CALIFORNIA STATE UNIVERSITY, NORTHRIDGE

PROTEIN FOLDING: PLANAR CONFIGURATION SPACES OF DISC ARRANGEMENTS AND HINGED POLYGONS

A thesis submitted in partial fulfillment of the requirements for the degree of
Master of Science in Applied Mathematics

by

Clinton Bowen

August 2014

The thesis of Clinton Bowen is approved:

Dr. Silvia Fernandez	Date
Dr. John Dye	Date
Dr. Csaba Tóth, Chair	Date

California State University, Northridge

ABSTRACT

PROTEIN FOLDING: PLANAR CONFIGURATION SPACES OF DISC ARRANGEMENTS AND HINGED POLYGONS

By

Clinton Bowen

Master of Science in Applied Mathematics

Chapter 1

Background

We consider four decision problems surrounding graph theory and geometry. The graph theory based problems involve polygonal linkages and the geometry based problems involve something called a contact graph of disks. In each problem, we decide whether a polygonal linkage or contact graph has a certain realization in the plane.

This thesis first presents preliminary information needed to pose our four problems, then we formally pose each problem and then provide the hardness results in all four cases. We show that all four problems are intractable, or NP hard (see definition below).

1.1 Graphs

A *graph* is an ordered pair $G = (V, E)$ comprising of a set of vertices V and a set of edges E. An edge is a two element subset of V. Note that with this definition of an edge it is not possible to have one element subset of V as an edge (sometimes referred to as a self-adjacent edge or loop). The *degree* of a vertex v is the number of edges that v is an element of. Vertices are said to be *adjacent* if they form an edge in E. *Neighbors* of a vertex v are the vertices adjecent to v. Edges are said to be adjacent if they share a vertex.

A *simple graph* has no self-adjacent vertices. In this thesis every graph is a simple graph. Given a graph $G = (V, E)$, a set of vertices $S \subset V$ is *independent* if no two vertices in S are joined by an edge. A *vertex cover* of a graph $G = (V, E)$ is a set of vertices $S \subset V$ if every edge $e \in E$, has at least one end corresponding in S. If $G' = (V', E')$ is a graph such that $V' \subset V$ and $E' \subset E$, then G' is a *subgraph* of G.

To formally show when two graphs are the same, we use the concept of graph isomorphism. Two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are *isomorphic* if there exists a bijective function $f : V_1 \mapsto V_2$ such that for any two vertices $u, v \in V_1$, we have $\{u, v\} \in E_1$ if and only if $(f(u), f(v)) \in E_2$. See an example in Table **??** and Figure 1.1.

Graph	Vertices	Edges
G_1	$\{a,b,c,d,e\}$	$\{(a,b),(b,c),(c,d),(d,e),(e,a)\}$
G_2	$\{1,2,3,4,5\}$	$\{(1,2),(2,3),(3,4),(4,5),(5,1)\}$

Table 1.1: Two graphs that are isomorphic with the alphabetical isomorphism $f(a) = 1$, $f(b) = 2$, $f(c) = 3$, $f(d) = 4$, $f(e) = 5$.

Figure 1.1: This figure depicts the graph isomorphism shown in Table **??** between V_1 and V_2.

To visualize a graph, G, we create a drawing Γ, of G. The *drawing* of a graph $G = (V, E)$ is an injective mapping $\Pi : V \mapsto \mathbb{R}^2$ which maps vertices to distinct points in the plane and for each edge $\{u, v\} \in E$, a continuous, injective mapping $c_{u,v} : [0, 1] \mapsto \mathbb{R}^2$ such that $c_{u,v}(0) = \Pi(u)$, $c_{u,v}(1) = \Pi(v)$, and the curve $c_{u,v}$ does not pass through any other vertex in V. In this thesis, we will work with straight line drawings and orthogonal drawings. Straight line drawings have mappings $c_{u,v}$ that are straight line segments. Orthogonal drawings have mappings $c_{u,v}$ which are a sequence of alternating horizontal line segments and vertical line segments. For orthogonal drawings, the endpoint of one line segment is the starting point of the next line segment, i.e. every $c_{u,v}$ is piecewise continuous. The endpoints of the line segments of $c_{u,v}$ that are not $\Pi(u)$ or $\Pi(v)$ are called *bends*.

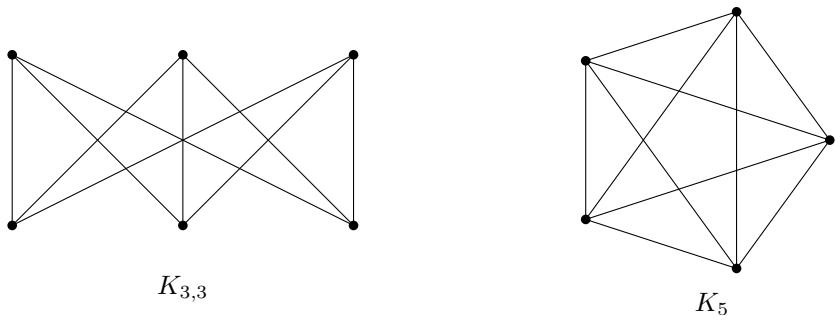


Figure 1.2: The K_5 and $K_{3,3}$ drawn in the plane.

Kuratowski's theorem characterizes finite planar graphs. A finite graph is planar if and only if it does not contain a subgraph that is a subdivision of K_5 or $K_{3,3}$ [14]. Figure 1.2 shows a drawing of K_5 and $K_{3,3}$. Two edges in a drawing *cross* if they have a common interior point. The *crossing number* of a graph is the smallest number of edge crossings for a graph over all drawings. A drawing is said to be *planar* if no two distinct edges cross [3]. A planar drawing is also called an *embedding*. Two embeddings of a graph G are *equivalent* if for every vertex the counter-clockwise order of neighbors are the same.A combinatorial *embedding* is a planar drawing with a corresponding counter-clockwise order of the neighbors of each vertex. An orientation

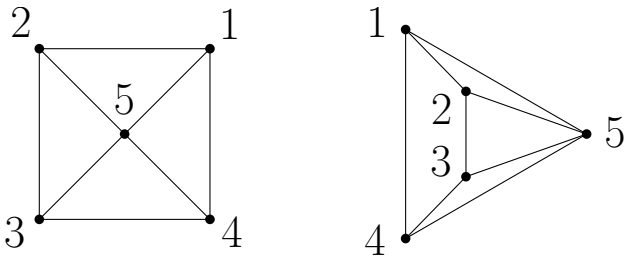


Figure 1.3: Here is a wheel graph, W_5, in two separate drawings with the same counterclockwise ordering of neighbors for each vertex.

preserving rigid transformation (i.e., rotation and translation) map an embedding to an equivalent embedding. Reflections reverse the counter-clockwise order around each vertex.

Figure 1.3 depicts two different drawings of the wheel graph W_5.The drawings have the followings counterclockwise order of neighbors for each vertex: Referencing table **??** and Figure 1.3, we realize that the two drawings of W_5 are equivalent.

Vertex	Left & Middle Drawing	Right Drawing
1	$(2,5,4)$	$(4,5,2)$
2	$(3,5,1)$	$(1,5,3)$
3	$(2,4,5)$	$(5,4,2)$
4	$(1,5,3)$	$(3,5,1)$
5	$(2,3,4,1)$	$(4,3,2,1)$

Table 1.2: A table showing the counter-clockwise circular ordering of neighbors for the left and right drawing in Figure 1.3. Note that the permutation cycles are equivalent for the right and left drawings.

1.1.1 Trees

A *path* is a sequence of vertices in which every two consecutive vertices are connected by an edge.

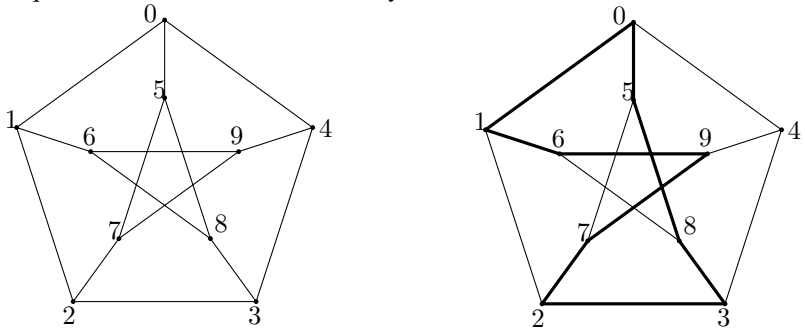


Figure 1.4: An embedding of the Peterson graph with a simple cycle of (2,7,9,6,1,0,5,8,3).

A *simple cycle* of a graph is a sequence, $(v_1, v_2, \ldots, v_{t-1}, v_t)$, of distinct vertices such that every two consecutive vertices are connected by an edge, and the last vertex, v_t, connects to v_1 (see Figure 1.4). A graph is *connected* if for any two vertices, there exists a path between the two points. A *tree* is a graph that has no simple cycles and is connected (see Figure 1.5). Every tree is planar. A *forest* is a disjoint union of trees.

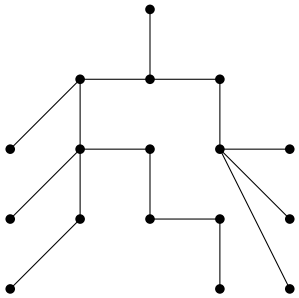


Figure 1.5: An example of a tree.

An *ordered tree* is a tree T together with a cyclic order of the neighbors for each vertex (see Figure 1.6).

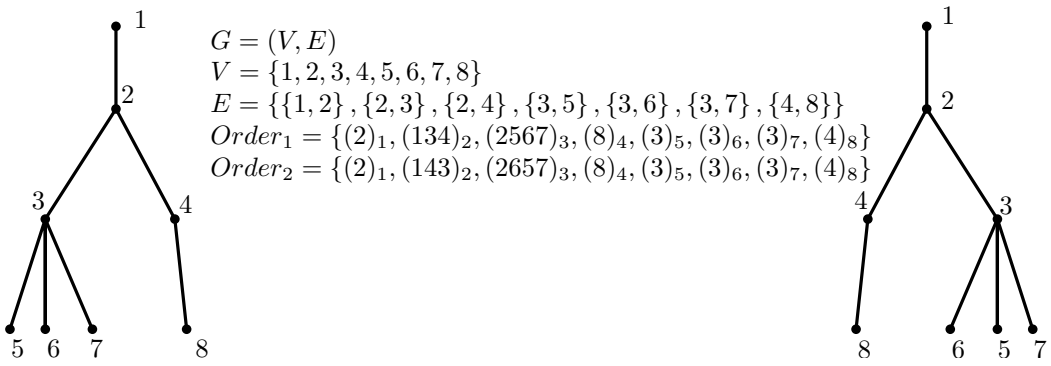


Figure 1.6: A tree with two embeddings with different cyclic orderings around vertices.

Embeddings of ordered trees are combinatorially equivalent if for each node the counter-clockwise ordering of adjacent nodes are the same.

1.2 Linkages

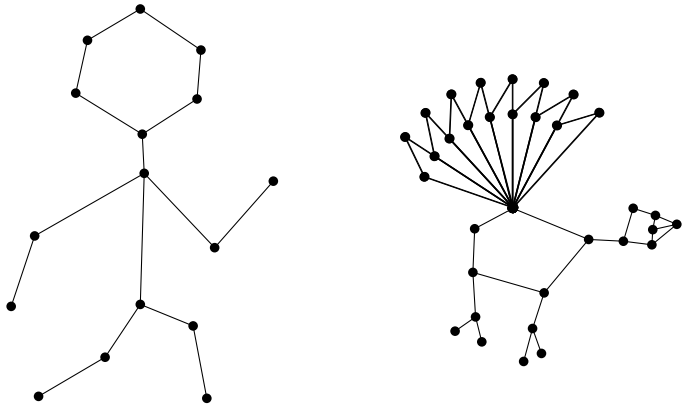


Figure 1.7: Here are skeleton drawings of a human and a turkey. When animating skeletons, one tends to make sure that the lengths of the skeleton segments are kept the same length throught the animation. Otherwise, the animation may depart from what is ideally understood of skeletal motions.

When graph drawings model physical objects, other qualities about the graph can be contextualized in a geometric sense. Distance, angular relationships and other geometric qualities may be relevant. In any drawing, edges have length, angles formed by adjacent edges, and so on. In this thesis we are interested in the inverse problem where we would like to embed a graph with specific geometric properties, for example, an embedding with specified edge lengths. This motivates the following definition. A *length assignment* of a graph $G = (V, E)$ is a function $\ell : E \mapsto \mathbb{R}^+$. If $\ell(e)$ is the length of an edge e, $\ell(e)$ must be strictly positive in a drawing, otherwise it may result in two distinct vertices with the same coordinates. Similar to combinatorial embeddings which is an equivalence class of embeddings of the same counter-clockwise order of vertices, we can also define an equivalence class of drawings with the same length assignment. A *linkage* is a graph $G = (V, E)$ with a length assignment $\ell : E \mapsto \mathbb{R}^+$ (e.g., see Figure **??**).

1.3 Polygonal Linkages

A generalization of linkages is a polygonal linkage where edges of given lengths are replaced with rigid polygons. Formally, a *polygonal linkage* is an ordered pair $(\mathscr{P}, \mathscr{H})$ where $\mathscr{P}$ is a finite set of polygons and $\mathscr{H}$ is a finite set of hinges; a *hinge* $h \in \mathscr{H}$ corresponds to two or more points on the boundary of distinct polygons in $\mathscr{P}$. A *realization of a polygonal linkage* is an interior-disjoint placement of congruent copies of the polygons in $\mathscr{P}$ such that the copies of a hinge are mapped to the same point (e.g., Figure

1.8). A *realization of a polygonal linkage with fixed orientation* is a realization in which each polygon is

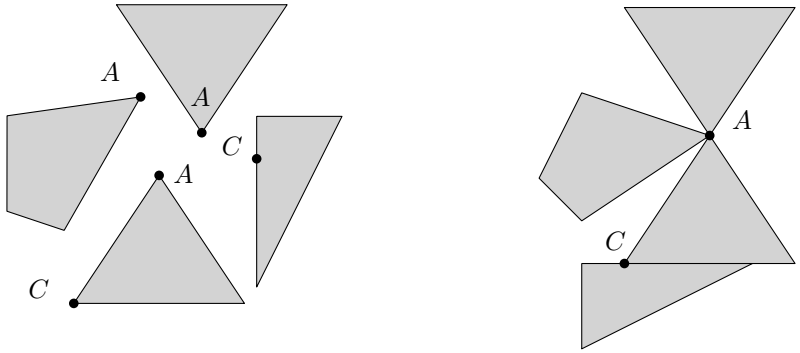


Figure 1.8: (a) A polygonal linkage with a non-convex polygon and two hinge points corresponding to three polygons. Note that hinge points correspond to two distinct polygons.(b) Illustrating that two hinge points can correspond to the same boundary point of a polygon.

translated and rotated copy of a polygon in $\mathscr{P}$; at each hinge the incident polygons are in a given counter-clockwise order (refer to Figure **??**). Note that oriented polygonal linkage realizations do not allow for

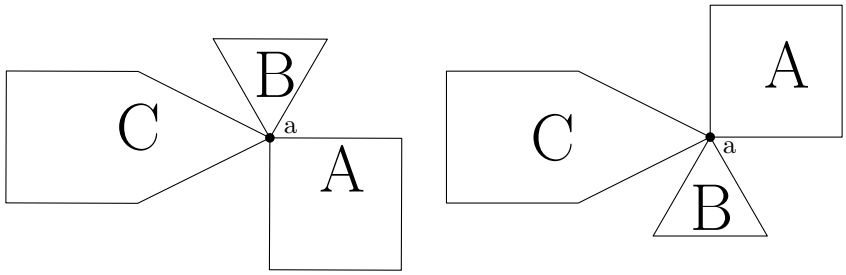


Figure 1.9: Two realizations of the same polygonal linkage with that differ in the counter-clockwise order of polygons around vertex *a*.

reflection transformations of polygons in $\mathscr{P}$.

These two realization types allow one to pose two different problems, the realizability problem for polygonal linkages and the realizability problem for polygonal linkages with fixed orientation:

Problem 1 (Realizibility Problem for Polygonal Linkages). Given a polygonal linkage, does it have a realization?

Problem 2 (Realizibility Problem for Polygonal Linkages with Fixed Orientation). Given a polygonal linkage with fixed orientation, does it have a realization?

Not every polygonal linkage has a realization. Consider the 7 congruent copies of an equilateral triange with a common hinge point in Figure 1.10. To show it does not have a realization, suppose it is realizable. Each angle of every triangle is $\frac{\pi}{3}$ radians. The sum of 7 angles formed by the triangles is $\frac{7\pi}{3} > 2\pi$. The total radian measure around A is 2π. The contradiction is that the sum of 7 angles formed by the triangles in an interior disjoint placement is $\frac{7\pi}{3}$. The polygonal linkage of Figure 1.10 would overlap itself and does not have a realization.

There are polygonal linkages that admit realizations but every realization requires rotation. Figure **??** show the congruent copies of the polygons A, B, C, and D in two different configurations, the far right is a realization, the middle fails to be a realization because of the interiors of B and D intersecting and the left

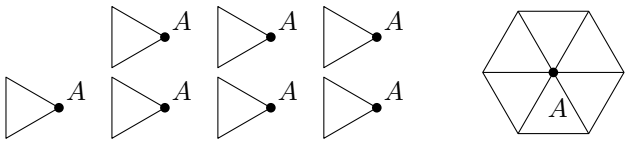


Figure 1.10: Here we have 7 congruent copies of an equilateral triangle with a hinge point of A. The polygonal linkage is not realizable. The best we can realize is at most 6 congruent copies of an equilateral triangle with the hinge point of A in the plane.

showing the polygons in $\mathscr{P}$. In fact, this polygonal linkage cannot admit a realization with fixed orientation. Indeed this polygonal linkage cannot satisfy Problem 2, suppose there is a realization with fixed orientation. Without loss of generality, fix the placement of C. A, B, and D have unique placement around triangle C. In this placement C and D overlap. Figure **??**, satisfies Problem 1 but not Problem 2. The far right is a realization

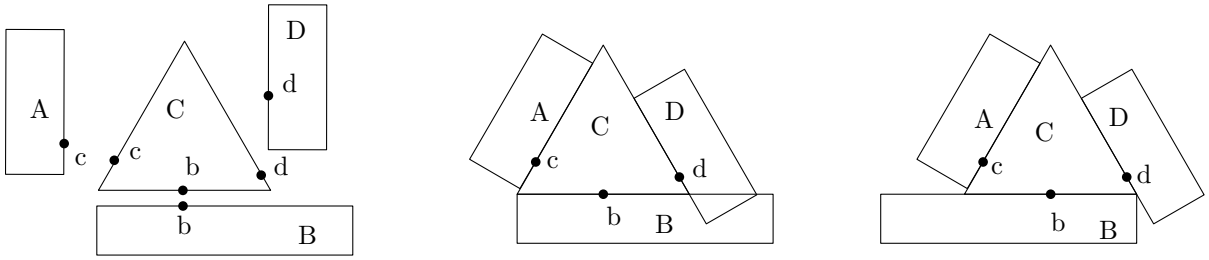


Figure 1.11: This example shows yet another example where two realizations of the same polygonal linkage. One realization where there is an intersection and another where there isn't an intersection.

but with polygon B reflected.

Figure **??** does not quite get at the heart of the challenge with Problem 1 because the counter-clockwise order of the polygons around hinges is not considered.

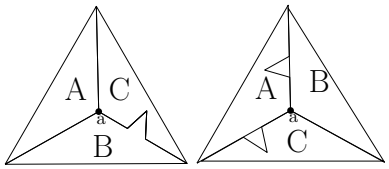


Figure 1.12: Here we have two realizations of a polygonal linkage with two different counter-clockwise order (C,B,A) and (B,C,A) respectively. Note that the placement with ordering (B,C,A) has an overlap.

Figure 1.12 shows three polygons with a common hinge. In the counter-clockwise order (A,B,C), the polygonal linkage admits a realization whereas in the counter-clockwise order (A,C,B), it does not admit a realization. The examples above show that answers to Problem 1 and 2 could be yes or no; the answer could be negative for various reasons. Sections 2 and 3 of this thesis address the computational complexity of solving Problems 1 and 2. We show that both problems are intractable.

