Technische Universität Berlin

FAKULTÄT II
MATHEMATIK UND
NATURWISSENSCHAFTEN
Institut für Mathematik

Algorithms for Manufacturing Paperclips and Sheet Metal Structures

by

Esther M. Arkin   Sándor P. Fekete
Joseph S. B. Mitchell

No. 710/2001

Algorithms for Manufacturing Paperclips and Sheet Metal Structures *

Esther M. Arkin[†] Sándor P. Fekete[‡] Joseph S. B. Mitchell[*]

Abstract

We study algorithmic aspects of bending wires and sheet metal into a specified structure. Problems of this type are closely related to the question of deciding whether a simple non-self-intersecting wire structure (a "carpenter's ruler") can be straightened, a problem that was open for several years and has only recently been solved in the affirmative.

If we impose some of the constraints that are present in industrial manufacturing, we get quite different results. In particular, we show that it is NP-complete (in the weak sense) to decide if a final configuration (simple chain in the plane, or simple polyhedral surface) is obtainable from a straight wire or flat sheet, using "all-or-nothing" folds, while keeping the structure *simple* (non-self-intersecting). (Each bend along a crease is made *once*, from the initial (flat) state to the final desired angle.) We also show that it is NP-complete to determine if a polygonal chain can be straightened, if the chain is allowed to have a vertex degeneracy, and only one joint can be opened at a time (allowing partial openings).

If we are required to make bends sequentially from one or both ends of the wire, as is often the realistic situation in wire forming manufacturing, we show that the problem becomes easier. We give efficient algorithms for these cases.

1 Introduction

The following is an algorithmic problem that arises in the study of the manufacturability of sheet metal parts: *Given a flat piece, F, of sheet metal (or cardboard, or other bendable "stiff" sheet material), can a desired final polyhedral part, P, be made from it?* The 2-dimensional version is the wire-bending ("paperclip") problem: *Given a straight piece, F, of wire, can a desired simple polygonal chain, P, be made from it?* This problem also arises in the fabrication of hydraulic tubes, e.g., in airplane manufacturing.[1] In either version of the problem, we require that any intermediate configuration during the manufacture of the part be feasible – it should not be self-intersecting. In particular, the "paperclips" that we manufacture are not allowed to be "pretzels" – we assume that the wire must stay within the plane, and not cross over itself. See Figure 1 for an illustration. We acknowledge that some real paperclips are designed to cross over themselves, such as the "butterfly" style of clip shown in the figure.

[*]A preliminary version of this work was presented at the 17th European Workshop on Computational Geometry, March 26-28, 2001, Berlin.

[†]Department of Applied Mathematics and Statistics, State University of New York, Stony Brook, NY 11794-3600, USA. email: {estie, jsbm}@ams.sunysb.edu.

[‡]Department of Mathematical Optimization TU Braunschweig, Pockelsstr. 14, D-38106 Braunschweig, Germany. email: sandor.fekete@tu-bs.de.

[1]We thank Karel Zikan for introducing to us the hydraulic tube bending problem at Boeing's factory.

Our problem is one of automated process planning: Determine a sequence (if one exists) for performing the bend operations in sheet metal manufacturing. We take a somewhat idealized approach in this paper, in that we do not attempt to model here the important aspects of tool setup, grasp positions, robot motion plans, or specific sheet metal material properties which may affect the process. Instead, we focus on the precise algorithmic problem of determining a sequence for bend operations, on a given sheet of material with given bend lines, assuming that the only constraint to performing a bend along a given bend line is whether or not the structure intersects itself at any time during the bend operation. To our knowledge, no prior algorithmic study of these problems has been done.

Motivation. Our foldability problem is motivated from process planning in manufacturing of structures from wire, tubing, sheet metal, and cardboard; see the prior work cited below. It is also motivated by the mathematical study of origami, which has received considerable attention in recent years (again, see below). Finally, we are motivated by the study of linkage problems; in fact, in the time since this paper was first drafted, the "carpenter's ruler conjecture" (of Lenhart-Whitesides, Mitchell, and others) has been resolved. Our hardness results are particularly interesting and relevant in light of these new developments, since we show that even slight changes in the assumptions about the model or the allowed input results in linkages that cannot be opened, and it is NP-hard to decide if they can be opened.[2]

Related Work. The CAD/CAM scientific community has studied extensively the problem of manufacturability of sheet metal structures; see the thesis of Wang [33] for a survey. Systems have been built (e.g., PART-S [11] and BendCad [16]) to do computer-aided process planning in the context of sheet metal manufacturing; see also [3, 18, 35, 9, 34]. See [23] for a motion planning approach to the problem of computing folding sequences for folding three-dimensional cardboard cartons. Considerable effort has gone into the design of good heuristics for determining a bend sequence; however, the known algorithms are based on heuristic search (e.g., A^*) in large state spaces; they are known to be worst-case exponential. (Wang [33] cites the known complexity as $O(n!2^n)$.)

Recently, [10] and [31] have obtained proofs of the *carpenter's ruler conjecture* of Lenhart-Whitesides and Mitchell: Any polygonal *linkage* with fixed length links and hinged joints, can be straightened while maintaining simplicity (i.e., without the linkage crossing or touching itself). In fact, Streinu [31] gives an algorithmic solution that bounds the complexity of the unfolding and is somewhat more general than the slightly earlier results of [10]. (Note that these results also imply a positive answer if only the joints of the linkage can only be changed one at a time, but as often as necessary.) Earlier and related work on linkages includes [6, 7, 21, 22, 28, 29, 32].

In mathematics of origami, Bern and Hayes [5] have studied the algorithmic complexity of deciding if a given crease pattern can be folded flat; they give an NP-hardness proof. Lang [19, 20] gives algorithms for computing crease patterns in order to achieve desired shapes in three dimensions. Other work on computational origami includes [1, 12, 13, 14, 17, 26, 27, 30].

A closely related problem is that of flat foldings of polyhedra. It is a classic open question whether or not every convex polytope in three dimensions can be cut open along its edges so that it unfolds flat, without overlaps. Other variants and special cases have been studied; see [2, 4, 8, 24, 25].

[2]Note that the problem of determining if a bend sequence exists that allows a structure to *unfold* is equivalent to that of determining if a bend sequence exists that allows one to *fold* a flat (or straight) input into the desired final structure: the bending operations can simply be reversed.

Summary of Results.

(1) We show that the problem of sequencing bend operations is (weakly) NP-complete, even in the special case in which the desired structure is a rectilinear polygonal chain. (This immediately implies that the sheet metal bending problem is hard as well, even in the case of parallel bend lines and an orthohedral structure P.) Here we assume that each bend is a "complete" bend operation.

(2) We prove that it is (weakly) NP-hard to determine if a polygonal linkage can be straightened, if there is a vertex degeneracy (two vertices coincide). Here we assume that a bend operation consists of rotating at a single joint, by any angle. (The bends need not be "complete.")

(3) We also give an efficient ($O(n \log^2 n)$) algorithm for deciding whether the bends in a wire-bending problem can be done sequentially in order from one end along the wire (as is required in many automated wire-bending machines). We show that one can test in time $O(n^2 \log n)$ if any particular bend sequence is feasible. Further, we give efficient polynomial-time algorithms for deciding whether there is a feasible sequence for wire bending from one of two special sequence classes arising in practice (i.e., manufacturing "from" or "towards" both ends).

2 Preliminaries

We let F denote the input *flat* material.

In the 3-dimensional sheet metal folding problem, we assume that F is a polygon (possibly with holes) having m edges, with n *bend lines* (segments), $B = \{b_1, \ldots, b_n\}$, lying interior to it. We distinguish between the *top* and *bottom* sides of F. The set of segments B forms a planar graph that partitions F into a set of polygonal facets. Each bend line $b_i \in B$ has an associated *bend angle*, $\theta_i \in (0, 2\pi)$, which gives the desired dihedral angle at b_i, measured between the tops of the two facets incident on b_i. We let P denote the desired polyhedral surface structure for the final part. The facets of P are precisely the faces in the bend line decomposition of F, with the dihedral angle at each edge given by the associated bend angles.

In the 2-dimensional wire folding problem, we assume that F is an (oriented) straight line segment (having a *left* and *right* side), with n *bend points* (or *joints*), $B = (b_1, b_2, \ldots, b_n)$, in order along $F = b_0 b_{n+1}$, having bend angles $\theta_i \in (0, 2\pi)$ (measured between the right sides of F). We let P denote the desired (oriented) polygonal chain (of the same length as F, of course). The edges of P have lengths $|b_0 b_1|$, $|b_1 b_2|$, ..., $|b_n b_{n+1}|$, and the vertices (also denoted b_0, b_1, ..., b_{n+1}) have associated angles θ_i, for $i = 1, \ldots, n$.

