Modeling Virus Self-Assembly Pathways Using Computational Algebra and Geometry

Meera Sitharam and Mavis Agbandje-Mckenna

ABSTRACT. We develop a tractable model for elucidating the assembly pathways by which an icosahedral viral shell forms from 60 identical constituent protein monomers. This poorly understood process a remarkable example of macromolecular self-assembly occuring in nature and possesses many features that are desirable while engineering self-assembly at the nanoscale.

The model uses static geometric constraints to represent the driving (weak) forces that cause a viral shell to assemble and hold it together. The goal is to answer focused questions about the structural properties of a successful assembly pathway. Pathways and their properties are carefully defined and computed using computational algebra and geometry, specifically by crucial modifications to state of the art methods for geometric constraint decomposition. The model is analyzable and refinable and avoids expensive dynamics. It has a provably tractable and accurate computational simulation and that its predictions are roughly consistent with known information about viral shell assembly.

Overall the paper's new contributions are: (a) elucidation of a new model of virus assembly illustrating a strong, direct, mutually beneficial interplay between the concepts underlying macromolecular assembly and a wide variety of established as well as novel concepts from combinatorial and computational algebra, geometry and algebraic complexity; and (b) crucial modifications to existing geometric constraint decomposition methods to obtain a tractable and accurate simulation of the model.

Organization of Paper

–Introduction and Motivation
–Geometric Constraints Background
–The Virus Assembly Model: Pathways and Effort
–A Tractable and Accurate Simulation

supported in part by NSF-EIA 0218435.

1. Introduction and Motivation

Icosahedral viral shell assembly is an outstanding example of nanoscale, macromolecular self-assembly occuring in nature [**30**]. Mostly identical *coat protein* monomers assemble with high rate of efficacy into a closed icosahedral *capsid* or *shell*; onset and termination are spontaneous, and assembly is robust, rapid and economical. All of these requirements are both desirable and difficult to achieve when engineering macromolecular self-assembly. See Figures 1.

However the viral assembly process - just like any other spontaneous macromolecular assembly process such as molecular crystal formation - is poorly understood. Answering focused questions about viral assembly pathways can help both to encourage macromolecular assemblies for engineering, biosensor and gene therapy applications, but and also discourage assembly for arresting the spread of viral infection.

This paper addresses the relevance of computational algebra and geometry to develop static, analyzable, refinable models and fast, accurate computational simulations for answering focused questions about virus assembly pathways. The paper also discusses the relevance of random walks on markov chains for randomly sampling algebraic structures. Specifically, we use the following. First, we use state of the art methods for decomposition of geometric constraint system [**36, 38, 39, 37**] [**33, 47, 51, 72, 62, 64, 68, 71, 67, 70**], as well as solving and estimating number of solutions. These leverage both combinatorial approaches related to rigidity theory [**31**], as well as standard algebraic techniques for sparse elimination and solving [**15, 24, 40, 73**]. Second, we use random walks [**4, 5, 12, 26, 50, 57, 29**] to obtain statistically good samples for enumerating and counting specific structures that have an algebraic interpretation as decompositions of the underlying geometric constraint systems, and are combinatorially related to rigidity matroids [**65, 63**].

Existence of purely algebraic methods of estimating such statistics [**56**] is likely and would be useful to find. Furthermore, although they are not used in this paper, the problem provides direct scope (see [**66**] for techniques and concepts (standard and novel) from other areas of combinatorial and computational algebraic geometry, both standard and novel. For example, [**66**] sketches the direct relevance of methods for robot arm non-colliding path planning, including the

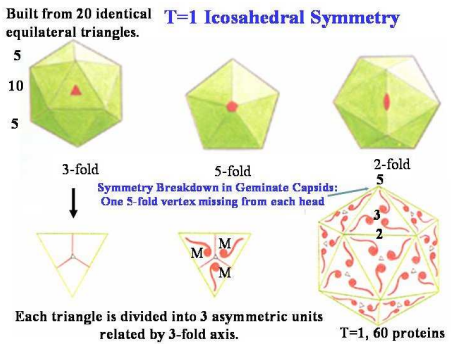

FIGURE 1. Basic Viral Structure.

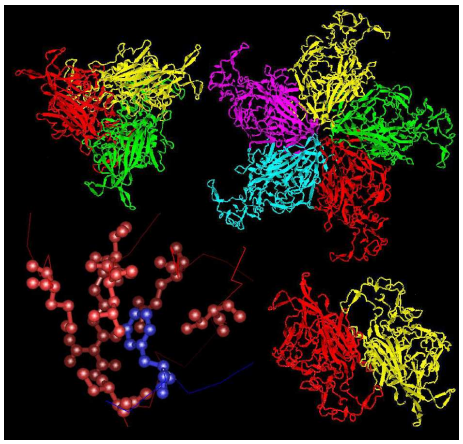

FIGURE 2. Dimer, trimer and pentamer interactions and close-up of atomic interaction

roadmapping of configuration spaces of mechanisms; [**14, 7, 61, 74, 75, 76, 2**] determining ideal membership, determining or imposing symmetries and invariants for specific classes of ideals obtained from geometric constraints; and algebraically representing the solutions of specific global polynomial constraints as the equilibria of local game-theoretic mechanisms [**63, 60**].

1.1. Virus Preliminaries. The viral shell is important in that it packages viral "life" i.e, the genomic nucleic acid, which could be single stranded DNA (*ssDNA*), double-stranded DNA, or RNA. However, in many cases, viral shell assembly occurs with no interference from the enclosed genetic material: empty shells, or shells packaging incomplete genomic material form with equal facility [**1**], a fact that simplifies the modeling. A symmetric shell [**21**] is a consequence of its consisting of (almost) identical monomers. The predominant structure of viral shells is icosahedral since the exact five-fold, three-fold and two-fold point-group symmetry of the icosahedron permits the *quasi-equivalent* symmetry [**16**] required to construct structures with a large number of monomers (see Figures 1 , 12, 11, 9). The number of monomers for each vertex of each triangle of the (20-triangle) icosahedron is refered to as the (typically small) '*T*' number: a T=1 virus shell has 60 identical monomers, a T=7 virus shell has 420 monomers etc. Our focus here is mainly on ssDNA T=1 viruses. See Figures 1 11, 9.

Virus assembly involves [**80**] highly specific monomer -monomer (protein -protein), - and possibly protein -genomic material *interactions*, all of which are governed by geometry or by weak forces that can be treated geometrically [**17**] (see Figure 2). More specifically, the final viral structure can be viewed formally as the solution to a system of geometric constraints that translate to algebraic equations and inequalities.

1.2. State of the Art. While there is a well developed structure theory of *complete* viral shells [**21, 16**], verified by X-ray crystallography and other experimental data, the *processes* of viral shell assembly are poorly understood. From an experimental point of view, this lack of understanding is due to the extreme speed of the assembly so that wet-lab snapshots of intermediate sub-assemblies are generally unsuccessful.

From a modeling point of view, this lack of understanding is due to the fact that existing computational models [**8, 9, 10, 80, 79, 78, 58, 41, 59, 48**] generally involve dynamics of (simplified versions) of virus assembly (further description of these approaches and comparison with our approach can be found in [**66**] Dynamics are currently used even when the assembly models only seek to elucidate the structure of *pathways*, See Section 4. See Figures 13, 14. See Section 4 for definition. By carefully defining the probability space, we obtain the probabilities of pathway trees/dags *that are known to result in successful assemblies* using a purely static model based on geometric constraints.

Models whose output parameters are defined only as the end result of a dynamical process are computationally costly, often requiring oversimplifications to ensure tractability. In addition, such models are also not easily tunable or refinable since their input-to-output function is generally not analyzable and therefore do not provide a satisfactory conceptual explanation of the phenomenon being modeled.

2. Contributions and their Significance

Contribution 1: (Section 4) We describe a mathematical model of viral shell assembly whose *input parameters* are: information extracted from (a) the geometric structure of the coat protein monomer that forms the viral shell, including all relevant (rigid) conformations; (b) the geometric and weak-force interactions - between pairs of monomers - that drive assembly (see

Figure 2); and (c) (optional) the neighborhood structure of the complete viral shell. The latter is crucial for a focused model that *only* deals with *those* pathways that are known *apriori* to lead to a complete viral shell. However, the model can be generalized to the case where (c) is not part of the input and unsuccessful assemblies are included.

The output information sought from the model: first, the probability of a specific successful assembly pathway that incorporates a specific subassembly and leads to the complete viral shell and has bounded total *effort*; in short, a probability distribution over successful, bounded effort assembly pathways that incorporate certain substructures; this has a straightforward generalization ([**66**]) to a distribution over all possible assembly pathways (not necessarily successful) within an effort bound. The model satisfies the following requirements.

(i) (Section 4) The description of the model - i.e. the input-to-output function - is static, i.e. does not rely on dynamics of the assembly process. This is essential for forward analyzabilty.

(ii) The assumptions of the model are mathematically and biochemically justifiable. These justifications and rigorous comparisons of the model with existing models of viral shell assembly are given in [**66**].

(iii) (Section 5) We show that the model is computationally tractable, i.e. *we develop an new, efficient algorithm for computing (a provably good approximation of) the pathway probability distribution. The required algorithms are crucial modificiations of state-of-the art geometric constraint decomposition algorithms*. This is essential for backward analyzability which is needed for two reasons: first, for iteratively refining the model so that its output matches known biochemical information or experimental results; and second, for engineering a desired output, for example engineering the monomer structure to prevent/encourage certain subassemblies, inorder to force certain pathways to become more likely than others, or to prevent successful assembly.

(iv) Preliminary simulation results (not given here, see [**66**]) show that, in principle, the model's predictions are qualitatively consistent with known studies of viruses. More conclusive biochemical validation using 3 carefully chosen, ssDNA T=1 viruses is in process [**66**].

