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

The thesis of Clinton Bowen is approved:

_____   _____
Dr. Silvia Fernandez   Date

_____   _____
Dr. John Dye   Date

_____   _____
Dr. Csaba Tóth, Chair   Date

California State University, Northridge

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

Chapter 1

Background

In this section, we cover the background subjects needed to formally pose the problem and present 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. We then look at satisfiability problems and then review a framework, the logic engine, which can encode a type of satisfiability problem. Finally, we cover the basic definitions of algorithm complexity for **P** and **NP**.

1.1 Graphs

A *graph* is an ordered pair $G = (V, E)$ comprising of a set V of vertices and a set E of edges or lines. Every edge $e \in E$, is an unordered pair of distinct vertices $u, v \in V$ (the edge represents their adjacency, $\{u, v\} \in E$). With this definition of a graph, there are no loops (self adjacent vertices) or multi-edges (several edges between the same pair of vertices).

A motivation for using graphs is modelling physical objects like molecules. This requires an embedding into the plane or $\mathbb{R}^3$. An *embedding* of the graph $G = (V, E)$ is an injective mapping $\Pi : V \mapsto \mathbb{R}^2$ (see Figure 1.1).
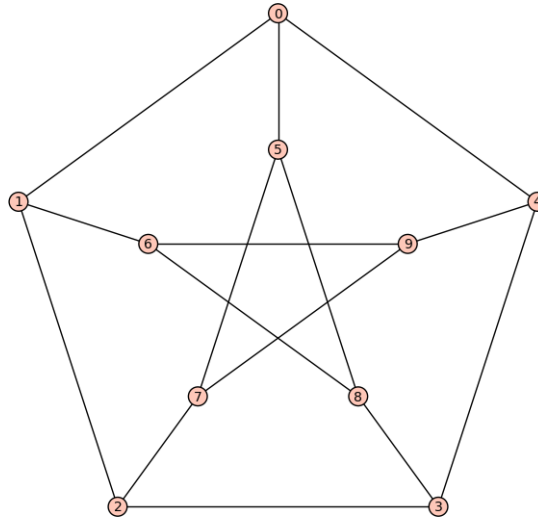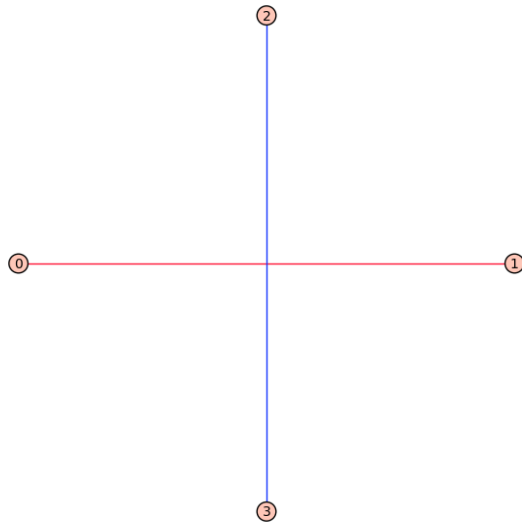

Figure 1.1: An embedding of the Peterson graph.
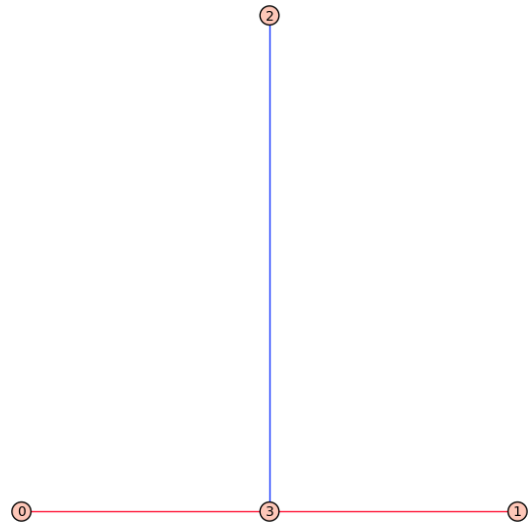
1.1.0.1 Edge Crossings

We define *plane embeddings* to be an embedding where the following degenerate configurations do not occur:

  (i) the interiors of two or more edges intersect, or

 (ii) an edge passing through a vertex

(a) The interior of the edges intersect.