For $S \subseteq B$, we let $P(S)$, denote the structure (polyhedral surface or polygonal chain) that is the result of performing exactly the bends $b_i \in S$ on the input material F. (So, $P(B) = P$.) We let $P(S; i, \theta)$, for $1 \le i \le n$ with $i \notin S$, denote the structure that is the result of performing exactly the bends $b_i \in S$ and bending b_i to be at angle θ (between the top/right sides of F). We say that structure $P(S)$ or $P(S; i, \theta)$ is *feasible* if it has no self-intersections (more precisely, no two facets intersect unless they share a common vertex/edge in F, in which case they intersect only at that common vertex/edge).

We say that bend b_i is *foldable* (or is a *feasible fold*) for structure $P(S)$ if $P(S; i, \theta)$ is feasible for all θ in the range between π and θ_i (more precisely, for all $\theta_i \le \theta \le \pi$, if $\theta_i < \pi$, or for all $\pi \le \theta \le \theta_i$, if $\theta_i > \pi$). We say that a permutation $\sigma = (i_1, i_2, \ldots, i_n)$ of the indices $\{1, 2, \ldots, n\}$ is *foldable for* P if, for $j = 1, 2, \ldots, n$, joint b_{i_j} is foldable for $P(\{b_{i_1}, \ldots, b_{i_{j-1}}\})$, i.e., if P can be manufactured from F using the bend sequence σ.

The problems of Wire Bend Sequencing and of Sheet Metal Bend Sequencing can both be formally stated as: Determine a foldable permutation σ, if one exists, for a given wire/sheet-metal structure P.

We give an example in Figure 1 of some common paperclip shapes, (a)–(c). We also show an example, (d), of a 5-link paperclip that cannot be manufactured using complete bends, for any permutation σ of the bends. Finally, we show an example of a 6-link paperclip for which the foldable permutations are $\{(2,3,4,5,1),(3,2,4,5,1)\}$; we show the sequence of bends, with the intermediate structures, for the permutation $\sigma = (2,3,4,5,1)$.

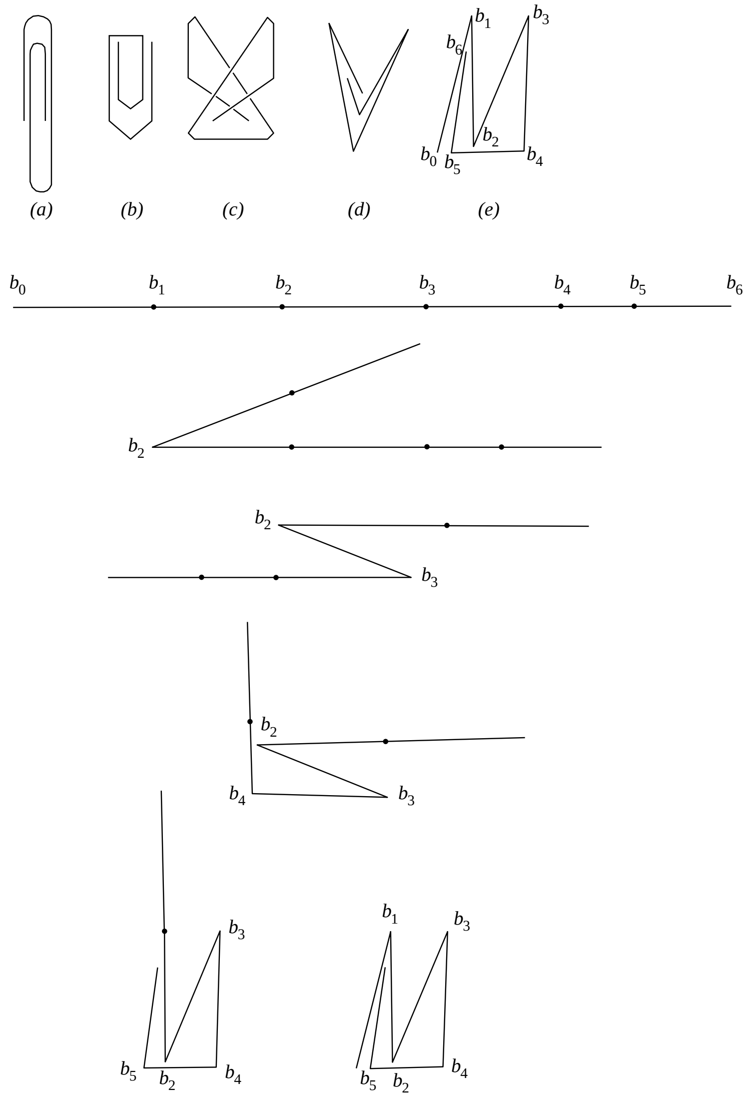

Figure 1: Examples of paperclips: (a) and (b) are standard versions, which are readily constructed. (c) is a “butterfly” paperclip, which is not a planar structure and is not among the wire structures considered in our two-dimensional model. (d) shows a 5-link paperclip that cannot be manufactured using complete folds in the plane. (e) shows a 6-link structure that can be manufactured, e.g., using the bend sequence animated below it for the bend sequence $\sigma = (2,3,4,5,1)$.

3 Hardness Results

In this section, we describe two results illustrating that a positive answer to the carpenter's ruler conjecture is strongly dependent on both using more than one link at a time, and not allowing the wire to touch itself along the way.

3.1 Complete Folding

The first result shows that the practical requirement of only being allowed to bend the wire once at each joint (in order to avoid breaking it) makes the problem quite hard, even if we are limited to a fixed number of monotone subsequences of folding. We call this variant "complete folding", since each individual angle has to be achieved completely in one move.

Theorem 3.1 *The* WIRE BEND SEQUENCING *problem is (weakly) NP-complete, even if P is rectilinear and we restrict ourselves to making only four monotone passes of complete foldings.*

Proof: Our reduction is from PARTITION: Given a set S of n integers, a_i, which sum to $A = \sum_i a_i$, determine if there exists a partition of the set into two subsets each of which sums to $A/2$.

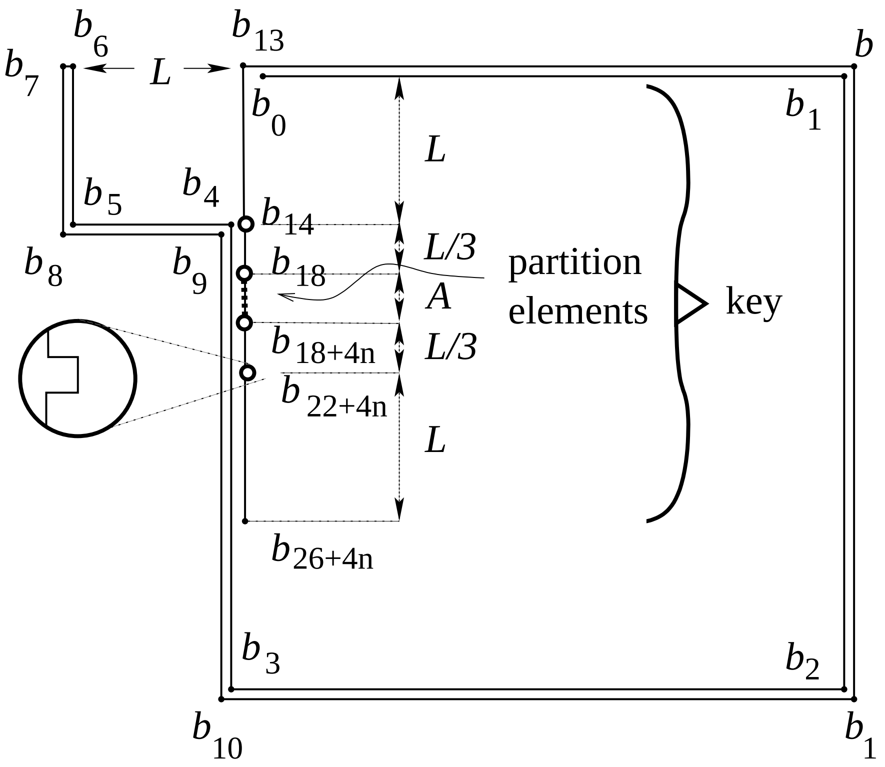

Figure 2: NP-hardness of complete folding for rectilinear chains: Frame and key.

The key idea of our construction uses two components, as shown in Figure 2, where $L > A/3$: One is a rigid "frame" that can only be unfolded if one end of the chain can be removed from within this frame. The other component is a "key" that has to be collapsed to a small length in order to be removable. Collapsing the key is possible if and only if there is a partition of the integers into two sets of equal sum. The total number of segments will be $\ell = 26 + 4n$; we write b_i ($i = 0, 1, \ldots, 26 + 4n$) for the vertices, and $s_i = (b_{i-1}, b_i)$ for the segments.

Note that we have two types of joints in the figure: the "ordinary" ones (indicated by solid black dots) form the frame and can only be accessed once. The "quadruple" ones (indicated by hollow dots in Figure 2) are found along the key and consist of four ordinary joints that are connected by short segments of length ε, making it possible to simulate opening and closing such a joint a limited number of times. (In the following, all lengths are adjusted with small changes of size $O(\varepsilon)$ to $O(n\varepsilon)$, such that coordinates vary by $O(n\varepsilon)$. We choose ε sufficiently small and note that all other lengths are integers; for clarity of presentation, we do not discuss these small variations explicitly.)