Contribution 2: Overall, the paper provides an indication of the direct, mutually beneficial interplay between (a) the concepts underlying macromolecular assembly and (b) established as well as novel concepts from combinatorial and computational algebraic geometry and algebraic complexity. Several promising open problems are indicated in [**66**].

3. Geometric constraint solving background

Geometric constraint systems arise in a wide variety of applications including robotics, mechanical computer aided design, and teaching geometry For recent reviews of the extensive literature on geometric constraint solving see, e.g, [**36, 44, 27, 64**]. More relevantly, geometric constraints are used in molecular conformational structure determination and representation [**22**], [**25**], [**18**]. Most of the geometric constraint solvers so far deal with 2D constraint systems, although some of the newer approaches including [**34, 35, 38, 39**] [**37, 13, 52**] [**33, 47, 51, 72, 62, 71**], extend to 3D constraint systems.

A *geometric constraint system* consists of a finite set of geometric objects and a finite set of constraints between them. The constraints can usually be written as algebraic equations and inequalities whose variables are the coordinates of the participating geometric objects. For example, a distance constraint of d between two points (x_1, y_1) and (x_2, y_2) in 2D is written as $(x_2 - x_1)^2 + (y_2 - y_1)^2 = d^2$

A *solution or realization* of a geometric constraint system is the real algebraic variety or (set of) real zero(es) of the corresponding algebraic system. In other words, the solution is a class of valid real instantiations of (the position, orientation and any other parameters of) the geometric elements such that all constraints are satisfied. Here, it is understood that such a solution is in a particular geometry, for example the Euclidean plane, the sphere, or Euclidean 3 dimensional space. A constraint system can be classified as *overconstrained*, *well-constrained*, or *underconstrained*. Well-constrained systems have a finite, albeit potentially very large number of *rigid* solutions; their solution space is a zero-dimensional variety. Underconstrained systems have infinitely many solutions; their solution space is not zero-dimensional: it is called a *conformational or configuration space* A *roadmap* of this conformational set (space) – capturing connectivity and representing conformational regions of topologically distinct classes of configurations – is usually part of the realization. Overconstrained systems do not have a solution unless they are *consistently overconstrained*. Well or overconstrained systems are called *rigid* systems.

The question of "to what extent can geometric constraint problems be approached combinatorially?" is important. Since a significant proportion of the results of this paper rely on combinatorial approaches, we discuss these and their limitations here.

3.1. Constraint Graphs and Degrees of Freedom.

A geometric constraint graph $G = (V, E, w)$ corresponding to geometric constraint problem is a weighted graph with n vertices (representing geometric objects) V and m edges (representing constraints) E; $w(v)$ is the weight of vertex v and $w(e)$ is the

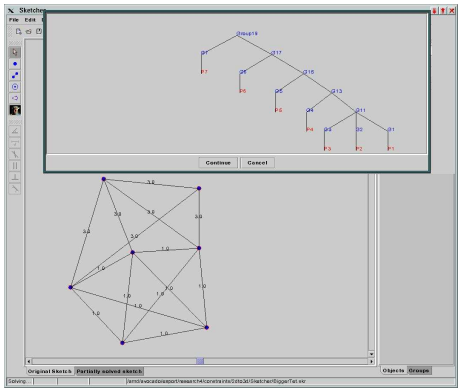

FIGURE 3. Input 3D constraint system, output decomposition or DR plan (using FRONTIER [**72**])

weight of edge e, corresponding to the number of *degrees of freedom (dofs)* available to an object represented by v and number of degrees of freedom removed by a constraint represented by e respectively.

For example Figure 3 shows a 3D constraint system and corresponding graph and Figure 4 shows a 2D constraint graph. All of these examples involve only points and distances: see [**62, 64**] for a variety of examples including other objects and constraints.

Note that the constraint graph could be a *hypergraph*, each hyperedge involving any number of vertices. A subgraph $A \subseteq G$ that satisfies

(3.1)
$$\sum_{e \in A} w(e) + D \geq \sum_{v \in A} w(v)$$

is called *dense*, where D is a dimension-dependent constant, to be described below. The function $d(A) = \sum_{e \in A} w(e) - \sum_{v \in A} w(v)$ is called *density* of a graph A.

The constant D is typically $\binom{d+1}{2}$ where d is the dimension. The constant D captures the degrees of freedom of a rigid body in d dimensions. For planar contexts and Euclidean geometry, we expect $D = 3$ and for spatial contexts $D = 6$, in general. If we expect the rigid body to be fixed with respect to a global coordinate system, then $D = 0$.

Next we give some purely combinatorial properties of constraint graphs based on density. These will be later shown to be related to properties of the corresponding constraint systems.

A dense graph with density strictly greater than $-D$ is called *overconstrained*. A graph that is dense and all of whose subgraphs (including itself) have density at most $-D$ is called *wellconstrained*. A graph G is called *well-overconstrained* if it satisfies the following: G is dense, G has atleast one overconstrained subgraph, and has the property that on replacing all overconstrained subgraphs by wellconstrained subgraphs, G remains dense. A graph that is wellconstrained or well-overconstrained is said to be a *cluster*. A dense graph is *minimal* if it has no dense proper subgraph. Note that all minimal dense subgraphs are clusters but the converse is not the case. A graph that is not a cluster is said to be *underconstrained*. If a dense graph is not minimal, it could in fact be an underconstrained graph: the density of the graph could be the result of embedding a subgraph of density greater than $-D$.

To discuss how the graph theoretic properties based on *degree of freedom (dof) analysis* described above relate to corresponding properties of the corresponding constraint *system*, we need to introduce the notion of *genericity* of e.g, [**20**]. Informally, constraint system is generically rigid if it rigid for most of choices of coefficients of system. Formally, a property is said to hold *generically* for polynomials $f_1, \ldots, f_n$ if there is a nonzero polynomial P in the coefficients of the f_i such that this property holds for all $f_1, \ldots, f_n$ for which P does not vanish.

Thus the constraint system E is generically rigid if there is a nonzero polynomial P in the coefficients of the equations of E - or the parameters of the constraint system - such that E is solvable when P does not vanish. For example, if E consists of distance constraints, the parameters are the distances. Even if E has no overt parameters, i.e, if E is made up of constraints such as incidences or tangencies or perpendicularity or parallelism, E in fact has hidden parameters capturing the extent of incidence, tangency, etc., which we consider to be the parameters of E.

According to Laman's theorem [**45**] in 2D, if all geometric objects are points and all constraints are distance constraints between these points then any minimal dense cluster represents a generically rigid system. However, in 3D or in 2D with other constraints such as angle constraints, a generically rigid system always gives a cluster, but the converse is not always the case. In fact, there are well-constrained clusters whose corresponding systems are not generically rigid and are in fact generically not rigid.

Another standard example, in 4 dimensions the graph $K_{7,6}$ representing distances is minimal dense, and hence a cluster, but it does not represent a generically rigid system.

In fact, there is no known, tractable characterization of generic rigidity of systems for 3 or higher dimensions, based purely properties of the constraint graph [**31**].

NOTE: Having noted these problems, we will nevertheless rely heavily on combinatorial dof analysis of constraint graphs - carefully augmented by some checks and corrections for algebraic dependences such as the

"bananas" and "hinge" problems [**31**], given in [**71**], to determine generic rigidity constraint systems; hence from now on we will use the terms *rigid system* and *cluster* interchangeably.

3.2. The need for decomposition: DR-plans and their properties. Now we describe a structure called the DR-plan which is crucial for our viral assembly pathway model. These structures are natural decompositions of geometric constraint systems and one of their many motivations (see [**36, 38, 39, 47, 33, 62, 64**]) is that the overwhelming cost of solving a geometric constraint system is the size of the largest subsystem that is solved using a direct algebraic/numeric solver. This size dictates the practical utility of the overall constraint solver, since the time complexity of the constraint solver is at least *exponential* in the size of the largest such subsystem.

The DR-planner is a graph algorithm that outputs a *decomposition-recombination plan (DR-plan)* of the constraint *graph*. In the process of combinatorially constructing the DR-plan in a bottom up manner, at stage i, it locates a wellconstrained subgraph or cluster S_i in the current constraint graph G_i, and uses an abstract *simplification* of S_i to to create a transformed constraint graph G_{i+1}. The solution to any subsystem in the DR-plan can be (recursively) *recombined* by solving other small subsystems. Such a recombination is straightforward, provided all the subsystems are generically rigid (have only finitely many solutions).

Formally, a DR-plan of a constraint graph G is a directed acyclic graph (DAG) whose nodes represent clusters in G, and edges represent containment. The leaves or sinks of the DAG are all the vertices (primitive clusters) of G. The roots or sources are a complete set of maximal clusters of G. See Figures 3 and 4. There could be many DR-plans for G. An *optimal* DR-plan is one that minimizes the maximum fan-in. The *size* of a cluster in a DR-plan is its fan-in (it represents the size of the corresponding subsystem, once its children are solved).

The DR-plan additionally incorporates another partial order called the *solving priority order*, which is consistent with the DR-plan's DAG order, but is more refined. This is particularly useful in 3D for correcting inaccurate dof analyses [**71**]. The intent is that clusters that appear later in the order need to be solved after the clusters that appear earlier. In fact, the nodes in such a DR-plan may not be independent clusters that appear in the original constraint graph or constraint system. They become wellconstrained clusters only in the transformed constraint system (resp. graph) after earlier clusters in the solving order are already solved (resp. simplified).

