Convex Transversals

Esther M. Arkin* Claudia Dieckmann† Christian Knauer‡
Joseph S. B. Mitchell* Valentin Polishchuk§ Lena Schlipf† Shang Yang¶

Abstract

We answer the question initially posed by Arik Tamir at the Fourth NYU Computational Geometry Day (March, 1987): "Given a collection of compact sets, can one decide in polynomial time whether there exists a convex body whose boundary intersects every set in the collection?"

We prove that when the sets are segments in the plane, deciding existence of the convex stabber is NP-hard. The problem remains NP-hard if the sets are scaled copies of a convex polygon. We also show that in 3D the stabbing problem is hard when the sets are balls. On the positive side, we give a polynomial-time algorithm to find a convex transversal of a maximum number of pairwise-disjoint segments (or convex polygons) in 2D if the vertices of the transversal are restricted to a given set of points.

We also consider stabbing with vertices of a regular polygon – a problem closely related to approximate symmetry detection: Given a set of disks in the plane, is it possible to find a point per disk so that the points are vertices of a regular polygon? We show that the problem can be solved in polynomial time, and give an algorithm for an optimization version of the problem.

1 Introduction

Let S be a finite set of line segments in the plane. We say that S is *stabbable* if there exists a convex polygon whose boundary $\mathcal{C}$ intersects every segment in S; the closed convex chain $\mathcal{C}$ is then called a (convex) *transversal* or *stabber* of S.

Research on transversals is an old and rich area. Most of the work, however, has focused on *line* transversals, i.e., on determining properties of families of *lines* that stab sets of various types of geometric objects. Stabbing has attracted interest from various perspectives: purely combinatorial (complexity of the set of transversals, orders induced by stabbers), algorithmic (computing the stabbers), and applied (using transversals in curve reconstruction, line simplification, graphics, motion planning) – see [12] and references therein. In some of these applications it is natural to consider convex transversals as generalizations of line transversals. For example, it may be of interest to establish whether the data collected in an experiment is consistent with the assumption that the measured value is a convex function of the input. In this case, instead of fitting a linear function into the noisy measurements (a classical regression problem; see, e.g., [1] for an application in wireless sensor networks), one has to fit a convex

*Department of Applied Mathematics and Statistics, Stony Brook University, USA. {estie,jsbm}@ams.stonybrook.edu

†Institute of Computer Science, Freie Universität Berlin, Germany. {dieck,schlipf}@mi.fu-berlin.de

‡Institute of Computer Science, Universität Bayreuth, Germany. christian.knauer@uni-bayreuth.de

§Department of Computer Science, University of Helsinki, Finland. polishch@helsinki.fi

¶Mathworks, USA. yangshang1985@gmail.com

function, i.e., to solve a non-parametric convex regression problem (see, e.g., [5, 9] for recent work on convex regression).

The problem of computing a convex transversal for line segments was posed in 1987 [14]. For the case of stabbing *vertical* line segments, an optimal algorithm for the problem was presented by Goodrich and Snoeyink in [8]. They stated the problem of finding a convex stabber for a set of *arbitrary* segments in the plane as open. To the best of our knowledge, there has been no progress on the problem in the roughly 20 years since then.

Note that we address the *decision* problem – can a given set of objects be stabbed by the *boundary* of a convex polygon? A different line of work (see [6] for the recent results) considers an optimization version – find a *minimum-perimeter* polygon whose boundary *or interior* intersects a given set of segments.

Contributions

We prove that finding a convex transversal for a set of segments in the plane is NP-hard; the problem remains NP-hard for a set of scaled copies of a given convex polygon. We also show that in 3D, it is NP-hard to decide stabbability of a set of balls.

We then turn to positive results: Section 3 presents a dynamic program (DP) to decide if a set of *pairwise-disjoint* segments is stabbable by a stabber whose vertices are a subset of a given candidate set of points; if the segments are not stabbable, we can output a convex stabber that intersects the maximum number of segments. The algorithm readily generalizes to the case of disjoint convex polygons. (In an earlier version of the paper (see, e.g., [3]) we erroneously claimed that there always exists a stabber with edges supported by bitangents between elements of S. We also claimed that our algorithm extends directly to the case of convex pseudodisks; however, the details of that extension are not straightforward and will be the topic of a future follow-on paper.)

We also consider the *approximate symmetry detection* problem: Given a set of n disks in the plane and an integer i, is it possible to find one point per disk such that the points form a set invariant under rotations by $2\pi/i$? For general i, the problem is NP-hard [11]; in Section 4 we give a polynomial-time algorithm for the case $i = n$. That is, we answer the question: is it possible to find one point per disk such that the points are vertices of a regular polygon? We also consider an optimization variant of the problem: Given a set of points in the plane, find the minimum δ^* such that shifting each point by at most δ^* brings the points into a symmetric position.

Closed stabbers vs. Terrains The stabbing problem formulation is isotropic in the sense that it does not single out any specific direction in the space. In function approximation and statistics applications (unlike in surface reconstruction), it is often the case that the transversal represents the graph of a function. That is, the stabber is a *terrain* – a surface that intersects every vertical line in at most one point. A *convex* terrain is a part of the boundary of a convex polygon (polytope in 3D).

Finding a convex terrain stabber is a special case of finding a convex stabber – to see this, just place one point far below the input (Figure 1). Our results, both positive and negative, are as strong as possible with respect to the distinction between convex terrain and convex stabbers: Our DP allows one to find even a convex stabber (and, hence, also to find a convex terrain stabber); our negative results show that it is hard already to find a convex terrain (and, hence, it is also hard to find a closed convex stabber).

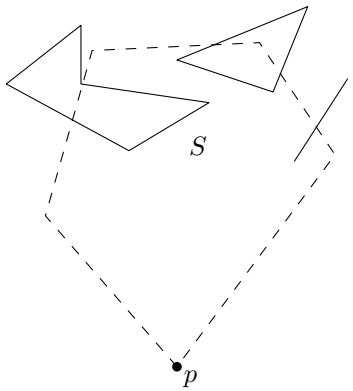

Figure 1: S' is S augmented with a point p. S can be stabbed by a convex terrain if and only if S' has a convex stabber. Thus, any algorithm that finds a stabber can also find a terrain. Conversely, if finding a terrain is hard, finding a stabber is also hard.

2 Hardness results

This section gives an answer to the question from [8,14] by showing that the problem is NP-hard.

2.1 Stabbing segments in the plane is NP-hard

Our reduction is from 3SAT. The reduction has the same spirit as the one used to show hardness of finding the largest-area convex hull of a set of points that are restricted to lie on line segments [13]. As in [13], we "thread" variable and clause gadgets on a convex chain, and connect clauses to their literals with segments. The details of our construction (the gadgets themselves) are different from [13].

Our reduction is shown in Figures 2 and 3. We use n and m to denote the number of the 3SAT variables and clauses, respectively.

Variable gadget For each variable we have a gadget that consists of three points (segments of zero length) and one segment. There are two ways to traverse the gadget (shown with dotted and dashed paths) that differ in the order in which the middle point and the segment are visited. The two ways correspond to setting the variable True or False. The important property of the gadget is that it will be possible to place a certain "connecting" segment in either of the two ways: so that it touches only the False subpath but not the True, and vice versa.

"Squashing" We make the variable gadget "thin" by moving all three points close to the supporting line of the segment, and, in addition, by moving the non-middle points far apart.

Variable chain Variable gadgets are placed along a convex chain, called the *variable chain*. The chain is almost vertical, bending to the right only slightly. The variable gadgets are "clenched" onto the chain, and the distance between consecutive gadgets is large. Thus, the only way to traverse the gadgets with a convex terrain is to visit them one by one, in the order as they appear along the chain, assigning truth values to the variables in turn in each gadget.

Clause gadgets The clause gadgets are similarly arranged, one after one, on another almost vertical convex chain, slightly bending to the left; this clause chain is placed to the right of the variable chain. Each clause gadget consists of 2 points and a segment; the only way to traverse

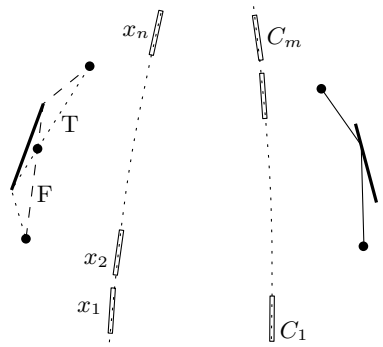

Figure 2: The variable gadget and the 2 ways to traverse it. The variable gadgets are threaded onto a convex chain; similarly, the clause gadgets are threaded. The chains (dotted) are not parts of the construction and are shown only for reference. The clause gadget can be traversed in only one way.

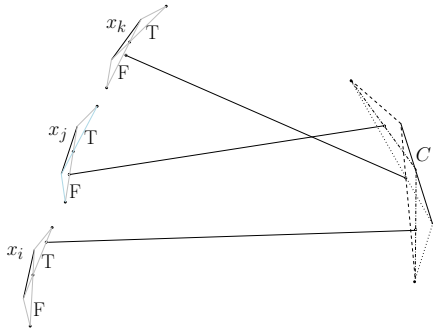

Figure 3: A clause $C = x_i \vee \bar{x_j} \vee \bar{x_k}$: three paths are shown that pick different subsets of the three connecting segments. The gadgets and their locations are not to scale: the gadgets are thinner, so that the points are very close to the supporting line of the segment – this makes the turn angles of the paths close to π; also, consecutive gadgets along each chain are separated so that a convex terrain can make independent choices in each of them.

the gadget is to visit the first point, then the segment and then the second point – the only flexibility is where to touch the segment.

