Journal of Mathematical Chemistry Vol. 27, No. 4, 2000

Geometric and computational aspects of polymer reconfiguration

Michael Soss and Godfried T. Toussaint

School of Computer Science, McGill University, Montreal, QC, Canada H3A 2A7

Received 1 May 2000

We examine a few computational geometric problems concerning the structures of polymers. We use a standard model of a polymer, a polygonal chain (path of line segments) in three dimensions. The chain can be reconfigured in any manner as long as the edge lengths and the angles between consecutive edges remain fixed, and no two edges cross during the motion. We discuss preliminary results on the following problems.

Given a chain, select some interior edge $\overline{uv}$, defining two subchains which are adjacent to $\overline{uv}$. We keep the two subchains individually rigid and rotate one around $\overline{uv}$ while leaving the other fixed in space, while maintaining the vertex-angles at $\overline{uv}$. We call this motion an *edge spin* at $\overline{uv}$. An $O(n^2)$ algorithm for this problem is given as well as an $\Omega(n \log n)$ lower bound on the time complexity.

In determining whether a chain can be reconfigured from one conformation to another, it is useful to consider reconfiguring through some canonical conformation. In our three-dimensional case, the most obvious choice is to flatten a chain into the plane. However, we demonstrate that determining if a given chain can be reconfigured into the plane without self-intersecting is NP-hard, even if the restriction that it must lie monotonically is added. We then provide an $O(n)$ algorithm to decide if a chain has a non-crossing convex coil conformation (where all angles turn in the same direction), although we cannot yet decide if a sequence of motions to reconfigure a chain into a convex coil conformation exists.

KEY WORDS: polymer conformations, polymer motions, macromolecule conformations, edge spin, flattening

1. Introduction

During the past several years, questions regarding reconfiguring chains, polygons, and trees have received widespread interest in the computational geometry community and literature [1]. Most of these questions deal with unfolding a linkage, in other words, straightening chains, convexifying polygons, or flattening trees, all while maintaining the lengths of all edges and not allowing self-intersections during the motion. The idea is that if two conformations of the same chain can each be reconfigured into the same canonical conformation (a straight chain, convex polygon, or flattened tree), then because the motions are reversible, it is possible to reconfigure one conformation into the other via the canonical one.

0259-9791/00/1200-0303$18.00/0

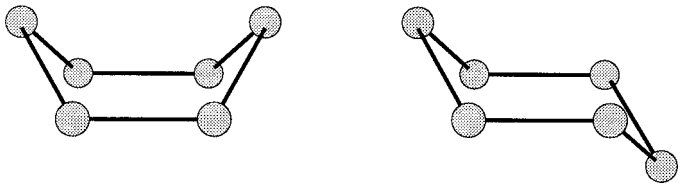

Figure 1. The two possible conformations of cyclohexane, the six-atom carbon ring C_6H_{12}. Only the carbon atoms are drawn; the hydrogen atoms are omitted. Left: the "boat" conformation, which has enough freedom to move to other similar boat conformations. Right: the "chair" conformation, which is rigid.

The most famous problem in this field, which stumped several mathematicians for almost ten years, was just recently closed: can all planar chains (and polygons) be straightened (or convexified) in the plane? This question was recently answered in the affirmative by Connelly et al. [2]. Other results include demonstrating that some three-dimensional chains are unstraightenable [3–5], and that all chains can be straightened in four dimensions [6].

The applications of these questions are numerous, including robot arm path planning [7] and wire and sheet metal bending [8], but the application with which we are concerned here is of mapping conformation spaces of molecules. The chemistry, biology, and physics communities have long been studying polymers, molecules of often several thousand atoms in length, in order to understand how they fold and to predict their final conformation. The objectives are rather diverse, including straightening polymers to make more resilient rubber [9], efficient drug design [10], and understanding the structure of DNA [11]. Researchers have been using Monte Carlo techniques to approximate the minimum energy polymer foldings [12] with some degree of success, although this problem has been recently proven to be NP-hard [13]. Due to these difficulties, physicists have also taken to simplifying the problem by imposing restrictions, most notably by restricting the protein to lie on a lattice as opposed to a continuous space [14], although finding the minimum energy conformation on the lattice is also NP-hard [15].

In 1987, Richard Randell designed a mathematical framework for geometrically and topologically mapping the conformation space of molecules [16]. His research provided topological explanations of chemical phenomena, such as demonstrating that the conformation space of cyclohexane, the six-atom carbon ring C_6H_{12}, has two distinct path-connected components [17]. (Randell later discovered that the same calculations had been made from an algebraic viewpoint a century earlier [18].) Cyclohexane takes the shape of an equilateral hexagon with all bond angles of about 109.5°, as illustrated in figure 1. Randell proves several other interesting results, including that a carbon ring of fewer than 11 atoms must be unknotted.

