Generic rigidity in three-dimensional bond-bending networks

1998 J. Phys. A: Math. Gen. 31 6653

(http://iopscience.iop.org/0305-4470/31/31/012)

J. Phys. A: Math. Gen. **31** (1998) 6653–6668. Printed in the UK PII: S0305-4470(98)87463-5

Generic rigidity in three-dimensional bond-bending networks

D J Jacobs†

Department of Physics and Astronomy, Michigan State University, East Lansing, MI 48824, USA

Received 10 September 1997, in final form 27 January 1998

Abstract. Much progress can be made in studying the mechanical stability of frameworks when they are treated as generic, which lack any special symmetries. This is because testing for rigidity becomes topological in nature rather than geometrical. Generic rigidity, synonymous with *graph rigidity*, depends only on the connectivity of the network, making it a simpler problem to deal with in principle. A complete combinatorial constraint counting characterization of graph rigidity is given by Laman's theorem for two dimensions. Unfortunately there is no known corresponding theorem for three dimensions. Herein it is proposed that the theorem of Laman generalizes to three dimensions for *bar-joint* networks that have no implied-hinge joints. Particular attention is given to *bond-bending networks*, having a truss structure with constraints between nearest and next-nearest neighbours, that are suitable for modelling many covalent network glasses and macromolecules. It is shown that implied-hinge joints do not exist in bond-bending networks. Based on the proposition that an all subgraph constraint counting characterization of generic rigidity is recovered in three-dimensional bar-joint networks having no implied hinge joints, an efficient combinatorial algorithm is constructed for bond-bending networks. Complete agreement is found with exact calculations involving diagonalization of dynamical matrices, for systems up to 10^3 degrees of freedom.

1. Introduction

In the rich history of graph theory many important developments [1] have been made through interdisciplinary research. In this spirit, I show that combinatorial constraint counting is sufficient to completely characterize generic rigidity within a certain class of three-dimensional bar-joint frameworks. In particular, proposition 4.9 states that a generic bond-bending network in three dimensions with N sites and C constraints does not have a redundant bond if no subsection of the network containing n sites and c constraints violates $c \leqslant 3n - 6$. The importance of this work is two-fold. First, it advances the graph-theoretic understanding of graph rigidity on bar-joint frameworks in three dimensions. The approach used here is distinctly different from a body-bar multigraph description [2] of generic rigidity. Second, this work has been motivated by the desire to efficiently study the mechanical stability in real covalent network glasses [3–6] and macromolecules [7–10].

Many glass networks and macromolecules have strong covalent bonding forces that define bond lengths between all pairs of atoms and bond angles between all triples of atoms. The covalent bonding is modelled as hard rigid constraints between neighbouring pairs of atoms (central forces) and between next-nearest neighbour pairs of atoms (bending

† E-mail address: jacobs@pa.msu.edu

0305-4470/98/316653+16$19.50

forces). By neglecting all other weaker forces, one obtains a bond-bending network that is essentially a molecular truss structure that may be flexible with floppy motions through the dihedral-angle degrees of freedom. The bond-bending model gives a natural cut-off in identifying rigid molecular clusters having strong intra- and weak inter-atomic interactions. This approach has been successful in characterizing covalent network glasses as being either an amorphous solid or a polymeric glass [11, 12].

Some properties of network rigidity that are of interest include the number of independent degrees of freedom or floppy modes [13], the rigid cluster decomposition [14–16] and the identification of overconstrained (internally stressed) regions [16–18]. These properties are generally very difficult to calculate [19, 20] on large networks because rigidity depends on geometry in a nonlocal way [21]. Analysing network rigidity can also be used in conjunction with other numerical applications. For example, it is possible to study many systems more efficiently using molecular dynamics of weakly interacting coupled rigid bodies. This strategy is being incorporated in molecular dynamics simulation of protein folding [7, 8, 10]. However, it is generally calculationally more expensive to identify the rigid clusters and independent degrees of freedom than the calculational expense that would be saved using the rigidity information. For these advanced molecular dynamic simulation techniques to be implemented successfully on more challenging networks beyond tree-like topologies, an exact algorithm must be developed to identify network rigidity that is orders of magnitude faster than current available numerical algorithms [9, 20].

Recently [14] it has been appreciated that network glasses and macromolecules are well suited to be modelled as generic networks. Within generic networks, algebraic dependences that lead to geometrical singularities [21–23] are eliminated. This simplifies the problem to one of *graph rigidity* [23] involving only the network connectivity. Nevertheless, the properties of *generic rigidity*, taken here to be synonymous with graph rigidity, are still difficult to calculate because of the inherent long-range nature of rigidity.

The theorem of Laman [24] (stated in section 2.1) gives a complete graph-theoretic characterization of generic rigidity in two dimensions. This proved to be of tremendous calculational value [16, 18] in the study of generic rigidity percolation in two dimensions [14, 15, 17]. However, one is faced with the unfortunate situation that there is currently no known graph-theoretic characterization of generic rigidity [23, 25] for a general graph in three dimensions. Therefore, the development of an *exact* and efficient algorithm for testing generic rigidity in bar-joint networks is at the same juncture as finding a complete graph-theoretic characterization.

Progress has been made on both these fronts for bond-bending networks by taking advantage of special properties that are discussed in section 4 following a general background on generic rigidity in section 2, and after it is argued in section 3 that Laman's theorem can be generalized to bar-joint frameworks having no implied-hinge joints. These results can be used to construct a combinatorial constraint counting algorithm for bond-bending networks similar to the two-dimensional *pebble game* [16]. In section 5 a short discussion is given on how these results compare with other approaches and about the type of extensions that can possibly be made from the graph-theoretic developments given here. A conclusion is then given in section 6.

2. Generic rigidity

Only the basic concepts of graph rigidity are introduced here, while leaving the details to be found elsewhere [2, 22, 23, 25–27]. The structure of a network will be described as a graph $G(V, E)$ consisting of a set of vertices, V, and a set of edges E. The vertices correspond

to atoms and the edges correspond to central-force bonds and possibly bond-bending force constraints. Each edge, e, connects a pair of vertices (v_i, v_j). An edge is said to be incident to vertex v_i if it connects vertex v_i to some other vertex v_j. Within graph $G(V, E)$, let $G'(V', E')$ be a subgraph defined by a set of vertices, V', consisting of a set of edges, E', that are incident to only those vertices.

