CALIFORNIA STATE UNIVERSITY, NORTHRIDGE

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

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

by

Clinton Bowen

August 2014

Table of Contents

ABSTRACT

PROTEIN FOLDING: PLANAR CONFIGURATION SPACES OF DISC ARRANGEMENTS AND

HINGED POLYGONS: *PROTEIN FOLDING IN FLATLAND*

By

Clinton Bowen

Master of Science in Computer Science

abstract goes here

Complex structures in nature are often composed of elementary pieces that obey simple local composition rules. Molecular biology, nanomanufacturing, and self-assembly are prime examples. Mathematical models for this phenomenon typically rely on rigidity theory and formal languages. In this paper, we study the realizability of complex structures that are given with a local specification. We consider two models in Euclidean plane.

1. A **polygonal linkage** is a set $\mathscr{P}$ of convex polygons, and a set $\mathscr{H}$ of hinges, where each hinge $h \in \mathscr{H}$ corresponds to two points on the boundary of two distinct polygons. A *realization* of a polygonal linkage is an interior-disjoint placement of congruent copies of the polygons in $\mathscr{P}$ such that the points corresponding to each hinge are identified (Fig. 1, left).
2. A **disk arrangement** is a set $\mathscr{D}$ of pairwise interior-disjoint disks in the plane. The contact graph of a disk arrangement $\mathscr{D}$ is a graph $G = (\mathscr{D}, E)$ where two vertices are adjacent if the corresponding disks intersect (kiss). A *realization* of a vertex-weighted graph G as a contact graph of disks is a disk arrangement whose contact graph is G and the radius of each disk is the corresponding vertex weight.

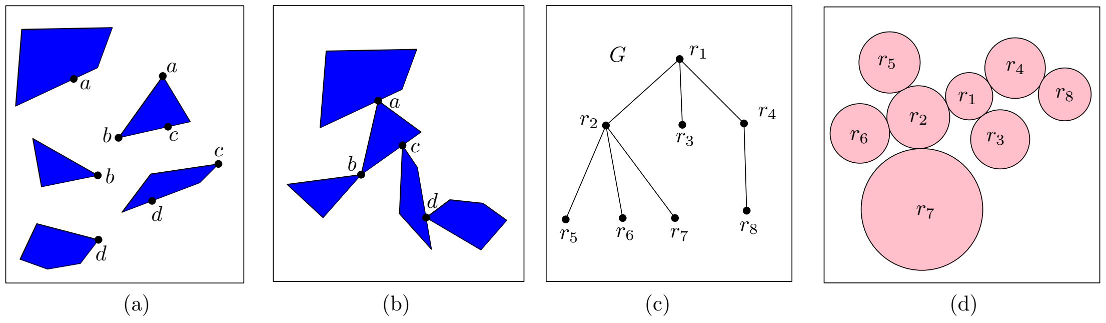


Figure 1: (a) A set of convex polygons and hinges. (b) A realization of the polygonal linkage from (a). (c) A graph G with vertex weights $r_1, \ldots, r_8$. (d) A disk arrangement that realizes the weighted graph G as a contact graph with radii equal to the corresponding weights.

Each model has two variants, depending on whether *reflection* is allowed for the realization of each piece independently. For polygonal linkages, an *oriented realization* requires translated and rotated copies of the polygons in $\mathscr{P}$ (i.e., reflection is not allowed). An *ordered contact graph* for a disk arrangement is a *plane graph G*, where the circular order of the neighbors of each vertex is specified, and an *oriented realization* is disk arrangement with the given ordered contact graph.

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 [**?**, **?**]. 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 [**?**], 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 [**?**, **?**] and the coin graphs in [**?**] have many cycles.

In this paper, 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 0.0.1. *It is NP-complete to decide whether a polygonal linkage whose hinge graph is a tree can be realized (both with and without orientation).*

Theorem 0.0.2. *It is NP-complete to decide whether a given tree (resp., plane tree) with positive vertex weights is the contact graph (resp., ordered 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 0.0.1 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 **??**).