Connectors We now place $3m$ more segments, connecting a variable gadget to a clause gadget whenever the variable appears in the clause (Figure 3). The placement of the segments' endpoints within variable gadgets is as follows: if the variable appears unnegated, the segment touches the True path through the gadget and does not intersect the False subpath; on the contrary, if the variable appears negated, the segment touches only the False subpath. In every clause gadget, segments' endpoints look the same – see Figure 3; as can be easily checked, a convex terrain can intersect any two of the segments, but not all three. This finishes the construction.

The reduction If the 3SAT instance is feasible, the stabber can traverse the variable gadgets according to the satisfying truth assignment. In each of the clauses, at least one of the connecting segments (the one connecting to the satisfying variable) can be omitted; the other two are picked up by one of the three paths.

Conversely, if there exists a stabber, it must omit (at least) one connecting segment per clause. Set the variable True or False depending on whether the omitted segment connects from a True or False part of the variable gadget; this satisfies all the clauses. The True/False setting is consistent because any segment omitted by the stabber in the clause gadget must

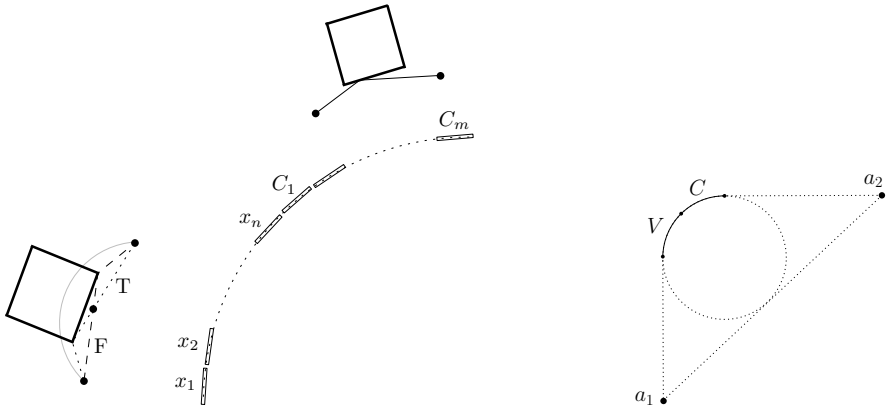

Figure 4: Left: Variable and clause gadgets (not to scale: each variable gadget is fit into the circular arc of length $1/(8n)$ and each clause gadget is fit into the circular arc of length $1/(8m)$; also consecutive gadgets are separated so that a convex terrain can make independent choices in each gadget). Right: V and C mark placements for variable and clause gadgets; the points a_1, a_2 ensure that the connector squares can either be intersected at a variable gadget or a clause gadget but nowhere else.

have been stabbed in the variable gadget, and there either only the True-subpath or only the False-subpath segments could have been stabbed, but not both.

We thus have our main negative result:

Theorem 2.1. *Finding a convex (terrain) transversal for a set of segments in the plane is NP-hard.*

In the remainder of this section we modify our proof to show hardness of stabbing scaled copies of a convex polygon (Section 2.2) and hardness of stabbing balls in 3D (Section 2.3).

2.2 Stabbing squares and scaled copies of a convex polygon

To show hardness of stabbing squares we again reduce from 3SAT. The construction (Figure 4) is very similar to the one for segments.

Variable gadgets The variable gadget consists of three points (squares of zero area) and a square. The gadget is nearly the same as in the construction for segments, but instead of a segment we use a square (the same holds for the clause gadgets). There are two ways to traverse a gadget; one corresponds to setting the variable True and the other to setting the variable False.

"Fitting" We fit the variable gadget into a circular arc by putting the two non-middle points on the arc. The middle point and the lower edge of the square (the edge that is closest to the three points) lie inside the circular arc, see Figure 4. Each variable gadget is fit into an arc of $1/(8n)$ of a unit circle.

Variable arc The variable gadgets are placed next to each other on an arc of one eighth of a unit circle. We call this arc the *variable arc*. The only way to traverse the variable gadgets with a convex terrain is to visit them one by one, in the order they appear on the arc, assigning truth values in turn in each gadget.

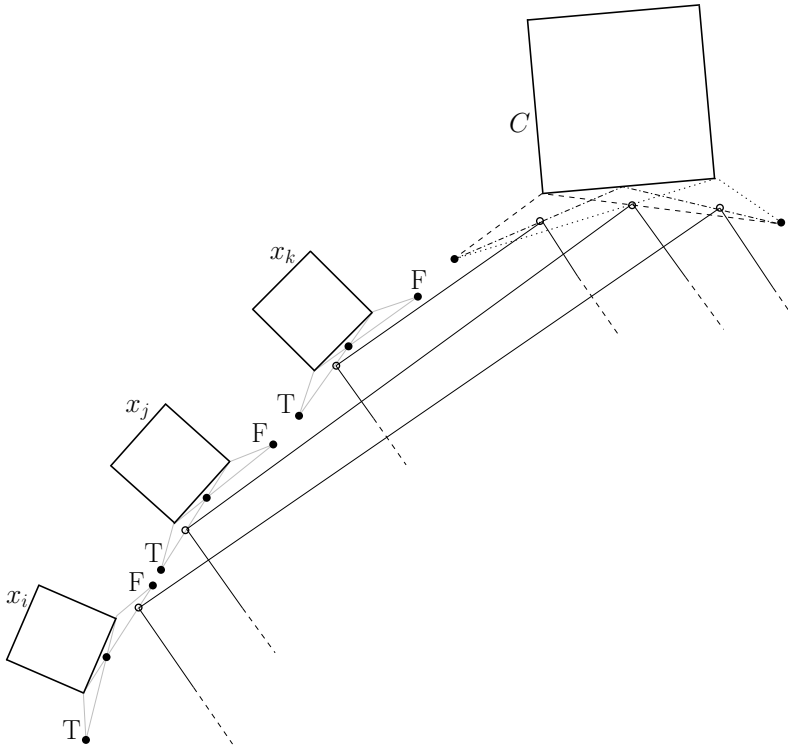

Figure 5: Clause $C = x_i \vee \bar{x_j} \vee \bar{x_k}$. Three path are marked that pick up different subsets of the three connecting squares.

Clause gadgets The clause gadgets are placed in the same way as the variable gadgets, on an arc of one eighth of the unit circle and next to the variable arc. Each clause gadget consists of two points and a square.

Connectors We place $3m$ more squares, connecting a variable gadget to a clause gadget whenever the variable appears in the clause (Figure 5). One edge of each square is placed exactly in the same way as the connector segment in the construction for line segments. This means that one endpoint of the edge lies within the variable gadget as follows: if the variable appears unnegated, the edge touches the True subpath through the gadget and does not intersect the False subpath; on the contrary, if the variable appears negated, the edge touches only the False subpath. In every clause gadget, the endpoints of these edges look the same.

To ensure that a convex terrain can intersect connecting squares only near the gadgets, we add two points to the construction (Figure 4, right); a convex terrain that traverses these points and all gadgets cannot intersect the unit circle (on which the gadgets are placed) except at the gadgets. Thus, the connectors can be intersected only at endpoints of the edges that are placed in the same way as the connector segments in the construction for line segments. (Note that for the latter property to hold, it was crucial to fit all variable and clause gadgets on the quarter of a unit circle – this way all connector squares lie inside the unit circle.)

The reduction Assume that the 3SAT formula is feasible. Then the stabber can traverse the variable gadgets according to the satisfying assignment. In each clause gadget one of the three connecting squares has to be omitted by the stabber; let this be the one connecting to the satisfying variable.

On the other hand, if there exists a stabber, it must omit at least one connecting square

per clause. Set the variables True or False depending on whether the omitted square connects from a True or False path of the variable gadget; this satisfies all the clauses. This setting is consistent since any square omitted by the stabber in the clause gadget has to be stabbed in the variable gadget and there either only the True-subpath or the False-subpath squares could have been traversed, but not both.

Theorem 2.2. *Finding a convex (terrain) transversal for a set of squares in the plane is NP-hard.*

Generalization The above proof can be adapted to show that stabbing regular k-gons is NP-hard for any $k > 2$: just replace the squares with the k-gons, and (to ensure again that the connectors lie inside the unit circle) place the variable and clause gadgets on an arc of $1/(2k)$ of a unit circle. That is, fit each variable gadget into an arc of $1/(4kn)$, and each clause gadget into an arc of $1/(4km)$.

Theorem 2.3. *For arbitrary constant $k > 2$, finding a convex (terrain) transversal for a set of regular k-gons in the plane is NP-hard.*

It is not crucial that the polygons are regular, as only one edge of each polygon is important for the construction. Hence, the construction works in the same way if we consider a set of scaled copied of a given polygon instead of regular polygons.

Theorem 2.4. *Finding a convex (terrain) transversal for a set of scaled copies of a given convex polygon in the plane is NP-hard.*

An interesting open question is whether stabbing disks is NP-hard. Our reduction above does not extend to this case – the reason is that we want the connectors to lie inside the unit disk onto which the variable and clause gadgets are threaded (this is needed to ensure that the connectors cannot be stabbed outside the unit circle – the only places to stab them are near the gadgets).