In this paper, we use a model similar to the one used by Randell and several others [10,13] to solve problems concerning polymers. We consider a polymer to be a three-dimensional chain with fixed edge lengths and fixed angles (vertex-angles) between consecutive edges, modelling the fixed bond lengths and bond angles between atoms. Any such chain can be reconfigured by a *vertex-angle preserving motion*, meaning that the edges of the chain are free to move about so long as the edge lengths and the vertex-

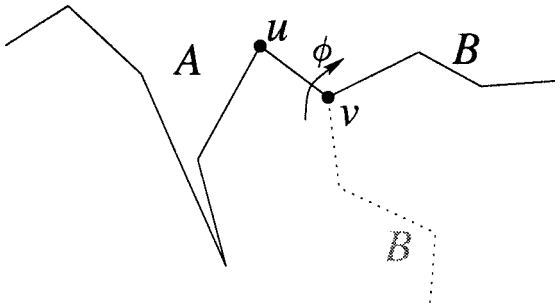

Figure 2. An edge spin of angle ϕ at $\overline{uv}$. Note that due to the fixed vertex-angle at v, B spins about an axis of rotation collinear with the edge $\overline{uv}$.

angles between consecutive edges remain fixed. Furthermore, the chain is not permitted to self-intersect throughout the motion. An example of a vertex-angle preserving motion of a chain is illustrated in figure 2.

We begin in section 2 with a discussion of arguably the simplest form of vertex-angle preserving motion, an edge spin. We continue in section 3 to the question of reconfiguring one conformation to another via the canonical conformation of a non-intersecting planar chain. Section 4 concludes and lists directions for future work.

2. The Edge Spin problem

We start our discussion on reconfiguring polygonal chains with fixed vertex-angles with the following simple motion. Given a chain, select some interior edge $\overline{uv}$. This defines two subchains A and B such that $u \in A$ and $v \in B$. We can keep the two subchains individually rigid and, leaving A fixed in space, rotate B around $\overline{uv}$ (while maintaining the fixed vertex-angle at v) by some angle ϕ. We call this motion an *edge spin* of angle ϕ at $\overline{uv}$. An illustration is in figure 2.

The remainder of this section deals with the following problem.

Problem 1 (Edge Spin). Given a three-dimensional polygonal chain, a selected edge $\overline{uv}$, and an angle ϕ, can we perform an edge spin of angle ϕ at $\overline{uv}$ without causing the chain to self-intersect?

We now sketch an algorithm to answer the above question, followed by a proof of a lower bound on the complexity of the problem.

2.1. An algorithm to solve the Edge Spin problem

We give a quadratic time algorithm to solve the Edge Spin problem.

Theorem 1. The Edge Spin problem is solvable in $O(n^2)$ time and $O(n)$ space, where n is the number of edges in the chain.

Proof. The main idea is to pretend to spin one of the subchains around $\overline{uv}$ completely (by angle 2π), and examine all self-intersections which would occur along the way. We compute the angle of rotation at which intersection occurs during the motion, and if none occur before angle ϕ is reached, the edge spin can be performed.

To compute the intersections during the spin of angle 2π, consider any plane P incident to $\overline{uv}$. Without considering intersections between the two subchains, pretend to spin subchain A around $\overline{uv}$, and trace where the subchain would sweep through P during the motion. Do the same for subchain B. Since a line segment rotating about a line sweeps out a portion of a hyperboloid, we have two arrangements of $O(n)$ hyperbolic arcs each in the plane.

Note that spinning subchain A is equivalent to spinning subchain B, modulo a rotation of the entire chain, because both rotations are about the same axis, edge $\overline{uv}$. Thus if executing an edge spin causes an intersection, there will be an intersection of the two arrangements. We can compute if any arcs corresponding to A intersect an arc corresponding with B with brute force in $O(n^2)$ time. Faster yet, we can also use the intersection algorithm of Bentley and Ottmann [19]. This yields a time complexity of $O(n \log n + k)$, where k is the sum of the self-intersections in each arrangement and the intersections between arrangements. (Although k could be as large as $\Theta(n^2)$ in the worst case, it would prove faster in practice than brute force.) Regardless of the method used to detect intersections, as each intersecting pair of arcs is detected, we determine the angle of rotation required before such an intersection would occur during the edge spin. Thus, the entire algorithm finishes in $O(n^2)$ time. Furthermore, once we detect a pair of intersecting arcs and determine the angle of rotation required, we will never again need to examine that intersection. Therefore, we do not need to store intersections already detected, so we require only linear space by using brute force or Bentley and Ottmann. □

Rather than asking the Edge Spin problem for a particular ϕ, one may wish to know if a complete edge spin (of angle 2π) can be performed. We show that this problem may be easier by demonstrating a faster algorithm.

Theorem 2. For $\phi = 2\pi$, the Edge Spin problem is solvable in deterministic time $O(n2^{\alpha(n)} \log^2 n)$ and space $O(n2^{\alpha(n)})$ and in expected time $O(n2^{\alpha(n)} \log n)$ and space $O(n2^{\alpha(n)})$, where $\alpha(n)$ is the slow-growing inverse of the familiar Ackermann function.

