \{\tilde{p}_1, \ldots, \tilde{p}_m\}$ if $p_i \in \tilde{p}_i$ for all i. We also write $P \Subset \tilde{P}$ in this case.

Fig. 1(a) shows an example of such a set of imprecise points and a possible precise realisation.

Hausdorff distance. The *directed Hausdorff distance* h from a point set $P = \{p_1, \ldots, p_m\}$ to a point set $Q = \{q_1, \ldots, q_n\}$ with an underlying Euclidean metric is defined as

$$h(P, Q) = \max_{p \in P} \min_{q \in Q} \|p - q\|$$

and can be computed in $O((n + m) \log n)$ time, see Alt et al. [2]. An example is shown in Fig. 1(b).

We define the directed Hausdorff distance between a precise and an imprecise or two imprecise point sets as the interval of all possible outcomes for that distance.

$$h(P, \tilde{Q}) = \left\{ h(P, Q) \mid Q \Subset \tilde{Q} \right\}, \quad h(\tilde{P}, Q) = \left\{ h(P, Q) \mid P \Subset \tilde{P} \right\}$$
$$h(\tilde{P}, \tilde{Q}) = \left\{ h(P, Q) \mid P \Subset \tilde{P}, Q \Subset \tilde{Q} \right\}.$$

Further, we denote the tight upper and lower bounds of this interval by $h_{\max}$ and $h_{\min}$, respectively, for example

$$h_{\max}(P, \tilde{Q}) = \max h(P, \tilde{Q}) \quad \text{and hence} \quad h(P, \tilde{Q}) = [h_{\min}(P, \tilde{Q}), h_{\max}(P, \tilde{Q})].$$

Voronoi diagrams. The *Voronoi diagram* of a set Q of n points (or sites) in the plane is the decomposition of the plane into cells such that all points in a cell share a common nearest site in Q. An *additively weighted* Voronoi diagram (also known as Apollonius diagram) of a set of weighted sites is a generalisation of the Voronoi diagram. Intuitively, we can view the weighted sites as discs whose radii correspond to their weights. If the weights are positive and the resulting discs do not overlap then the additively weighted Voronoi diagram corresponds to the Voronoi diagram of the discs. Formally, the additively weighted Voronoi diagram (of a set Q of weighted sites) decomposes the plane into cells (such that all points in a cell share a common nearest site in Q), where the weight of a site is subtracted from its distance to a point in the plane. Note that this definition does not require the weights to be positive. Observe that adding a constant amount to all weights does not change the structure of the diagram. Hence it is possible to use a Voronoi diagram data structure which only handles non-negative weights by adding the absolute amount of the largest disc radius to all weights. Voronoi diagrams and additively weighted Voronoi diagrams can be computed in $O(n \log n)$ time [5].

For a given set $\tilde{Q}$ of n discs in the plane, we define the *inverted additive Voronoi diagram* or *iaVD* to be the additively weighted Voronoi diagram whose sites are the centres of the discs, where the weight of each site corresponds to the negative radius of the disc. Hence, the iaVD partitions the plane into cells, where a point x in the plane is associated to a disc in $\tilde{Q}$ in the following way: For point x, we consider the point in each disc in $\tilde{Q}$ that is furthest away from x, and among those points, the one that is closest to x determines the disc we associate with x. See Fig. 3(a) for an example. The edges of the iaVD are pieces of hyperbolae. Adding the radius of the largest disc to the weight of each site we get an additively weighted Voronoi diagram with non-negative weights only, and such the iaVD can be computed in $O(n \log n)$ time.

Geometric k-centre. In the *geometric k-centre* problem, we are given a set P of points in the plane which should be covered by k discs of the same radius. The objective is to minimise the radius of these discs. This problem is known to be NP-hard, but can be approximated within a factor 2 [4] in $O(n \log k)$ time.

Matching. The *maximum matching* problem in a bipartite graph $G = (V, E)$ is to find a set of vertex-disjoint edges that is as large as possible. Using the algorithm of Hopcroft and Karp [8], this problem can be solved optimally in $O(|E|\sqrt{|V|})$ time.

3. Algorithms for computing the tight upper bound

In this section, we consider the following problem. Let $\tilde{P}$ and $\tilde{Q}$ be two sets of discs, of size m and n, respectively. The radii may all be different; an example input is shown in Fig. 2(a). Our aim is to compute point sets $P \Subset \tilde{P}$ and $Q \Subset \tilde{Q}$ such as to maximise the directed Hausdorff distance $h(P, Q)$. In other words, we want to place the points in P and Q such that one point of P is as far away as possible from all points in Q. The placements of the remaining points of P do not matter. So, we need to identify which point $\hat{p} \in P$ will play this important role. We need to place $\hat{p}$ such that after we placed all points in Q as far away from $\hat{p}$ as possible, this distance is maximised. Fig. 2(b) shows the optimal solution for the example.

Please cite this article in press as: C. Knauer, et al., The directed Hausdorff distance between imprecise point sets, Theoretical Computer Science (2011), doi:10.1016/j.tcs.2011.01.039

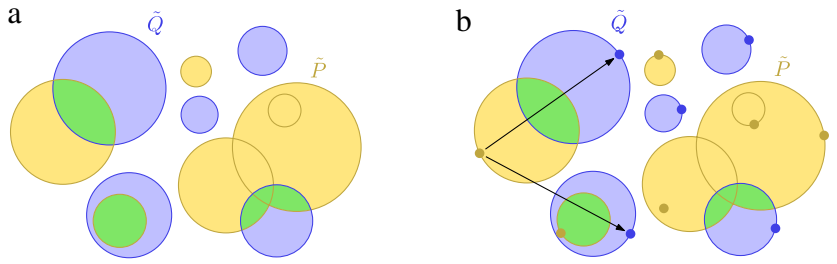

Fig. 2. (a) An example input of two imprecise point sets. (b) The optimal solution. The points in Q are all placed as far away from $\hat{p}$ as possible.

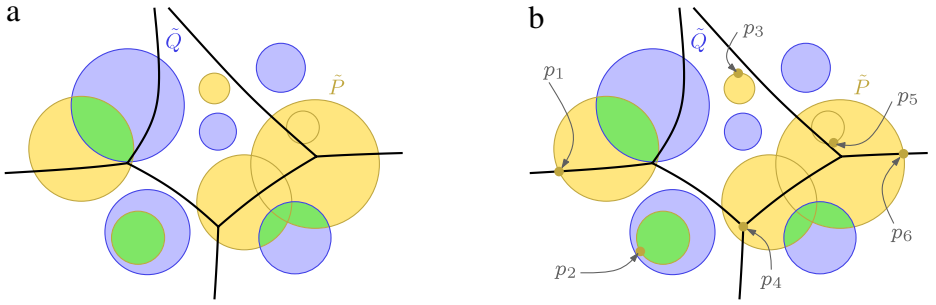

Fig. 3. (a) The inverted additive Voronoi diagram (iaVD) of $\tilde{Q}$. (b) The point set P placed locally optimal; p_4 is a type 1 placement, p_1, p_6 are type 2 and p_2, p_3, p_5 are type 3.

In the next section, we first describe a basic algorithm that solves the problem in quadratic time. After that, we show that under certain conditions, the running time can be improved to $O(n \log n)$.

3.1. Basic algorithm

We will first compute the *inverted additive Voronoi diagram* (iaVD) of $\tilde{Q}$ in $O(n \log n)$ time. Recall from Section 2 that this is the additively weighted Voronoi diagram of the centres of $\tilde{Q}$, weighted by their negative radii. Using this iaVD, we can identify three possible placement types for $p \in \tilde{p} \in \tilde{P}$ that are locally optimal inside $\tilde{p}$, as is illustrated in Fig. 3(b):

1. A vertex of the iaVD that lies inside disc $\tilde{p}$.
2. An intersection point between a Voronoi edge and the boundary of $\tilde{p}$.
3. A point on the boundary of $\tilde{p}$ that is furthest away from a site in the iaVD. This point lies in the Voronoi cell that also contains the centre of $\tilde{p}$, since it lies on the line going through the centre of its Voronoi site and the centre of $\tilde{p}$ and every Voronoi cell is star shaped.