2.3 Stabbing balls in 3D is NP-hard

We again reduce from 3SAT, employing similar ideas as those for segments in 2D.

2.3.1 Variables

Variable gadget The basic variable gadget consists of three points a, b, c (balls of 0 radius) and one ball B of large radius whose center is denoted by d (Figure 6, left). The three points and the center of the ball all belong to a horizontal plane, which we call the *supporting plane* of the gadget.

True and False touches The cross-section of B by the supporting plane is a disk. The points t, f where the tangents from a and b touch the disk are called the True and the False *touches*.

"Squashing" As with the segments in 2D, we make the dashed and dotted paths (see Figure 6, left) have the turn angles close to π. For that, we make the three points a, b, c almost collinear, moving a and b far apart, and using a large radius for the ball B.

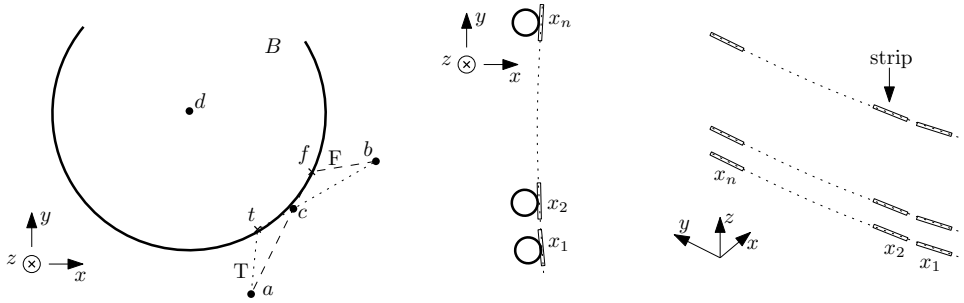

Figure 6: Left: The (cross-section by the supporting plane of the) variable gadget. The gadget is not to scale; actually, the turn angles of both the dashed and the dotted paths are close to π. Middle: The variable gadgets are threaded onto a convex chain; the chain (dotted) is not a part of the construction and is shown only for reference. Right: The variable grid.

Variable chain Also as with segments in 2D, the variable gadgets are placed along an almost "flat" convex chain (Figure 6, middle). Again, the gadgets are "clenched" onto the chain so that the three points of every gadget are very close to the chain. All gadgets and the chain are aligned, in that the supporting planes of all gadgets coincide, and the chain also lives in this common horizontal plane. We thus also call the plane the *supporting plane* of the chain. The balls are "sticking out" of the chain, i.e., the centers of the balls are placed outside the convex hull of the chain.

As with segments in 2D, consecutive gadgets along the chain are separated by large enough distance so that the cross-section of the stabber by the supporting plane must visit the gadgets one by one, assigning truth values to the variables in turn in each gadget. We call this whole construction—the gadgets threaded on the chain—the *variable chain*.

Variable grid We place $m+2$ copies of the variable chain, one copy directly above another (Figure 6, right). We number the copies from 0 to $m+1$. The first and the last copies are "dummy"; we have them only to enforce consistency of the "choices" that the stabber must make in each of the chains (see below). The other copies correspond to the clauses.

We call the $m+2$ gadgets corresponding to variable x_i in all $m+2$ chains the i-th *variable strip*. This way, our construction so far is a "grid" of n strips $\times$ $m+2$ chains (see also Figure 8 below).

Consistency In Section 2.3.4 we argue that any convex terrain must make the same "choices" at each of the m copies of the variable in a strip. That is, for all $j = 1, \ldots, m$, in the cross-section by the supporting plane of the jth chain, the stabber either uses the True touch or uses the False touch.

2.3.2 Clauses

Clause chains Place $2(2m+2)$ points (0-radius balls) on two identical parallel almost vertical convex chains P, Q, slightly bending to the left, lying in planes that are perpendicular to the y-axis (Figure 7, left). More specifically, the first two points p_0, p_1^- of P are the endpoints of a segment parallel to the z-axis. The next two points, p_1^+, p_2^- are the endpoints of a segment making a slightly larger than 0 angle with the z-axis. The next two points, p_2^+, p_3^- are endpoints of a segment making even larger angle with the z-axis, and so on: the segments $p_j^+ p_{j+1}^-$ make larger and larger angles with the z-axis as $j = 1, \ldots, m-1$ increases. The last two points of P are p_m^+, p_{m+1}. That is, P has 2 points per clause, plus the points p_0, p_{m+1}.

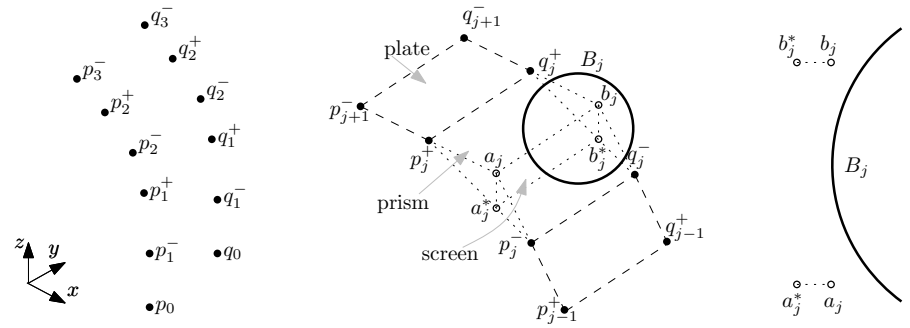

Figure 7: Left: The chains P, Q. Middle: The plates (dashed), and the prism and the screen (dotted). The figures are not to scale; actually, the chains are almost vertical (and the supporting planes of screens are almost horizontal). Right: The cross-section of the gadget for clause j by the supporting plane of the jth screen.

The chain Q is a parallel shift of P in the y direction. The points belonging to Q are analogously numbered $q_0, q_1^-, \ldots, q_m^+, q_{m+1}$. We place P and Q to the right of the variable grid (see Figure 8, left).

Plates Because P and Q are parallel to each other, the quadruple of points p_0, p_1^-, q_1^-, q_0 lie in the same (vertical) plane parallel to the y-axis; we call the rectangle $p_0 p_1^- q_1^- q_0$ the *0th plate*. Similarly, $p_1^+, p_2^-, q_2^-, q_1^+$ lie in the same plane (making a positive angle with the z-axis) also parallel to the y-axis; the rectangle $p_1^+ p_2^- q_2^- q_1^+$ is the *1st plate*. In general, points $p_{j-1}^+, p_j^-, q_j^-, q_{j-1}^+$ lie in a plane parallel to the y-axis, making larger angle with the z-axis for larger j; the rectangle $p_{j-1}^+ p_j^- q_j^- q_{j-1}^+$ is the *jth plate* (Figure 7, middle).

Clause gadgets By construction, the plates must be inside the convex hull of the stabber (including the possibility of some plates being part of the stabber itself). Extend the plates by sliding out the sides $p_j^+ q_j^+, p_j^- q_j^-$ until the sides intersect along a segment $a_j b_j$. The points $p_j^+, q_j^+, p_j^-, q_j^-, a_j, b_j$ define a right triangular prism (with bases perpendicular to the y-axis), which we call the *jth prism*. (Note that the triangles in the prism base are not right; it is the prism that is right.)

The *jth screen* is the rectangle $a_j b_j b_j^* a_j^*$ where $a_j a_j^*, b_j b_j^*$ are altitudes of the triangles $p_j^- p_j^+ a_j$, $q_j^- q_j^+ b_j$, respectively. The gadget for the clause j is a ball B_j with the center in the supporting plane of the screen. The ball intersects $a_j b_j$ but does not intersect $a_j^* b_j^*$ (Figure 7, right). The exact placement of B_j depends on which variables constitute the clause j, as detailed below in Section 2.3.5.

2.3.3 The reduction

The points and balls described so far can be stabbed by a convex (surface) stabber: the stabber can traverse the variable grid making arbitrary (but consistent, across the chains) truth assignments for the variables, and then turn onto the P, Q-side where the stabber can use plates and prisms. We now place $3m$ more balls, connecting a variable gadget to a clause gadget whenever the variable appears in the clause; this turns our instance into one that has a stabber if and only if there exists a satisfying assignment in the 3SAT.

Tall, narrow and deep First of all, we align the variable grid and clause chains P, Q so that each pair (jth chain, jth clause gadget) lives in its own horizontal slab, called the *jth slab*. We

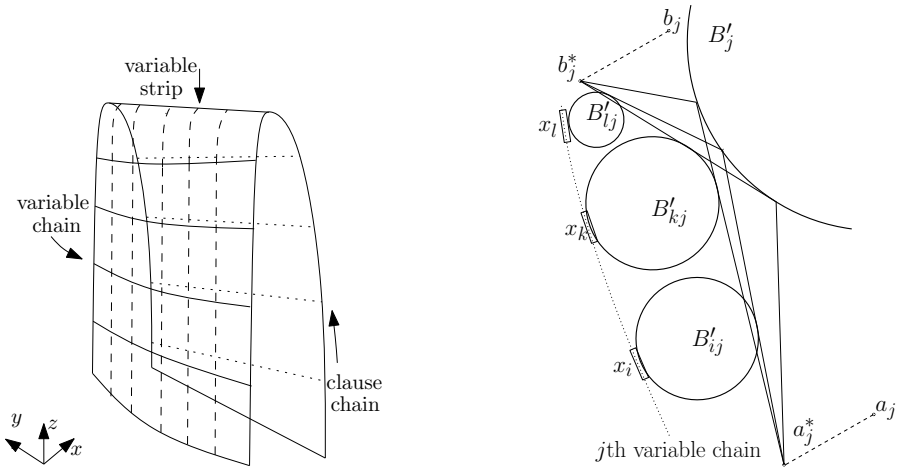

Figure 8: Left: The "frame" of the construction. Right: The cross-section by the supporting plane of the jth screen.

fine-tune the angles of P, Q so that the supporting plane of the jth screen almost coincides with the supporting plane of the jth variable chain (they cannot coincide fully because the supporting plane of the variable chain is horizontal, while that of the screen is not). Next, we make the whole construction "tall" and "narrow" so that the distance between the jth chain and the jth clause gadget is smaller than the height of the jth slab. We also make the construction "deep" in the y direction: the y-span of a variable chain, as well as the distance between P and Q, is large in comparison to the distance between the jth variable chain and the jth clause gadget, for any j. Refer to Figure 8, left.