(b) An edge passes through a vertex.

Figure 1.2: These figures exhibit the 4 types of edge crossings.

A *planar graph* is an abstract graph that admits a plane embedding.

1.1.1 Trees

A *path* is a sequence of vertices that represent a traversal of edges within a graph. A *simple cycle* of a graph is a sequence of distinct vertices that do not repeat with the exception of the starting (ending) vertex. A graph is connected if for any two vertices $u, v \in V$, there exists a path between the two points. A *tree* is a graph that has no simple cycles and is connected.
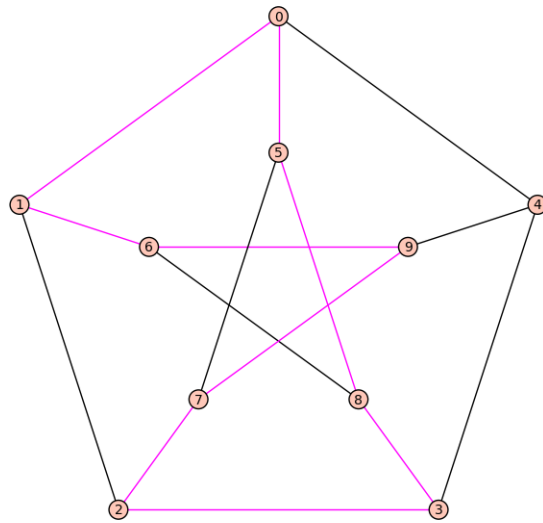
(a) A proper embedding of the Peterson graph with a simple cycle of (2,7,9,6,1,0,5,8,3,2).
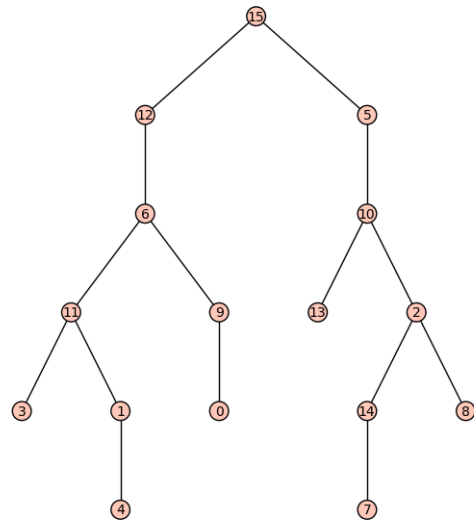
(b) A tree with 16 vertices

1.1.2 Ordered Trees

An *ordered tree* is a rooted tree in which the counter-clockwise ordering of adjacent vertices. Embeddings

(a) An embedding of a tree.   (b) Another embedding of a tree.
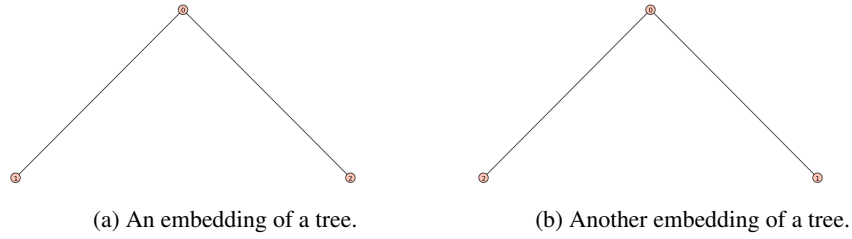
Figure 1.4: A tree with adjacent nodes having a different counter-clockwise ordering. Note that these ordered trees are not equivalent.

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

1.1.3 Graph Isomorphism

To determine when two graphs are equivalent, we need to define an isomorphism for graphs. Given two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, a *graph isomorphism* $f : V_1 \mapsto V_2$ 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 $(f(u), f(v)) \in E_2$.