Related Previous Work. Polygonal linkages (or body-and-joint frameworks) are a generalization of classical linkages (bar-and-joint frameworks) in rigidity theory. A linkage is a graph $G = (V, E)$ with given edge lengths. A realization of a linkage is a (crossing-free) straight-line embedding of G in the plane. Bhatt and Cosmadakis [**?**] proved that the realizability of linkages is NP-hard. Their "logic engine" method [**?**, **?**, **?**, **?**], has become a powerful tool in graph drawing. The logic engine is a graph composed of rigid 2-connected components, connected by cut vertices (hinges). The two possible realizations of each 2-connected component (that differ by a single reflection) represent the truth assignment of a binary variable. This method does not applicable to the *oriented* version of the realizability, where the circular order of the neighbors of each vertex is part of the input. Cabello et al. [**?**, **?**] proved that the realizability of 3-connected linkages (where the orientation is unique by Steinitz's theorem) is NP-hard, but efficiently decidable for near-triangulations [**?**, **?**].

Note that every *tree* linkage can be realized in $\mathbb{R}^2$ (with almost collinear edges). According to the celebrated *Carpenter's Rule Theorem* [**?**, **?**], every realization of a path (or a cycle) linkage can be continuously moved (without self-intersection) to any other realization. In other words, the realization space of such a linkage is always connected. However, there are trees of maximum degree 3 with at few as 8 edges whose realization space is disconnected [**?**]; and deciding whether the realization space of a tree linkage is connected is PSPACE-complete [**?**]. (Earlier, Reif [**?**] showed that it is PSPACE-complete to decide whether a polygonal linkage can be moved from one realization to another among polygonal obstacles in $\mathbb{R}^3$.) Cheong et al. [**?**] considers the "inverse" problems of introducing the minimum number of point obstacles to reduce the configuration space of a polygonal linkage to a unique realization.

Connelly et al. [**?**] showed that the Carpenter's Rule Theorem generalizes to certain polygonal linkages, which are obtained by replacing the edges of a path linkage with special polygons called (*slender adornments*). Our Theorem 0.0.1 indicates that if we are allowed to replace the edges of a path linkage with arbitrary convex polygons, then deciding whether the realization space is empty or not is already NP-hard.

Recognition problems for intersection graphs of various geometric object have a rich history [**?**]. Breu and Kirkpatrick [**?**] proved that it is NP-hard to decide whether a graph G is the contact graph of unit disks in the plane (a.k.a. recognizing *coin graphs* is NP-hard). A simpler proof was later provided via the logic engine [**?**]. It is also NP-hard to recognize the contact graphs of pseudo-disks [**?**] and disks of bounded radii [**?**] in the plane, and unit disks in higher dimensions [**?**, **?**]. All these hardness reductions produce graphs of high genus, and do not apply to trees. Note that the contact graphs of disks (of arbitrary radii) are exactly the planar graph (by Koebe's circle packing theorem), and planarity testing is polynomial. Consequently, every tree is the contact graph of disks of *some* radii in the plane.

0.1 Background

In this section, we cover the background subjects needed to formally pose the problem and solutions in this thesis. We start with two types of combinatorial structures, linkages and polygonal linkages. We then discuss the configuration spaces of linkages and polygonal linkages. We then look into an alternate representation of linkages, disk arrangements and state the disk arrangement theorem but do not show its proof. We then look at satisfiability problems, a logic engine that encodes a type of satisfiability problems. Finally, we cover the basic definitions of algorithm complexity for **P** and **NP**

0.1.1 Linkages

There are two parts to a linkage, the graph and the edge length mapping. A *graph* is an ordered pair $G = (V, E)$ comprising of a set V of vertices or together with a set E of edges or lines. For every edge $e \in E$, there is a distinct pair of vertices in V that it represents, $(u, v) \in E$. While graphs can have vertices that may not correspond to any edges, we rule out this possibility for linkages. The edge length assignement mapping of a linkage is $l : E \mapsto \mathbb{R}^+$. A linkage is still an abstract combinatorial structure until an *embedding* on the graph of the linkage is posed, i.e. $\Pi : V \mapsto \mathbb{R}^2$. Π has the *proper embedding* property, if for every edge $(u, v) \in E$ such that $l((u, v)) = |\Pi(u) - \Pi(v)|$ is true. The *realization* of the the linkage is said to be the range Π, i.e. $\Pi(V)$. Without loss of generality, for this paper, we focus on linkages that have simple planar graph properties, i.e.:

(i) does not have edges that cross,

(ii) does not have loops, i.e. $(v, v) \in E$, or

(iii) does not have multiple edges between any pair of vertices.