Connectors Suppose now that the jth clause contains variables $x_i, x_k, x_l, i < k < l$. We place three *connecting balls* B_{ij}, B_{kj}, B_{lj} with the centers lying in the supporting plane of the jth screen (Figure 8, right). The balls are placed opposite the variable gadgets for x_i, x_k, x_l in the jth variable chain, and each ball spans the space inside the construction between the jth variable chain and the jth clause gadget (the height, the narrowness and the depth of the construction allows us to place the balls so that they are disjoint from the analogous balls in the other slabs).

Similarly to the case of segments in 2D, if x_i is unnegated in clause j, the ball B_{ij} touches the True (dotted, in Figure 6, left) path through the x_i's gadget in the jth variable chain and does not touch the False (dashed) path. Otherwise, the ball touches the False path and not the True path. The placement of B_{kj}, B_{lj} is analogous. The interaction of the balls with the jth clause gadget is also similar to the segments in 2D: we place the balls so that there exists a convex terrain intersecting any two of the three balls, but not all three (Figure 8, right). Section 2.3.5 details how to do this.

Correctness If the 3SAT instance is feasible, then the stabber can traverse the variables gadgets according to the satisfying truth assignment. In each of the clauses, the ball connecting to the satisfying variable is omitted by the stabber; the other two are picked up in the clause gadget.

Conversely, if there exists a stabber, it must consistently traverse the variable gadgets in the variable grid setting the truth assignment. On the clauses side, the stabber must contain (inside its convex hull) all plates. "In between" the plates (i.e., inside the prisms) the stabber is free to do whatever it likes; however, no matter how it goes it will not be able to stab more

than two connecting balls per clause. The unstabbed ball satisfies the clause; the consistency of the satisfying assignment follows from the fact that a variable cannot be set both to True and to False by the same stabber.

Precision We were informal in saying that parts of the construction are "large" enough, "far" enough, etc. Still, the equations and inequalities involving the coordinates of the points in the gadgets have polynomial-size coefficients. E.g., a variable (resp. clause) chain can be part of the boundary of the regular $O(n)$-gon (resp. $O(m)$-gon). Thus, the construction can be done so that it has the required properties and the coordinates specifying positions of the parts of the gadgets are polynomial in n and m.

Overall, we have:

Theorem 2.5. *Finding a convex (terrain) transversal for a set of balls in 3D is NP-hard.*

2.3.4 Consistency of choices in a variable strip

Let $\mathcal{T}$ be a convex terrain stabber, and consider a fixed i. We claim that the cross-section of $\mathcal{T}$ by the supporting plane of the jth chain looks the same for all $j = 1, \dots, m$ in the vicinity of the variable gadget for x_i: the stabber either uses the True touch or uses the False touch of the gadget. The proof is based on the following straightforward observations:

Lemma 2.6. *Let $\mathcal{C}$ be the convex hull of $\mathcal{T}$. Consider any basic variable gadget for x_i (Figure 6, left). We have:*

i. Either the True or the False touch belongs to $\mathcal{C}$, but not both.

ii. No point of the segment cd other than c belongs to $\mathcal{C}$; i.e., $cd \cap \mathcal{C} = c$.

Say that the stabber makes a *switch* if it sets x_i True in the jth variable chain but sets x_i False in the $j+1$st chain, or vice versa, for some $j = 1, \dots, m$. Consider the two cases:

There is more than one switch. Without loss of generality suppose that x_i is set to True in chains j_-, j_+ and to False in a chain j, for $j_- < j < j_+$. Let t_-, t, t_+ be True touches in x_i's gadget in the chains j_-, j_+; let f be the False touch in the chain j (Figure 9, left). We know that $t_-, t_+, f \in \mathcal{C}$. The True touches of x_i's gadgets in all chains lie on a common line, i.e., t is a point on the segment t_-t_+. Thus, since $\mathcal{C}$ is convex, $t \in \mathcal{C}$. This, together with $f \in \mathcal{C}$ contradicts Lemma 2.6i.

There is exactly one switch. Without loss of generality suppose that x_i is set to True in a chain j and to False in the chain $j+1$. Since $j \geq 1$, there exists chain $j-1$. If x_i is set to False in it, then there is more than one switch. Otherwise, let t_- be the True touch in $j-1$st chain and let f^+ be the False touch in $j+1$st chain (Figure 9, right); let h be the intersection of the segment t_-f_+ with the supporting plane of the jth chain. By symmetry, $h \in cd$ where c and d are the middle point and the center of the large ball in the jth gadget for x_i (refer to Figure 6, left). Since $t_-, f_+ \in \mathcal{C}$ and $\mathcal{C}$ is convex, $h \in \mathcal{C}$. This, together with $h \in cd, h \neq c$ contradicts Lemma 2.6ii.

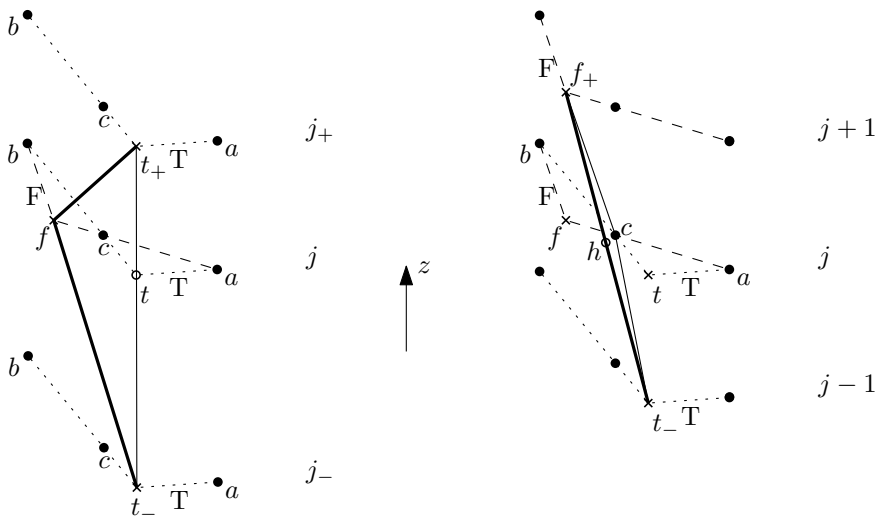

Figure 9: Left: If f is in the stabber, then t is not; however if t_-, t_+ are in, then t must be in too. Right: If t_-, f_+ are in the stabber, then a point $h \neq c$ of the segment cd is in the stabber.

2.3.5 Two, but not three, connecting balls can be stabbed in a clause gadget

We show how to place the three connecting balls and the ball of the jth clause gadget so that any two balls can be stabbed by a convex terrain, but all three cannot.

At most two balls are stabbed. First of all, each of the segments $a_j a_j^*, b_j b_j^*$ (sides of the jth screen) must have a point of the stabber in it. Denote by $B'_{ij}, B'_{kj}, B'_{lj}, B'_j$ the cross-sections of the balls $B_{ij}, B_{kj}, B_{lj}, B_j$ by the supporting plane of the jth screen (Figure 10). Let $a_j^* a_k, b_j^* b_k$ be the rays from a_j^*, b_j^* tangent to B'_{kj}. Choose the radius of B'_{kj} so that the intersection point of the tangents is close to $a_j b_j$ (but is still inside the screen rectangle $a_j b_j b_j^* a_j^*$).

Increase the radius of B_{ij} from 0 just past the value at which B'_{ij} is tangent to $a_j^* a_k$ (i.e., the ray $a_j^* a_k$ cuts off a positive-area cup from the disk B'_{ij}). Choose the radius of B'_{lj} similarly. Now draw the tangent $a_j^* a_i$ from a_j^* to B'_{ij} and the tangent $b_j^* b_l$ from b_j^* to B'_{lj}; let r_k be the intersection of the tangents. We place the ball B_j so that B'_j goes through r_k and is tangent to the rays $a_j^* a_k, b_j^* b_k$ (the tangency points are denoted r_i, r_l). It easy to see that no convex terrain can stab all 3 balls B_{ij}, B_{kj}, B_{lj} provided it stabs B_j.