A framework, $p(G)$, is a realization of the graph G through the mapping $p : V \to R^3$ which assigns each vertex to a point in three-dimensional space [16, 18, 22]. The set of generic realizations is dense in the space of all realizations, and almost all realizations are generic. As an example of a generic realization, it is sufficient to consider the coordinates of all vertices to be algebraically independent over the rationals. However, all that is necessary is to avoid some specific algebraic dependences.

The basic question of interest is to find the sections of a framework that can be flexed while keeping all edge lengths unchanged. A finite flexing of a framework $p(G)$ is a continuous family of realizations of G, such that the position, r_i, of each vertex v_i, is a differentiable function of time, t. There is a set of constraints imposed on each realization of the form $|r_i(t) - r_j(t)|^2 =$ constant for every $e \in E$. Differentiating the edge length constraints with respect to time, it follows that

$$(u_i - u_j) \cdot (r_i - r_j) = 0 \qquad \text{for every } e \in E \tag{1}$$

where u_i is the instantaneous velocity of vertex v_i. More generally, an assignment of velocities that satisfies equation (1) for a single realization defines an *infinitesimal motion* for the framework. The existence of a finite flexing implies an infinitesimal motion, but the converse is not always true. However, each infinitesimal motion does correspond to a finite flexing [22, 27] in a generic realization.

A matrix can be constructed from the entire set of equations of the form of equation (1). This matrix, consisting of $|E|$ rows and $3|V|$ columns (in three dimensions), is called a rigidity matrix [22] where each row corresponds to an edge and each column corresponds to a coordinate of a vertex. For each edge e, its corresponding row has only six non-zero elements corresponding to the difference in the coordinate values of its two associated incident vertices. The construction of this matrix is useful, because linear algebra can be used to determine the number of rows that are linearly independent by obtaining its rank. For a generic realization, the rank is maximal because there is no special algebraic dependencies [18, 22, 28].

The rigidity matrix is introduced here for the purpose of defining a redundant edge. Alternatively, a dynamical matrix could have been used for the same purpose [12], and it turns out that the two approaches are closely related [28].

Definition 2.1. A *redundant edge* is an edge for which if removed from the framework, the rank of the rigidity matrix does not change.

Conversely, an edge is independent if upon removal from the framework, the rank of the rigidity matrix decreases. Because we are considering only generic realizations the network rigidity becomes a property of the underlying graph by the following theorem [26].

Theorem 2.2. If a graph has a single infinitesimally rigid realization, then all its generic realizations are rigid.

As a result, one can speak about characterizing generic rigidity of a graph, which formally corresponds to characterizing rigidity of the typical behaviour of a framework, without reference to a specific realization. Two observations are made below which follow from the above discussion.

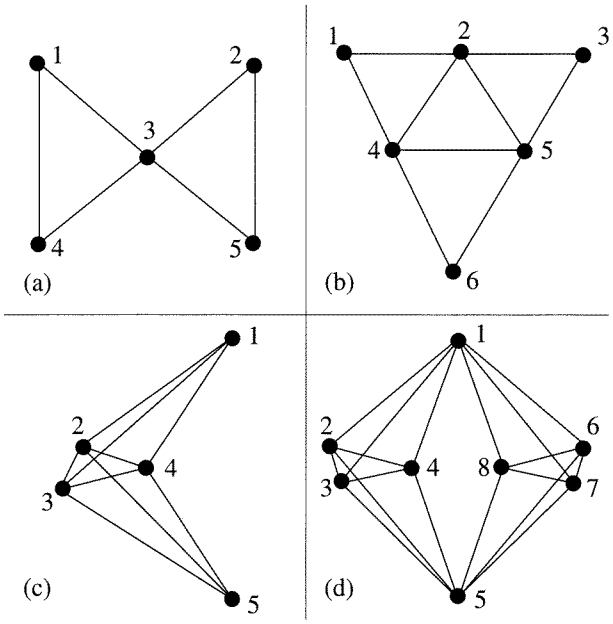

Figure 1. Four example graphs that are embedded in three dimensions. (*a*) Vertex 3 is a *pivot joint*. (*b*) The vertex pairs (2, 4), (2, 5) and (4, 5) are *hinge joints*. (*c*) The vertex pair (1, 5) is an *implied edge*. (*d*) The vertex pair (1, 5) is an *implied-hinge joint*.

Observation 2.3. A set of two or more vertices in a generic framework are mutually rigid when the distance between all pairs of these vertices is constant for all possible continuous motions of the framework.

Observation 2.4. Let V_1 and V_2 define two distinct sets of mutually rigid vertices. If $V_1 \cap V_2$ contains three or more vertices, then the vertices in set $V_3 = V_1 \cup V_2$ are mutually rigid.

Observation 2.3 is treated formally using the properties of the rigidity matrix [22, 23], and observation 2.4 is part of the generic gluing lemma [25]. Some basic definitions are given below which will be needed in later sections.

Definition 2.5. A *rigid cluster* is a subgraph $G'(V', E')$ having vertices, V', that are mutually rigid within graph $G(V, E)$ such that no other vertex, $v \in \overline{V'}$, is mutually rigid with respect to all V' vertices for $|V'| \leqslant 3$ or any three vertices for $|V'| > 3$.

The graph G will generally consist of rigid clusters that are connected and disconnected. In three dimensions the connected rigid clusters can share only three types of floppy elements between themselves, which need to be classified. The most floppy of these elements is the *pivot joint* acting between two rigid clusters.

Definition 2.6. A *pivot joint* is formed at vertex v when it is the only vertex common to any two rigid clusters.

For example, the centre vertex in the 'bow-tie' graph of figure 1(*a*) is a pivot joint between two rigid triangles. Three degrees of freedom are generally required to specify the relative orientation between two rigid clusters, each of three or more vertices, that are connected by a pivot joint.

Definition 2.7. A *hinge joint* is formed at edge e when it is the only edge common to any two rigid clusters.

Specifying the relative orientation of two rigid clusters connected by a hinge joint requires one degree of freedom, corresponding to a dihedral angle. In figure 1(b) the edges between the pairs of vertices $(2, 4)$, $(2, 5)$ and $(4, 5)$ are examples of hinge joints. Since two distinct rigid clusters cannot share more than two vertices by observation 2.4, there cannot be a less floppy element between two rigid clusters than a hinge joint. However, there is a variant to the hinge joint, which derives from an implied edge.