We may visit special cases in which we look at planar graphs that satisfy the last two conditions but not the first, e.g.:

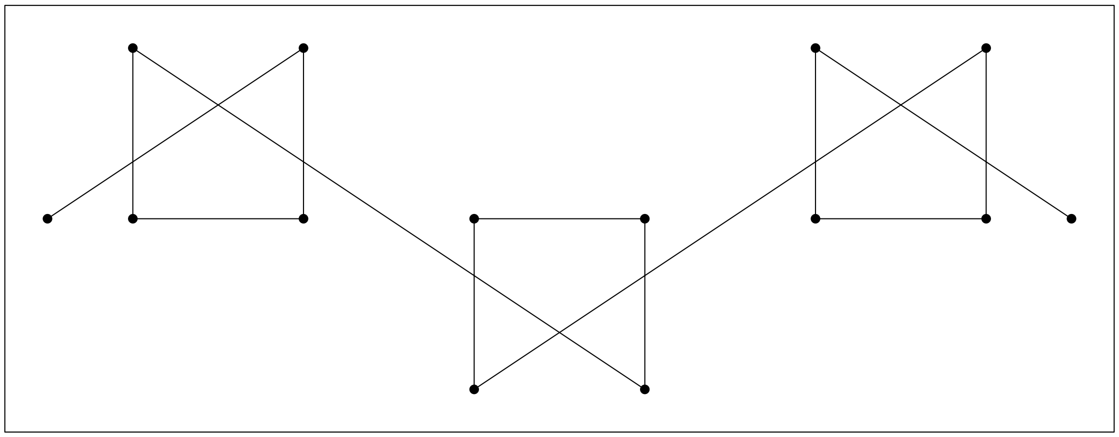


Figure 2: A linkage where edges cross however it does not contain loops or multiple edges between vertices.

0.1.2 Polygonal Linkages

Polygonal linkages have some differences to linkages. Polygonal linkages have a combinatorial structure that is graph-like. A polyonal linkage's combinatorial structure is an ordered pair $(\mathscr{H}, \mathscr{P})$. Instead of a set of vertices, there is a set of hinges, $\mathscr{H}$; instead of a set of edges, there is a set of polygons, $\mathscr{P}$. Formally, a *hinge* $h \in \mathscr{H}$ corresponds to two points on the boundary of two distinct polygons in $\mathscr{P}$. Since polygons do not exhibit a length property, a polygonal linkage does not have an edge length mapping.

Polygon linkages are similar to linkages. They are an ordered pair of sets, with the exception that the set of vertices become a set of hinges, $\mathscr{H}$, and the set of edges become a set of polygons, $\mathscr{P}$. Formally, a *polygonal linkage* is an ordered pair, $G = (\mathscr{H}, \mathscr{P})$, comprises of a set of polygons, $\mathscr{P}$, and a set of hinge points $\mathscr{H}$ where each hinge $h \in \mathscr{H}$ corresponds to two points on the boundary of two distinct polygons in $\mathscr{P}$. Mapping the linkage G into the plane is said to be the *embedding*, i.e. $L : \mathscr{H} \mapsto \mathbb{R}^2$. We consider a *realization* of a polygonal linkage is range of L, i.e. $l(\mathscr{H})$.

In figure (3), we illustrate that two hinge points can reside on the same point in the plane. Figure (3) and figure (2) are examples of special cases that we may run into, but do not want to focus heavily on. They

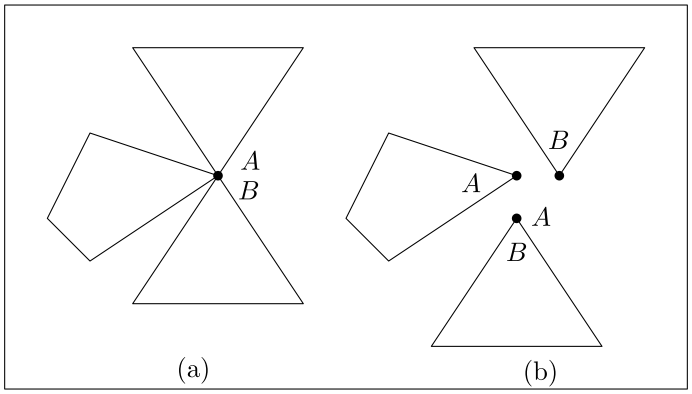


Figure 3: (a) A polygonal linkage with a non-convex polygon and two hinge points corresponding to three polygons. Note that hinge points correspond to two distinct polygons.(b) Illustrating that two hinge points can correspond to the same boundary point of a polygon.