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 $f(a) = 1$, $f(b) = 2$, $f(c) = 3$, $f(d) = 4$, $f(e) = 5$.
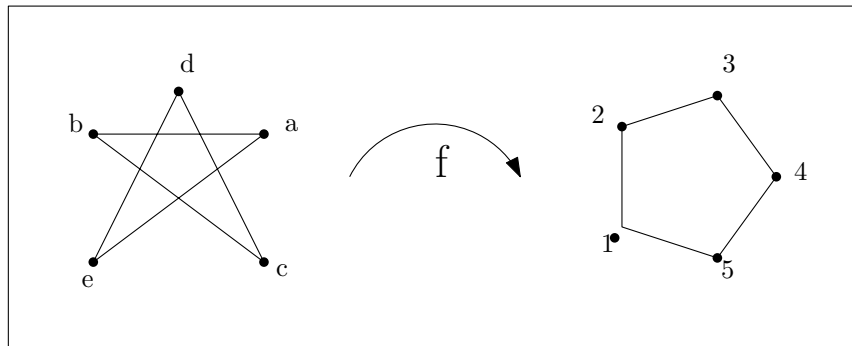

Figure 1.5: This figure depicts the graph isomorphism shown in Table (**??**) between V_1 and V_2 in the plane.

1.1.4 Linkages

When graphs model physical objects, distances between adjacent vertices matter. The length assignment of a graph $G = (V, E)$ is $l : E \mapsto \mathbb{R}^+$. A *linkage* is a graph $G = (V, E)$ with a length assignment $l : E \mapsto \mathbb{R}^+$.

We consider embeddings of a graph that respects the length assignment. A *realization* of a linkage, *G* and *l*, is an embedding fo a graph, Π, such that for every edge $\{u,v\} \in E$, $l(\{u,v\}) = |\Pi(u) - \Pi(v)|$.
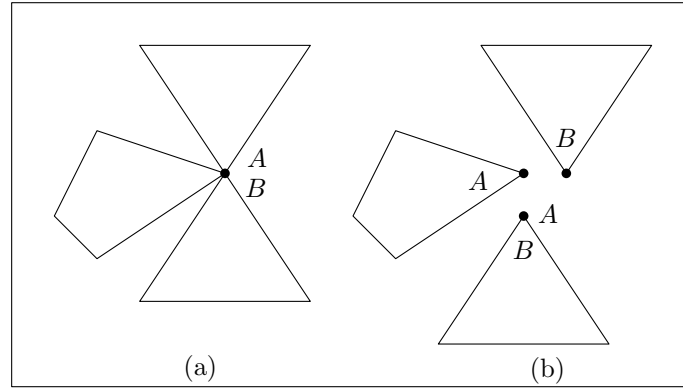
1.1.5 Polygonal Linkages

Figure 1.6: (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.

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 of sets $(\mathscr{P},\mathscr{H})$ where $\mathscr{P}$ is a set of polygons and $\mathscr{H}$ is a set of hinges. Formally, a *hinge $h \in \mathscr{H}$* corresponds to two points on the boundary of two 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 points corresponding to each hinge are identified (Fig. **??**). This definition of realization rules well known geometric dissections (e.g. Fig. 1.7).
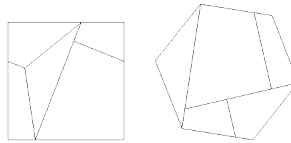

Figure 1.7: Two configurations of polygonal linkage where the polygons touch on boundary segments instead of hinges. These two realizations of the polygonal linkage are invalid to our definitions.

For the remainder of this thesis, we'll focus on the polygonal linkages with the following restrictions:

1. Embedded polygons must be convex, i.e. for any two embedded points $u,v \in P'$, the set:

$$\{u \cdot t + (1-t) \cdot v : t \in [0,1]\} \in P'$$

2. Polygons can only intersect at hinge points. No two polygons can intersect at their boundary or interior with the exception of possible a hinge point.