Proof. We repeat the above technique as in the proof of theorem 1. However, for this choice of ϕ, we do not have to check to see which intersection occurs first in our arrangements; if any intersection at all exists between the arrangements, the edge spin cannot be performed. Given two arrangements of hyperbolic arcs which pairwise intersect at most twice, we can detect an intersection between the arrangements in the stated deterministic time and space using algorithms of Agarwal and Sharir [20] and of Guibas et al. [21]. By replacing the latter with the algorithm of Chazelle et al. [22], we can achieve the stated expected time and space. □

2.2. A lower bound for the Edge Spin problem

In this subsection, we prove the following result.

Theorem 3. The time complexity of the Edge Spin problem on a chain of n edges is $\Omega(n \log n)$ under the algebraic decision tree model of computation.

We prove theorem 3 with a reduction from Element Uniqueness, which states as follows.

Problem 2 (Element Uniqueness). Given a set $S = \{s_1, \ldots, s_n\}$, is every element unique? In other words, does $i \neq j$ imply that $s_i \neq s_j$?

The time complexity of the Element Uniqueness problem is known to have an $\Omega(n \log n)$ lower bound under the algebraic decision tree model, as shown by Ben-Or [23]. Given a set S, we create a polygonal chain and select an edge in linear time such that answering the Edge Spin problem also answers Element Uniqueness on S.

2.2.1. Construction of a tree

Let N be the number of elements in the set S. We consider all elements to be strictly positive. If this is not the case, then we can add some suitable integer to each element in the set (which can be done in linear time).

As the chain we will build is difficult to visualize, it will be far easier first to explain how a tree can be constructed in the xz-plane to answer our query. We will not address the issue of constructing the tree; rather, we will later show how to construct a chain in linear time which behaves in an identical manner.

For the purposes of terminology, we discuss the tree in three parts, the "base" and the left and right "gadgets". The main idea is that the base and the left gadget will remain stationary while the right gadget spins about an edge on the base. If and only if element uniqueness holds, the two gadgets will not collide.

We begin our construction of the tree by drawing the *base*, a three-edge chain from $(0, 0, 0)$ to $(0, 0, -1)$ to $u = (N + 3/2, 0, -1)$ to $v = (N + 3/2, 0, 0)$.

For the left gadget, we first draw a vertical edge from $(0, 0, 0)$ to $(0, 0, \max\{S\})$. We call this edge the *stem* of the gadget. For each s_i, we connect a new edge $\langle(0, 0, s_i), (i, 0, s_i)\rangle$ to the stem. In other words, from the stem we build an edge extending to the right at height s_i, of length i. If s_i is not unique, then we shall have two overlapping edges, but for simplicity sake, we allow the intersection. (We will worry about this later when we create the chain.)

We create the right gadget similarly, except each edge has length $N - i + 1$. We first draw the stem from $(N + 3/2, 0, 0)$ to $(N + 3/2, 0, \max\{S\})$. For each s_i, we connect a new edge $\langle(N + 3/2, 0, s_i), (2N + 3/2, 0, s_i)\rangle$ to the stem. In other words, from the stem we build an edge extending to the right at height s, of length $N - i + 1$. (Again, ignore the problems of intersecting if s_i is not unique.)

Examples are shown in figures 3 and 4.

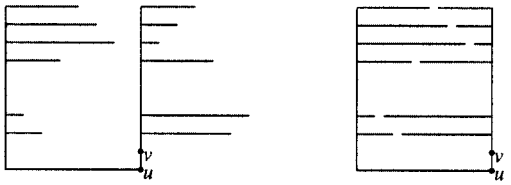

Figure 3. Left: the tree constructed for a set with no repeated elements, {2, 1, 5, 8, 7, 6}. The edge spin occurs at $\overline{uv}$. Right: the tree in mid-spin, after a rotation of π.

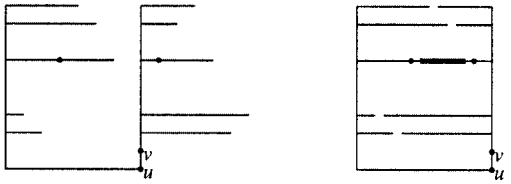

Figure 4. Left: the tree constructed for a set with a repeated element, {2, 1, **5**, 8, 7, **5**}. The edge spin occurs at $\overline{uv}$, and the endpoints of the overlapped edges are encircled. Right: the tree in mid-spin, after a rotation of π. The bold segment is where the collision occurs.

We now perform a complete edge spin (of angle 2π) at $\overline{uv}$. Since the tree is orthogonal, every edge will stay at the same height during the motion. Therefore, the only chance for collision is that two edges of the same height, and thus corresponding to elements of the same value, collide. Note that the two stems are distance $N + 3/2$ apart; therefore, if and only if the total lengths of two edges of the same height on opposite gadgets are at least $N + 3/2$ will there be a collision. Since the lengths of the two edges corresponding to the same vertex sum to $N + 1$, we can only have a collision if two edges corresponding to different elements collide. Note that this will necessarily happen if there exist two elements s_i and s_j $(i < j)$ with the same value. Then the left gadget edge corresponding to s_j has length j, and the right gadget edge corresponding to s_i has length $N - i$. Since $i < j$, the two edges have total length at least $N + 2$. Since $s_i = s_j$, the edges are at the same height, and we have a collision. Next we create a chain with the same properties.