are presented to the reader to facilitate understanding of the definitions of polygonal linkages and linkages respectively. Without loss of generality, for this paper, we focus on polygonal linkages that are equivalent to simple planar graphs. For the remainder of this thesis, we'll focus on the polygonal linkages with the

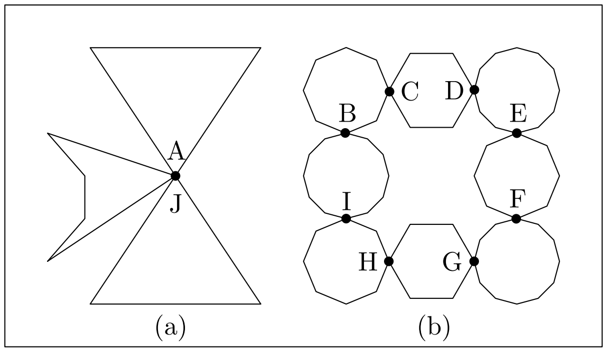


Figure 4: (a) A polygonal linkage with a non-convex polygon and a hinge point corresponding to three polygons. (b) A polygonal linkage with 8 regular polygons.

following restrictions:

1. For each hinge point $h \in \mathscr{H}$, there exists some subset $P \subset \mathscr{P}$ with exactly cardinality of 2, and
2. every polygon in $\mathscr{P}$ is convex.

Formally, we define a *polygonal linkage* as an ordered pair $L = (\mathscr{H}, \mathscr{P})$ comprising of a set of hinges, $\mathscr{H}$, where each hinge $h \in \mathscr{H}$ corresponds to two points on the boundary of two distinct polygons. A *realization* of a polygonal linkage is an interior-disjoint placement of congruent copies of the polygons in $\mathscr{P}$ such that the points corresponding to each hinge are identified (Fig. 1).

0.1.3 Properties of Linkages and Graphs

Given two graphs $G = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, a bijection $f : V_1 \mapsto V_2$ such that for any two vertices $u, v \in V_1$ that are adjacent, i.e. $(u, v) \in E_1$, if and only if $(f(u), f(v)) \in E_2$. Next we add restrictions to our graph isomorphisms to narrow our focus:

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

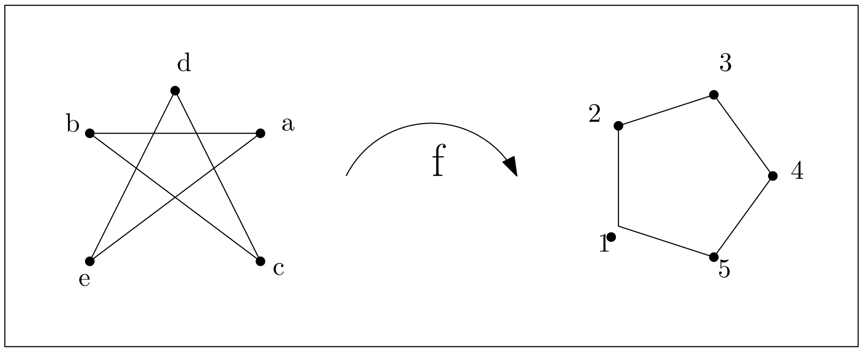


Figure 5: This figure depicts the graph isomorphism shown in Table (**??**) between V_1 and V_2 in the plane.

(i) We focus on isomorphisms for planar graphs and or polygonal linkages, simple planar graphs, and

(ii) the isomorphism preserves edge lengths (polygonal area), e.g. $d(u,v) = d(f(u), f(v))$.