A difference between hinge points and embedded vertices is that two or more hinge points can be mapped to that same point in $\mathbb{R}^2$. In figure (1.8), we illustrate that two hinge points can reside on the same point in $\mathbb{R}^2$. Figure (1.6) and figure (**??**) are examples of special cases that we may run into, but do not want to focus heavily on. They are presented to the reader to facilitate understanding of the definitions of polygonal
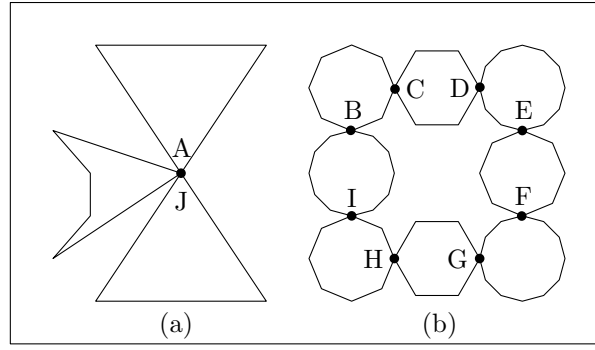
Figure 1.8: (a) A polygonal linkage with a non-convex polygon and a hinge point corresponding to three polygons. (b) A polygonal linkage with 8 regular polygons.

linkages and linkages respectively. Without loss of generality, for this paper, we focus on polygonal linkages that are equivalent to simple planar graphs.

1.1.6 Disk Arrangements

It turns out the disk arrangements are an equivalent way to to represent plane graphs. By representing vertices as disks and edges are represented as points of contact, a *kissing point* between two disks. The graph corresponding to a given disk arrangement, $\mathscr{D}$, is said to be the *contact graph*.

A *disk arrangement* is a set, $\mathscr{D}$, is pairwise interior-disjoint disks in the plane. $\{C_i\}_{i=1}^n$ such that for any circle $C \in \{C_i\}_{i=1}^n$, C is tangent to a different circle of $\{C_i\}_{i=1}^n$.

Theorem 1.1.1 (**??**Disk Packing Theorem). *For every graph G, there is a disk arrangement in the plane whose contact graph is isomorphic to G.*

Proposition 1.1.1. *For every graph L, there is a disk arrangement in the plane whose contact graph is isomorphic to L.*
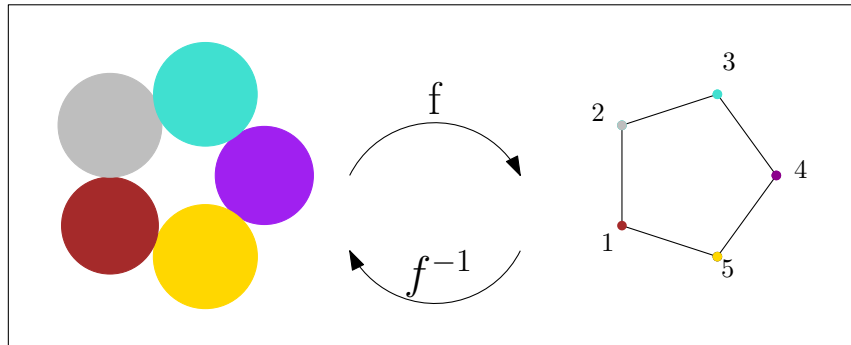
Figure 1.9: This example represents a disk arrangement transformed to and from its corresponding graph G_2

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

1.1.6.1 Ordered Disk Arrangement

Suppose we're given a tree. By the disk packing theorem we can ascertain a sense of order for the isomorphic disk packing. An *ordered disk arrangement* is a rooted tree in which the counter-clockwise ordering of adjacent vertices.

5

1.1.6.2 Disk Packing Confinement Problem

By adding constraints to the embeddings of disk arrangements, we can devise realizability problem by a volume argument.

1. Round 1: Start with a disk of diameter 2.

2. Round 2: Add two kissing disks, each of diameter 2, that do not intersect with any other disk (they may kiss other disk).