Definition 2.8. An *implied edge* is formed between any pair of vertices (v_1', v_2') within a rigid cluster, $G'(V', E')$, that does not have an edge between the vertices v_1' and v_2'.

By observation 2.3 there is a fixed distance between each pair of vertices within a rigid cluster. The physically imposed distance constraints are represented by the edges in set E. All other pairs of vertices within a rigid cluster form implied edges, such as between vertices $(1, 5)$ shown in figure 1(c).

Definition 2.9. An *implied-hinge joint* is formed when any two rigid clusters share only two vertices v_1 and v_2 without a common edge connecting (v_1, v_2).

An implied-hinge joint is formed between the pair of vertices $(1, 5)$ within the 'double banana' graph shown in figure 1(d). Effectively implied-hinge joints act like regular hinges, but as will be seen they make the graph-theoretic characterization of generic rigidity in three dimensions difficult.

2.1. Constraint counting

Clearly subgraphs with many edges are more likely to be rigid than those with only a few, since the edges are constraining the possible movements of the vertices. It is expected that as long as the edges are uniformly randomly distributed throughout the network, the number of independent degrees of freedom will decrease as the number of edges increases. The number of independent degrees of freedom or floppy modes, F, in a network can be estimated by constraint counting [12, 13]. The approximation is given by $F \approx 3N - C$ where N is the total number of vertices and C is the total number of edges (constraints). However, because not all the constraints are independent this estimate serves as a lower bound. It is well known [22, 23, 25] that in d-dimensions there need not be more than $dn - d(d+1)/2$ constraints to make a set of $n \geqslant d$ vertices mutually rigid or more than $n(n-1)/2$ constraints for a set of $n < d$ vertices. Otherwise, the number of excess constraints among n sites are redundant.

By counting the number of constraints between all possible combinations of sites in the network, one can hope to identify every redundant constraint. Each rigid region can then be identified whenever there are just enough *independent constraints* distributed among a set of sites to make them mutually rigid. In two dimensions this logic is correct as stated by Laman's theorem [24, 29].

Theorem 2.10. The edges of a graph, $G(V, E)$, are independent in two dimensions if and only if no subgraph, $G'(V', E')$ has more than $2|V'| - 3$ edges.

The essence of theorem 2.10 is that in two dimensions, generic rigidity is completely characterized by applying constraint counting to all possible combinations of sites. In other words, constraint counting must be done on all levels of the network. As long as every subgraph is considered (the number of subgraphs is exponential in $|V|$) it is possible to

select a basis set for the independent constraints and to eliminate redundant constraints. Efficient algorithms [16, 18] are available for performing this counting by using Laman's theorem recursively, and thus it is possible to test for generic rigidity in the plane.

Intuitive as constraint counting may seem, this approach generally fails in higher dimensions! The double banana graph in figure 1(*d*) is the infamous counterexample [23, 25] for why a generalized version of Laman's theorem fails in three dimensions. There are no subgraphs having more than $3|V'|-6$ edges connecting $|V'|$ vertices. Accordingly, the edges are well distributed among the vertices and one would be lead to the wrong conclusion that this graph is rigid. Nevertheless, applying constraint counting to all combinations of sites (i.e. all subgraphs) within a generic network serves as a lower bound for the number of floppy modes within any dimension. The combinatorial algorithm of Franzblau [28, 30] is constructed along these lines for giving a good lower bound on the number of floppy modes for generic bond-bending networks in three dimensions.

3. Generalization of Laman's theorem

It is well known [22, 23, 25] that the '*if*' part of Laman's theorem generalizes to all dimensions for a general graph $G(E, V)$ simply by replacing the maximum number of edges that can be found in a subgraph with $d|V'| - d(d+1)/2$ for $|V'| \geqslant d$ and $|V'|(|V'|-1)/2$ otherwise. However, the '*only if*' part generally fails in three and higher dimensions as demonstrated by the double banana graph [25] in figure 1(*d*). The double banana graph is floppy with one internal motion about the implied-hinge joint between vertices $(1, 5)$. This means that there is a redundant bond somewhere. Note that the left banana, separately shown in figure 1(*c*), forms a rigid cluster with an implied edge between vertices $(1, 5)$. When the right banana is joined to vertices $(1, 5)$, the implied edge (now an implied-hinge joint) causes one edge in the right banana to become redundant. Evidently the implied-hinge joint is acting like an ordinary edge, but is *missed* during a straight counting of constraints.

The generalized version of theorem 2.10 in three dimensions *does work* for the double banana of figure 1(*d*) if an additional edge is placed either between vertices $(1, 5)$ or $(4, 8)$ for example. In the first case, the rigidity properties is unchanged but now the distribution of edges are correctly counted, yielding two redundant edges. This is because an edge between vertices $(1, 5)$ corresponds to a *physical constraint* that can be tangibly counted. In the latter case, there is only one rigid cluster because the added edge locks the two bananas together. One redundant edge is correctly counted.

Although connectivity between a set of vertices that are mutually rigid is a necessary condition in graph G, within individual rigid clusters connectivity is *not required*. As the 'triple banana' graph of figure 2 illustrates, there are three rigid clusters formed by face sharing tetrahedra (the bananas) and a rigid triangle with vertices $\{1, 2, 3\}$ that are not connected as a subgraph. The vertices $\{1, 2, 3\}$ are mutually rigid because the implied-hinge joints between vertex pairs $(1, 2)$, $(1, 3)$ and $(2, 3)$ act as independent constraints. Therefore it is possible to have a mutually rigid set of vertices that are not contiguously connected to one another! In general, when constraint counting is applied on all levels of a network, the noncontiguous rigid clusters will be missed. Note that this additional nonlocal feature of rigidity does not occur in two dimensions.