We can then iterate over all $\tilde{p} \in \tilde{P}$ and its locally optimal placements, and we place a point p at the local optimal placements, as if it were $\hat{p}$. We determine $\hat{p}$ by keeping track of the locally optimal placement $p \in \tilde{p}$ such that the shortest distance between p and (the furthest point on) any disc in $\tilde{Q}$ is maximised.

Once $\hat{p}$ is known, we place all points in Q as far away from $\hat{p}$ as possible, and all points in $P \setminus \{\hat{p}\}$ anywhere inside their discs. The result is shown in Fig. 2(b). As it is possible that there are $O(mn)$ locally optimal placements of the second type (namely: an intersection between a disc boundary and a Voronoi edge), we conclude with the following theorem.

Theorem 1. *Given two sets $\tilde{P}$ and $\tilde{Q}$ of imprecise points of size m and n, respectively, we can compute $h_{\max}(\tilde{P}, \tilde{Q})$ and precise realisations $P \in \tilde{P}$ and $Q \in \tilde{Q}$ with $h(P, Q) = h_{\max}(\tilde{P}, \tilde{Q})$ in $O(nm + n \log n)$ time.*

3.2. Faster algorithms in special cases

In practice, it may be unlikely that we have to consider $O(mn)$ locally optimal placements. Indeed, in this section we show how the above result can be improved under certain assumptions. To speed up the algorithm, we make some observations about the nature of locally optimal placements.

Lemma 1. *Let $\tilde{p}$ be a disc in $\tilde{P}$, and let $\tilde{q}_1$ and $\tilde{q}_2$ be two discs in $\tilde{Q}$, such that part of the bisector of $\tilde{q}_1$ and $\tilde{q}_2$ forms an edge e of the iaVD that slices through $\tilde{p}$ (that is, e is not connected to a vertex of the iaVD inside $\tilde{p}$). Then the optimal placement of p occurs on the same side of e where the centre of $\tilde{p}$ is.*

Please cite this article in press as: C. Knauer, et al., The directed Hausdorff distance between imprecise point sets, Theoretical Computer Science (2011), doi:10.1016/j.tcs.2011.01.039

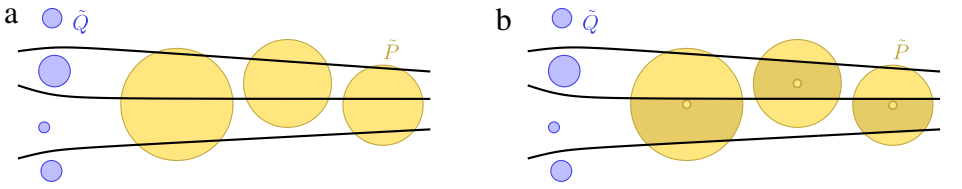

Fig. 4. (a) There could be a quadratic number of intersections between the edges of the iaVD of $\tilde{Q}$ and the discs in $\tilde{P}$. (b) However, when an edge slices through a disc $\tilde{p}$, we only need to inspect the cell that contains the centre of $\tilde{p}$.

Proof. Let p_c be the centre of $\tilde{p}$, q_{c1} the centre of $\tilde{q}_1$ and q_{c2} the centre of $\tilde{q}_2$. Now let f_1 be the point on the boundary of $\tilde{p}$ that is furthest away from q_{c1} (this would be a type 3 placement if $\tilde{q}_1$ was the only element of $\tilde{Q}$), and similarly let f_2 be the point furthest away from q_{c2}. Observe that e is part of a hyperbolic arc, of which q_{c1} and q_{c2} are the foci. Suppose w.l.o.g. that p_c is on the same side of e as q_{c1}.

Now, suppose that the optimal placement p is on the other side of e (that is, on the side of q_{c2}). Then q_{c2}, p_c and f_2 lie on a line. Because q_{c2} is a focus of e, the half-line starting from q_{c2} in the direction of p_c and f_2 intersects e only once. Since q_{c2} and p_c lie on opposite sides of e, it follows that p_c and f_2 must lie on the same side of e. This means that along the boundary of $\tilde{p}$, the intersection points with e have a better value than any other point on the side of q_{c2}, in particular, better than p, which is a contradiction. (Note that if there are other cells of the iaVD involved, the value of p could only be lower). □

This lemma basically says that if we want to place a certain point p locally optimally, we can start looking by walking from the centre of $\tilde{p}$ and never have to cross edges of the iaVD that slice through $\tilde{p}$. This can still mean, however, that we have to inspect a quadratic number of placements, but we can quantify it as follows.

Corollary 1. *Let $\tilde{p}$ be a disc in $\tilde{P}$, and suppose that the iaVD has t vertices inside $\tilde{p}$. Then we can find the locally optimal placement for p in $O(t)$ time.*

This immediately implies that if the discs of $\tilde{P}$ do not overlap, we can simply place all points p independently in linear time. Fig. 4 shows an example where there are a quadratic number of placements of type 2, but which do not all have to be inspected because of Lemma 1.

In fact, we can show the same result for something slightly stronger than disjoint discs. Now, assume that the intersection depth of the discs of $\tilde{P}$ is at most some constant c. Then, clearly, each vertex of the iaVD can appear in at most c discs of $\tilde{P}$. So, if each disc $\tilde{p}_i$ contains t_i vertices of the iaVD, we have $\sum_i t_i \leq cn$, and we can find all locally optimal placements in $O(n)$ time.

Theorem 2. *Given two sets $\tilde{P}$ and $\tilde{Q}$ of imprecise points of size m and n, respectively, where the discs in $\tilde{P}$ have constant intersection depth, we can compute $h_{\max}(\tilde{P}, \tilde{Q})$ and precise realisations $P \in \tilde{P}$ and $Q \in \tilde{Q}$ with $h(P, Q) = h_{\max}(\tilde{P}, \tilde{Q})$ in $O((m+n)\log(m+n))$ time.*

The algorithm described in this section works in the most general setting. In some more specific settings, the algorithm can be simplified. For example, if Q is a set of precise points, or if the regions of $\tilde{Q}$ are unit discs, the iaVD is the normal Voronoi diagram. This, however, does not influence the running time. If, on the other hand, P is a set of precise points, then there are only m possible locations for $\hat{p}$, and we do not need to look for all three placement types. In this case, we can compute $h_{\max}(P, \tilde{Q})$ more efficiently than in the general case, as stated in the following theorem.