1.3.1 Geometric Dissections

Hilbert's third problem asks: given any two polyhedra of equal volume, is it always possible to cut the first into finitely many polyhedral pieces which can be reassembled to yield the second [2]? In three dimensions the answer is no however for two dimensions it is true [23].

The Wallace-Bolyai-Gerwien Theorem simply states that two polygons are congruent by dissection iff they have the same area. A *dissection* being a collection of smaller polygons whose interior disjoint union forms a polygon. Hinged dissections of a polygon P is a polygonal linkage that admits a realization that forms P. Demaine et. al. [1] showed that any two polygons of the same area have a common hinged dissection where polygonal pieces must hinge together at vertices to form a connected realization and that there exists a continuous motion between the two realizations (refer to section 1.5.3). The was an outstanding problem for many years until 2007.

The Haberdasher Puzzle was proposed in 1902 by Henry Dudeney: can a square and an equilateral triangle of the same area have a common dissection into four pieces?

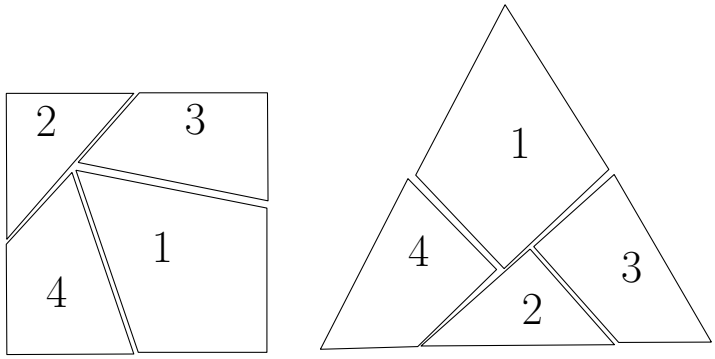


Figure 1.13: The Haberdasher Puzzle was proposed in 1902 and solved in 1903 by Henry Dudeny. The dissection is for polygons that forms a square and equilateral traingle.

Geometric dissections are closely related to polygonal linkages. Figure 1.14 shows two arrangements of the same polygons to form a hexagon and a square. The polygons are not hinged and are arranged in differing order. The polygons are merely tiled together to form the hexagon and square.

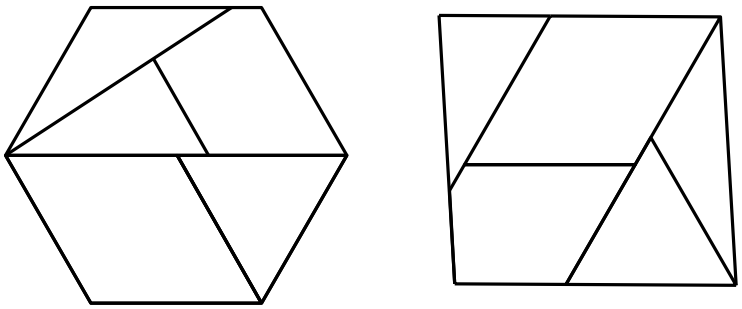


Figure 1.14: Two configurations of polygonal linkage where the polygons touch on boundary segments instead of hinges. These two realizations of the polygonal linkage are invalid to our definitions.

Figure 1.15, shows the Haberdasher problem with hinges. This makes the Haberdasher problem as a type of polygonal linkage where the polygons are free to move about their hinge points and take the form of a triangle or square.

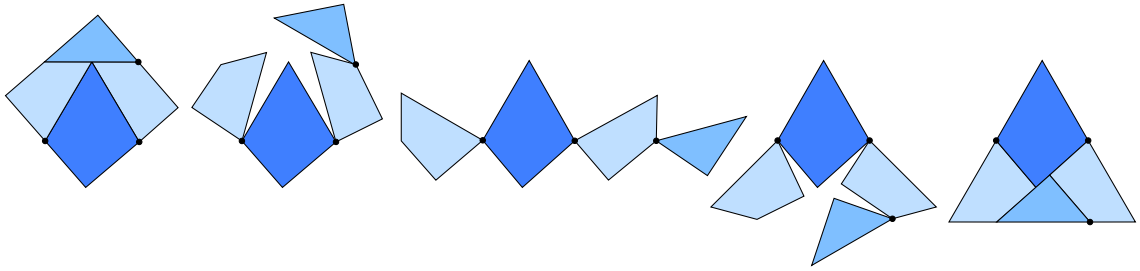


Figure 1.15: This shows the Haberdasher problem in the form of polygonal linkage [1]. This is a classic example of two polygons of equal area that have a common hinged dissection.

1.4 Disk Arrangements

A *disk arrangement* is a set of interior disjoint disks, D. If for any pair of disks in D intersect at a boundary point, they are said to be in contact (kissing).

Figure 1.16: This example represents a disk arrangement and its contact graph.

A *contact graph* $G = (V,E)$ corresponding to a given disk arrangement where there is a bijection $b_V : V \mapsto D$ and a bijection that maps an edge $e_{i,j} \in E$ to an interior disjoint pair of disks $d_i, d_j \in D$ (see Figure 1.16). Given a disk arrangement, the contact graph can be thought of as a linkage because the distance between two kissing disk equal the sum of radii. However if the two disks don't kiss, the distance between their centers is strictly greater than the sum of their radii. Given a disk arrangement, the contact graph can be thought of as a linkage because the distance between two kissing disk equal the sum of radii. However if the two disks don't kiss, the distance between their centers is strictly greater than the sum of their radii.

Koebe's theorem states that for every planar graph G, there exists a planar disk arrangement whose contact graph is G [13]. This motivates the question of whether a planar graph G is a contact graph of a disk arrangement with given radii. The radii can be given by a weight function. Let $\omega : V \mapsto \mathbb{R}^+$ be the *weight function*. ω assigns a weight to each vertex in V. Let $\Pi : V \mapsto \mathbb{R}^2$ be that planar mapping of vertices.

For planar graphs with positive weighted vertices, we pose two realizability problems:

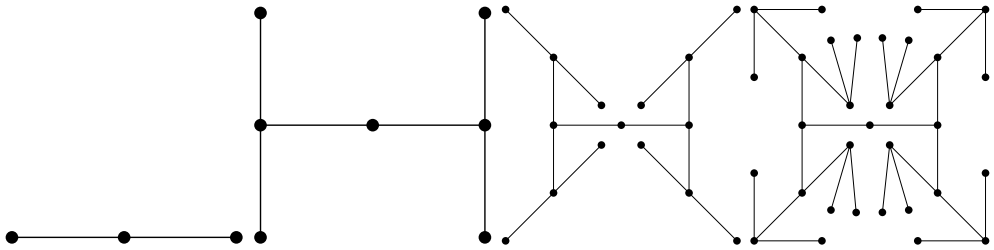
Problem 3 (Unordered Realizibility Problem for a Contact Graph). Given a planar graph with positive weighted vertices, is it a contact graph of some disk arrangement where the radii equal the vertex weights?

Problem 4 (Ordered Realizibility Problem for a Contact Graph). Given a planar graph with positive weighted vertices and a combinatorial embedding, is it a contact graph of some disk arrangement where the radii equal the vertex weights and the counter-clockwise order of neighbors of each disk is specified by the combinatorial embedding?

An instance of Problem 3 is shown in Figure 1.16 where the cycle graph C_5 is the contact graph of unit disks. It is not difficult to see that there exists a planar graph with positive weights with no realizable disk arrangement. Consider the a star graph with 6 leafs, each vertex with unit weight. In any realization, the angle between two consecutive edges must be greater than $\frac{\pi}{3}$. The sum of 6 angles is 2π however, the sum

of 6 consecutive angles is greater than 2π. The contradiction shows that no realization is possible (refer to Figure **??**). Note that with the wheel graph W_7 is realizable as a contact graph of unit disks.

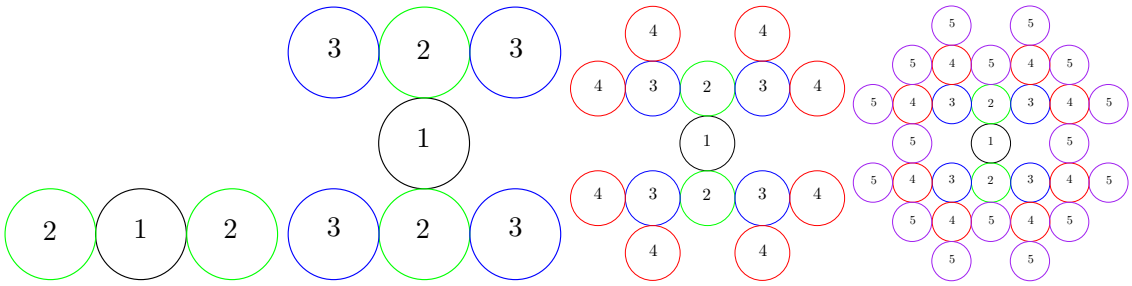
Every path with arbitrary positive radii is realizable as a contact graph, place the vertices on a line. We show that not all binary trees are realizable, even with unit disks.Consider the balanced binary trees of depth i $\{T_i\}_{i=1}^{\infty}$ with unit weights on the vertices (see Figure **??**). These trees are not realizable for sufficiently large i. Let i be a positive integer and suppose that T_i is a contact graph of unit disks. The balanced binary tree T_i



(a) Binary tree T_2. (b) Binary tree T_3. (c) Binary tree T_4. (d) Binary tree T_5.

Figure 1.17: We show the linkages T_2 through T_5 with distance 2 between adjacent vertices.

has 2^i-1 vertices. The total area of the disks is $\left(2^i-1\right)^2\cdot\pi$. We now derive an upper bound for this area. Suppose the disk corresponding to the root of the tree is centered at the origin. The centers of the disks at level j are at a distance at most $2\cdot(j-1)$ away from the origin. The centers of all disks are at distance at most $2\cdot(i-1)$ away from the origin. All unit disks are contained in a disk of radius $2i-1$ centered at the origin. The total area of the disks is at most $(2i-1)^2$. A upper bound of the total area of the disk arrangement is the area of the bounding box of the disks.



(a) A disk arrangement with two layers of disks (b) A disk arrangement with three layers of disks (c) A disk arrangement with four layers of disks (d) A disk arrangement with five layers of disks

Figure 1.18: For $i=2,3,4,5$ the tree T_i is a contact graph of unit disks.

Figure **??** shows the first four non-trivial trees as a contact graph of unit disks. Every disk at level up to i is contained in a disk of radius $2\cdot i-1$ centered at the origin. The total area of the disk arrangement is $(2\cdot i-1)^2\cdot\pi$. When $i\geq 8$ we have a contradiciton.

There are instances where a planar graph with weights admits a realization but the cyclic order of neighbors may not be the same as the combinatorial embedding. Define G as follows: start with a star centered at C and with 6 leafs, A_1 through A_6; attach two leafs, B_1 and B_2, to A_1 and A_2 respectively (see Figure **??**). Let the weight of C be $1+\varepsilon$ for sufficiently small $\varepsilon>0$. The neighbors of C have unit weight. The weights of the two leaves have weight $\frac{1}{\varepsilon}$/ The right of Figure **??**) shows a realization where A_1 and A_2 are in opposite position of the counter-clockwise order around C. If A_1 and A_2 are required to be consecutive in the counter-clockwise order around C, there is no realization.

Suppose there is a realization where $A_1,\ldots,A_6$ are in the counter-clockwise order around C (see Figure

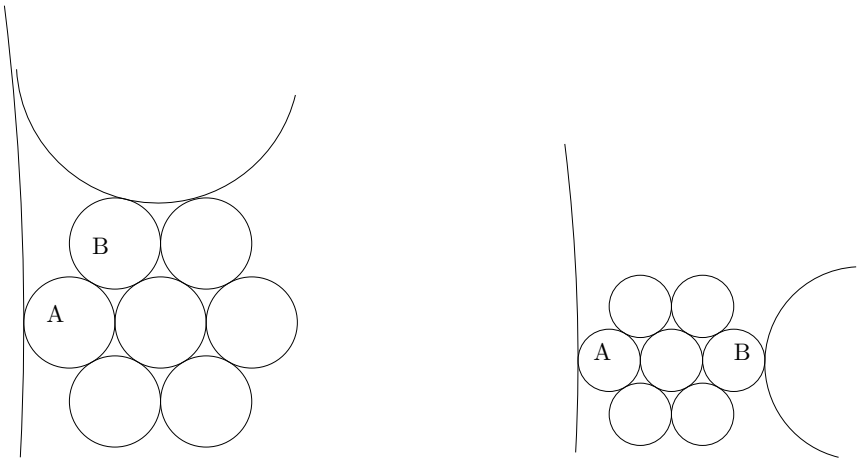


Figure 1.19: Consider these two ordered disk arrangements where A and B are in the concentric rings of disks. The large disks are in contact to A and B respectively. If A and B are adjacent, then there is a restriction of how large the size of the disks can be that are attached to them as seen in on the left. Whereas if A and B are not adjacent in this disk arrangment as shown on the right, the size of the kissings disks could be arbitrarily large.

??). If $\varepsilon > 0$ is sufficiently small, then the centers of of $A_1, \ldots, A_6$ are arbitrarily close to the vertices of a regular hexagon. Consider the common tangent lines between A_1 and B_1 and A_2 and B_2. The possible position of tangent line between A_1 and B_1 ranges from the common tangent line of A_1 and A_6 to the common tangent line of A_1 and A_2. Similarly, The possible position of tangent line between A_2 and B_2 ranges from the common tangent line of A_2 and A_3 to the common tangent line of A_1 and A_2. In any position, the common tangent lines between A_1 and B_1 and A_2 and B_2 intersect. If $\frac{1}{\varepsilon}$ is sufficiently large, then the disks D_1 and D_2 also intersect. This contradicts that there is a realization.

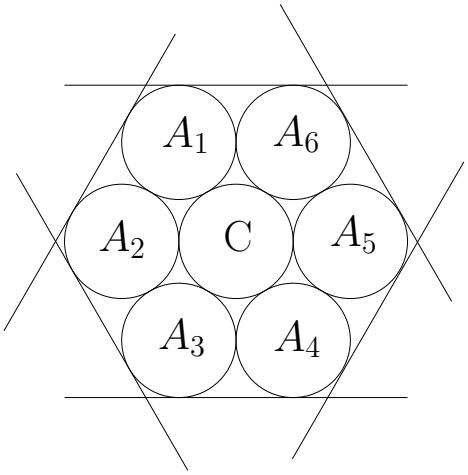


Figure 1.20: This example represents a disk arrangement and its contact graph.

Figure **??** shows how an ordered contact graph may not be realizable. On the left, it shows a limitation on the weights of the disks that are in contact with disks A and B. On the right, the figure shows the order where A and B are on opposing ends of the ring of disks and can allow of arbitrary size of wieghted disks in contact with A and B.

1.5 Configuration Spaces

Just as one can compose colors or forms, so one can compose motions.

Alexander Calder, 1933

Recall Figure 1.15 illustrating the hinged dissection that formed a square and triangle and several drawings of the hinged dissections that simulate the motion of moving the polygons around the hinge points to form each shape. The set of all drawings in that motion represents the *configuration space* for that polygonal linkage. In this section we will formally describe the configuration space for each object we've drawn thus far.

1.5.1 Configuration Spaces of Graph Drawings

Recall that for a graph drawing we have an injective mapping $\Pi : V \mapsto \mathbb{R}^2$ which maps vertices to distinct points in the plane. The mapping Π uniquely determines each edge. An edge $\{u,v\} \in E$, is mapped to a straight line segment, $c_{u,v} : [0,1] \mapsto \mathbb{R}^2$ such that $c_{u,v}(0) = \Pi(u)$ and $c_{u,v}(1) = \Pi(v)$, and does not pass through other vertices. Let $\mathscr{D}_G$ be the set of all drawings of the graph G. By labelling the vertices of G, e.g. $v_1, v_2, \ldots, v_k, \ldots, v_n$, we can create a mapping from $\mu : \mathscr{D}_G \mapsto \mathbb{R}^{2|V|}$ where the coordinates of $\Pi(v_k)$ are the $(2k)^{\text{th}}$ and $(2k+1)^{\text{st}}$ coordinates in $\mathbb{R}^{2|V|}$. $\mu(\Pi)$ is a configuration. The configuration space is the set of $\mu(\Pi)$ for all drawings Π.

1.5.2 Configuration Spaces of Linkages

Consider drawings of a graph that respects the length assignment. A *realization* of a linkage, (G,ℓ), is a drawing of a graph, Π, such that for every edge $\{u,v\} \in E$, $\ell(\{u,v\}) = |\Pi(u) - \Pi(v)| = |\Pi(v) - \Pi(u)|$. A *plane realization* is a plane drawing with the property, $\ell(\{u,v\}) = |\Pi(u) - \Pi(v)|$. First let's define the space of realizations for a corresponding linkage, i.e.:

$$P_{(G,\ell)} = \{\, \Pi \in \mathscr{D}_G \,|\forall \{u,v\} \in E,\, \ell(\{u,v\}) = |\Pi(u) - \Pi(v)| \,\}$$

With respect to $P_{(G,\ell)}$, we can establish a configuration space that allows one to study problems of motion. For each vertex of G, the drawing of the vertex lies in the plane, i.e. $\Pi(v) \in \mathbb{R}^2$. By enumerating each vertex of G, e.g. $v_1, v_2, \ldots, v_k, \ldots, v_n$, we can create a mapping from $\mu : P_{(G,\ell)} \mapsto \mathbb{R}^{2|V|}$ where the corresponding coordinates of $\Pi(v_k)$ are in the $(2k)^{\text{th}}$ and $(2k+1)^{st}$ coordinates in $\mathbb{R}^{2|V|}$. The configuration space is $\mu\left(P_{(G,\ell)}\right)$.

Using standard definitions from real analysis, we can begin to pose problems about linkages with respect to a corresponding configuration space. A continuous function $\gamma : [0,1] \mapsto \mu\left(P_{(G,\ell)}\right)$ is a path from a realization $\gamma(0)$ to another realization $\gamma(1)$. γ can be thought of as an animation of drawings that starts at $\gamma(0)$ and ends at $\gamma(1)$. Any two realizations in the same path-connected component can be animated from one to the other continuously. The Carpenter's Rule states that every realization of a path linkage can be continuously moved (without self-intersection) to any other realization [7, 21]. To ask if $\mu\left(P_{(G,\ell)}\right)$ is a connected space, is to ask if $\mu(P)$ is connected in $\mathbb{R}^{2|V|}$. In other words, the realization space of such a linkage is always path-connected.

1.5.3 Configuration Spaces of Polygonal Linkages

The placement of a polygon is described by an isometry of Euclidian plane. An isometry is a composition of a translation, a rotation, and a possible reflection. As such, the isometry can be descibed as three parameters: (a,b,θ) where (a,b) is a translation vector; if $\theta \geq 0$, it describes a counter-clockwise rotation by θ and if $\theta < 0$, it descibes a reflection in the x-axis followed by a counter-clockwise rotation θ. For m polygons there will be $3m$ parameters. Recall a realization of a polygonal linkage is an interior-disjoint placement of congruent copies of the polygons in $\mathscr{P}$ such that the copies of a hinge are mapped to the same point (e.g., Figure 1.8). First consider the set of all realizations for the polygonal linkage $(\mathscr{P},\mathscr{H})$ and call it P. $\mu : P \mapsto \mathbb{R}^{3m}$ where m is the number of polygons in $\mathscr{P}$ is the configuration space function and the

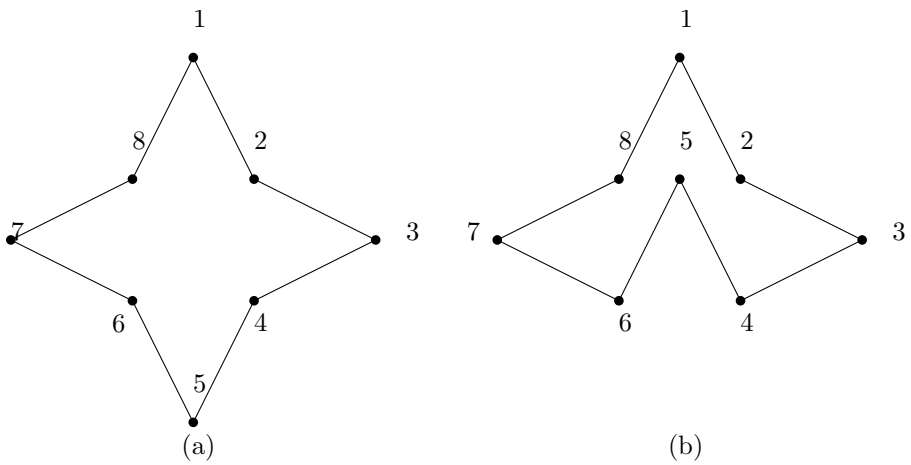


Figure 1.21: (a) and (b) show a linkage in two embeddings. Any realization of a path can be continuously moved without self-intersection to any other realizations.

configuration space is the set $\mu(P)$.