As the examples above indicate, implied-hinge joints give rise to various kinds of *banana structures* that can cause errors in constraint counting or lead to rigid clusters that are not contiguously connected. It will be argued that only implied-hinge joints are responsible for the difficulties encountered in characterizing three-dimensional generic rigidity with constraint counting. This may be surprising because after all: what type of simplification

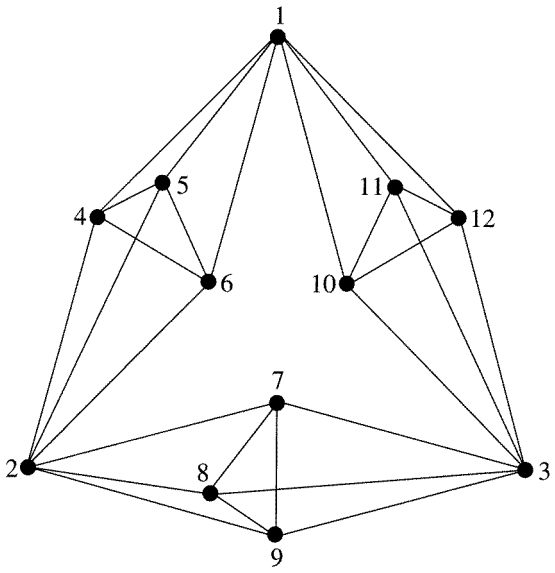

Figure 2. A triple banana graph in three dimensions consisting of three implied-hinge joints between vertices $(1, 2)$, $(1, 3)$ and $(2, 3)$. This graph decomposes into four rigid clusters.

can result if edges are placed between pairs of vertices that form implied-hinge joints? To answer this, consider the following thought experiment.

(i) Decompose (the process of identification) the network, G, into its unique set of rigid clusters. Note that all vertices and all edges will belong to one or more rigid cluster.

(ii) Place an edge between all pairs of vertices that form an implied-hinge joint—thereby creating a regular hinge joint. Note that these edges are redundant, as their presence is not required.

(iii) For each hinge joint between a pair of vertices shared by n rigid clusters, place $n - 1$ additional edges between the same pair of vertices. Note that all these additional edges are also redundant.

(iv) Separate the rigid clusters from one another, with the assignment of an edge (representing a hinge joint) to each rigid cluster that shares a hinge joint from the set of duplicated edges created in step (iii).

(v) Consider each subgraph separately (formerly a rigid cluster in the network), and choose one that is no longer rigid as an isolated entity. This subgraph, G_ψ, will have $\psi > 0$ internal degrees of freedom that must have been constrained by the remaining part of the network.

(vi) Since all implied-hinge joints acting on subgraph G_ψ have been accounted for and preserved, the required ψ implied constraints that make G_ψ rigid (as a subgraph in G) must be derived from a mechanism that couples G_ψ to the remaining part of G.

(vii) For the mechanism to couple G_ψ to the rest of the network, it must consist of some vertices, V_{out} belonging to rigid clusters in G other than those in G_ψ, as well as vertices, V_{in}, that belong to G_ψ.

(viii) With the ψ implied constraints imposed by the mechanism coupling G_ψ to the rest of the network, G_ψ becomes a rigid subgraph in G. However, this means that all relative motion between the vertices in the mechanism is locked, which is a consequence of Newton's third law.

(ix) As the mechanism locks, other vertices outside of G_ψ become rigid with respect to G_ψ itself, which is inconsistent with the rigid cluster decomposition. Therefore, in applying step (v), all the isolated subgraphs are indeed rigid [31].

As ascertained from the thought experiment an observation can be made as follows.

Observation 3.1. Implied constraints acting on a rigid cluster within a network can only be imposed by connecting rigid clusters through implied-hinge joints.

Note that implied-hinge joints act as independent constraints upon all but one of the rigid clusters that share it, making constraint counting difficult. Without these implied-hinge joints the rigid clusters have a much simpler nature.

Theorem 3.2. Each rigid cluster, $G'(V', E')$, is a connected subgraph within $G(V, E)$ if no implied-hinge joint exists in G.

Proof. Divide the set of mutually rigid vertices in V' into two subsets such that $V' = V'_a \cup V'_b$ and $V'_a \cap V'_b = \{\ \}$. A rigid cluster is disconnected when only implied edges join vertices from set V'_a to the vertices in set V'_b. Without support of graph G the rigid cluster cannot exist. Therefore, other vertices and edges not within the rigid cluster are responsible for making some pairs of vertices (v'_a, v'_b) mutually rigid. From observation 3.1, a disconnected rigid cluster will share at least one implied edge between vertices (v'_a, v'_b) with another rigid cluster, forming an implied-hinge joint. □

The converse of theorem 3.2 is not true. Having only connected rigid clusters in G does not prevent the existence of implied-hinge joints. This can be seen in the double banana graph of figure 1(*d*).

Proposition 3.3. The edges of a graph, $G(V, E)$, having no implied-hinge joints, are independent in three dimensions if and only if no subgraph, $G'(V', E')$, has more than $3|V'| - 6$ edges for $|V'| > 2$ or more than $|V'|(|V'| - 1)/2$ edges for $|V'| \leqslant 2$.

Proof. All that is needed to show is that if a redundant edge, e, between a pair of vertices, (v'_1, v'_2) exist, then the stated condition on the number of edges is necessarily violated. If edge e is redundant, then the pair of vertices (v'_1, v'_2) are mutually rigid without edge e. Therefore this pair of vertices belong to a rigid cluster. Select this rigid cluster as a test subgraph within graph G. Since it is rigid, it must have $3|V'|-6$ independent constraints for $|V'| > 2$ or $|V'|(|V'| - 1)/2$ independent constraints otherwise. Since it is given that there are no implied-hinge joints, it follows that every independent constraint required to keep the vertices in V' mutually rigid must be an edge, not an implied edge. Thus, by adding a redundant edge e, the number of edges must exceed the minimum required number of edges. □

According to proposition 3.3, if a network has no implied-hinge joint, then constraint counting on all levels will completely characterize generic rigidity. This result may not seem useful because one does not know whether an implied-hinge joint does not exist in a given graph until after a rigid cluster decomposition is made. Nevertheless, proposition 3.3 can be directly used to construct a simple recursive algorithm for testing generic rigidity in any network that has no implied-hinge joint and where all its hinge joints can be selected as part of an independent basis set of constraints. More importantly, these conditions are not overly restrictive for many physical problems of interest. In the next section it will be shown that bond-bending networks (truss frameworks) fit all requirements, making proposition 3.3

of practical importance in applications for determining rigidity in covalent glass networks and macromolecules.

Note that the above logic can be used to prove Laman's theorem in one and two dimensions, where the elementary floppy units do not include implied-hinge joints. More specificly, rigid clusters in one and two dimensions cannot be externally acted upon by implied constraints brought in by connected rigid clusters. However, in this broader context the above thought experiment is known to fail in dimensions greater than three, where it is possible to construct a mechanism that creates an implied edge between a single pair of vertices [25, 31] without having a subdimensional interface between two rigid clusters.