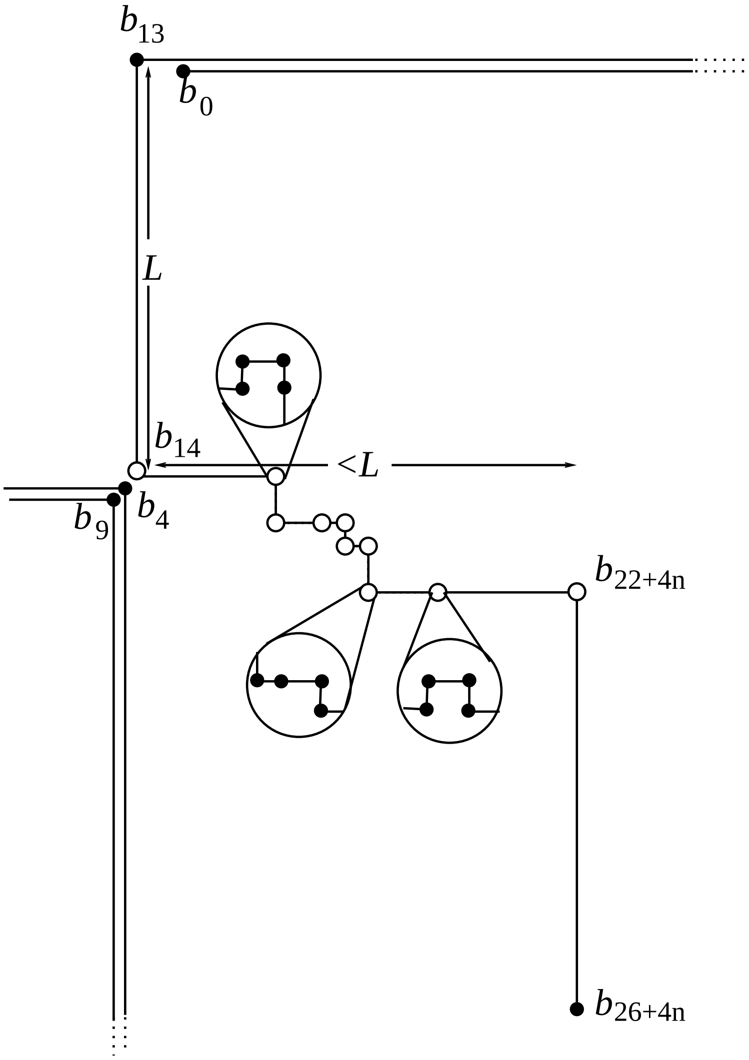

Figure 3: Turning the key into a stair.

In order to see that the key can be removed from the frame if there is a partition $S = S_1 \dot\cup S_2$, such that $\sum_{i \in S_1} a_i = A/2$, we first convert it into the "stair" configuration shown in Figure 3: We make one monotone pass over the chain from the key end, and straighten one ordinary joint per quadruple joint whenever this joint separates two segments from different S_i. Thus, segments corresponding to numbers in S_1 will be horizontal, while those for numbers in S_2 will be vertical. Making a similar monotone pass from the other end, we can convert the stair into a "flat harmonica", with segments from S_1 pointing "down", while those in S_2 pointing "up". By assumption about the partition, the endpoints b_{15} and $b_{\ell-2}$ will be at the same vertical position, $2L/3$ below b_{13}. Therefore, the last segment $s_\ell = b_{\ell-1}b_\ell$ of length L in the chain will be placed in the same vertical position as the segment $b_{13}b_{14}$, with all other segments strictly in-between. This collapsed arrangement can be rotated about b_{13} without colliding with any frame segments. Then it is easy to open up the remaining frame (by straightening b_{12}, b_{11}, b_{10}, b_8, b_7, b_6, b_5, b_4, b_3, b_2 as one monotone pass). Finally, the resulting monotone chain can be straightened in one last monotone pass.

Conversely, assume now that the chain can be straightened. It is clear that b_{13} must be straightened before any other joint in the set $\{b_2, \ldots, b_{12}\}$. In order to avoid hitting vertex b_4 during this motion, the endpoint b_{26+4n} must be strictly within the circle of radius L around b_{13}. s_ℓ is an axis-parallel segment, and the distance of b_{25+4n} is strictly less than L. Furthermore, the rigid frame ensures that segment s_{14} cannot lie to the left of s_{13}, and it follows that s_ℓ can only lie within the quarter circle below and to the right of b_{13}. Therefore, s_ℓ must be vertical, at the same horizontal position as the segment s_{13}. This implies that the total length of the segments between s_{15} and $s_{\ell-2}$ that point "up" is equal to the segments in that set that point "down". This induces a partition of the desired form, completing the proof. □

3.2 Incomplete Folding

Now we consider the case where each link may be changed an arbitrary number of times before achieving a desired angle. In analogy to the previous scenario, we call this "incomplete folding". This case is even more closely related to the carpenter's ruler problem, which was solved in the affirmative by [10, 31]. In fact, we observe the following:

Theorem 3.2 *Any simple (not self-intersecting and not self-touching) polygonal chain P can be opened (and thus manufactured from a straight wire) by a sequence of moves that bends only one joint at any one time.*

Proof: Consider the feasible set S of points in state space, where moving individual joints corresponds to axis-parallel motion. If self-touching is prohibited, S is an open bounded set. By Streinu's result [31], there is an opening motion of the chain that consists of a bounded number of individual moves. For this motion, let ε be the (positive) infimum distance of a feasible position from the boundary of S. Then we can replace each part of the feasible path by a series of axis-parallel moves of size $\varepsilon/2$, yielding a feasible path with the required property. □

We will refer to a sequence of small individual moves that mimics an overall large-scale motion of several joints as "wiggly", since the overall motion may be achieved through back-and-forth motions of individual segments that gradually changes individual angles.

The following results show that allowing even a single point of self-incidence along the wire changes the overall situation quite drastically.

Lemma 3.1 *There are polygonal chains P with a single vertex-to-vertex incidence that cannot be manufactured by arbitrary (noncomplete) single-joint bends.*

Proof: See Figure 4. The chain has eight joints (labeled $b_0, \ldots, b_7$) and seven segments (of the form $s_i = (b_{i-1}, b_i)$). The end point b_0 coincides with joint b_5. It is easily checked that none of the joints $b_1, \ldots, b_4$ can be changed without causing a self-intersection: Assume that there is a feasible motion of a joint b_i with $0 < i < 5$. Then the points b_0 and b_5 would move away from each other along a circle around b_i. Without loss of generality, assume that b_5 remains in place, while b_0 is moving. Now consider the first such rotation that starts with b_0 and b_5 coinciding, and that avoids a crossing of s_1 with both s_5 and s_6. If b_0 moves clockwise around b_i, it is easy to see that the angle between s_1 and s_{13} must be at least $\pi/2$ when starting the motion, or s_1 and s_{13} intersect. If b_0 moves counterclockwise around b_i, the same follows for the angle between s_1 and s_{14}. Therefore, the center of rotation must lie within the shaded region shown in the figure. However, none of the joints $b_1, \ldots, b_4$ lies inside of this feasible region. It follows that $b_0, \ldots, b_5$ form a rigid frame, as long as the angle at b_5 stays smaller than $\pi/2$.

On the other hand, it is easy to see that b_7 cannot be removed from the pocket formed by b_1, b_2, and b_3 if only the two remaining "free" joints b_5 and b_6 can be changed. The claim follows. □

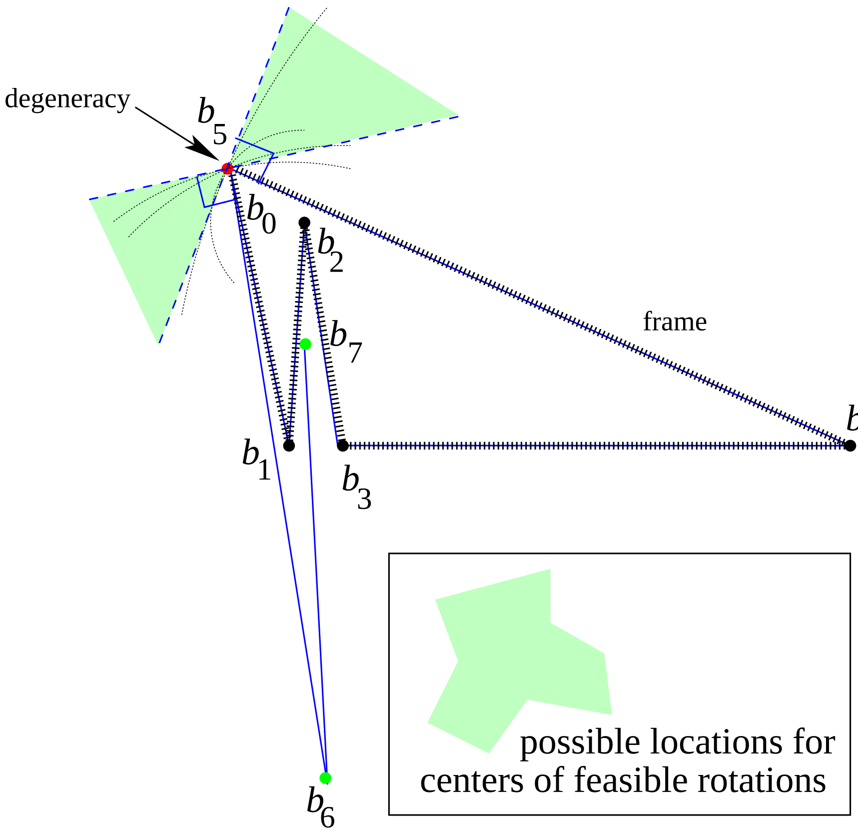

Figure 4: A polygonal chain that cannot be opened with single-joint moves.