Theorem 3. *Given a set P of points and a set $\tilde{Q}$ of imprecise points of size m and n, respectively, we can compute $h_{\max}(P, \tilde{Q})$ and a precise realisation $Q \in \tilde{Q}$ with $h(P, Q) = h_{\max}(P, \tilde{Q})$ in $O((m+n)\log n)$ time.*

4. Hardness results for tight lower bounds

In this section, we consider a transformation from the known NP-complete problem PLANAR 3-SAT [9] to the problem of computing $h_{\min}(P, \tilde{Q})$ for a set P of points and a set $\tilde{Q}$ of discs with radius r. In the PLANAR 3-SAT problem, we are given as input a 3-SAT formula f with the additional property that the graph $G(f)$ is planar, where $G(f)$ has a vertex for each variable and each clause in f, and there is an edge between a variable vertex and a clause vertex if the variable occurs in the clause. Having the boolean formula f and a planar embedding of $G(f)$, the transformation is as follows (see Fig. 5(a, b) for a general overview):

For each variable in f (or variable vertex v in $G(f)$), we construct a *cycle* C of alternating points in P and discs in $\tilde{Q}$. The discs have fixed radius r, and the distance between consecutive points and discs along the cycle is ϵ, such that $r = 2.5\epsilon$ (see Fig. 5(c)). There may be bends up to a certain angle, and also other geometric features necessary to connect cycles and chains (defined below). When looking only at the points P^C and discs $\tilde{Q}^C$ corresponding to a cycle C, we observe that by the construction of C, there are two realisations Q_0^C and Q_1^C of $\tilde{Q}^C$, such that $h(P^C, Q_0^C) = \epsilon$ and $h(P^C, Q_1^C) = \epsilon$ (see Figs. 5(d)

Please cite this article in press as: C. Knauer, et al., The directed Hausdorff distance between imprecise point sets, Theoretical Computer Science (2011), doi:10.1016/j.tcs.2011.01.039

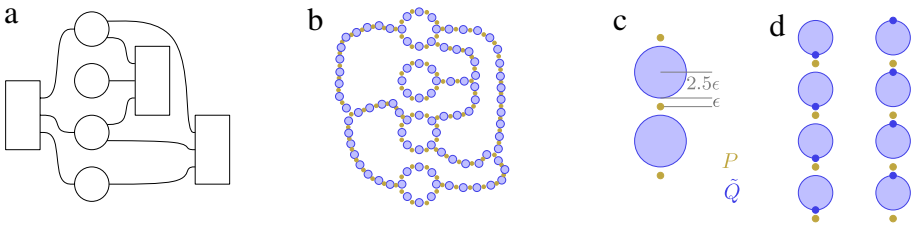

Fig. 5. (a) Planar embedding of $G(f)$, circles represent variables and rectangles represent clauses; (b) rough overview of how $G(f)$ is transformed into P and $\tilde{Q}$, some details are misrepresented; (c) alternating points and discs with geometric details; (d) two realisations with Hausdorff distance ϵ, representing opposite boolean values.

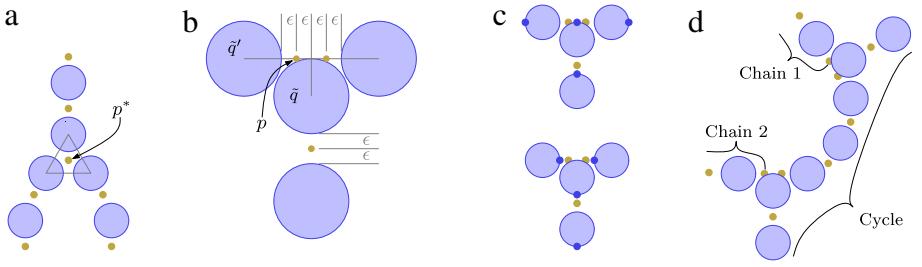

Fig. 6. (a) The endings of three chains arranged to represent a clause; (b) connection of a chain to a cycle with geometric details, the chain starts with p followed by $\tilde{q}'$, all other points and discs belong to the cycle; (c) two realisations with Hausdorff distance ϵ of the structure in the left subfigure; (d) two chains connecting to the same cycle where the chains tap opposite boolean values.

and 6(c)). These two realisations represent the two possible boolean values the variable for that cycle can have. Similar observations can also be made about chains to be described next.

For each edge $\{v, c\}$ in $G(f)$, we construct a *chain* of alternating points in P and discs in $\tilde{Q}$, where the discs have radius r, and the distance between consecutive points and discs along the chain is ϵ (see Fig. 5(c, d)). The chain connects the cycle corresponding to the variable v and the representation of clause c. More precisely, one end of this chain is a disc that will be part of a representation of clause c (see below for details). And the other end of this chain is a point p that is placed near a disc $\tilde{q} \in \tilde{Q}$ of the cycle C for variable v, such that p has a distance of ϵ to either $Q_0^C \cap \tilde{q}$ or $Q_1^C \cap \tilde{q}$, depending on whether v occurs negated or non-negated in c (see Fig. 6(b, d)). Also chains may have bends.

Each clause (or clause vertex c in $G(f)$) is represented by three discs and one additional point p^*, such that the disc centres lie on the vertices of an equilateral triangle, and the point has distance ϵ to each of the discs, and the three discs are ends of chains that connect to cycles that correspond to the three literals in the clause (see Fig. 6(a)).

Theorem 4. *Let P be a precise point set and $\tilde{Q}$ be an imprecise point set of pairwise disjoint discs. It is NP-hard to compute a δ-approximation of the directed Hausdorff distance $h_{min}(P, \tilde{Q})$ for $1 \leq \delta < 3$.*

Proof. For a given instance f to the PLANAR 3-SAT problem, let $G(f)$ be the planar graph corresponding to f, embedded such that all variables are on a line, and all clauses are on either side of them, see Fig. 5(a) ($G(f)$ can always be drawn in this way [9]). From this embedding, we compute (as described above) a set P of precise points, a set $\tilde{Q}$ of imprecise points, and numbers $\epsilon > 0$ and $r = 2.5\epsilon$.

Claim 1. *If f is satisfiable, then $h_{min}(P, \tilde{Q}) \leq \epsilon$.*

Proof of Claim 1. We consider an assignment with boolean values of the variables in f, such that f is satisfied, and we need to show that there exists a realisation $Q \Subset \tilde{Q}$, such that $h(P, Q) \leq \epsilon$. (Note that by construction, there is no realisation $Q' \Subset \tilde{Q}$, such that $h(P, Q') < \epsilon$.) For each cycle C of a variable, we choose either Q_0^C or Q_1^C as realisations of the imprecise point set $\tilde{Q}^C$, depending on whether the variable is false or true. Discs on chains are realised in the following way: at the ending of the chain that connects to a cycle C, we have a point p near a disc $\tilde{q} \in \tilde{Q}^C$, and $\tilde{q}$ is realised by a point q. The next object along the chain is a disc $\tilde{q}'$ (see Fig. 6(b)). We realise $\tilde{q}'$ in either of two ways as depicted in Fig. 6(c), depending on whether the distance between p and q is equal to ϵ or greater than ϵ. This corresponds to a variable being either true or false. And the boolean value of the corresponding literal is then propagated to the other end of the chain to a clause c. Since f is satisfiable, there is at least one literal in each clause that satisfies the clause. Hence, there is at least one chain with a realisation such that the point p^* has distance at most ϵ to a point of this realisation. □

Claim 2. *If f is not satisfiable, then $h_{min}(P, \tilde{Q}) \geq 3\epsilon$.*

Proof of Claim 2. We will prove the contraposition, namely if $h_{\min}(P, \tilde{Q}) < 3\epsilon$, then f is satisfiable. We consider a realisation $Q \Subset \tilde{Q}$ with $h(P, Q) < 3\epsilon$, and we need to construct a variable assignment that satisfies f. We first observe

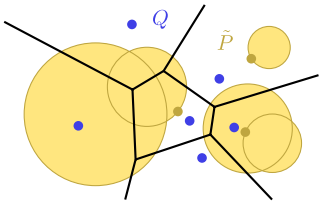

Fig. 7. Placing points in P as close as possible to their closest neighbour in Q. A point in P is hidden by a point in Q when the points are identical.

that the only way where two points in P can be matched to the same point $q \in \tilde{q} \in \tilde{Q}$ is where a chain connects to a cycle. (Otherwise, the distance between the two points in P is larger than 6ϵ, and hence, they cannot be matched to the same point q.) And in this case, one of the points in P is the end point of a chain, the other point in P belongs to a cycle, and $\tilde{q}$ belongs to the same cycle (see Fig. 6(b)). From this we make an observation about how the points along chains and cycles are matched to discs along the same chains and cycles. Let us consider the sequence $p_0, \tilde{q}_0, p_1, \tilde{q}_1, p_2, \tilde{q}_2, \ldots$ of points and discs ordered along a fixed cycle C. Exactly one of the following two things is true for all $i = 0, 1, 2, \ldots$ (modulo length of C):

- p_i is matched to a point $q_i \in \tilde{q}_i$, i.e. $||p_i, q_i|| < 3\epsilon$; or
- p_i is matched to a point $q_{i-1} \in \tilde{q}_{i-1}$, i.e. $||p_i, q_{i-1}|| < 3\epsilon$.