4. Generic rigidity in bond-bending networks

Generic rigidity in three-dimensional bond-bending networks corresponds to graph rigidity within the restricted class of squared graphs. A graph to the nth power is defined [1] as follows.

Definition 4.1. For any given graph G, the nth power of G, denoted as G^n, has $V(G^n) = V(G)$ with v_1, v_2 adjacent in G^n whenever the distance $d(v_1, v_2) \leqslant n$ in G, where the distance between two vertices is the shortest path between them.

Let graph $G = G(V, E_1)$, having $|V|$ vertices and $|E_1|$ edges, define the underlying description of the bond-bending network. The edges of set E_1 correspond to central-force constraints. Let:

$$G_2(V, E) = G_2(V, E_1^*, E_2) = G^2(V, E_1) \tag{2}$$

where E_2 is an *induced* set of edges joining all next-nearest neighbour vertices present in the underlying graph G, $E = E_1 \cup E_2$ and E_1^* contains every edge in set E_1 that does not overlap with an induced edge from set E_2 such that $E_1^* \cap E_2 = \{\ \}$.

As seen in figure 3(a) each edge in E_2 corresponds to a bending-force constraint that is induced by a pair of edges in E_1 that are incident to the same vertex. There can be overlap between the edges in sets E_1 and E_2. This situation occurs in the example graph of figure 3(b) for edges joining the vertex pairs (3, 4), (3, 5) and (4, 5). When more than one edge is placed between a given pair of vertices the degenerate set together acts as a single edge, which may be independent or redundant. The edges making up set E_1^* for this graph are between the vertices (1, 2), (1, 6) (2, 3) and (5, 6).

When the graph G consist of disconnected subgraphs, then graph G_2 will have the same set of disconnected subgraphs since the edges of E_2 are induced between pairs of vertices that are already connected. Graph rigidity is tested separately for each connected subgraph in G, and thus G_2. The graph rigidity of connected subgraphs consisting of isolated vertices or single edges is trivial. Therefore let us work with an underlying graph $G(V, E_1)$ that is connected and has $|V| \geqslant 3$.

The first important characteristic of graph G_2 is that it is made up of edge sharing triangles. Evidently, the smallest rigid cluster possible is a complete graph of three vertices forming a triangle. Note, however, that all graphs consisting of edge sharing triangles do not generally define squared graphs. For example, the graph in figure 1(b) is made up of four edge sharing triangles, but it is not a squared graph. The distinction between edges as either an element of the sets E_1 or E_2 imposes additional constraints on the connectivity of a G_2 graph. Note that vertices in G_2 are connected via edges from E_1, and all induced triangles have two edges from set E_1 and one edge from set E_2. As the following theorems

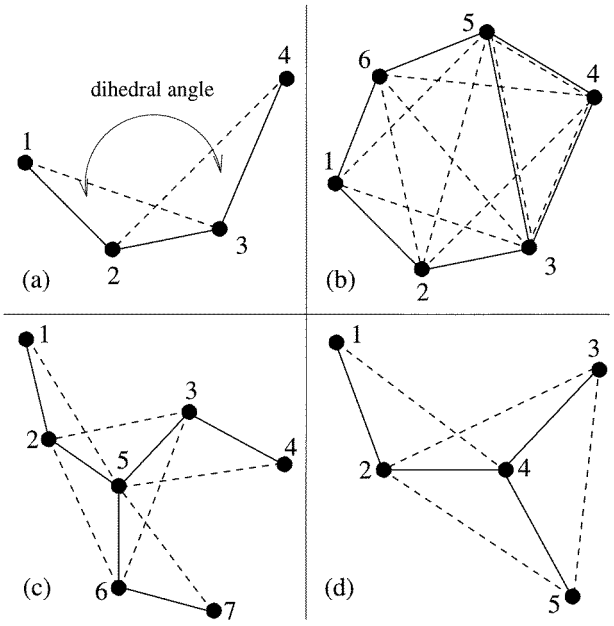

Figure 3. Four examples of squared graphs embedded in three dimensions. Full (broken) lines denote the underlying (induced) edges. (*a*) Two edge sharing triangles forms a hinge joint having a free dihedral angle. (*b*) A rigid cluster. (*c*) Four rigid clusters. (*d*) Two rigid clusters.

show, these additional constraints on the connectivity of a squared graph are sufficient to eliminate the existence of pivot joints and implied-hinge joints.

Theorem 4.2. No pivot joints exist in a squared graph $G_2 = G^2$.

Proof. For a pivot joint to exist, there must be at least two rigid clusters that share only one vertex, v. However, a G_2 graph consist of edge sharing triangles so that at least two vertices are shared by any two rigid clusters having vertex v in common. □

It can be seen in figure 3(*c*) why a vertex cannot be a pivot joint in a G_2 graph. The three triangles having vertices $\{1, 2, 5\}$, $\{3, 4, 5\}$ and $\{5, 6, 7\}$ all share vertex 5. However, the induced edges between vertices $(2, 3)$, $(2, 6)$ and $(3, 6)$ eliminates vertex 5 as a pivot joint and instead forms a rigid cluster with vertices $\{2, 3, 5, 6\}$. The basic floppy element that survives is a hinge joint occurring between the vertices $(2, 5)$, $(3, 5)$ and $(5, 6)$. Although hinge joints can exist in a G_2 graph, it will now be shown that implied-hinge joints cannot.

Theorem 4.3. No implied-hinge joints exist in a squared graph $G_2 = G^2$.

Proof. Assume an implied-hinge joint incident to vertices (v_1, v_2) exist in G_2. Since G_2 consist of edge sharing triangles, each pair of associated rigid clusters are floppy with respect to one another via two hinge joints; incident to vertices v_1 and v_2. Therefore, each pair of rigid clusters have at least three distinct vertices in common. By observation 2.4 each rigid cluster pair are mutually rigid, leading to a contradiction. □

The only floppy element that survives in a bond-bending network is the hinge joint between rigid clusters, and they also have certain restricted characteristics as follows.

Lemma 4.4. In a squared graph, $G_2(V, E_1^*, E_2) = G^2(V, E_1)$, the vertices within a rigid cluster are connected by the underlying edges in set E_1.

Proof. By theorems 4.3 and 3.2 all the vertices in a rigid cluster are connected. Induced edges in E_2 only form between pairs of vertices already connected by edges in E_1. □

Lemma 4.5. In a squared graph, $G_2(V, E_1^*, E_2) = G^2(V, E_1)$, the induced edges E_2 cannot form a hinge joint.