Using the frame as a gadget, we can show the following:

Theorem 3.3 *It is NP-hard to decide if a polygonal chain P with a single vertex-to-vertex incidence can be manufactured by arbitrary (noncomplete) single-joint bends.*

Proof: The basic idea is similar to the one in Theorem 3.1 and also establishes a reduction of PARTITION. (Refer to Figure 5 for an overview.) As before, we write b_i for the joints, and $s_i = (b_{i-1}, b_i)$ for the segments. We use the idea of the construction from Lemma 3.1 to construct a rigid frame, with the key corresponding to the free end of that chain. The frame has one end, b_0, of the polygonal chain wedged into the corner b_{13}, which has angle $\varphi < \pi/2$. Because of the degeneracy at b_{13}, none of the joints $b_1, \ldots, b_{12}$ can be moved individually without causing a self-intersection between b_0 and the chain in the neighborhood of b_{13}. Again, the "key" consisting of the n segments $s_{18}, \ldots, s_{18+n}$ of integral lengths $a_1, \ldots, a_n$ encodes an instance of PARTITION, as in the previous proof. As before, let S denote the set of integers for the PARTITION instance. Again, we use "long" auxiliary segments of length $L/2$ and L, here they are segments s_{18}, s_{19+n}, s_{20+n}, and all sizes are subject to small adjustments of size $O(\varepsilon)$, for sufficiently small ε. The critical dimensions of the frame are chosen such that the key can just be removed from the frame through the narrow bottleneck formed by the segments s_3 and s_9 by extending the "spring" at b_{14}, and b_{15}, if and only if the key can be collapsed to a critical length L. The latter is possible if and only if there is a feasible partition.

In order to establish this equivalence, we only have to prove two facts:

1. As long as φ remains below $\pi/2$, no joint in the frame can be moved in a feasible way; i.e., the frame is rigid.

2. The key can be removed from the lock if and only if there is a feasible solution for the PARTITION instance.

The first claim follows just like in Lemma 3.1. Therefore, we must be able to open φ to at least $\pi/2$ before being able to change the frame. Clearly, this requires moving the key through the bottleneck indicated in the figure.

It is not hard to check that by choosing an appropriately narrow layout for the bottleneck, this can be achieved if and only if the we can move b_{16} down by a vertical distance of $L + \varepsilon$ without causing any collisions between key and frame. (For this purpose, assume that the links b_2, b_3, b_8, b_9 form a narrow rectangle, i.e., the "horizontal" distance between b_3 and b_8 and between b_2 and b_9 is smaller by a factor of ε/L than the "vertical" length of the segments s_3 and s_9.) Since the edge s_{15} has length $L/2 + \varepsilon$, performing a "wiggly" sequence of small individual moves has this desired effect, provided that the key can be placed within a narrow channel extending the bottleneck.

This is possible if there is a partition of S: Perform a "wiggly" sequence that folds the segments for the set S like a harmonica until we get a sufficiently flat arrangement, leaving the segments for the set S_1 pointing "upwards", the segments for S_2 pointing "downwards"; then b_{18+n} is within ε of b_{18}, b_{19+n} within ε of b_{17}, and the final segment s_{20+n} is in (almost) vertical position. Then extend the spring by another wiggly sequence, such that the key is moved through the bottleneck. Now φ can be opened, and the whole frame can be straightened by a straightforward sequence of moves.

To see the converse, assume that segment e_{19+n} can be moved through the bottleneck. Because of the narrow dimensions of the bottleneck, the possible slopes of e_{19+n} are quite restricted: b_{20+n} can move a distance of ε below the connection between b_3 and b_8, if and only if b_{19+n} is within a distance of ε from b_{17}. By construction, this is only possible if there is a feasible partition of S.

□

4 Algorithms

In this section, we turn our attention to positive algorithmic results, giving efficient algorithms for deciding if particular bend sequences are feasible. We consider here only the case of complete folds.

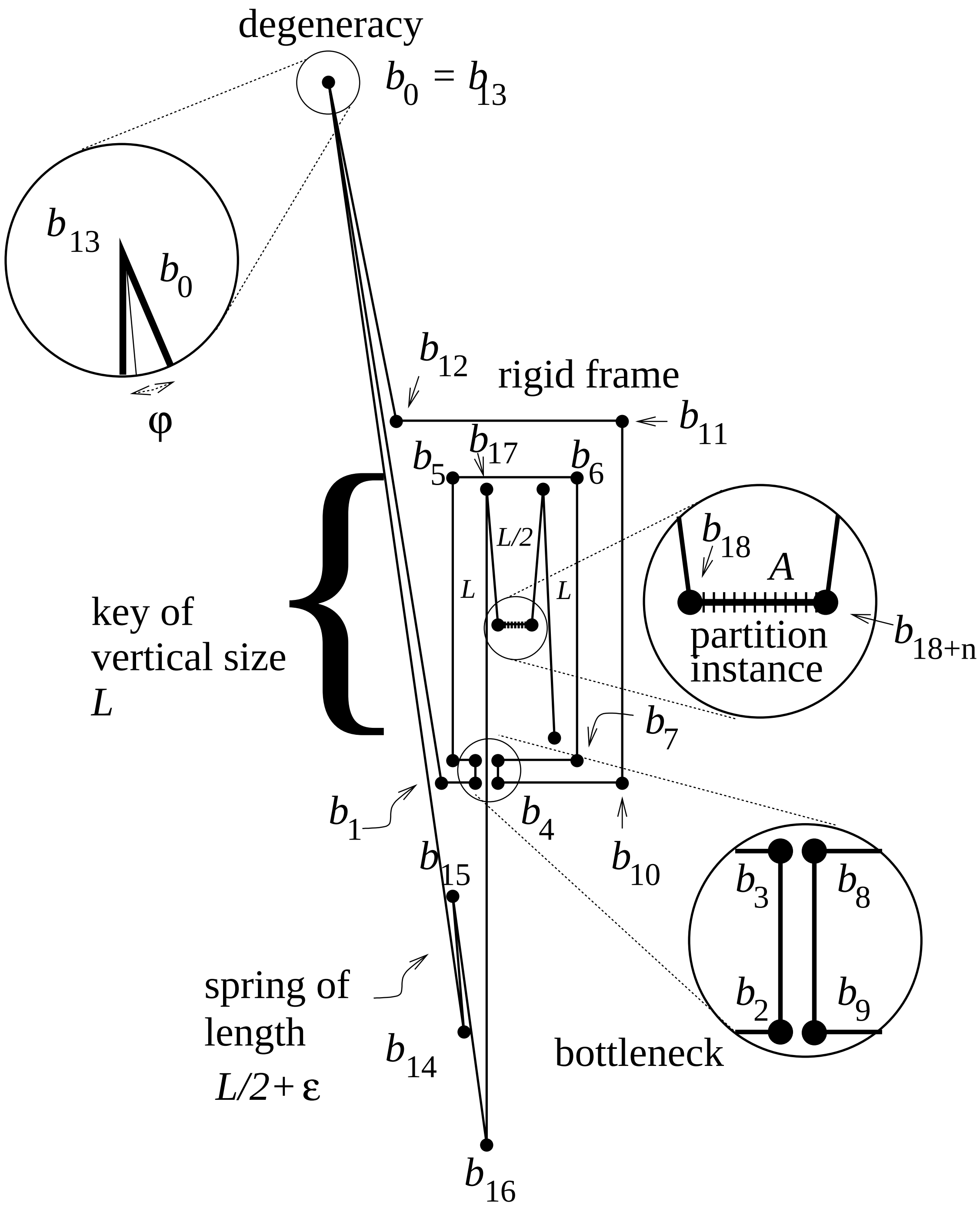

Figure 5: Illustration of the proof of Theorem 3.3.