A few other properties of DR-plans are of interest. We would like the *width* i.e, *number* of clusters in the DR-plan to be small, preferably linear in the size of G: this reflects the complexity of the planning process

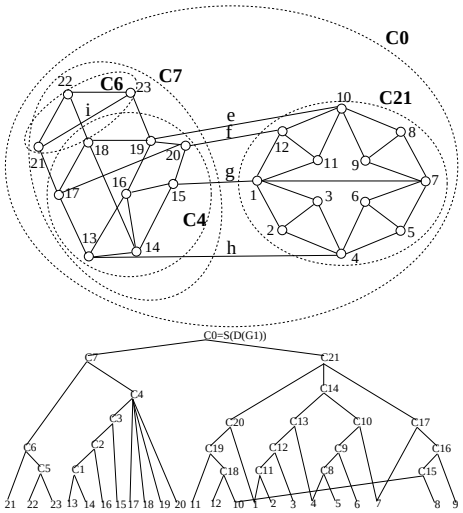

FIGURE 4. 2D constraint graph G1 and DR-plan; all vertices have weight 2 and edges weight 1

and affects the complexity of the solving process that is based on the DR-plan. Since non-minimal dense subgraphs could be misclassified as clusters, and for other reasons such as correcting misclassifications due to algebraic overconstraints, correcting combinatorial overconstraints, ease of solving and for updating the constraint system, [**47, 33, 62, 71, 68**], it is desirable for DR-plans to at least have the *cluster minimality* property: i.e., for any node in the DR-plan, no proper subset of its children induces a cluster. Another desirable property is that the DR-plan *incorporate an input partial decomposition*. I.e., given an input DAG P whose nodes are subgraphs of G and whose edges represent containment, a DR-plan of every node in P should be embedded in the output DR-plan for G.

All properties defined above for DR-plans transfer as performance measures of the *DR-planners* or DR-planning algorithms. It is shown in [**47**], that the problem of finding the optimal DR-plan of a constraint graph is NP-hard, and approximability results are shown only in special cases. Nonapproximability results are not known. However, most DR-planners make adhoc choices during computation (say the order in which vertices are considered) and we can ask of how well (close to optimal) the *best* computation path of such a DR-planner would perform (on the worst case input). We call this the *best-choice approximation factor* of the DR-planner.

4. The Virus Assembly Model

The main idea is to use a geometric constraint formulation of the driving forces that assemble a viral shell and hold it together. The virus assembly model

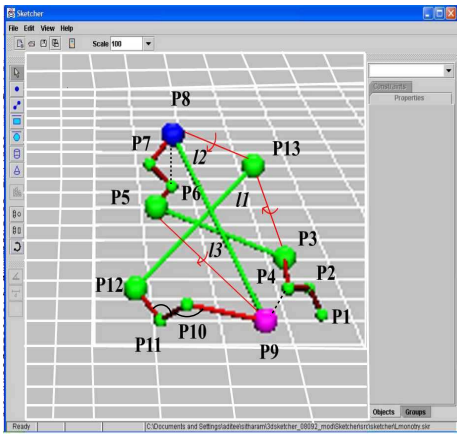

FIGURE 5. Example monomer primitives and constraints. Balls (points) - atomic markers; Green line segments - variable length bonds; Arrows - torsion angles between green line segments (primary structure) Red - distances representing fixed length bonds (primary structure), Arcs – angles (primary structure), Dotted lines – distances (weak force) (using FRONTIER [**72**])

will leverage the fact that forces can be treated as geometric constraints using so-called *tensegrity frameworks* [**23**], [**6**], [**19**], with local geometric constraints enforcing stability, or global (algebraic) constraints minimizing potential energy. This is used to give a formal, static definition of a viral assembly pathway (i.e, a partial order of subassemblies), as a certain type of DR-plan of the underlying constraint system. We then give an effort rating for each pathway indicating the difficulty of subassembly formation along the pathway.

The input parameters of the viral assembly model are viewed as a geometric constraint system which is then represented as a geometric constraint graph based on degrees of freedom as described in Section 3. More specific definitions below.

The original architects of the currently accepted so-called *quasi-equivalence* theory of viral structure and viral shell self-assembly, Caspar and Klug, [**16**], by their own admission, *derived their inspiration from tensegrity structures.* However, barring this inspirational connection, a precise formulation of viral structure and assembly in terms of tensegrity has not been investigated or exploited.

4.1. Formal definition of the Model Input. A *viral geometric constraint system* is specified in 4 parts.

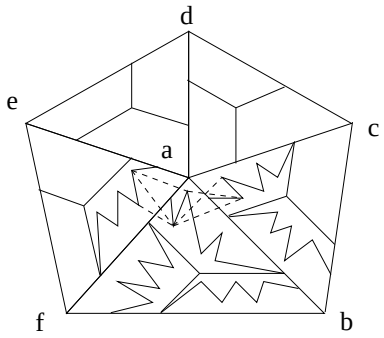

FIGURE 6. Pentamer interface constraints

It is based on the assumption that the viral shell is icosahedral and made up of 60 identical protein molecules as in Figure 1. Hence both the *monomer structure constraints* and the *interface constraints* ((1) and (2) below) are specified just for a single reference monomer.
(1) The *monomer structure constraint system* (see Figure 5): the primitive objects in this system are points representing essential *atomic markers* and line segments representing essential *bonds* on the backbone and side chains of the protein monomer. The constraints in this system are of three distinct types.
(a) The *primary stucture* constraints consisting of distance, angle, torsion angle intervals between the points and line segments; these represent bond lengths, bond angles and torsion angles involving the corresponding atomic markers and bonds. These constraints typically form a polygonal chain with side chains.
(b) The *monomer weak force* constraints consisting of distance intervals and tensegrity forces between the points; these represent hydrogen bonds and other weak forces between atomic markers. This is also called the *monomer contact map*.
(c) Required relative orientation constraints on subsets of 4 points, expressed as polynomial (determinantal) inequalities: these represent allowed chiralities of the corresponding atomic markers and are used pick out the allowed (rigid) conformations of the monomer. These constraints can be replaced in some cases by extra distance constraints between the relevant points (atomic markers).
(2) The *interface constraint system* (see Figures 5, 6, 7, 8): This consists of 3 constraint systems called A-P(A), A-T(A) and A-D(A), involving 4 monomers A, P(A), T(A) and D(A). Each constraint system is between the reference monomer A and one of its reference neighbors P(A), T(A) or D(A), across a pentamer, trimer or dimer interface respectively. The monomer A participates in two more symmetric pentamer and trimer interfaces P^{-1}(A)-A and T^{-1}(A)-A, with monomers P^{-1}(A) and T^{-1}(A), but these need not be specified as they can be inferred from the constraint systems A-P(A), A-T(A), where P^{-1}(A) (resp. T^{-1}(A)) takes

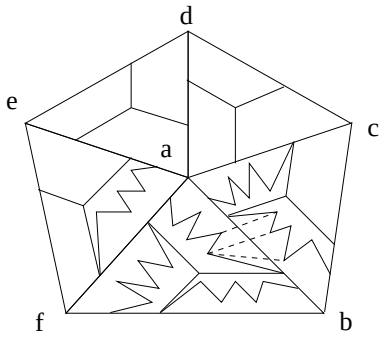

FIGURE 7. Dimer interface constraints

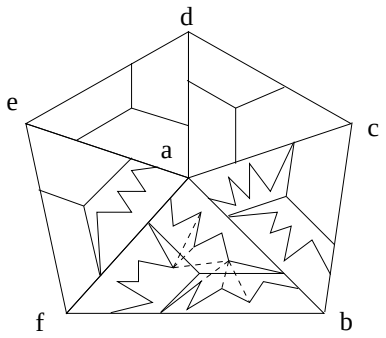

FIGURE 8. Trimer interface constraints

the place of A, and A takes the place of P(A) (resp. T(A)). Each constraint system consists of distance intervals or tensegrity forces. Each of these constraints involves one point in A and one point in P(A), D(A) or T(A), depending on the interface. These represent the weak forces or interactions between the monomers that drive assembly. These are also called the *interface contact maps*. (3) The *neighborhood structure* (see Figures 9:) A regular, directed graph with labeled vertices representing the monomers and edges representing the icosahedral adjacency structure imposed by the interfaces. The labels are typically of the form Nx, where N is the label of one of the icosahedral triangular faces and x is one of the icosahedral vertices. Nx is the monomer closest to vertex x and face N. As described above, each vertex A has 3 outgoing edges A-P(A), A-T(A) and A-D(A), representing its forward pentamer, forward trimer and dimer interfaces and 2 incoming edges P^{-1}(A)-A and T^{-1}(A)-A, representing its backward pentamer and backward trimer interfaces (the backward dimer is generally irrelevant and is omitted).

As in the case of the monomer chirality constraints, these constraints can be replaced in some cases by extra distance constraints between the relevant points (atomic markers). These constraints represent external restrictions on the conformations of the complete viral shell that are not captured by the input (1c) restrictions on monomer conformations. See [**66**] for the method by which these constraints are chosen.

4.2. Formal definition of Pathways and Effort ratings. A *pathway* P for a viral constraint graph G is a DR-plan for G (see Section 3 and Figures 4 3) that additionally satisfies 3 properties.
(i) No pair of clusters in the pathway intersect on a non-trivial cluster (for DR-plans, this invariant is slightly weaker – see below) unless one is contained in the other;
(ii) No cluster is formed entirely by overlap constraints between the children (i.e., only the overlap constraints between the children should be inadequate to form the parent cluster).