With these restrictions of our isomorphisms, we can begin to describe a range of motion to transform a linkage. That range of motion is said to be the configuration space of that linkage. To expand on this concept, for given linkage, $L = (V,E)$, and for a given vertex $v \in V$, the set of points in which v can be realized in the plane would be the configuration space for that vertex, C_v. Defining some order of the vertices in L, i.e. $V = \{v_n\}_{i=1}^{n}$, then the *configuration space* for L is said to be the cartesion product of the configuration space of vertices:

0.1.4 Summary

	Linkages	Polygonal Linkages
Ordered Pair	G	G
Edges	E	$\mathscr{H}$
Vertices	V	$\mathscr{P}$
l	l	N.A.
Embedding of G	$\Pi : V \mapsto \mathbb{R}^2$	$\mathscr{P}' = \{P_i'\}_{i=1}^{n}$
Realization	See (a)	See (b)

Table 2:

Table 3: (a)The realization for a linkage is for any edge $(u,v) \in E$ such that $|\Pi(u) - \Pi(v)| = l(u,v)$.(b)A *realization* of a polygonal linkage is an interior-disjoint placement of congruent copies of the polygons in $\mathscr{P}$ such that the points corresponding to each hinge are identified (Fig. 1, left).(c).

0.1.5 Configuration Spaces

0.1.5.1 Configuration Spaces of Linkages

Let's focus on the space of embeddings of a linkage. The if there are n vertices of a linkage, *configuration space* of a linkage is said to be a vector space of dimension $2n$ where edge length is preserved. A *configuration*

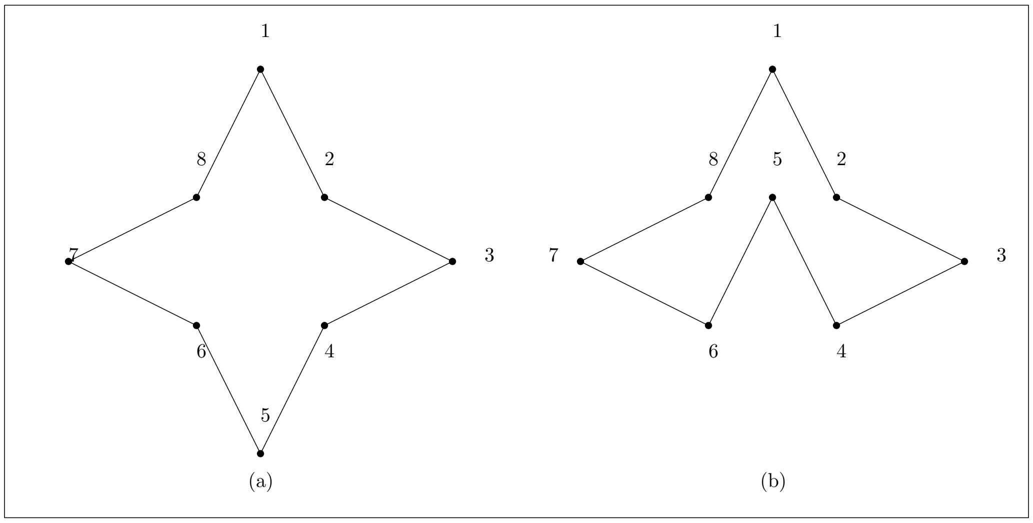


Figure 6: (a) and (b) show a linkage in two embeddings.

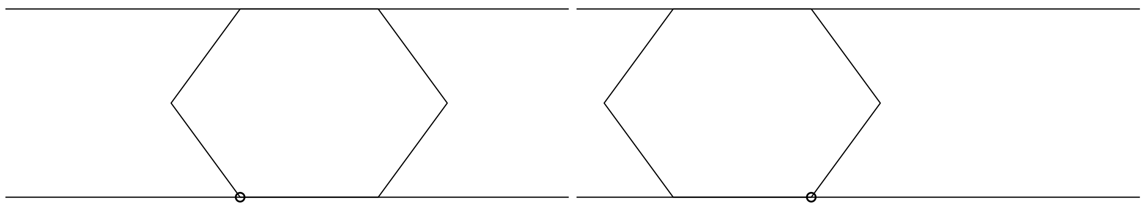
space for a linkage G and corresponding proper embedding, L_1 is said to be for any other proper embedding of a linkage G, L_2, such that the lengths of every edge of G is preserved between the two embeddings, i.e.:

$$l((u,v)) = |L_1(u) - L_1(v)| = |L_2(u) - L_2(v)|$$

Equivalent embeddings include translations and rotations about the center of mass on $L(V)$. We further our embeddings by requiring that one vertice is pinned to the point of origin on the plane as well as a neighboring vertex.