1.5.4 Configuration Spaces of Disk Arrangements

rewrite where theta is not needed. disks are symmetric and rotation does not help. The placement of a polygon is described by an isometry of Euclidian plane. An isometry is a composition of a translation, a rotation, and a possible reflection. As such, the isometry can be descibed as three parameters: (a,b,θ) where (a,b) is a translation vector; if $\theta \geq 0$, it describes a counter-clockwise rotation by θ and if $\theta < 0$, it descibes a reflection in the x-axis followed by a counter-clockwise rotation θ. For m polygons there will be $3m$ parameters. Recall a realization of a polygonal linkage is an interior-disjoint placement of congruent copies of the polygons in $\mathscr{P}$ such that the copies of a contact are mapped to the same point (e.g., Figure 1.8). First consider the set of all realizations for the polygonal linkage $(\mathscr{P},\mathscr{H})$ and call it P. $\mu : P \mapsto \mathbb{R}^{3m}$ where m is the number of polygons in $\mathscr{P}$ is the configuration space function and the configuration space is the set $\mu(P)$.

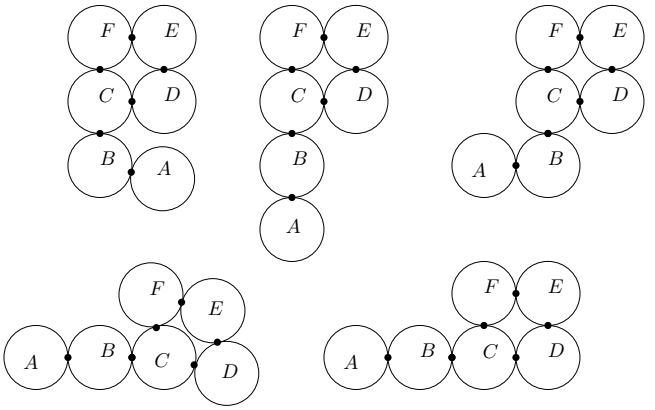


Figure 1.22: An example of a disk arrangement where A and B have a large range of freedom to move around. C, D, E, and F are limited in their range of motion to due to their contact points.

Consider the set of realizations P for a given disk arrangement $\mathscr{D} = \{D_i\}_{i=1}^{n}$. For any realization $R \in P$, there exists a corresponding contact graph, C. The configuration spaces of $\mathscr{D}$ are sets of $R \in P$ that are classified by the equivalent contact graphs, i.e. if R_1, $R_2 \in P$ and their corresponding contact graphs C_1 and

C_2 have a graph isomorphism, ϕ, then R_1 and R_2 belong to the same configuration space.

1.6 Algorithm Complexity

Algorithms are a list of instructions executed with a given input. The efficiency of an algorithm can be measured in terms of the amount of resources it uses such as time, memory, and power. Ideally, a desirable algorithm would primarily have a small run time and secondarily utilize a small amount of resources.

The time and space used by an algorithm is measured with units defined by a model of computation. The actual running time of an algorithm depend on a variety of factors for example: the processor, the hardware, the temperature, etc. Mathematical models of computation have been developed to measure running time of algorithms independent of the machine it runs on. One of the oldest and most popular models is the random access machine (RAM) model. RAM measures the unit of space in the number of words used where each word can store an arbitrary integer. In the real RAM, each word can store an arbitrary real number. The units of time is measured in the number of arithmetic operations and number of memory accesses (read or write).

The *running time* of an algorithm on a given input, is the time it takes to terminate.The *worst-case* running time is the largest running time over all inputs of a given size N. It is a function of N, it is usually monotonically increasing function since larger inputs tend to take more time to process. The key parameter of the efficiency of an algorithm is the growth rate of its worst case running time in terms of N. An algorithm is said to be *efficient* if the time needed to perform the list of instructions can be determined from a polynomial. Devising an efficient algorithm for a given problem is often a difficult task.

The growth rate of running times are typically compared upto constant vectors. Let f and g be defined on some subset of $\mathbb{R}$. $f(x) = O(g(x))$ if and only if there exists a constant M and x_0 such that

$$|g(x)| \leq M|f(x)|$$

for all $x \geq x_0$

1.6.1 Complexity Classes

Problems can be categorized by their running times. Each algorithm computes a function $f(I)$ on an input I, however many different algorithms can compute the same function. Algorithms are differentiated by their running times but the function is characterized by the fastest algorithm that can compute it. A problem can be formulated as follows, given input I find $f(I)$.

Problems can be categorized into complexity classes based on the fastest algorithms that solve them. The class of problems that can be solved in polynomial running time is called the *polynomial time* class, P.

A second property of problems is whether its solution can be verified efficiently. This property is independent of whether it can be solved efficiently. B is said to be an *efficient certifier* for a problem X if the following properties hold:

(i) B is a polynomial-time algorithm that takes two inputs s and t.

(ii) There exists a polynomial function p such that for every string s, we have $s \in X$ if and only if there exists a string t such that $|t| \leq p(|s|)$ and $B(s,t) =$ 'yes'.

The class of problems which have an efficient certifier is said to be the *nondeterministic polynomial time* class, NP. We continue with the definitions for NP-hard and NP-complete. A problem is NP-hard if every problem in NP can be reduced to it in polynomial time. A *polynomial time reduction* is when arbitrary instances of problem Y be solved using a polynomial number of standard computational steps, plus a polynomial number of calls to a black box that solves problem X, i.e. Y is reduced in polynomial time to X. A problem is NP-complete if it NP and NP-hard, i.e. NP-complete = NP $\cap$ NP-hard.

1.6.2 RSA Cryptosystem

Cryptography is the study of secure communication between parties in an untrusted or unsecure communication channel. Cryptography has three primary purposes for secure communications: provide confidentiality, authenticate entities, and verification of data. Modern cryptography is based on hard math problems such as integer factorization, discrete logarithmic problem, and pre-image problems. Hard math problems are not found in P and found in NP. In most forms of modern cryptography, *keys*, are data parameters used to form function outputs for the use in a communication channel. There are three common types of keys in cryptography:

1. A *secret key* is known by certain entities. Typically a secret key is used by entities to encrypt and decrypt data that is communicated over an untrusted channel. Secret keys require a secure channel to exchange between all entities.
2. When there are no means to exchange secret keys, one can use a private and public key scheme. A *public key* is a key that can be shared with any entity, i.e. trusted and untrusted entities can know it. The *private key* is a key that is kept to one entity. It is treated like a secret key and should only be known to that entity.

A *cryptosystem* is a suite of algorithms used to establish secure communication channel between parties. The RSA cryptosystem is the first practical cryptosystem in modern cryptography that allows for encryption of data (confidentiality), authentication of entites, and verify message integrity using just one underlying hard math problem, integer factorization.

RSA is named after its second inventors, Ron Rivest, Adi Shamir, and Leonard Adelmen. These three individuals devised and published the algorithm in 1977. The original inventor of RSA was Clifford Cocks in 1973 however, it was only known to the public since 1997 that Clifford Cocks was the original inventor because his was was classified by Government Communication Headquaters, an intelligence agency of the United Kingdom.

For the RSA cryptosystem, we first want to pose the communication security problem. Suppose we have two entities, Alice and Bob, that wish to communicate over an unsecure channel. Should Alice and Bob agree to using RSA, each entity will have a pair of keys, a private key and a public key. Most implementations of RSA use the following key derivation [18]:

1. Let $n = p \cdot q$ where p and q are randomly chosen prime numbers.
2. Choose an integer e such that $1 < e \leq \phi(n)$ and $gcd\,(e, \phi\,(n)) = 1$ where ϕ is the Euler totient function.
3. Let $d \equiv e^{-1} \mod \phi(n)$.

The RSA cryptosystem is based around the following formula:

$$(m^e)^d \mod n = \left(m^d\right)^e \mod n \equiv m \mod n$$

where we have natural numbers e, d, m, and n, such that $m < n$ and $gcd(m,n) = 1$.

If Alice derives a public and private key in the manner described, her public key is (n,e) and her private key is d. Suppose Bob wants to send Alice a message M. Bob will represent M in binary form, m. Alice sends her public key (n,e) to Bob and keeps her private key d secret. If $gcd(m,n) \neq 1$, then Bob modifies m by padding m with additional digits so that m becomes co-prime with n. There are efficient padding schemes to modify m such that m and n are co-prime [10]. Bob sends the following value, c, to Alice:

$$c \equiv m^e \mod n$$

c is said to be a ciphertext. Alice can decrypt the ciphertext and recover m by computing:

$$m \equiv c^d \mod n = (m^e)^d \mod n$$

The RSA cryptosystem is based on integer factorization and the RSA problem, given n where $n = p \cdot q$ where p and q are prime numbers, find p and q. For sufficiently large integers, factorization and the RSA problem becomes very difficult. If there were an algorithm that solved integer factorization in polynomial time, then one can also solve the RSA problem as well. Suppose a third party listens into the conversation and knows Alice's public key (n,e) and the ciphertext c. If the the third party can factor $n = p \cdot q$ then the attacker can computer $\phi(n)$ and compute $d \equiv e^{-1} \mod \phi(n)$ using the extended euclidian algorithm. Once they have d, the attacker can compute $m \equiv c^d \mod n$.

Currently, the most efficient factorization algorithm is the general number sieve algorithm [15]. It's an exponential running time algorithm. If a polynomial running time algorithm for integer factorization existed, it would allow for compromise of security that is provided to communicating parties by the RSA cryptosystem. It would allow for a reduction of the integer factorization problem, a problem that exists in NP but not P. The algorithm would allow for an adversary to attack RSA [17] in polynomial time.

1.7 Satisfiability

Let $x_1, \ldots, x_n$ be boolean variables. A boolean formula is a combination of conjunction, disjunctions, and negations of the boolean variables $x_1, \ldots, x_n$. A *clause* is a disjunction of distinct literals. A *literal* is a variable or a negated variable, x_i or $\bar{x}_i$, for $i = 1, \ldots, n$. A boolean formula is *satisfiable* if one can assign true or false value to each variable so that the formula is true. It is known that every boolean formula can be rewritten in *conjunctive normal form* (CNF), a conjunction of clauses, via DeMorgan's law and distributive law. Furthermore, it is also known that every boolean formula can be written in CNF such that each clause has exactly three literals. This form is called 3-CNF. For example, consider the clause $A \vee B \vee C \vee D \vee E$. This clause can be rewritten in 3-CNF form as $(A \vee B \vee x_1) \wedge (\neg x_1 \vee C \vee x_2) \wedge (\neg x_2 \vee D \vee E)$ where x_1 and x_2 are literals that allow us to form 3-CNF clauses. Here are the problem statements for satisfiability:

Problem 5 (Satisfiability Problem (SAT)). Given a boolean formula, is it satisfiable? [19]

Brute force is when an algorithm tries all possibilities to see if any formulates a satisfiable solution. It is clear that SAT is decidable in exponential time by testing all possibilities. This is callled a brute force solution. It is not known whether SAT admits a polynomial time solution, that is whether it is in P.

Problem 6 (3-SAT Problem). Given a boolean formula in 3-CNF, is it satisfiable?

The problems we focus on in this thesis have a geometry. A special geometric 3-SAT problem is that Planar 3-SAT Problem. Given a 3-CNF boolean formula, Φ, with n variables and m clauses. We define the *associated graph* $A(\Phi)$ as follows: the vertices correspond to the variables and clauses in Φ. We place an edge in the graph if variable x_i appears in clause C_j.

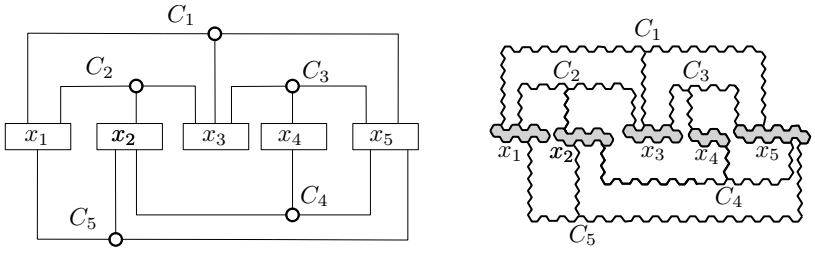


Figure 1.23: Left: the associated graph $A(\Phi)$ for a Boolean formula Φ. Right: the schematic layout of the variable, clause, and transmitter gadgets in the auxiliary construction showing in Section 3.1

Problem 7 (Planar 3-SAT). Given a boolean formula Φ in 3-CNF such that its associated graph is planar, decide whether it is satisfiable is a *3-SAT problem*.

Figure 1.23 is associated to a family of boolean formulas. One such associated boolean formula is:

$$(x_1 \vee x_3 \vee \neg x_5) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_3 \vee x_4 \vee \neg x_5) \wedge (x_2 \vee x_4 \vee \neg x_5) \wedge (\neg x_1 \vee x_2 \vee x_5)$$

Note that the figure establishes an edge relation between variables and clauses whereas the clauses in boolean formulas do not have variables but literals of variables.

Problem 8 (Not All Equal 3 SAT Problem (NAE3SAT)). Given a boolean formula in 3-CNF, is it satisfiable so that each clause contains a true and a false literal?

Problems 5—8 are known to be NP-hard and are often used to show other problems are NP-hard as well [16, 11].

1.8 Contribution

The *realizability* problem for a polygonal linkage asks whether a given polygonal linkage has a realization (resp., orientated realization). For a weighted planar (resp., plane) graph,, it asks whether the graph is the contact graph (resp., ordered contact graph) of some disk arrangement with specified radii. These problems, in general, are known to be NP-hard. Specifically, it is NP-hard to decide whether a given planar (or plane) graph can be embedded in $\mathbb{R}^2$ with given edge lengths [6, 8]. Since an edge of given length can be modeled by a suitably long and skinny rhombus, the realizability of polygonal linkages is also NP-hard. The recognition of the contact graphs of unit disks in the plane (a.k.a. coin graphs) is NP-hard [5], and so the realizability of weighted graphs as contact graphs of disks is also NP-hard. However, previous reductions crucially rely on configurations with high genus: the planar graphs in [6, 8] and the coin graphs in [5] have many cycles.

In this thesis, we consider the above four realizability problems when the union of the polygons (resp., disks) in the desired configuration is simply connected (i.e., contractible). That is, the contact graph of the disks is a tree, or the "hinge graph" of the polygonal linkage is a tree (the vertices in the *hinge graph* are the polygons in $\mathscr{P}$, and edges represent a hinge between two polygons). Our main result is that realizability remains NP-hard when restricted to simply connected structures.

Theorem 1. *It is strongly NP-hard to decide whether a polygonal linkage whose hinge graph is a **tree** can be realized.*

Theorem 2. *It is strongly NP-hard to decide whether a polygonal linkage whose hinge graph is a **tree** can be realized with fixed orientation.*

Our proof for Theorem 2 is a reduction from PLANAR-3-SAT (P3SAT): decide whether a given Boolean formula in 3-CNF with a planar associated graph is satisfiable. Our proof for Theorem 1 is a reduction from NOT-ALL-EQUAL-3-SAT (NAE3SAT): decide whether a given Boolean formula in 3-CNF is it satisfiable so that each clause contains a true and a false literal?

Theorem 3. *It is NP-Hard to decide whether a polygonal linkage whose hinge graph is a tree can be realized (both with and without orientation).*

Theorem 4. *It is NP-Hard to decide whether a given tree (resp., plane tree) with positive vertex weights is the contact graph (resp., contact graph) of a disk arrangements with specified radii.*

The unoriented versions, where the underlying graph (hinge graph or contact graph) is a tree can easily be handled with the logic engine method (Section **??**). We prove Theorem 3 for *oriented* realizations with a reduction from PLANAR-3SAT (Section **??**), and then reduce the realizability of ordered contact trees to the oriented realization of polygonal linkages by simulating polygons with arrangements of disks (Section 1.4).

Boris Klemz's Master's thesis "Weighted Disk Contact Graph" shows the Unit Disk Touching Graph Recognition problem is NP-Hard. It also improves the Breu and Kirkpatrick result on Disk Touching Graph Recognition problem [12, 5].

Chapter 2

Decision Problems for Hinged Polygons and Disks

2.1 The Logic Engine

The *logic engine* is a planar, mechanical device that simulates an instance NAE3SAT problem. It was introduced in Bhatt et. al. [4].

2.1.1 Construction of the Logic Engine

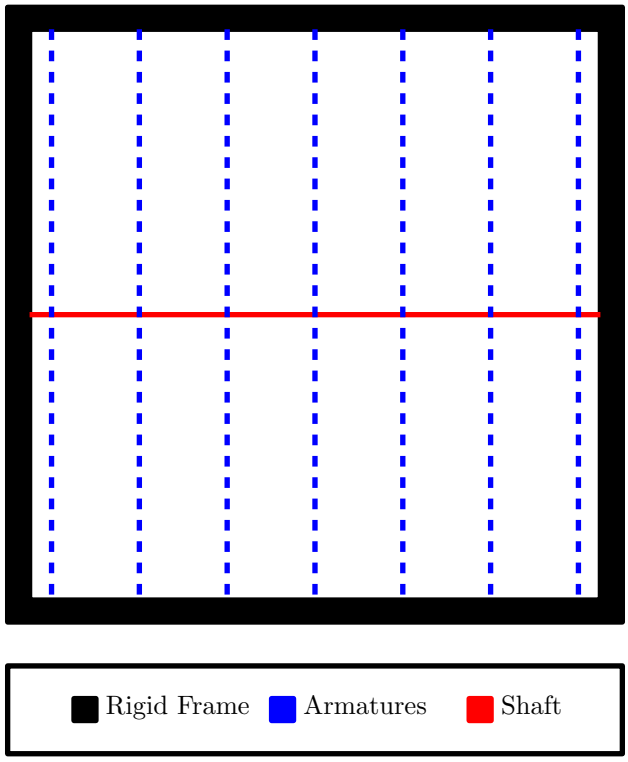


Figure 2.1: A logic engine frame with veritical armatures and a horizontal shaft.

For a given a boolean formula, Φ, in 3-CNF with n variables and m clauses, we construct a logic engine. The logic engine has a *rigid frame* which houses the mechanical components of the the logic engine. The rigid frame is the boundary of which the logic engine can operate within. The *shaft* is a horizontal line segment that is placed at mid-height of the rigid frame. The *armatures* are veritical line segments whose midpoints are on the shaft. Each armature has two orientations with respect to the shaft. There will be some flags on each armature that will be described later. The armatures each have length $2n$ units, the shaft has length m units, and the frame has a height of $2n$ and width of m units.

Each armature corresponds to a variable in Φ. There are two literals for each variable, i.e. the literal x_j and the negated literal $\bar{x}_j$. To describe the flagging arrangement, first partition each armature into $2m$ units, vertical line segments. Label the segments on the j^{th} armature starting from the shaft by $\ell_{j,1},\ldots,\ell_{j,n}$ on one side and $\bar{\ell}_{j,1},\ldots,\bar{\ell}_{j,n}$ on the other side of the shaft. Attach regular triangles, called *flags*, to some of these segments. Each segment is either flagged one or zero flags, i.e. *flagged* or *unflagged*. If the literal x_j is found in clause C_k, then $\ell_{j,k}$ is unflagged. If the literal $\bar{x}_j$ is found in clause C_k, then $\bar{l}_{j,k}$ is unflagged.