2.2.2. Construction of the chain

We now create a three-dimensional chain which does not self-intersect and behaves exactly like the tree. In fact, from the viewpoint of $y = +\infty$, the tree and the chain look identical.

The base is the same, a three-edge subchain from $(0, 0, 0)$ to $(0, 0, -1)$ to $u = (N+3/2, 0, -1)$ to $v = (N+3/2, 0, 0)$. For the left gadget, create the first edge similarly to the tree; draw an edge up from $(0, 0, 0)$ to $(0, 0, s_1)$, and then horizontally to $(1, 0, s_1)$. To avoid intersections in the chain, we exploit the third dimension, y. We draw an edge back to a point just in front of the stem, at $(0, -1/(4N), s_1)$. Now we are free to draw a vertical edge from $(0, -1/(4N), s_1)$ to $(0, -1/(4N), s_2)$, which places us at height s_2 in preparation for the next two edges, to $(2, 0, s_2)$ and back to $(0, -2/(4N), s_2)$. We

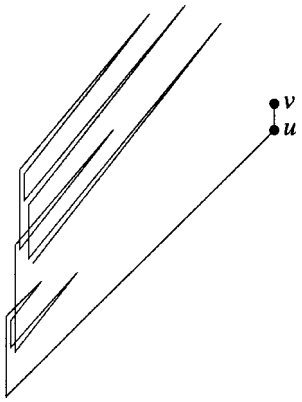

Figure 5. Oblique view of the base and left gadget, for the set (2, 1, 5, 8, 7, 5). Note there are two pairs of edges at height $z = 5$, corresponding to the repeated element. (The edge spin occurs at $\overline{uv}$.)

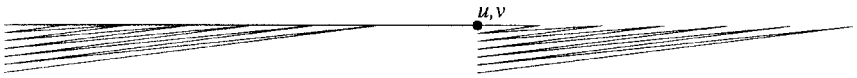

Figure 6. View of the chain from above ($z = +\infty$). Not to-scale; y-direction is magnified for clarity. (The edge spin occurs at $\overline{uv}$.)

continue this pattern drawing edges to $(0, -(i-1)/(4N), s_i)$, $(i, 0, s_i)$, $(0, -i/(4N), s_i)$, and so on until the gadget is complete. We construct the right chain in the exact same fashion, except that we reverse the labelling of the elements from $s_1, \ldots, s_n$ to $s_n, \ldots, s_1$.

An illustration of the left gadget and the base is in figure 5; a bird's eye view of the whole chain is in figure 6.

We perform a complete edge spin (of angle 2π) at $\overline{uv}$. Note that since all edges are within radius N of their respective gadget stem, and the stems are $N + 3/2$ apart, no part of either gadget will enter a cylinder of radius $1/2$ around the stem of the opposite gadget. Since all the vertical edges are contained in a cylinder of radius $1/4$ around the stems, only the horizontal edges corresponding to elements in the set S can collide. Thus, the behavior of the chain mimics that of the tree exactly. The entire chain is illustrated in figure 7.

3. Reconfiguring chains into the plane

We may also wish to consider not just a single edge spin, but rather whether or not we can reconfigure a chain from a given conformation to a certain goal conformation. An intuitive approach is to ask if both can be reconfigured into some canonical conformation. If so, then because the motions are reversible, one can reconfigure the chain from the given conformation to the canonical conformation, and then reconfigure from the canonical conformation to the goal conformation. In the computational geometry literature where the angles between edges are free to change, the canonical conformation for chains is a straight segment. Obviously, this is unachievable for chains with

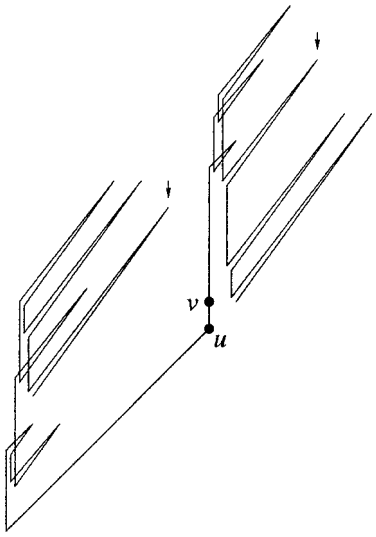

Figure 7. Oblique view of the chain for the set $(2, 1, 5, 8, 7, 5)$. If an edge spin is performed at $\overline{uv}$, the edges indicated at the arrows, which correspond to the repeated element 5, will collide.

fixed vertex-angles between edges, so a natural question to ask is whether or not we can reconfigure a three-dimensional chain such that it lies flat in the plane without crossing itself. (We do not claim that one planar conformation can be reconfigured to any other, but rather that simply reconfiguring a chain into the plane is a good start.) In this section, we discuss this problem as well as two variants.

