Combinatorial and Geometric Structure in the Self-Assembly of Polyhedra

by

Daniel C. L. Johnson

B.S., Rensselaer Polytechnic Institute; Troy, NY, 2009

Sc.M., Brown University; Providence, RI, 2012

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in The Division of Applied Mathematics at Brown University

PROVIDENCE, RHODE ISLAND

May 2015

This dissertation by Daniel C. L. Johnson is accepted in its present form by The Division of Applied Mathematics as satisfying the dissertation requirement for the degree of Doctor of Philosophy.

Date_____ _____

Govind Menon, Ph.D., Advisor

Recommended to the Graduate Council

Date_____ _____

Basilis Gidas, Ph.D., Reader

Date_____ _____

Miranda Holmes-Cerfon, Ph.D., Reader

Approved by the Graduate Council

Date_____ _____

Peter Weber, Dean of the Graduate School

Vitae

Acknowledgements

Abstract of "Combinatorial and Geometric Structure in the Self-Assembly of Polyhedra " by Daniel C. L. Johnson, Ph.D., Brown University, May 2015

Contents

List of Tables

List of Figures

Chapter One

Introduction

Self-assembly is a class of formation process in which a product is constructed with out explicit manipulation of its parts. Many—often identical—parts come together by utilizing the dynamics of their environment to create a finished structure. Sometimes such assemblies can be encouraged by manipulating broadly controlled parameters of the assembly environment such as temperature, solution content, and assembly subunit selection.

There are many examples of both natural and synthetic self-assembly process that take a wide variety of length scales and serve a plethora of functions. An area of much active research is the self-assembly of RNA, proteins, and viral capsids. These biological processes act on the nano scale and, while it is known that their formation can be aided by a variety of secondary mechanisms such as helper RNA, the formation process is not well understood in general. Synthetic examples of molecular self assembly include the formation of supramolecular cages. These supramolecular structures that can encapsulate a smaller molecule show promise in contributing to a number of medical and scientific fields including drug delivery and nano-scale circuits.

–Leventhal Paradox

In general, the processes of biological self-assembly are not well understood due to difficulties arising from their small length scale. Conversely, synthetic self-assembly processes often lack the complexity and sophistication of their biological equivalents. The goal of our research is to explore discrete geometric models of self assembly. By using analysis made possible by the simplicity of said models, important properties of the processes are to be identified. Of primary importance is identifying the pathways of formation which consist of a specific order in which unfinished intermediate states are visited before the assembly is completed. It is thought that these pathways

are often robust in the sense that the process follows a very small, and sometimes unique, number of pathways to form the end product. Also of interest is identifying mechanisms by which failed or flawed formations can be avoided or minimized. Possible strategies include the selection of specific precursors, the selection of solution, and the mid-assembly control of experimental parameters.

1.1 Polyhedra

–model molecules, proteins, but also on a larger scale, polyhedra are 'nice' choice for assembly.

–origins

–classes of polyhedra

–basic definitions

–polytopal complex

1.2 Scientific Motivation

1.2.1 Self-Folding Polyhedra

In experiment, Pandey et al have been able to form closed sub-millimeter scale polyhedral structures from the self-folding of flat polyhedral nets. Using photolithogra-

phy, these nets are cut from a two dimensional sheet with great precision. By adding a specific amount of solder at the net's hinges, surface tension from the melted solder causes the net to fold up into the closed polyhedron. Several different polyhedra have been attempted with varying degrees of success. It has been argued that the geometric structure of the polyhedron are largely determinant of the net's propensity to successfully assemble into the polyhedron. In some cases, two distinct polyhedral isomers can be formed from the same net.

1.2.2 Molecular Cages

Self-assembly of molecular cages are the subject of much active research. By isolating molecules of interest inside a molecular cage, targeted application of these molecules is theoretically possible. This has enormous implications in the medical industry. Additionally, by creating a grid-like network of such cages, it may be possible to construct functional electric circuitry at the nano scale.

Fujita et. al. have theorized and subsequently synthesized a family of organometallic cages consisting of metallic connector molecules (M) that each connect four bent ligand molecules (L). With the M molecules as vertices and the L molecules as edges, they can form polyhedral cages. Due to geometric constraints, the number of M and L molecules in a completed cage must satisfy $|M| = k$ and $|L| = 2k$ for $k \in 6, 12, 24, 30, 60$. Each of these five choice of k results in a cage that geometrically resembles a specific Archemedian solid.

In one experiment, two different ligands (L_1 and L_2) with slightly different bend angles were used. As the ratio of $L_1 : L_2$ was varied, each ratio only resulted in the formation of one of the possible cages. More specifically, for ratios $L_1 : L_2 < 0.25$

only $M_{12}L_{24}$ formed and for $L_1 : L_2 > 0.25$ only $M_{24}L_{48}$ was formed. This steep and curious cutoff thus far defies tangible explanation.

In experiments by Liu et al, polyhedral supramolecular cages made from different molecular species were synthesized. Theoretically, tiling type behavior of the molecular species is possible, but it was not observed experimentally. Additional experiments in which copies of two types of molecular species could be combined to form two different polyhedral cages were proposed. It is unknown if both potential polyhedral cages would form or if one would be significantly more favorable over the other. Being able to predict the results of such experiments via mathematical analysis and simulation would be a significant contribution toward optimizing strategies for the self-assembly of supramolecular cages.

1.2.3 Viral Capsid Assembly

With forces such as friction playing a much bigger role on their length scale, biological viruses are remarkable at robustly replicating themselves. While this topic has been well studied and a wide variety of mechanisms have been evidenced to contribute, there still is no general understanding of the process in which a virus' capsid is formed. Viral capsids come in many shapes and sizes, but a significant portion of them are icosahedral in structure. Understanding of their pathways of formation may be a key to preventing or slowing down replication.

1.2.4 RNA and Protein Folding

Protein and RNA folding are active fields of research. If we can predict the structure of a RNA or protein based on its amino acid sequence, we will know more about its biological function. As many human and animal disorders are caused by proteins folding abnormally, their cures may lie in the understanding of the folding pathway. While we do not directly consider applications relating to RNA and protein folding, success with the above problems may also provide insights for this complex topic.

Just as in self-assembly, folding constitutes the passage between several intermediate states along a pathway leading to formation of a final structure. In both examples, the majority of successful assemblies are thought to funnel through a small number of formation pathways. The identification of these pathways may allow for the targeted intervention at a specific intermediate in the case of abnormal folding behavior.

1.3 Mathematical Constructs for Self-Assembly

1.3.1 Attachment

First proposed by Zlotnick et al, the Building Game (BG) is a discrete attachment model that simulates the sequential construction of polyhedral structures. The BG can describe the behavior of many two dimensional face molecules interacting in a solution and bonding together to ultimately form polyhedral molecule.

–More

1.3.2 Folding

Another model that we refer to as the Folding Model, was used to model the self-folding of meso-scale metallic polyhedra mentioned in section 1.2.1. In this model, a polyhedron's net is sequentially folded up into the finished polyhedron. At each stage one vertex is closed to form a pyramidal structure by a folding move. In analogy with the building game, we can define folding intermediates to be partially folded states and a state spaces where intermediates are connected if one can form the other with a single fold. The state space and corresponding intermediates are shown in figure 1.1.

Interestingly, each polyhedron may has multiple nets. In fact, the number of distinct nets grows rapidly with the size of the polyhedron. Certain nets have different folding pathways and measuring the favorability of one of these nets over another introduces an design problem. If nets that fold into the completed polyhedron most reliably can be identified, the efficiency of the self-assembly process can be optimized.

Figure 1.1: The folding model on the octahedron.

sequence of folds,+ both octa and boat.

As a feature, this model has been shown to allow folds that do not result in the initially intended polyhedron and can terminate in other polyhedra and blocked states. For example, many of the intermediates that cannot fold into the octahedron end up folding into a non-convex boat intermediate.

Like in the building game, the state space of the folding model experiences a combinatorial explosion as the number of faces the polyhedron has increases. Surprisingly, the building game can be used to recover the part of the folding state space which allows folding to the originally intended polyhedron.

–More on this elsewhere?

1.3.3 Local Rules

A large number of previously studied models for self-assembly can be classifies as *local rules* based approaches. Such models typically consist of a collection of components that can be combined according to a specified grammar. While this is a basic idea, the variety off possible component and grammars makes this a widely flexible class of models. Schwartz et al used such a model to describe the assembly of viral capsids. Since these capsids are fundamentally composed of proteins that can each assume a number of conformations. By putting a grammar on the ways in which proteins of different conformation can combine, they were able to successfully form a variety of viral capsids in simulation. While this class of models is powerful, we have not actively pursued any such models. In the future it may be worth trying to use a local rules model to generalize the building game in a way that allows formation errors as in the folding model.

1.3.4 Dominant Formation Pathways

The concept of formation pathways that occur with overwhelming frequency relative to the myriad of other theoretically possible pathways is a primary interest of our work. To identify the nature of such dominant pathways is to identify the mechanism of self-assembly itself. This knowledge would hopefully enable the proliferation of the self-assembled products to become more efficient and faster, or in cases such as biological viruses, inhibit formation altogether. There are several means, both quantitative and qualitative, by which we seek to identify and study these dominant

pathways.

1.3.5 Energy Landscapes

–general idea

–reversible Markov process

Markov processes can be used in a similar way for a variety of different discrete models for self-assembly. The folding model tends to describe an irreversible process, so some of the concepts inherent to Markov processes, such as the stationary distribution would not apply. The dynamical aspects, however, may still be appropriately modeled with Markov processes. While we have not explored the topic extensively, it seems as though, due to their modeling and analytic flexibility, Markov processes provide a natural approach to many local rules type models as well.

Due to the scale of many of these self-assembly processes, statistical mechanics provides a framework to think about these models. By finding an analogy with a diffusion process on an implied potential energy surface, we can use statistical mechanical tools to examine the properties of this potential surface.

1.3.6 Disconnectivity Graphs

Wales et al introduced a graphical way to represent the structure of a potential surface possessing many local minima and intermediary transition states. Named a *discontinuity graph* (DG), the tree with leaves representing local minima of the

potential and other connecting nodes representing transition states that the minima can be reached from provides a way of picturing the structure of the a potentially high dimensional energy landscape that looks past simple energy differences. The vertical height of each node is used to represent the potential energy of that state. Horizontally, the tree is organized to partition these local minima into corresponding funnels. If the graph is truncated at a specific energy level, two minima are reachable via transitions to states strictly below this truncation energy if and only if they remain connected in the truncated disconnectivity tree.

Figure 1.2: Example of disconnectivity tree.

While not a quantitative way of assessing a model's pathways, disconnectivity graphs are extremely useful in identifying broad qualitative properties and can be instrumental in gaining insight into a model's dynamics. The use of such graphical and otherwise qualitative techniques should not be overlooked as they can inspire techniques for more quantitative analysis.

1.4 Original Contributions

– Formalized mathematical framework for BG

– Used energetic perspective to model BG processes as a reversible Markov chain.

– Enumerated combinatorial configuration space for all Platonic, Archimedean, and Catalan solids up to 30 faces.

– First to enumerate the number of shellings of these polyhedral classes.

– Counted degrees of freedom of BG intermediates.

– Developed a manifold reflected Brownian motion scheme to sample configurations.

– Used sampled configurations to estimate transition rates of reversible Markov chain.

Chapter Two

The Building Game: Modeling

2.1 The Building Game as a Mathematical Framework for Self-assembly

The Building Game (BG) was first considered by Zlotnick [2] as a model for the assembly of polyhedral viral capsids. In the model, a capsid is idealized as a polyhedron with each face treated as a subunit. Assembly proceeds from a singe face with a second face attached to the first along an edge. At each subsequent step of the process, an additional face is added along an edge of one of the already added faces. The process ends when all of the faces have been added resulting in a completed polyhedron.

A useful way to think about the Building game is as a sequential coloring process. Given a polyhedron with each face painted white, choose a face and paint it black. At each subsequent step, choose a white face that is adjacent to a black face and paint it black. Repeat until all faces are black. Here the black faces represent a face being present at a given step of the assembly process.

2.2 Formal Definition

We formalize the Building Game in terms of the group action of the polyhedron's rotation group G acting on subsets of F, the polyhedron's face set. Using the polyhedron's dual graph representation $\mathfrak{G}(F, E)$ whose nodes are the faces F of the polyhedron and connections E correspond to the pairs of faces sharing an edge we can define the mathematical structures that comprise different building game configurations.

Definition 1. *A Building Game* ***state*** $x \subset F$ *is a non-empty subset of the faces* F *of a polyhedron such that the subgraph* $\mathfrak{G}|_x$ *restricted to the faces in* x *is a single connected component.*

It is often useful to depict building game states with Schlegel diagrams CITE which are two dimensional projections of the three dimensional polyhedron. Figure 2.1 uses Schoolgirl diagrams to depicts all of the states of the tetrahedron. Since every face of the tetrahedron is adjacent to every other face, any non-empty subset of faces is a building game state. Thus, there are $\binom{4}{k}$ tetrahedron states that have k faces and $2^4 - 1 = 15$ states in total. As pictured in figure 2.2, some subsets of faces

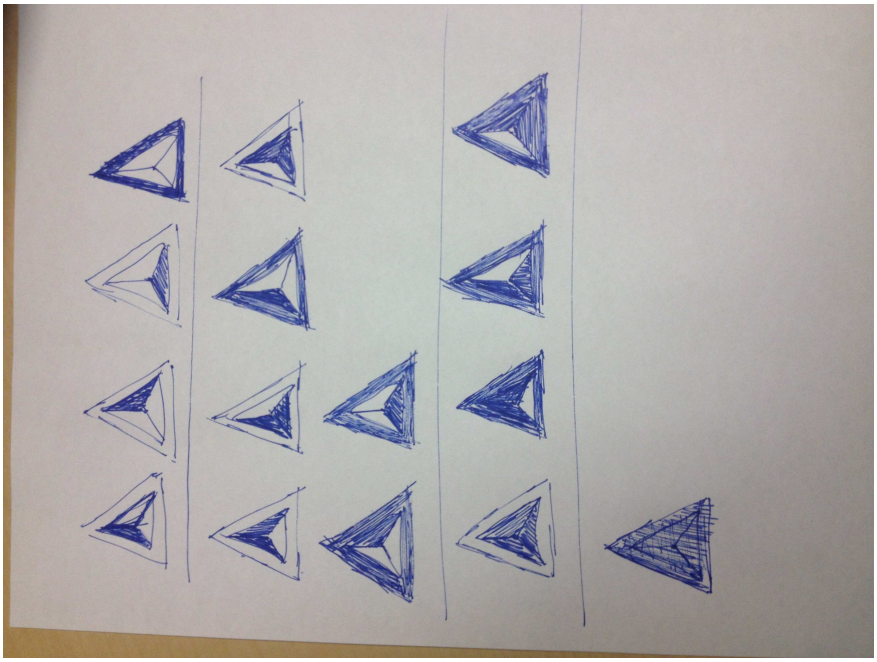

Figure 2.1: The building game states of the tetrahedron.

are not states. For instance, the subset of octahedron faces with only two faces that are not adjacent is not a state. Similarly, the subset of two faces meeting only at a vertex is not a state as they are not connected through edge adjacency in the graph $\mathfrak{G}$. However, when more faces are added to connect these faces in $\mathfrak{G}$ the subset is indeed a state.