3. Round 3 and Higher: For each new kissing disk added, add two more non-intersecting kissing disks of diameter 2 to it.

For each round i we are adding $2^{(i-1)}$ disks, each with an of π. The area that the disk arrangement is bounded by at round i is a box of length $2 \cdot (2 \cdot (i-1) + 1)$ totalling to an area of $(4 \cdot i^2 - 4 \cdot i + 1)$. Meanwhile the total area of the disk arrangement at round i is $\pi \cdot (2^i - 1)$. The exponential growth rate of the disk packing will exceed its bounded area for sufficiently large i.

Figure (1.10) 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.

(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

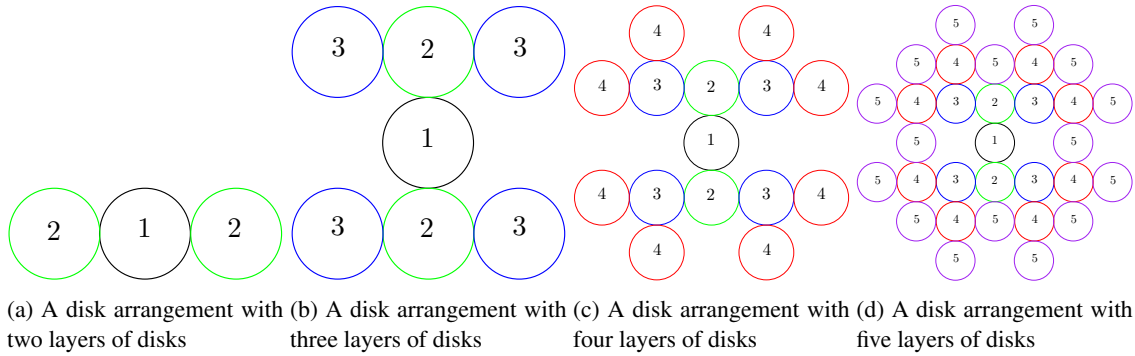(d) A disk arrangement with five layers of disks

Figure 1.10: 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.

1.2 Configuration Spaces

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

Alexander Calder, 1933

We'd like to describe motions and range of motions of embedded graphs, linkages, polygonal linkages, and disk arrangements.

Theorem 1.2.1 ([?, ?]Carpenter's Rule Theorem). *Every realization of a 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.*

A configuration space is *connected* if for any two configurations, there is .....

1.2.1 Configuration Spaces of Linkages

Let's focus on the space of embeddings of a linkage. If there are n vertices of a linkage, the *configuration space* of a linkage is said to be a vector space of dimension $2 \cdot n$ where edge length is preserved. A *configuration space* for a linkage G and corresponding proper embedding, L_1 is said to be for any other
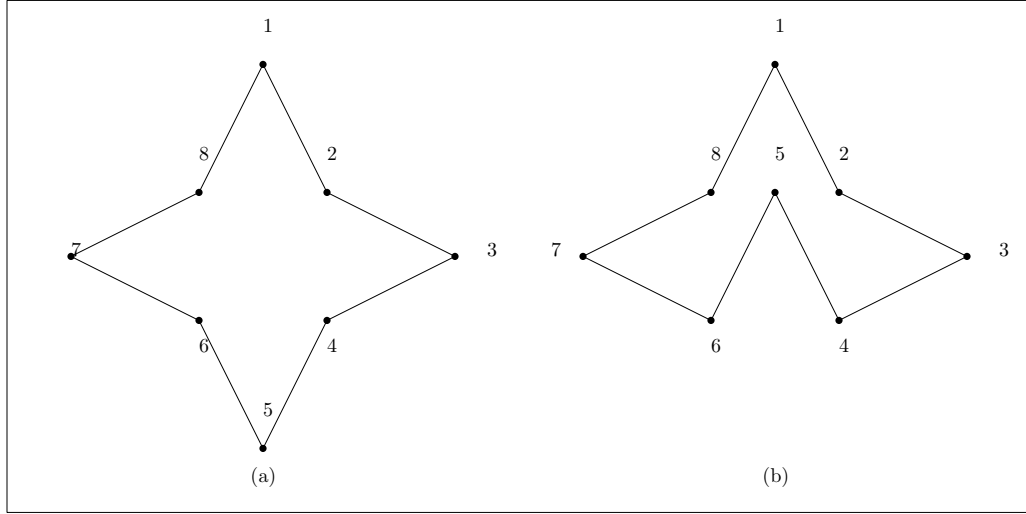

Figure 1.11: (a) and (b) show a linkage in two embeddings.