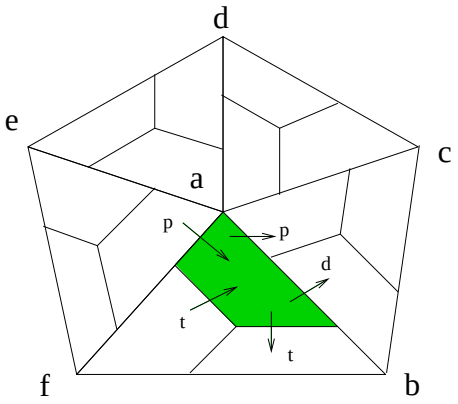

FIGURE 9. Icosahedral neighborhood graph - the 3rd input parameter

Instead of the cluster-minimality property of DR-plans, pathways should satisfy the following:
(iii) Every cluster in the pathway has the *overlap-minimality* property defined below.

Note that properties (i) and (ii) are inconsistent with the cluster-minimality property of DR-plans. Some (usually stronger) form of cluster-minimality check has to be performed by the DR-planner before subgraphs can be correctly classified as clusters: however, due to Properties (i) and (ii) above, those clusters may have to be enlarged, before they are put into the pathway. See Section 5 for a description of this process. Intuitively, Properties (i) and (ii) assert that when two clusters overlap on a primitive object, then they are linked by that object. I.e., a biophysically valid decomposition cannot in general make "copies" of the variables corresponding object, treat them independently in separate clusters and then equate them as an overlap constraint, unless this overlap constraint is does not force the participating clusters to form a cluster. This is crucial in giving

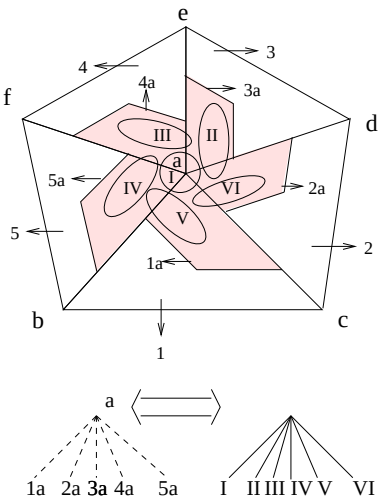

FIGURE 10. Clusters that overlap parts of monomers; decomposition of stable pentamer a into non-monomer clusters

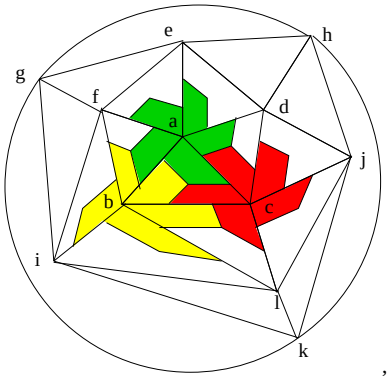

FIGURE 11. Vertex numbers: trimers of pentamers

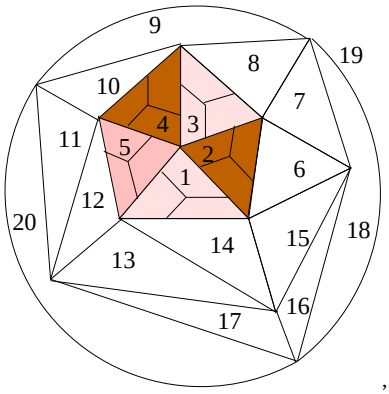

FIGURE 12. Facenumbers: pentamer of trimers

a biophysically valid measure of *effort* required form clusters defined below.

DEFINITION 4.1. A cluster C with children $C_1, \ldots, C_m$ is *overlap-minimal* if for any cluster C' formed by a proper subset of child clusters $C_1, \ldots, C_k$, $1 < k < m$, it holds that: C' along with the remaining child clusters $C_{k+1}, \ldots, C_m$ form the cluster C using overlap constraints alone. See Figure 17.

Notice that the Property (i) above guarantees the the *width* (see Section 3) of a pathway and in fact the total number of clusters in a pathway is roughly linear in the number of vertices in the viral constraint graph.

Assumption 1: Overlap-minimal clusters of a viral constraint graph G have a constant *size or fan-in* (see Section 3), i.e., number of children, which is independent of the number of vertices of G. This is based on observations of a large number of known viral structures. See [**66**] for detailed biochemical justifications of this and other model assumptions.

Notice that clusters could be constructed from parts of monomers. I.e., clusters are not necessarily *subassemblies*, i.e., consisting of whole monomers. This provides our pathways a sophistication that appears to be essential for making good predictions. See Figure 10.

Each cluster C does have a subassembly denoted *sub*(C) loosely associated with it, namely, that subassembly consisting of that set of monomers that the cluster overlaps.

A cluster C represents a (λ)-*stable* subassembly sub(C) only if a suficiently large fraction $(\geq \lambda)$ of the points or atomic markers in sub(C) are actually present in C. A subassembly is *stabilizable* if a sufficiently large portion of it is a cluster. While a subassembly may be stabilizable, it may not be stable at a particular point in a pathway, i.e., when the corresponding cluster in the pathway does not encompass a large enough portion of the subassembly. Furthermore, by our definition of pathways, even a minimal stable subassembly (i.e., no subassembly contained in it is stable) may have several pathways to *its* formation.

Thus each pathway P embeds a unique coarser sub-pathway consisting of those clusters that represent stable subassemblies. We call this sub-pathway the embedded *stable subassembly pathway* in P. Figures 11, and 13 show a stable subassembly pathway based on 2 stable subassemblies: pentamers and trimers of pentamers. Figures 12, and 14 show a stable subassembly pathway based on 2 stable subassemblies: trimers and pentamers of trimers.

Note that a pathway could have single root or many roots or sources depending on whether the complete viral constraint system is (or forces the shell to be) rigid.

Assumption 2: The constraints in a viral constraint graph G are sufficient to enforce a stable entire assembly or viral shell; i.e., the pathways for G will contain one giant root or source cluster C whose associated subassembly sub(C) is the entire assembly. In other words, C contains the bulk (at least λ fraction) of the entire assembly or shell. The other sources of the pathway represent other smaller rigid components that appear within flexible sidechains and surface decorations of the shell.a (See [**66**] for justification of biochemical assumptions)

Note that Assumption 2 implies that for any pathway P of a viral constraint graph G, the embedded stable subassembly pathway has a single root.

We have just defined *labeled* pathways (i.e., where the leaves or geometric primitives are labeled). But we will be primarily interested in the probabilities associated with unlabeled pathways or *pathway isomorphism classes* for a viral constraint graph G. These are defined in the obvious manner as equivalence classes of the equivalence relation *perm* between pathways, induced by the automorphisms of G (i.e, those permutations of the vertices of G that preserve the geometric object types that the vertices represent; and additionally preserve the edges, including the geometric constraint types associated with the edges) More specifically, for two pathways P and Q of G are related by *perm*(P, Q) if P can be obtained from Q by applying some permutation - from the automorphism group of G - to the leaves or sinks, i.e., the primitive geometric objects of P.

Each pathway is assigned an *effort rating*, which will be formally based on (and inversely related to) the inherent algebraic complexity of the subsystems that appear in the pathway or alternately, the difficulty of formation of the clusters. We would like this to be a complexity measure satisfying several requirements. (i) It should be somewhat independent of the properties specific to known algorithms for solving the subsystems, or other arbitrary variables, for example elimination order (the complexity measure should, for example, assume the best order) (ii) Furthermore, we would like the value of this measure to be polynomial time computable, given a subsystem. (iii) On the other hand, as explained in [**66**] it would be desirable to have an algorithm for solving the subsystem whose running time provides a reasonably close upper bound on the value of this measure atleast for a large, well-defined class of subsystems. (iv) Finally, as mentioned earlier in the context of overlap-minimality, the measure should have a biophysical justification, for example, it should be related to the energy barriers that are overcome in physically forming the cluster (solving the subsystem) or the geometric or topological precision of the motions of the child clusters that are necessary to form the cluster.

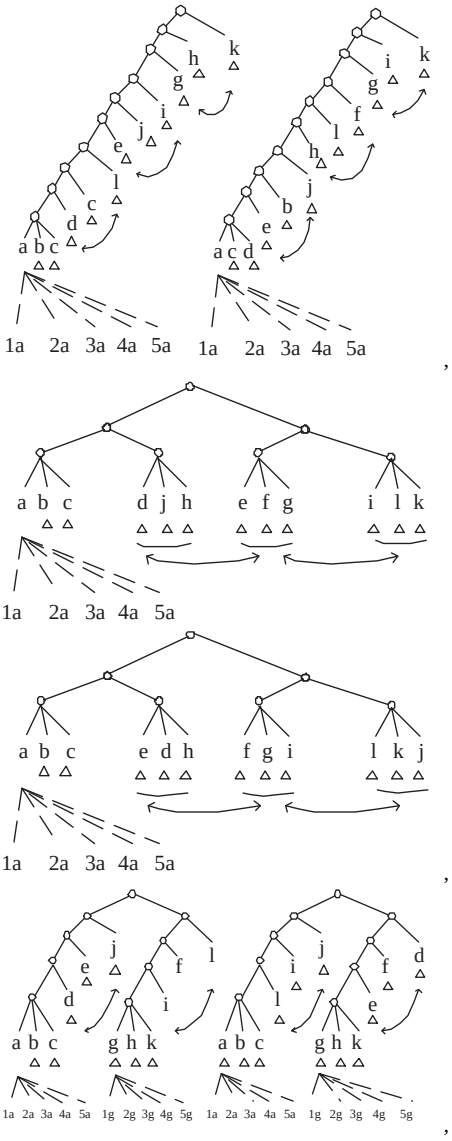

FIGURE 13. Pathways based on pentameric stable subassemblies

These considerations and the fact that viral geometric constraint systems are typically sparse, motivate us to use a refined version of the familiar BKK bound [**11**] as an appropriate measure. This bounds the number of solutions of a polynomial system, is usually superior to the Bezout bound for sparse systems,

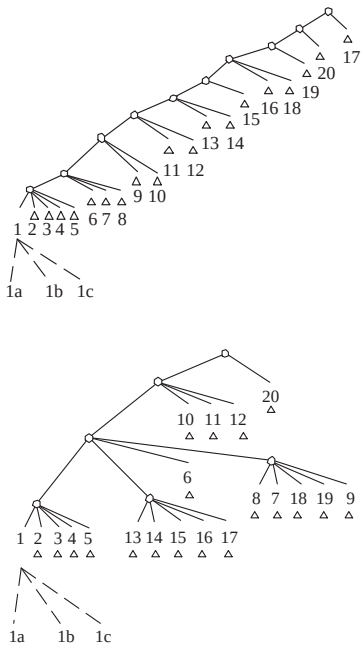

FIGURE 14. Pathways based on trimeric stable subassemblies

is computed as their *mixed volume* and is by now used as a standard tool in sparse polynomial resultant computation, elimination and root finding algorithms, both symbolic and semi-numeric [**15**], [**24**], [**40**], [**73**]. Although we are interested only in (isolated) real solutions, we find the BKK bound better for our purposes although it bounds the number of complex roots rather than the the Khovanskii fewnomial [**43**] bounds which actually estimate real roots. First some standard definitions.