Consider an arbitrary permutation, $\sigma = (i_1, \ldots, i_n)$, of the bends along a wire. In order for σ to be a foldable sequence, it is necessary and sufficient that for each $j = 1, \ldots, n$ the bend b_{i_j} is foldable. Recall that in our notation $P(\{i_1, \ldots, i_{j-1}\})$ denotes the partially bent chain after the bends at $b_{i_1}, \ldots, b_{i_{j-1}}$ have been completed. The point b_{i_j} splits $P(\{i_1, \ldots, i_{j-1}\})$ into two subchains; let P_0 (resp., P_{n+1}) denote the subchain containing the endpoint b_0 (resp., b_{n+1}). Now, joint b_{i_j} is foldable if the bend at b_{i_j} can be executed without causing a collision to occur between P_0 and P_{n+1} at any time during the rotation about b_{i_j}. We can assume, without loss of generality, that P_0 is fixed and that P_{n+1} is pivoted about b_{i_j}. During the bend at b_{i_j}, each vertex, u, of P_{n+1} moves along a circular arc, A_u, subtending an angle θ_{i_j}, centered on b_{i_j}. Let $\mathcal{A}$ denote the set of all such circular arcs defined by the vertices of P_{n+1}. See Figure 6.

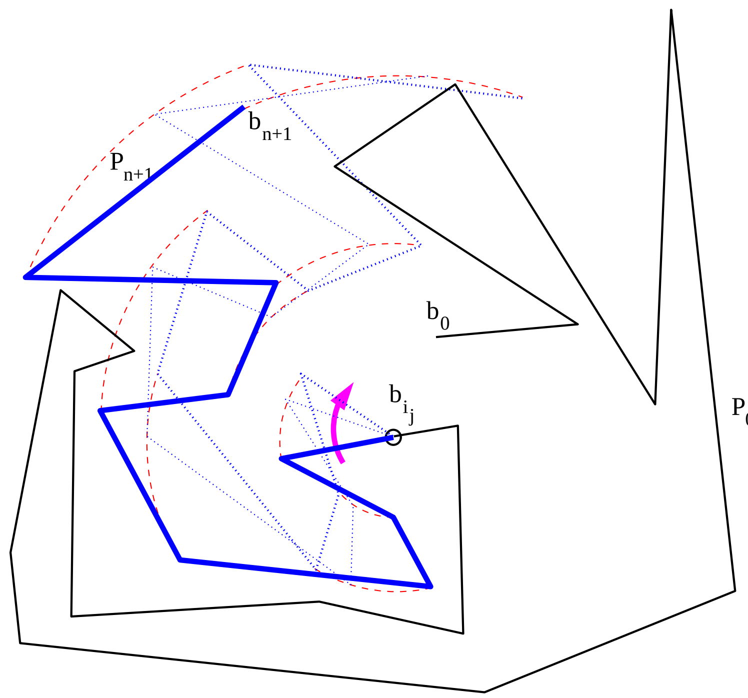

Figure 6: Foldability of the joint b_i: The subchain P_{n+1} is shown with thicker lines (two dotted copies show it after different stages of rotation about b_i). Each vertex of P_{n+1} moves along a circular arc, shown in dashed. In this case, the rotation shown is *not* feasible, as it fails both conditions (1) and (2) of the lemma.

Lemma 4.1 *Joint b_{i_j} is foldable if and only if (1) no arc of $\mathcal{A}$ intersects P_0, and (2) after the bend, no segment of P_{n+1} intersects a segment of P_0.*

Proof: If joint b_{i_j} is foldable then, by definition, there can be no intersection of P_{n+1} with P_0 during its rotation about b_{i_j}. This implies conditions (1) and (2).

If conditions (1) and (2) hold, then we claim that there can be no intersection of P_{n+1} with P_0 during the rotation. Consider a segment, s, of P_{n+1}. During the rotation, it sweeps a region R_s that is bounded by two circular arcs centered at b_{i_j}, corresponding to the trajectories of its endpoints during the rotation, and two line segments, corresponding to the positions of s before and after the rotation. Our claim follows from the fact that P_0 is a simple (connected) chain: It cannot intersect s at some intermediate stage of the rotation unless it intersects the boundary of the region R_s. Such an intersection is exactly what is being checked with conditions (1) and (2). □

Lemma 4.2 *For any $S \subseteq B$, and any $b_i \notin S$, one can decide in $O(n \log n)$ time if joint b_i is foldable for the chain $P(S)$.*

Proof: Using standard plane sweep methods, adapted to include circular arcs, we can check in $O(n \log n)$ time both conditions ((1) and (2)) of Lemma 4.1. □

Remark. Of course, condition (2) can be tested in linear time, by Chazelle's triangulation algorithm. We believe that condition (1) can also be tested in linear time. It involves testing for rotational separability of two *simple* chains about a fixed center point (b_i), which is essentially a polar coordinate variant of translational separability (which is easily tested for simple chains using linear-time visibility (lower envelope) calculation). The issue that must be addressed for our problem, though, is the "wrap-around" effect of the rotation; we believe that this can be resolved and that this idea should lead to a reduction in running time of a factor of $\log n$.

Corollary 4.1 *The foldability of a permutation σ can be tested in $O(n^2 \log n)$ time.*

We obtain improved time bounds for testing the feasibility of a particularly important folding sequence: the identity permutation. Many real tube-bending and wire-bending machines operate in this way, making bends sequentially along the wire/tube. (Such is the case for the hydraulic tube-bending machines at Boeing's factory, where this problem was first suggested to us.) Of course, there are chains P that can be manufactured using an appropriate folding sequence but cannot be manufactured using an identity permutation folding sequence; see Figure 1(e). However, for this special case of identity permutations, we obtain an algorithm for determining feasibility that runs in nearly linear time:

Theorem 4.1 *In time $O(n \log^2 n)$ one can verify if the identity permutation ($\sigma = (1, 2, \dots, n)$) is a foldable permutation for P.*

Proof: Consider executing the bends in the order given by the identity permutation $(b_1, b_2, \dots, b_n)$, and consider the moment when we are testing the foldability of b_i. The subchain P_{n+1} is a single line segment, $b_i b_{n+1}$. Thus, verifying the foldability of bend b_i amounts to testing if the segment $b_i b_{n+1}$ can be rotated about b_i by the desired amount, without colliding with any other parts of the structure P_0 (which consists of the polygonal chain from b_0 to b_i, after the bends at $b_1, \dots, b_{i-1}$ have been completed). In other words, we must do a *wedge emptiness query* with respect to P_0, defined by b_i, segment $b_i b_{n+1}$, and the angle θ_i. Since P_0 is connected, emptiness can be tested by verifying that the boundary of the wedge does not intersect P_0. (See Figure 7.) Thus, we could perform this query by using (straight) ray shooting and circular-arc ray shooting in P_0; the important issue is that P_0 is *dynamically changing* as we proceed with more bends. However, in order to avoid the development of potentially complex dynamic circular-arc ray shooting data structures, we devise a simple and efficient method that "walks" along portions of P_0, testing for intersection with the circular arc, γ, from b_{n+1} to b'_{n+1}, where b'_{n+1} is the location of b_{n+1} *after* the bend at b_i has been performed.

In particular, we keep track of a "painted" portion of P_0, which corresponds to the subset of P_0 that has been "walked over." We consider the chain P_0 to be a degenerate simple polygon, having two *sides* which form a counterclockwise loop around P_0. We consider the case in which the bend at

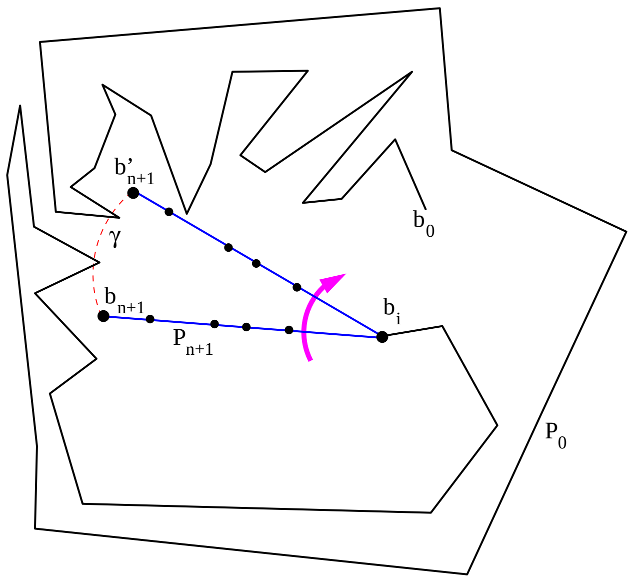

Figure 7: Foldability of the joint b_i when $b_{i+1}, \ldots, b_n$ have not yet been folded.

b_i is a rotation of the segment $b_i b_{n+1}$ *clockwise* to the segment $b_i b'_{n+1}$; the case of a counterclockwise bend at b_i is handled similarly. When we perform a bend at b_i, we walk (counterclockwise) along the *unpainted* portions of P_0, between two points, a and a', on the boundary of P_0, where a and a' are defined according to cases that depend on the outcomes of two ray-shooting queries:

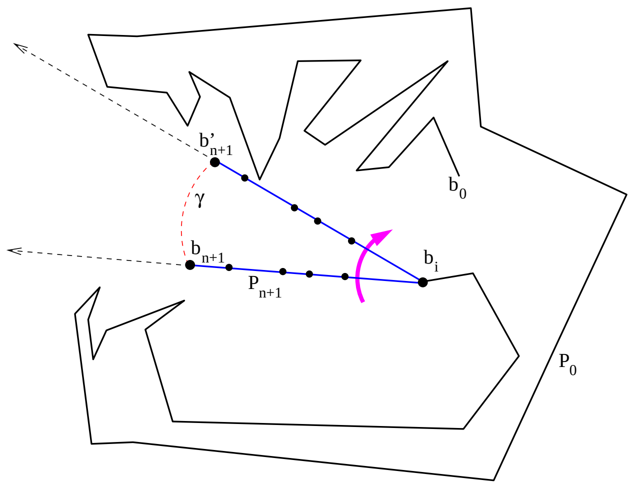

Figure 8: Case (a): Both of the rays $b_i b_{n+1}$ and $b_i b'_{n+1}$ miss P_0 and go off to infinity.