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.:

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

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.

1.2.2 Configuration Spaces of Polygonal Linkages

1.2.3 Configuration Spaces of Disk Arrangements

1.3 Problem

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 1.3.1. *It is NP-complete to decide whether a polygonal linkage whose hinge graph is a tree can be*

7

realized (both with and without orientation).

Theorem 1.3.2. *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 1.3.1 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 **??**).

1.3.1 Problem Statement

text

1.3.2 Decidability of Problem

test

1.4 Satisfiability

Problem 1.4.1 (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 1.4.1 (Planar 3-SAT Problem)**.** Given a boolean 3-SAT formula B, define the associated graph of B as follows:

$$G(B) = (\{v_x | v_x \text{ represents a variable in } B\} \cup \{v_C | v_C \text{ represent a clause in } B\}, \{(v_x, v_C) | x \in C \text{ or } \bar{x} \in C\})$$
$$\text{(1.1)}$$

If $G(B)$ in equation (1.1) is planar, then B is said to be a *Planar 3-SAT Problem* [**?**].

1.4.1 Not All Equal 3 SAT 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?

1.4.2 Planar 3 SAT Problem

Problem 1.4.3 (Planar 3 SAT Problem). A planar 3SAT instance is a 3SAT instance for which the graph built using the following rules is planar:

1. add a vertex for every x_i and $\bar{x}_i$

2. add a vertex for every clause C_j

3. add an edge for every $(x_i, \bar{x}_i)$ pair

4. add an edge from vertex x_i (or $\bar{x}_i$) to each vertex that represent a clause that contains it

5. add edges between two consecutive variables (x_1, x_2), (x_2, x_3), ...,(x_n, x_1)

In particular, rule 5 builds a "backbone" that splits the clauses in two distinct regions.

1.4.3 Logic Engine

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

1.4.4 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 $-h$.

1.4.5 Encoding the Logic Engine

For each clause of an NAE3SAT, there exists a set of corresponding chains, namely the hth clause is the set of chains on the armatures at the hth 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 ith 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 1.4.1. *An instance of NAE3SAT is a "yes" instance if and only if the corresponding logic engine has a flat, collision-free configuration.*

Proof 1.4.1. *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:*

$$t(x_j) = \begin{cases} 1 & x_j \text{ and } \bar{x}_j \text{are placed at the top and bottom respectively} \\ 0 & x_j \text{ and } \bar{x}_j \text{are placed at the bottom and top respectively} \end{cases} \tag{1.2}$$

For each clause $c_i \in C$, there exists a variables in c_i such that $t(y_i) = 1$ and $t(z_i) = 0$. This implies that there exists an unflagged chain in the ith and $-i^{\text{th}}$ 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. [?]

Problem 1.4.4 (Unordered Realizibility Problem for Linkages). For a linkage, it asks whether its corresponding graph is the contact graph (resp., ordered contact graph) of some disk arrangement with specified radii.

Problem 1.4.5 (Unordered Realizibility Problem for Polygonal Linkages). The *realizability* problem for a polygonal linkage asks whether a given polygonal linkage has a realization.

Problem 1.4.6 (Ordered Realizibility Problem for Linkages). For a linkage, it asks whether its corresponding graph is the ordered contact graph of some disk arrangement with specified radii.

Problem 1.4.7 (Ordered Realizibility Problem for Polygonal Linkages). The *realizability* problem for a ordered polygonal linkage asks whether a given polygonal linkage has a realization with respect to order.