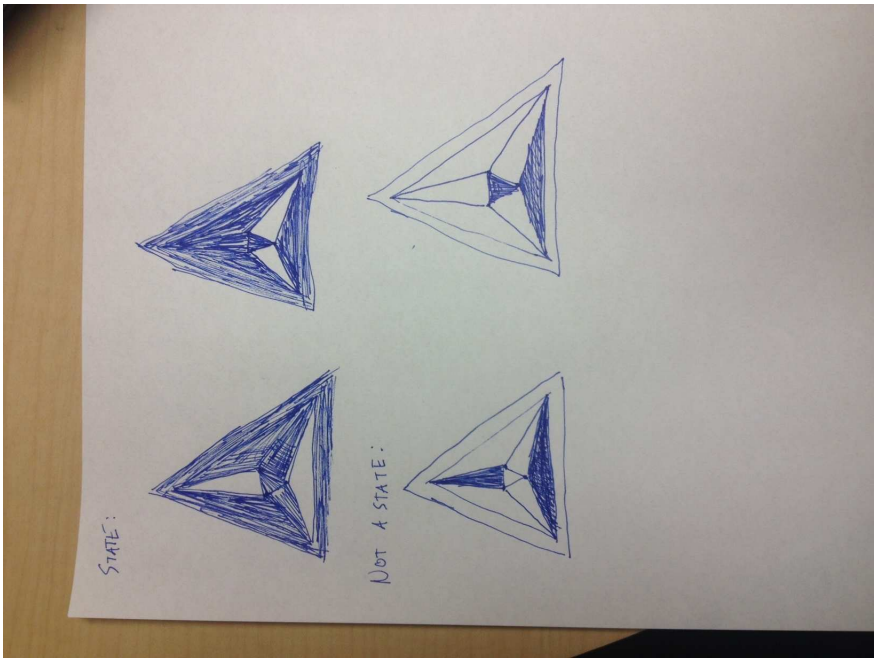

Figure 2.2: Examples of octahedron states and non-states.

2.2.1 Group Actions

It is easy to see that many states are combinatorially equivalent and are just rotations of each other. As in Zlotnick, we group the states into sets that are rotations of each other CITE. However, to do so, we must first build the mathematical infrastructure using group actions.

Definition 2. *A* ***group action*** *of a group G on a set X is a function mapping $G \times X$ to X with $(g, x) \mapsto g.x$ that satisfies: (i) $(gh).x = g.(h.x)$ for $g, h \in G$ and (ii) $e.x = x$ for e the identity element of G.*

We typically use a polyhedron's rotation group or one of its subgroups as G and 2^F, the set of subsets of F, as the set X that G acts on. In this case, the action $g.x$ permutes the faces of the polyhedron according to the element g of the rotation group and results in a new subset of faces. Here, we introduce the group action concepts of orbits and stabilizer subgroups as they play an important role in our

later analyses.

Definition 3. *Let G be a group acting on a set X, the **orbit** of an element $x \in X$ is the subset $G.x \doteq \{g.x : g \in G\}$ of X [1].*

We also use the shorthand notation $[x]$ to refer to the orbit $G.x$ since an orbit can be thought of as an equivalence class under the relation $x \sim \hat{x}$ if there is a $g \in G$ such that $x = g.\hat{x}$.

Definition 4. *For a group G acting on a set X, the **stabilizer subgroup** for an element $x \in X$ is the subgroup $G_x \doteq \{g \in G : g.x = x\}$ of G that fixes x [1].*

Now, we introduce three classical results of group actions that relate orbits and stabilizer subgroups and add a corollary that we will use frequently.

Theorem 1 (Orbit-Stabilizer [1])**.** *Let G be a group acting on a set X, then for any $x \in X, |G.x| = [G : G_x]$, the number of left-cosets of G_x.*

Theorem 2 (Lagrange [1])**.** *If G is a finite group and $S \leq G$, then $|S|$ divides $|G|$ and $[G : S] = |G|/|S|$*

Lemma 1 (Burnside [1])**.** *Let G be a finite group acting on a set X, then*

$$|X/G| = \frac{1}{|G|} \sum_{g \in G} |X^g|$$

where $|X/G|$ is the number of orbits and $|X^g| = \{x \in X : g.x = x\}$

Corollary 1. *Let G be a finite group acting on a set X, then for any $x \in X$*

$$|G.x| = \frac{|G|}{|G_x|}.$$

Proof. This result follows trivially from the Orbit-Stabilizer and Lagrange's theorems. □

2.2.2 Building Game Intermediates

With this framework of group actions, we now formally define the principle unit of the building game.

Definition 5. *A Building Game* ***intermediate*** $[x] \doteq \{g.x : g \in G\}$ *is the orbit of the state* x *under the polyhedra's rotation group.*

Since the orbits of a group action form a partition on the set it acts on, each state belongs to a single intermediate and two states x and y are part of the same intermediate if there is a $g \in G$ such that $y = g.x$. In the case of the tetrahedron, as pictured in figure 2.1, there are only four intermediates since any state with the same number of faces can be rotated to reach the others.

Since we have defined the intermediates to be the orbits of states under the polyhedral group, we are naturally also interested in stabilizer subgroups of building games states.

Definition 6. *The* ***symmetry number*** r_{x^j} *(or simply* r_j*) of a state* x^j *is the order of its stabilizer subgroup* $|G_{x^j}|$.

In figure 2.3 we see three states and their orbit stabilizer subgroups. The first state, with only a single face, has a symmetry nummber of 3 since any rotation by a multiple of $\frac{2\pi}{3}$ fixes the face. The second has a symmetry number of 2 since only the identity and a π rotation will fix the faces. The final state, with three faces, cannot be fixed by any rotation other than the identity and thus hs symmetry number 1.

Theorem 3. *If the states* x *and* $\hat{x}$ *are members of the same intermediate* $[x]$*, they have the same symmetry number. Thus, we extend the notion of a symmetry number to be a property of an intermediate.*

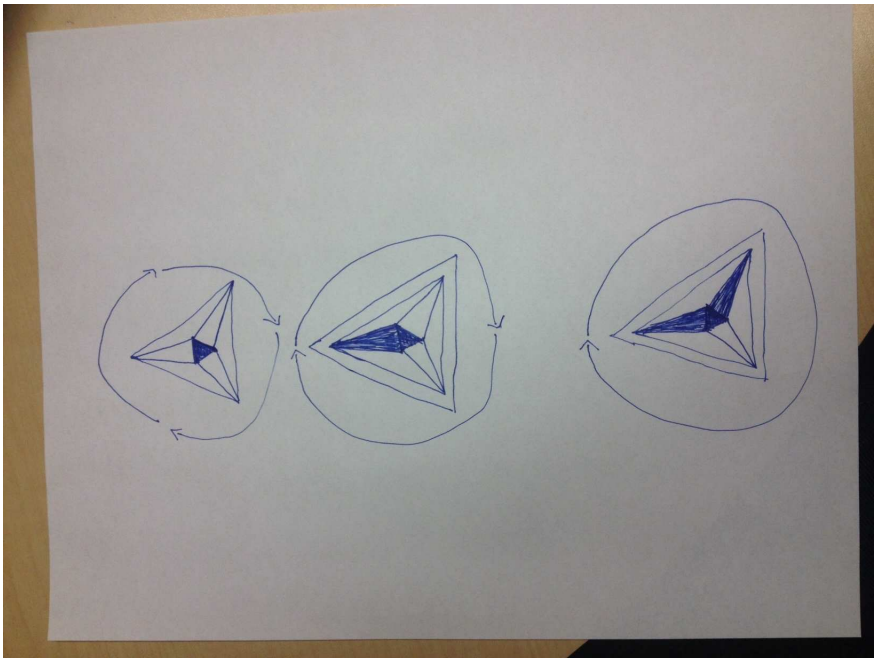

Figure 2.3: The stabilizer subgroups for various octahedron states.

Proof. By Corollary 1 and since $G.x \doteq [x] = [\hat{x}] \doteq G.\hat{x}$, the result follows.

$$\begin{aligned} r_x &= |G_x| && (2.1)\\ &= \frac{|G|}{|G.x|} && (2.2)\\ &= \frac{|G|}{|G.\hat{x}|} && (2.3)\\ &= |G_{\hat{x}}| && (2.4)\\ &= r_{\hat{x}} && (2.5) \end{aligned}$$

□

Since the building game is at its core an attachment model, we are interested in which intermediates can be formed from others by attaching a face to a particular intermediate.

Definition 7. *Two distinct intermediates* $[x]$ *and* $[y]$ *are* ***connected*** *if there exist*

states $x \in [x]$ and $y \in [y]$ such that one of the following holds:

- $\exists f \in y : y = x \cup \{f\}$
- $\exists f \in x : x = y \cup \{f\}$

Lemma 2. *If intermediates $[x]$ and $[y]$ are connected, then for every state $x \in [x]$ there is a state $y \in [y]$ such that $\exists f \in y : y = x \cup \{f\}$ or $\exists f \in x : x = y \cup \{f\}$.*

Proof. Without loss of generality, assume $|x| < |y|$. Since $[x]$ and $[y]$ are connected, let $\hat{x} \in [x]$, $\hat{y} \in [y]$, and $\hat{f} \in \hat{y}$, be such that $\hat{y} = \hat{x} \cup \{\hat{f}\}$. Then, for any $x \in [x]$, pick $g \in G$ such that $x = g.\hat{x}$. By choosing $y = g.\hat{y}$ and $\{f\} = g.\{\hat{f}\}$, we have

$$\begin{aligned} y &= g.\hat{y} && (2.6)\\ &= g.(\hat{x} \cup \{\hat{f}\}) && (2.7)\\ &= g.\hat{x} \cup g.\{\hat{f}\} && (2.8)\\ &= x \cup \{f\} && (2.9)\\ & && (2.10) \end{aligned}$$

and our result is shown. □

Definition 8. *A Building Game **pathway** is a sequence of intermediates $[x^{p_1}], [x^{p_2}], \ldots, [x^{p_N}]$ such that $[x^{p_i}]$ is connected to $[x^{p_{i+1}}]$, $|x^{p_i}| = i$, and $x^{p_N} = F$.*

Figure 2.4 shows a Building Game pathway for the octahedron using Schlegel diagrams. The pathway has 8 intermediates since there must be exactly one intermediate x^{p_i} satisfying $h\left(x^{p_i}\right) = i$ for each $i = 1, 2, \ldots, 8$.

With many pairs of connected intermediates, we organize these relations in a graph.

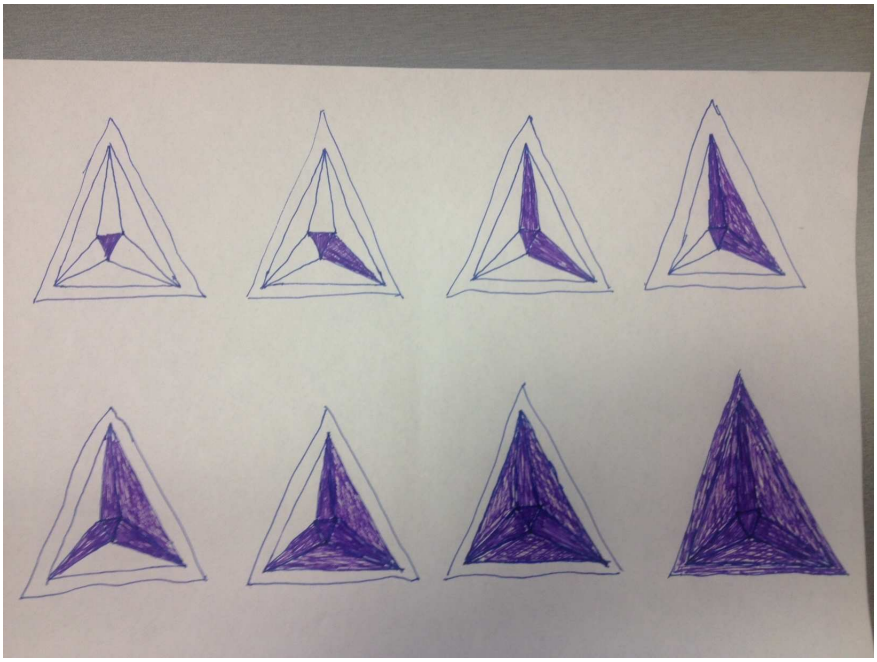

Figure 2.4: One Building Game pathway for the Octahedron.

Definition 9. *The Building Game* ***combinatorial configuration space*** *for a polyhedron is a graph in which the nodes are the polyhedron's intermediates and a graph edge exists between two intermediates if and only if they are connected.*

When the intermediates are partitioned by the number of faces they possess, it is natural to arrange the combinatorial configuration space into columns or tiers according to this partition. Figure 2.5 shows the Building Game state space for the cube. As seen, each column has intermediates with the same number of faces and connections thus exist with intermediates that are either in the tier directly above or below them. We can also see that there are three distinct pathways contained in the state space.

Interestingly, it is not the case that the addition or removal of each face of an intermediate results in a distinct intermediate.

Definition 10. *For two connected intermediates* $[x^j]$ *and* $[x^k]$*, the set of different*

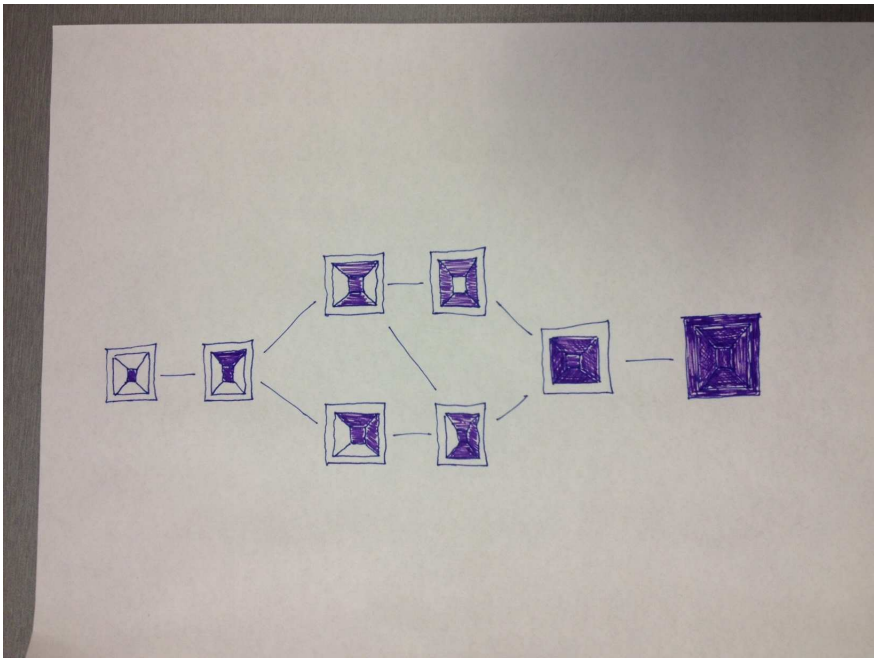

Figure 2.5: The Building Game combinatorial configuration space of the cube.

faces

$$F_{jk} \doteq \left\{ f \notin x^j : x^j \cup \{f\} \in [x^k] \right\} \cup \left\{ f \in x^j : x^j \setminus \{f\} \in [x^k] \right\}$$

that can be added or removed from x^j *to get an element of* $[x^k]$ *is called the* ***degeneracy set*** *and the number of such faces* $S_{jk} \doteq |F_{jk}|$ *is called the* ***degeneracy number****.*

The act of attaching an additional face is referred to as a forward step in the building game and the removal of a face is a backward step. As such, degeneracies are sometimes referred to as forward or backward degeneracies. It is important to note that in general the degeneracy number is not symmetric, i.e. $S_{jk} \neq S_{kj}$ for some connections $[x^j] \leftrightarrow [x^k]$ in the state space. Figure 2.6 depicts the forward and backward degeneracy numbers for a particular connection. As illustrated, there are two faces that can be added to the first intermediate to form the second. However, removing any of the three faces of the second intermediate will result in the first. Thus the forward degeneracy number is two and the backward degeneracy number

is three.

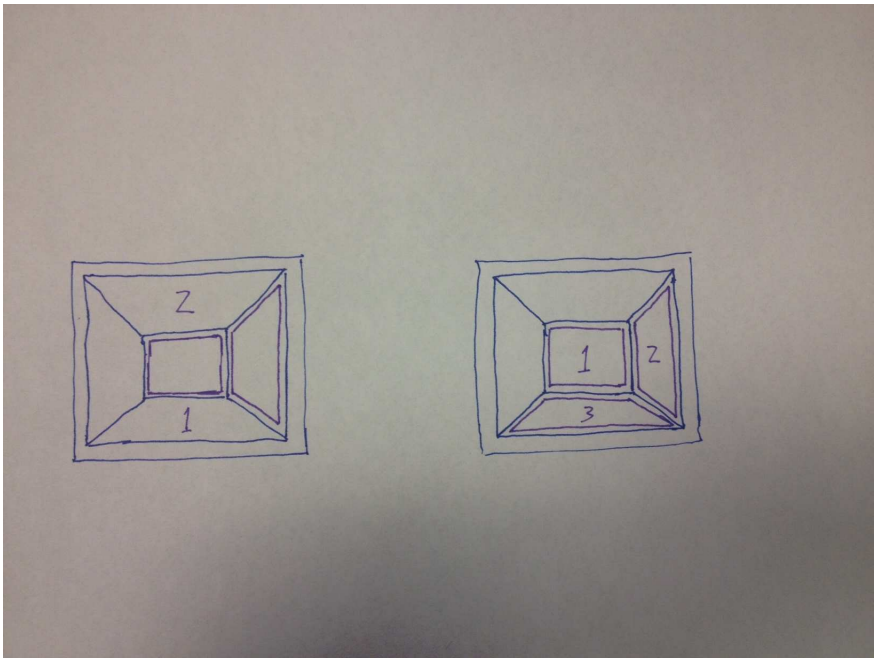

Figure 2.6: Degeneracies between two connected cube intermediates.