3.1. Non-crossing planar conformations

We focus on the following.

Problem 3 (Planar Flattening). Given a polygonal chain in three dimensions, does there exist a sequence of vertex-angle preserving motions which place the chain into a non-crossing planar conformation?

As mentioned above, an efficient algorithm for this problem could prove to be very useful. However, we demonstrate Planar Flattening to be NP-hard via a reduction from Partition, which reads as follows.

Problem 4 (Partition). Given a set of integers S, can it be partitioned into two disjoint sets S_a and S_b such that $\sum(s\colon\ s \in S_a) = \sum(s\colon\ s \in S_b)$?

Theorem 4. Planar Flattening is NP-hard.

Proof. Given a set S, we show that in polynomial time we can create a chain which can be reconfigured into the plane if and only if the set has a partition.

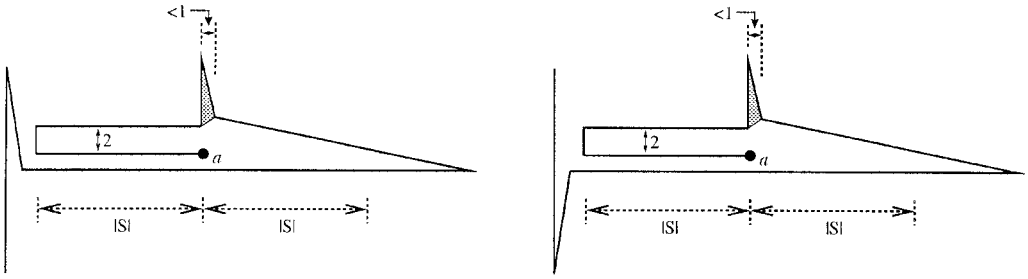

Figure 8. The two possible planar conformations of the *cage* of the chain constructed in the NP-hardness reduction. The *door* is the shaded triangle.

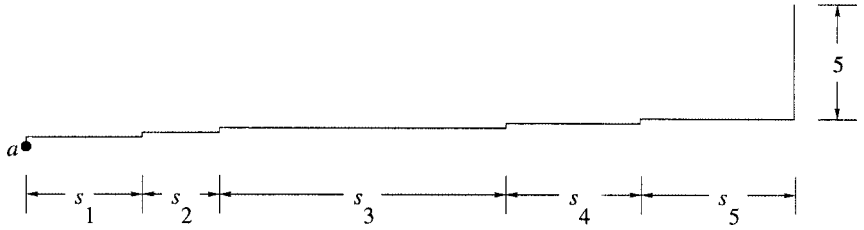

Figure 9. The subchain starting at vertex a.

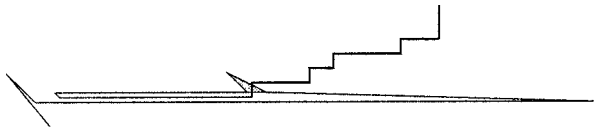

Figure 10. The chain constructed in the proof of theorem 4.

We begin by creating, in the plane, either of the chains in figure 8, where $|S|$ is the sum of the absolute values of all elements of S. An enumeration will show that these are the only two planar conformations possible for this chain. We refer to this chain as the *cage* and the shaded triangle as the *door*. We then add to vertex *a* (as labelled in figure 8) a long subchain ending in an edge which we call the *key*. The general idea is to fashion the chain such that any flattened conformation must have the key placed inside the door, and then show that the key fits if and only if a partition exists for the set S.

We now create the remainder of the chain from vertex a. We build an orthogonal subchain, such that every vertex-angle between edges is $\pi/2$. We start at vertex a, and for every element $s_i \in S$, place an edge moving upward (normal to the plane of the cage) of length $1/n$ followed by an edge e_i of length s_i in a plane parallel to the cage, and repeat. We then place one more edge, an upward edge of length 5, which we call the *key*. The subchain is illustrated in figure 9; the entire chain is illustrated in figure 10.

If we wish to collapse our chain to the plane, we can do the following. Consider the subchain as a directed chain. Because of its orthogonality, when the chain is planar each edge e_i will point either directly left or right, and all others up or

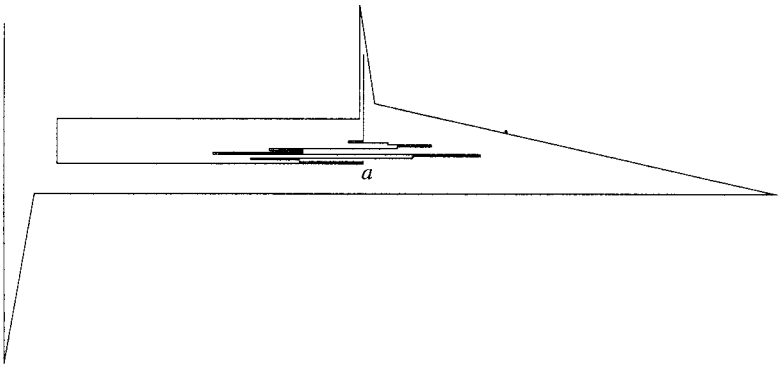

Figure 11. A chain whose key fits in the door.