Proof. Choose any induced edge, e, of set E_2 and either one of its incident vertices, v. A hinge joint forms when two or more rigid clusters share edge e. By lemma 4.4 the vertex v of edge e has at least one incident edge from set E_1 for each rigid cluster. Induced edges triangulate all underlying edges incident to vertex v, and thereby prohibit edge e to be a hinge joint. □

Corollary 4.6. In a squared graph, $G_2(V, E_1^*, E_2) = G^2(V, E_1)$, the induced edges E_2 belong to only one rigid cluster.

Proof. For an edge to belong to more than one rigid cluster it must be a hinge joint. By lemma 4.5 any edge in set E_2 cannot be a hinge joint. □

As a result of lemma 4.5 only edges in set E_1^* may act as a hinge joint. These hinge joints also have certain restrictions.

Lemma 4.7. In a squared graph, $G_2(V, E_1^*, E_2) = G^2(V, E_1)$, the edges E_1^* can at most belong to two rigid clusters.

Proof. Assume edge, e, of set E_1^* between vertices (v_1, v_2) is a hinge joint. All edges incident to vertex v_1 from set E_1 are members of a single rigid cluster since their induced edges form a complete subgraph. Likewise the underlying edges from set E_1 incident to vertex v_2 belong to a single rigid cluster. Therefore the edge e, having vertices v_1 and v_2 in common, can at most belong to two rigid clusters. □

In figure 3(d) it is seen that the underlying edge between the pair of vertices $(2, 4)$ forms a hinge joint. All the underlying edges incident to vertex 2 and 4 form the rigid cluster with vertices $\{1, 2, 4\}$ and $\{2, 3, 4, 5\}$ respectively. Therefore the hinge joint belongs to these two rigid clusters. Adding more underlying edges incident to vertex 2 or 4 will not increase the number of rigid clusters that share the hinge joint. A result of having each hinge joint being at most two-fold degenerate places a restriction on the number of distinct rigid clusters that any given vertex may belong.

Corollary 4.8. Within squared graph $G_2 = G^2$, any vertex that is of degree n in the underlying graph G can belong to at most $n + 1$ rigid clusters.

Proof. Given a vertex, v, with n incident underlying edges from set E_1. Vertex v belongs to a rigid cluster containing all of its incident edges from E_1 plus by lemma 4.7, one additional rigid cluster for each underlying hinge joint incident to v. At most there can be n hinge joints. □

A vertex v is said to be n-coordinated in graph $G_2 = G^2$ if it is of degree n in the underlying graph G. As shown in figure 3(c), vertex 5 is 3-coordinated where all of its incident underlying edges between vertices $(2, 5)$, $(3, 5)$ and $(5, 6)$ are hinge joints. In this case, the 3-coordinated vertex belongs to four rigid clusters consisting of the vertices $\{2, 3, 5, 6\}$, $\{1, 2, 5\}$, $\{3, 4, 5\}$ and $\{5, 6, 7\}$.

Many special properties of generic rigidity within a bond-bending network has been derived thus far by using the local connectivity of a squared graph. In addition, a complete combinatorial constraint counting characterization can be stated as follows.

Proposition 4.9. The edges of a squared graph, $G_2(V, E) = G^2(V, E_1)$, are independent in three dimensions if and only if no subgraph, $G_2'(V', E')$, has more than $3|V'| - 6$ edges for $|V'| > 2$ or more than $|V'|(|V'| - 1)/2$ edges for $|V'| \leqslant 2$.

Proof. According to theorem 4.3, a squared graph has no implied-hinge joints. Therefore, this result follows immediately from proposition 3.3. □

The main result of proposition 4.9 is that every constraint within any subgraph of G_2 is accountable by the presence of an edge, and the only way an edge can be redundant is if a subgraph has more edges than the required number of constraints to make those vertices mutually rigid. Consequently, the number of degrees of freedom or floppy modes, F, can be related to the rigid cluster decomposition in the form of a sum rule. For completeness, this sum rule is given for a general underlying graph G which may have disconnected subgraphs with 1, 2 or more vertices.

Theorem 4.10. The number of independent degrees of freedom, F, in a generic squared graph $G_2 = G^2$ is given by:

$$F = 3N - \sum_{n=3}^{N} [C_n(3n - 6)] - C_2 + H$$

where $N = |V|$ of graph G_2, C_n is the number of rigid clusters having n vertices and H is the number of hinge joints.

Proof. Since there are no implied-hinge joints, every rigid cluster in G_2 is connected, and taken separately each has $3n - 6$ independent edges for $n \geqslant 3$ vertices and one independent edge for $n = 2$. For every pair of rigid clusters connected by a hinge joint one independent edge is doubled counted. This is because at a given hinge joint between the pair of vertices (v_1, v_2), the vertices v_1 and v_2 are independently mutually rigid within each cluster. Therefore, one independent edge (taken from either rigid cluster) becomes redundant as the two rigid clusters are joined at v_1 and v_2. □

4.1. The three-dimensional pebble game

By using proposition 4.9 recursively, a three-dimensional *pebble game* algorithm has been constructed in a similar fashion as the two-dimensional *pebble game* [16]. One difference is that three pebbles are assigned to each vertex rather than two. There is also an important difference in regard to building the network up one edge at a time to obtain a maximal independent edge set. A distinction is made between an edge, $e \in E_1$, representing a central-force bond to that of an induced edge, $e \in E_2$, representing a bond-bending constraint. The

edges of set E_1 are free to be placed in any order. However, once a central-force edge is placed, all its associated locally induced edges, from set E_2, must be placed before the next edge in set E_1 is placed. The order in placing the set of locally induced edges does not matter. This simple restriction *nearly* conserves the connectivity properties to be that of a squared graph throughout the building process. The preference of central-force edges over bond-bending edges turns out to be sufficient for producing an exact constraint counting algorithm.

Like the *two-dimensional pebble game* algorithm [14–16], the *three-dimensional pebble game* exactly counts the number of floppy modes and identifies all rigid clusters in a bond-bending network. The *three-dimensional pebble game* has been implemented and extensively tested on hundreds of thousands of networks, with a wide range of variety. Networks containing up to 10^6 vertices have been considered. The maximum coordination number for a given vertex has ranged from 3 to 10, using underlying topologies of the diamond, simple cubic, body centred cubic, face centred cubic, honeycomb, square and triangular lattices with random dilution in either the edges or vertices. Another type of network topology was generated by randomly placing central-force edges between any pair of vertices up to a given maximum coordination number, and then the edges or vertices are randomly diluted.