In other words, each point on C is matched to the next disc on C in clockwise order, or each point on C is matched to the next disc on C in counter-clockwise order, but there is no mix of these along C. From these two possibilities for cycle C, we derive the boolean value of the variable corresponding to C. This assignment is in accordance with the two realisations Q_0^C and Q_1^C (as defined above), which represent false and true. What is left to show is that this assignment satisfies f. To see this, we consider any clause c of f and argue that c is satisfied. From the construction, we know that c is represented by one point $p^* \in P$ and three discs being the endings of three chains. There must be a point $q \in Q$ such that $||p^*, q|| < 3\epsilon$, and q must lie in one of the discs that represent the clause c. This disc $\tilde{q}_0$ is the ending of a chain $\tilde{q}_0, p_0, \tilde{q}_1, p_1, \tilde{q}_2, \ldots, p_j$. In a similar way as above, we conclude for this chain that:

- p^* is matched to a point $q_0 \in \tilde{q}_0$, i.e. $||p^*, q_0|| < 3\epsilon$; and
- p_i is matched to a point $q_{i+1} \in \tilde{q}_{i+1}$, for $i = 0, \ldots, j-1$; and
- p_j is matched to a point $q_j \in \tilde{q}_j$, for some disc $\tilde{q}_j$ on some cycle C.

The variable corresponding to C has a boolean value, according to the realisation of the discs along C. Depending on whether this variable occurs negated or non-negated in the clause c, the chain $\tilde{q}_0, p_0, \tilde{q}_1, p_1, \tilde{q}_2, \ldots, p_j$ is connected to the cycle C, such that "the boolean value true is propagated along the chain". Hence, by construction we have that the boolean value of the variable corresponding to C satisfies the clause c. □

The proof of Theorem 4 follows now from Claims 1 and 2, and from observing that the construction can be done without any intersection between discs and/or points, and such that chains and/or cycles are far enough apart from each other not to interfere. We also note that the size of P and $\tilde{Q}$ is polynomial in the size of $G(f)$, which follows from our planar embedding of $G(f)$. □

5. Algorithms for tight lower bounds

In this section we present algorithms for computing the minimum of $h(\tilde{P}, Q)$ and $h(P, \tilde{Q})$. As we have seen in the previous section, the latter problem is NP-hard and even hard to approximate in some settings. In the following we give a 4-approximation for the general case, an optimal 3-approximation for disjoint discs and an algorithm for the case which is not NP-hard when the Hausdorff distance is small. Many results in this section rely on similar ideas. Therefore, we will describe several (sub-) algorithms with different approximation factors and running times depending on the value d of the optimal solution. Afterwards, we discuss how to apply them to obtain the results claimed in Table 1.

5.1. *Algorithm* PlaceTogether

In this section, we describe an algorithm for the case where we have an imprecise point set $\tilde{P}$ and a precise point set Q. We place all points in $\tilde{P}$ as close to a point in Q as possible. Fig. 7 shows an example. For each pair $(\tilde{p}, q)$ with $\tilde{p} \in \tilde{P}$ and $q \in Q$ we could simply compute the placement $p \in \tilde{p}$ minimising the Hausdorff distance and keep track of the longest distance over all pairs. This takes $O(mn)$ time.

However, unless n is exponentially larger than m we can do better because of the following observation:

Observation 1. *Given a disc $\tilde{p}$ and a point set Q. The point in Q closest to the centre of $\tilde{p}$ is also closest to any point in $\tilde{p}$.*

Please cite this article in press as: C. Knauer, et al., The directed Hausdorff distance between imprecise point sets, Theoretical Computer Science (2011), doi:10.1016/j.tcs.2011.01.039

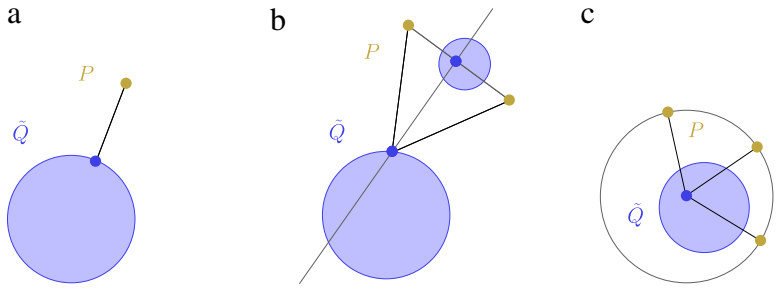

Fig. 8. There are only polynomially many candidates for the infimum of $h(P, \tilde{Q})$ which are determined by (a) one point of P, (b) by two points of P or (c) three or more points of P.

As a consequence, we can do the following. First we compute, in $O(n \log n)$ time, the Voronoi diagram of Q. Then, we locate the centre of each point in $\tilde{P}$ in this Voronoi diagram in total $O(m \log n)$ time. Once located, we place in constant time each point $p \in \tilde{p}$ as close as possible to the site whose Voronoi cell contains the centre of $\tilde{p}$.

Theorem 5. *Let $\tilde{P}$ denote an imprecise point set consisting of m discs and Q denote a precise point set consisting of n points. The tight lower bound of $h(\tilde{P}, Q)$ can be computed in $O(\min\{mn, (m+n)\log n\})$ time.*

5.2. Sub-algorithm Candidates

In the case where P is precise and $\tilde{Q}$ is imprecise, we start with a simple sub-algorithm Candidates to establish Lemma 2. The result will be used later.

Lemma 2. *Let P denote a precise point set consisting of m points and $\tilde{Q}$ denote an imprecise point set consisting of n discs. It is possible to reduce the possible values of $h_{min}(P, \tilde{Q})$ to $O(m^3 + m^2 n)$ many candidates in $O(m^3 + m^2 n)$ time.*

The Hausdorff distance is realised locally at one or several points of Q and P, i.e. we only need to look at the placements for these points of Q. Let $q \in \tilde{q} \in \tilde{Q}$ be such a placement of an imprecise point in $\tilde{Q}$ which realises the Hausdorff distance d. The distance d can be determined by q together with one or several points of P. These points may not be the only points of P which are matched to q, but among the points matched to q they have the largest distance to q. If d is determined by one point of P there are $O(mn)$ possibilities, see Fig. 8(a). If d is determined by two points $p_1, p_2 \in P$ the point q lies on the bisector of the line segment p_1p_2, see Fig. 8(b), for which $O(nm^2)$ possibilities exist. Finally, if d is determined by three or more points, all these points lie on a circle whose centre is q. Such a circle is determined by three of its points and thus there are $O(m^3)$ possible locations, see Fig. 8(c). The algorithm simply computes and returns all $O(m^3 + m^2 n)$ locations in $O(m^3 + m^2 n)$ time.

5.3. Algorithm IndependentSets

This algorithm computes exactly the Hausdorff distance from a precise point set P to an imprecise point set $\tilde{Q}$ when the distance is small. This is an exception to the general NP-hardness of that setting.