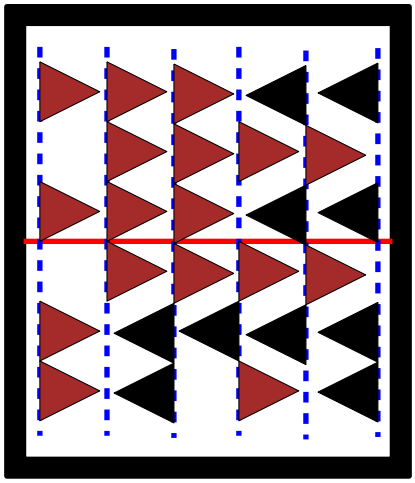
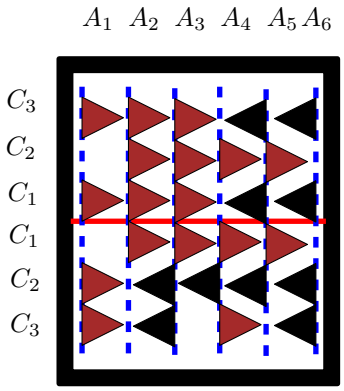


Figure 2.2: A logic engine that corresponds to a boolean formula in NAE3SAT form, Φ. The picture shows the outer rigid frame, the shaft, the armatures that correspond to the variables in Φ, with oriented flags.

Each flag has two orientations with respect to armature it is attached to. Each flag has four potential positions, the flag can reflect left or right about the armature and the armature can reflect up or down about the shaft. The *flags* are equilateral triangles attached to the armatures. The placement of the flags is dependent



$$C_3 = (x_4 \cup \bar{x}_2 \cup \bar{x}_5)$$
$$C_2 = (x_1 \cup x_6 \cup \bar{x}_2)$$
$$C_1 = (\bar{x}_1 \cup \bar{x}_5 \cup x_4)$$

Figure 2.3: A logic engine constructed from the boolean formula $\Phi = C_1 \cap C_2 \cap C_3$.

on the instance of the NAE3SAT boolean formula. Each flag as two orientations.

For the NP hardness reduction in Theorem 6 we need to make sure that all parameters in the logic engine can be specified polynomially in terms of the size of the boolean formula. Given Φ, the corresponding logic engine is constructed as follows: all components will be specified with a quantity and coordinates defined as polynomials in m and n.

Component	Quantity	Set Definition
Rigid Frame	1	$\left\{ (x,y) \in \mathbb{R}^2 \middle\vert \text{The boundary of } \left[\frac{1}{2}, n+\frac{1}{2}\right] \times [-m,m] \right\}$
Shaft	1	$\left\{ (x,y) \in \mathbb{R}^2 \middle\vert x \in \left[\frac{1}{2}, n+\frac{1}{2}\right] \text{ and } y = 0 \right\}$
Armatures	n	For the j^{th} armature we have $\left\{ (x,y) \in \mathbb{R}^2 \middle\vert x = j \text{ and } y \in [-m,m] \right\}$
Flags	$2mn-3m$	if $\ell_{j,k}$ is flagged then the attached flag is a regular triangle with side length 1.

Table 2.1: The quantity and coordinates of the logic engine components.

2.1.2 The mechanics of the logic engine

In any of the following cases, a *collision* of flags occurs:

1. flags in the same row on adjacent armatures point toward each other.
2. a flag from the rightmost armature A_n points towards the outer rigid frame.
3. a flag from the leftmost armature A_1 points inwards of A_1.

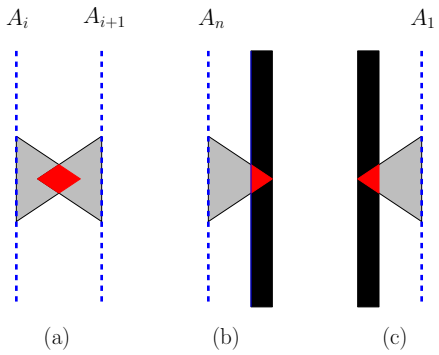


Figure 2.4: (a) Illustrates a adjacent flag collision at the same height, (b) and (c) illustrates a rigid frame collision.

The logic engine representation corresponding to Φ is to be configured such that no horizontally adjacent flags collide and flags do not collide with the rigid frame.

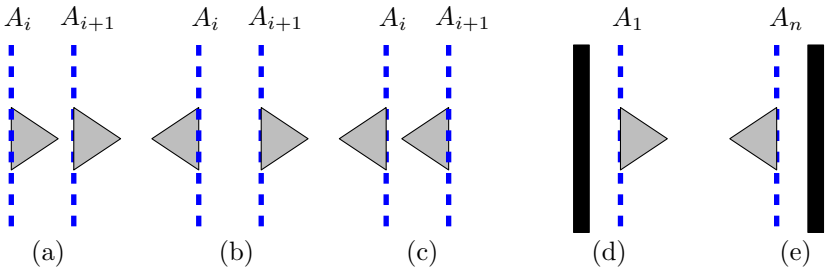


Figure 2.5: The following configuration of adjacent flags and flags that are adjacent to the rigid frame.

Lemma 1. *A row has a collision-free configuration if and only if it has at least one unglagged armature.*

Proof. Suppose all armatures are flagged in a row. The flag on armature A_1 must point to the right otherwise we result in a rigid frame collision. A_2 must point to the right otherwise we result in a rigid frame collision. Without loss of generality, A_i and A_{i+1} must point to the right in order to prevent an adjacent flag collision. This implies that A_n must also point to the right which results into a rigid frame collision.

A same argument holds with the argument beginning with the flag on the armature A_n pointing to the left. Thus there is no collision-free configuration with all armatures flagged.

Suppose there is an unflagged armature in a row. Turn all flags towards the nearest unflagged armature. If there are flags on A_1 and A_n, point toward the interior thus they do not collide with the rigid frame. If there are flags on two consecutive armatures, they do not collide because the nearest unflagged armature cannot be between them. Therefore the row has a collision-free configuration. □

A logic engine is said to be *collision-free configurable* when every row has a collision-free configuration.

2.1.2.1 The Relationship of the Logic Engine and NAE3SAT

We show that given an boolean formula in 3-CNF form, Φ, and a truth assignement, τ, where the variables are given a truth assignment such that there is at least one true literal and one false literal in each clause of Φ, then the corresponding logic engine to Φ is collision-free configurable.

Theorem 5. *Given an instance of a NAE3SAT, it is a "yes" instance if and only if the corresponding logic engine is collision-free configurable.*

Proof. Suppose we have an instance of a *NAE3SAT* that is a "yes" instance. This implies that there is a truth assignment such that each clause contains a true and a false literal. Now consider the logic engine corresponding to this instance. We now show that it has a collision free configuration.

For variables that are true, configure the armatures such that the flags corresponding to the non-negated literals reside above the shaft and the flags that correspond to the negated literals reside below this shaft. For variables that are false, configure the armatures in the opposite orientation. Each clause corresponds to a pair of rows in the logic engine, one row for non-negated literals and one for negated literals. Because the *NAE3SAT* is a yes instance, every row contains at least one unflagged armature. By Lemma 1, every row has a collision-free configuration.

Suppose we have an instance of a *NAE3SAT* such that the corresponding logic engine has a collision-free configuration. By Lemma 1 every row at least one unflagged armature. The k^{th} clause is represented by the k^{th} rows above and below the shaft. If the literal x_j is found in clause C_k, then the armature is unflagged in that row. If the literal $\bar{x}_j$ is found in clause C_k, then $\bar{l}_{j,k}$ is unflagged. All flags corresponding to negated literals reside below the shaft and flags corresponding to non-negated literals reside above the shaft. All together we have that every clause has a true literal and a false literal. Thus, we have a 'yes' instance of the *NAE3SAT*. □

Theorem 6. *Deciding whether a logic engine is collision-free configurable is NP-Hard.*

Proof. In table **??**, we defined the components of the logic engine in terms of polynomials in m and n. If there were a polynomial time algorithm that decides whether a given logic engine is collision-free configurable, then by Theorem 5 we would have a polynomial time algorithm to decide whether an instance of the NAE3SAT is a 'yes' instance. Since NAE3SAT is NP-Hard [9], there is no such algorithm unless $P = NP$. □

2.2 Logic Engines Represented as Polygonal Linkages

In the previous section, we introduced the logic engine. This section builds an analogous structure that is formed from a polygonal linkage and we may interchangeably say subcomponent for polygon. We can modify the mechanical structure of the logic engine to form a polygonal linkage. For a given a boolean formula, Φ, in 3-CNF with n variables and m clauses, the rigid frame is broken into two polygons, each polygon on the extremity of the structure. The shaft is broken into n polygons. Each armature is broken into two parts, each part containing m subcomponents. In figure 2.7, each flag becomes a rectangle.

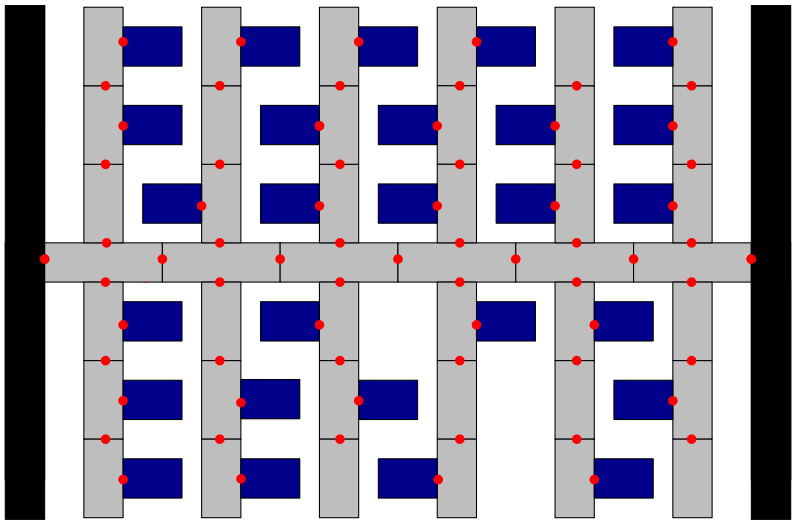


Figure 2.7: A logic engine realized as a polygonal linkage.

2.2.1 Construction of the Polygonal Linkage Logic Engine

Suppose we are given an boolean formula with m clauses and n variables in 3-CNF form, Φ, we construct the polygonal linkage similarly to the logic engine. The corresponding polygonal linkage $P_\ell = (\mathscr{P}, \mathscr{H})$ is detailed in Table **??**.

Component	Height	Width	Quantity
Large Frame Subcomponent	$2 \cdot m$	1	2
Shaft Subcomponent	1	3	n
Armature Subcomponent	2	1	$2 \cdot m$
Flag	1	1.5	$2mn - 3m$

Table 2.2: The components of $\mathscr{P}$ specified polynomially in terms of the size of the boolean formula Φ.

The large frame subcomponents are hinged on the left most and right most shaft subcomponents. Each adjacent shaft subcomponents are hinged and each shaft subcomponent has two orientations, a reflecion up and a reflection down about the shaft hinge points. On each shaft subcomponent there are two armature shaft subcomponents, one above the shaft subcomponent and one below the shaft subcomponent. By the two orientations of the shaft subcomponent, each armature subcomponent has to possible positions. Each armature comprises of m armature subcomponents that are hinged together; in total there are $2n$ armatures. Each armature subcomponent has two orientations, a reflection left and a reflection right about the armature hinge points. Label the armature subcomponents on the j^{th} armature starting from the shaft by $\ell_{j,1}, \ldots, \ell_{j,n}$

on one side and $\bar{\ell}_{j,1}, \ldots, \bar{\ell}_{j,n}$ on the other side of the shaft. Attach a rectangular flag specified in Table **??**, to some of these segments. Each segment is either flagged one or zero flags.

1. If the literal x_j is found in clause C_k, then $\ell_{j,k}$ is unflagged.
2. If the literal $\bar{x}_j$ is found in clause C_k, then $\bar{l}_{j,k}$ is unflagged.

Each flag has two orientations with respect to armature it is attached to. Each flag has four potential positions, the flag can reflect left or right about the armature and the armature can reflect up or down about the shaft.

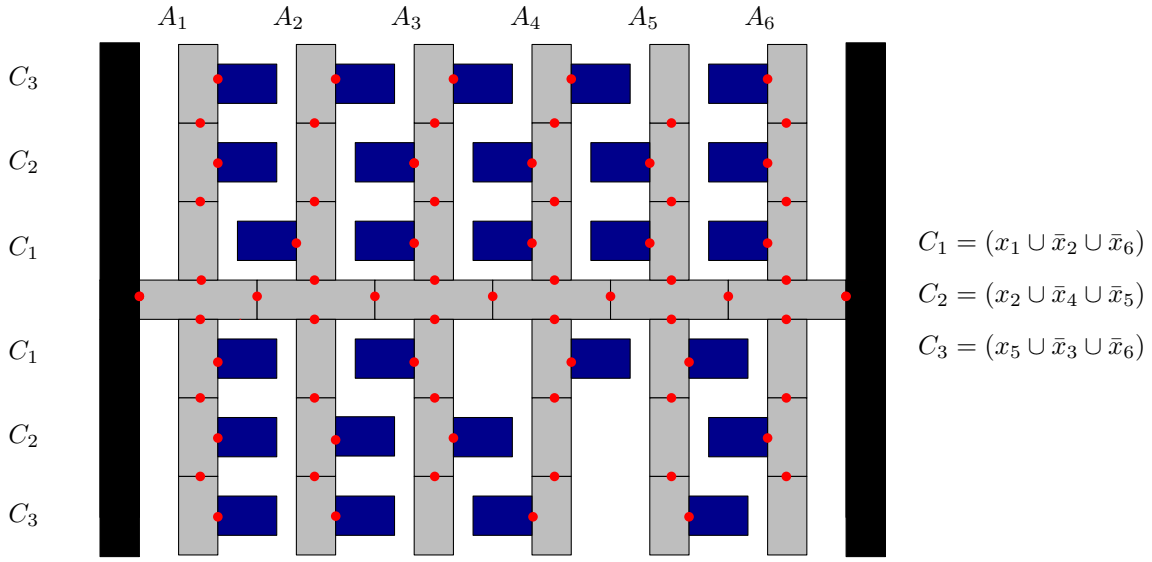


Figure 2.8: A polygonal linkage logic engine that corresponds to the boolean formula $\Phi = C_1 \cap C_2 \cap C_3$.

Theorem 7. *Given an instance of a NAE3SAT, it is a "yes" instance if and only if the corresponding polygonal linkage logic engine has a collision-free configuration.*

Proof. Suppose we have an instance of a *NAE3SAT* that is a "yes" instance. This implies that there is a truth assignment such that each clause contains a true and a false literal. Now consider the polygonal linkage logic engine corresponding to this instance. We now show that it has a collision free configuration.

For variables that are true, configure the armatures such that the flags corresponding to the non-negated literals reside above the shaft and the flags that correspond to the negated literals reside below this shaft. For variables that are false, configure the armatures in the opposite orientation. Each clause corresponds to a pair of rows in the polygonal linkage logic engine, one row for non-negated literals and one for negated literals. Because the *NAE3SAT* is a yes instance, every row contains at least one unflagged armature. By Lemma 1, every row has a collision-free configuration.

Suppose we have an instance of a *NAE3SAT* such that the corresponding polygonal linkage logic engine has a collision-free configuration. By Lemma 1 every row at least one unflagged armature. The k^{th} clause is represented by the k^{th} rows above and below the shaft. If the literal x_j is found in clause C_k, then the armature is unflagged in that row. If the literal $\bar{x}_j$ is found in clause C_k, then $\bar{l}_{j,k}$ is unflagged. All flags corresponding to negated literals reside below the shaft and flags corresponding to non-negated literals reside above the shaft. All together we have that every clause has a true literal and a false literal. Thus, we have a 'yes' instance of the *NAE3SAT*. □

Chapter 3

Realizability of Polygonal Linkages with Fixed Orientation

We begin the chapter with describing several gadgets that translates the associated graph $A(\Phi)$ of a P3SAT boolean formula. These gadgets will be used together to form a special hexagonal tiling that behaves in a similar nature to the logic engine that encoded a NAE3SAT instance of Chapter 2 but instead encodes a Planar 3-SAT and its associated graph. Together the gadgets will form what is called the auxilary construction. A hexagonal tiling and several gadgets enclosed in a frame (a frame that is conceptually similar to the frame found in a logic engine) would then be used to prove Theorem 1: *it is strongly NP-hard to decide whether a polygonal linkage whose hinge graph is a* ***tree*** *can be realized with counter-clockwise orientation.*

Our proof is a reduction from P3SAT. Given an instance Φ of P3SAT with n variables and m clauses and its associated graph $A(\Phi)$, we construct a simply connected polygonal linkage $(\mathscr{P},H)$, of polynomial size in n and m, such that Φ is satisfiable if and only if $(\mathscr{P},H)$ admits a realization with fixed orientation.

We construct a polygonal linkage in two main steps: first, we construct an auxiliary structure where some of the polygons have fixed position in the plane (called *obstacles*), while other polygons are flexible, and each flexible polygon is hinged to an obstacle. Second, we modify the auxiliary construction into a polygonal linkage by allowing the obstacles to move freely, and by adding new polygons and hinges as well as an exterior *frame* that holds the obstacle polygons in place. All polygons in our constructions are regular hexagons or long, skinny rhombi. In Chapter 4 we can "simulate" these shapes with disk arrangements to show related results.

Storer, Tamassia, and Tollis showed that if $G=(V,E)$ is planar, it can be embedded in an $|V|\times|V|$ grid with $S(G)=2.4|V|+4$ bends [20, 22]. Note that this result implies that a planar graph can be embedded in a grid of size $S(G)\times S(G)$; and this allows $S(G)$ to act as a fundamental polynomial problems with planar graphs defined in a embedding in a grid. We can define a *bend polynomial*, $s(n,m)$ that will be used to construct our gadgets which will be used to prove Theorem 1. For our constructions, our bend polynomial can be any polynomial strictly greater than $S(G)$, e.g.:

$$6(n+m)+4.$$

The table below is a glossary of formulas that are used to throughout this chapter. It will serve as a useful reference for the reader.

$z(n,m)$	$=$	$4s(n,m)$
$J_h(z)$	$=$	$6z(n,m)+1=24s(n,m)+1$
$J_d(z)$	$=$	$4z(n,m)+1=16s(n,m)+1$
$N(n,m)$	$=$	$\frac{5t-1}{2}=\frac{5s^{\kappa}-1}{2}$
$t(n,m)$	$=$	s^{κ}
$H(n,m)$	$=$	$(12s+1)(5t-1)\sqrt{3}+12s\left(\sqrt{3}+\frac{1}{250t-50}\right)$
		$x\le \sin^{-1}x\le x+\frac{x^3}{6}$ $\quad 0<x<1$
		$x-\frac{x^3}{3}\le \tan^{-1}x\le x$ $\quad 0<x<1$
		$x-\frac{x^3}{6}\le \sin x\le x$ $\quad 0<x<1$
		$1-\frac{x^2}{2}\le \cos x\le 1-\frac{x^2}{2}$ $\quad 0<x<1$
		$1\le \sec x\le 1+\frac{x^2}{2}$ $\quad 0<x<1$

The formula $t(n,m) = s^{\kappa}$ has an exponent κ which is a sufficiently large integer that is chosen later on so that all arguments in this chapter are satisfied. The trigonmetric functions in the table above each are expressed as either the first term or the first and second term of their Maclaurin Series expansion.

Modifying the Associated Graph of a P3SAT. Given an instance of P3SAT boolean formula Φ of n variables and m clauses with associated graph $A(\Phi)$, we construct a finite *honeycomb* grid $H_{A(\Phi)}$ of regular hexagons over the plane centered at origin. We modify the associated graph drawing $A(\Phi)$ by overlaying it onto a honeycomb in the following way:

1. **Variable:** A vertex representing a variable shall encompass a consecutive set of hexagons along a horizontal line in the honeycomb (see Figure 3.1).

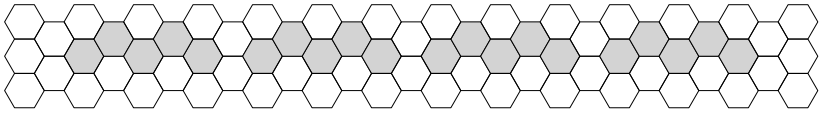
   

   Figure 3.1: The four shaded groups of horizontally adjacent hexagons represent four distinct variables from a boolean formula in the honeycomb.

   Let $D = \max_{v \in V} \deg(v)$ where V is the set of vertices of $A(\Phi)$. Every variable vertex v must encompass at least $2 \cdot \deg(v)$ consecutive hexagons but can encompass up to $2 \cdot D$ consecutive hexagons.

2. **Clause:** A vertex representing a clause shall be a vertex of a hexagon in the honeycomb.