0.1.5.2 Configuration Spaces of Polygonal Linkages

Confining Linkages to a Restricted Space Within a Configuration Space So we've covered the idea of linkages within a plane; now let's constrain the plane to a strip and have a linkage that is a *polygon*, i.e. a linkage that forms a closed chain (e.g. Table **??**), hugging the boundaries of the strip: So here we have

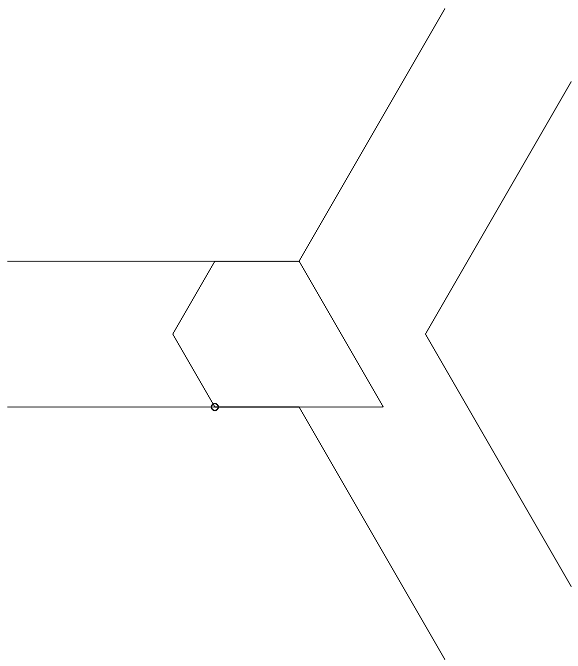


(a) A bounded hexagon that resides in a channel with a pinned vertex

(b) The second realization of the hexagon residing in a channel with a pinned vertex.

Figure 7: Due to the strip in the plane that the hexagon is bounded within the configuration space is limited to just two realizations.

a linkage whose conifguration space is limited to just two realizations. With just two realizations, we can assign a binary value to them and have the linkage act as a boolean variable. We will revisit this concept when we cover satisfiability problems later on in the paper.


(a) A pentagon that is pinned in a channel junction that is formed by the sides of 3 large regular hexagons. It has two possible configurations, much like that of 3

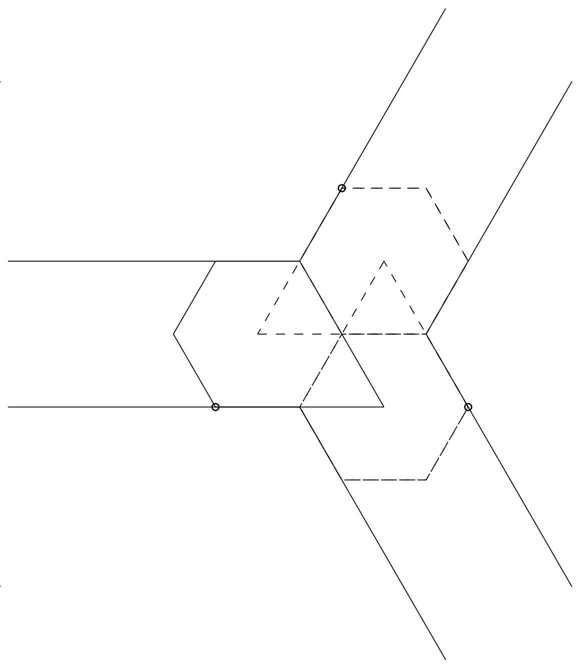

(b) A pinned pentagon residing in a channel junction that is formed by the sides of 3 large regular hexagons with 2 dashed pentagons intersecting it.

Figure 8: Suppose the channel formed is a junction of three regular hexagons. The polygon partially residing in the junction is a regular hexagon with an equalateral triangle appended at an edge. This polygon would prevent other polygons (i.e. the dashed polygons) of the same shape residing in the center of the channel without intersection. This demonstrates that a the configuration space within a multichannel environment can have concurrency issues, i.e. some configurations cannot be realizable.