Figure 12. Left: a monotone chain. Right: a non-monotone chain.

down. Thus the x-coordinate of the final edge, the key, is the x-coordinate of a plus the sum $\sum(s_i\colon\ e_i$ points right$) - \sum(s_i\colon\ e_i$ points left$)$. Since the door has width less than 1, and the elements s_i are integers, the key fits in the door if and only if $\sum(s_i\colon\ e_i$ points right$) = \sum(s_i\colon\ e_i$ points left$)$, solving the Partition problem. This situation can be examined in figure 11.

If we let all the small edges of length $1/n$ lie in a northerly (increasing in y-coordinate) direction, as in figure 11, then the subchain cannot self-intersect. The only possible self-crossing in the entire chain occurs at the key, meaning that there exists a planar conformation if and only if a partition exists. This completes the reduction. □

(Due to its relevance to the reconfiguration of linkages, it is worth pointing out that this proof is similar to the NP-hardness proof of Ruler Folding [7].)

3.1.1. Non-crossing monotone conformations

If the Planar Flattening problem is NP-hard, it might be easier to decide more restrictive versions of the problem. We may also wish to consider whether or not a chain has a *monotone* conformation in the plane. A planar polygonal chain is monotone if every vertical line intersects the chain in at most one point or one segment. In other words, as one traverses the chain edge by edge, the chain always progresses to the right. Examples of a monotone chain and a non-monotone chain are in figure 12.

Problem 5 (Planar Monotone Reconfiguration). Given a polygonal chain in three dimensions, does there exist a sequence of vertex-angle preserving motions which place

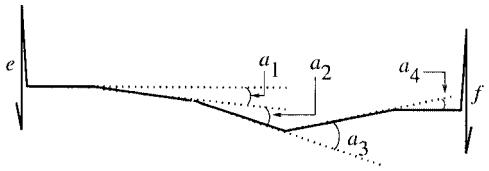

Figure 13. A chain which has a monotone conformation. Here $a_3 = a_1 + a_2 + a_4$, solving the Partition problem.

the chain into a non-crossing planar monotone conformation?

Despite its apparent simplicity, this problem is also NP-hard, which we demonstrate in a similar fashion to the reduction above.

Theorem 5. Planar Monotone Flattening is NP-hard.

Proof. Suppose we have a Partition problem on some set S. We create a chain similar to the one in figure 13, such that the angles a_i are proportional to the elements s_i, such that $a_i = s_i/|S|$ where $|S|$ is the sum of all elements. Note that the chain can only be placed in a monotone conformation if the edges e and f are extremely close to parallel, which we can specify to be less than $1/|S|$. If e and f are almost parallel in this fashion, the sum of the vertex-angles turning to the left minus the vertex-angles turning to the right is less than $1/|S|$, which solves the Partition problem. Furthermore, if e and f are near parallel, then the chain is guaranteed to be monotone. Since the sum of the a_i terms is at most 1 rad, the chain cannot progress to the left (and therefore violate monotonicity) except at the edges adjacent to e and f.

As an aside, it is worth noting that in the most widely used models of computation, such as the Real RAM model, trigonometric functions are not computable. Therefore, building a chain with specific angles is not possible in our model of computation since computing the coordinates of the vertices would involve trigonometry. We can easily circumvent this shortcoming by never referring to the actual angles, but by approximating the sine and cosine of the angle $1/|S|$ and using identities (such as $\cos(A + B) = \cos A \cos B - \sin A \sin B$) to derive a consistent approximation for the entire chain. □

3.1.2. Reconfiguring into a convex coil

We say that a *convex coil* is a non-crossing planar chain composed of only right or left turns. We would like to answer the following problem.

Problem 6 (Convex Coil Reconfiguration). Given a polygonal chain in three dimensions, does there exist a sequence of vertex-angle preserving motions which place the chain into a convex coil conformation?

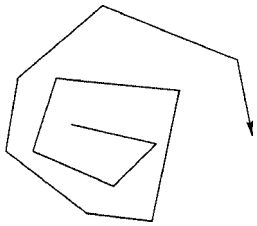

Figure 14. A chain which is spiralling outward.

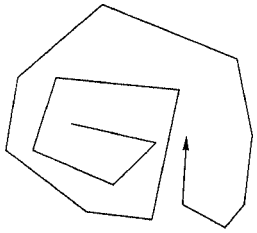

Figure 15. A chain which is spiralling inward.

Unfortunately, the above problem is not yet solved, but the question of whether such a conformation exists for a given chain can be answered fairly easily. Because all vertex-angles are fixed and turn either all right or all left, we have no freedom in choosing where to place the edges. All we can do is draw the chain in the plane with all right or all left turns and check for intersections.