(a) If both of the rays $\overrightarrow{b_i b_{n+1}}$ and $\overrightarrow{b_i b'_{n+1}}$ miss P_0 (and go off to infinity), then there is nothing more to check: the rotation at b_i can be done without interference with P_0, since P_0 is a (connected) polygonal chain lying in the complement of the wedge defined by $\overrightarrow{b_i b_{n+1}}$ and $\overrightarrow{b_i b'_{n+1}}$. See Figure 8.

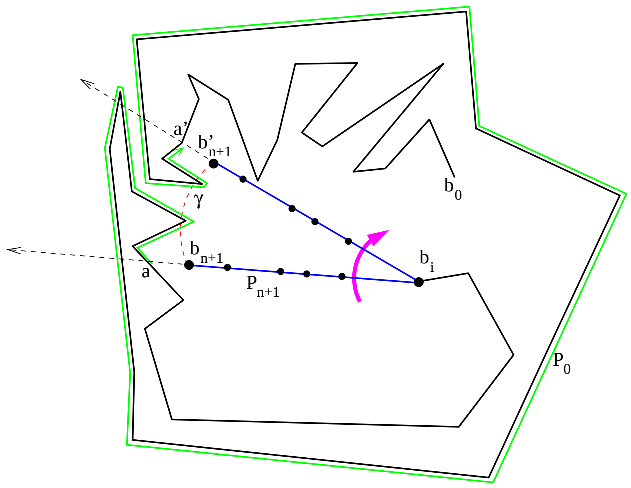

Figure 9: Case (b): Both of the rays $b_i b_{n+1}$ and $b_i b'_{n+1}$ hit P_0. The walk extends from a to a' over the gray highlighted portion of P_0, painting any previously unpainted portion of it.

(b) If both of the rays $\overrightarrow{b_i b_{n+1}}$ and $\overrightarrow{b_i b'_{n+1}}$ hit P_0, then we let a and a' (respectively) be the points on the boundary of P_0 where they hit P_0. See Figure 9.

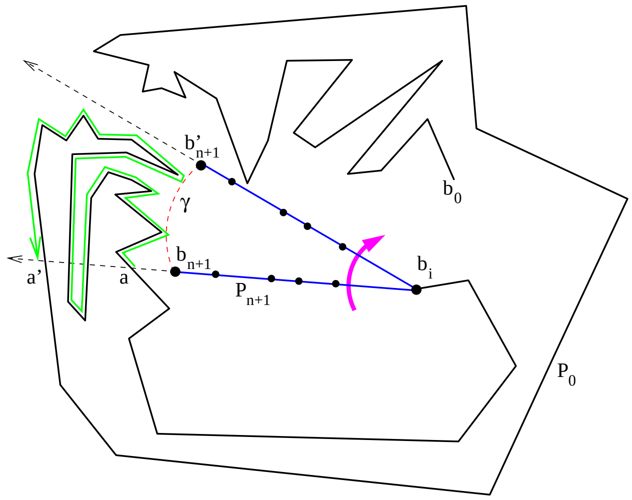

Figure 10: Case (c): Exactly one of the rays $b_i b_{n+1}$ and $b_i b'_{n+1}$ hits P_0. The walk extends from a to a' over the gray highlighted portion of P_0, painting any previously unpainted portion of it.

(c) If exactly one of the rays $\overrightarrow{b_i b_{n+1}}$ and $\overrightarrow{b_i b'_{n+1}}$ hits P_0 while the other misses P_0 (and goes off to infinity), then we define a and a' as follows. Assume that the ray $\overrightarrow{b_i b_{n+1}}$ hits P_0 (and the ray $\overrightarrow{b_i b'_{n+1}}$ misses P_0); the other case is handled similarly. We define a to be the point on the boundary of P_0 where the ray $\overrightarrow{b_i b_{n+1}}$ hits P_0 and we define a' to be the point on the boundary of P_0 where a ray from infinity in the direction $\overrightarrow{b_{n+1} b_i}$ (towards b_i) hits P_0. See Figure 10.

During the walk from a to a' along the boundary of P_0, we test each segment for intersection with the circular arc γ in time $O(1)$. Whenever we reach a portion of the boundary that is already painted, we skip over that portion, going immediately to its end. As we walk, we mark the corresponding portions over which we walk as "painted." By continuity, it is easy to verify that the painted portion of any one segment of P_0 is connected (is a single subsegment). During a walk, we charge the tests that we do for intersection with γ off to the segments that are being painted. The remainder of the justification of the algorithm is based on two simple claims:

Claim 1 *There is no need to walk back over a painted portion in order to check for intersections with an arc γ at some later stage.*

Proof: The fact that we need not walk over a painted portion testing again for intersections with γ follows from the fact that with each bend in the sequence, the length of the segment $b_i b_{n+1}$ that we are rotating goes *down* (by the length of the last link). (In other words, if the motion of the tip, b_{n+1}, sweeps an arc γ that does not reach a portion μ of the boundary of P_0 when the link $b_i b_{n+1}$ is *straight*, it cannot later be that a link $b_j b_{n+1}$ ($j > i$) can permit the tip b_{n+1} to reach the same portion μ when pivoting is done about b_j; this is a consequence of the triangle inequality.) □

Claim 2 *In testing for intersection with γ, we check enough of the chain P_0: if any part of it intersects γ, then it must lie on the portion between a and a' over which we walk.*

Proof: In case (a), there is nothing to check. In case (b), the closed Jordan curve from b_i to a (along a straight segment), then along the boundary of the simple polygon P_0 to a', then back to b_i (along a straight segment) forms the boundary of a region whose only intersection with P_0 is along the shared boundary from a to a'; thus, if γ lies within this region (i.e., does not intersect the boundary of P_0 from a to a'), then γ does not intersect any other portion of P_0. In case (c) we argue similarly, but we use the Jordan region defined by the segment from b_i to a, the boundary of P_0 from a to a', the ray from a' to infinity (in the direction of $\overrightarrow{b_i b_{n+1}}$), then the reverse of the ray $\overrightarrow{b_i b'_{n+1}}$ back to b_i. □

The total time for walking along the chain P_0 can be charged off to the vertices of P, resulting in time $O(n)$ for tests of intersection with arcs γ, exclusive of the ray shooting time. The final time bound is then dominated by the time to perform n straight ray shooting queries in a dynamic data structure for the changing polygonal chain P_0; these ray shooting queries are utilized both in testing for intersection with the segment $b_i b'_{n+1}$ and in determining the points a and a' that define the walk. These ray-shooting queries and updates are done in time $O(\log^2 n)$ each, using existing techniques ([15]), leading to the claimed overall time bound. □

Next, we turn to two other important classes of permutations. We say that a permutation is an *outwards folding sequence* (resp., *inwards folding sequence*) if at any stage of the folding, the set of bends that have been completed is a subinterval, $b_i, b_{i+1}, \ldots, b_j$ (resp., a pair of intervals $b_1, b_2, \ldots, b_i$ and $b_j, b_{j+1}, \ldots, b_n$); thus, the next bend to be performed is either b_{i-1} or b_{j+1} (resp., b_{i+1} or b_{j-1}). Inwards and outwards folding sequences model the case in which the bending operations must be done sequentially along the wire; this is a constraint with some forming machines. See Figure 11. The identity permutation is a folding sequence that is a special case of both an inwards and an outwards folding sequence.

We show that one can efficiently search for a folding sequence that is inwards or outwards. Our algorithms are based on dynamic programming.