3. **Edge:** Edges of the associated graph $A(\Phi)$ are paths between the variable x_i and clause C_j. An edge $\{x_i, C_j\}$ of the associated graph is pariwise edge disjoint. The edges of the drawing shall traverse the edges of hexagons in a vertically or horizontally zigzagging manner (see Figure 3.2) in the honeycomb from the literal to the corresponding clause. Edges traverse a hexagon in two edges vertically, three edges horizontally. The vertical zigzagging edge segments traverse the left or right sides of a hexagon(s). The horizontal zigzagging edge segments traverse the top or bottom halves of a hexagon(s). When the edge transisitions from a vertical to horizontal traversal, the edge traverses in over 4 edges about the hexagon. The length of the edges are bounded above by $6 \cdot (\ell_1(x_i, C_j) + D)$ where ℓ_1 is the L_1 norm.

Figure 3.2 illustrates an associated graph of a P3SAT overlayed on a honeycomb. This type of construction emulates an *orthoganal drawing* over a hexagonal grid; an orthoganal drawing where edges are drawn with alternating vertical and horizontal line segments. Let the region in which the construction lies in be a regular hexagon region with polynomial side length $s(n,m)$.

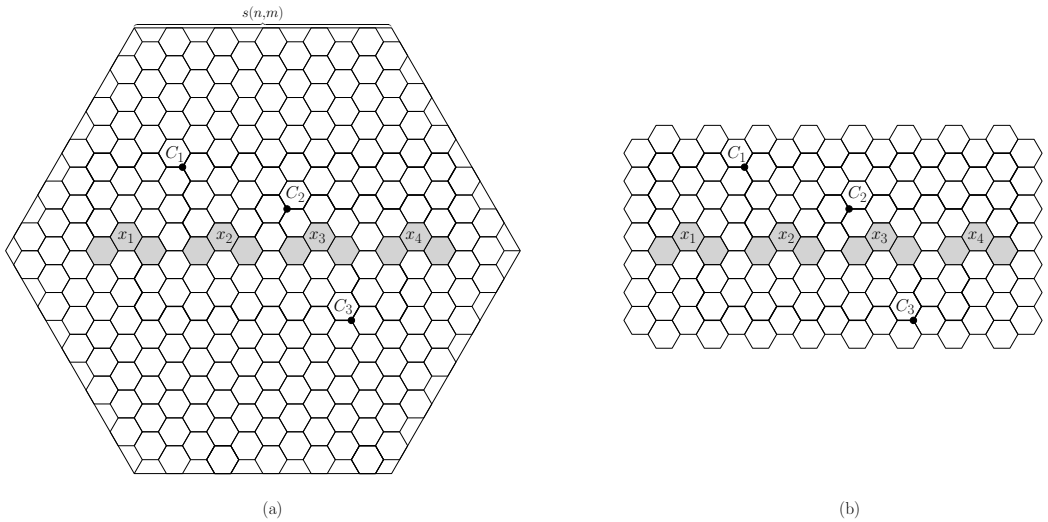


Figure 3.2: (a) This is an instance of an associated graph for a P3SAT overlayed onto a honeycomb grid and placed into a regular hexagonal region. This honeycomb graph could correspond to Boolean formula $(\neg x_1 \vee \neg x_2 \vee x_4) \wedge (x_2 \vee \neg x_3 \vee x_4) \wedge (x_1 \vee \neg x_3 \vee \neg x_4)$. (b) This is the same instance as (a) shown without the hexagonal region.

The honeycomb construction will act as preliminary concept that will be refined further in the Auxilary Contruction.

3.1 Auxilary Construction

Let Φ be a Boolean formula of P3SAT with variables $x_1, \ldots, x_n$ and clauses $C_1, \ldots, C_m$, where $A(\Phi)$ is the associated planar graph and $\tilde{A}(\Phi)$ be corresponding honeycomb graph. We continue to modify $\tilde{A}(\Phi)$ to form the auxilary construction. Consider a large (polynomial-size) regular hexagon J with side length $s(n,m)$ that contains all gadgets in our construction and hexagonal grid. For each hexagon of the hexagonal grid contained in J, scale the hexagon in the following way: first we fix the center of the hexagon and then scale (shrink) the hexagon; adjacent hexagons in the honeycomb no longer touch each other and form corridors and junctions between the hexagons (See Figure 3.3).

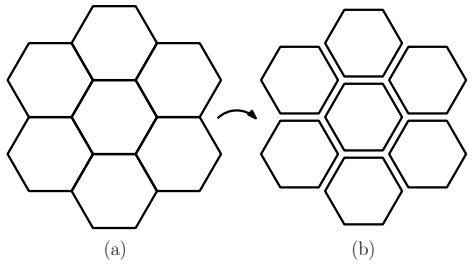


Figure 3.3: (a) This figure shows a region of a hexagonal grid scaled in place to form corridors between adjacent hexagons(shown in (b)).

Formally, let a *corridor* be a channel between two adjacent hexagons and a *junction* be a region where three corridors meet.

Formal Decscription of the Auxilary Construction. Given the side length of J, $s(n,m)$, we need to scale the grid of hexagons of the hexagonal grid in the interior of J accordingly.

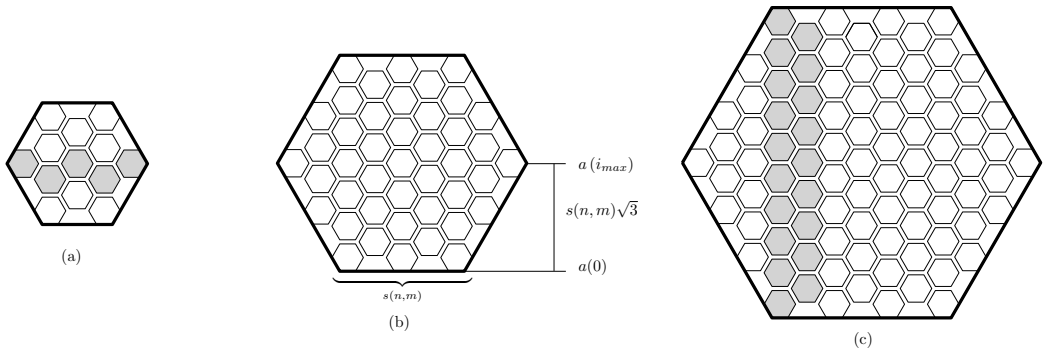


Figure 3.4: (a) shows a *formal auxilary construction* with $k = 2$. k is the number of hexagons of the hexagonal grid in the interior of J that is on the bottom most row. (b) shows a formal auxilary construction with $k = 3$. (c) shows a formal auxilary construction with $k = 4$. Note that in each figure

In Figure 3.4, we show the three smallest possible formal constructions of J, J_1, J_2, J_3. A *formal construction* of J is when six hexagons of the hexagonal grid each have two adjacent lie on the perimeter of J. We have shown informal construction earlier where this does not occur. Unless otherwise specified, we will assume the use formal constructions. Each of the figures in Figure 3.4 shows J in bold and the hexagonal grid in its interior. Notice that in each case we have six hexagons of the hexagonal grid with each of the six hexgaons having two adjacent sides that lie on the perimeter of J.

The height and diamater of J can be described by the number of hexagons in the grid a vertical or horizontal line may cross. We'll denote these qualities as the hexagonal height and hexagonal diameter of J. Figure Figure 3.4(c) and Figure 3.4(a) show the hexagons of the hexagonal height and hexagonal diamters of J_1 and J_3 respectively. The formula for calculating hexagonal height of J_z is

$$J_h(z) = 6z + 1 \tag{3.1}$$

The formula for calcuating the hexagonal diameter of J_z is

$$J_d(z) = 4z + 1 \tag{3.2}$$

If an associated graph of a P3SAT instance can be encoded into a honeycomb grid of size $s(n,m) \times s(n,m)$, then let $z(n,m) = 4 \cdot s(n,m)$ to enclose the same honeycomb to be enclosed into the interior of J_z.

Figure 3.4(b) illustrates the side length of J as $s(n,m)$ and half the height of J as $s(n,m)\sqrt{3}$. For any J, there is a fixed number of hexagons in the hexagonal grid that lie on one side of the perimeter of J; denote this number as k. We can denote the number of hexagons in a row of the hexagonal grid in J_z with the following sequence as follows:

$$\begin{aligned} a(0) &= k \\ a(1) &= k-1 \\ a(2) &= k \\ a(i) &= a(i-3)+1 \\ 0 \leq \quad i \quad &\leq \left\lceil \tfrac{J_h(z)}{2} \right\rceil \end{aligned} \tag{3.3}$$

The i^{th} number of Sequence 3.3 indicates the number of hexagons on the i^{th} row of the hexagonal grid in the interior of J from the perimeter of J up to the half height of J. In Figure 3.4(c) from the bottom to the mid-height of J, the sequence $a(i)$ (i.e. the number of hexagons in each subsequent row) is 4, 3, 4, 5, 4, 5, 6, 5, 6, 7. Denote the hexagons of the hexagonal grid in J as *obstacle hexagons*.

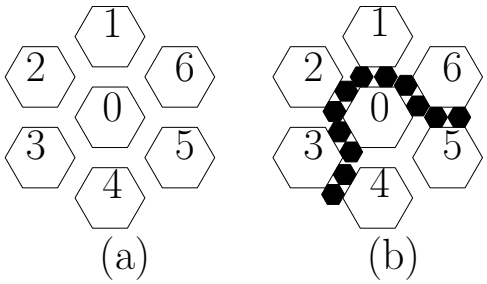


Figure 3.5: (a) A region of the honeycomb shown with scaling. The corridors and junctions formed from the first scaling is preserved after scaling the honeycomb grid to where the side lengths of the hexagon are $N(n,m)$. (b) The same region in (a) containing flags.

Let the numbered hexagons of Figure 3.5(a) be obstacle hexagons that are fixed. In Figure 3.5(b), we have smaller hexagons within some corridors and junctions. These hexagons are flags. For each edge in $\tilde{A}(\Phi)$, we insert flags into the corridor corresponding to that edge. Flexible hexagons are hinged at the vertex closest to origin and the side of the corridor (See Figure 3.6). Let $t(n,m) = s^{\kappa}$ be the number of flags in a corridor (see Figure 3.6). Scale J and the obstacle hexagons in the interior of J independently from their centers (see Figure 3.3) such that each obstacle hexagon has side length:

$$N(n,m) = \frac{5t(n,m)-1}{2}.$$

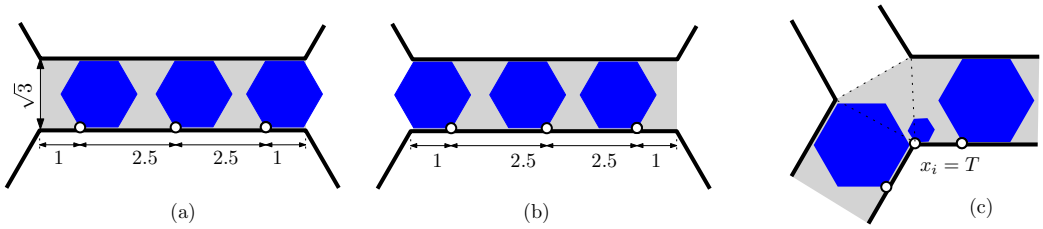


Figure 3.6: (a) A corridor when all unit hexagons are in state R. (b) A corridor where all unit hexagons are in state L. (c) A junction where a small hexagon between two corridors ensures that at most one unit hexagon enters the junction from those corridors.

Between two adjacent obstacle hexagons, there is a $\frac{5t-1}{2} \times \sqrt{3}$ rectanglar corridor. Three adjacent corridors meet at a regular triangle, which we call a junction. For each corridor, there are two junctions adjacent to it; of these two junctions, we denote the junction from which a flag in the corridor enters into as the *active junction* (see Figure 3.7).

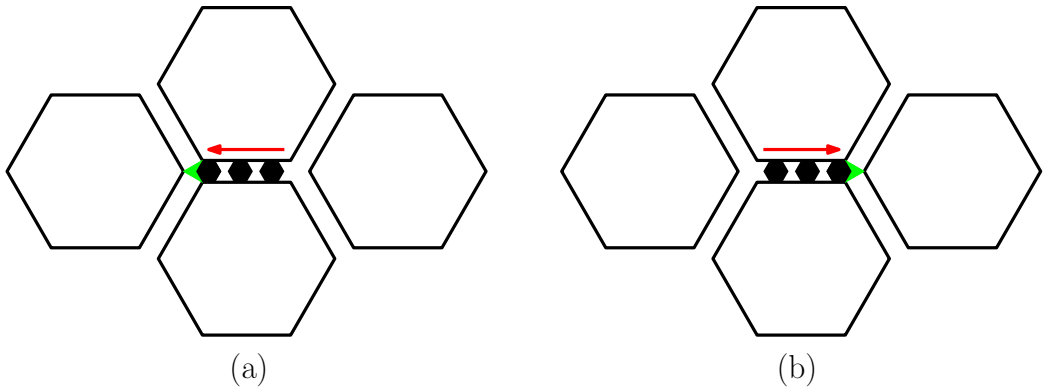


Figure 3.7: The active junction in (a) is the junction on the left and in (b) the active junction is on the right. The active junction is the junction in which a flag enters from a corridor.

We next describe variable, clause, and transmitter gadgets. The basic building block of both variable and transmitter gadgets consists of t regular hexagons of side length 1 (*unit hexagons*, for short) attached to a wall of a corridor such that the hinges divide the wall into $t+1$ intervals of length $(1, 2.5, \ldots, 2.5, 1)$ as shown in Fig. 3.6(a-b).

In some of the junctions, we attach a small hexagon of side length $\frac{1}{3}$ to one or two corners of the junction (see Fig. 3.6(c) and Fig. 3.11).

Variable Gadget. The **variable gadget** for variable x_i is constructed as follows. Recall that variable x_i corresponds to a cycle in the associated graph $\tilde{A}(\Phi)$, which has been embedded as a cycle in the hexagonal tiling, with corridors and junctions. In each junction along this cycle, attach a small hexagon in the common boundary of the two corridors in the cycle. Figure 3.8 depicts a *variable gadget* in the hexagonal grid.

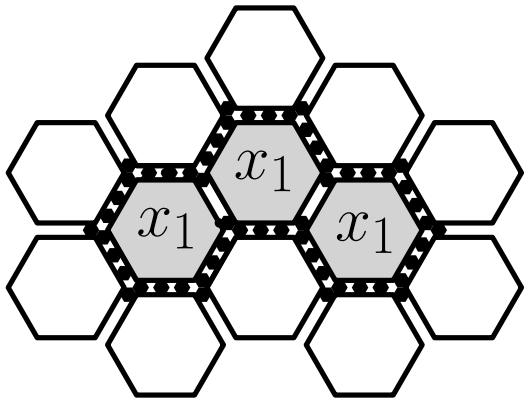


Figure 3.8: This depicts a variable gadget with $x_1 = T$. Carefully note that the flags around x_1 are in the state R. Corridors adjacent to two obstacles of a variable in the honeycomb do not have t flags; these corridors simply have the flexible hexagons at the junctions.

Clause Gadget. Recall that a clause from a Boolean formula Φ in 3-CNF has three literals. If Φ is a 'yes' instance, then at least one literal in every clause of Φ is true. We construct the clause gadget to model this fact about Boolean formulas in 3-CNF.

The **clause gadget** lies at a junction adjacent to three transmitter gadgets (see Fig. 3.9 and Section 3.1). At such a junction, we attach a unit line segment to an arbitrary vertex of the junction, and a small hexagon of side length $\frac{1}{3}$ to the other end of the segment. If unit hexagons enter the junction from all three corridors

(i.e., all three literals are false), then there is no space left for the small hexagon.

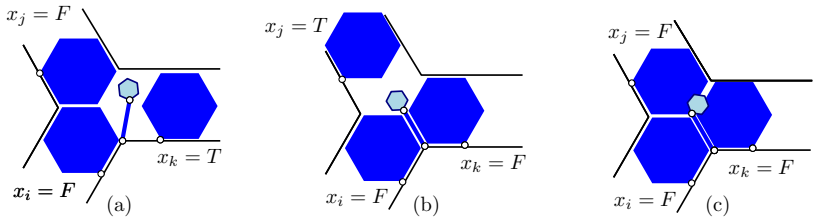


Figure 3.9: (a-b) A clause gadget $(x_i \vee x_j \vee x_k)$ is realizable when at least one of the literals is TRUE. (c) The clause gadget cannot be realized when all three literals are FALSE.

But if at most two unit hexagons enter the junction (i.e., one of the literals is true), then the unit segment and the small hexagon are realizable.

Transmitter Gadget. In the planar 3-SAT graph $A(\Phi)$, every variable vertex has an associated cyclic order of edges. Suppose we have a variable vertex x_i with counter-clockwise cyclic order of edges $(\{x_i, C_1\}, \{x_i, C_2\}, \ldots, \{x_i, C_k\})$. Assign distinct junctions of the variable cycle of x_i to the edges $\{x_i, C_j\}$ in the same cyclic order (refer to Figure 3.10 for an example).

A **transmitter gadget** is constructed for each edge $\{x_i, C_j\}$ of the graph $A(\Phi)$; it consists of a sequence of junctions and corridors from a variable gadget's junction to a clause junction.

For each junction in the transmitter gadget, we attach a small hexagon in the junction as shown in Figure 3.11 except at the clause junction. Choosing the location of the small hexagon depends on whether the non-negated or negated literal is found in the clause.

(a) For an edge (x_i, C_j) of the graph $A(\Phi)$, if the non-negated literal of x_i exists in C_j, attach the small hexagon to the left side of the junction (see Figure 3.10(a)).

(b) For an edge (x_i, C_j) of the graph $A(\Phi)$, if the negated literal of x_i exists in C_j, attach thhe small hexagon to the right side of the junction (see Figure 3.10(b)).

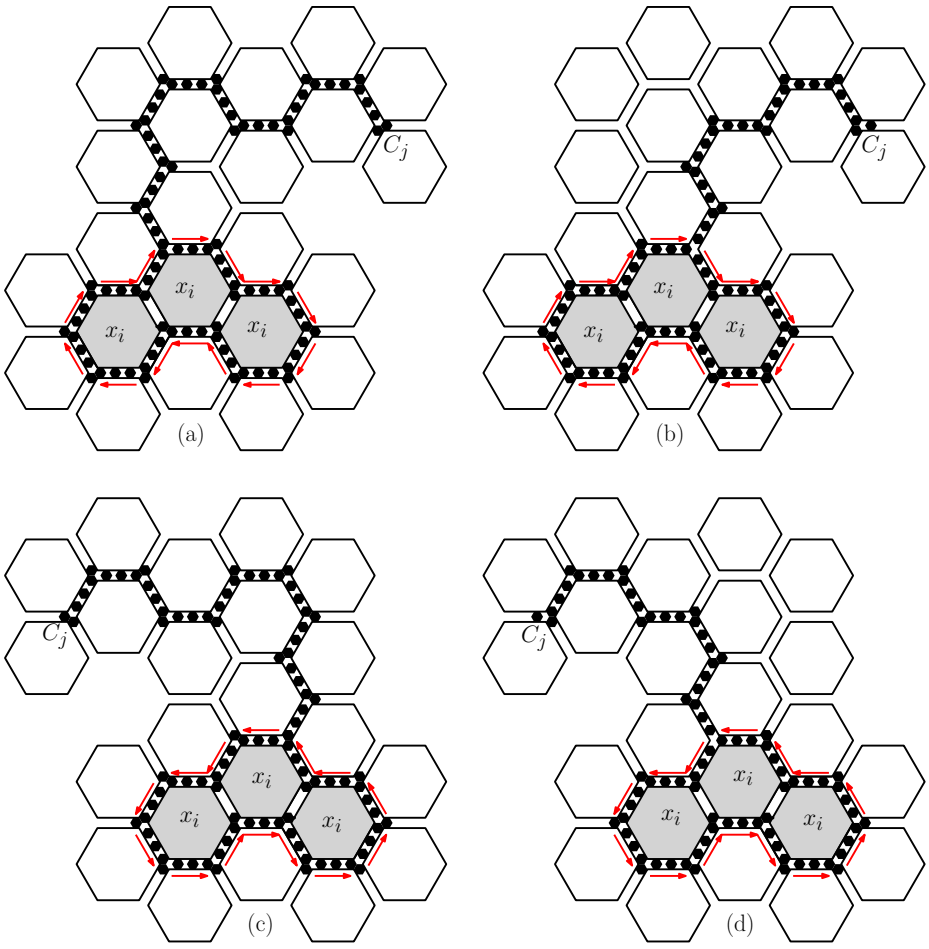


Figure 3.10: These four figures depict an example of placing a transmitter gadget corresponding to edge $\{x_i, C_j\}$.