Stabbing two balls. On the other hand, any two of the balls can be intersected by a convex stabber. For that, one can use a *barn roof* (Figure 11) which is a construction, with 4 faces, fully lying inside the jth prism (so as not to break the overall convexity of the stabber). The two opposite faces of the roof are congruent triangles $p_j^- q_j^- s_j, p_j^+ q_j^- t_j$ attached to the adjacent plates coplanarly with the plates. The segment $s_j t_j$ is parallel to $p_j^- p_j^+$ (and $q_j^- q_j^+$). The segment can be shifted between $p_j^- p_j^+$ and $q_j^- q_j^+$ arbitrarily so that the cross-section of the roof by the supporting plane of the jth screen looks like either of the 3 paths a_j^*-r_i-b_j^*, a_j^*-r_k-b_j^*, a_j^*-r_l-b_j^* in Figure 10. That is, the roof can intersect any two of the balls B_{ij}, B_{kj}, B_{lj} in the clause gadget (Figure 11, right).

3 Stabbing disjoint segments

This section presents a dynamic program (DP) to decide stabbability of a set S of *pairwise-disjoint* segments in the plane by a convex stabber whose vertices are restricted to come from

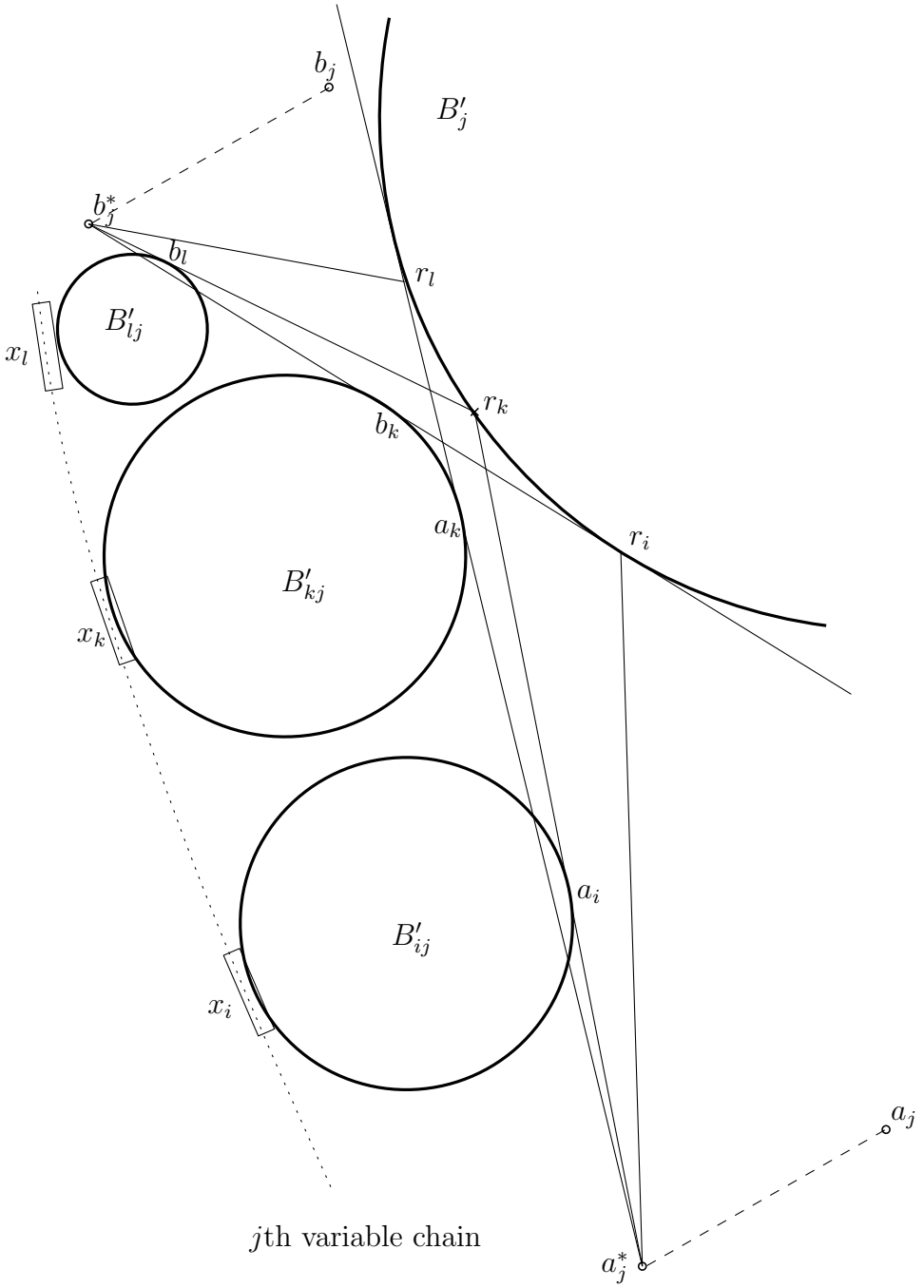

Figure 10: The cross-section by the supporting plane of the jth screen.

a given discrete set $C \subset \mathbb{R}^2$ of candidate points. A subproblem in the DP is specified by a pair of potential stabber edges together with a constant-complexity "bridge" between the edges (the bridge is either a single segment or a segment—visibility-edge—segment chain). The disjointness of the segments allows us to determine which segments must be stabbed within the subproblem. We show that a segment-free triangle can be found that separates a subproblem into smaller subproblems, which allows the DP to recurse.

Arcs and nodes, chords and bridges A straight-line segment between two points from C (i.e., a potential stabber edge) is called an *arc*. Two arcs pq, rt are *compatible* if either they

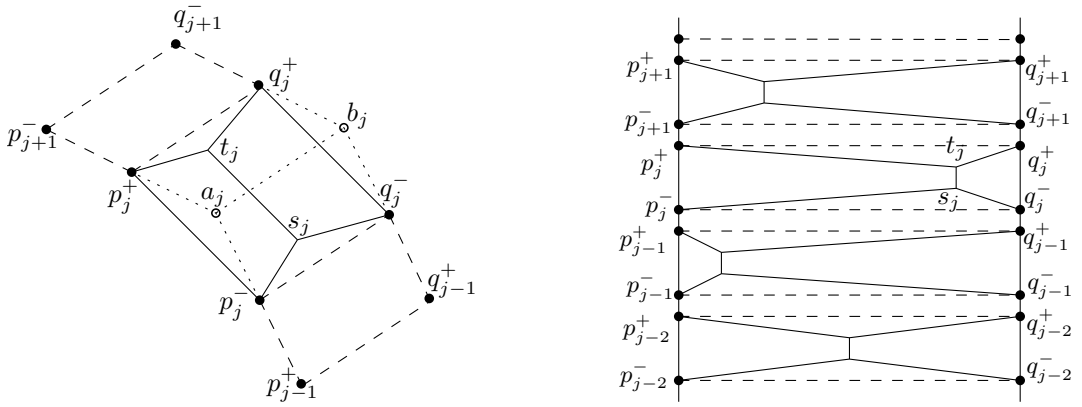

Figure 11: Left: The roof. s_jt_j is below a_jb_j. s_j (resp. t_j) lies in the plane of the plate $p^+_{j-1}p^-_j q^-_j q^+_{j-1}$ (resp. $p^-_{j+1}p^+_j q^+_j q^-_{j+1}$). Right: View of the clause-side of the stabber from a point at $+\infty$ on the x-axis; the stabber edges are bold, the plates boundaries are dashed. s_jt_j can be shifted freely to grab any two of the balls B_{ij}, B_{kj}, B_{lj} as in Figure 10; similarly, the ridges of the roofs can be shifted independently within each clause gadget.

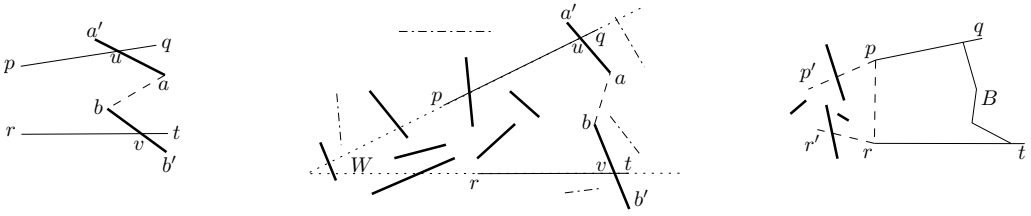

Figure 12: Left: $p, q, r, t \in C, aa', bb' \in S, u, v \in \mathcal{P}', p, q, r, t, a, a', b, b', u, v \in \mathcal{P}$. pq, rt are compatible arcs. $rp, ba, bu, av, ta, aq, tu, bp, rb, ra, ap$ are chords. $rp, vbau, vau, vbu, vbp$ are some of the bridges; rp is chordless. Middle: The wedge W (boundaries dashed) and the bridge $B = vbau$. The segments in $S_{pq,rt,B}$ that have to be stabbed to the left of B are solid; the segments in $S \setminus S_{pq,rt,B}$ are dash-dotted. Right: An empty subproblem (pq, rt, B) and an induced subproblem $(p'p, r'r, rp)$.

have a common endpoint or the supporting lines of the arcs intersect outside each of pq, rt. In other words, the points p, q, r, t are in convex position, and pq, rt have the potential to be sides of a convex polygon – the stabber. Refer to Figure 12, left.

Let $\mathcal{P}'$ denote the set of points at which arcs intersect segments from S. Let $\mathcal{P}$ be the union of $\mathcal{P}$', C, and endpoints of segments from S; call points in $\mathcal{P}$ *nodes*. A *chord* is a straight-line segment whose interior intersects no segment from S, and whose each endpoint is a node.