Theorem 6. *Let P denote a precise point set consisting of m points and $\tilde{Q}$ denote an imprecise point set consisting of n* disjoint *discs. Algorithm IndependentSets computes whether the tight lower bound for $h(P, \tilde{Q})$ is smaller than $r(\sqrt{5-2\sqrt{3}}-1)/2$ where r is the radius of the smallest disc in $\tilde{Q}$. If this is the case, the exact value of $h_{min}(P, \tilde{Q})$ is computed. The running time is $O(m^3 + m^2 n + n \log^2 n)$.*

First we compute the set of possible candidates for $h_{\min}(P, \tilde{Q})$ by Candidates and discard all values greater or equal than $c = r(\sqrt{5-2\sqrt{3}}-1)/2$. Now we perform a binary search on the remaining values in order to determine the smallest value d for which the predicate described below evaluates to true. If such a candidate exists, the algorithm returns d as the value of the bound. Otherwise $h_{\min}(P, \tilde{Q}) \geq r(\sqrt{5-2\sqrt{3}}-1)/2$.

Let $p(d)$ denote the disc of radius d around a point $p \in P$. There must be at least one point of Q in $p(d)$ to which p can be matched within a distance smaller or equal than d. The computation of the predicate relies on two observations: All considered values are so small that no $p(d)$ intersects more than two discs of $\tilde{Q}$. Note that a disc that intersects more than two disjoint discs of $\tilde{Q}$ has a radius of at least $r(2/\sqrt{3}-1)$, which is greater than c, see Fig. 9(a). Thus, there are at most two possible matching partners for each point $p \in P$.

The second observation is that each $p(d)$ has to intersect at least one disc of $\tilde{Q}$, otherwise the Hausdorff distance would be greater than d at p.

Please cite this article in press as: C. Knauer, et al., The directed Hausdorff distance between imprecise point sets, Theoretical Computer Science (2011), doi:10.1016/j.tcs.2011.01.039

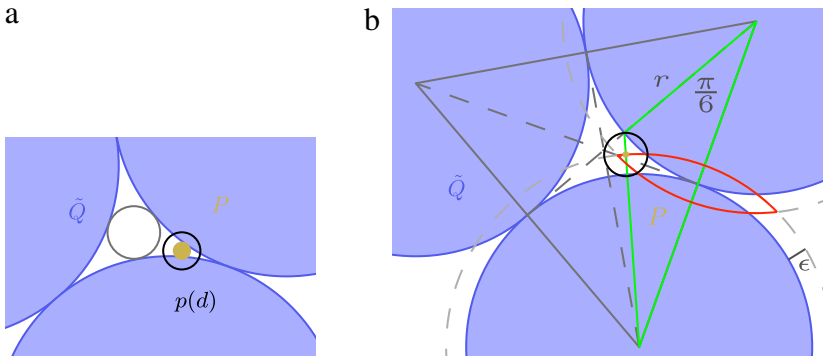

Fig. 9. (a) The grey circle in the middle has the minimal radius of $r(2/\sqrt{3}-1)$ which is necessary to intersect at least three discs of $\tilde{Q}$. Because all considered candidates for the minimal Hausdorff distance are smaller than the radius of the grey disc, there are at most two possible matching partners for each point in P. (b) The black circle has the maximal radius c for which no two circles intersecting two different pairs of discs in $\tilde{Q}$ can be stabbed by only one point $q \in \tilde{q}$. Since it has to intersect two discs in $\tilde{Q}$, its centre must lie within the red lune of these discs grown by c. Furthermore, its boundary must not intersect the green segment denoted by r because it could intersect another black circle intersecting the upper two discs of $\tilde{Q}$, otherwise. Using the cosine formula it holds that $(r+2c)^2 < r^2 + (2r)^2 - 2r2r\cos\frac{\pi}{6}$. Solving the latter inequality for c yields $c < r(\sqrt{5-2\sqrt{3}}-1)/2$.

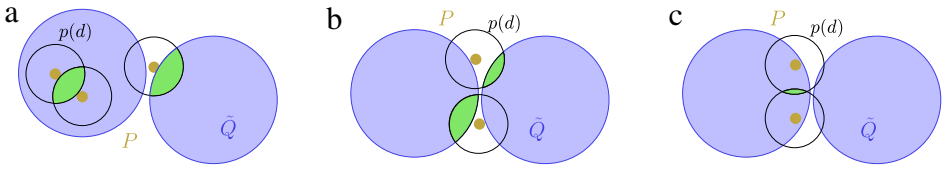

Fig. 10. (a) The green areas are the feasible regions of the discs in $\tilde{Q}$. A point $q \in \tilde{q}$ may only be placed within its feasible region. The two points of P lying to the left have degree 1, the point lying to right has degree 2. (b, c) The points p have degree two. Both cases show a scenario which allows to match the points locally, i.e. only considering the set D and its two corresponding feasible regions. Case (b) shows one of the two possible matchings. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

We define a point $p \in P$ to have degree 1 if $p(d)$ intersects just one disc $\tilde{q} \in \tilde{Q}$ and to have degree 2 if it intersects two discs of $\tilde{Q}$.

The predicate tests, whether $h_{\min}(P, \tilde{Q}) \leq d$. To this end we associate with each $\tilde{q}_i \in \tilde{Q}$ a *feasible region* F_i and a set C_i called *children* of $\tilde{q}_i$. The feasible region contains the valid placements of $q_i \in \tilde{q}_i$. The children of a point $\tilde{q}_i$ are the points of P that can only be matched to $\tilde{q}_i$ because otherwise $h_{\min}(P, \tilde{Q})$ would be greater than d. In other words, the children of $\tilde{q}_i$ demand that q_i is placed in its feasible region F_i. See Fig. 10(a) for an illustration.

We restrict the feasible regions and children in an iterative manner with the help of the sub-algorithms Remove-degree-1-discs and Remove-degree-2-discs. If a feasible region turns out to be empty, the computation stops and the predicate returns false. The first sub-algorithm considers only points p of degree 1 and restricts the feasible regions of the discs intersected by $p(d)$. The second sub-algorithm restricts the feasible regions of discs intersected by points of degree 2.

Before calling Remove-degree-1-discs we define a set P_R of all points which are not matched so far and set $P_R := P$.

Remove-degree-1-discs

1 **forall the** $\tilde{q}_i \in \tilde{Q}$ **do**
2 | set $F_i := \tilde{q}_i$
3 | set $C_i := \varnothing$
4 **while** *there is some* $p \in P$ *such that* $p(d)$ *intersects only one* F_i **do**
5 | set $F_i := F_i \cap p(d)$
6 | **if** $F_i = \varnothing$ **then**
7 | | **return** false
8 | set $C_i := C_i \cup \{p\}$
9 set $P_R := P_R \setminus \bigcup_i C_i$
10 Remove-degree-2-discs

In line 9 the remaining points $P_R = P \setminus \bigcup_i C_i$ are points whose disc $p(d)$ intersects exactly two feasible regions. It is still possible to match points in P to points in $\tilde{Q}$ by only analysing their local environment. This is done by Remove-2-discs.

Please cite this article in press as: C. Knauer, et al., The directed Hausdorff distance between imprecise point sets, Theoretical Computer Science (2011), doi:10.1016/j.tcs.2011.01.039

```
REMOVE-DEGREE-2-DISCS
1  foreach pair of feasible regions (F_i, F_j), i ≠ j do
2      compute the set D of discs p(d) intersecting both F_i and F_j
3      if
4      D can be stabbed by one point of either F_i or F_j (but not by both of them)   or
5      D needs one point of F_i and one point of F_2 to be stabbed
6      then
7          restrict F_i and F_j accordingly
8          if F_i = ∅ ∨ F_j = ∅ then
9              return false
10         REMOVE-DEGREE-1-DISCS
11 BUILDGRAPH
```

Note that, all sets D of line 2 partition the set of the discs $p(d)$ of the points in P_R. Line 7 restricts the matching for points of degree 2 whose matching does not interfere with the matching of points with other pairs of feasible regions. See Fig. 10(b) and (c) for an illustration of the two possible scenarios allowing a local matching.

In line 11 all discs of each subset D can be stabbed by only one point of the two feasible regions F_i and F_j they intersect. Further, it holds that no two discs $p(d)$ of different sets D can be stabbed by only one point, because $d < r(\sqrt{5-2\sqrt{3}}-1)/2$, see Fig. 9(b).