Finally, we extend the notion of a pathway to to be a sequence of intermediates of arbitrary lengths, such that the sequence begins at an intermediate with a single face and ends with the completed polyhedron and each pair of successive intermediates are connected.

Definition 11. *A Building Game* ***reversible pathway*** *is a sequence of intermediates* $[x^{p_1}], [x^{p_2}], \ldots, [x^{p_N}]$ *such that* $[x^{p_i}]$ *is connected to* $[x^{p_{i+1}}]$, $|x^{p_0}| = 1$, *and* $|x^{p_N}| = |F|$.

2.2.3 Group Theoretic Results

Lemma 3. *Let* $g \in G$ *and* $x, y \subset F$. *Then* $g.(x \cup y) = g.x \cup g.y$

Proof. Let $f \in g.(x \cup y)$. Then $g^{-1}.\{f\} \in x \cup y$. Without loss of generality, assume

$g^{-1}.\{f\} \in x$. It follows that $f \in g.x$ and $f \in g.x \cup g.y$ as well. Therefore, $g.(x \cup y) \subset g.x \cup g.y$.

Conversely, suppose $\hat{f} \in g.x \cup g.y$ and without loss pick $\hat{f}$ to be in $g.x$. It follows that $g^{-1}.\{\hat{f}\} \in x, x \cup y$ and subsequently $\hat{f} \in g.(x \cup y)$. From this we have the reverse relation $g.x \cup g.y \subset g.(x \cup y)$ which proves our equality. □

Lemma 4. *Let the Building Game intermediates $[x^j]$ and $[x^k]$ be connected in the combinatorial configuration space. Given $x^j \in [x^j]$, $x^k \in [x^k]$ and $f \in F$ such that $x^k = x^j \cup \{f\}$, the stabilizer subgroups $G_{x^j,\{f\}} = G_{x^k,\{f\}} = G_{x^j,x^k}$ are all equal.*

Proof.

$$
\begin{aligned}
G_{x^k,\{f\}} &= \left\{g \in G : g.x^k = x^k, g.\{f\} = \{f\}\right\} && (2.11)\\
&= \left\{g \in G : g.(x^j \cup \{f\}) = x^j \cup \{f\}, g.\{f\} = \{f\}\right\} && (2.12)\\
&= \left\{g \in G : g.x^j \cup g.\{f\} = x^j \cup \{f\}, g.\{f\} = \{f\}\right\} && (2.13)\\
&= \left\{g \in G : g.x^j \cup \{f\} = x^j \cup \{f\}, g.\{f\} = \{f\}\right\} && (2.14)\\
&= \left\{g \in G : g.x^j = x^j, g.\{f\} = \{f\}\right\} && (2.15)\\
&= G_{x^j,\{f\}} && (2.16)\\
&= \left\{g \in G : g.x^j = x^j, g.\{f\} = \{f\}\right\} && (2.17)\\
&= \left\{g \in G : g.x^j = x^j, g.x^j \cup g.\{f\} = x^j \cup \{f\}\right\} && (2.18)\\
&= \left\{g \in G : g.x^j = x^j, g.(x^j \cup \{f\}) = x^j \cup \{f\}\right\} && (2.19)\\
&= \left\{g \in G : g.x^j = x^j, g.x^k = x^k\right\} && (2.20)\\
&= G_{x^j,x^k} && (2.21)
\end{aligned}
$$

□

Lemma 5. *For connected intermediates $[x]$ and $[y]$, with $x \cup \{f\} = y$ and $x \cup \{\hat{f}\} \doteq$*

$\hat{y} \in [y]$, the stabilizer subgroups $|G_{x,y}|$ and $|G_{x,\hat{y}}|$ satisfy $|G_{x,y}| = |G_{x,\hat{y}}|$.

Proof. If there is an $h \in G_x$ such that $\{f\} = h.\{\hat{f}\}$, the result follows similar to lemma XXX as $|G_{x,y}| = |hG_{x,\hat{y}}h^{-1}| = |G_{x,\hat{y}}|$.

However, if $\{f\} \neq h.\{\hat{f}\}$ for every $h \in G_x$, the result STILL NEEDS PROOF. In the cases we consider, the lemma has been confirmed to be true through brute force computation.

□

Lemma 6. *Let the Building Game intermediates $[x^j]$ and $[x^k]$ be connected in the combinatorial configuration space, the number of orbits $|F_{jk}/G_{x^j}| = |F_{kj}/G_{x^k}|$.*

Proof. Let $G_{x^j}.\{f\} \in F_{jk}/G_{x^j}$. Then, since $f \in F_{jk}$ we know that $x^j \cup \{f\} \in [x^k]$ and thus there is a $g \in G$ such that $g.(x^j \cup \{f\}) = x^k$. Since $g.x^j \in [x^j]$, we know that $g.\{f\} \in F_{kj}$. Then, for any $h \in G_{x^k}$ we see that $hg.(x^j \cup \{f\}) = h.x^k = x^k$. This means that $G_{x^k}.(g.\{f\}) \in F_{kj}/G_{x^k}$.

Now, we similarly assume $G_{x^k}.\{\hat{f}\} \in F_{kj}/G_{x^k}$. Then, $x^k \setminus \{\hat{f}\} \in [x^j]$ and there is a $\hat{g} \in G$ such that $\hat{g}.(x^k \setminus \{\hat{f}\}) = x^j$. Therefore, $\hat{g}.\{\hat{f}\} \in F_{jk}$ as $\hat{g}.x^k \in [x^k]$. So, for every $\hat{h} \in G_{x^j}$ we get $\hat{h}\hat{g}.(x^k \setminus \{\hat{f}\}) = \hat{h}.x^j = x^j$. Thus $G_{x^j}.(\hat{g}.\{\hat{f}\}) \in F_{jk}/G_{x^j}$.

Since we have shown that for every orbit in F_{jk}/G_{x^j} there is a corresponding orbit in $F_{kj}/G_{x^k}|$ and vice versa, the total number of orbits in each set must be the same and our result $|F_{jk}/G_{x^j}| = |F_{kj}/G_{x^k}|$ holds. □

Theorem 4. *For two Building Game intermediates $[x^j]$ and $[x^k]$ are connected in the combinatorial configuration space, $r_k S_{jk} = r_j S_{kj}$.*

Proof. Without loss of generality, assume that $[x^j]$ has one fewer face than $[x^k]$ and pick $x^j \in [x^j]$, $x^k \in [x^k]$ and $f \in F$ such that $x^k = x^j \cup \{f\}$. Then by Burnside lemma, we have the following [1].

$$|F_{jk}/G_{x^j}| = \frac{1}{|G_{x^j}|} \sum_{g \in G_{x^j}} |(F_{jk})^g| \tag{2.22}$$

$$= \frac{1}{|G_{x^j}|} \sum_{\hat{f} \in F_{jk}} |G_{x^j,\{\hat{f}\}}| \tag{2.23}$$

$$= \frac{|G_{x^j,\{f\}}|}{r_j} \sum_{\hat{f} \in F_{jk}} 1 \tag{2.24}$$

$$= \frac{|G_{x^j,\{f\}}| S_{jk}}{r_j} \tag{2.25}$$

Then, by lemmas 5 and 6 we have,

$$\frac{r_j}{S_{jk}} = \frac{|G_{x^j,\{f\}}|}{|F_{jk}/G_{x^j}|} \tag{2.26}$$

$$= \frac{|G_{x^k,\{f\}}|}{|F_{kj}/G_{x^k}|} \tag{2.27}$$

$$\doteq \frac{r_k}{S_{kj}} \tag{2.28}$$

and the result $r_k S_{jk} = r_j S_{kj}$ follows. □

2.3 Stochastic Modeling Results

Since the Building Game is a sequential process with several choices at each step, it is natural to consider it as a stochastic process. If we define a Building Game process that allows faces to be sequentially added or removed in a reversible way, the process consists of transitions from intermediate to intermediate along connections in the combinatorial configuration space. By specifying a distribution on these transitions,

it will induce a stationary distribution on the state space and provide a framework for compute relevant statistics such as expected formation times.

We define the Markov process X_t by the transition rate matrix Q, with the heuristic that the rate of transition to an intermediate $[x^k]$ from an intermediate $[x^j]$ should be proportional to the number of faces that can be added or removed from $[x^j]$ to reach $[x^k]$. For this reason, we include the degeneracy number S_{jk} as a factor in the transition rate matrix. Furthermore, we model the process using an energetic interpretation in which each intermediate has an energy and to transition between intermediates, an energy barrier $E_{jk} = E_{kj}$ must be overcome.

$$Q_{jk} = \begin{cases} S_{jk} e^{-\beta\left(E_{jk} - E_j\right)} & \text{if } [x^j] \leftrightarrow [x^k] \\ -z_j & \text{if } j = k \\ 0 & \text{else} \end{cases} \tag{2.29}$$

Here, $z_j \doteq \sum_{\ell:\ell\neq j} S_{j\ell} e^{-\beta\left(E_{j\ell} - E_j\right)}$ is the rate at which the process leaves x^j.

2.3.1 Stationary Distribution

Theorem 5. *If the transition rate matrix Q can be decomposed as $Q = DC$ where D is diagonal with each entry of the diagonal positive and C is a symmetric matrix with the non-diagonal entries $C_{jk} > 0$ if and only if $[x^j]$ and $[x^k]$ are connected and $C_{jk} = 0$ if they are not, then X_t has the unique stationary distribution $\pi = \mathrm{diag}\,(D^{-1})$.*

Proof. First, we show Q and π satisfy detailed balance.

$$\pi_j Q_{jk} = \left(\frac{1}{D_{jj}}\right)(D_{jj}C_{jk}) \tag{2.30}$$

$$= C_{jk} \tag{2.31}$$

$$= C_{kj} \tag{2.32}$$

$$= \left(\frac{1}{D_{kk}}\right)(D_{kk}C_{kj}) \tag{2.33}$$

$$= \pi_k Q_{kj} \tag{2.34}$$

Since the combinatorial state space is connected and $Q_{jk}, Q_{kj} > 0$ for all connected intermediates $[x^j]$ and $[x^k]$, the process it trivially aperiodic and positive recurrent. CITE □

Theorem 6. *The Markov process X_t defined by the transition rate matrix Q in equation 2.29 admits the unique stationary distribution $\frac{1}{zr_j}e^{-\beta E_j}$ where $z \doteq \sum_\ell \frac{1}{r_\ell}e^{-\beta E_\ell}$ is the partition function.*

Proof. We take $C_{jk} \doteq \frac{S_{jk}}{zr_j}e^{-\beta E_{jk}}$ and notice that it is symmetric by theorem 4. With $D_{jj} \doteq zr_j e^{\beta E_j}$ we have our partition.

$$Q_{jk} = S_{jk}e^{-\beta\left(E_{jk}-E_j\right)} \tag{2.35}$$

$$= \left(zr_j e^{\beta E_j}\right)\left(\frac{S_{jk}}{zr_j}e^{-\beta E_{jk}}\right) \tag{2.36}$$

$$= D_{jj}C_{jk} \tag{2.37}$$

Thus, by theorem 5, $\pi_j = \frac{1}{D_{jj}} = \frac{1}{zr_j}e^{-\beta E_j}$. □

2.3.2 Hitting Times

While the stationary distribution is an important piece in understanding the nature of our Markov process, other statistics which describe the dynamics are also useful. For instance, we may be interested in the expected time it will take the process to travel from an intermediate $[x^j]$ to a specific subset A of the combinatorial configuration space. Equation 2.38 provides a mathematical definition for such a stopping time.

$$\tau_j^A \doteq \inf\left\{t \geq 0 : X_t \in A, X_0 = x^j\right\} \tag{2.38}$$

Oftentimes, we choose A to be the intermediate of the fully completed polyhedron. This means the the stopping time will represent the expected formation time from a given intermediate.

Since we are looking at events in which the process first reaches a particular set of intermediates, it is helpful to use X_t's discrete time partner process Y_n which simply records the sequence of intermediates that X_t passes through. Here we formally define Y_n in a recursive manner, such that it is the first intermediate visited after the process X_t leaves Y_{n-1}.

$$Y_n \doteq \arg\min_{y \neq X_t}\{s : s > t, X_s = y, X_t = Y_{n-1}\} \tag{2.39}$$

$$\sim X_{\tau_{Y_{n-1}}^{\{Y_{n-1}\}^C}} \tag{2.40}$$

Using first step analysis, we derive the expected hitting time τ_j^A. Consider the

case of $j \notin A$.

$$E\left[\tau_j^A\right] = E\left[E\left[\tau_j^A | Y_1\right]\right] \tag{2.41}$$

$$= E\left[Exp\left(z_j\right) + \tau_{Y_1}^A\right] \tag{2.42}$$

$$= \frac{1}{z_j} + E\left[\sum_k \tau_{Y_1}^A \mathbb{1}_{Y_1=k}\right] \tag{2.43}$$

$$= \frac{1}{z_j} + \sum_{k:k\neq j} E\left[\tau_k^A\right] P\left(Y_1 = k\right) \tag{2.44}$$

$$= \frac{1}{z_j}\left(1 + \sum_{k:k\neq j} Q_{jk} E\left[\tau_k^A\right]\right) \tag{2.45}$$

$$1 = -\sum_k Q_{jk} E\left[\tau_k^A\right] \tag{2.46}$$

$$\tag{2.47}$$

Now, clearly for $j \in A$ we trivially have a stopping time of zero:

$$E\left[\tau_j^A\right] = 0. \tag{2.48}$$

$$\tag{2.49}$$

Putting together the two case, we write the solution as a linear system with τ^A the vector of stopping times with each possible initial intermediate.

$$\left(\text{diag}\left(\mathbb{1}_A\right) - \text{diag}\left(\mathbb{1}_{A^c}\right) Q\right) E\left[\tau^A\right] = \mathbb{1}_{A^c} \tag{2.50}$$

$$\tag{2.51}$$

Thus, we have

$$E\left[\tau^A\right] = \left[\left(\text{diag}\left(\mathbb{1}_A\right) - \text{diag}\left(\mathbb{1}_{A^c}\right) Q\right)\right]^{-1} \mathbb{1}_{A^c}. \tag{2.52}$$

$$\tag{2.53}$$

which contains the particular value $E[\tau_j^A]$ that we are interested in.

