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

Table of Contents

Signature page ii

Abstract iv

1 Decision Problems for Hinged Polygons and Disks 1
1.1 The Logic Engine 1
1.1.1 Construction of the Logic Engine 1
1.1.2 The mechanics of the logic engine 3
1.2 Logic Engines Represented as Polygonal Linkages 6
1.2.1 Construction of the Polygonal Linkage Logic Engine 6

ABSTRACT

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

By

Clinton Bowen

Master of Science in Applied Mathematics

Chapter 1

Decision Problems for Hinged Polygons and Disks

1.1 The Logic Engine

The *logic engine* is a planar, mechanical device that simulates an instance NAE3SAT problem. It was introduced in Bhatt et. al. [1].

1.1.1 Construction of the Logic Engine

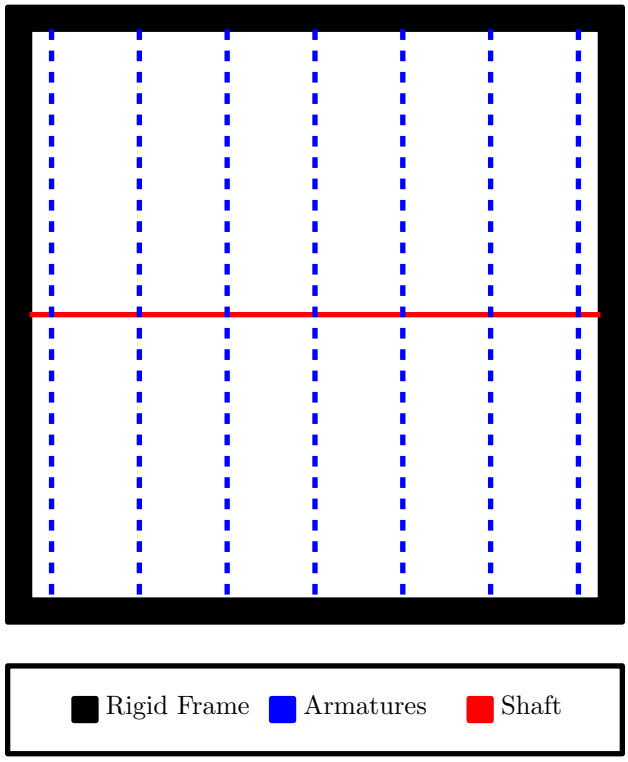


Figure 1.1: A logic engine frame with veritical armatures and a horizontal shaft.

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.

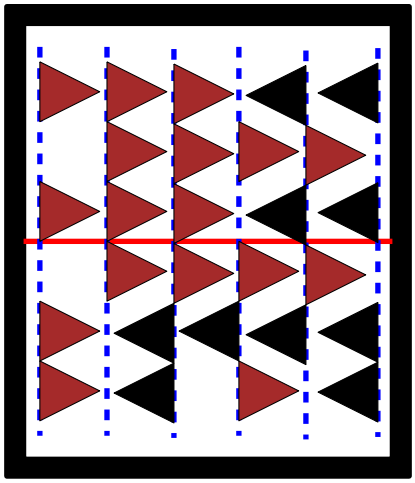
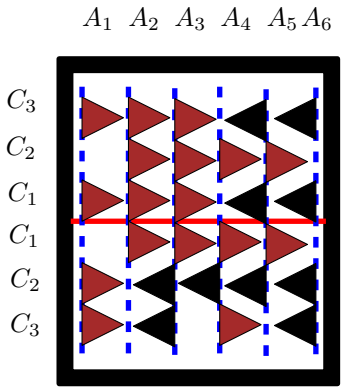


Figure 1.2: A logic engine that corresponds to a boolean formula in NAE3SAT form, Φ. The picture shows the outer rigid frame, the shaft, the armatures that correspond to the variables in Φ, with oriented flags.

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 1.3: 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 2 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	$\{(x,y) \in \mathbb{R}^2 \,\vert \text{The boundary of } [\frac{1}{2}, n+\frac{1}{2}] \times [-m,m]\}$
Shaft	1	$\{(x,y) \in \mathbb{R}^2 \,\vert x \in [\frac{1}{2}, n+\frac{1}{2}] \text{ and } y = 0\}$
Armatures	n	For the j^{th} armature we have $\{(x,y) \in \mathbb{R}^2 \,\vert x = j \text{ and } y \in [-m,m]\}$
Flags	$2mn - 3m$	if $\ell_{j,k}$ is flagged then the attached flag is a regular triangle with side length 1.

Table 1.1: The quantity and coordinates of the logic engine components.

1.1.2 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.

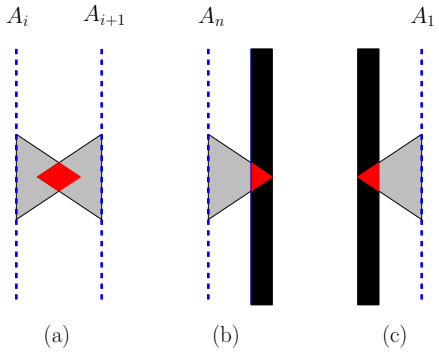


Figure 1.4: (a) Illustrates a adjacent flag collision at the same height, (b) and (c) illustrates a rigid frame collision.