Now, it is possible to check for a valid point matching of the remaining points in P_R by computing the maximum matching in a bipartite graph as follows: BUILDGRAPH builds a bipartite graph on the feasible regions and the sets D of the partition of the discs $p(d)$ of the points in P_R. For each cell D of the partition there are two edges in the graph connecting D with the two feasible regions that the discs in D intersect. We now compute a maximum matching on that graph. If this matching connects all D-vertices with a feasible region, the predicate returns true and the bound for the Hausdorff distance is smaller or equal than v. Otherwise the predicate returns false.

It is simple to return a matching which realises the Hausdorff distance which the predicate proved to be realisable. Therefore, we first consider the feasible regions which are adjacent to a vertex in the maximum matching. We place the point in such a feasible region such that it intersects all discs in the adjacent set D of discs. For all other $\tilde{q}_i \in \tilde{Q}$ we place their point q_i somewhere within its feasible region F_i.

Running time. The algorithm consists of three phases: It first computes a polynomial set of candidates which takes $O(m^3 + m^2n)$ time. On this set we perform a binary search using the predicate. The computation of the predicate is done by some recursive calls of the two sub-algorithms REMOVE-DEGREE-1-DISCS and REMOVE-DEGREE-2-DISCS. These need to know the intersections of the discs $p(d)$ with the feasible regions, which are the discs in $\tilde{Q}$ in the beginning. We store these intersections distributed with every point $p \in P$ and store references with each feasible region to the $p(d)$ it intersects. The initial set of the intersections can be computed using a sweep-line in $O(m+n)\log(m+n)$ time. The restrictions of the feasible regions can take at most $O(m)$ time. Further we maintain one point set containing points $p \in P$ with degree 1 and a second point set for points of degree 2. We move a point from the second to the first point set if its degree is decreased. Thus, having the initial intersection set, all calls of REMOVE-DEGREE-1-DISCS without line 10 take $O(m)$ time.

The sub-algorithm REMOVE-DEGREE-2-DISCS needs to iterate over all pairs of feasible regions. Instead of considering all possible pairs we only maintain a set of region pairs which indeed intersect some discs $p(d)$. Because all D's partition the points in P there are at most m discs to consider in the stabbing analysis from lines 1–6, thus REMOVE-DEGREE-2-DISCS needs $O(m)$ time per call. Since it is called at most m times by REMOVE-DEGREE-1-DISCS its overall running time is $O(m^2)$.

Finally we need to compute a maximum matching in the bipartite graph. Using the algorithm of Hopcroft and Karp [8] this needs $O(m\sqrt{m+n^2})$ time.

Putting all together we get a running time of $O(m^3 + m^2n + ((m+n)\log(m+n) + m^2 + m\sqrt{m+n^2})\log(m^3+m^2n))$ which can be simplified to $O(m^3 + m^2n + n\log^2 n)$.

5.4. *Algorithm* GROWNDISCS

In this section, we present an approximation algorithm for precise P and imprecise $\tilde{Q}$. As a subroutine in this algorithm, we assume that we are given an algorithm that computes a c-approximation to the geometric k-centre problem (see Section 2), in time $T(k, n)$. We need this because when we have k discs of $\tilde{Q}$ which partially overlap, and there are n points of P in the overlap, computing a lower bound on the Hausdorff distance for this subset is exactly solving the geometric k-centre problem. Fig. 11(a) shows an example of the problem.

We first compute the set of possible values of the Hausdorff distance using the algorithm CANDIDATES, followed by a binary search on the resulting candidate values in order to determine the smallest value d for which the predicate described below evaluates to true. Therefore, we now first describe a decision algorithm.

Decision algorithm. Let d be any given positive value. If there exists a solution to our problem with distance at most d, the decision algorithm returns a solution with distance at most $(c+2)d$. If no solution of distance at most d exists, the algorithm either still returns a solution with distance at most $(c+2)d$, or it returns false.

Please cite this article in press as: C. Knauer, et al., The directed Hausdorff distance between imprecise point sets, Theoretical Computer Science (2011), doi:10.1016/j.tcs.2011.01.039

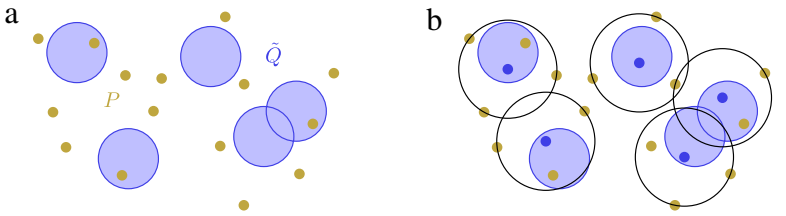

Fig. 11. (a) An example input, consisting of a set P of precise points and a set $\tilde{Q}$ of imprecise points. (b) The optimal output. A set of circles of radius d is shown around the points in Q, which cover the points in P.

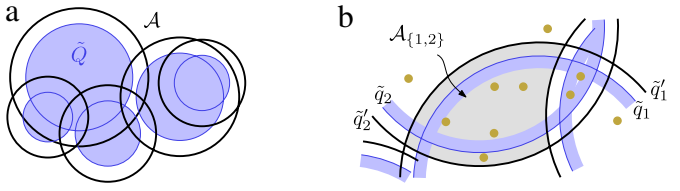

Fig. 12. (a) The black circles form the arrangement $\mathcal{A}$ of the expanded discs $\tilde{Q}'$. (b) Each cell of the arrangement is determined by the indices of the discs it lies inside.

Let $\tilde{q}_1, \ldots, \tilde{q}_n$ be the discs in $\tilde{Q}$ where disc $\tilde{q}_i$ has radius r_i. We define the *grown* disc $\tilde{q}'_i$ to be the disc with the same centre point as $\tilde{q}_i$, but with radius $r_i + d$. We call the resulting set $\tilde{Q}'$.

Observation 2. *If P is not covered by $\tilde{Q}'$, then there exists no solution of value d.*

The correctness of Observation 2 is easy to see. Suppose the Hausdorff distance was smaller than or equal d, then every point of P would lie at most d away from a placement of a point $\tilde{q}$ and thus lie within $\tilde{Q}'$.

So, we assume P is covered by $\tilde{Q}'$. We can test this easily, and immediately return false if this is not the case. Now, we compute the arrangement $\mathcal{A}$ of $\tilde{Q}'$, i.e. the partition of the plane formed by the boundary of the expanded discs. This arrangement has a quadratic complexity in the worst case. Fig. 12(a) shows an example of the arrangement. If $I \subseteq \{1, \ldots, n\}$ is a certain set of indices, denote by $\mathcal{A}_I$ the cell of the arrangement in the intersection of all discs $\{\tilde{q}'_i \mid i \in I\}$, but not inside any other disc. (Of course, most of these cells do not exist, since there is only a quadratic number of cells.) Each cell of this arrangement contains a subset of P; we define P_I to be the set of points of P inside $\mathcal{A}_I$. Fig. 12(b) shows an example.