Figure 3.10 shows an example of each rule on choosing a junction to attach a transmitter gadget. The first column transmits a "true" value between the variable gadget and clause junction. The second column transmits a "false" value between the variable gadget and clause junction. The variable gadgets in the first row are are in state R, i.e. variable $x_i = T$. The variable gadgets in the second row are are in state L, i.e. variable $x_i = F$.

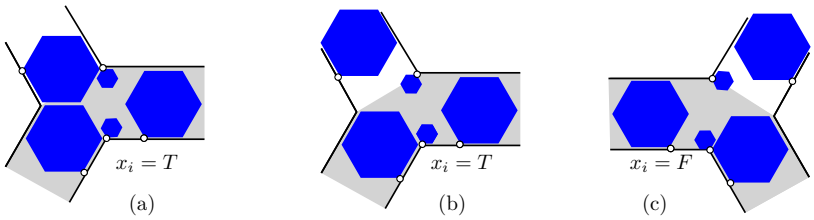


Figure 3.11: The common junction of a variable gadget and a transmitter gadget. (a) When $x_i = T$, a hexagon of the transmitter may enter the junction of the variable gadget. (b) When $x_i = T$, the transmitter gadget has several possible realizations. (c) When $x_i = F$, no hexagon from the transmitter enters a junction of the variable gadget.

3.1.1 Functionality of the Auxilary Construction and Gadgets

If the literal x_i (resp., $\bar{x}_i$) appears in C_j, then we attach a small hexagon to the corner of this junction such that if $x_i = F$ (resp., $\bar{x}_i = F$), then the unit hexagon of the transmitter gadget cannot enter this junction.

A variable gadget for vertex v in the associated graph of a P3SAT Boolean formula encompasses at least $2 \cdot \deg(v)$ consecutive obstacle hexagons. The arrangement of the consecutive obstacle hexagons are in staggered fashion about a horizontal line where there are at least $\deg(v)$ obstacle hexagons in the upper portion of the staggering arrangement and at least $\deg(v)$ obstacle hexagons in the lower portion of the staggering arrangement.

Section 3.1 is a formal description of the auxilary construction and its gadgets. This subsection covers the underlying assumptions and proofs about the functionality of the auxilary construction. The first observations about the functionality of the auxilary construction are about the flags.

Observation 1. (1) If the leftmost hexagon is in state R, then all t hexagons are in state R, and the rightmost hexagon enters the junction on the right of the corridor.

(2) Similarly, if the rightmost hexagon is in state L, then all t hexagons are in state L, and the leftmost hexagon enters the junction on the left of the corridor.

Observation 1 and the small hexagons ensure that the state of any unit hexagon along the cycle determines the state of all other unit hexagons in the cycle. This property defines the binary variable x_i: If $x_i = T$, then all unit hexagons in the top horizontal corridors are in state R; and if $x_i = F$, they are all in state L.

When a binary variable $x_i = T$, we will say that the variable in state R and that the cycle of small hexagons around the variable gadget are in a "clockwise direction". When a binary variable $x_i = F$, we will say that the variable is in state L and that the cycle of small hexagons around the variable gadget are in a "counter-clockwise direction".

The proof of the Observation 1 is similar to the proof of Lemma 1 regarding a row in a logic engine having a collision-free configuration.

Proof. Suppose the leftmost hexagon, h_1, is in state R in a corridor. Denote the t flags in a corridor as h_1, h_2, …, h_t from leftmost to rightmost respectively. h_2 must be in state R otherwise we result in a collision between h_1 and h_2. Without losss of generality, h_i and h_{i+1} must be in a state R in order to prevent an adjacent flag collision. This implies that rightmost flag h_t must also be in state R; this implies that h_t enters the junction that is on the right of the corridor.

Similarly, suppose the rightmost hexagon, h_t, is in state L in a corridor. Denote the t flags in a corridor as h_1, h_2, …, h_t from leftmost to rightmost respectively. h_{t-1} must be in state L otherwise we result in a collision between h_t and h_{t-1}. Without losss of generality, h_i and h_{i+1} must be in a state L in order to prevent an adjacent flag collision. This implies that rightmost flag h_1 must also be in state L; this implies that h_1 enters the junction that is on the left of the corridor. □

The flags of the auxilary construction help communicate the boolean value of a variable gadget to the rest of the auxilary construction. This communication property of the flags in a corridor is analagous to the flags in a row of a logic engine.

Each junction is a regular triangle, adjacent to three corridors. In some of the junctions, we attach a small hexagon of side length $\frac{1}{3}$ to one or two corners of the junction (see Fig. 3.6(c) and Fig. 3.11). Importantly, we have the following observation:

Observation 2. If a small hexagon is attached to a vertex at a junction between two adjacent corridors, then a flag can enter the junction from at most one of those corridors.

Proof. Suppose there is a small hexagon attached to a vertex at a junction between two adjacent corridors. Suppose it is not that case that a flag can enter the junction from at most one of these adjacent corridors. Then there are two flags entering the junction, one from each adjacent corridor. The angular sum of the vertex about the adjacent corridors consists of the obstacle hexagon, both flags, and the small unit hexagon. Each angle of each hexagon is $\frac{2\pi}{3}$ radians, totalling to an angular sum of $\frac{8\pi}{3} > 2\pi$. This is a contradiction with the total angular sum of a vertex on the plane to be 2π. □

Observation 1 and the small hexagons ensure that the state of any unit hexagon along the cycle determines the state of all other unit hexagons in the cycle. This property defines the binary variable x_i: If $x_i = T$, then all unit hexagons in the top horizontal corridors are in state R; and if $x_i = F$, they are all in state L.

Suppose there is an edge $\{x_i, C_j\}$ in the graph $A(\Phi)$.

Lemma 2. *If $x_i = T$ and its negated literal is in C_j, then a flag enters into the clause gadget of C_j, otherwise it need not enter; if $x_i = F$ and its non-negated literal is in C_j, then a flexible hexagon enters into the clause gadget of C_j, otherwise it need not enter.*

Proof. The transmitter gadget for each literal is placed on an active junction of the variable gadget. This junction is "activated" by the variable gadget. By Observation 2, the flag nearest of the transmitter gadget to the variable gadget does not enter the transmitter-variable junction. By Observation 1 and the state of the flag nearest of the transmitter gadget to the variable gadget implies that the flags in that transmitter corridor activate the junction opposite the transmitter-variable junction. The subsequent flags in the transmitter gadget corridors have the same state of the flag in the transmitter gadget nearest of the transmitter-variable junction by Observations 1 and 2. This activation process continues up to the clause junction and the flag in the transmitter gadget nearest the clause junction enters the clause junction. □

Lemma 3. *Hexagons in a clause junction have a non-overlapping placement if and only if at least one of the three literals is true.*

Proof. Suppose we have a hexagons in a clause junction that have a non-overlapping placement. To show that there is at least one of the three literals is true, we do a proof by contradiction. Suppose all literals of the clause are false. If all literals of the clause are false, then all flags in each transmitter gadget nearest their clause junction enters the clause junction, as shown in Figure 3.9(c) which show the small hexagon overlapping flags in the clause junction, a contradiction with hexagons in the clause junction have a non-overlapping placement.

If at least one of the three literals is true, then by Lemma 2, this literal's flag need not enter the transmitter-variable junction. There allows for the small hexagon in the clause junction to move into the area where this literal's flag could enter the junction and thus allow non-overlapping placement of hexagons in the junction. □

For a variable gadget x_i, place horizontal axis h at mid-height of the gadget. Then we have the following lemma:

Lemma 4. *If variable $x_i = T$, then all flags above h are in state R and all flags below h are in state L; if variable $x_i = L$, then all flexible hexagons above h are in state L and all flexible hexagons below h are in state R.*

Proof. Suppose we have two adjacent corridors k_i and k_{i+1} sharing junction J_i and without loss of generality, k_i is the left most corridor. Observation 2 implies that there can only be one hexagon entering J_i from either k_i or k_{i+1}. If the hexagon that enters J_i is from corridor k_i, then this hexagon has state R and all flags in corridor k_i are in state R by Observation 1. Since the nearest flag of corridor k_{i+1} cannot enter the junction J_i, it must also have state R. All flags in corridor k_{i+1} are in state R by Observation 1.

The argument is similar if the hexagon entering J_i is from corridor k_{i+1} and all flags in both corridors k_i and k_{i+1} have state L.

Because variable gadgets form a simple cycle of corridors and junctions $(k_1, J_1, k_2, J_2, \ldots, k_n, J_n)$ and the argument above, all flags about a variable gadget have the same state. □

Lemma 5. *For every instance Φ of P3SAT, the above polygonal linkage with flexible and obstacle polygons has the following properties: (1) it has polynomial size; (2) its hinge graph is a forest; (3) it admits a realization such that the obstacle polygons remain fixed if and only if Φ is satisfiable.*

Proof. (1) We can bound the number of obstacle hexagons to represent a variable gadget by $2D$, where $D = (\max_{v \in V} \deg(v))$. The number of clause junctions is n. To give an upper bound on the number of flags in the auxiliary construction, we have to account for the flags in the transmitter gadgets, the extra hexagons found in junctions, and the flexible hexagons around the variable gadgets.

Recall that that the number of flags in a corridor are $t = 2N(m,n)^3 + 1$ where $N(m,n)$ is a polynomial. Recall that the drawing of $A(\Phi)$ have edges drawn in vertically and horizontally and can join at some "elbow". The distance can be measured in the ℓ_1 norm. Similarly in the honeycomb construction, the flexable hexagons zig-zig vertically and horizontally through out honeycomb. The number of corridors about an obstacle hexagon is 6. To give a generous upper bound on the number of flags in a transmitter gadget, is $6 \cdot t \cdot \ell_1(v_i, C_j)$, assuming each obstacle hexagon is of unit height.

The number of junctions in the auxiliary construction is the number of junctions to form all variable gadgets, transmitter gadgets, and clause gadgets. We know there are at most $2 \cdot D$ obstacle hexagons to form each variable gadget and 6 junctions for each obstacle hexagon. Therefore an upper bound for the number of flags around variable gadgets is $m \cdot 6 \cdot t \cdot 2 \cdot D$. The upper bound for the number of junctions in a transmitter gadget is $6\ell_1(v_i, C_j)$. Thus, the upper bound of all junctions in all transmitter gadgets is

$$6 \cdot \sum_{\{v_i, C_j\} \in E} \ell_1(v_i, C_j).$$

The upper bound on the total number of flags is

$$m \cdot 6 \cdot t \cdot 2 \cdot D + 6 \cdot \sum_{\{v_i, C_j\} \in E} \ell_1(v_i, C_j).$$

(2) Recall that a forest is a disjoint union of trees. By construction, each flag is hinged to exactly one obstacle hexagon. There are no hinges between obstacle hexagons. Consequently, each component of the hinge graph is a star, where the center corresponds to an obstacle hexagon and the leafs corresponds to the flexable hexagons attached to it.

(3) The final statement is to show an if and only if statement: it admits a realization such that the obstacle polygons remain fixed if and only if Φ is satisfiable.

Suppose Φ is satisfiable. Each variable has a boolean value and we can encode the corresponding auxilary construction accordingly. For each variable, we encode the boolean value by the state of the flags surrounding the variable gadget to R or L. Lemma 4 shows that the corridors and junctions around the variable gadget are realizable. Lemma 2 also show that for each transmitter gadget, every corridor and junction are also realizable. Lemma 3 shows that there is at least one hexagon in the clause junction and that the clause is realizable. Thus all parts of the auxilary construction realizable and thus we have a realization.

Suppose the construction admits a realization such that the obstacle polygons remain fixed. Each variable gadget's flags are configured to state L or R. The variable's corresponding state correspond to the variable's truth value, i.e. R for true and L for false. Using Lemma 4, the boolean state of the variable gadget is transmitted to all transmitter gadgets associated to it. Each clause is realizable and so for every clause, there

exists one true literal in the clause corresponding to a variable by Lemma 3. If every clause has some true literal, then the corresponding 3-CNF boolean formula is satisfiable. □

Modified Auxilary Construction. Recall that in Theorem 1 we want to show that it is strongly NP-hard to decide whether a polygonal linkage whose hinge graph is a tree can be realized with counter-clockwise orientation. We modify the auxiliary construction allowing all polygons to move freely, and by adding extra polygons and hinges so that the hinge graph becomes a *tree*, and the size of the construction remains polynomial. The auxiliary construction is based on a polynomial sized area of the hexagonal grid, using obstacle hexagons of side lengths $N(n,m)$, unit hexagons (of side length 1), and small hexagons of side length $\frac{1}{3}$. We modify it in five steps as follows:

1. Move the obstacle hexagons apart such that the width of each corridor increases from $\sqrt{3}$ to $\sqrt{3}+1/(100N)$.
2. Replace the unit segment in each clause gadget by a skinny rhombus of diameter $\sqrt{1+(100N)^{-2}}$ and width $1/(100N)$.
3. Enclose the regular hexagon region J containing all gadgets by a *frame* of 6 congruent regular hexagons, as shown in Fig. 3.14(a), hinged together in a path.
4. Connect the frame and the obstacles in J into a simply connected polygonal linkage: in each obstacle hexagon, the bottom side is adjacent to the frame or to a corridor. Introduce a hinge at the midpoint of one such side in each obstacle hexagon. If this side is adjacent to the frame, then attach the hinge to the frame. Otherwise, the hinge is attached to a new *connector* polygon: a skinny rhombus of diameter 1 and width $\frac{1}{100N}$. The far corner of each rhombus is hinged to the unit hexagon in the middle of the corridor at shown in Figure 3.14(b).
5. The construction so far constains rows and columns of obstacle hexagons. Every other column of obstacle hexagons is hinged to the bottom side of the perimiter of J.

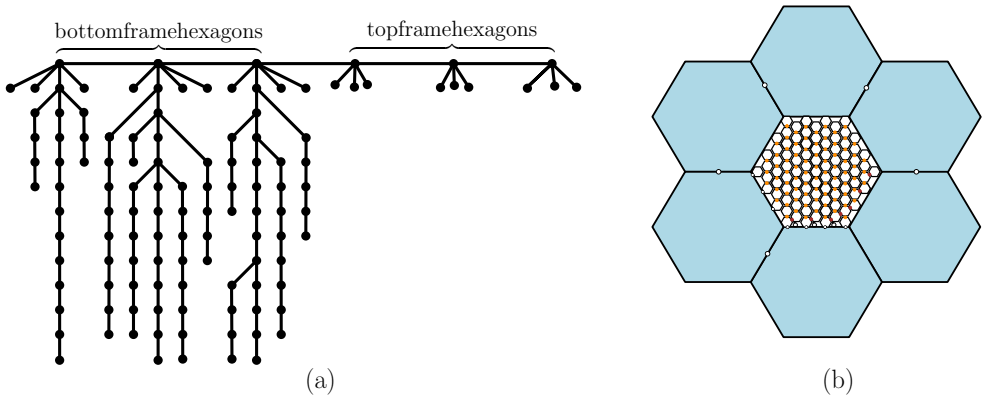


Figure 3.12: (a) illustrates a tree corresponding to the modified auxilary construction in (b). On the left half of the tree, we have the bottom most frame hexagons and the the hexagons in the interior of J as children of the the bottom frame hexagons. The top most frame hexagons only have the half sized hexgons attached to them. (b) is the corresponding modified auxilary construction.

These bottom most obstacle hexagons have a hinge point on its side. The columns that do not have an obstacle hexagon hinged to the perimeter of J has a half-sized hexgon hinged to the perimeter of J and a locked flag with no state to the first obstacle hexagon above it (See Figure 3.13).

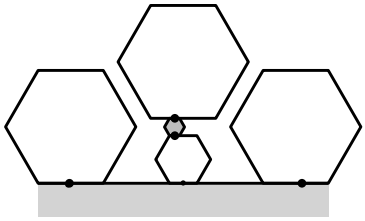


Figure 3.13: In this figure we illustrate the bottom of the perimeter of J with three obstacle hexagons, a half sized hexagon, and a locked flag whose hinge points lock the flag's state (becoming stateless).

We obtain a simply connected polygonal linkage. We now allow the obstacle hexagons to move freely, and call their original fixed position *canonical*. (3) We may assume without loss of generality that the frame is at its original position. It is enough to show that the obstacle hexagons are still confined to an $1/N$-neighborhood of their canonical position, then it follows that the polygonal linkage is realizable if and only if Φ is satisfiable.

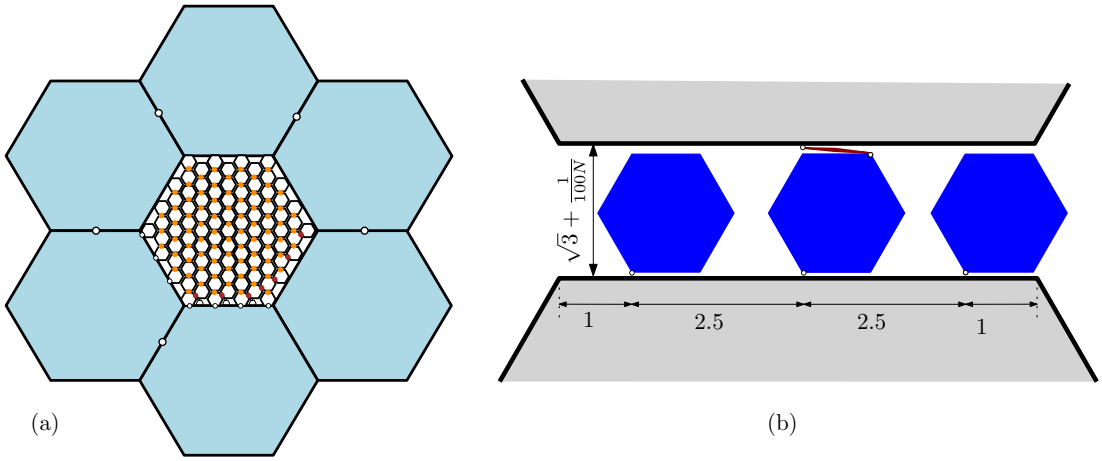


Figure 3.14: (a) A frame (built of 6 hinged regular hexagons) encloses a hexagonal tiling, and vertical paths connect all obstacle hexagons to the frame. (b) A corridor is widened to $\sqrt{3}+\frac{1}{N^2}$. A connection between two adjacent obstacle hexagons is established via a skinny rhombus.

The position of each hexagon can be defined by the isometry from its canonical position; an isometry is given by the triple (α,β,δ) where α is a counter clockwise rotation about the center of the hexagon and (β,δ) is a translation vector. Canonical position would have each obstacle hexagon's position as $(0,0,0)$.

Lemma 6. *Let P be a polygonal linkage obtained from the modified auxilary construction. In every realization of P, the obstacle polygons are close to canonical position.*

Lemma 6 serves as assurance that once a boolean formula of P3SAT is encoded into an arbitrary realization of the modified auxilary construction, the information of the boolean formula is preserved regardless of the positioning of the gadgets and components in the construction. This quality shows that the information is stable and preserved in an arbitrary realization of the modified auxilary construction.

Proof. We need to show that the modified auxilary construction could not deform in such a way that any information the construction encodes is lost or modified and the functionality of the gadgets within the construction behave as stated in the description. For example, in Figure 3.15, we have a column of obstacle hexagons veering off ℓ.

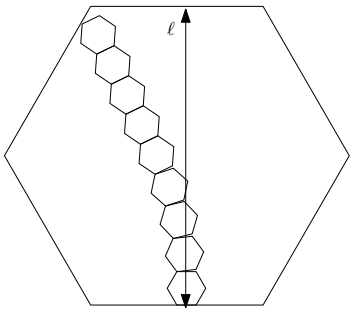


Figure 3.15: This figure depicts a column of obstacle hexagons rotated such that the obstacle hexagons veer of the vertical line ℓ.

This is an example of extreme angular rotation that should not occur over a vertical stack of hexagons. If a P3SAT boolean formula were encoded into such a realization, the information encoded could be lost by the extreme angular rotation of the obstacle hexagons.