Since the *three-dimensional pebble game* builds a given network up by adding edges recursively, each network was built up ten times by placing one edge at a time in *different random order*. The exact same results for the number of floppy modes and rigid cluster decomposition were always obtained. Self-consistency checks against theorems 4.7 and 4.10 were successful on all the rigid cluster decomposition results. In addition, numerical methods involving determining a rank of a matrix (using generic coordinates) have been used to calculate the number of floppy modes. The rigid cluster decomposition was also obtained using an exact numerical method as suggested by Franzblau [32] which relies on finding the rank of a matrix to identify individual redundant bonds. Comparison between the numerical methods and the *three-dimensional pebble game* was made on network sizes containing up to 400 vertices. The number of floppy modes and the entire rigid cluster decomposition from both methods were always found to be identical.

5. Discussion

Propositions 4.9 and 3.3 dealing with bar-joint networks, have an analogous similarity with the theorems by Tay [2] that deal with body-bar networks using multigraphs, where each node is regarded as a rigid body. For generically placed bars between rigid bodies, it is possible to completely characterize rigidity [2, 25] by combinatorial constraint counting on all levels of subgraphs. In particular, if two neighbouring rigid bodies share five bars, then this situation is equivalent to having a generic screw axis between the two rigid bodies. Whiteley [25] proposed making a 'molecular model' such that all atoms are considered to be rigid bodies with *six* degrees of freedom with a screw axis between neighbouring pairs of atoms. This molecular model has been *conjectured*† to be equivalent to the three-dimensional bond-bending network.

The common element between the three-dimensional bar-joint frameworks without implied-hinge joints and a body-bar network is that generic rigidity is completely determined by a combinatorial constraint counting characterization. Therefore it should not be surprising

† There is a technical problem in that algebraic dependences are introduced when shrinking a solid sphere to a point.

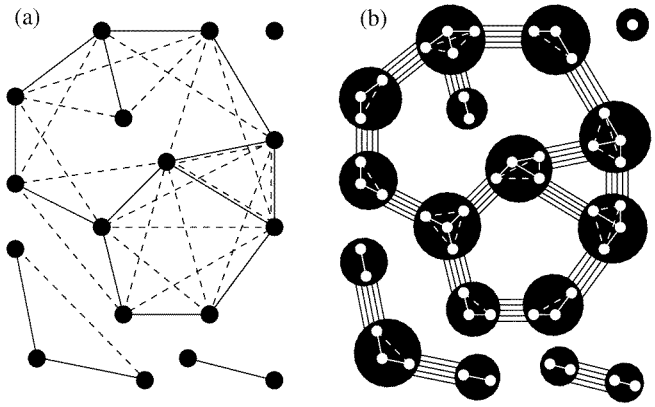

Figure 4. Example of the exact mapping between the bar-joint and body-bar representations of a three-dimensional bond-bending network. (*a*) A bar-joint framework including an isolated edge and vertex. (*b*) The corresponding body-bar network using multigraphs. Each full circle is a node in the multigraph corresponding to a local rigid cluster (shown in white) from the bar-joint framework. A node containing one, two or (more than two) vertices represents a rigid object having three, five or six free degrees of freedom respectively. All connected rigid objects share five generic bars; representing a screw axis.

to find an exact mapping between the two formulations; depending on how the elementary degrees of freedom and constraints are grouped for counting. However, there is a real *physical* difference between assigning an atom six or three degrees of freedom that goes beyond a regrouping of the various components in a network. Moukarzel [18] has shown that there is an exact mapping between the bar-joint and body-bar formulation in two dimensions provided sub-dimensional rigid bodies are included. Likewise, an exact mapping exist between a bar-joint and body-bar representation of a bond-bending network in three dimensions.

The mapping is derived here by construction. Each vertex of a squared graph belongs to a *local* rigid cluster consisting of itself and its nearest neighbours. An isolated vertex is considered a rigid sub-dimensional object of three degrees of freedom. A 1-coordinated vertex forms a sub-dimensional rigid dimer having five degrees of freedom. All other n-coordinated vertices for $n \geqslant 2$, belong to a rigid body having six degrees of freedom. From theorems 4.5–4.7 it follows that the only floppy motion between neighbouring rigid clusters is through a dihedral angle with a central-force bond acting as a hinge. The relative motion about a hinge-joint between two neighbouring *local* rigid clusters defines a generic screw axis. Therefore, the mapping consist of partitioning a squared graph into locally defined rigid bodies associated with each vertex while preserving the underlying graph, $G(V, E_1)$. Each vertex, $v \in V$, is transformed into a node representing a rigid body of either three, five or six degrees of freedom. Each edge, $e \in E_1$, is transformed into five generic bars placed between the corresponding nodes.

An example of this mapping is shown in figure 4. The nature of each node in figure 4(*b*) is determined by the *local molecular rigid cluster* about each corresponding vertex in the bond-bending network of figure 4(*a*). These locally defined rigid clusters are shown within each rigid body of the body-bar network. All rigid bodies representing a molecular rigid cluster containing one, two or more than two vertices have three, five or six degrees of freedom respectively. Consequently, an isolated atom maps back into an atom with three degrees of freedom. A dimer maps into two rigid objects each having five degrees of

freedom with five independent generic bars between them, and this gives back a single rigid dimer having five degrees of freedom. In general, all dimers, associated with 1-coordinated vertices will become isostaticly rigid [21] to their neighbouring rigid body.

Short of a general graph-theoretic characterization of generic rigidity, it would be insightful to develop combinatorial algorithms that test for generic rigidity on bar-joint networks outside the class of squared graphs, but have no implied-hinge joints. To go beyond networks lacking banana structures, note that proposition 3.3 suggests that if one can *a priori* identify implied-hinge joints, then combinatorial constraint counting will completely characterize generic rigidity for a general graph in three dimensions. This idea falls along the lines of the Dress conjecture [23]. Taking a pragmatic approach of gradually generalizing the network type could ultimately lead to a complete characterization of generic rigidity for any general graph G in three dimensions.