A *bridge* is a polygonal path with the following properties:

- Its endpoints are nodes.
- It has at most three links.
- (i) If it has exactly one link, then the link is either a chord or a part of a segment from S – in the latter case, the bridge is *chordless*; (ii) if it has exactly two links, then one of the links is a chord, and the other is a part of a segment; (iii) if it has exactly three links, then they are a part of a segment from a node to the segment endpoint, a chord, and a part of another segment from the segment endpoint to a node (that is, the chord connects endpoints of the two segments).

Subproblems A subproblem in our DP is specified by two compatible arcs and a bridge. More specifically, let $p, q, r, t \in C$ be the four candidate points forming compatible arcs pq, rt.

Without loss of generality let rt be below the line pq, and let q, p, r, t be the order in which the nodes appear counterclockwise on the convex hull of the arcs. We define the wedge W to be the region that is below the line supporting pq and above the line supporting rt. In addition to the two arcs, the subproblem has in the input a bridge B that connects some point of pq to some point of rt. Refer to Figure 12, middle.

Subproblem's responsibility The crucial observation that allows us to run the DP is the following: Assuming that the arcs pq, rt are part of the stabber, we know for each segment $s \in S$ whether it should be stabbed to the left or to the right of the bridge. Indeed, only those segments that have non-empty intersection with the wedge W can be stabbed. On the other hand, no segment can have points on both sides of the bridge – for that it would have to cross the bridge, and this is impossible: the chord is not crossed by definition, and no segment is crossed by another segment due to the assumption of pairwise-disjointness of segments in S.

Let $S_{pq,rt,B}$ denote the segments that must be stabbed to the left of the bridge B; i.e., the segments that intersect W in the part of the wedge that lies to the left of B.

The function Stab$(\cdot)$ Define a Boolean function Stab(pq, rt, B) to be True if the segments $S_{pq,rt,B}$ can be stabbed (assuming pq, rt is a part of the stabber), and to be False otherwise; for an incompatible pair of arcs pq, rt define Stab$(pq, rt, \cdot)$ to be always False. The function shows whether the stabber can be "completed" having pq, rt as its part.

In the remainder of this section we show how to evaluate the function on a subproblem given its values at other subproblems, i.e., how to solve the DP.

Empty subproblems The subproblem (pq, rt, B) is *empty* (Figure 12, right) if no segment from S penetrates the region of W that is to the left of the bridge but to the right of rp (this includes the possibility that the bridge is the segment rp itself). An empty subproblem is *closed* if $p = r$. Closed subproblems are at the lowest level of our DP: clearly, Stab(σ) = True for a closed subproblem σ.

Let (pq, rt, B) be an empty subproblem. We say that a subproblem $(p'p, r'r, rp)$ is an *induced* subproblem of (pq, rt, B) if pp' is below (the supporting line of) pq, and rr' is above rt. That is, the angles qpp' and trr' are convex, and thus both qpp' and trr' can potentially be parts of a convex chain – the stabber-to-be. Empty subproblems are easy to reduce to induced subproblems: Stab(pq, rt, B) = True for an empty subproblem (pq, rt, B) if and only if Stab$(p'p, r'r, B)$ is True for at least one subproblem induced by (pq, rt, B).

General subproblems Let $\mathcal{C}$ be the sought stabber that has pq, rt as two of the sides (Figure 13). (Of course, we do not know $\mathcal{C}$, but we will not use its existence in the algorithm, we will only use $\mathcal{C}$ to argue that we can split the subproblem into smaller ones.) Let $\mathcal{C}$' be the (convex) region bounded by $\mathcal{C}$, and let P be the part of $\mathcal{C}$' to the left of the bridge B (i.e., P is what is chopped off $\mathcal{C}$' by B). Consider the set $P' = P \setminus \bigcup_{s \in S_{pq,rt,B}} s$. That is, P' is P "pierced" by the segments $S_{pq,rt,B}$ that are stabbed in the subproblem (pq, rt, B).

Because $\mathcal{C}$ is a stabber, every segment in $S_{pq,rt,B}$ intersects the boundary of P. This means that P' is a (weakly) simple polygon (i.e., no segment makes a hole in P' by being fully contained in the interior of P'). Each vertex of P' belongs to one of the following 6 (overlapping) sets:

P_0**:** p, q, r, t

P_1**:** vertices of the bridge;

P_2**:** nodes that reside on the arcs pq, rt;

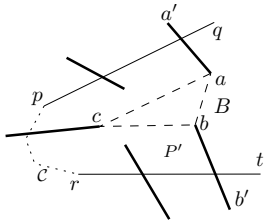

Figure 13: The (unknown) part of the stabber $\mathcal{C}$ is dotted. P' is the simple polygon bounded by the unknown part of $\mathcal{C}$, by pq, rt, by the bridge $B = tbaq$, and by the piercing segments. abc is a separating, i.e., segment-free triangle inside P'.

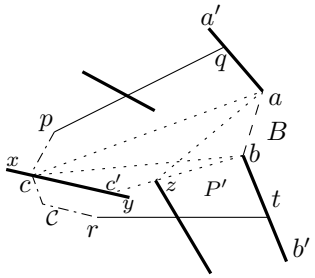

Figure 14: abc is a triangle in a triangulation of P'; move c inside P'. abz is the sought triangle.

P_3**:** nodes that belong to $\mathcal{C}$ except those in P_2;

P_4**:** endpoints of segments from $S_{pq,rt,B}$ that are stabbed by pq or rt;

P_5**:** endpoints of segments from $S_{pq,rt,B}$ that are stabbed by $\mathcal{C} \setminus pq, rt$.

Note that only P_3 is not known to us (because we do not know $\mathcal{C}$); all the other sets are known as soon as the subproblem (pq, rt, B) is specified.

We define the *important* link ba of the bridge B as follows: if B is chordless, then $ba = B$; otherwise ba is the chord of B. We assume that a is closer to pq, and b is closer to rt along B. Our algorithm will search for a separating, i.e., segment-free triangle abc within P' where c is a vertex of P' *and* $c \notin P_3 \cup P_0$. We first argue that such a triangle exists, and next describe what to do depending on the set, among P_1, P_2, P_4, P_5, to which c belongs.

Lemma 3.1. *There exists a vertex c of P' such that $c \notin P_3 \cup P_0$ and no segment intersects the interior of abc.*

Proof. The link ba is a side of P'; thus, any triangulation of P' has a triangle abc, with c being a vertex of P'. If there exists a triangulation such that $c \notin P_3 \cup P_0$, we are done. Otherwise, let xy be the segment that contains c; i.e., $c = xy \cap \mathcal{C}$ (Figure 14). Move c along xy inside P'. Either c reaches the endpoint of the segment (in which case we are done because $c \in P_5$) or one of the sides of abc, say, bc hits an endpoint z of a segment from $S_{pq,rt,B}$; let c' be the position of c on xy when this happens. The convex quadrilateral $cc'ba$ has no segments in the interior, and abz is the sought triangle. The case $c \in P_0$ is treated similarly. □

We emphasize that even though we used $\mathcal{C}$ in arguing the existence of the vertex as in the above lemma, we can find such a vertex without knowing $\mathcal{C}$ (e.g., just by trying all vertices in P_1, P_2, P_4, P_5).

Let $B = vbau$ be the bridge. We now show how our DP recurses into subproblems defined by the sides of the triangle abc (Figure 15):

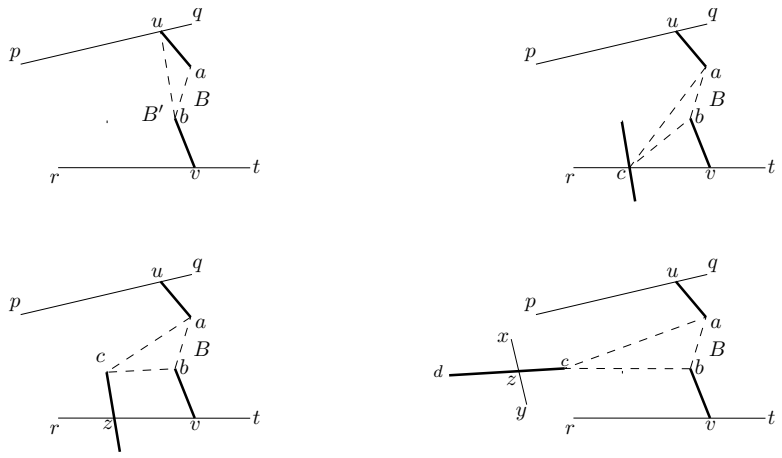

Figure 15: The DP recursion. Top: $c \in P_1, c \in P_2$. Bottom: $c \in P_4, c \in P_5$.

Case I: c is a vertex of the bridge; $c \in P_1$. Then the bridge has one fewer links, and $\text{Stab}(pq, rt, B) = \text{Stab}(pq, rt, B')$ where B' is the new bridge.

Case II: c is on pq, rt*;* $c \in P_2$. Without loss of generality suppose that $c \in rt$. If there exists a segment $s \in S_{pq,rt,B}$ that lies in the interior of the triangle vbc (i.e., s is not stabbed by tc), then s cannot be stabbed in the subproblem, and, hence, $\text{Stab}(pq, rt, B) = \text{False}$. Otherwise (i.e., if no segment intersects vbc or any segment that intersects tbc is already stabbed by rt), $\text{Stab}(pq, rt, B) = \text{Stab}(pq, rt, cau)$.