We can use the algorithm of Balaban [24], which can report whether of not a collection of n line segments contains an intersection in $O(n \log n)$ time and $O(n)$ space. This is not necessary, however, as the structure of a convex coil gives us a great deal of extra information. We now show how to determine whether or not a chain with fixed vertex-angles has a planar convex coil conformation in $O(n)$ time.

We draw the chain in the plane starting at one endpoint and drawing one edge successively after the other, always checking if the last edge drawn intersects any of the edges which precede it (which have already been drawn). We identify two distinct stages. Initially, each new edge will lie outside the convex hull of the drawn portion of the chain. In this case, there is no possibility for collision with the drawn portion of the chain. We say that the chain is *spiralling outward* in this case, as in figure 14. When a new edge is drawn inside the convex hull of the chain, collisions might occur. Furthermore, from this point onward, all edges will lie inside the convex hull, since a left turn is needed to escape. Here the chain is *spiralling inward*, as in figure 15.

Our algorithm works as follows. As long as the chain is spiralling outward, we continue, because no self-intersections are possible in this stage. Our only interest is detecting when the chain begins to spiral inward. This is easily accomplished by running Melkman's $O(n)$ incremental convex hull algorithm [25], insuring that each drawn edge expands the convex hull.

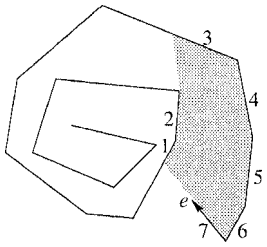

Figure 16. Determining intersections when the chain is spiralling inward. Visibility polygon is shaded.

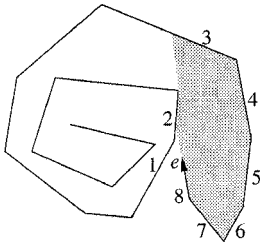

Figure 17. The chain with a new edge drawn since figure 16.

When we detect that the chain is spiralling inward, we begin checking for intersections. We start by computing the polygon consisting of all points visible from the endpoint e of the last edge in the convex direction. We call this polygon the *visibility polygon*, as illustrated in figure 16. This polygon is computable in $O(n)$ time by an algorithm of Avis and El-Gindy [26], but is more easily computed with the help of the following observation.

Because the chain consists of all right turns, the visibility polygon consists of a chain of reflex vertices (of edges 1 and 2 in figure 16) in the middle of the chain, followed by a chain of convex vertices (of edges 3–7) which includes e. Therefore, once we find one reflex vertex visible from e (if one exists), we know that the reflex chain of the visibility polygon must lie around this vertex and that the convex chain of the visibility polygon is adjacent to e. We just advance along these chains until we can no longer see e to obtain the polygon.

The visibility polygon contains all the edges that any new edges could possibly intersect, since getting to any edge in the chain but not in the polygon would require a left turn. The remainder of the algorithm is quite simple. We keep the edges of the polygon in a queue, such as in figure 16. For each new edge added, our visibility polygon changes. The new edge is added to the end of the queue (edge 8 in figure 17), and we advance the top of the queue to find the new first edge of the polygon. If the first edge of the visibility polygon has been intersected, we stop and report failure. If the first edge (edge 1) is neither intersected nor visible in a convex direction, we pop it from the queue and examine the next (edge 2). We repeat this procedure until we find an edge which is visible or intersected. In figure 17, the new polygon consists of edges 2–8.

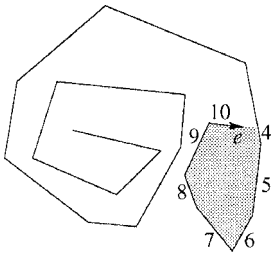

Figure 18. The chain with two new edges drawn since figure 17.

Figure 18 illustrates the same example a few steps later. Eventually, we will either find an intersection or draw the entire chain, answering the question.

4. Conclusion

We described an $O(n^2)$-time algorithm to solve the Edge Spin problem and a lower bound of $\Omega(n \log n)$ on the time complexity of the problem. We also provide near-optimal algorithms for the problem when the angle of the edge spin is 2π. Clearly our discussion suggests the following open problem.

Open problem 1. Tighten the complexity bounds on the Edge Spin problem.

There are several other relevant questions which we have not yet mentioned. We list a few below.

Open problem 2 (Preprocessing). Computing the feasibility of edge spins has a lower bound on the time complexity $\Omega(n \log n)$. But can a chain be preprocessed so that repeated queries of edge spin feasibility can be answered quickly?

Open problem 3 (Reconfiguration). Flattening a chain into the plane is NP-hard. Is it easier to place a chain into other canonical structures, such as on the convex hull of a polyhedron?

Several beautiful geometric problems in this area arise when more than one chain is considered. We give one such example.

Open problem 4 (Bonding). Given one flexible molecule A and one rigid molecule B with one or more bonding sites, can A be reconfigured to bond with B? Can it be done while respecting certain constraints?