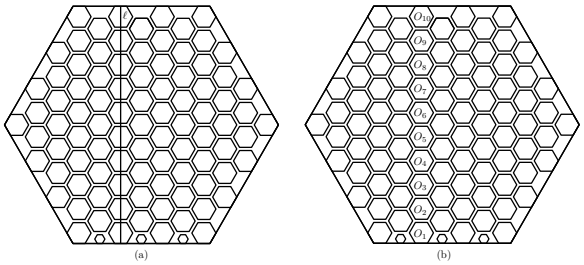


Figure 3.16: (a) depicts a column of obstacle hexagons O_1, ..., O_{10} along the vertical line ℓ; (b) identifies obstacle hexgons O_1, ..., O_{10} in (a).

Without loss of generality, we can identify a column of obstacle hexagons O_i along a vertical line ℓ (See Figure 3.16). In this proof, unless otherwise specified, we assume that the argument refers to a column that starts and ends with an obstacle hexagon. In total there will be $u+1$ number of obstacle hexagons and u corridors in a column. Note that:

$$\begin{aligned} u &= \tfrac{J_h(z)}{2} - \tfrac{1}{2} \\ &= \tfrac{1}{2}(6z+1-1) \\ &= 3z \\ &= 12s \end{aligned}$$

where J_h is defined in Equation 3.1.

The width of a skinny rhombus in canonical position is $\frac{1}{100N}$. The obstacle hexagon has height of $2N(n,m)\sqrt{3}$, and the flag is of height $\sqrt{3}$. The height $H(n,m)$ (and ℓ in Figure **??**(a)) can be expressed as a sum of the heights of the corridors and obstacle polygons:

$$(u+1)2N\sqrt{3} + u\left(\sqrt{3} + \frac{1}{100N}\right)$$

which reducest to:

$$\begin{aligned} (u+1)2N\sqrt{3} + u\left(\sqrt{3} + \frac{1}{100N}\right) &= (12s+1)2\frac{5t-1}{2}\sqrt{3} + 12s\left(\sqrt{3} + \frac{1}{100^{\frac{5t-1}{2}}}\right) \\ &= (12s+1)(5t-1)\sqrt{3} + 12s\left(\sqrt{3} + \frac{1}{250s^{\kappa}-50}\right) \end{aligned}$$

$$H(n,m) = (12s+1)(5t-1)\sqrt{3} + 12s\left(\sqrt{3} + \frac{1}{250s^{\kappa} - 50}\right) \tag{3.4}$$

The cross section of the corridor must have a minimum height of $\sqrt{3}$ everywhere. Otherwise will result in overlapping polygons.

Angular Rotation α. First we show that the angular rotation of the obstacle hexagons with respect to canonical position is small. We first look at the relative angular difference between two adjacent obstacle polygons

$$|\alpha_i - \alpha_{i+1}|.$$

Given an arbitrary instance of a modified auxilary construction, consider O_i, O_{i+1}, and the corridor between O_i and O_{i+1} (see Figure 3.17 for illustration).

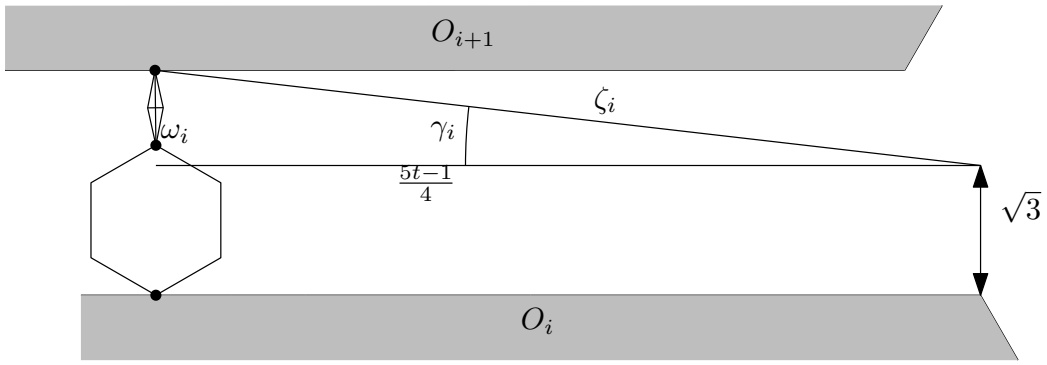


Figure 3.17: The obstacle hexagon here is in noncanonical position, and showing the side lengths adjacent to α_i.

In Figure 3.16(a) we see ℓ in the center of the column of obstacle hexagons. Our goal is to show that the column of hexagons cannot tilt in the manner shown in Figure 3.15 where the column veers greatly into the space occupied by other corridors and obstacle hexagons. The cross section of an arbitrary corridor must have a height of at least $\sqrt{3}$ everywhere. Otherwise, a flag would overlap with an obstacle hexagon; it would no longer remain a realization since the height of a flag is $\sqrt{3}$. In Figure 3.17, we illustrate an obstacle hexagon, its upper corridor with the central flag rotated counterclockwise $\frac{\pi}{6}$ radians that has a hinge to the skinny rhombus in a vertical position. The skinny rhombus has length $\sqrt{1+(100N)^{-2}}$. The rhombus is hinged at the midpoint of the upper side of the corridor. The length from a corridor's midpoint to one end of the corridor is $\frac{5t-1}{4}$. γ_i is the angle between ζ_i and the horizontal axis at the height of the flag ($i = 1,2,\ldots,u$).

The bound of γ_i is:

$$\begin{aligned}
\gamma_i &\leq \tan^{-1}\left(\frac{\left(2-\sqrt{3}\right)+\sqrt{1+\left(\frac{1}{100N}\right)^2}}{\frac{5t-1}{4}}\right) \\
&\leq \tan^{-1}\left(\frac{4\left(\frac{1}{3}+\sqrt{1+\left(\frac{1}{100N}\right)^2}\right)}{5t-1}\right) \\
&\leq \tan^{-1}\left(\frac{\frac{4}{3}+\sqrt{16+\left(\frac{4}{100N}\right)^2}}{5t-1}\right) \\
&\leq \tan^{-1}\left(\frac{\frac{4}{3}+\sqrt{16+\frac{1}{625\left(\frac{5t-1}{2}\right)^2}}}{5t-1}\right) \\
&\leq \tan^{-1}\left(\frac{\frac{4}{3}+\sqrt{16+9}}{5t-1}\right) \\
&\leq \tan^{-1}\left(\frac{\frac{4}{3}+5}{5t-1}\right) \\
&\leq \frac{19}{(15s^\kappa-3)} \\
&\leq \frac{19}{14s^\kappa} \\
&\leq \frac{3}{2s^\kappa}
\end{aligned} \tag{3.5}$$

Inequality 3.5 uses the first term Maclaurin series of $\tan^{-1}$ and holds for sufficiently large s. Thus the relative rotational difference between adjacent obstacle hexagons is:

$$|\alpha_i - \alpha_{i+1}| \leq \frac{3}{2s^\kappa} \tag{3.6}$$

The relative difference between α_i and α_{i+1} is small. The bottom most obstacle hexagon is hinged to the frame (see Figure 3.13 for illustration). This implies that $\alpha_1 = 0$ and

$$|\alpha_1 - \alpha_2| = |\alpha_2| \leq \frac{1}{16s^{3\kappa}}.$$

There are a total of u obstacle hexagons in a column with possibly up to $u-1$ nonzero obstcle hexagons rotations. We can derive 1) a bounded sum of rotational displacement over a column of obstacle hexagons:

$$\sum_{i=1}^{u-1} |\alpha_i - \alpha_{i+1}| \leq \frac{(12s-1)3}{2s^\kappa} \tag{3.7}$$

and 2) derive the maximum rotational displacement at the i^{th} obstacle hexagon:

$$\begin{aligned}
\alpha_i &\leq \frac{3}{2s^\kappa}\sum_{j=1}^{i} j \\
&\leq \frac{3}{2s^\kappa}\frac{i^2+i}{2} \\
&\leq \frac{3}{2s^\kappa}i^2 \\
&\leq \frac{3u^2}{2s^\kappa}
\end{aligned}$$

For any i, the bound for α_i:

$$\alpha_i \leq \frac{144s^2+12s}{32s^{3\kappa}} \tag{3.8}$$

Vertical Displacement δ When an obstacle hexagon is rotated by α_i, the height of the obstacle hexagon becomes $h \sec \alpha_i$ where h is the canonical height of the obstacle hexagon (see Figure 3.18 for reference). Figure 3.18 shows the geometry of a rotated obstacle hexagon.

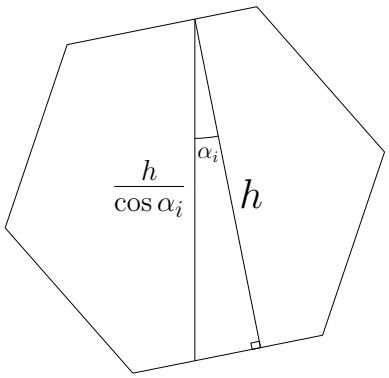


Figure 3.18: This figure shows a right triangle with angle α_i and sides of length h and $\frac{h}{\cos \alpha_i}$.

To show that the vertical displacement from canonical position is small, we first consider a column of obstacle hexagons in canonical position (see Figure 3.16 for illustration). For canonical position, the j^{th} obstacle has $\delta_j = 0$.

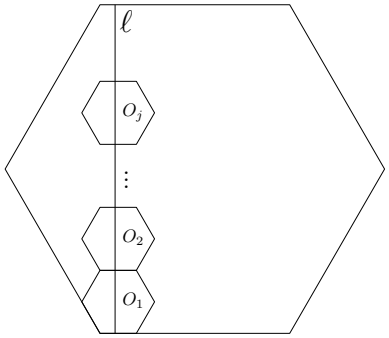


Figure 3.19: This illustration is of a column of obstacle hexagons in canonical position along a vertical line segment ℓ.

From Equation 3.4, we know the exact height of ℓ in terms of the heights of the corridors and obstacle hexagons in canonical position. Consider the first j terms for the height of the column of obstacle hexagons and corridors for an arbitrary construction with angular rotation and vertical displacement for *one* obstacle

hexagon $|\delta_v| > 0$, where $j = 2, \cdots, u+1$ and $1 < v \leq j$.

$$\begin{aligned}
\sum_{i=1}^{j}\left(2\sqrt{3}N\sec(\alpha_i)\right)+\delta_v+(j-1)\left(\frac{1}{100N}+\sqrt{3}\right) &\leq j\cdot 2N\sqrt{3}+(j-1)\cdot\left(\frac{1}{100N}+\sqrt{3}\right)\\
2\sqrt{3}N\sum_{i=1}^{j}\sec(\alpha_i)+\delta_v &\leq j\cdot 2\sqrt{3}N\\
\sum_{i=1}^{j}\sec(\alpha_i)+\delta_v &\leq j\\
\delta_v &\leq j-\sum_{i=1}^{j}\sec(\alpha_i)\\
\delta_v &\leq j-\left(j-\sum_{i=1}^{j}\frac{\alpha_i^2}{2}\right)\\
\delta_v &\leq \sum_{i=1}^{j}\frac{\alpha_i^2}{2}
\end{aligned}$$

Using Inequalities 3.7 and 3.8, we derive the following result:

$$\begin{aligned}
\Sigma_{i=1}^{j}\frac{\alpha_i^2}{2} &\leq \frac{1}{2}\Sigma_{i=1}^{j}\left(\frac{144s^2+12s}{32s^{3\kappa}}\right)^2\\
&\leq \frac{1}{2}\cdot\left(\frac{144s^2+12s}{32s^{3\kappa}}\right)^2\cdot j\\
&\leq \frac{1}{2}\cdot\left(\frac{144s^2+12s}{32s^{3\kappa}}\right)^2\cdot u\\
&\leq \frac{1}{2}\cdot\left(\frac{144s^2+12s}{32s^{3\kappa}}\right)^2\cdot 12s\\
&\leq \frac{6s\left(144s^2+12s\right)^2}{\left(32s^{3\kappa}\right)^2}
\end{aligned}$$

Thus we finally say that the bound for δ_v, where $1 < v \leq j \leq u$, is small:

$$\delta_v \leq \frac{6s\left(144s^2+12s\right)^2}{\left(32s^{3\kappa}\right)^2} \tag{3.9}$$

Horizontal Displacement β We show that the relative horizontal displacement between two vertically adjacent obstacle hexagons is small and polynomial size. We first identify where horizontal displacement can occur in the modified auxiliary construction.

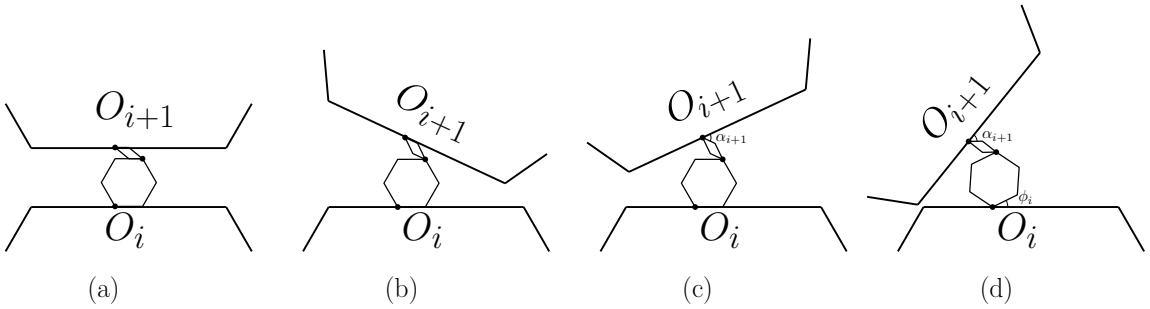
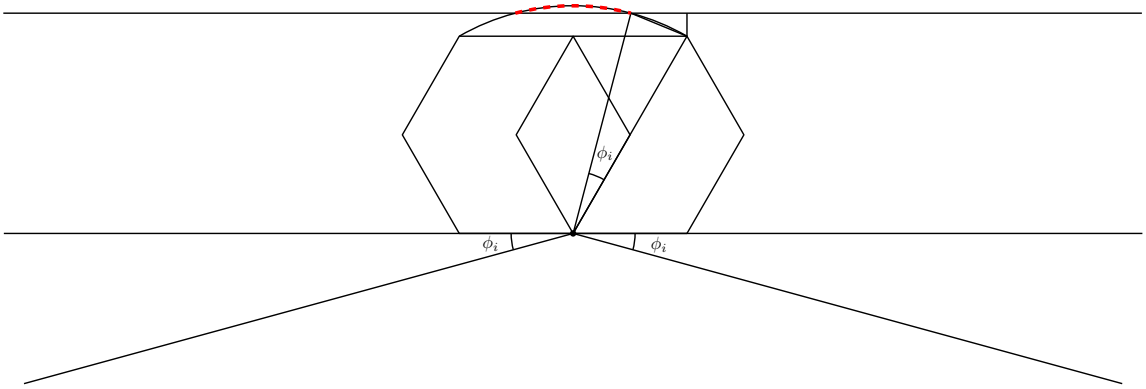


Figure 3.20: (a) A pair of vertically adjacent obstacle hexagons and their corresponding corridor in canonical position. (b) shows the same obstacle hexagons and corridors with the exception that the skinny rhombus is not in cannonical position. (c) is the same as (b) with the exception that the obstacle hexagon O_{i+1} has rotation displacement. (d) shows the rotation that is between the central flag and O_i.

Refer to the illustrations in Figure 3.20. If O_i is fixed, then O_{i+1} has three degrees of freedom. Together, the central flag and the skinny rhombus below O_{i+1} have three hinges upon which these elements can move and ultimately, move O_{i+1}. Figure 3.20(a) shows the cannonical position of all objects. Figure 3.20(b) illustrates some range of angular motion ω_i that a skinny rhombus can rotate on its hinge with a flag. Figure 3.20(c) shows a rotational displacement of O_{i+1} with the skinny rhombus' angular displacement. Figure 3.20(d) is the same as (c) with the exception of an addtional rotation ϕ_i that is between the central flag and O_i. Each of these rotations can create horizontal displacement. Let β_{α_i} be the horizontal displacement generated by the rotational displacement of O_i. Let β_{ω_i} be the horizontal displacement generated by the rotational displacement between the skinny rhombus and the central flag. Lastly, let β_{ϕ_i} be the horizontal displacement generated by the rotational displacement between the central flag and O_i.



Lemma

Figure 3.21

6 Conclusion. We have thus shown that the triple (α, β, δ) for any obstacle hexagon is small and bounded. Therefore the entire modified auxiliary construction in any realization is canonical or close to canonical where the encoded P3SAT information is stable. □

Lemma 7. *Lemma 3. For every instance Φ of P3SAT, the corresponding modified auxiliary construction has the following properties: (1) it has polynomial size; (2) the hinge graph of the modified auxilary construction is a tree; (3) it admits a realization such that the obstacle polygons remain fixed if and only if Φ is satisfiable.*

Proof. (1) We can bound the number of obstacle hexagons to represent a variable gadget by $2D$, where $D = (\max_{v \in V} \deg(v))$. The number of clause junctions is n. To give an upper bound on the number of flexible hexagons in the auxiliary construction, we have to account for the flexible hexagons in the transmitter gadgets, the extra hexagons found in junctions, and the flexible hexagons around the variable gadgets.

Recall that that the number of flexible hexagons in a corridor are $t = 2N(m,n)^3 + 1$ where $N(m,n)$ is a

polynomial. Recall that the drawing of $A(\Phi)$ have edges drawn in vertically and horizontally and can join at some "elbow". The distance can be measured in the ℓ_1 norm. Similarly in the honeycomb construction, the flexable hexagons zig-zig vertically and horizontally through out honeycomb. The number of corridors about an obstacle hexagon is 6. To give a generous upper bound on the number of flexible hexagons in a transmitter gadget, is $6 \cdot t \cdot \ell_1(v_i, C_j)$, assuming each obstacle hexagon is of unit height.

The number of junctions in the auxiliary construction is the number of junctions to form all variable gadgets, transmitter gadgets, and clause gadgets. We know there are at most $2 \cdot D$ obstacle hexagons to form each variable gadget and 6 junctions for each obstacle hexagon. Therefore an upper bound for the number of flexible hexagons around variable gadgets is $m \cdot 6 \cdot t \cdot 2 \cdot D$. The upper bound for the number of junctions in a transmitter gadget is $6\ell_1(v_i, C_j)$. Thus, the upper bound of all junctions in all transmitter gadgets is

$$6 \cdot \sum_{\{v_i, C_j\} \in E} \ell_1(v_i, C_j).$$

The upper bound on the total number of flexible hexagons is

$$m \cdot 6 \cdot t \cdot 2 \cdot D + 6 \cdot \sum_{\{v_i, C_j\} \in E} \ell_1(v_i, C_j).$$

For each corridor, there is one skinny rhombus attached to one flexible hexagon in the corridor. If the number of corridors is bounded polynomially, then the number of skinny rhombi is bounded by the same bound of the corridor.

(2) Recall that in the original auxilary construnction is a forest. each obstacle hexagon with hinged flexible hexagons (and small hexagons) is disjoint from the remainder of the the construction. The skinny rhombi in the modified auxilary construction connect the disjointed trees to form one tree.

(3) The final statement is to show an if and only if statement: the modified auxiliary construction admits a realization such that the obstacle polygons remain fixed if and only if Φ is satisfiable.

Suppose Φ is satisfiable. Each variable has a boolean value and we can encode a corresponding auxilary construction and then modify it as a modified auxiliary construction. For each variable, we encode the boolean value by the state of the flags surrounding the variable gadget to *R* or *L*. Lemma 4 shows that the corridors and junctions around the variable gadget are realizable. Lemma 2 also show that for each transmitter gadget, every corridor and junction are also realizable. Lemma 3 shows that there is at least one hexagon in the clause junction and that the clause is realizable. Thus all parts of the auxilary construction realizable. Transforming it insto a modified auxilary construction and by Lemma 6, the modified auxiliary construction is realizable.