Observation 3. *Let I be a set of indices. In the optimal solution $Q \Subset \tilde{Q}$, all the points of P_I are covered by circles of radius d around the points in $\{q_i \mid i \in I\}$.*

Proof. Since the optimal solution has Hausdorff distance $h(P, Q) \leq d$, we know that each point $p \in P$ is covered by some circle of radius d around a point $q \in Q$. Now assume that $p \in P_I$. Then we know that $|pq| \leq d$, and $q \in \tilde{q}$, therefore $p \in \tilde{q}'$. So, by definition of $\mathcal{A}_I$, q must be q_i for some $i \in I$. □

This observation suggests we can solve the problem somehow separately in each cell of $\mathcal{A}$. For a given cell $\mathcal{A}_I$, the optimal solution uses $k_I \leq |I|$ circles of radius d (centred around points of Q) to cover the points in P_I. Now, we could compute such a set of circles (most likely a different set) by applying the c-approximation algorithm for geometric k-centre $O(\log k_I)$ times in binary search-like fashion on the value of k, until we find the smallest number k'_I for which the algorithm returns a solution of radius smaller than cd. Note that the approximation guarantee implies that $k'_I \leq k_I$. This would provide us a set C_I of $k'_I \leq k_I$ circles of radius cd that cover P_I. However, there is a problem with this approach. The solutions are not independent: it is possible that a certain circle of the optimal solution covers points from two different cells of the arrangement. This means we may have constructed more than n circles.

So, what we do instead is this. We process the cells of the arrangement in any order. For the first cell $\mathcal{A}_I$, we compute a set C_I of at most k_I circles of radius cd that cover P_I. Now, we grow our circles until they have radius $(c + 2)d$. This ensures that any points of P outside A_I that were covered by discs of the optimal solution that were covering any points of P_I, are now also covered by C_I. Fig. 13 illustrates this.

A second complication comes from the fact that we required the centres of the circles to be in $\tilde{Q}$, not just in $\tilde{Q}'$. In order to ensure this, we simply move the circle centres to the closest point in their discs, moving them by at most d. Since the circles are enlarged by $2d$, the moved and enlarged circles will still cover all points of P that were covered by the original circles. Fig. 14 shows this case. Furthermore, this case does not interfere with the case described above, because a circle cannot at the same time be close to the boundary of $\mathcal{A}_I$ and far enough away from it not to cover a point that is covered by a circle that also covers a point from a neighbouring cell.

For each next cell, we only consider those points that have not been covered yet, and otherwise proceed in the same way.

This procedure results in a set C of at most n circles, composed of a set C_I for each cell $\mathcal{A}_I$ of the arrangement. This set has the property that each C_I contains no more circles than the corresponding set in the optimal solution. This implies that there exists a matching between C and $\tilde{Q}'$ in the graph that has an edge between circle o and disc $\tilde{q}'_i$ if c is in a set C_I where $i \in I$.

Please cite this article in press as: C. Knauer, et al., The directed Hausdorff distance between imprecise point sets, Theoretical Computer Science (2011), doi:10.1016/j.tcs.2011.01.039

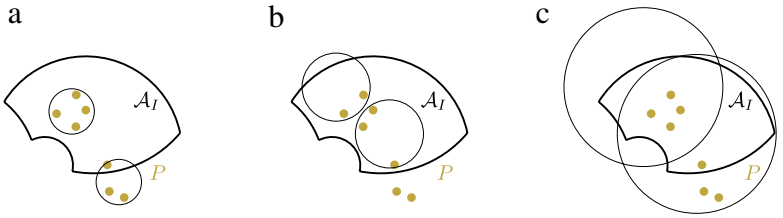

Fig. 13. (a) A cell $\mathcal{A}_I$ of the arrangement, and a set of two circles of radius d covering P_I in the optimal solution. (b) A different set of at most two circles of radius cd covering P_I, as produced by the subroutine. (c) The enlarged circles with radius $(c+2)d$ also cover all points outside $\mathcal{A}_I$ that could be covered by the circles of the optimal solution.

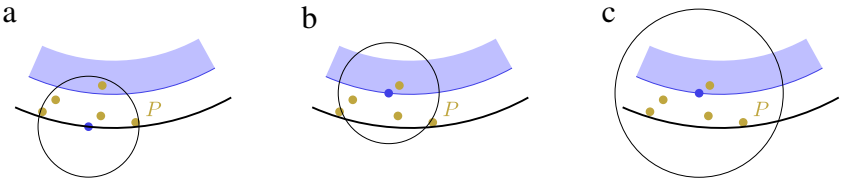

Fig. 14. (a) A circle of radius cd covers a number of points of P_I inside a certain cell $\mathcal{A}_I$ of the arrangement. The centre q of the circle lies inside $\mathcal{A}_I$, but not inside the region $\tilde{q}$. (b) The point q has been moved into $\tilde{q}$, but now some points of P_I that were covered are no longer covered. (c) The enlarged circle of radius $(c+2)d$ covers the points again.

Clearly, this means that the centre of o is inside $\tilde{q}'_i$. Since an optimal matching exists, we can also compute one efficiently (although it may be a different one).

For each value d, we spend $O(n^2)$ time to compute $\mathcal{A}$, and $\log k_I \cdot T(k_I, |P_I|)$ time per cell to solve the geometric k-centre problem. If k is the largest value of k_I over all I, then a crude upper bound for this is $n^2 T(k, m)$. As seen in the previous section, a matching can be computed in $O(mn + m\sqrt{m})$ time [8]. So, we spend $O(n^2 \log k \cdot T(k, m) + mn + m\sqrt{m})$ time in total on the decision algorithm.

We can summarise:

Lemma 3. *Let a value d and an algorithm that can compute a c-approximation to the geometric k-centre problem in time $T(k, n)$ be given. We can compute, in $O(n^2 \log k \cdot T(k, m) + mn + m\sqrt{m})$ time, either a solution with distance d' that is at most $(c+2)$ times larger than d, or decide that no solution of distance d exists.*

Optimisation algorithm. We now describe how to use the decision algorithm to obtain an optimisation algorithm. First, we compute the set of possible values of the Hausdorff distance using algorithm Candidates. Then we do a binary search on the value of d, by picking the middle of the remaining candidates and recursing up if the decision algorithm returns "no" or down if it returns a solution. This procedure will result in two consecutive candidate values d_1 and d_2, such that d_1 returns no solution but d_2 does. Let d' be the solution returned by the decision algorithm run on d_2, and let d^* be the optimal solution. Then we know that $d' \leq (c+2)d_2$. We also know that $d^* > d_1$, and since the next candidate is d_2, this means $d^* \geq d_2$. Therefore, $d' \leq (c+2)d^*$. Since obviously also $d^* \leq d'$, we conclude that d' is a $(c+2)$ approximation of d^*.

As for the running time, we first execute Candidates in $O(m^3 + m^2 n)$ time. This results in a set of $O(m^3 + m^2 n)$ candidates; doing a binary search on them we execute the decision algorithm described above $O(\log(m+n))$ times, for a total of $O((n^2 \log k \cdot T(k, m) + mn + m\sqrt{m}) \log m + n)$ time. Some terms are dominated by the computation of the algorithms, which results in the following theorem:

Theorem 7. *Let P denote a precise point set consisting of m points and $\tilde{Q}$ denote an imprecise point set consisting of n discs. Given a c-approximation to the geometric k-covering problem that runs in $T(k, m)$ time, we can compute a $(c+2)$-approximation to the tight lower bound of $h(P, \tilde{Q})$ in $O(m^3 + m^2 n + mn \log(m+n) + n^2 \log(m+n) \log k \cdot T(k, m))$ time, where $k \leq n$ is an internal parameter of the optimal solution.*

5.5. Algorithm CentrePoints

A trivial algorithm is to place all points in Q in the centres of their discs. This algorithm is a c-approximation if the smallest possible Hausdorff distance is at least $r_{\max}/(c-1)$, where $r_{\max}$ denotes the radius of the largest disc in $\tilde{Q}$.