First, consider the case of outwards folding sequences. We keep track of the *state* as the pair (i, j) representing the interval of bends $(b_i, b_{i+1}, \ldots, b_j)$ already completed. We construct a graph

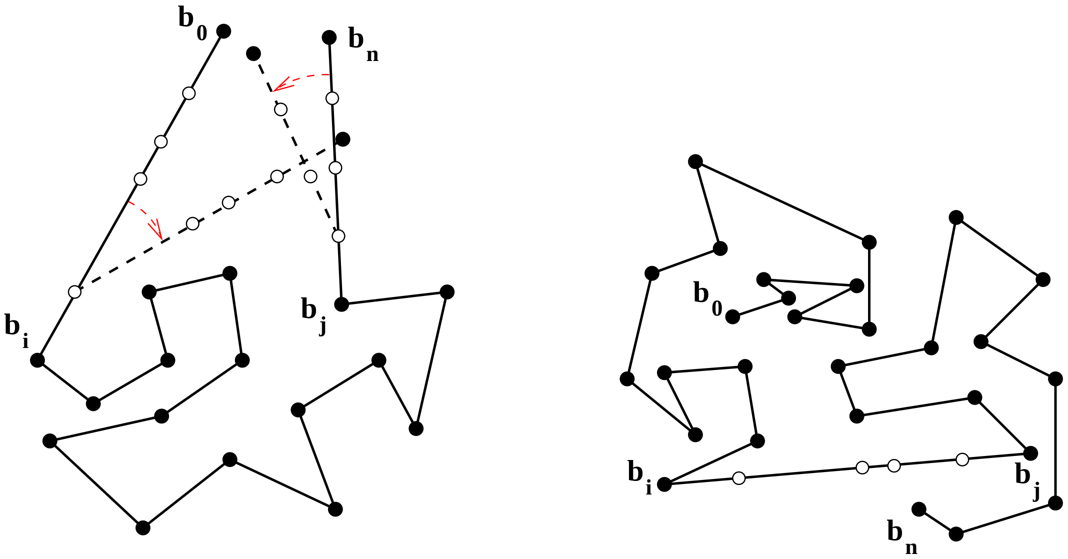

Figure 11: An intermediate state (i, j) in the folding of an outwards (left) and an inwards (right) folding sequence. For the outwards folding sequence on the left, the next bend is either b_{i-1} or b_{j+1}; the new positions of the chain are shown dashed. For the inwards folding sequence on the right, the next bend is either b_{i+1} or b_{j-1}.

$\mathcal{G}$ whose $O(n^2)$ nodes are the states (i, j) (with $1 \leq i < j \leq n$) and whose edges link states that correspond to the action of completing a bend at b_{i-1} or b_{j+1} (if the starting state is (i, j)). Thus, each node has constant degree. Our goal is to determine if there is a path in this graph from some (i, i) to $(1, n)$. This is done in $O(n^2)$ time once we have the graph constructed. (Alternatively, we can construct the graph as we search the graph for a path.) In order to construct the graph, we need to test whether bend b_{i-1} or b_{j+1} can be performed without intersecting the folded chain, P', linking b_{i-1} to b_{j+1}. This is done in a manner very similar to that we described above for the case of identity permutations: we perform ray-shooting queries in time $O(\log^2 n)$ and then use a "painting" procedure to keep track of the states of $2n - 2$ "walks" that determine circular-arc ray shooting queries. In particular, there is a separate painting procedure corresponding to each of the $n - 1$ choices of i and to each of the $n - 1$ choices of j. For example, for a fixed choice of i, the painting procedure will consider each of the possible bends $b_{i+1}, \ldots, b_n$ in order, allowing us to amortize the cost of checking for intersections with the circular arc γ associated with each bend. In total, the cost of the walks is $O(n^2)$, while there may also be $O(n^2)$ ray shooting queries (in a dynamically changing polygon). Thus, the total cost is dominated by the ray shooting queries, giving an overall time bound of $O(n^2 \log n)$.

For the case of an inwards folding sequence, we build a similar state graph and search it. However, the cost of testing if a bend is feasible is somewhat higher, as we do not yet have an efficient procedure for testing the foldability of a polygonal chain. (Our painting procedures exploit the fact that the link being folded is straight.) Thus, we apply the relatively naive method of testing feasibility given in Lemma 4.2, at a cost of $O(n \log n)$ per test (which potentially improves to $O(n)$ time, if our conjecture mentioned in the remark after the Lemma is true). Thus, the overall cost of the algorithm is dominated by the $O(n^2)$ feasibility tests, at a total cost of $O(n^3 \log n)$. In summary, we have:

Theorem 4.2 *In time $O(n^2 \log^2 n)$ one can determine if there is an outwards folding sequence; in time $O(n^3 \log n)$ one can determine if there is an inwards folding sequence.*

5 Conclusion

We conclude with some open problems that are suggested by our work:

(1) Is the bend sequencing problem for wire folding *strongly* NP-complete, or is there a pseudopolynomial-time algorithm? If not in wire bending, is it strongly NP-complete for the 3-dimensional sheet metal folding problem?

(2) Is it NP-hard to decide if a polygonal chain in three dimensions can be straightened? In [6] simple examples of locked chains in three dimensions are shown; can these be extended to a hardness proof for the decision problem?

(3) If we consider only structures that are manufacturable using a punch and die on a press brake (which significantly limits the set of foldable bends, since the punch and die must be accessible), what is the complexity of the bend sequencing problem? The wire-bending version of this question may be modeled as requiring that each joint that we bend must lie on a line that crosses the part at the joint, reaching to infinity in both directions without intersecting the current structure. (This models the need for the punch and die to be accessible at the bend point.)

(4) Can the foldability of a permutation be decided in subquadratic time for wire bending? At issue is designing a dynamic data structure that will permit efficient (sublinear) queries for the foldability of a vertex; this seems to be an interesting question in its own right.

Acknowledgments

We thank S. Skiena for valuable input in the early stages of this research and, in particular, for contributing ideas to the hardness proof of Theorem 3.1. E. Arkin acknowledges support from the National Science Foundation (CCR-9732221) and HRL Laboratories. S. Fekete acknowledges support from the National Science Foundation (ECSE-8857642,CCR-9204585) during his time at Stony Brook (1992-93), when this research was initiated. J. Mitchell acknowledges support from HRL Laboratories, the National Science Foundation (CCR-9732221), NASA Ames Research Center, Northrop-Grumman Corporation, Sandia National Labs, Seagull Technology, and Sun Microsystems.

References

[1] E. M. Arkin, M. A. Bender, E. D. Demaine, M. L. Demaine, J. S. Mitchell, S. Sethia, and S. S. Skiena. Recognizing simply foldable origami. Manuscript (submitted), University at Stony Brook, 2000.

[2] B. Aronov and J. O'Rourke. Nonoverlap of the star unfolding. *Discrete Comput. Geom.*, 8:219–250, 1992.

[3] V. Ayyadevara, D. Bourne, K. Shimada, and R. H. Sturges. Determining near optimal interference-free polyhedral configurations for stacking. *Proceedings of IEEE International Symposium on Assembly and Task Planning*, pages 286–293, July 1999.

[4] M. Bern, E. D. Demaine, D. Eppstein, and E. Kuo. Ununfoldable polyhedra. In *Proc. 11th Canad. Conf. Comput. Geom.*, pages 13–16, 1999. Full version: LANL archive paper number cs.CG/9908003.

[5] M. Bern and B. Hayes. The complexity of flat origami. In *Proc. 7th ACM-SIAM Sympos. Discrete Algorithms*, pages 175–183, 1996.

[6] T. Biedl, E. Demaine, M. Demaine, S. Lazard, A. Lubiw, J. O'Rourke, M. Overmars, S. Robbins, I. Streinu, G. Toussaint, and S. Whitesides. Locked and unlocked polygonal chains in 3D. In *Proc. 10th ACM-SIAM Sympos. Discrete Algorithms*, pages 866–867, Jan. 1999.

[7] T. Biedl, E. Demaine, M. Demaine, A. Lubiw, J. O'Rourke, M. Overmars, S. Robbins, I. Streinu, and S. Whitesides. On reconfiguring tree linkages: Trees can lock. In *Proc. 10th Canad. Conf. Comput. Geom.*, pages 4–5, 1998.

[8] T. Biedl, E. Demaine, M. Demaine, A. Lubiw, J. O'Rourke, M. Overmars, S. Robbins, and S. Whitesides. Unfolding some classes of orthogonal polyhedra. In *Proc. 10th Canad. Conf. Comput. Geom.*, pages 70–71, 1998. Fuller version in Elec. Proc. http://cgm.cs.mcgill.ca/cccg98/proceedings/welcome.hml.

[9] D. Bourne and C.-H. Wang. Design and manufacturing of sheet metal parts: Using features to resolve manufacturability problems. In A. Busnaina, editor, *Computer in Engineering 1995*, pages 745–753. ASME, New York, 1995.

[10] R. Connelly, E. Demaine, and G. Rote. Every polygon can be untangled. In *Proc. 41st Annu. IEEE Sympos. Found. Comput. Sci.*, pages 432–442, 2000.

[11] L. J. de Vin, J. de Vries, A. H. Streppel, and H. J. J. Kals. PART-S, a CAPP system for small batch manufacturing of sheet metal components. In *Proc. of the 24th CIRP International Seminar on Manufacturing Systems*, pages 171–182, 1992.