The *Newton polytope* $New(P)$ of a k-variate polynomial P in variables $x_1, \ldots, x_k$ of total degree d, denoted as $P(x) = \sum_{\alpha} a_\alpha x^\alpha$ (where $\alpha = \alpha_1, \ldots, \alpha_k$, $\sum_i \alpha_i \leq d$ and x^α denotes $x_1^{\alpha_1} \ldots x_k^{\alpha_k}$) is the convex hull of the points $\alpha \in \mathbb{Z}^k$ for which the coefficient a_α is nonzero. The *mixed volume* of a set S of polynomial equations $P = 0$ in k variables is defined as an alternating sum of the volumes of the Minkowski sums of subsets the Newton polytopes $New(P)$: $MV(S) = \sum_{Q \subseteq S} (-1)^{|Q|} MS_{P \in Q}(New(P))$ where MS denotes the Minkowski sum. The *BKK* (Bernstein-Kuchirenko-Khovanskii) bound [**11**] says that the number of solutions to a system S of k polynomials in k variables is bounded by $MV(S)$ (this is an equality if the polynomials are generic, i.e., if their coefficients are algebraically independent).

Mixed volume computations are done using so-called *mixed subdivisions* of the underlying polytopes. Using these, algorithms that are polynomial in the degree and number of terms, and only singly exponential in the number of variables [**15**] are known for computing the resultant and solutions of sparse systems. Moreover, mixed subdivisions give rise to a class of numeric algorithms called *polyhedral homotopy* algorithms [**40, 73**] that find all roots, and which run for time roughly proportional to the mixed volume. Together with the fact that clusters in pathways (see definition above), are not underconstrained, i.e., they suit the BKK bound, and have only finitely many solutions, and the fact that they are overlap-minimal, along with Assumption 1 stating that overlap-minimal clusters of viral geometric constraint systems have constant size, our above Requirements (i), (ii) and (iii) are therefore met by the mixed volume and the BKK bound. For a discussion of how well the measure addresses Requirement (iv), see [**66**].

We now describe the effort rating of a pathway based on the BKK bound. The crucial measure is the effort rating of a cluster in a pathway and it incorporates three intuitive ideas. (a) Primary structure constraints are - in a physical sense - "solved" constraints, i.e, the underlying monomers and subassemblies inherently satisfy them; hence the effort in forming a cluster is based on the other unsolved (weak force) constraints both between its child clusters. (b) Overconstrained clusters offer different ways of resolution and the best choice is assumed in the computation of effort. (c) Although the clusters in pathways are overlap-minimal, they could be further *algebraically reduced* in well-defined ways that are biophysically justifiable: for example, as mentioned earlier under the context of overlap-minimality, the decomposition cannot indiscriminately "make copies" of the variables corresponding to overlapped objects.

One such method of algebraic reduction is shown in Figure 15 [**69**]. On Left: solving by first fixing 6 dofs (degrees of freedom) of 1 cluster resolving 2 of the overlaps; and then solving the algebraically reduced system consisting of 1 overlap (3 constraints) and 3 distance constraints simultaneously for all 3 rotational dofs for each of the other two clusters – *6 constraints, 6 variables.* On Right: first solving one algebraically reduced system of a triangle of distances, one from each cluster, between the overlapped points, fixing the 6 dofs of this new triangular cluster and resolving the 3 overlaps (this fixes all but 1 rotational dof for each of the 3 original triangular cluster); then solving the second algebraically reduced cluster 3 distance constraints simultaneously for 1 rotational dof per original triangular cluster – *only 3 constraints and 3 variables.*

DEFINITION 4.2. Let S be an overlap-minimal, algebraically-reduced cluster (subsystem) of a viral constraint system, and let S contain a set Q of primary

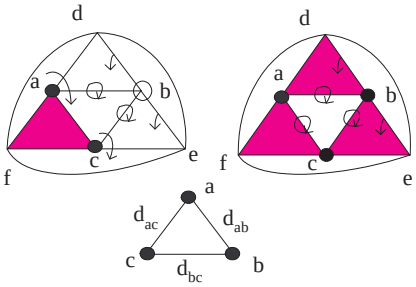

FIGURE 15. Example of algebraic reduction: Octahedron split into 3 triangle clusters joined by 3 overlap points and 3 distance constraints – see description in text.

structure constraints. Let $S_1, \ldots, S_k$. be the child clusters (subsystems) of S; hence k is the size or fan-in of S. S could be overconstrained, but any of its wellconstrained overlap-minimal subsystems must involve all of $S_1, \ldots, S_k$, due to overlap-minimality. The *effort rating* of S is denoted $Ef(S)$ and defined below. Here the minimum runs over all well-constrained overlap-minimal subsystems S' of S. $Ef(S) = w_1 2^{w_2 k}$ $+ \min_{S' \subseteq S} w_3(\prod_i Sol(S_i))/Sol(S') + w_4 Sol(S')$, where $Sol(S') = MV(S') - MV(S' \cap Q)$, and $w_1, \ldots w_4$ are the viral model's tunable parameters.

Here the first summand in $Ef(S)$ is simply the conjectured (and observed) time complexity of solving a cluster S of size or fan-in k, i.e., a sparse system S in $O(k)$ variables. The second summand represents the ratio of the size of the search space to the size of the solution space, and the second summand roughly represents the size of the solution space or the effort in listing all the solutions of S' assuming that the equations in $S' \cap Q$ (primary structure constraints) are already solved. This intuitively justifies the presence of both summands in the definition of $Ef(S)$.

4.3. The Pathway Probability Space. Our probability space is the set of all labeled pathways of a complete viral constraint graph. By definition of effort, and the isomorphism classes of pathways, isomorphic pathways have the same effort. For any fixed effort value, we assume a uniform probability distribution over all labeled pathways of that effort.

Assumption 3: An underlying natural assumption is an additional inverse relationship between a pathway's probability and its effort (see [**66**] for biochemical justifications of modeling assumptions). While this assumption intuitively motivates the results of this paper, they are not formally based on this assumption. For fixed effort bounds, we are generally interested in the probability of an unlabeled pathway or a pathway isomorphism class, which is proportional to the size of the isomorphism class. This type of probability and effort computation is sufficient to answer our focused questions about pathways. For a more detailed argument, see [**66**].

4.4. A Restricted Model. While the formal definition of pathways and effort ratings given above are well-posed for the general viral constraint systems given above, the definitions are meaningful only for restricted viral constraint systems that include no inequalities and involve tensegrity forces in highly limited ways. Inorder to include these, we need a suitable generalization of clusters (and hence pathways) to include underconstrained systems. See [**66**]. More importantly, the tractable and accurate computational simulation of the model given in Section 5 applies only to the restricted viral constraint systems.

5. Tractable and Accurate Computational Simulation of the Model

It is not viable to give a combinatorial enumeration of pathways of viral constraint graphs – even those that satisfy the restrictions of Section 4.4 – using generating functions or in any manner give a closed form analysis of the probability distribution over pathways of bounded total effort. Also, despite the overall icosahedral symmetry, the number of possible pathway isomorphism classes is prohibitively large, and hence an exhaustive enumeration and over the pathways is not tractable.

We use the following approach. We
(1) use the fact that pathways for viral constraint graphs satisfying the restrictions of Section 4.4 are a modified DR-plan (see Section 3);
(2) modify and randomize an existing DR-planner called the *Frontier vertex* algorithm (FA) that runs in time cubic in the number of vertices of the input constraint graph, so that it generates a random valid pathway that contains a given subassembly or sub-pathway, and computes the pathway's rating efficiently;
(3) use the fact that viral constraint systems inherit the icosahedral symmetries :
(4) use the fact that DR-plans have a correspondence with distinct bases of the underlying rigidity matroid (see [**77**], [**32**], [**31**]);
(5) and the fact that matroid bases form a markov chain on which random walks converge in polynomial time to stationary distributions that permit random sampling and approximate counting [**4, 5, 12, 26, 57, 50, 29, 65**] and
(6) use (3), (4) and (5) to argue that the algorithm in (2) generates a representative sample of pathways that approximates their true probability distribution well.

5.1. Generating Random Pathways using the Frontier Vertex (FA) DR-Planner. DR-planners based on geometric constraint graphs have been proposed since

the early 90's for restricted classes of graphs that are decomposable simply by detecting certain patterns such as triangles ("triangle decomposable") [**28, 53, 54, 52**] [**46, 49**]; and based on Maximum Matching [**55, 3, 46, 44**], and rigidity matroids (for 2D points and distances) [**77, 32**]. However, prior to [**38**], the DR-planning problem and appropriate (and strongly competing) desirable properties for DR-planners were not formally defined or motivated. That paper also gives a table comparing 3 main types of DR-planners, with respect to these performance measures including those in Section 3. These performance measures were optimized by the Frontier vertex DR-plans and the corresponding DR-planner (FA DR-planner) described very briefly below [**39, 37, 47, 51, 33, 72, 71, 62, 64**].