Further, we can also compute the exact distribution of the stopping times τ^A. First, we define the CDF as follows.

$$\psi_j^A(t) \doteq P\left(\tau_j^A \leq t\right) \tag{2.54}$$

$$\psi_j^A(0) = \mathbb{1}_{j \in A} \tag{2.55}$$

$$\psi_j^A(t) = 0 \ \forall j \in A \tag{2.56}$$

Thus, for the case of $j \notin A$ we find

$$\psi_j^A(t) \doteq P\left(\tau_j^A \leq t\right) \tag{2.57}$$

$$= \sum_k P\left(\tau_j^A \leq t | Y_1 = x^k\right) P\left(Y_1 = x^k\right) \tag{2.58}$$

$$= \frac{1}{z_j} \sum_{k:k \neq j} Q_{jk} P\left(Exp(z_j) + \tau_k^A \leq t\right) \tag{2.59}$$

$$= \frac{1}{z_j} \sum_{k:k \neq j} Q_{jk} \int_0^t P\left(\tau_k^A \leq t - s\right) z_j e^{-z_j s} ds \tag{2.60}$$

$$= \sum_{k:k \neq j} Q_{jk} \int_0^t \psi_k^A(t-s) e^{-z_j s} ds \tag{2.61}$$

$$= \sum_{k:k \neq j} Q_{jk} \int_0^t \psi_k^A(r) e^{-z_j (t-r)} dr \tag{2.62}$$

$$e^{z_j t} \psi_j^A(t) = \sum_{k:k \neq j} Q_{jk} \int_0^t e^{z_j r} \psi_k^A(r) dr \tag{2.63}$$

$$e^{z_j t} \frac{d\psi_j^A}{dt} + z_j e^{z_j t} \psi_j^A(t) = \sum_{k:k \neq j} q_{jk} e^{z_j t} \psi_k^A(t) \tag{2.64}$$

$$\frac{d\psi_j^A}{dt} = \sum_k q_{jk} \psi_k^A(t). \tag{2.65}$$

Combining both cases, we get the linear system and solution.

$$\frac{d\psi^A}{dt} = \operatorname{diag}\left(\mathbb{1}_{A^c}\right) Q\psi^A \tag{2.66}$$

$$\psi^A\left(0\right) = \mathbb{1}_A \tag{2.67}$$

$$\psi^A\left(t\right) = e^{\operatorname{diag}(\mathbb{1}_{A^c})Qt}\mathbb{1}_A \tag{2.68}$$

$$\tag{2.69}$$

This is the solution for the CDF of the stopping time τ^A, but we can also compute the PDF explicitly for $t > 0$.

$$p\left(\tau^A = t\right) = \frac{d\psi^A}{dt} \tag{2.70}$$

$$= \operatorname{diag}\left(\mathbb{1}_{A^c}\right) Q\psi^A \tag{2.71}$$

Another hitting time statistic we may be interested in is, given an initial intermediate, what is the probability we will hit a subset of intermediates A before some other disjoint subset of intermediates B.

$$\rho_j^{A,B} \doteq P(\tau_j^A < \tau^B) \tag{2.72}$$

Trivially, we have

$$\rho_j^{A,B} = 1 \text{ if } j \in A \tag{2.73}$$

$$\rho_j^{A,B} = 0 \text{ if } j \in B \tag{2.74}$$

but must compute the case of $j \in (A \cup B)^C$.

$$\rho_j^{A,B} = P(\tau_j^A < \tau_j^B) \tag{2.75}$$

$$= \sum_k P(\tau_j^A < \tau_j^B, Y_1 = k) \tag{2.76}$$

$$= \sum_k P(\tau_j^A < \tau_j^B | Y_1 = k) P(Y_1 = k) \tag{2.77}$$

$$= \sum_k P(\tau_k^A < \tau_k^B) P(Y_1 = k) \tag{2.78}$$

$$= \frac{1}{z_j} \sum_{k \neq j} \rho_k^{A,B} Q_{jk} \tag{2.79}$$

$$0 = Q\rho^{A,B} \tag{2.80}$$

Again, putting each of these cases together, we get the linear system

$$\left(\text{diag}\left(\mathbb{1}_A\right) + \text{diag}\left(\mathbb{1}_B\right) + \text{diag}\left(\mathbb{1}_{(A\cup B)^c}\right) Q\right) \rho^{A,B} = \mathbb{1}_A \tag{2.81}$$

and our solution is

$$\rho^{A,B} = \left[\text{diag}\left(\mathbb{1}_A\right) + \text{diag}\left(\mathbb{1}_B\right) + \text{diag}\left(\mathbb{1}_{(A\cup B)^c}\right) Q\right]^{-1} \mathbb{1}_A \tag{2.82}$$

Using this result, an insightful choice for A and B would be $A = \{x_k\}$ and $B = x_{|F|}$. Then, the value of $\rho_1^{A,B}$ would correspond to the probability that a particular intermediate x^k is in a reversible pathway between the intermediate with a single face and the completed polyhedron.

Chapter Three

The Building Game: Enumeration

3.1 Known Enumerative Results

When treating the Building Game for a polyhedron as a stochastic process, we must first know what the the combinatorial configuration space is exactly. This is a computational enumeration problem, but the results of this enumeration are also of mathematical interest outside its stochastic use.

Attachment models like the Building Game have been the topic of much research in combinatorial mathematics. A well known example of this is the study of polyominoes. An n-omino is a configuration of n attached squares in a 2-dimensional lattice. As seen in figure 3.1, the classic video game Tetris uses each of the seven rotationally unique tetrominoes ($n = 4$) as game pieces. The enumeration of n-ominoes has been extensively studied and there are many enumeration results, both explicit and asymptotic. As in our case, the general goal is to enumerate the polyominoes that are unique when acted on by some group (rotation, reflection, etc) or under certain constraints which are often topological.

Figure 3.1: The seven tetrominoes used in Tetris.

The Building Game has a wealth of combinatorial enumeration problems to consider. Some have been addressed in the work of Zlotnick et all CITE. The original Building Game paper only considers a single pathway and thus is not concerned with enumeration of the entire combinatorial configuration space. However in Endres et al, the number of intermediates for each of the Platonic solids is computed. The entire combinatorial configuration space for the dodecahedron and icosahedron is also illustrated CITE.

Before the work of Endres et al, David Wilson enumerated the number of interme-

diates composed of each number of faces for the icosahedron CITE. This computation seems to have occurred independent of the scientific context of self-assembly as Wilson's enumerations are referred to as the "Number of one-sided triangular n-ominoes (or triominoes) on the icosahedron." He also includes variants such as the "Number of triangular n-ominoes on the icosahedron" which enlarges the equivalence classes to identify intermediate that are reflections of each other.

3.2 New Enumerative Results

Previous enumeration results for the Building Game have focused on the number of intermediates of the Platonic solids. We extend these results to also count the number of Building Game connections and pathways in the combinatorial configuration space. Additionally, we consider the Platonic, Archimedean, and Catalan solids classes with these results presented in figure 3.2.

As we consider polyhedra with more and more faces, there is a combinatorial explosion in the number intermediates in combinatorial configuration space. This was noted in Endres et al. CITE, where the dodecahedron has 73 intermediates and the icosahedron has $2,649$. We have enumerated the intermediates for polyhedra of up to 30 faces in our three classes of polyhedra. The 30-faced rhombic triacontahdron has the most intermediates of all polyhedra in our computation with $2,423,212$.

–Scatter plot commentary.

Polyhedra Name	$\|F\|$	Intermediates	Connections	Pathways
Tetrahedron	4	4	3	1
Cube	6	8	9	3
Octahedron	8	14	21	14
Dodecahedron	12	73	263	17,696
Icosahedron	20	2,649	17,241	57,396,146,640
Truncated Tetrahedron	8	28	63	402
Cuboctahedron	14	340	1,634	10,170,968
Truncated Cube	14	499	2,729	101,443,338
Truncated Octahedron	14	555	3,069	68,106,377
Rhombicuboctahedron	26	638,850	6,459,801	164,068,345,221,515,292,308
Truncated Cuboctahedron	26	1,525,658	17,672,374	13,837,219,462,483,379,105,902
Triakis Tetrahedron	12	98	318	38,938
Rhombic Dodecahedron	12	127	493	76,936
Triakis Octahedron	24	12,748	81,296	169,402,670,046,670
Tetrakis Hexahedron	24	50,767	394,377	4,253,948,297,210,346
Deltoidal Icositetrahedron	24	209,675	1,989,548	418,663,242,727,526,726
Pentagonal Icositetrahedron	24	345,938	3,544,987	2,828,128,000,716,774,492
Rhombic Triacontahedron	30	2,423,212	26,823,095	161,598,744,916,797,017,978,128

Figure 3.2: Building game combinatorial configuration space enumerative results for the Platonic, Archimedean, and Catalan solids.

Figure 3.3: The relation between number of faces and intermediates in the Building Game combinatorial configuration space.

3.2.1 Shellability

The theory of polytopes provides a mathematical method for constructing a polytope called a *shelling*. The process, similar to the building game, involves sequentially adding facets of the polytope until all of the facets are added. However, the rules as to which facets may be added at each step of the process are a bit more restrictive. Ziegler provides the following definitions of a shelling and a shellable polytopal complex CITE.

Definition 12. *Let $\mathcal{C}$ be a pure k-dimensional polytopal complex $\mathcal{C}$. A **shelling** of $\mathcal{C}$ is a linear ordering $F_1, F_2, \ldots, F_s$ of the facets of $\mathcal{C}$ such that either $\mathcal{C}$ is 0-dimensional (and thus the facets are points), or it satisfies the following conditions:*

(i) The boundary complex of the first facet F_1 has a shelling.

Figure 3.4: The relation between number of faces and connections in the Building Game combinatorial configuration space.

Figure 3.5: The relation between number of faces and paths in the Building Game combinatorial configuration space.

(ii) For $1 < j \leq s$ the intersection of the facet F_j with the previous facets is non empty and is a beginning segment of a shelling of the $(k-1)$-dimensional boundary complex of F_j, that is,

$$F_j \cap \left(\bigcup_{i=1}^{j-1} F_i \right) = G_1 \cup G_2 \cup \cdots \cup G_r$$

for some shelling $G_1, G_2, \ldots, G_r, \ldots, G_t$ of $\mathcal{C}(\partial F_j)$, and $1 \leq r \leq t$. (In particular, this requires that $F_j \cap (\bigcup_{i=1}^{j-1} F_i)$ has a shelling, so it has to be a pure $(k-1)$-dimensional, and connected for $k > 1$.)

Definition 13. *A polytopal complex is **shellable** if it is pure and has a shelling.*

Since we are only concerned with 3-dimensional polyhedra, the corresponding polytopal complex simply consists of the faces, edges, and vertices of the polyhedron. The facets of the complex are just the faces.

Definition 14. *A Building Game intermediate x is called a **shellable intermediate** if for every state $x \in [x]$ there is a linear ordering $f_1, f_2, \ldots, f_{|x|}$ on the faces of x such that this ordering is a beginning of a shelling $f_1, f_2, \ldots, f_{|x|}, \ldots f_{|F|}$ of the polytopal complex $F \cup E \cup V$.*

Lemma 7. *The polytopal complex consisting of the edges and vertices of a polygonal face is shellable.*

Proof. We see that condition (i) from definition 12 is satisfied since the boundary complex of each edge is 0-dimensional.

Condition (*ii*) is also satisfied since each newly attached edge and the existing partial shelling is a subset of the attached edge's vertices. Since the vertices of the attached edge are 0-dimensional, any linear order of its vertices is a shelling. □

Definition 15. *A Building Game connection is said to be a* ***shellable connection*** *if the two connected intermediates are both shellable and a Building Game pathway is said to be a* ***shellable pathway*** *if each of its intermediates are shellable.*

Figure 3.6 details the shellability statistic for the Platonic, Archimedean, and Catalan solids classes. Because of the added shellability restriction, these count statistics are lower than in the general case, but the combinatorial growth as the number of faces increases is similar.

Polyhedra Name	$\|F\|$	Intermediates	Connections	Pathways
Tetrahedron	4	4	5	1
Cube	6	7	7	2
Octahedron	8	11	13	4
Dodecahedron	12	52	155	2,166
Icosahedron	20	469	1,985	105,999,738
Truncated Tetrahedron	8	21	40	174
Cuboctahedron	14	136	468	477,776
Truncated Cube	14	247	1,000	5,232,294
Truncated Octahedron	14	342	1,464	5,704,138
Rhombicuboctahedron	26	70,887	462,721	64,308,526,503,247,584
Truncated Cuboctahedron	26	515,335	4,070,813	13,890,723,216,176,694,816
Triakis Tetrahedron	12	48	115	5,012
Rhombic Dodecahedron	12	67	195	6,258
Triakis Octahedron	24	1,021	4,237	210,459,770,300
Tetrakis Hexahedron	24	4,224	21,125	5,894,431,702,846
Deltoidal Icositetrahedron	24	33,046	208,317	703,619,122,996,096
Pentagonal Icositetrahedron	24	95,326	657,013	7,572,459,719,248,765
Rhombic Triacontahedron	30	97,741	702,219	7,057,239,571,753,327,764

Figure 3.6: Building game enumerative shellability results for the Platonic, Archimedean, and Catalan solids.

Figure 3.7: The relation between number of faces and shellable intermediates in the Building Game combinatorial configuration space.

Figure 3.8: The relation between number of faces and shellable connections in the Building Game combinatorial configuration space.

Figure 3.9: The relation between number of faces and shellable paths in the Building Game combinatorial configuration space.

3.2.2 Shelling Enumeration

To our knowledge, the enumeration of the number of shellings of the polyhedra in the Platonic, Archimedean, and Catalan solid classes remains an open problem. Here we present these enumerations for the polyhedra of up to 30 faces. The concept of a shelling is similar to to that of a pathway. Using the structure of each computed combinatorial configuration space, a method for counting the number of shellings of a polyhedron is derived.

Theorem 7. *The total number of shellings for a polyhedral complex generated by a polyhedron is*

$$\#(shellings) = \sum_{p \in \{shellpaths\}} |[x^{p_1}]| \prod_{j}^{|F|-1} S_{p_j p_{(j+1)}}. \tag{3.1}$$

Proof.

$$\#(shellings) = \sum_{f_1 \in F} \mathbb{1}_{sh(\emptyset, f_1)} \cdots \sum_{f_k \in F \setminus \{f_1, \ldots, f_{k-1}\}} \mathbb{1}_{sh(\{f_1, \ldots, f_{k-1}\}, f_k)} \cdots \sum_{f_{|F|} \in F \setminus \{f_1, f_2, \ldots, f_{|F|-1}\}} \mathbb{1}_{sh(\{f_1, f_2, \ldots, f_{|F|-1}\}, f_{|F|})} \tag{3.2}$$

$$= \sum_{f_1 \in F} \mathbb{1}_{sh(\emptyset, f_1)} \cdots \sum_{f_k \in F \setminus \{f_1, \ldots, f_{k-1}\}} \mathbb{1}_{sh(\{f_1, \ldots, f_{k-1}\}, f_k)} \cdots \sum_{x^{p_{|F|}} : [x^{p_{|F|-1}}] \xrightarrow{shell} [x^{p_{|F|}}]} S_{p_{|F|-1} p_{|F|}} \tag{3.3}$$

$$= \sum_{f_1 \in F} \sum_{x^{p_2} : [x^{p_1}] \xrightarrow{shell} [x^{p_2}]} S_{p_1 p_2} \cdots \sum_{x^{p_{|F|}} : [x^{p_{|F|-1}}] \xrightarrow{shell} [x^{p_{|F|}}]} S_{p_{|F|-1} p_{|F|}} \tag{3.4}$$

$$= \sum_{[x^{p_1}} |[x^{p_1}]| \sum_{x^{p_2} : [x^{p_1}] \xrightarrow{shell} [x^{p_2}]} S_{p_1 p_2} \cdots \sum_{x^{p_{|F|}} : [x^{p_{|F|-1}}] \xrightarrow{shell} [x^{p_{|F|}}]} S_{p_{|F|-1} p_{|F|}} \tag{3.5}$$

$$= \sum_{p \in \{shellpaths\}} |[x^{p_1}]| \prod_{j}^{|F|-1} S_{p_j p_{(j+1)}} \tag{3.6}$$

□

Using dynamic programming we can compute the number of shellings explicitly without having to explicitly consider each pathway individually. Define the quantity a_{ik} as follows.

$$a_{i,k} \doteq \sum_{\substack{\text{shellable subpaths:} \\ p_1, \ldots, p_k \\ x^{p_k} \in [x^i]}} |G.x^{p_1}| \prod_{j=1}^{k-1} S_{p_j p_{j+1}} \tag{3.7}$$

Using this definition, we first note that the number of shellings, which is the quantity of interest, is equal to $a_{N,|F|}$ where $x^N = F$. Now, we set up the following recursion

that will be the basis for our computation.

$$a_{i,k} \doteq \sum_{\substack{\text{shellable subpaths:} \\ p_1,\ldots,p_k \\ x^{p_k} \in [x^i]}} |G.x^{p_1}| \prod_{j=1}^{k-1} S_{p_j p_{j+1}} \tag{3.8}$$

$$= \sum_{[x^\ell]:[x^\ell] \xrightarrow{shell} [x^i]} \sum_{\substack{\text{shellable subpaths:} \\ p_1,\ldots,p_{k-1} \\ x^{p_{k-1}} \in [x^\ell]}} |G.x^{p_1}| S_{\ell i} \prod_{j=1}^{k-2} S_{p_j p_{j+1}} \tag{3.9}$$

$$= \sum_{[x^\ell]:[x^\ell] \xrightarrow{shell} [x^i]} S_{\ell i} \sum_{\substack{\text{shellable subpaths:} \\ p_1,\ldots,p_{k-1} \\ x^{p_{k-1}} \in [x^\ell]}} |G.x^{p_1}| \prod_{j=1}^{k-2} S_{p_j p_{j+1}} \tag{3.10}$$

$$= \sum_{[x^\ell]:[x^\ell] \xrightarrow{shell} [x^i]} S_{\ell i} a_{\ell,k-1} \tag{3.11}$$

Thus, using the base cases $a_{j,1} = |G.x^j|$ for each single faced intermediate $[x^j]$, we can use this relation to recursively solve for the number of shellings $a_{N,|F|}$.