Acknowledgements

The authors would like to thank Jeff Erickson for bringing to their attention the results of [20] and [22], and Ferran Hurtado for helpful discussions and for pointing out an error in an earlier manuscript.

References

[1] J. O'Rourke, Folding and unfolding in computational geometry, in: *Proc. Japan Conf. Discrete Comput. Geom. '98* (December 1998) pp. 142–147 (revised version submitted to LNCS).
[2] R. Connelly, E. Demaine and G. Rote, Every polygon can be untangled, in: *Proc. 16th European Workshop on Computational Geometry* (March 2000).
[3] T. Biedl, E. Demaine, M. Demaine, S. Lazard, A. Lubiw, J. O'Rourke, M. Overmars, S. Robbins, I. Streinu, G.T. Toussaint and S. Whitesides, Locked and unlocked polygonal chains in 3D, in: *Proc. 10th ACM-SIAM Symposium on Discrete Algorithms* (1999) pp. 866–867.
[4] J. Cantarella and H. Johnston, Nontrivial embeddings of polygonal intervals and unknots in 3-space, Journal of Knot Theory and Its Ramifications 7(8) (1998) 1027–1039.
[5] G.T. Toussaint, A new class of stuck unknots in Pol_6, Technical report SOCS-99.1, School of Computer Science, McGill University (April 1999).
[6] R. Cocan and J. O'Rourke, Polygonal chains cannot lock in 4D, in: *Proc. 11th Canadian Conference on Computational Geometry* (1999).
[7] J.E. Hopcroft, D.A. Joseph and S.H. Whitesides, On the movement of robot arms in 2-dimensional bounded regions, SIAM J. Comput. 14 (1985) 315–333.
[8] E.M. Arkin, S. Fekete, J.S.B. Mitchell and S.S. Skiena, On the manufacturability of paperclips and sheet metal structures, Technical report, Department of Applied Mathematics and Statistics, State University of New York at Stony Brook (1999).
[9] F.M. McMillan, *The Chain Straighteners* (MacMillan, 1979).
[10] P.W. Finn, D. Halperin, L.E. Kavraki, J.-C. Latombe, R. Motwani, Ch. Shelton and S. Venkatasubramanian, Geometric manipulation of flexible ligands, in: *Applied Computational Geometry* (Springer-Verlag, 1996) pp. 67–78.
[11] M.D. Frank-Kamenetskii, *Unravelling DNA* (Addison-Wesley, 1997).
[12] S.D. Stellman and P.J. Gans, Efficient computer simulation of polymer conformation. I. Geometric properties of the hard-sphere model, Macromolecules 5 (1972) 516–526.
[13] R. Unger and J. Moult, Finding the lowest free energy conformation of a protein is an NP-hard problem: proof and implications, Bull. Math. Biol. 55(6) (1993) 1183–1198.
[14] N. Madras and G. Slade, *The Self-avoiding Walk* (Birkhäuser, Boston, MA, 1993).
[15] A.S. Fraenkel, Complexity of protein folding, Bulletin of Mathematical Biology 55(6) (1993) 1199–1210.
[16] R. Randell, A molecular conformation space, in: *Studies in Physical and Theoretical Chemistry* (Elsevier, Amsterdam, 1988) pp. 125–140.
[17] R. Randell, Conformation spaces of molecular rings, in: *Studies in Physical and Theoretical Chemistry* (Elsevier, Amsterdam, 1988) pp. 141–156.
[18] H. Sachse, Über die Konfigurationen der Polymethylenringe, Z. Phys. Chem. 10 (1892) 202–241.
[19] J.L. Bentley and T.A. Ottmann, Algorithms for reporting and counting geometric intersections, IEEE Trans. Comput. C-28(9) (September 1979) 643–647.
[20] P.K. Agarwal and M. Sharir, Red–blue intersection detection algorithms, with applications to motion planning and collision detection, SIAM J. Comput. 19(2) (1990) 297–321.
[21] L.J. Guibas, M. Sharir and S. Sifrony, On the general motion planning problem with two degrees of freedom, Discrete Comput. Geom. 4 (1989) 491–521.

[22] B. Chazelle, H. Edelsbrunner, L.J. Guibas, M. Sharir and J. Snoeyink, Computing a face in an arrangement of line segments and related problems, SIAM J. Comput. 22 (1993) 1286–1302.
[23] M. Ben-Or, Lower bounds for algebraic computation trees, in: *Proc. 15th Annu. ACM Sympos. Theory Comput.* (1983) pp. 80–86.
[24] I.J. Balaban, An optimal algorithm for finding segment intersections, in: *Proc. 11th Annu. ACM Sympos. Comput. Geom.* (1995) pp. 211–219.
[25] A. Melkman, On-line construction of the convex hull of a simple polyline, Inform. Process. Lett. 25 (1987) 11–12.
[26] H. ElGindy and D. Avis, A linear algorithm for computing the visibility polygon from a point, J. Algorithms 2 (1981) 186–197.