The challenge met by the FA DR-planner (along with subsequent modifications) is that it provably meets several competing requirements. See Section 3 for definitions.
(a) It deals with key problems of constraint dependencies and rotational symmetries and hence rectifies misclassification of clusters for a large class of 2D constraint graphs containing angle and incidence constraints as well as 3D constraint graphs.
(b) For wellconstrained graphs it outputs a DR-plan with a single root representing the entire graph as a cluster: for underconstrained graphs - it outputs a complete set of maximal clusters as sources of the DR-plan.
(c) It controls the width of the DR-plan to ensure a polynomial time algorithm.
(d) It ensures the cluster-minimality of clusters in the DR-plan.
(e) It outputs DR-plans consistent with input partial decompositions.

The FA algorithm builds the DR-plan bottom up by successively locating clusters and simplifying them to create a transformed comstraint graph, called the *flow graph* or *cluster graph*. The graph transformation performed by the FA cluster simplification is described formally in [**39, 37**] that provide the vocabulary for proving the properties of FA that follow directly from this simplification. However, other properties of FA require details of the actual DR-planner that ensures them [**47, 51, 33, 72, 62, 64, 71**].

Note. A detailed pseudocode of the FA DR-planner can be found in [**62**]. Properties (a) and (e) above are proven in [**71**], [**70**] and Properties (b), (c), (d) are proven in [**47**].

5.2. Crucial Modifications for obtaining Random Pathways. The randomization procedure is straightforward and yet performs well in providing a representative sample of pathways (see discussion on simulation accuracy in Section 5.2.1). The exact randomization cannot be formally described without refering in detail to the FA pseudocode in [**47**] and [**62**]. However, it is conceptually simple and easy to describe

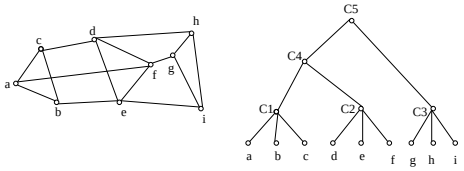

FIGURE 16. Maintaining linear width of DR-plan: see text

intuitively: everywhere that the FA DR-planner performs operations on vertices, edges or clusters in *lexicographic* order or *queue* order, this is replaced by a *random* order.

Recall from Section 4, that the additional requirements that a pathway has to meet 3 properties beyond the DR-plan properties.

Property (i) is ensured by modifying the FA DR-planner as follows.

FA achieves a linear bound on DR-plan width by maintaining the following invariant of the cluster or flow graph: *every pair of clusters in the flow graph (top level of the DR-plan) at any stage intersect on at most a rotationally symmetric subgraph.* FA does this by performing 2 operations after a new potential cluster-minimal cluster is isolated by the cluster-minimality routine (see [**47, 62, 71**]). Note that this routine is required even though we do not require cluster-minimality for pathways. It is required to ensure that the found cluster has not been misclassified due to combinatorial and algebraic overconstraints, i.e., to ensure Property (a) of the FA DR-planner (see [**47, 62, 71**]).

The first operation is an *enlargement* of the found cluster. In general, a new found cluster N is enlarged by any cluster D_1 currently in the flow graph (top level of the DR-plan), if their nonempty intersection is *not* a rotationally symmetric or trivial subgraph. In this case, N does not enter the top level of the DR-plan Only $N \cup D_1$ enters the DR-plan, as a parent of both D_1 and the other children of N. It is easy to see that the *sizes* of the subsystems corresponding to both $N \cup D_1$ and N are the same, since D_1 would already be solved.

For the example in Figure 16, when the DR-plan finds the cluster C_2 after C_1, the DR-planner will find that C_1 can be enlarged by C_2 The DR-planner forms a new cluster C_4 based on C_1 and C_2 and puts C_4 into the top level of the DR-plan, instead of C_2.

The second operation is to iteratively *combine* $N \cup D_1$ with any clusters $D_2, D_3, \ldots$ based on a nonempty overlap that is not rotationally symmetric or trivial. In this case, $N \cup D_1 \cup D_2$, $N \cup D_1 \cup D_2 \cup D_3$ etc. enter the DR-plan as a staircase, or chain, but only the single cluster $N \cup D_1 \cup D_2 \cup D_3 \cup \ldots.$ enters the top level of the DR-plan.

Ofcourse, both of these processes are distinct from the original flow distribution process that *locates* clusters.

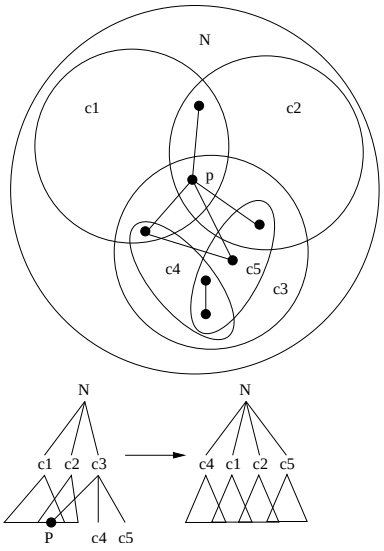

FIGURE 17. Ensuring Property (ii): C_5 and C_4 are directly made children of C, destroying C_3; the operation is performed recursively on C_4 and C_5

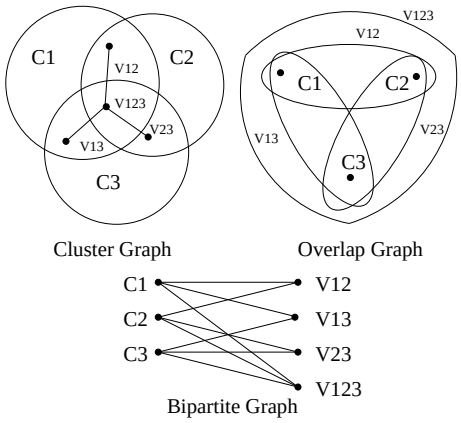

FIGURE 18. Ensuring Overlap minimality – finding minimum vertex separator on bipartite graph obtained from the overlap graph of the original cluster graph

Now, in order to ensure that the DR-plan has the Property (i) of pathways mentioned above, the enlargement process subsumes the combining process. The cluster N is iteratively enlarged by the clusters D_i so that only $N \cup D_1 \cup D_2 \ldots$ enters both the DR-plan as a common parent of N and the D_i's; and it enters the cluster graph, after removing all D_i.

Property (ii) is ensured by checking each newly found cluster N to see if it can be formed entirely by overlap constraints between the children. This requires a careful dof count using inclusion-exclusion. If overlap constraints are sufficient to form the cluster N, then the last found child cluster C of N is removed from the DR-plan and its children are directly made children of N (leaving out any children that are trivial clusters in the intersection of C and one of the other original children of N). See Figure 17.

To ensure Property (iii) or overlap-minimality of clusters, they are treated as follows after the combining step has found the cluster C that can no longer be combined with any of the clusters in the flow graph, i.e., top level clusters of the current DR-plan. Recall that overlap-minimality of C with children $C_1, \ldots, C_m$ requires that for any cluster C' formed by a proper subset of child clusters $C_1, \ldots, C_k$, it holds that: C' along with the other child clusters $C_{k+1}, \ldots, C_m$ form the cluster C using overlap constraints alone. If C does not satisfy overlap-minimality, then it is decomposed into a sub-DR-plan consisting of overlap-minimal clusters. This is done by a recursive method *Overlapmin* similar to the cluster-minimality routine of [**47**], [**62**]. This method takes as arguments the subgraph S of the cluster graph, (S does not have to be a cluster) and the set $S = \{C_1, \ldots, C_m\}$) of the overlap-minimal clusters that constitute it (we identify the set of clusters with the subgraph of the cluster graph induced by them). The base case is when S consists of only 2 clusters C_1 and C_2. In this case, if C_1 and C_2 have a nontrivial overlap, then S is returned, since it is itself overlap-minimal. If not, C_1 and C_2 are returned as the maximal overlap-minimal clusters within S. The recursion is acheived by first locating a proper subgraph S' of S that overlaps the remaining set of clusters in $S \setminus S'$ only on a trivial subgraph. This is done by running a minimum vertex separator (or cut) algorithm such as [**42**] on the *bipartite* graph obtained from the standard *overlap hypergraph* of the clusters in S. In the hypergraph, clusters are vertices and each overlap vertex is an edge; in the bipartite graph, one side consists of vertices representing vertices of the hypergraph and the other consists of vertices representing edges of the hypergraph. See Figure 18

If no such subset is located, then C is overlap minimal. If such a subset S' is located, then *Overlapmin* calls itself twice. The first call is with arguments: the subgraph induced by S' and the set of clusters $\{C_1, \ldots, C_k\}$ within S'. The method generates a sub-DR-plan of S' whose leaves are the clusters $C_1, \ldots, C_k$ and in which each intermediate cluster is overlap-minimal. If S' is itself a cluster, then it is the single root or source of this sub-DR-plan. If not, a complete set T of *maximal* clusters within S' appear as the sources. The second call is with arguments: S

and the union of the set T and the remaining clusters $\{C_{k+1} \ldots, C_m\}$ in $S \setminus S'$.