Expanding upon the idea of 3, forming channels with junctions as shown in Figure 4 can be formed as such by evenly spacing the edges of a hexagonal lattice. Visually, it is shown that only one of three possible pentagons can reside in the channel at one time. By asserting certain conditions on the lattice, and extending the problem to a greater region of a hexagonal lattice, we will be able to pose a realizability problem of whether a configuration $\mathscr{A}$ can be reconfigured to $\mathscr{B}$ by switching pentagons without violating overlapped polygon conditions.

0.1.6 Disk Arrangements

It turns out the disk arrangements are an equivalent way to to represent linkages and their corresponding problems. Before we establish the relation, we will cover some fundamental concepts of disk arrangements. A *disk arrangement* is a set, $\mathscr{D}$, is pairwise interior-disjoint disks in the plane.

Theorem 0.1.1 (Disk Packing Theorem)**.** *For every connected simple planar graph G there is a disk arrangement in the plane whose intersection graph is (isomorphic to) G.*

1. Show the relation between polygonal linkages and disk arrangengements.

0.1.6.1 Oriented Realizations

(i) have the professor explain this part.

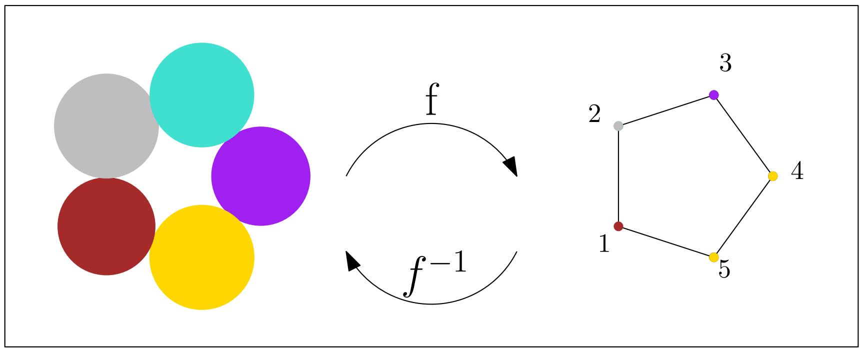
Linkages


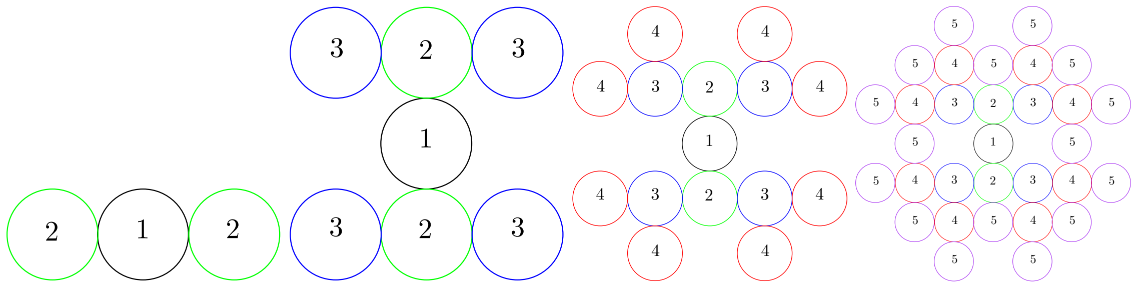
Figure 9: This example represents a disk arrangement transformed to and from its corresponding graph G_2

0.1.6.2 Disk Packing Confinement Problem

Consider the iterative problem:

1. Start with a circle of diameter 1.
2. Add two kissing circles, each of diameter 1, that do not intersect with any other circle (they may kiss other circles).
3. For each new kissing circle added, add two more non-intersecting kissing circles to it.

Figure (**??**) illustrates the iterative problem. The problem with this is that the area in which is necessary to contain this disk growing disk arrangement will exceed the area needed to contain it. This is so frustrating to



(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 10: The gradual growth of disk arrangements by adding two kissing disks to each of the previously generated disks. By continuing this arrangement growth, the space needed to contain the kissing disks will exceed the area containing the disk arrangements.

explain. I understand it, I have trouble explaining it. Then I'd like to show how this relates to our problem.