Case III: c is an endpoint of a segment from $S_{pq,rt,B}$ *stabbed by* pq, rt*;* $c \in P_4$. Without loss of generality suppose that c is the endpoint of a segment that is stabbed by rt; let z be the point of the stabbing. If there exists a segment $s \in S_{pq,rt,B}$ that lies in the interior of the quadrilateral $vbcz$ (i.e., s is not stabbed by tz), then s cannot be stabbed in the subproblem, and, hence, $\text{Stab}(pq, rt, B) = \text{False}$. Otherwise, $\text{Stab}(pq, rt, B) = \text{Stab}(pq, rt, zcau)$.

Case IV: c is an endpoint of a segment from $S_{pq,rt,B}$ *stabbed by* $\mathcal{C} \setminus pq, rt$*;* $c \in P_5$. Let d be the other endpoint of the segment touched by the triangle abc. Then $\text{Stab}(pq, rt, B) =$ True if and only if there exists an arc xy that intersects dc (say, at a point z) such that both $\text{Stab}(pq, xy, zcau)$ and $\text{Stab}(yx, rt, vbcz)$ are true. Formally,

$$\text{Stab}(pq, rt, B) = \bigvee_{\substack{\text{arc } xy: \\ dc \cap xy = z \neq \emptyset}} \left(\text{Stab}(pq, xy, zcau) \wedge \text{Stab}(yx, rt, vbcz) \right)$$

DP's complexity Let $m = |C|, n = |S|$ denote the number of the candidate stabber vertices and the number of segments resp. Every subproblem is defined by a pair of potential stabber edges ($O(m^4)$ pairs). In addition, the subproblem is defined either by a segment from S (n choices) or by a visibility edge between segments endpoints ($O(n^2)$ choices). Thus, the total number of subproblems is $O(m^4 n^2)$, which is the space requirement of our DP. The running time of the program is thus $O(m^8 n^4)$; it likely can be reduced, but such improvements are beyond the scope of the paper.

Extensions Our DP can be modified straightforwardly to find a convex stabber that stabs as many segments as possible. For that, we let the function $\text{Stab}(pq, rt, B)$ denote the number of elements of S stabbed by pq, rt plus the maximum number of other segments that can be stabbed in the subproblem (pq, rt, B). The recursions for the function change to reflect that

Stab(pr, qt, B) is the sum of the values of the function on the subproblems. Further, our DP extends immediately to solve the convex stabbing problem for disjoint convex polygons.

4 Stabbing with vertices of a regular polygon

In this section we present an algorithm to decide whether a given set of disks can be stabbed by a regular polygon. Specifically, the approximate symmetry detection problem is: Given a set of n disks in the plane and an integer i, is it possible to find one point per disk such that the points form a set invariant under rotations by $2\pi/i$? While the problem is NP-hard for general i [11], we solve the case $i = n$, i.e., we determine whether it is possible to find one point per disk so that the points are vertices of a regular n-gon.

4.1 The decision problem

Let $D = \{d_1, \ldots, d_n\}$ be the given disks. For points $p, c \in \mathbb{R}^2$ and integer $k \leq n$ let $\rho_c^k(p)$ denote the image of p after rotation around c by the angle $k2\pi/n$. For a pair of disks $d_i, d_j \in D$, let $A_{ij}^k = \{(p, c) | c \in \mathbb{R}^2, p \in d_i, \rho_c^k(p) \in d_j\} \subset \mathbb{R}^4$ be the set of all pairs (p, c) of points $p \in d_i, c \in \mathbb{R}^2$ such that p moves to d_j after rotating by $k2\pi/n$ around c; we call A_{ij}^k the *apex region*.

Fix a disk d_1. A regular n-gon with a vertex per disk of D exists if and only if there exist $p \in d_1$ and $c \in \mathbb{R}^2$ (the center of the n-gon) such that $\rho_c^j(p) \in d_{j+1}$ for $j = 1, \ldots, n-1$, or in other words, if and only if the intersection of $n-1$ apex regions A_{1j+1}^j is non-empty (here the vertices of the regular n-gon stab the disks in the order $d_1, d_2, \ldots$; of course this order is not known in advance). This prompts us to go through "all possible" intersections between the apex regions, checking for each of the intersections whether an n-gon exists.

Specifically, consider the $(n-1)^2$ apex regions $A_{1,j}^k, j = 2, \ldots, n, k = 1, \ldots, n-1$. Call a point $(p, c) \in \mathbb{R}^4$ *feasible* if it belongs to some $n-1$ of the regions, with each region being from a different disk with a different angle. Our problem has a feasible solution if and only if there exists a feasible point in $\mathbb{R}^4$.

There are $O(n^2)$ apex regions, and each is defined by 2 polynomials of constant degree; thus, the arrangement of the regions has polynomial complexity. The feasibility of a point in $\mathbb{R}^4$ does not change as the point moves inside the cell of the arrangement; hence, in order to determine existence of a feasible point, it is enough to check the feasibility of an arbitrary representative point $r = (p, c)$ inside every cell. By [4], a representative for each cell can be obtained in $O(n^2)$ time.

To check if $r = (p, c)$ is feasible, build the bipartite graph G_r; the $n-1$ nodes on one part correspond to the disks $D \setminus d_1$, the $n-1$ nodes on the other part correspond to the angles $\{\pi/n, 4\pi/n, 6\pi/n, \ldots, (n-1)2\pi/n\}$. There is an edge between a disk node d_j and an angle node $k2\pi/n$ if p rotated around c by the angle $k2\pi/n$ lands in d_j (i.e., $\rho_c^k(p) \in d_j$). There is a perfect matching in G_r if and only if c is the center of a regular n-gon with vertices in the disks from D.

Running time Each apex region consists of 2 algebraic surfaces in $\mathbb{R}^4$; thus, the arrangement defined by all surfaces consists of $O(n^8)$ cells. It takes $O(n^{10})$ time to compute a representative for each cell [4]. For each representative of a cell we compute the corresponding bipartite graph and check if the graph has a perfect matching. A maximum matching in a bipartite graph with n nodes and m edges can be computed in $O(m\sqrt{n})$ time [10]. Since the graphs G_r and $G_{r'}$ differ in only one edge whenever r and r' lie in neighboring cells, we can use a dynamic matching algorithm. We traverse the cells in depth first search order. Each update in the

dynamic matching algorithm can be done in $O(m)$ time [2]. Thus, the total running time for our algorithm is $O(n^{10})$.

The above algorithm can be used for objects other than disks, only the running time will change depending on the complexity of the apex regions.

4.2 Optimization problem: Symmetry with imprecision

We now consider the following problem: Given a set $P = \{p_1, \ldots, p_n\}$ of n points, find the minimum δ^* such that shifting each point by at most δ^* brings the points in symmetric position (which means they are vertices of a regular n-gon). We give an exact algorithm, a quick constant-factor approximation, and a PTAS for the problem.

4.2.1 Exact solution

It is immediate that in the optimal solution, some J points of P are shifted by exactly δ^*; we argue that $J \leq 5$. Renumber the points in P so that the points shifted by δ^* are $p_1, \ldots, p_J$, and let $q_1, \ldots, q_J$ be the shifted points. Suppose we know that q_j is the k_j-th vertex of the optimal n-gon, where $k_1, \ldots, k_J$ are some distinct integers between 1 and n. We can then write one equation for each $j = 1, \ldots, J$:

$$|R^{k_j \frac{2\pi}{n}}(p_j - c) - (q_1 - c)| = \delta^*$$

where c is the center of symmetry of the n-gon and $R^{k_j 2\pi/n}$ is the rotation matrix with rotation angle $k_j 2\pi/n$. Overall, we have J equations in 5 variables (two for each of c and q_1, and one for δ^*) . The system has a solution with an isolated δ^* when $J = 5$.

The above observations lead to a (high) polynomial-time algorithm for the problem: Guess 5 points of P and 5 numbers $k_1, \ldots, k_5$. For each guess, solve the above described system of 5 equations in 5 unknowns to get (a constant number of) candidate values for δ^*; for each candidate run the symmetry detection algorithm from Section 4.1 with radius-δ^* disks centered on points of P in the input.

4.2.2 $O(1)$-approximations

We start with two auxiliary lemmas:

Lemma 4.1. *Let Q be an arbitrary regular n-gon; let c be its center. Let $r \in \mathbb{R}^2$ be an arbitrary point; let q be the vertex of Q closest to r. Moving each vertex of Q by at most $|qr|$, the regular n-gon Q can be modified to a regular n-gon $Q_{c,r}$ that is also centered at c and has r as a vertex.*

Proof. Let the vertices of Q be $q, q_1, q_2, \ldots, q_{n-1}$ in counterclockwise direction. Consider the translation vector $\vec{pr}$ that moves p to r. Each vertex q_i of Q is translated by a vector that is defined by $\vec{pr}$ rotated by $i2\pi/n$ around c, see Figure 16. Hence, each vertex is moved by a distance of $|qr|$ and the points $r, q'_1, \ldots, q'_{n-1}$ build a regular n-gon with center c, $Q_{c,r}$. □

Let $Q^* = q_1^*, \ldots, q_n^*$ be the optimal regular n-gon ($|p_i q_i^*| \leq \delta^*$), and let c^* be the center of Q^*. Let $g = \frac{1}{n}\sum_{i=1}^{n} p_i$ be the centroid of P.

Lemma 4.2. $|c^* g| \leq \delta^*$.