Polyhedra Name	$\lvert F\rvert$	Shellings
Tetrahedron	4	24
Cube	6	480
Octahedron	8	4,224
Dodecahedron	12	19,041,600
Icosahedron	20	1,417,229,099,520
Truncated Tetrahedron	8	9,216
Cuboctahedron	14	113,055,744
Truncated Cube	14	654,801,408
Truncated Octahedron	14	937,087,104
Rhombicuboctahedron	26	4,728,400,467,971,102,208
Truncated Cuboctahedron	26	688,499,026,944,479,645,952
Triakis Tetrahedron	12	587,040
Rhombic Dodecahedron	12	5,836,800
Triakis Octahedron	24	66,063,419,534,592
Tetrakis Hexahedron	24	1,389,323,257,015,296
Deltoidal Icositetrahedron	24	125,987,819,253,281,472
Pentagonal Icositetrahedron	24	1,144,572,832,023,047,616
Rhombic Triacontahedron	30	15,574,782,555,813,226,074,240

Figure 3.10: Number of Shellings for the Platonic, Archimedean, and Catalan solids of up to 30 faces.

3.2.3 Bounds and Asymptotics

There is a clear relation between the number of faces in a polyhedron and then number of intermediates it has. However, that relationship also greatly depends on the polyhedral symmetry group. For instance, if you have a polyhedron with a small number of faces and a trivial rotation group consisting only of the identity, every edge-connected subset of the polyhedron's faces will be a distinct intermediate. In aggregate, this may mean that the polyhedron has more intermediates than another polyhedron with more faces, yet a larger symmetry group.

An upper bound on the number of intermediates is possible using the theory of group actions. Consider the set of all subsets 2^F of a polyhedron with rotation group G. Trivially $|2^F/G|$ is an upper bound on the number of intermediates since it simply relaxes the connectivity requirement for a subset to be a building game state. Using Burnside's lemma, we see that

$$|2^F/G| = \frac{1}{|G|}\sum_{g\in G}|(2^F)^g| \tag{3.12}$$

$$> \frac{|(2^F)^e|}{|G|} \tag{3.13}$$

$$= \frac{|2^F|}{|G|} \tag{3.14}$$

$$= \frac{2^{|F|}}{|G|} \tag{3.15}$$

which is not a particularly good bound in practice. The precise value of $|2^F/G|$ is calculable with minimal computer assistance, but details of this computation are omitted due to its relative uselessness. For the cube, the bound is fairly tight, only including the two non-intermediates corresponding to the empty subset of faces, and the non-connected subset consisting of the top and bottom faces. Thus the cube has

the bound $|2^F/G| = 10 \geq 8$. In the case of the tetrahedron, the only over-counted subset of faces is the empty one and the bound is $|2^F/G| = 5 \geq 4$. However, in the case of the icosahedron we have $|2^F/G| \geq \frac{2^{20}}{60} \approx 17476.3 \gg 2649$. Here we use the approximate bound $\frac{2^{|F|}}{|G|}$ which is the largely dominant term in the sum from equation 3.12.

We can get a similar bound on the number of intermediates with a particular number of faces,

$$|\{x \in 2^F : |x| = k\}/G| = \frac{1}{|G|} \sum_{g \in G} |\{x \in 2^F : |x| = k\}^g| \tag{3.16}$$

$$> \frac{|\{x \in 2^F : |x| = k\}^e|}{|G|} \tag{3.17}$$

$$= \frac{|\{x \in 2^F : |x| = k\}|}{|G|} \tag{3.18}$$

$$= \frac{\binom{|F|}{k}}{|G|} \tag{3.19}$$

but again, this is not particularly useful, especially for intermediates with around $\frac{1}{2}|F|$ faces.

Since the building game is similar in spirit to polyomino enumeration, one might try to assimilate some the techniques used for polyominoes. For example, through fairly simple arguments, one can show that $s_m s_n \leq s_{m+n}$ where s_m is the number of unique polyominoes with m subunits CITE. This leads to the bound $s_m \leq (const)^m$. Trying to set up such a relation in the building game is sounds initially appealing, but there is a fundamental difference between the two growth models that makes this approach futile. In the polyomino case, there is no limit to the number of subunits that can be considered. Importantly, this is not the case for the building game since an intermediate can only have $|F|$ faces at most. Thus any such recurrence relation

for the building game will result in a good upper bound for the intermediates with a small number of faces at best.

The formulation of meaningful bounds for the number of building game intermediates with k faces remains an open problem, especially for $k \sim \frac{1}{2}|F|$. At the root of the problem is the difficulty in mathematically describing the subsets of F are edge connected. Future approaches may incorporate enumeration results for connected subgraphs or Hamiltonian paths since these topics explicitly acknowledge connectedness properties.

From looking at the statistics on number of faces $|F|$ of a polyhedron and the number of intermediates in its combinatorial configuration space, it is natural to want to make statements about the asymptotic growth of the combinatorial configuration space's size. Unfortunately, when formed in this way, the problem is ill-posed. To discuss asymptotics, we must first specify an infinite class of polyhedra. The Platonic, Archimedean, and Catalan Solid classes that we've worked with thus far are all finite though, so other choices must be considered. One option is to take an existing polyhedron in one of these classes and create an infinite family by describing finer and finer tilings on top of the polyhedron's faces. If designed carefully each member of the tiled polyhedron family will have the same symmetry group, even as the number of faces grows.

Similar to polyhedra with tiled faces are the icosahedral viral capsids indexed by T-number. This number is related to the number of protein subunits in the virus. When each subunit is idealized as a polygon, a T-capsid will consist of 12 pentagons and $10(T-1)$ hexagons. Interestingly, this makes most of the icosahedral viral capsid equivalent to the dual of an icosahedron with each face consisting of T triangular tiles. This would certainly be an interesting an relevant family of polyhedra to

consider, though we leave it as an open problem.

3.3 Computational Methods

To compute the combinatorial configuration space and enumerate the intermediates for a particular polyhedron, we use a brute force method. Computation begins with first enumerating the intermediates with a single face. This enumeration is then used to compute the intermediates with two faces. This process proceeds iteratively until all intermediates are accounted for. Figure 3.11 outlines the detailed algorithm for this computation.

At each stage of our algorithm, we know the set of intermediates that have k faces, which we call A_k, and use this information to compute the set of faces with $k+1$ faces, A_{k+1}. Since all intermediates in A_{k+1} must be formed by adding a single face to an intermediate from A_k, we take each intermediate $[x] \in A_k$ and try adding each face to x that is allowable under the building game rules. This means we look at every face $f \in F$ and check if $f \notin x$ and also that x is edge connected to a face $\hat{f}$ that is in x. For every such faces f, we look at the new $(k+1)$-faced state $y \doteq x \cup \{f\}$. Since we know that $[y]$ is a building game intermediate, it must be represented in A_{k+1}, however before adding y to A_{k+1}, we must verify that there is no $\hat{y}$ already in A_{k+1} such that $y \in [\hat{y}]$.

The act of comparing two states y and $\hat{y}$ to check if they are members of the same intermediate is the task where the majority of computational time is spent. The brute force method of checking if $y \sim \hat{y}$ involves checking if $g.y = \hat{y}$ for each $g \in G$. If a g is found that makes this equality hold, then $[y] = [\hat{y}]$ and the computation

$A_1 \leftarrow \{\{f\} : f \in F\}$
for $\{f\} \in A_1$ **do**
  if $\exists \{\hat{f}\} \in A_1 \setminus \{f\} : [\{\hat{f}\}\,] = [\{f\}]$ **then**
    $A_1 \leftarrow A_1 \setminus \{\{f\}\}$
  end if
end for
$A_2, \ldots, A_{|F|} \leftarrow \{\}$
for $k = 0, \ldots, |F| - 1$ **do**
  for $x \in A_k$ **do**
    for $f \in F \setminus x$ such that $\exists \hat{f} \in x$ with f and $\hat{f}$ sharing an edge. **do**
      NewIntermediate $\leftarrow True$
      for $\hat{y} \in A_{k+1}$ **do**
        if $y \in [\hat{y}]$ **then**
          NewIntermediate $\leftarrow False$
          Add connection $[x] \leftrightarrow [\hat{y}]$ to combinatorial configuration space.
        end if
      end for
      if NewIntermediate $= True$ **then**
        $A_{k+1} \leftarrow A_{k+1} \cup \{y\}$
        Add connection $[x] \leftrightarrow [y]$ to combinatorial configuration space.
      end if
    end for
  end for
end for

Figure 3.11: Algorithm for iteratively enumerating the Building Game combinatorial configuration space.

terminates and we know that $[y]$ is not a new intermediate. In this case, nothing is added to A_{k+1} but a connection $x \leftrightarrow \hat{y}$ is added in the combinatorial configuration space. Furthermore, by tracking how many time a particular connection $x \leftrightarrow \hat{y}$ is found, the forward degeneracy numbers can be computed quickly.

3.3.1 Hash Functions

In practice, we implement a hash function $\mathfrak{h}$ that maps each state to an integer with the property that if $[y] = [\hat{y}]$, then $\mathfrak{h}(y) = \mathfrak{h}(\hat{y})$. If we can design such a function and it is computable in significantly less time relative to the brute force method of

trying every rotation, the overall computation can be majorly diminished.

When checking if two states y and $\hat{y}$ are members of the same intermediate, we first check if $\mathfrak{h}(y) = \mathfrak{h}(\hat{y})$. If they do not have the same hash, then they cannot be members of the same intermediate and the check is complete. Alternatively, if they do have identical hashes, it is not a guarantee that they are members of the same intermediate, so the brute force rotation method must be used. If the hash is carefully designed then this false positive rate ($\mathfrak{h}(y) = \mathfrak{h}(\hat{y})$ when $[y] \neq [\hat{y}]$) is small and almost all of the brute force calculations will result in a positive match.

In an ideal world, a hash function that also gives the property $[x] = [\hat{x}]$ whenever $\mathfrak{h}(x) = \mathfrak{h}(\hat{x})$ would be best. However we have not been able to find such an $\mathfrak{h}$ that is computable in a relatively reduced amount of time in comparison to the brute force method. We leave the existence of such a hash as a possibility.

Since any hash we choose must a function that maps states that are rotations of each other to the same integer, we look at local connectivity properties. For each face $f_k \in F$ define its local connectivity within y as $\mathfrak{h}_k(y) = |\{\hat{f} \in F : \hat{f} \in y, f_k \cap \hat{f} \in E\}|$, the number of faces adjacent to f_k that are also in y. Since this connectivity is preserved by rotations, even though we wont necessarily have $\mathfrak{h}_k(y) = \mathfrak{h}_k(\hat{y})$ when $[y] = [\hat{y}]$, if we take a histogram of all values of $\mathfrak{h}_k$ for $k = 1, \ldots, |F|$, the histogram will be the same for both y and $\hat{y}$. Once a histogram is computed, the final hash $\mathfrak{h}$ is just a simple function that maps histograms to integers.

There are many variants of statistics that this histogram strategy can be used with. We found that taking separate histograms for faces in y and faces not in y provided a hash with less false positives, while only increasing the hash computation time marginally. Depending on the polyhedron, the implementation of this hash lead

to an over all speed up of at least an order of magnitude over the purely brute force method.

3.3.2 Data Structures

Before the computation of the combinatorial configuration space, we hard-code an enumeration $f_1, \ldots, f_{|F|}$ of the faces of our polyhedron. Then, any state x is represented by a binary vector of length $|F|$ with a one in the kth entry if $f_k \in x$ and zero otherwise. With this convention, each rotation $g \in G$ corresponds to a permutation of the indices of x. Each such permutation in the group is precomputed and then applied as necessary when performing a brute force comparison of two states. To track the connectivity structure of the polyhedron, an adjacency list on faces is stored as a two dimensional array. For each face number, the adjacency list specifies the index of adjacent faces.

Each group of k-faced intermediates A_k is stored as a hash table using the previously described hash function. This allows for order one look-up of intermediates already in the A_k that share the hash of a new proposal intermediate.

3.3.3 Run Time

The computing time required to enumerate the combinatorial configuration space is heavily dependent on the number of intermediates that are found. Since we do not have a tight bound on the number of intermediates as a function of simple statistics of the polyhedron (faces, edges, etc.), it is difficult to provide a meaningful estimates on the time required to compute the combinatorial configuration space without explicit

knowledge of its size a priori.

That said, we can express an upper bound on the number of state to state comparisons that are required as a function of the intermediate sets $A_1, \ldots, A_{|F|}$.

$$\#\text{comparisons} \leq \sum_{k=1}^{|F|-1} \sum_{x \in A_k} \sum_{f \in F \setminus x} \sum_{\hat{y} \in A_{k+1}} 1 \tag{3.20}$$

$$\leq |F| \sum_{k=1}^{|F|-1} |A_k||A_{k+1}| \tag{3.21}$$

3.3.4 Implementation

All computation was carried out on a desktop computer running 64-bit Ubuntu 14.04 LTS with 15.6 GiB memory and a quad-core 3.20GHz Intel processor. To compute the building game combinatorial configurations space, we developed C++ code with the help of the Boost graph library for storing the connectivity structure CITE. Computation of the combinatorial configuration space took less than a minute for most polyhedra, though–as expected–this time grew dramatically with the number of faces. Our largest case, the 30-faced Rhombic Triacontahedron took on the order of 10 minutes to compute.

Additional computation–especially for pathway and shelling statistics–was completed using Python and the mpmath module for arbitrary precision arithmetic CITE.

Chapter Four

Constraint Models and Embedding Intermediates in Space

While models such as the Building Game treat assembly intermediates as idealized structures, the intermediates of the physical application we are trying to model may face a chaotic and volatile range of forces. To more realistically model the ways in which our intermediate might flex and move under these forces, we impose a constraint model in with the rigidity of individual components of an intermediate are assumed, but in which the edges at which they meet are treated like a hinge. This constraint system specifies the ways in which an intermediate has freedom to move if it is able to move at all. A physically motivated way of addressing questions that the discrete models themselves are not adequate to answer is provided by this framework.

4.0.5 Configuration Space

To characterize the freedoms of three-dimensional intermediates composed of rigid two-dimensional faces and connected at hinged edges, we parameterize each face and impose constraints in parameter space. Theoretically, the location and orientation of each face of an intermediate can be parameterized by six parameters: three for translation, three for rotation. This means the configuration of the entire intermediate can be represented by at most $6|F|$ parameters. In practice, however, it is often advantageous for ease of analysis and computational implementation to use more parameters to represent a given configuration. Often times, we use 3 parameters to represent the location each vertex of each face, even if some of these vertices share common locations. This means that we often have an ambient parameter space of $\mathbb{R}^N$ with $N > 6|F|$.

Upon this parameter space we place three types of constraints. The first removes the configuration's 6 trivial degrees of freedom due to translation and rotation. Since

the intermediate is connected, this can be achieved by fixing the parameters of one of the intermediate's faces. Additionally, we enforce a rigidity constraint on each face ensuring that the structure of a constituent face will not change with movement of the larger structure. The final type of constraint ensures that the connections between two faces have hinge-like mobility. Since a shared edge has two vertices belonging to each face, this constraint is imposed by identifying the corresponding vertex locations.

The *configuration space* is defined to be the subset of ambient space $\left\{z \in \mathbb{R}^N : \varphi(z) = 0\right\}$ for which all M of the constraint equations $\varphi : \mathbb{R}^N \to \mathbb{R}^M$ are satisfied. Since it is assumed that the standard configuration of the intermediate given by the model satisfies the constraints, the configuration space is non-empty. The configuration space is an algebraic variety, because the constraint equations can typically be represented as (quadratic) polynomials. The *degrees of freedom* of a particular configuration is taken to be the dimension of the null space of the Jacobian of φ. This is due to the fact that any move in the ambient space that prevents the constraint equations from changing must also be in the configuration space. Interestingly, it is possible for some members of configuration space to have a different number of degrees of freedom than others.