Suppose the construction admits a realization such that the obstacle polygons remain fixed. Each variable gadget's flags are configured to state *L* or *R*. The variable's corresponding state correspond to the variable's truth value, i.e. *R* for true and *L* for false. Using Lemma 4, the boolean state of the variable gadget is transmitted to all transmitter gadgets associated to it. Each clause is realizable and so for every clause, there exists one true literal in the clause corresponding to a variable by Lemma 3. If every clause has some true literal, then the corresponding 3-CNF boolean formula is satisfiable. □

At the beginning of this chapter, we stated Theorem 1: It is strongly NP-hard to decide whether a polygonal linkage whose hinge graph is a tree can be realized with counter-clockwise orientation. We know the P3SAT is NP-hard [16] and by Lemma 7, we can conclude that deciding whether modified auxiliary construction (a tree) of a given P3SAT Boolean formula is NP-Hard.

Chapter 4

Realizability Problems for Weighted Trees

In this chapter our goal is to prove Theorem 4 which states: "It is NP-Hard to decide whether a given tree with positive vertex weights is the contact graph of a disk arrangements with specified radii." This chapter's approach to proving Theorem 4 introduces an ordered weighted tree T and perturbed ordered weight tree T_ε, the Hausdorff distance, and then prove the following lemma:

Lemma 8. *for ever $\varepsilon > 0$, there exists an ordered weighted tree T_ε such that every realization of T_ε as an ordered disk contact graph where the radii of the disks equal the vertex weights.*

Using Lemma 8, we prove Theorem 4 by extending the modified auxiliary construction in Chapter 3.

We first cover the preliminary concepts of Hausdorff distance and the ordered weighted tree families of T and T_ε. We then continue with the proof of Lemma 8 and Theorem 4.

4.1 Hausdorff Distance

Let A and B be sets in the plane. The *directed Hausdorff distance* is:

$$d(A,B) = \sup_{a\in A} \inf_{b\in B} ||a-b|| \tag{4.1}$$

$d(A,B)$ finds the furthest point $a \in A$ from any point in B. *Hausdorff distance* is

$$D(A,B) = \max\{d(A,B), d(B,A)\} \tag{4.2}$$

In Figure 4.1, we have two sets X and Y and illustrate $d(X,Y)$ and $d(Y,X)$. From this, it is possible to calculate the Hausdorff distance between X and Y.

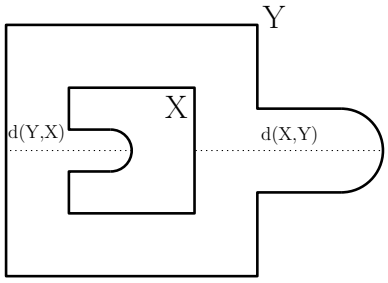


Figure 4.1: An illustrative example of $d(X,Y)$ and $d(Y,X)$ where X is the inner curve, and Y is the outer curve.

ε-approximation The weighted graph, G, is an *ε-approximation* of a polygon P if the Hausdorff distance between every realization of G as a contact graph of disks and a congruent copy of P is at most epsilon. A weighted graph G is said to be a *$O(f(x))$-approximation* of a polygon P if there is a positive constant M such that for all sufficiently large values of x the Hausdorff distance between every realization such realization of G as a contact graph of disks and a congruent copy of P is at $M \cdot |f(x)|$. A weighted graph G is said to be a *stable* if it has the property that for every two such realizations of G, the distance between the centers of the corresponding disks is at most ε after a suitable rigid transformation.

An example of an ε-approximation .....

Problem 9 (Appoximating Polygonal Shapes with Contact Graphs). For every $\varepsilon > 0$ and polygon P, there exists a contact graph $G = (V,E)$ such that the Hausdorff distance $d(P,G) < \varepsilon$

4.2 Weight Trees T_k

In this section we describe a particular family of unit weight trees and corresponding contact graphs disk arrangements called *snowflakes*. Note that we regard snowflakes with unit weight as a weight of $\frac{1}{2}$. For $i \in \mathbb{N}$, the construction of the snowflake tree, T_i, is as follows:

- Let v_0 be a dvertex that has six paths attached to it: $p_1, p_2, \ldots, p_6$. Each path has i vertices.
- For every other path p_1, p_3, and p_5:
  - Each vertex on that path has two paths attached, one path on each side of p_k.
  - The number of vertices that lie on a path attached to the j^{th} vertex of p_k is $i - j$.

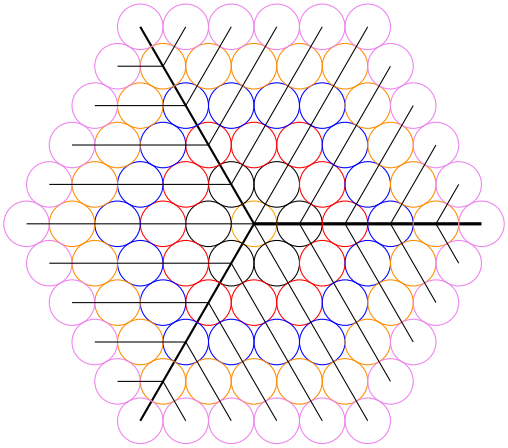


Figure 4.2: The same contact graph as in figure 4.3 overlayed with the a perfectly weighted snowflake tree.

A *perfectly weighted snowflake tree* is a snowflake tree with all vertices having weight $\frac{1}{2}$. A *perturbed snowflake tree* is a snowflake tree with all vertices having weight of 1 with the exception of v_0; in a perturbed snowflake tree, v_0 will have a weight of $\frac{1}{2} + \gamma$. For our analysis, all realizations of any snowflake, perfect or perturbed, shall have v_0 fixed at origin.

Perfectly Weighted Snowflake Tree. Consider the graph of the triangular lattice with unit distant edges:

$$\begin{aligned} V &= \left\{ a \cdot (1,0) + b \cdot \left(\frac{1}{2}, \frac{\sqrt{3}}{2} \right) : a,b \in \mathbb{Z} \right\} \\ E &= \{ \{u,v\} : ||u-v|| = 1 \text{ and } u,v \in V \} \end{aligned}$$

The following graph, $G = (V,E)$ is said to be the *unit distance graph* of the triangular lattice. We can show that no two distinct edges of this graph are non-crossing. First suppose that there were two distinct edges that crossed, $\{u_1,v_1\}$ and $\{u_2,v_2\}$. With respect to u_1, there are 6 possible edges corresponding to it, with each edge $\frac{\pi}{3}$ radians away from the next. Neither edge crosses another; and so we have a contradiction that there are no edge crossings with $\{u_1,v_1\}$.

The perfectly weighted snowflake tree that is a subgraph over the *unit distance graph*, $G = (V,E)$, of the triangular lattice. To show this, for any S_i, fix $v_0 = 0 \cdot \cdot (1,0) + 0 \cdot \left(\frac{1}{2}, \frac{\sqrt{3}}{2} \right) = (0,0) \in V$ at origin. Next

consider the six paths attached from origin. Fix each consecutive path $\frac{\pi}{3}$ radians away from the next such that the following points like on the corresponding paths: $(1,0) \in p_1, \left(\frac{1}{2}, \frac{\sqrt{2}}{3}\right) \in p_2, \left(-\frac{1}{2}\mathsf{p}_4, \frac{\sqrt{3}}{2}\right) \in p_3, (-1,0) \in p4, \left(-\frac{1}{2}, -\frac{\sqrt{3}}{2}\right) \in p_5, \left(\frac{1}{2}, -\frac{\sqrt{3}}{2}\right) \in p_6$. For S_i, there are i vertices on each path.

We define the six paths from origin as follows:

$$\begin{aligned}
p_1 &= \left\{ a \cdot (1,0) = \vec{v} \middle| a \in \mathbb{R}^+ \right\} \\
p_2 &= \left\{ a \cdot \left(\frac{1}{2}, \frac{\sqrt{3}}{2}\right) = \vec{v} \middle| a \in \mathbb{R}^+ \right\} \\
p_3 &= \left\{ -a \cdot (1,0) + a \cdot \left(\frac{1}{2}, \frac{\sqrt{3}}{2}\right) = a\left(-\frac{1}{2}, \frac{\sqrt{3}}{2}\right) = \vec{v} \middle| a \in \mathbb{R}^+ \right\} \\
p_4 &= \left\{ a \cdot (-1,0) = \vec{v} \middle| a \in \mathbb{R}^+ \right\} \\
p_5 &= \left\{ a \cdot \left(-\frac{1}{2}, -\frac{\sqrt{3}}{2}\right) = \vec{v} \middle| a \in \mathbb{R}^+ \right\} \\
p_6 &= \left\{ a \cdot (1,0) - a \cdot \left(\frac{1}{2}, \frac{\sqrt{3}}{2}\right) = a \cdot \left(\frac{1}{2}, -\frac{\sqrt{3}}{2}\right) \middle| a \in \mathbb{R}^+ \right\}
\end{aligned}$$

For S_i there exists i vertices on each path. We shall denote the i^{th} vertex on the j^{th} path as $v_{j,i}$. For each path defined above, the paths are defined as a set of vectors, $\vec{v} = a \cdot \vec{p}$ for some $a \in \mathbb{R}^+$ and $\vec{p} \in \mathbb{R}^2$. By setting $a = 1,2,\ldots,i$, we obtain points that are contained in V. For $j = 1,3,5$ and $l = 1,...,i$, there exists two paths attached to each vertex $v_{j,l}$. We borrow the term *petiole* from botany to describe the two paths attached to $v_{j,l}$. In botany, the stalk that attaches to a stem of a plant is called a petiole; petioles usually have leaves attached to their ends. For S_i, each petiole attached to the k^{th} vertex of p_j, there are $i-k$ vertices. We will need to show that each of the $i-k$ vertices on each corresponding path are also in V.

The triangular lattice is symmetric under rotation about v_0 by $\frac{\pi}{3}$ radians. For each vertex $v_{1,l}$ for $l = 1,2,\ldots,i-k$, we place two petioles from it; the first petiole $\frac{\pi}{3}$ above p_1 at $v_{1,l}$ and $\frac{-\pi}{3}$ below p_1 at $v_{1,l}$ and call these petioles $p^+_{1,l}$ and $p^-_{1,l}$ respectively. With respect to $v_{1,l}$, one unit along $p^+_{1,l}$ is a point on the triangular lattice and similarly so on $p^-_{1,l}$. Continuing the walk along these paths, unit distance-by-unit distance, we obtain the next point corresponding point on the the triangular lattice up to $i-k$ distance away from $v_{1,l}$. This shows that each of the $i-k$ vertices on $p^-_{1,l}$ and $p^+_{1,l}$ are in V. By rotating all of the paths along p_1 by $\frac{2\pi}{3}$ and $\frac{4\pi}{3}$, we obtain the the paths along p_3 and p_5 respectively, completing the construction.

In Figure 4.3, we have a set of unit radius disks arranged in a manner that outlines regular, concentric hexagons.

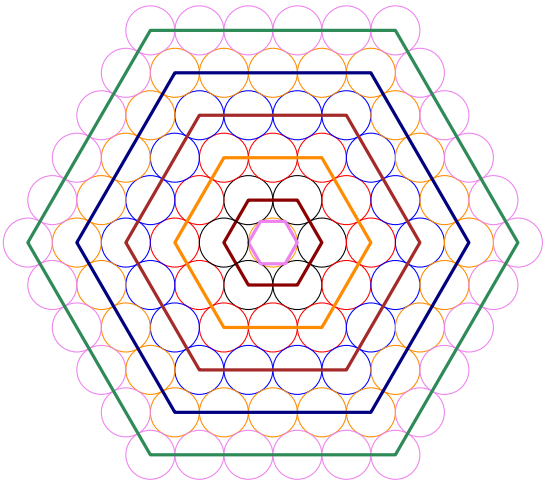


Figure 4.3: A contact graph that resembles the shape of concentric hexagons.

4.3 Proofs of Lemmas ?? through ??

We now prove the Lemmas **??** through **??**.

4.3.1 Proof of Lemma ??

Proof. Recall that we are to show that for any realized perturbed snowflake S_i, the gaps created in subset $S_1 \subset S_i$ are small.

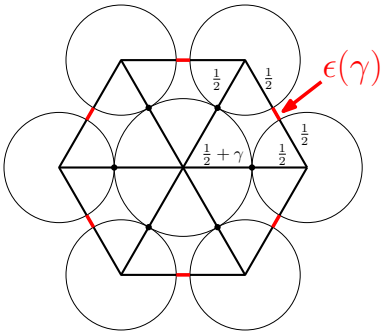


Figure 4.4: A canonical disk arrangement from a perturbed snowflake with 6 unit disks around a central disk with radius $\frac{1}{2}+\gamma$.

One way to do this is to demonstration that the sum of gaps for any realization of a contact graph of a perturbed snowflake S_1 is small. Denote the vertices around v_0 as v_1 through v_6 in a clockwise pattern about v_0. Without loss of generality, given a realization denote $\varepsilon_{k,k+1}(\gamma) \geq 0$ as the gap created between adjacent disks corresponding to v_1 thourgh v_6.

Consider the realization where $\varepsilon_{k,k+1}(\gamma) = 0$ with the exception of $\varepsilon_{1,6} > 0$. That is, ever consecutive pair of disks about the central disk is in contact with each other with the exception of D_1 and D_6. The realization provides 5 congruent triangles between the centers of D_0 and (D_1,D_2), (D_2,D_3), (D_3,D_4), (D_4,D_5), and (D_5,D_6). Given perturbation $\gamma > 0$, the side lengths between (D_0,D_i) are $1+\gamma$ and the side length of (D_i,D_{i+1}) is 1. Using law of cosine the angle formed between (D_0,D_i) and (D_0,D_{i+1}) is

$$2\tan^{-1}\frac{1}{2(1+\gamma)}.$$

The angle between (D_6, D_1) is

$$y = 2\pi - 5 \cdot \left(2\tan^{-1} \frac{1}{2(1+\gamma)} \right).$$

The side length of (D_6, D_1) is

$$\sqrt{-2(\gamma+1)^2 \cos\left(-10 \arctan\left(\frac{1}{2(\gamma+1)}\right)\right) + 2(\gamma+1)^2}.$$

Note that as $\gamma \to 0$, the side length of (D_6, D_1) is approximately 1 + .466861, where $\varepsilon(\gamma) \approx .466861$ as $\gamma \to 0$. This establishes an upperbound on the maximal displacement about S_1 with respect to the side lengths between the centers of disks about D_0.

The lower bound is established using the configuration found in Figure 4.4. he realization provides 6 congruent triangles between the centers of D_0 and each disk about D_0. Without loss of generality, to find the side length of between neighboring disks about D_0, we find need to $\varepsilon(\gamma)$. The angle between (D_0, D_i) and (D_0, D_{i+1}) is $\frac{\pi}{3}$; using the law of cosine, we can determine the side length of (D_i, D_{i+1}) is

$$\sqrt{1 + 2\gamma + \gamma^2}.$$

Thus the perturbation about S_1 in and confiugration is bounded and small. □

4.3.2 Proof of Lemma ??

Proof. Recall that we are to show that for any realized perturbed snowflake S_i, the angular value of α_k and β_k are small. In canoncical position, the angle between $p^+_{k,j}$ and $p^-_{k,j}$ is $\frac{\pi}{3}$. In a non-canonical position, we define the change in angle to be $f(\varepsilon)$. About a We prove this with induction.

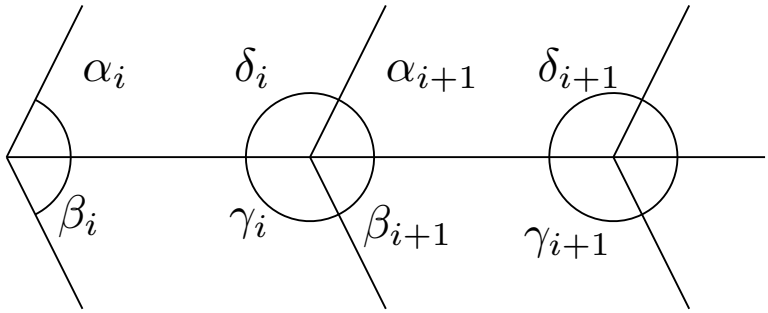


Figure 4.5

$$\begin{aligned} \alpha_i + \beta_i &\leq 120 + f(\varepsilon) \\ 2\pi &\leq \gamma_i + \delta_i + \tfrac{2\pi}{3} + f(\varepsilon) \\ 2\pi - (\gamma_i + \delta_i) &\leq \tfrac{2\pi}{3} + f(\varepsilon) \\ \alpha_{i+1} + \beta_{i+1} &\leq \tfrac{2\pi}{3} + f(\varepsilon) \end{aligned}$$

□

4.3.3 Proof of Lemma ??

Proof. Recall that we are to show that for any realized perturbed snowflake S_i, the distance between disks $D_{k,j}$, $D_{k+1,j}$, $D_{k+1,j+1}$, and $D_{k,j+1}$ are relatively small with respect to the relative distance in a perfect

snowflake where $k = 1, \ldots, 6$ and $j = 2, \ldots, i$.

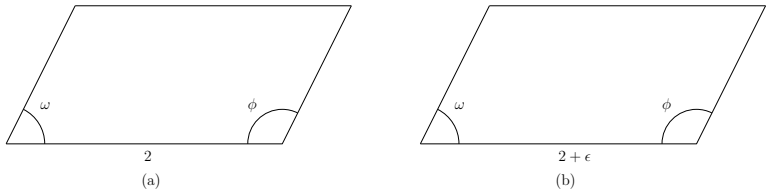


Figure 4.6

□

Bibliography

[1] Timothy G Abbott, Zachary Abel, David Charlton, Erik D Demaine, Martin L Demaine, and Scott Duke Kominers. Hinged dissections exist. *Discrete & Computational Geometry*, 47(1):150–186, 2012.

[2] Martin Aigner and Günter M Ziegler. Hilbert's third problem: decomposing polyhedra. *Proofs from the book*, pages 53–61, 2010.

[3] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, 1999.

[4] Sandeep N Bhatt and Stavros S Cosmadakis. The complexity of minimizing wire lengths in vlsi layouts. *Information Processing Letters*, 25(4):263–267, 1987.

[5] H. Breu and D. G. Kirkpatrick. Unit disk graph recognition is NP-hard. *Comput. Geom*, 9:3–24, 1998.

[6] R. Connelly, E. D. Demaine, M. L. Demaine, S. P. Fekete, S. Langerman, J. S. B. Mitchell, A. Ribó, and G. Rote. Locked and unlocked chains of planar shapes. *Discrete Comput. Geom*, 44:439–462, 2010.

[7] R. Connelly, E. D. Demaine, and G. Rote. Straightening polygonal arcs and convexifying polygonal cycles. *Discrete Comput. Geom*, 30:205–239, 2003.

[8] P. Eades and N. C. Wormald. Fixed edge-length graph drawing is NP-hard. *Discrete Applied Mathematics*, 28:111–134, 1990.

[9] Michael R Garey and David S Johnson. *Computers and Intractibility*. WH Freeman and company New York, 1979.

[10] J Jonsson and B Kaliski. Public-key cryptography standards (pkcs)# 1: Rsa cryptography specifications, version 2.1. internet request for comments 3447 (rfc 3447), 2003.

[11] Richard M Karp. *Reducibility among combinatorial problems*. Springer, 1972.

[12] Boris Klemz. Weighted disk contact graph. 2014.

[13] Paul Koebe. *Kontaktprobleme der konformen Abbildung*. Hirzel, 1936.

[14] Casimir Kuratowski. Sur le probleme des courbes gauches en topologie. *Fundamenta mathematicae*, 1(15):271–283, 1930.

[15] Arjen K Lenstra, Hendrik W Lenstra Jr, Mark S Manasse, and John M Pollard. *The number field sieve*. Springer, 1993.

[16] David Lichtenstein. Planar formulae and their uses. *SIAM journal on computing*, 11(2):329–343, 1982.

[17] Alfred J Menezes, Paul C Van Oorschot, and Scott A Vanstone. *Handbook of applied cryptography*. CRC press, 1996.

[18] Ronald L Rivest, Adi Shamir, and Len Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[19] S.S. Skiena. *The Algorithm Design Manual*. Springer, 2009.

[20] James A Storer. On minimal-node-cost planar embeddings. *Networks*, 14(2):181–212, 1984.

[21] I. Streinu. Pseudo-triangulations, rigidity and motion planning. *Discrete Comput. Geom*, 34:587–635, 2005.

[22] R Tamassia and IG Tollis. Efficient embedding of planar graphs in linear time. In *Proc. IEEE Int. Symp. on Circuits and Systems*, pages 495–498, 1987.

[23] E. C. Zeeman. On hilbert's third problem. *The Mathematical Gazette*, 86(506):241–247, 2002.