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.

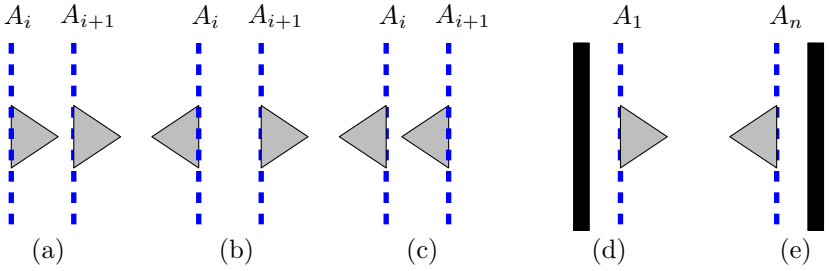


Figure 1.5: The following configuration of adjacent flags and flags that are adjacent to the rigid frame.

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.

1.1.2.1 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 1. *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 2. *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 1 we would have a polynomial time algorithm to decide whether an instance of the NAE3SAT is a 'yes' instance. Since NAE3SAT is NP-Hard [2], there is no such algorithm unless $P = NP$. □

1.2 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 1.7, each flag becomes a rectangle.

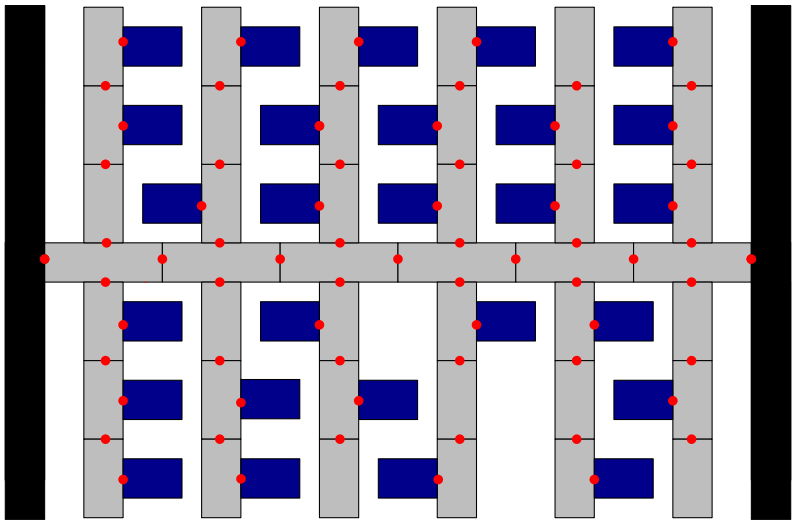


Figure 1.7: A logic engine realized as a polygonal linkage.

1.2.1 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 $P_\ell = (\mathcal{P}, \mathcal{H})$ 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 1.2: The components of $\mathcal{P}$ 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.

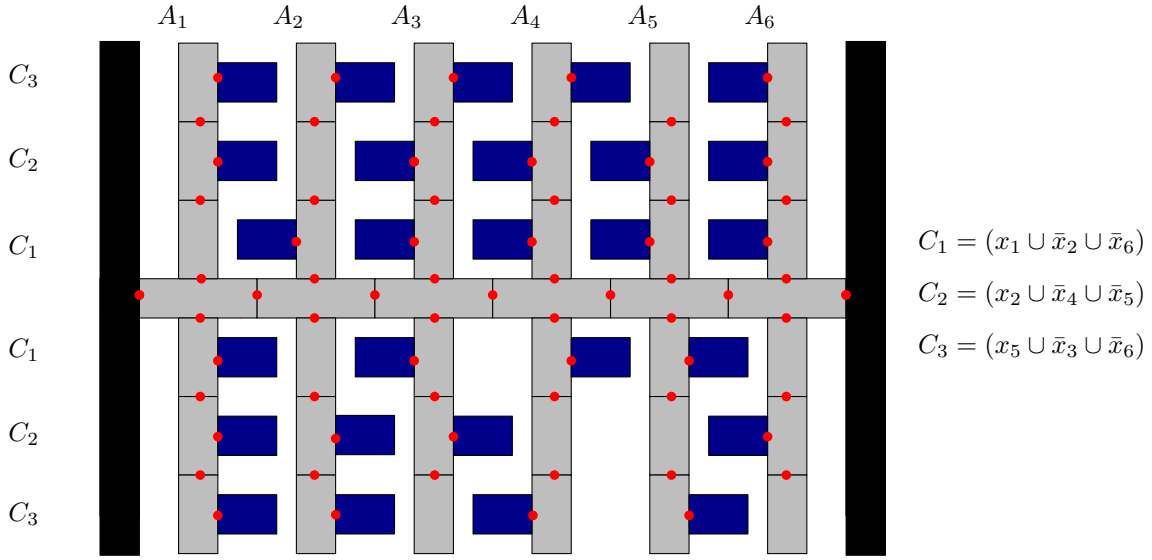


Figure 1.8: A polygonal linkage logic engine that corresponds to the boolean formula $\Phi = C_1 \cap C_2 \cap C_3$.

Theorem 3. *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 $NAE3SAT$. □

References

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

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