It can be formally proven that the above modifications ensure the additional Properties (i), (ii) and (iii) of pathways and do not affect the Properties (a), (b), (c), (e) guaranteed by the FA DR-planner ([**47**] and [**71, 70, 62**]). In particular, the Property (e) can be leveraged by adding a particular (labeled) subassembly or even a particular (labeled) sub-pathway as an input partial decomposition, so that the output pathway will contain them. Moreover, the pathway's effort can be computed in time directly proportional to the number of nodes in the pathway, since by the assumption on viral constraint graphs in Section 4, the number of children of a overlap-minimal cluster of a viral constraint graph is bounded by a constant and hence the effort rating of such a cluster can be computed in constant time, independent of the size of the viral constraint graph (number of atomic markers). Finally, since the FA-DR-planner's property I.e., we obtain the following theorem.

THEOREM 5.1. *On an input viral constraint graph G and a labeled subassembly or sub-pathway P the FA-DR-planner - as modified above - outputs a valid assembly pathway A for G that contains P, as well as A's effort rating, within time cubic in the number of vertices (or essential atomic markers) of G.*

Note The FA DR-planner is implemented in the (open-source, available from GNU) FRONTIER geometric constraint software [**72**], [**62**], [**64**] developed at the University of Florida. The new algorithms described here are straightforward to incorporate. Together this provides a hands-on efficient computational simulation and visualization tool for viral pathway simulation. See Figures 5, 19, 20, 3.

5.2.1. *Proving Accuracy of Simulation.* The above theorem proves the efficiency or tractability of the simulation. However, it is still necessary to show that the randomization procedure mentioned above outputs a truly random pathway; i.e., it outputs a representative sample of pathways (under a given effort bound) that accurately reflects the probability distribution over all successful assembly pathways (under the same effort bound) predicted by the formal model in Section 4. For example, it is necessary to show that bounded effort pathways obtained over m trials of the simulation accurately (with error decreasing, say, proportionally to $m^{1/k}$ for some k) represent the probability distribution over the entire set of successful, bounded effort pathways. I.e., we need to prove the following type of theorem.

Definition. Let M be a multiset of successful pathways of effort at most b that contain a particular (labeled) subassembly or subpathway P for some viral constraint graph G. Let $0 < \epsilon < 1$ be a desired accuracy. Let m_T be the number of pathways in M that are in the same isomorphism class T. Let p_T be the probability that a b-effort pathway of G that contains P belongs to the isomorphism class T. We say that M is an *ϵ-representative* sample (of b-effort pathways for G containing P) if for every pathway isomorphism class T, $|m_T/|M| - p_T| \leq \epsilon$. ♣ (end of definition).

THEOREM 5.2. *There is a fixed k such that for any $0 < \epsilon < 1$ the following holds. Let M be any set of b-effort pathways containing a (labeled) subpathway P output over independent runs by the the randomized, modified FA-DR-planner (described above) on an input viral constraint graph G. Then M is an ϵ-representative sample provided $M \geq (1/\epsilon)^k$. This holds for effort bounds b that include a reasonably large number of pathways and for subpathways P of constant bounded size which includes the case when P is a single subassembly.*

In the general case of viral constraint graph G of Section 4.4, this theorem is still a conjecture. See [**66**]. However, the observed accuracy of the sample obtained by the above randomization is supported by the fact that the theorem has been proven in [**65**] for a special class of of viral constraint graph G in which each pathway corresponds to a distinct, well-defined basis class of the rigidity matroid that is obtained from G (see [**77**], [**32**], [**31**]). The proof further leverages (1) the icosahedral symmetry of the constraint system, and (2) the fact that distinct matroid bases form a markov chain on which random walks converge rapidly (in time polynomial in the size of the original graph and in $1/\epsilon$) to stationary distributions that provide ϵ-representative samples. Various results of this type are shown in [**26**], [**29**], [**50**] for so-called balanced matroids. The proof requires the extension of these results to (1) a class of rigidity matroids; (2) where each element in the markov chain is a pathway, i.e, a distinct well-defined basis class of the underlying rigidity matroid, as opposed to a distinct basis. Finally, the proof shows that the simple randomization of the FA-DR-planner described above corresponds in a natural manner to a random walk on the above markov chain.

Simulation results using the FRONTIER geometric constraint solver [**72**], [**62**] and preliminary validation of the model are given in [**66**]. See Figures 19, 20.

References

[1] M Agbandje, R McKenna, MG Rossmann, ML Strassheim, and PR Parrish. Structure determination of feline panleukopenia virus empty particles. *Proteins*, 16:155–171, 1993.

[2] Oswin Aichholzer, Gunter Rote, Bettina Speckmann, and Ileana Streinu. The zigzag path of a pseudo-triangulation. In *Proc. 8th International Workshop on Algorithms and Data Structures (WADS) , Ottawa, Canada*, pages 377–388, 2003.

[3] S. Ait-Aoudia, R. Jegou, and D. Michelucci. Reduction of constraint systems. In *Compugraphics*, pages 83–92, 1993.

[4] D. Aldous. The random walk construction for spanning trees and uniform labeled trees. *SIAM J. of Disc. Math.*, pages 450–465, 1990.

[5] Yossi Azar, Andrei Z. Broder, and Alan M. Frieze. On the problem of approximating the number of bases of a matroid. *Information Processing Letters*, 50(1):9–11, 1994.

[6] W Whiteley B Roth. Tensegrity frameworks. *Transactions of the AMS*, 265:419–446, 1981.

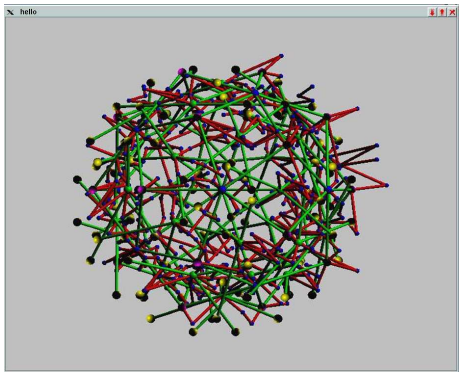

FIGURE 19. The assembled viral shell: the solved viral geometric constraint graph of Figures 5, 6, 7, and 9

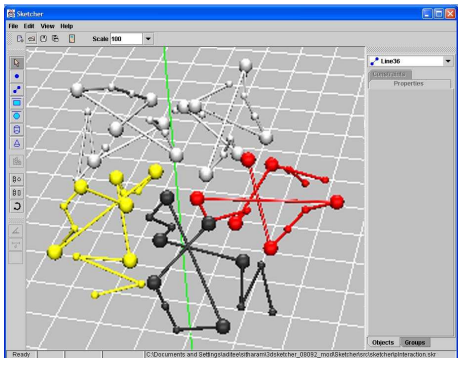

FIGURE 20. Pentamer subassembly of the viral geometric constraint graph of Figures 5, 6, 7 and 9

[7] Saugata Basu, Richard Pollack, and Marie-Françoise Roy. Computing roadmaps of semi-algebraic sets (extended abstract). In *STOC 1996*, pages 168–173.

[8] B. Berger, P. Shor, J. King, D. Muir, R. Schwartz, and L. Tucker-Kellogg. Local rule-based theory of virus shell assembly. *Proc. Natl. Acad. Sci. USA*, 91:7732–7736, 1994.

[9] B Berger and PW Shor. On the mathematics of virus shell assembly. 1994.

[10] B Berger and PW Shor. Local rules switching mechanism for viral shell geometry. *Technical report, MIT-LCS-TM-527*, 1995.

[11] Bernstein and David Naumovich. The number of roots of a system of equations. *Functional Analysis and its Applications (translated from Russian)*, 9(2):183–185, 1975.

[12] A.A. Broder. Generating random spanning trees. In *FOCS*, pages 442–447, 1989.

[13] B. Bruderlin. Constructing three-dimensional geometric object defined by constraints. In *ACM SIGGRAPH*. Chapel Hill, 1986.

[14] John Canny. Computing roadmaps of general semi-algebraic sets. *Computer Journal*, 36:504–514, 1993.

[15] John F. Canny and Ioannis Z. Emiris. A subdivision-based algorithm for the sparse resultant. *Journal of the ACM (JACM)*, pages 417–451, May 2000.

[16] D Caspar and A Klug. Physical principles in the construction of regular viruses. *Cold Spring Harbor Symp Quant Biol*, 27:1–24, 1962.

[17] P Ceres and A Zlotnick. Weak protein-protein interactions are sufficient to drive assembly of hepatitis b virus capsids. *Biochemistry*, 41:11525–11531, 2002.

[18] CNS. http://cns.csb.yale.edu/v1.0/. In *NMR structure software*.

[19] Robert Connelly. Tensegrity structures: why are they stable? *Rigidity Theory and Applications*, pages 47–54, 1998.

[20] D. Cox, J. Little, and D. O'Shea. *Using algebraic geometry*. Springer-Verlag, 1998.

[21] FHC Crick and JD Watson. Structure of small viruses. *Nature*, 177:473–475, 1956.

[22] G. Crippen and T. Havel. *Distance Geometry and Molecular Conformation*. John Wiley & Sons, 1988.

[23] A Edmondson. *A Fuller Explanation: The Synergetic Geometry of R. Buckminster Fuller*. Birkhauser Verlag, 1987.

[24] Ioannis Emiris and John Canny. A practical method for the sparse resultant. In *International Conference on Symbolic and Algebraic Computation, Proceedings of the 1993 international symposium on Symbolic and algebraic computation*, pages 183–192, 1993.

[25] A Bringer et al. Crystallography and nmr systems: a new software suite for macromolecular structure determination. *Acta Crystallographica*, D54:905–921, 1998.

[26] Tomas Feder and Milena Mihail. Balanced matroids. In *Annual ACM Symposium on Theory of Computing, Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*, pages 26–38, 1992.

[27] I. Fudos. *Geometric Constraint Solving*. PhD thesis, Purdue University, Dept of Computer Science, 1995.

[28] I. Fudos and C. M. Hoffmann. A graph-constructive approach to solving systems of geometric constraints. *ACM Transactions on Graphics*, 16:179–216, 1997.