0.1.7 Satisfiability

Problem 0.1.1 (Satisfiability Problem). Let $\{x_i\}_{i=1}^n$ be boolean variables, and $t_i \in \{x_i\}_{i=1}^n \cup \{\bar{x}_i\}_{i=1}^n$. A *clause* is is said to be a disjuction of distinct terms:

$$t_1 \vee \cdots \vee t_{j_k} = C_k$$

Then the *satisfiability problem* is the decidability of a conjuction of a set of clauses, i.e.:

$$\wedge_{i=1}^{m} C_i$$

[**?**] A *3-SAT problem* is a SAT problem with all clauses having only three boolean variables.

Definition 0.1.1 (Planar 3-SAT Problem)**.** Given a boolean 3-SAT formula B, define the associated graph of B as follows:

$$G(B) = (\{\, v_x \,|v_x \text{ represents a variable in } B \,\} \cup \{\, v_C \,|v_C \text{ represent a clause in } B \,\}, \{\, (v_x, v_C)\,|x \in C \text{ or } \bar{x} \in C \,\}) \tag{1}$$

If $G(B)$ in equation (**??**) is planar, then B is said to be a *Planar 3-SAT Problem* [**?**].

0.1.7.1 Logic Engine

The logic engine simulates the well known Not All Equal 3 SAT Problem (NAE3SAT).

0.1.7.2 Not All Equal 3 SAT Problem

Problem 0.1.2 (Satisfiability Problem). Give a set of clauses C, each containing three boolean variables, can each clause contain at least one true variable and one false variable?

0.1.7.3 Construction of the Logic Engine

The components of the logic engine are as follows: the rigid frame, the shaft, the armatures, the chains, and the flags. The *rigid frame* is a rectangular enclosure with a horizontal shaft place at mid-height. The *armatures* are concentric rectangular frames contained within the rigid frame. Each armature can rotate about the shaft; other motions on the armature are disallowed. Given an NAE3SAT, for each variable there is a corresponding armature. On each armature, there are chains. A pair of *chains*, a_j and $\bar{a}_j$ correspond to the variable x_j and $\bar{x}_j$ respectively. The pair is placed on each armature, reflected at a height of h above and below the shaft, i.e. one place above the shave at a height of h, the other placed below the shaft at a height of $-h$.

0.1.7.4 Encoding the Logic Engine

For each clause of an NAE3SAT, there exists a set of corresponding chains, namely the h^{th} clause is the set of chains on the armatures at the h^{th} row above and below the shaft. A chain is *flagged* if the corresponding variable resides within the clause. The flag can point in either the left or right directions indicating a truth assignment for that variable within the clause. A flag is attached to the i^{th} chain of every a_j^{th} and $\bar{a}_j^{\text{th}}$ chain with the following exceptions:

1. if the variable x_j is in clause C_i, then link i of a_j is unflagged,
2. if the variable $\bar{x}_j$ is in clause C_i, then link i of a_j is unflagged.

Theorem 0.1.2. *An instance of NAE3SAT is a "yes" instance if and only if the corresponding logic engine has a flat, collision-free configuration.*

Proof 0.1.1. *If an instance of NAE3SAT is a "yes", then every clause in C contains at least one true variable and one false variable. Now suppose the following truth assignment:*

$$t\left(x_j\right) = \begin{cases} 1 & x_j \textit{ and } \bar{x}_j \textit{are placed at the top and bottom respectively} \\ 0 & x_j \textit{ and } \bar{x}_j \textit{are placed at the bottom and top respectively} \end{cases} \tag{2}$$

For each clause $c_i \in C$, there exists a variables in c_i such that $t\left(y_i\right) = 1$ and $t\left(z_i\right) = 0$. This implies that there exists an unflagged chain in the i^{th} and $-i^{th}$ row of the frame. To avoid a collision in each row, trigger *the flags to point towards the unflagged link. Thus, the corresponding logic engine has a flat, collision-free configuration.*

*If the corresponding logic engine has a flat, collision-free configuration, then there must exist an unflagged link in each row. Without loss of generality, we have that the variables y_j and z_i is in clause C_i such that $t(y_i) = 1$ and $t(z_i) = 0$ for each i. Thus, we have an instance of NAE3SAT is a "yes" instance. [**?**]*