It is important to note that there are many possible parameterizations of a configuration and the choice of which will likely result in a different configuration space. The selection of which face to constrain in order to remove the trivial degrees of freedom may also affect the structure of the configuration space. While we must be cognizant of these choices, in many cases, the properties we wish to evaluate of the configuration space are independent of the choices.

4.1 Linkages and Frameworks

–izmestiev def of framework –freedom in machines def of linkage –used to describe configurations and freedoms

4.2 Geometric Configuration Space

–pictures of flexing. –embedding

Figure 4.1: Different geometric configurations of an octahedron intermediate.

Definition 16. *The* ***constraint space*** *corresponding to a constraint function* $c : \mathbb{R}^n \to \mathbb{R}^{n-m}$ *is its zero-set,*

$$\Omega \doteq \{z \in \mathbb{R}^n : c(z) = 0\}. \tag{4.1}$$

If c is composed of polynomial functions, then the constraint space is an algebraic variety. If this variety has no singularities, then it is also a manifold.

Using the notion of a constraint space, we can define the space of different geometric configurations that a given intermediate can take. Since we are treating each intermediate as collection of rigid polygons attached to each other along hinged edges, there are two basic types of constraint we must enforce. First, we must ensure that the each individual face is rigid and has the correct geometric shape. Additionally, each edge to edge connection must remain fixed with only hinge-like motion allowed.

Definition 17. *The **geometric configuration space** of a building game intermediate is a constraint space that enforces the rigidity of individual faces and hinged motion along connected edges.*

To mathematically describe a particular configuration, we must specify the locations of the vertices of each face. Thus, for an intermediate $[x]$, its building game configuration space can be represented as a subset of the ambient space $\mathbb{R}^{3\times N_x}$ where $N_x \doteq \sum_{f\in x} s_f$ where s_f is the number of sides (and verticies) of face f. Using this representation, we must then identify the corresponding constaint equations that give rise to the combinatorial configuration space as a function of points in ambient space.

It is worth noting that while we represent each face with $3 \times s_f$ coordinates, only 6 are required to speciafy a face's position and orientation if they are chosen carefully. With this in mind, we will typically use a function of 6 of a face's vertex coordinates to constrain each of the face's remaining vertex coordinates.

Notationally, we refer to the kth vertex of the jth face of x as $v^{jk} = \left(v_x^{jk}, v_y^{jk}, v_z^{jk}\right)$. Since the context of a constraint space requires a function $c : \mathbb{R}^n \to \mathbb{R}^{n-m}$, we flatten the matrix of vertex coordinates into a vector of lenghth $n = 3N_x$.

$$z = \begin{bmatrix} v^{1,1} \\ \vdots \\ v^{1,s_{f_1}} \\ \vdots \\ v^{|x|,1} \\ \vdots \\ v^{|x|,s_{f_{|x|}}} \end{bmatrix} \in \mathbb{R}^n \tag{4.2}$$

There are four fundamental types of constraint equations: edge length constraints, angle constraints, and $2D$ face constraint to enforce the rigid structure of each face as well as vertex identification constraints to enforce the hinged connections.

Edge length constraints enforce that the lengths of the edges of each face in an intermediate cannot change. If the kth edge is definted to be that between the $(k-1)$st and kth vertices, we use the following function to constrain its lengths to a known value ℓ^{jk}.

$$c_{edge}^{j,k}(z) = \left|v^{j,k} - v^{j,k-1}\right|^2 - (\ell_{j,k})^2 \tag{4.3}$$

$$= \left(v_1^{j,k} - v_1^{j,k-1}\right)^2 + \left(v_2^{j,k} - v_2^{j,k-1}\right)^2 + \left(v_3^{j,k} - v_3^{j,k-1}\right)^2 - (\ell^{j,k})^2 \tag{4.4}$$

This uses the notational convention that $v^{j,0} \doteq v^{j,s_j}$. For reasons that will be explained, we only explicitly enforce the lengths of two edges ($k = 1, 2$) per face, so there are a total of $2|x|$ edge length constraints.

Angle constraints ensure that each face's polygonal angles are conserved. Using the dot product formula for angles, we can write this constraint as a polynomial.

$$c_{ang}^{j,k}(z) = (v^{j,k-1} - v^{j,k}) \cdot (v^{j,k+1} - v^{j,k}) - \ell^{j,k}\ell^{j,k+1}\cos(\theta^{j,k}) \tag{4.5}$$

Here, $\theta^{j,k}$ is the angle at $v^{j,k}$ between kth and $(k+1)st$ edges which is a constant that is known a priori. In practice, we only enforce that the $k = 1$st angle constraint for each face.

Now, between the two edge constraints and one angle constraints, we have described 3 constraints as a function of 9 vertex coordinates per face. Thus for any

choice of the 6 coordinates, $v_1^{j,1}, v_2^{j,1}, v_3^{j,1}, v_1^{j,2}, v_2^{j,2}, v_1^{j,0}$, the remaining 3 coordinates, $v_3^{j,2}, v_2^{j,0}, v_3^{j,0}$, are specified by the 3 constraint equations. Similarly, since each face's location and rotational orientation can be defined by this choice of 6 coordinates, we have enough information to specify the coordinates of the remaining verticies.

Since the positions of the first three verticies dictate the locations of the remaining verticies, the $2D$ face constraints use a map from the known vertex coordinates to the yet unknown locations. Using a template for what the ideal polygonal stucture for each face should be, we use a rotation matrix to specify these remaining verticies. If this template has vertices $\hat{v}^{j,0}, \hat{v}^{j,1}, \hat{v}^{j,2}, \ldots, \hat{v}^{j,k}, \ldots$ and the locations for $v^{j,0}, v^{j,1}$, and $v^{j,2}$ are known, we can identify the location of $v^{j,k}$ for $k > 2$. Using this template, we can define the following length and angle constants.

$$\ell^{j,k_1,k_2} \doteq |\hat{v}^{j,k_1} - \hat{v}^{j,k_2}| \tag{4.6}$$

$$\phi^{j,k_1,k_2,k_3} \doteq \cos^{-1}\left(\frac{\left(\hat{v}^{j,k_1} - \hat{v}^{j,k_2}\right) \cdot \left(\hat{v}^{j,k_3} - \hat{v}^{j,k_2}\right)}{\left(\ell^{j,k_1,k_2}\right)\left(\ell^{j,k_3,k_2}\right)}\right) \tag{4.7}$$

Figure 4.2: Face template and 2d face constraints.

Our basic stratigy is to first place a point $\bar{v}^{j,k}$ in the span of $v^{j,0} - v^{j,1}$ at a distance of $|\bar{v}^{j,k} - v^{j,1}| = \ell^{j,k,1}$. The choice $\bar{v}^{j,k} = v^{j,1} + \frac{\ell^{j,k,1}}{\ell^{j,0,1}}(v^{j,0} - v^{j,1})$ will work, since

$$|\bar{v}^{j,k} - v^{j,1}| = |\frac{\ell^{j,k,1}}{\ell^{j,0,1}}(v^{j,0} - v^{j,1})| \tag{4.8}$$

$$= \frac{\ell^{j,k,1}}{\ell^{j,0,1}}|v^{j,0} - v^{j,1}| \tag{4.9}$$

$$= \ell^{j,k,1}. \tag{4.10}$$

Then, a rotation matrix is used to rotate $\bar{v}^{j,k}$ by the correct angle into its position $v^{j,k}$. The rotation matrix is centered at $v^{j,1}$ and its axis of rotation is defined by

$u = \frac{1}{\ell^{j,0,1}\ell^{j,2,1}}(v^{j,0} - v^{j,1}) \times (v^{j,2} - v^{j,1})$. Similarly, the angle of rotation $\phi^{j,0,1,k}$ is the angle created by the two line segments in the template $(\hat{v}^{j,0}, \hat{v}^{j,1})$ and $(\hat{v}^{j,2}, \hat{v}^{j,1})$. Thus, using $R = R(\phi^{j,0,1,k}, u)$ and our equation for $v^{j,k}$ is

$$v^{j,k} = v^{1,k} + R(\bar{v}^{j,k} - v^{j,1}) \tag{4.11}$$

$$= v^{1,k} + \ell^{j,k,1} R(v^{j,0} - v^{j,1}) \tag{4.12}$$

$$\tag{4.13}$$

Since R is polynomial in $v^{j,0}, v^{j,1}, v^{j,2}$, we get the following polynomial 2D face constraint for each $k > 2$.

$$c_{2D}^{j,k}(z) \doteq v^{1,k} + \ell^{j,k,1} R(v^{j,0} - v^{j,1}) - v^{j,k} \tag{4.14}$$

The final constraint type, vertex identification, is used to enforce that the connection between edges of two faces has the mobility of a hinge. To do this, we simply need to ensure that that corresponding vertices on each edge share identical locations. This results in the relatively simple constraints:

$$c_{ident}^{j_1,k_1,j_1,k_1,d}(z) \doteq v_d^{j_1,k_1} - v_d^{j_2,k_2} \tag{4.15}$$

$$\tag{4.16}$$

where v^{j_1,k_1} and v^{j_2,k_2} are corresponding vertices from the faces j_1 and j_2 meeting at a hinged edge. If there are $|E_x|$ of these hinged connections in a building game state x, there must be $6|E_x|$ correcponding vertex identification constraints.

With all four constraint types explicitly defined, the aggregate constraint function for x will have $n-m = 2|F_x|+|F_x|+(N_x-3|F_x|)+6|E_x| = N_x+6|E_x|$ total constraints.

4.2.1 Special Case of Triangular Faces

In the case where all of the faces of the polyhedron we consider are triangles (tetrahedron, octahedron, icosahedron, etc.) we notice that by simple enforcing that each edge of each triangle has a specified length, the triangle will be rigid. This means that we do not have to use the angle and 2D face constraints. Further, rather that explicitly using vertex identification constraints, we can either treat them as length constraints with zero length between identified vertices or we can simply reindex the vertices so that identified vertices are actually treated as a single vertex. With either choice, in the trianglular case, we may only deal with length constraints if we wish. This will be a useful property in the next chapter.

Additionally, we make a conjecture that under certain trianglular conditions, our algebraic variety has no singularities and is thus a manifold.

Conjecture 1. *The geometric configuration space for each intermediate composed of equilateral triangular faces is a manifold.*

4.3 Degrees of Freedom

Roughly speeking, the degrees of freedom of a system are the different independent motions the system is able to exercise. Since the concept of degrees of freedom exists in many diverse scientific fields, such as mechanical engineering and statistical physics, many different formal definitions of degrees of freedom are used in the literature. McCarthy defines degrees of freedom of a mechanical system as follows. CITE

> We derive formulas for the number of parameters needed to specify the configuration of a mechanism, in terms of the number of links and joints and the freedom of movement allowed at each joint. This number is the *degrees of freedom* or *mobility* of the mechanism. Changing the values of these parameters changes the configuration of the mechanism. Thus, if we view the set of all configuration available to a mechanism as a manifold, then the mobility of the mechanism is the dimension of this manifold.

Since our geometric configuration space is an algebraic variety and not a manifold in general, this definition must be modified to make degrees of freedom a statistic of each individual configuration rather than a global statistic of the geometric configuration space.

Definition 18. *The number of* ***degrees of freedom*** *a building game intermediate at configuration z is the dimension of the geometric configuration space algebraic variety at z. If z is a singularity of the algebraic variety, then the degrees of freedom is undefined.*

In most cases we consider, rigid rotation and rigid translation will preserve the value of the constraint function since they do not move the verticies relative to one another. In other words, if $c(z) = 0$ then we also have $c(Rz + T) = 0$ where $R \in \mathbb{R}^n \times n$ rotates each vertex in the configuration by some $\hat{R} \in SO(3)$ and translates each vertex by $\hat{T} \in \mathbb{R}^3$. With this definition, R is the block diagonal matrix with each of the $n/3$ block being $\hat{R} \in SO(3)$ and $T \in \mathbb{R}^n$ is composed of $n/3$ copies of $\hat{T} \in \mathbb{R}^3$ stacked upon each other. These rigid body rotations account for 6 of the configuration's degrees of freedom: 3 rotational degrees of freedom and 3 translational.

Definition 19. *The **trivial degrees of freedom** a building game intermediate at configuration z are the* 6 *degrees of freedom corresponding to rigid body rotations and translations.*

Thus, the degrees of freedom we are most interested in are those that do represent movement of the verticies and faces relative to each other.

Definition 20. *The **internal degrees of freedom** a building game intermediate at configuration z are the degrees of freedom that are not trival.*

4.3.1 Computing Degrees of Freedom

For a general constraint space, we can find the dimension of the space at a point z by looking at the Jacobian matrix $C(z) \in \mathbb{R}^{(n-m)\times n}$ of the constraint function c. Since the number of degrees of freedom is defined to be the dimension of that space, we look at the rank of the Jacobian. Since this rank quantifies the number of independent constraints given by c at z, the number of degrees of freedom is given by the following.

$$DoF \doteq n - \text{rank}\left((C(z)\right) \tag{4.17}$$

Since the rank can take values between 0 and $\min\{n, n-m\} = n-m$, there can be anywhere from 0 degrees of freedom, when there are functionally no constraints on z, and m degrees of freedom in the case where all $n-m$ constraint equations are independent and $C(z)$ is of full rank.

In the typical case in which the constraint equations are invarient under three-dimensional rotation and translation, there will be six trivial degrees of freedom.

The number of internal degrees of freedom would then be $n - \text{rank}\left((C(z)\right) - 6$.

Definition 21. *If there are zero iternal degrees of freedom at a onfiguration z, we say that the configuration is* ***rigid****.*

To actually compute the number of degrees of freedom for configurations of building game intermediates, we must first find an explicit form for the Jacobian matrix C. Below are the partial derivatives of the constraint functions we specified above.

$$\frac{\partial c_{edge}^{j,k}}{\partial z_i} = \begin{cases} 2\left(v_d^{j,k} - v_d^{j,k-1}\right) & \text{if } z_i = v_d^{j,k} \\ -2\left(v_d^{j,k} - v_d^{j,k-1}\right) & \text{if } z_i = v_d^{j,k-1} \\ 0 & \text{else} \end{cases} \tag{4.18}$$

$$\frac{\partial c_{ang}^{j,k}}{\partial z_i} = \begin{cases} v_d^{j,k+1} - v_d^{j,k} & \text{if } z_i = v_d^{j,k-1} \\ 2v_d^{j,k} - v_d^{j,k-1} - v_d^{j,k+1} & \text{if } z_i = v_d^{j,k} \\ v_d^{j,k-1} - v_d^{j,k} & \text{if } z_i = v_d^{j,k+1} \\ 0 & \text{else} \end{cases} \tag{4.19}$$

$$\frac{\partial c_{2D}^{j,k}}{\partial z_i} = \begin{cases} & \text{if } z_i = v_d^{j,0} \\ & \text{if } z_i = v_d^{j,1} \\ & \text{if } z_i = v_d^{j,2} \\ -1 & \text{if } z_i = v_d^{j,k} \\ 0 & \text{else} \end{cases} \tag{4.20}$$

$$\frac{\partial c_{ident}^{j_1,k_1,j_2,k_2,d}}{\partial z_i} = \begin{cases} 1 & \text{if } z_i = v_d^{j_1,k_1} \\ -1 & \text{if } z_i = v_d^{j_2,k_2} \\ 0 & \text{else} \end{cases} \tag{4.21}$$

Since the number of degrees of freedom is defined as a local property of a specific configuration z, different configurations of the same building game intermediate could theoretically have differing numbers of degrees of freedom. However, we have yet to observe a building game intermediate that has this property and is evidence that supports conjecture 1. A contrived case in which a linkage of squares leads to a configuration space with differing degrees of freedon is described in section 4.3.2. When we report the number of degrees of freedom that a building game intermediate has, we use its connonical configuration in computation.

Definition 22. *The* ***cannonical configuraion*** *of a building game intermediate* $[x]$ *is the configuration that is simply the* 3*-dimensional embedding of the polyhedron restricted to the faces in* x

For example, in the case of the cube, the cannonical configuration would have all of the faces of the intermediate meeting at right angles just as they do in the cube itself.