[29] Anna Gambin. On approximating the number of bases of exchange preserving matroids. In *Mathematical Foundations of Computer Science*, pages 332–342, 1999.

[30] D Brenner S Lyshefski G Iafrate WA Goddard. *Handbook of nanoscience engineering and technology*. CRC press, 2002.

[31] Jack E. Graver, Brigitte Servatius, and Herman Servatius. *Combinatorial Rigidity*. Graduate Studies in Math., AMS, 1993.

[32] B. Hendrickson. Conditions for unique graph realizations. *SIAM J. Comput.*, 21:65–84, 1992.

[33] C Hoffman, M Sitharam, and B Yuan. Making constraint solvers more useable: the overconstraint problem. *to appear in CAD*, 2004.

[34] C. M. Hoffmann, A. Lomonosov, and M. Sitharam. Finding solvable subsets of constraint graphs. In Smolka G., editor, *Springer LNCS 1330*, pages 463–477, 1997.

[35] C. M. Hoffmann, A. Lomonosov, and M. Sitharam. Geometric constraint decomposition. In Bruderlin B. and Roller D., editors, *Geometric Constr Solving and Appl*, pages 170–195, 1998.

[36] Christoph M. Hoffmann, Andrew Lomonosov, and Meera Sitharam. Geometric constraint decomposition. In Bruderlin and Roller Ed.s, editors, *Geometric Constraint Solving*. Springer-Verlag, 1998.

[37] Christoph M. Hoffmann, Andrew Lomonosov, and Meera Sitharam. Planning geometric constraint decompositions via graph transformations. In *AGTIVE '99 (Graph Transformations with Industrial Relevance), Springer lecture notes, LNCS 1779, eds Nagl, Schurr, Munch*, pages 309–324, 1999.

[38] Christoph M. Hoffmann, Andrew Lomonosov, and Meera Sitharam. Decomposition of geometric constraints systems, part i: performance measures. *Journal of Symbolic Computation*, 31(4), 2001.

[39] Christoph M. Hoffmann, Andrew Lomonosov, and Meera Sitharam. Decomposition of geometric constraints systems, part ii: new algorithms. *Journal of Symbolic Computation*, 31(4), 2001.

[40] B. Huber and B. Sturmfels. A polyhedral method for solving sparse polynomial system. *Math. Comp.*, 64:1541–1555, 1995.

[41] J E Johnson and J A Speir. Quasi-equivalent viruses: a paradigm for protein assemblies. *J. Mol. Biol.*, 269:665–675, 1997.

[42] David R. Karger. Minimum cuts in near-linear time. *Journal of the ACM (JACM)*, 47(1):46–76, 2000.

[43] A. Khovanskii. Fewnomials. *Trans. of Math. Monographs*, 88, 1991.

[44] G. Kramer. *Solving Geometric Constraint Systems*. MIT Press, 1992.

[45] G. Laman. On graphs and rigidity of plane skeletal structures. *J. Engrg. Math.*, 4:331–340, 1970.

[46] R. Latham and A. Middleditch. Connectivity analysis: a tool for processing geometric constraints. *Computer Aided Design*, 28:917–928, 1996.

[47] Andrew Lomonosov and Meera Sitharam. Graph algorithms for geometric constraint solving. In *submitted*, 2004.

[48] C J Marzec and L A Day. Pattern formation in icosahedral virus capsids: the papova viruses and nudaurelia capensis β virus. *Biophys*, 65:2559–2577, 1993.

[49] A. Middleditch and C. Reade. A kernel for geometric features. In *ACM/SIGGRAPH Symposium on Solid Modeling Foundations and CAD/CAM Applications*. ACM press, 1997.

[50] Ravi Montenegro and Jung-Bae Son. Edge isoperimetry and rapid mixing on matroids and geometric markov chains. In *ACM Symposium on Theory of Computing*, pages 704–711, 2001.

[51] J. J. Oung, M. Sitharam, B. Moro, and A. Arbree. Frontier: fully enabling geometric constraints for feature based design and assembly. In *abstract in Proceedings of the ACM Solid Modeling conference*, 2001.

[52] J. Owen. www.d-cubed.co.uk/. In *D-cubed commercial geometric constraint solving software*.

[53] J. Owen. Algebraic solution for geometry from dimensional constraints. In *ACM Symp. Found. of Solid Modeling*, pages 397–407, Austin, Tex, 1991.

[54] J. Owen. Constraints on simple geometry in two and three dimensions. In *Third SIAM Conference on Geometric Design*. SIAM, November 1993. To appear in Int J of Computational Geometry and Applications.

[55] J.A. Pabon. Modeling method for sorting dependencies among geometric entities. In *US States Patent 5,251,290*, Oct 1993.

[56] Giovanni Pistone, Eva Riccomagno, and Henry P. Wynn. *Algebraic Statistics: Computational Commutative Algebra in Statistics*. CRC Press, December 2000.

[57] James Gary Propp and David Bruce Wilson. How to get a perfectly random sample from a generic markov chain and generate a random spanning tree of a directed graph. *Journal of Algorithms*, 27:170–217, 1998.

[58] D Rapaport, J Johnson, and J Skolnick. Supramolecular self-assembly: molecular dynamics modeling of polyhedral shell formation. *Comp Physics Comm*, 1998.

[59] V S Reddy, H A Giesing, R T Morton, A Kumar, C B Post, C L Brooks, and J E Johnson. Energetics of quasiequivalence: computational analysis of protein-protein interactions in icosahedral viruses. *Biophys*, 74:546–558, 1998.

[60] J. Maurice Rojas. A new approach to counting nash equilibria. In *Proceedings of the IEEE/IAFE Conference on Computational Intelligence for Financial Engineering*, pages 130–136, 1997.

[61] Gunter Rote, Francisco Santos, and Ileana Streinu. Expansive motions and the polytope of pointed pseudo-triangulations. In *Discrete and Computational Geometry - The Goodman-Pollack Festschrift*, pages 699–736, 2003.

[62] M Sitharam. Frontier, an opensource 3d geometric constraint solver: algorithms and architecture. *monograph, in preparation*, 2004.

[63] M Sitharam. A game-based decomposition of a class of real algebraic varieties. *preliminary manuscript, available upon request*, 2004.

[64] M Sitharam. Graph based geometric constraint solving: problems, progress and directions. In Dutta, Janardhan, and Smid, editors, *AMS-DIMACS volume on Computer Aided Design*, 2004.

[65] M Sitharam. Markov sampling of algebraic structures related to rigidity matroid bases. *preliminary manuscript, available upon request*, 2004.

[66] M Sitharam and M Agbandje-Mckenna. A geometry and tensegrity based virus assembly pathway model. *submitted, available upon request*, 2004.

[67] M Sitharam, A Arbree, Y Zhou, and N Kohareswaran. Solution management and navigation for 3d geometric constraint systems. *submitted, available upon request*, 2004.

[68] M Sitharam, J Peters, and Y Zhou. Solving minimal, wellconstrained, 3d geometric constraint systems: combinatorial optimization of algebraic complexity. *submitted to ADG 2004, available upon request*, 2004.

[69] M Sitharam, J Peters, and Y Zhou. Solving minimal, wellconstrained, 3d geometric constraint systems: combinatorial optimization of algebraic complexity. *submitted to ADG 2004, available upon request*, 2004.

[70] M Sitharam and Y Zhou. Mixing features and variational constraints in 3d. *submitted, available upon request*, 2004.

[71] M Sitharam and Y Zhou. A tractable, approximate, combinatorial 3d rigidity characterization. *submitted to ADG 2004, available upon request*, 2004.

[72] Meera Sitharam. Frontier, opensource gnu geometric constraint solver: Version 1 (2001) for general 2d systems; version 2 (2002) for 2d and some 3d systems; version 3 (2003) for general 2d and 3d systems. In *http://www.cise.ufl.edu/∼sitharam, http://www.gnu.org*, 2004.

[73] A. Sommese and J. Verschelde. Numerical homotopies to compute generic points on positive dimensional algebraic sets. *Journal of Complexity*, 16(3):572–602, 1999.

[74] Ileana Streinu. A combinatorial approach to planar non-colliding robot arm motion planning. In *Proc. 41st ACM Annual Symposium on Foundations of Computer Science (FOCS)*, pages 443–453, 2000.

[75] Ileana Streinu. Combinatorial roadmaps in configuration spaces of simple planar polygons. In *Proceedings of the DIMACS Workshop on on Algorithmic and Quantitative Aspects of Real Algebraic Geometry in Mathematics and Computer Science*, pages 181–206, 2003.

[76] Ileana Streinu and Walter Whiteley. Single-vertex origami and 3-dimensional expansive motion. submitted,Dec. 2003.

[77] T. Tay and W. Whiteley. Generating isostatic frameworks. *Topologie Structurale*, 11:21–69, 1985.

[78] A Zlotnick. To build a virus capsid: an equilibrium model of the self assembly of polyhedral protein complexes. *J. Mol. Biol.*, 241:59–67, 1994.

[79] A Zlotnick, R Aldrich, J M Johnson, P Ceres, and M J Young. Mechanisms of capsid assembly for an icosahedral plant virus. *Virology*, 277:450–456, 2000.

[80] A Zlotnick, JM Johnson, PW Wingfield, SJ Stahl, and D Endres. A theoretical model successfully identifies features of hepatitis b virus capsid assembly. *Biochemistry*, 38:14644–14652, 1999.

CISE DEPT., UNIVERSITY OF FLORIDA
E-mail address: sitharam@cise.ufl.edu

DEPARTMENT OF MOLECULAR BIOLOGY, UNIVERSITY OF FLORIDA
E-mail address: mckenna@ufl.edu