Proof.

$$|c^* g| = \left|\left| \frac{1}{n}\sum_{i=1}^{n} q_i^* - \frac{1}{n}\sum_{i=1}^{n} p_i \right|\right| \leq \frac{1}{n}\sum_{i=1}^{n} ||q_i^* - p_i|| \leq \delta^*$$

□

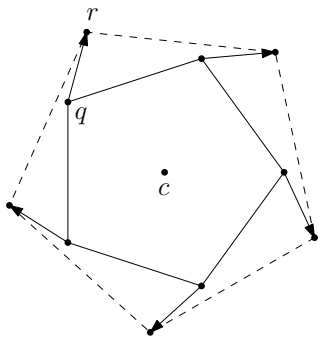

Figure 16: Q can be moved to $Q_{c,r}$ (the dashed polygon).

We are now ready to give our constant-factor approximation algorithms.

A 4-approximation Take any point $p \in P$ and compute, in $O(n)$ time, the regular n-gon $Q_{g,p}$ that has p as a vertex and g as center. Compute bottleneck matching between P and vertices of $Q_{g,p}$, i.e., find the ordering $q_1, \ldots, q_n$ of vertices of $Q_{g,p}$ and minimum $\delta_{g,p}$ such that for any $i = 1, \ldots, n$, $|p_i q_i| \leq \delta_{g,p}$.

Lemma 4.3. $\delta_{g,p} \leq 4\delta^*$

Proof. The n-gon $Q_{g,p}$ can be obtained from the optimal n-gon Q^* as follows: First, shift Q^* by $g - c^*$ (so that the center of the shifted polygon Q is at g), and then apply Lemma 4.1 (so that the polygon has p as a vertex). Let q^* be the vertex of the optimal n-gon Q^* closest to p. Before the shifting, we had $|q^*p| \leq \delta^*$. By Lemma 4.2, the shift is not larger than δ^*, and, hence, there is a vertex of the shifted polygon within distance $2\delta^*$ from p. By Lemma 4.1, $Q_{g,p}$ can be obtained from the shifted polygon, moving every vertex by at most $2\delta^*$. Overall, any vertex of $Q_{g,p}$ finds itself within distance $\delta^* + \delta^* + 2\delta^*$ from the corresponding point of P. □

Interestingly, constructing $Q_{g,p}$ alone does not yield a 4-approximation of the *value* of δ^* (even though we know that $Q_{g,p}$ is a 4-approximation); this is because (other than for p) we do not know which point of P moves to which vertex of $Q_{g,p}$. To know the value of $\delta_{g,p}$, one needs to compute the bottleneck matching between P and vertices of $Q_{g,p}$. While $Q_{g,p}$ itself can be computed in linear time, we know of no faster algorithm for computing $\delta_{g,p}$ than the general $O(n^{1.5} \log n)$-time algorithm of [7].

A 3-approximation To improve the approximation, run the above approximation algorithm with each point of P serving as the point p, and choose the one that leads to the smallest $\delta_{g,p}$ (overall, this algorithm takes $O(n^{2.5} \log n)$ time).

Lemma 4.4. $\min_{p \in P} \delta_{g,p} < 3\delta^*$

Proof. Consider the set of vectors $V = \vec{p_i q_i^*}, i = 1, \ldots, n$; they must "span the full 2π" (formally, any vector in $\mathbb{R}^2$ must be representable as a linear combination of vectors in V with non-negative coefficients). Thus, at least one vector $\vec{p^* q^*} \in V$ makes a positive angle with $\vec{c^* g}$ – the shift vector. Hence, the shift brings q^* closer to p^* – after the shift, the distance between the shifted vertex and p^* is smaller than it was before the shift, i.e., is smaller than δ^*. Applying the operations from Lemma 4.1 to the shifted polygon and p^*, moves each point of the shifted polygon by at most δ^*. Overall, any vertex of Q_{g,p^*} finds itself within distance $2\delta^* + \delta^*$ from the corresponding point of P. □

A PTAS Compute a 4-approximation δ of δ^*, and lay out $\frac{1}{\varepsilon} \times \frac{1}{\varepsilon}$ grids G_g and G_p in the δ-neighborhood of g and the δ-neighborhood of some point $p \in P$, respectively. Then, for each pair (g', p') of grid points from $G_g \times G_p$, compute the regular polygon $Q_{g',p'}$ centered at g' and having a vertex at p', and find the value $\delta_{g',p'}$ of the bottleneck matching between P and the vertices of $Q_{g',p'}$; this can be done in overall $O(\frac{1}{\varepsilon^4} n^{1.5} \log n)$ time.

Lemma 4.5. $\min_{g',p'} \delta_{g',p'} \leq (1 + O(\varepsilon))\delta^*$

Proof. Some vertex q^* of Q^* is within distance δ from p; thus, q^* is within distance $O(\varepsilon\delta^*)$ from some gridpoint $p^* \in G_p$. Shift the optimal polygon Q^* so that its center c^* moves onto the closest point $g^* \in G_g$. The shift moves each vertex of Q^* by $O(\varepsilon\delta^*)$; in particular, the shifted q^* remains $O(\varepsilon\delta^*)$-close to p^*. Applying Lemma 4.1, we obtain that each vertex of Q_{g^*,p^*} finds itself within distance $\delta^* + O(\varepsilon\delta^*) + O(\varepsilon\delta^*)$ from the corresponding vertex of P. □

5 Conclusion

We resolved a long-standing open question: Can one determine in polynomial time whether a set of objects has a convex transversal? We gave negative answers for segments and scaled copies of a convex polygon in 2D and for balls in 3D. Our construction showing hardness of stabbing non-disjoint segments in 2D can be lifted to 3D while removing the intersections between the segments; hence, stabbing disjoint objects in 3D is also hard.

Note that the segments/balls used in our hardness proofs are of drastically different sizes (in particular, we used zero-length segments and zero-volume balls which might be considered as somewhat artificial). Our construction for the segments can be easily extended to the case where all segments have length between 1 and $1 + \epsilon$ for any $\epsilon > 0$; we believe that similar extension is possible for the balls. However, for unit segments/balls our constructions fails. We leave these as open problems.

On the positive side, we gave a polynomial-time algorithm to determine a convex stabber, if it exists, for a set of disjoint line segments or disjoint convex polygons under the restriction that the stabber vertices come from a given set C of candidate points. The most intriguing open question is whether the restriction can be removed. Another open question is whether there are non-trivial lower bounds on the complexity of algorithms for finding transversals.

In general, convex transversals open a whole new research direction. Apart from the algorithmic study, it could be of interest to investigate combinatorial properties of convex stabbers, e.g., the number of geometric permutations induced by convex stabbers for different classes of objects.

Acknowledgments

We thank Joe Blitzstein (Harvard University) for pointers to work in convex regression, and the anonymous reviewer for helpful comments. E. Arkin and J. Mitchell are partially supported by the National Science Foundation (CCF-0729019, CCF-1018388). Work by L. Schlipf was supported by the Deutsche Forschungsgemeinschaft within the research training group "Methods for Discrete Structures"(GRK 1408). V. Polishchuk is supported by the Academy of Finland grant 138520.

References

[1] P. K. Agarwal, A. Efrat, C. Gniady, J. S. B. Mitchell, V. Polishchuk, and G. Sabhnani. Distributed localization and clustering using data correlation and the Occam's razor principle.

In *7th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 1–8. IEEE, 2011.

[2] D. Alberts and M. R. Henzinger. Average case analysis of dynamic graph algorithms. In *SODA '95*, pages 312–321, 1995.

[3] E. M. Arkin, J. S. B. Mitchell, V. Polishchuk, and S. Yang. Convex transversals. In *Fall Workshop on Computational Geometry*, 2010.

[4] S. Basu, R. Pollack, and M.-F. Roy. On computing a set of points meeting every cell defined by a family of polynomials on a variety. *J. Complex.*, 13(1):28–37, 1997.

[5] M. Birke and H. Dette. Estimating a convex function in nonparametric regression. *Scandinavian Journal of Statistics*, 34(2):384–404, 2007.

[6] A. Dumitrescu and M. Jiang. Minimum-perimeter intersecting polygons. *Algorithmica*, 63(3):602–615, 2012.

[7] A. Efrat, A. Itai, and M. J. Katz. Geometry helps in bottleneck matching and related problems. *Algorithmica*, 31(1):1–28, 2001.

[8] M. T. Goodrich and J. Snoeyink. Stabbing parallel segments with a convex polygon. *Comput. Vision Graph. Image Process.*, 49(2):152–170, 1990.

[9] P. Groeneboom, G. Jongbloed, and J. A. Wellner. Estimation of a convex function: characterizations and asymptotic theory. *The Annals of Statistics*, 29(6):1653–1698, 2001.

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

[11] S. Iwanowski. Testing approximate symmetry in the plane is NP-hard. *Theor. Comput. Sci.*, 80(2):227–262, 1991.

[12] H. Kaplan, N. Rubin, and M. Sharir. Line transversals of convex polyhedra in $\mathbb{R}^3$. *SIAM J. Comput.*, 39(7):3283–3310, 2010.

[13] M. Löffler and M. J. van Kreveld. Largest and smallest convex hulls for imprecise points. *Algorithmica*, 56(2):235–269, 2010.

[14] A. Tamir. Problem 4-2 (New York University, Dept. of Statistics and Operations Research), Problems Presented at the Fourth NYU Computational Geometry Day (3/13/87).