In practice, we use NumPy's numpy.linalg.svd module to take the singlular value decomposition of the compute Jacobian matrix in Python. The rank of the Jacobian is then the number of effectively non-zero singlular values where a small cutoff is used to determine whether a small singular value is indeed zero or non-zero.

4.3.2 Non-Manifold Example

Consider the linkage of six squares aranged in a 2×3 lattice depicted in figure 4.3. This linkage has three lines of hinges: 2 accross and 1 down. As shown, the two horizontal hinges can be manipulated independently with each of the vertical hinges

maintaining at an angle of π degrees. This seems to indicate two internal degrees of freedom. Alternatively, when folded along the verticle crease, the horizontal hinges remain fixed with dihedral angle π. This configuration corresponds to just a single internal degree of freedom. Thus, the transition between these two modes, when the linkage is completely planar, there is a singularity and the number of degrees of freedom is not definied.

Figure 4.3: A linkage of 6 squares with configurations of different degrees of freedom.

As stated previously, we have not observed this type of behavior from any building game intermediates. This leads us to speculate that the degenerate nature of this example is down to an uncommon alignment of the hinged edges. Clearly, this linkage could never be an intermediate of a convex polyhedron. Perhaps the building game geometric configuration space has no singularities and is hence a manifold for all convex polyhedra.

4.3.3 Results

4.3.4 Explicit Removal of Trivial Degrees of Freedom

Since we are often solely interested the internal degrees of freedom, there are a few was to augment the constraint equations to mod out the translation or rotations causing the trivial degrees of freedom. One possible choice is to pick a face and constrain its vertex locations to match a fixed template. The corresponding constraint

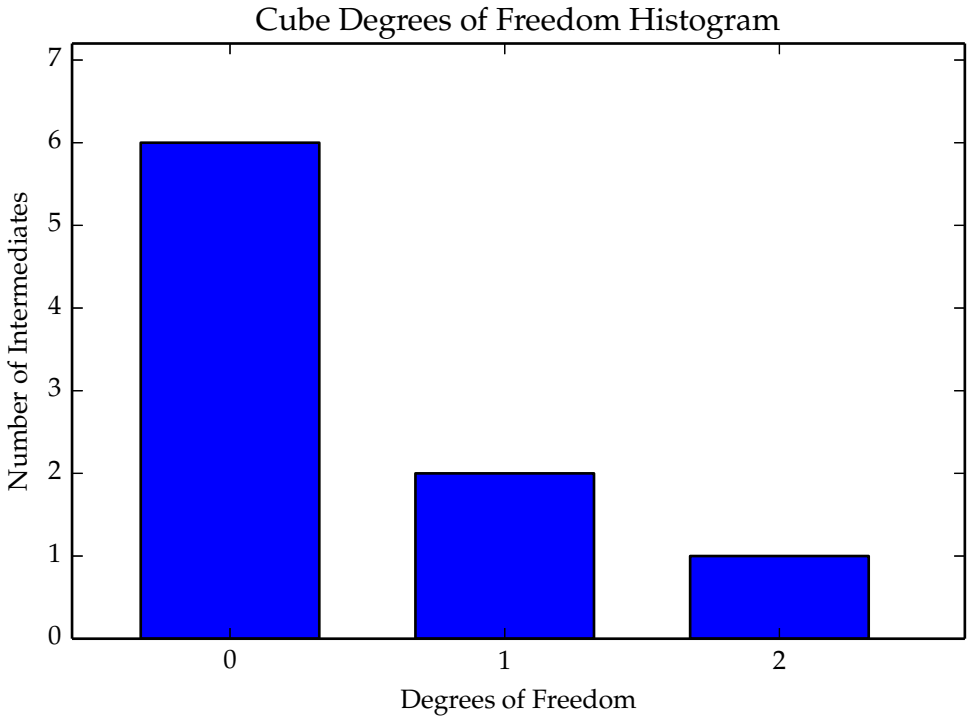

Figure 4.4

equations for fixing the jth face would be

$$c_{fix}^{j,k,d}(z) = v_d^{j,k} - \hat{v}_d^{j,k} \tag{4.22}$$

$$\frac{\partial c_{fix}^{j,k,d}}{\partial z_i} = \begin{cases} 1 & \text{if } z_i = v_d^{j,k} \\ 0 & \text{else} \end{cases} \tag{4.23}$$

for each dimension d of each vertex k of face j. For reasons we will explain in the next chapter, this may not always be an ideal choice if there is information other than the degrees of freedom we wich to derive from the constraint equations.

An alternative that prevents translation, but does allow rotation is to fix the

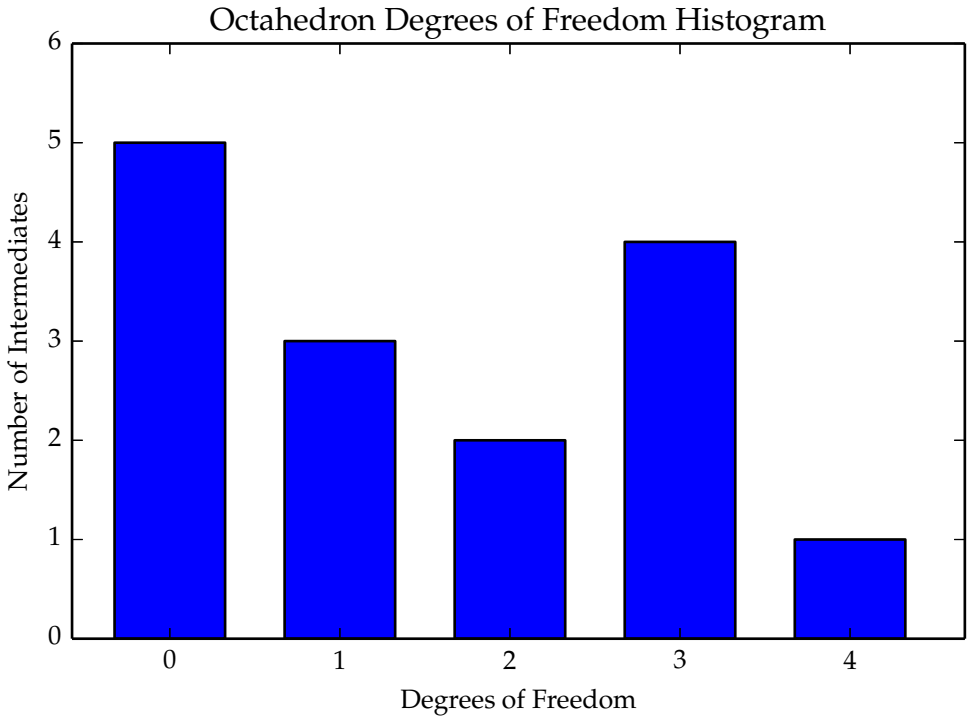

Figure 4.5

configuration's center of mass.

$$c_{com}^{d}(z) = \sum_{j=1}^{|F_x|} \sum_{k=1}^{s_{f_j}} v_d^{j,k} \tag{4.24}$$

$$\frac{\partial c_{com}^{d}}{\partial z_i} = \begin{cases} 1 & \text{if } z_i = v_d^{j,k} \\ 0 & \text{else} \end{cases} \tag{4.25}$$

This amounts to the sum over the dth entry of each vertex in the configuration.

4.4 Cyclohexane Application

??Include??

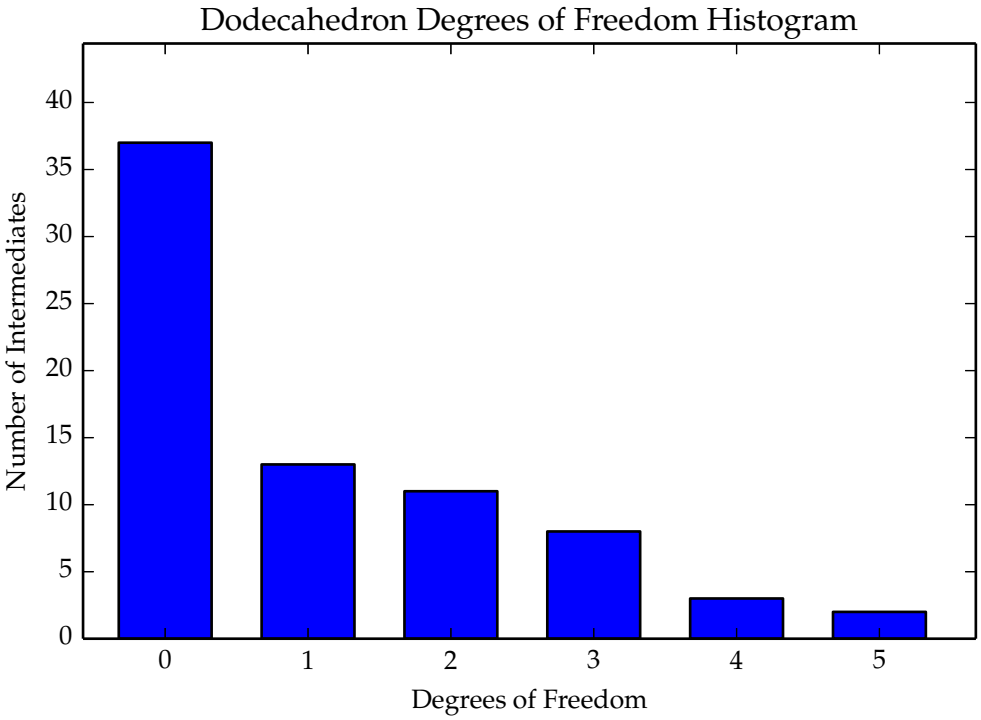

Figure 4.6

4.5 Folding Configuration Space

???include??

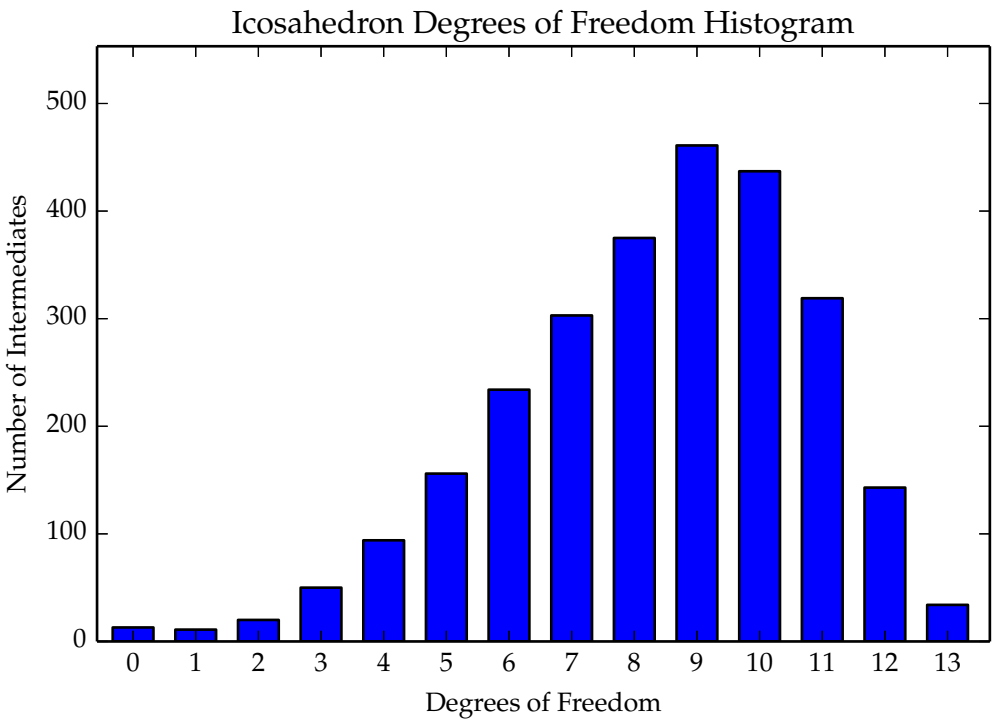

Figure 4.7

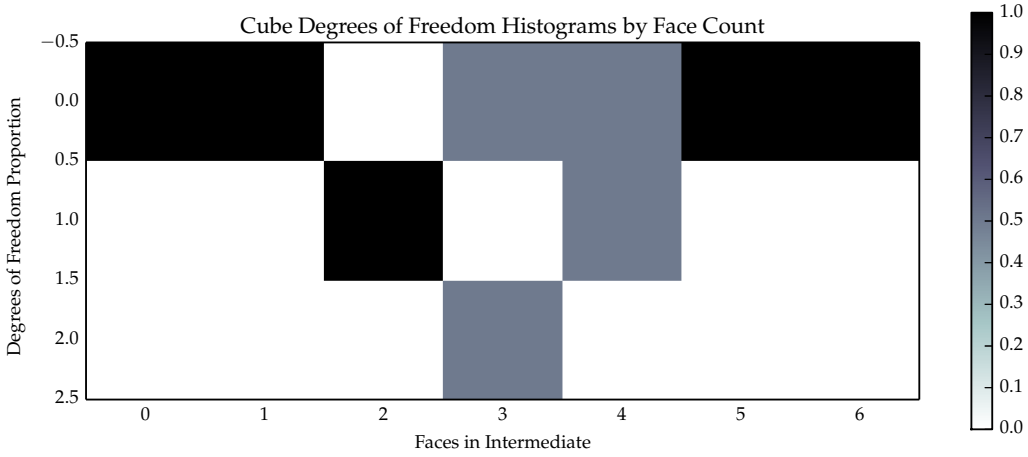

Figure 4.8

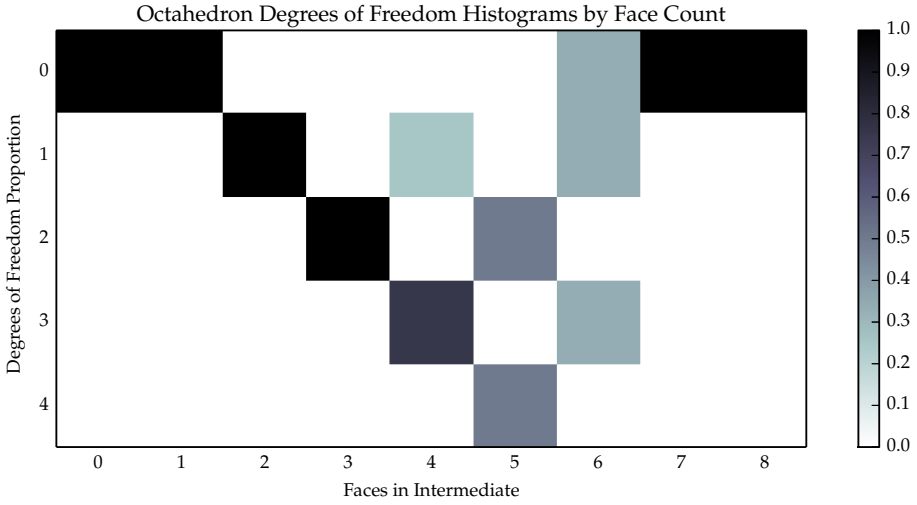

Figure 4.9

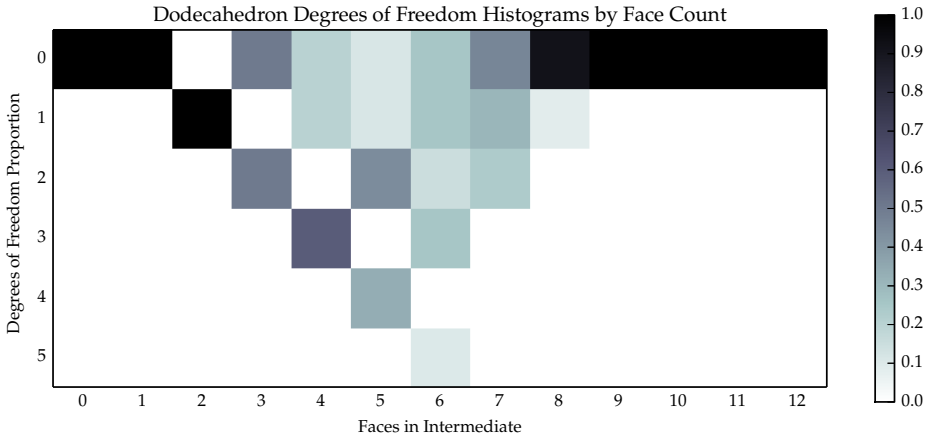

Figure 4.10

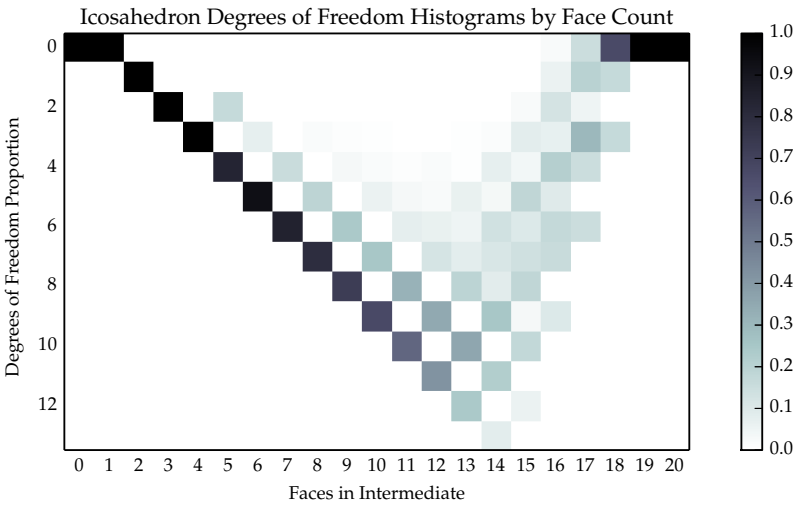
Icosahedron Degrees of Freedom Histograms by Face Count
Degrees of Freedom Proportion
0
2
4
6
8
10
12
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Faces in Intermediate
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0

Figure 4.11

Chapter Five

Processes in Constraint Spaces

5.1 Constrained Dynamics

Sometimes we are interested in more than simply the number of degrees of freedom a particular configuration has. To explore the configuration space, find similarities between different configurations, analyze the geometric configuration space's topology, or compute other relevant statistics, it can be useful to compute dynamic processes on the constraint space. There are clear computational challenges to computing such dynamics due to computational handling of constraints, the lack of an explicit parameterizations, and the fact that geometric configuration space is often of much smaller dimension than the ambient space it sits in.