Lemma 4. *Let P denote a precise point set consisting of m points and $\tilde{Q}$ denote an imprecise point set consisting of n discs. We can compute a c-approximation to the tight lower bound of $h(P, \tilde{Q})$ if the Hausdorff distance is at least $r_{max}/(c-1)$, where $r_{\max}$ denotes the radius of the largest disc in $\tilde{Q}$.*

5.6. Putting the algorithms together

When P is precise and $\tilde{Q}$ is imprecise, we note that by Theorem 7 Algorithm GrownDiscs immediately presents a 4-approximation for the case when the discs may have different radii and overlap, which we obtain by plugging in a 2-

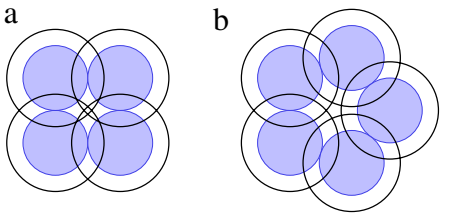

Fig. 15. (a) Four disjoint discs of radius r of $\tilde{Q}$ can be laid out such that the arrangement of enlarged circles of radius $1\frac{1}{2}r$ have a common intersection. (b) With five discs, this is not possible anymore.

approximation algorithm for geometric k-covering that runs in $O(m \log k)$ time [4]. (Note that although the authors also present a solution for approximating the value of k for a fixed radius that achieves the much better approximation factor of $(1+\varepsilon)$, we cannot use that solution since we require a guarantee on the radius, not the number of discs.)

Corollary 2. *Let P denote a precise point set consisting of m points and $\tilde{Q}$ denote an imprecise point set consisting of n discs. We can compute a 4-approximation to the tight lower bound of $h(P, \tilde{Q})$ in $O(m^3 + m^2n + mn^2 \log(m+n) \log^2 n)$ time in the worst case.*

We can improve this algorithm by first testing whether the tight lower bound is smaller than $r_{\min}(\sqrt{5-2\sqrt{3}}-1)/2$ where $r_{\min}$ denotes the radius of the smallest disc in $\tilde{Q}$ using Algorithm IndependentSets, without increasing the asymptotic running time. If it is, then we can actually compute the exact solution.

Furthermore, when the discs are disjoint and all have the same size, we can improve this result to a 3-approximation by combining the algorithms GrownDiscs and CentrePoints:

Theorem 8. *Let P denote a precise point set consisting of m points and $\tilde{Q}$ denote an imprecise point set consisting of n* disjoint *discs of the same radius. The tight lower bound for $h(P, \tilde{Q})$ is 3-approximable in time $O(m^3 + m^2n + n \log^2 n)$.*

First we test whether the lower bound on the Hausdorff distance is at least $r/2$ by applying CentrePoints and checking whether the resulting distance is at least $3/2r$. If it is, we are done. Otherwise, note that each cell of $\mathcal{A}$ is a subset of the intersection of $k \leq 4$ discs, because $\tilde{Q}$'s discs are disjoint and the Hausdorff distance is less than $r/2$, see Fig. 15(a) and (b). Therefore, by Theorem 7 we can obtain a 3-approximation from Algorithm GrownDiscs by plugging in an exact algorithm to solve the geometric 4-covering problem.

We can solve the geometric 4-covering problem exactly by computing the arrangement of discs of radius d around the points to be covered. The arrangement has quadratic complexity. We need to find out whether there is a set of four cells such that every disc of the arrangement contains at least one of the cells. There are $\binom{O(m^2)}{4} \in O(m^8)$ such combinations to test, and by keeping track of which discs are already taken care of each can be tested in constant time. So, using this algorithm, we have a $1+2=3$-approximation to the original problem for disjoint unit discs. The total running time now becomes $O(n^2m^8 \log(m+n))$.

6. Conclusions and future work

We study computing tight lower and upper bounds on the directed Hausdorff distance between two point set, when at least one of the sets has imprecision. We give efficient exact algorithms for computing the upper bound, prove that computing the lower bound is NP-hard in most settings, and provide approximation algorithms. Furthermore, we show that in one special case, our approximation algorithm is optimal. In other settings, a gap in the factor between the hardness result and approximation still remains. When both sets are imprecise, we do not have any constructive results for the lower bound.

All our results hold for the directed Hausdorff distance. A next step would be to extend them to the undirected Hausdorff distance. We can immediately solve the upper bound problem in that case using our results, since it is just the maximum of the two directed distances. However, computing lower bounds seems to be more complicated, because one needs to find a single placement of both point sets that minimises the distance in both directions at the same time.

Other directions of future work include looking at underlying metrics other than the Euclidean metric, similarity measures other than the Hausdorff distance, or, as is common in shape matching, allowing some transformation of the point sets.

Acknowledgements

This research was initiated during the Computational Geometry Workshop on Imprecise Data, which was held in December 2008 in Sydney and partly supported by NICTA. The authors thank the other participants of the workshop for helpful discussions and for providing a stimulating working environment, and also the anonymous reviewers of an earlier version of this paper for their suggestions and comments. NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program. M. Löffler was supported by The Netherlands Organisation for Scientific Research

(NWO) under the project GOGO, and by the US Office of Naval Research under grant N00014-08-1-1015. C. Knauer and M. Scherfenberg were supported by the German Research Foundation (DFG), grant AL 253/5-2.

References

[1] H.-K. Ahn, C. Knauer, M. Scherfenberg, L. Schlipf, A. Vigneron, Computing the discrete Fréchet distance with imprecise input, in: O. Cheong, K.-Y. Chwa, K. Park (Eds.), Algorithms and Computation, in: Lecture Notes in Computer Science, vol. 6507, Springer, Berlin, Heidelberg, 2010, pp. 422–433.
[2] H. Alt, B. Behrends, J. Blömer, Approximate matching of polygonal shapes, Annals of Mathematics and Artificial Intelligence 13 (3–4) (1995) 251–265.
[3] H. Alt, L. Guibas, Discrete Geometric Shapes: Matching, Interpolation, and Approximation — A Survey, in: Handbook on Computational Geometry, 1995, pp. 251–265.
[4] T. Feder, D. Greene, Optimal algorithms for approximate clustering, in: Proceedings of the twentieth Annual ACM Symposium on Theory of Computing, STOC'88, ACM, New York, NY, USA, 1988, pp. 434–444.
[5] S.J. Fortune, A sweepline algorithm for Voronoi diagrams, Algorithmica 2 (1987) 153–174.
[6] M.T. Goodrich, J. Snoeyink, Stabbing parallel segments with a convex polygon, Computer Vision, Graphics, and Image Processing 49 (2) (1990) 152–170.
[7] L.J. Guibas, D. Salesin, J. Stolfi, Epsilon geometry: building robust algorithms from imprecise computations, in: Proceedings of the Fifth Annual Symposium on Computational Geometry, SCG'89, ACM, New York, NY, USA, 1989, pp. 208–217.
[8] J.E. Hopcroft, R.M. Karp, An $n^{\frac{5}{2}}$ algorithm for maximum matching in bipartite graphs, SIAM Journal on Computing 2 (4) (1973) 225–231.
[9] D. Lichtenstein, Planar formulae and their uses, SIAM Journal on Computing 11 (2) (1982) 329–343.
[10] M. Löffler, M. van Kreveld, Largest bounding box, smallest diameter, and related problems on imprecise points, Computational Geometry: Theory and Applications 43 (2010) 419–433.
[11] A. Mukhopadhyay, E. Greene, S.V. Rao, On intersecting a set of isothetic line segments with a convex polygon of minimum area, International Journal of Computational Geometry and Applications 19 (6) (2009) 557–577.
[12] A. Mukhopadhyay, C. Kumar, E. Greene, B. Bhattacharya, On intersecting a set of parallel line segments with a convex polygon of minimum area, Information Processing Letters 105 (2) (2008) 58–64.
[13] T. Nagai, N. Tokura, Tight error bounds of geometric problems on convex objects with imprecise coordinates, in: Revised Papers from the Japanese Conference on Discrete and Computational Geometry, JCDCG'00, Springer-Verlag, London, UK, 2001, pp. 252–263.

Please cite this article in press as: C. Knauer, et al., The directed Hausdorff distance between imprecise point sets, Theoretical Computer Science (2011), doi:10.1016/j.tcs.2011.01.039