[12] E. D. Demaine, M. L. Demaine, and A. Lubiw. Folding and cutting paper. In *Proc. Japan Conf. Discrete Comput. Geom.*, Lecture Notes Comput. Sci. Springer-Verlag, 1998.

[13] E. D. Demaine, M. L. Demaine, and A. Lubiw. Folding and one straight cut suffice. In *Proc. 10th Annu. ACM-SIAM Sympos. Discrete Alg.*, pages 891–892, Jan. 1999.

[14] E. D. Demaine, M. L. Demaine, and J. S. B. Mitchell. Folding flat silhouettes and wrapping polyhedral packages: New results in computational origami. *Comput. Geom. Theory Appl.*, 16(1):3–21, 2000.

[15] M. T. Goodrich and R. Tamassia. Dynamic ray shooting and shortest paths in planar subdivisions via balanced geodesic triangulations. *J. Algorithms*, 23:51–73, 1997.

[16] S. K. Gupta, D. A. Bourne, K. H. Kim, and S. S. Krishnan. Automated process planning for sheet metal bending operations. *J. Manufacturing Systems*, 17(5):338–360, 1998.

[17] T. Hull. On the mathematics of flat origamis. *Congr. Numer.*, 100:215–224, 1994.

[18] K. K. Kim, D. Bourne, S. Gupta, and S. S. Krishnan. Automated process planning for robotic sheet metal bending operations. *Journal of Manufacturing Systems*, 17(5):338–360, September 1998.

[19] R. J. Lang. Mathematical algorithms for origami design. *Symmetry: Culture and Science*, 5(2):115–152, 1994.

[20] R. J. Lang. A computational algorithm for origami design. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, pages 98–105, 1996.

[21] W. J. Lenhart and S. H. Whitesides. Turning a polygon inside-out. In *Proc. 3rd Canad. Conf. Comput. Geom.*, pages 66–69, Aug. 1991.

[22] W. J. Lenhart and S. H. Whitesides. Reconfiguring closed polygonal chains in Euclidean d-space. *Discrete Comput. Geom.*, 13:123–140, 1995.

[23] L. Lu and S. Akella. Folding cartons with fixtures: A motion planning approach. In *Proc. 1999 IEEE International Conference on Robotics and Automation, Detroit, MI*, pages 1570–1576, 1999.

[24] A. Lubiw and J. O'Rourke. When can a polygon fold to a polytope? Technical Report 048, Dept. Comput. Sci., Smith College, June 1996. Presented at AMS Conf., 5 Oct. 1996.

[25] M. Namiki and K. Fukuda. Unfolding 3-dimensional convex polytopes: A package for Mathematica 1.2 or 2.0. Mathematica Notebook, Univ. of Tokyo, 1993.

[26] J. O'Rourke. Computational geometry column 33. *Internat. J. Comput. Geom. Appl.*, 8:381–384, 1999. Also in SIGACT News, 29(2):12-16 (1998), Issue 107.

[27] J. O'Rourke. Folding and unfolding in computational geometry. In J. Akiyama, M. Kano, and M. Urabe, editors, *Proc. Japan Conf. Discrete & Computational Geometry, Tokyo, Japan, December 9-12, 1998*, number 1763 in Lecture Notes in Computer Science, pages 258–266. Springer-Verlag, 2000.

[28] N. Pei and S. Whitesides. On the reachable regions of chains. In *Proc. 8th Canad. Conf. Comput. Geom.*, pages 161–166, 1996.

[29] N. Pei and S. Whitesides. On folding rulers in regular polygons. In *Proc. 9th Canad. Conf. Comput. Geom.*, pages 11–16, 1997.

[30] C. Schevon and J. O'Rourke. A conjecture on random unfoldings. Technical Report JHU-87/20, Johns Hopkins Univ., Baltimore, MD, July 1987.

[31] I. Streinu. A combinatorial approach to planar non-colliding robot arm motion planning. In *Proc. 41st Annu. IEEE Sympos. Found. Comput. Sci.*, pages 443–453, 2000.

[32] M. van Kreveld, J. Snoeyink, and S. Whitesides. Folding rulers inside triangles. *Discrete Comput. Geom.*, 15:265–285, 1996.

[33] C.-H. Wang. *Manufacturability-driven decomposition of sheet metal products.* PhD thesis, Carnegie Mellon University, The Robotics Institute, 1997.

[34] C.-H. Wang and D. Bourne. Concurrent decomposition for sheet metal products. In *ASME Design Engineering Technical Conference*, Sacramento, September 14-17 1997.

[35] C.-H. Wang and R. H. Sturges. Bendcad: a design system for concurrent multiple representations of parts. *Journal of Intelligent Manufacturing*, 7:133–144, 1996.

Reports from the group

"Combinatorial Optimization and Graph Algorithms"

of the Department of Mathematics, TU Berlin

711/2001 *Esther M. Arkin, Michael A. Bender, Sándor P. Fekete, Joseph S. B. Mitchell, and Martin Skutella:* The Freeze-Tag Problem: How to Wake Up a Swarm of Robots

710/2000 *Esther M. Arkin, Sándor P. Fekete, and Joseph S. B. Mitchell:* Algorithms for Manufacturing Paperclips and Sheet Metal Structures

705/2000 *Ekkehard Köhler:* Recognizing Graphs without Asteroidal Triples

704/2000 *Ekkehard Köhler:* AT-free, coAT-free Graphs and AT-free Posets

702/2000 *Frederik Stork:* Branch-and-Bound Algorithms for Stochastic Resource-Constrained Project Scheduling

700/2000 *Rolf H. Möhring:* Scheduling under uncertainty: Bounding the makespan distribution

698/2000 *Sándor P. Fekete, Ekkehard Köhler, and Jürgen Teich:* More-dimensional packing with order constraints

697/2000 *Sándor P. Fekete, Ekkehard Köhler, and Jürgen Teich:* Extending partial suborders and implication classes

696/2000 *Sándor P. Fekete, Ekkehard Köhler, and Jürgen Teich:* Optimal FPGA module placement with temporal precedence constraints

695/2000 *Sándor P. Fekete, Henk Meijer, André Rohe, and Walter Tietze:* Solving a "hard" problem to approximate an "easy" one: heuristics for maximum matchings and maximum Traveling Salesman Problems

694/2000 *Esther M. Arkin, Sándor P. Fekete, Ferran Hurtado, Joseph S. B. Mitchell, Marc Noy, Vera Sacristánm and Saurabh Sethia:* On the reflexivity of point sets

693/2000 *Frederik Stork and Marc Uetz:* On the representation of resource constraints in project scheduling

691/2000 *Martin Skutella and Marc Uetz:* Scheduling precedence constrained jobs with stochastic processing times on parallel machines

689/2000 *Rolf H. Möhring, Martin Skutella, and Frederik Stork:* Scheduling with AND/OR precedence constraints

685/2000 *Martin Skutella:* Approximating the single source unsplittable min-cost flow problem

684/2000 *Han Hoogeveen, Martin Skutella, and Gerhard J. Woeginger:* Preemptive scheduling with rejection

683/2000 *Martin Skutella:* Convex quadratic and semidefinite programming relaxations in Scheduling

682/2000 *Rolf H. Möhring and Marc Uetz:* Scheduling scarce resources in chemical engineering

681/2000 *Rolf H. Möhring:* Scheduling under uncertainty: optimizing against a randomizing adversary

680/2000 *Rolf H. Möhring, Andreas S. Schulz, Frederik Stork, and Marc Uetz:* Solving project scheduling problems by minimum cut computations (Journal version for the previous Reports 620 and 661)

674/2000 *Esther M. Arkin, Michael A. Bender, Erik D. Demaine, Sándor P. Fekete, Joseph S. B. Mitchell, and Saurabh Sethia:* Optimal covering tours with turn costs

669/2000 *Michael Naatz:* A note on a question of C. D. Savage

667/2000 *Sándor P. Fekete and Henk Meijer:* On geometric maximum weight cliques

666/2000 *Sándor P. Fekete, Joseph S. B. Mitchell, and Karin Weinbrecht:* On the continuous Weber and k-median problems

664/2000 *Rolf H. Möhring, Andreas S. Schulz, Frederik Stork, and Marc Uetz:* On project scheduling with irregular starting time costs

661/2000 *Frederik Stork and Marc Uetz:* Resource-constrained project scheduling: from a Lagrangian relaxation to competitive solutions

Reports may be requested from: Hannelore Vogt-Möller
Fachbereich Mathematik, MA 6–1
TU Berlin
Straße des 17. Juni 136
D-10623 Berlin – Germany

e-mail: moeller@math.TU-Berlin.DE

Reports are also available in various formats from

`http://www.math.tu-berlin.de/coga/publications/techreports/`

and via anonymous ftp as

`ftp://ftp.math.tu-berlin.de/pub/Preprints/combi/Report-`*number-year*`.ps`