Constraint algorithms have been widely studied for use in molecular dynamic simulations. Many methods use a two stage scheme in which the unconstrained dynamics are first used to evolve the system to an intermediary configuration $\hat{z}_{n+1}$ based on the previous configuration z_n. In the second stage, Lagrange multipliers are used to correct $\hat{z}_{n+1}$ to a new configuration z_{n+1} which satisfies the constraints. Previously, we used the SHAKE algorithm–which employs this two stage philosophy– to examine a constraint space used to model the configurations of cyclohexane CITE.

–Here get specific to our interests, BM using manifold metric Assuming connectedness of the configuration space, constraint algorithms can be used to compute various statistics of the configuration space. For example, the d-dimensional volume of a d-dimensional configuration space can be used to measure how much mobility the intermediate has in its configuration space. Additionally, if we have soft constraints, such as preferred angles between faces, we can institute cost functions on the configuration space with configurations having more favorable angles having a lower cost. Whether physically or artificially motivated, such cost functions can be treated

as potential energy functions which can imply specific dynamics. Under this set of dynamics, quantities such as the average or minimum energies may be of interest.

We may also be interested in the behavior of a small part of the configuration as it undergoes constrained dynamics. For instance, suppose we are interested in the tendency for two edges on distinct faces to come together and form a new connection. By looking at the distance between these edges, we can measure how frequently this event happens and the probability that it will occur before another event of interest. This is similar to the exit time problem for diffusion and may provide a physically motivated evaluation tool for the appropriateness of our transition rules we adopt in our models.

5.1.1 Cyclohexane Dynamics

??include??

5.2 Manifold Brownian Motion

5.2.1 Computational Scheme

Suppose we have a m dimensional manifold $\mathcal{M} \subset \mathbb{R}^n$ defined implicitly by $\mathcal{M} \doteq \{z \in \mathbb{R}^n : c(z) = 0\}$ for some function $c : \mathbb{R}^n \to \mathbb{R}^{n-m}$ with $c_1, \ldots, c_{n-m}$ independent constraint functions. We seek to approximate a Brownian motion on $\mathcal{M}$ by constructing a random walk. As mentioned before, this is done in two stages; starting at a point $z \in \mathcal{M}$ we take a step of size $\sqrt{\Delta t}$ in a tangent direction w and then

we project the point $z + \sqrt{\Delta t} w$ back onto $\mathcal{M}$.

Sampling in Tangent Space

At each point $z \in \mathcal{M}$ the tangent space is $\mathcal{T}_z\mathcal{M} \doteq \{w \in \mathbb{R}^n : C(z)\, w = 0\}$ where $C : \mathbb{R}^n \to \mathbb{R}^{(n-m)\times n}$ is the Jacobian of c. Thus, sampling from the tangent space reduces to sampling from the null space of $C(z)$.

First we must construct a basis for $\mathcal{T}_z\mathcal{M}$. Let $A = [C^T(z)B]$ be the concatenation of $C^T(z)$ and a randomly drawn matrix $B \in \mathbb{R}^{n\times m}$. We assume each entry of B is independent with $B_{jk} \sim \text{unif}(0, 1)$.

Lemma 8. *The matrix A is of full rank with probability 1.*

Proof. Since the constraints of c are independent, C is of full rank. Additionally, since B is drawn randomly, each column of B will be independent of the columns of C^T and also the other columns of B almost surely. Thus, the columns of A are independent and A is of full rank. □

Now, we compute the QR decomposition

$$A = \left[C^T B\right] = QR = \left[Q^{(1)} Q^{(2)}\right] R \tag{5.1}$$

where $Q^{(1)} \in \mathbb{R}^{(n-m)\times n}$ and $Q^{(2)} \in \mathbb{R}^{m\times n}$.

Theorem 8. *The columns of $Q^{(2)}$ are an orthonormal basis for $\mathcal{T}_z\mathcal{M} = N(C)$ and the columns of $Q^{(1)}$ are an orthonormal basis for $(\mathcal{T}_z\mathcal{M})^\perp = (N(C))^\perp$.*

Proof. The second claim is true by simple linear algebra arguments and the properties of the QR decomposition.

$$\begin{aligned} \text{col}(Q^{(1)}) &= \text{col}(C^T) && (5.2)\\ &= ((\text{col}(C^T))^\perp)^\perp && (5.3)\\ &= (N((C^T)^T)^\perp && (5.4)\\ &= (N(C))^\perp && (5.5) \end{aligned}$$

Here we use the vector space identities $X = (X^\perp)^\perp$ and $(\text{col}X)^\perp = N(X^T)$. This shows that columns of $Q^{(1)}$ are an orthonormal basis for $(\mathcal{T}_z\mathcal{M})^\perp$. Now, since

$$\begin{aligned} \text{col}(Q^{(1)}) \oplus \text{col}(Q^{(2)}) &= \text{col}(A) && (5.6)\\ &= \mathbb{R}^n && (5.7)\\ &= (N(C)) \oplus (N(C))^\perp && (5.8)\\ &= (N(C)) \oplus \text{col}(Q^{(1)}) && (5.9) \end{aligned}$$

we must have that $\text{col}(Q^{(2)}) = N(C)$ and that the columns of $Q^{(2)}$ provide an orthonormal basis for $\mathcal{T}_z\mathcal{M}$. □

Now, with our orthonormal basis for $\mathcal{T}_z\mathcal{M}$, we can define any $w \in \mathcal{T}_z\mathcal{M}$ by a vector $\alpha \in \mathbb{R}^m$ as follows.

$$w = \sum_{i=1}^{m} \alpha_i Q_i^{(2)} = Q^{(2)}\alpha$$

So, for any two $w, \tilde{w} \in \mathcal{T}_z\mathcal{M}$ we can define the manifold metric g as

$$g(w, \tilde{w}) = g\left(\sum_{i=1}^{m} \alpha_i Q_i^{(2)}, \sum_{j=1}^{m} \tilde{\alpha}_j Q_j^{(2)}\right) \tag{5.10}$$

$$= \sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \tilde{\alpha}_j g\left(Q_i^{(2)}, Q_j^{(2)}\right) \tag{5.11}$$

$$= \sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \tilde{\alpha}_j G_{ij} \tag{5.12}$$

$$= \alpha^T G \tilde{\alpha} \tag{5.13}$$

Then, our goal is to sample uniformly from $\Omega_G = \left\{w = Q^{(2)}\alpha : \alpha^T G \alpha = 1\right\}$. Since this set corresponds to a level set of the Multivariate normal distribution with mean zero and covariance matrix G^{-1}, our sampling problem reduces to sampling $u \sim \mathcal{N}(0, G^{-1})$. Given u, $w \doteq \frac{u}{|u|}$ is sampled from Ω_G as desired. We generally choose $G = I$, but this is not required.

Projection onto $\mathcal{M}$

After making a step away from z of size $\sqrt{\Delta t}$ in the direction $w \in N(C)$, our new point $z + \sqrt{\Delta t}w$ is not likely to be in $\mathcal{M}$. To return our point to $\mathcal{M}$, we make a projection step $w^\perp \in (N(C))^\perp$. Since $Q^{(1)}$ provides a basis for $(N(C))^\perp$, we can write any such step as

$$w^\perp = \sum_{i=1}^{n-m} \gamma_i Q_i^{(1)} = Q^{(1)}\gamma.$$

with $\gamma \in \mathbb{R}^{n-m}$. Thus, we want to identify a γ such that $c(z + \sqrt{\Delta t}w + Q^{(1)}\gamma) = 0$. Since c is nonlinear, we use Newton-Raphson iteration to solve for γ. By defining an

objective function F and its Jacobian $\mathcal{J}$ as

$$F(\gamma) = c(z + \sqrt{\Delta t}w + Q^{(1)}\gamma) \tag{5.14}$$

$$\mathcal{J}_{jk}(\gamma) = \frac{\partial F_j}{\partial \gamma_k}(\gamma) \tag{5.15}$$

$$= \sum_{i=1}^{n-m} \frac{\partial c_j}{\partial z_i}(z + \sqrt{\Delta t}w + Q^{(1)}\gamma)Q^{(1)}_{ik} \tag{5.16}$$

$$\mathcal{J}(\gamma) = C(z + \sqrt{\Delta t}w + Q^{(1)}\gamma)Q^{(1)} \tag{5.17}$$

we arrive at the iteration routine below, which typically commences with the initial guess $\gamma_0 = 0$.

$$C(z + \sqrt{\Delta t}w + Q^{(1)}\gamma)Q^{(1)}\left(\gamma_{k+1} - \gamma_k\right) = \mathcal{J}(\gamma_k)\left(\gamma_{k+1} - \gamma_k\right) \tag{5.18}$$

$$= -F(\gamma_k) \tag{5.19}$$

$$= -c(z + \sqrt{\Delta t}w + Q^{(1)}\gamma) \tag{5.20}$$

Sampling Algorithm

for $k = 1, \ldots, N$ **do**
  $B \leftarrow \text{unif}(0,1)^{n\times m}$
  $A \leftarrow \left[C^T B\right]$
  $[Q^{(1)}, Q^{(2)}], R \leftarrow$QRdecomposition$(A)$
  $\alpha \leftarrow \mathcal{N}(0, G^{-1})$
  $w \leftarrow \frac{Q^{(2)}\alpha}{|Q^{(2)}\alpha|}$
  $\gamma \leftarrow 0^{n-m}$
  while $|c(z + \sqrt{\Delta t}w + Q^{(1)}\gamma)| > \epsilon$ **do**
    $\Delta\gamma \leftarrow (C(z + \sqrt{\Delta t}w + Q^{(1)}\gamma)Q^{(1)})^{-1}(-c(z + \sqrt{\Delta t}w + Q^{(1)}\gamma))$
    $\gamma \leftarrow \gamma + \Delta\gamma$
  end while
  $z \leftarrow z + \sqrt{\Delta t}w + Q^{(1)}\gamma$
end for

Figure 5.1: Random Walk on $\mathcal{M}$

5.2.2 Validation and Test Cases

–SO(n) and ellipsoid examples.

–Finite time statistics?

5.2.3 MBM on Geometric Configuration Spaces

–two linkage case

–three linkage case

Fixing Trivial Degrees of Freedom

From the two and three triangle linkage examples, we see that the sampling scheme does not sample the dihedral angles of the configuration uniformly. This was unexpected, but possible explained by the choice to not explicitly mod out the trivial rotational and translational degrees of freedom.

The first apparent way of fixing the trivial degrees of freedom is to fix an individual face at a specific location. Without letting this face translate or rotate, we are also preventing these trivial degrees of freedom from entering the entire configuration. As seen in figure **??**, fixing of one of the faces in a two triangle linkage will give the dihedral angle a uniform distribution as hoped. However, when we consider

Figure 5.2: Self Intersection Boundary

the three triangle linkagae, there are two choices we have; we can either fix on of the

outside triangles or we can fix the interior face. As seen in figure **??**, if we fix the interior triangle, the distribution we observe will be uniform. However, if we choose to fixe an exterior triangle, the dihedral angle between the fixed triangle and the interior triangle will be uniformly sampled, but the other dihedral will not have a uniform distribution. Seen in figure 5.3, the distribution of this second dihedral angle appears to be the sum of trigonometric functions. This poses a bit of a problem. If

Figure 5.3: Three triangle linkage with interior triangle fixed.

Figure 5.4: Three triangle linkage with exterior triangle fixed.

the choice of face to fix affects the resulting distribution, then we must have justification for selecting one face over another. Since a clear reasoning for this choice is not apparent–especially in the case of an arbitrary linkage–we look to other methods for fixing the trivial degrees of freedom that the system has.

As mentioned in section 4.3.4, one way of fixing the translational degrees of freedom is to fix the center of mass of the configuration.

– govinds method for fixing rotation

– curvature of manifold makes for non-uniform sampling of dihedrals.

5.3 Manifold Reflected Brownian Motion

–Not physical!

– check for self intersection and dihedral switching before accepting proposal.

5.3.1 Computational Scheme

for $k = 1, \ldots, N$ **do**
  $B \leftarrow \text{unif}(0,1)^{n \times m}$
  $A \leftarrow \left[C^T B\right]$
  $[Q^{(1)}, Q^{(2)}], R \leftarrow$QRdecomposition$(A)$
  $\alpha \leftarrow \mathcal{N}(0, G^{-1})$
  $w \leftarrow \frac{Q^{(2)}\alpha}{|Q^{(2)}\alpha|}$
  $\gamma \leftarrow 0^{n-m}$
  while $|c(z + \sqrt{\Delta t}w + Q^{(1)}\gamma)| > \epsilon$ **do**
    $\Delta\gamma \leftarrow (C(z + \sqrt{\Delta t}w + Q^{(1)}\gamma)Q^{(1)})^{-1}(-c(z + \sqrt{\Delta t}w + Q^{(1)}\gamma))$
    $\gamma \leftarrow \gamma + \Delta\gamma$
  end while
  $z \leftarrow z + \sqrt{\Delta t}w + Q^{(1)}\gamma$
end for

Figure 5.5: Reflected Random Walk on $\mathcal{M}$

Self-Intersection Testing

– Triangle intersection checking CITE

HI

Figure 5.6: Self Intersection Boundary

5.3.2 Validation and Test Cases

– positive sphere/ellipsoid. others?

5.3.3 MRBM on Geometric Configuration Spaces

– two triangle

– three triangle

5.4 Computational Implementation

As with our enumeration work, simulations were carried out on a desktop computer running 64-bit Ubuntu 14.04 LTS with 15.6 GiB memory and a quad-core 3.20GHz Intel processor. The module for simulating the manifold Brownian motion was implemented in Python and was able to sample at a rate of roughly $X,000,000$ samples per hour. This rate varies with use of a boundary and the dimension of the manifold and ambient spaces as well.

Chapter Six

Results

6.1 Deriving Rates

6.2 Self-Assembly Statistics

Bibliography

[1] J Rotman. *An Introduction to the Theory of Groups*. Springer-Verlag, New York, NY, fourth edition, 1995.

[2] A Zlotnick. An equilibrium model of the self assembly of polyhedral protein complexes. *Journal of Molecular Biology*, 241:59–67, 1994.