6. Conclusion

A major conceptual advance in the fundamental understanding of mechanical stability in network glasses and macromolecules has been made by bridging the concepts of graph rigidity with material science [14]. Numerous glass-like materials are well modelled as a simple bar-joint structure which is generic in nature. Only the connectivity of the network becomes important in this setting. The surprise is that networks lacking any underlying symmetry are in principle easier to deal with. In practice, the advantage of applying concepts from graph rigidity goes only as far as whether an efficient combinatorial algorithm can be implemented.

It has been demonstrated that Laman's theorem generalizes to three-dimensional bar-joint networks that do not contain implied-hinge joints, allowing constraint counting on all levels of subgraphs to be sufficient to characterize generic rigidity. The approach taken here is more physically intuitive than using projective geometry [2] or matroid theory [23, 25]. The new results that have been put forward offer an avenue for developing algorithms that test for generic rigidity on bar-joint networks directly. In particular, it was shown that implied-hinge joints cannot exist within generic bond-bending networks, which enabled a *three-dimensional pebble game* algorithm, similar to the two-dimensional *pebble game* algorithm [16], to be constructed. It is now possible to count the exact number of floppy modes, determine the rigid cluster decomposition and to identify overconstrained regions within a three-dimensional generic bond-bending network. Moreover, these bond-bending networks are relevant to applications in many covalent network glasses and macromolecules.

Acknowledgments

I thank M F Thorpe for many in-depth discussions about generic rigidity and its relevance to glass networks and thank B Hendrickson for making many valuable suggestions. I would also like to thank Debbie Franzblau, P M Duxbury, C Moukarzel and W Whiteley for fruitful discussions and to Yu-Qing Xiao for helping out with some numerical calculations. This work was partially supported under NSF grant DMR-96 32182.

References

[1] Harary F 1969 *Graph Theory (Addison-Wesley Series in Mathematics)* (Reading, MA: Addison-Wesley)
[2] Tay T S 1984 Rigidity of multi-graphs. I. Linking rigid bodies in n-space *J. Comb. Theory* B **26** 95–112

[3] Bresser W, Boolchand P and Suranyi P 1986 Rigidity percolation and molecular clustering in network glasses *Phys. Rev. Lett.* **56** 2493–6
[4] Tanaka K 1989 Structural phase transitions in chalcogenide glasses *Phys. Rev.* B **39** 1270–9
[5] Boolchand P and Thorpe M F 1994 Glass-forming tendency, percolation of rigidity, and onefold-coordinated atoms in covalent networks *Phys. Rev.* B **50** 10 366–8
[6] Dove M T, Harris M J, Hannon A C, Parker J M, Swainson I P and Gambhir M 1997 Floppy modes in crystalline and amorphous silicates *Phys. Rev. Lett.* **78** 1070–3
[7] Watanabe M and Karplus M 1993 Dynamics of molecules with internal degrees of freedom by multiple time-step methods *J. Chem. Phys.* **99** 8063–74
[8] Jain A, Vaidehi N and Rodriguez G 1993 A fast recursive algorithm for molecular dynamics simulation *J. Comput. Phys.* **106** 258–68
[9] Nichols W L, Rose G D, Ten Eyck R L F and Zimm B H 1995 Rigid domains in proteins: an algorithmic approach to their identification *Proteins: Function, Structure Genetics* **23** 38–48
[10] Vaidehi N, Jain A and Goddard III W A 1996 Constant temperature constrained molecular dynamics: the Newton–Euler inverse mass operator method *J. Phys. Chem.* **100** 10 508–17
[11] Phillips J C 1980 Chemical bonding, internal surfaces, and the topology of non-crystalline solids *Phys. Status Solidi* b **101** 473
[12] Thorpe M F 1983 Continuous deformations in random networks. *J. Non-Cryst. Solids* **57** 355–70
[13] Thorpe M F 1994 Bulk and Surface floppy modes *J. Non-Cryst. Solids* **182** 135–42
[14] Jacobs D J and Thorpe M F 1995 Generic rigidity: The pebble game *Phys. Rev. Lett.* **75** 4051–4
[15] Jacobs D J and Thorpe M F 1996 Generic rigidity percolation in two dimensions *Phys. Rev.* E **53** 3682–93
[16] Jacobs D J and Hendrickson B 1996 An algorithm for two dimensional rigidity percolation: the pebble game *J. Comput. Phys.* accepted
[17] Moukarzel C and Duxbury P M 1995 Stressed backbone and elasticity of random central-force systems *Phys. Rev. Lett.* **75** 4055–8
[18] Moukarzel C 1996 An efficient algorithm for testing the generic rigidity of planer graphs *J. Phys. A: Math. Gen.* 1996 **29** 8079–98
[19] Arbabi S and Sahimi M 1993 Mechanics of disordered solids. I. Percolation on elastic networks with central forces *Phys. Rev.* B **47** 695–702
[20] Sahimi M and Arbabi S 1993 Mechanics of disordered solids. II. Percolation on elastic networks with bond-bending forces *Phys. Rev.* B **47** 703–12
[21] Guyon E, Roux S, Hansen A, Bibeau D, Troadec J P and Crapo H 1990 Non-local and non-linear problems in the mechanics of disordered systems: application to granular media and rigidity problems *Rep. Prog. Phys.* **53** 373–419
[22] Hendrickson B 1992 Conditions for unique graph realizations *SIAM J. Comput.* **21** 65–84
[23] Graver J, Servatius B and Servatius H 1993 *Combinatorial Rigidity (Graduate Studies in Mathematics)* (Providence, RI: American Mathematical Society)
[24] Laman G 1970 On graphs and rigidity of plane skeletal structures *J. Eng. Math.* **4** 331–40
[25] Whiteley W 1996 Some matroids from discrete applied geometry *Contemp. Math.* **197** 171–311
[26] Gluck H 1975 Almost all simply connected closed surfaces are rigid *Geometric Topology (Lecture Notes in Mathematics 438)* (Berlin: Springer) pp 225–39
[27] Roth B 1981 Rigid and flexible frameworks *Am. Math. Mon.* **88** 6–21
[28] Franzblau D 1995 Combinatorial algorithm for a lower bound on frame rigidity *SIAM J. Discrete Math.* **8** 388–400
[29] Lovasz L and Yemini Y 1982 On generic rigidity in the plane *SIAM J. Algebr. Discrete Methods* **3** 91–8
[30] Franzblau D 1996 Generic rigidity of 'molecular' graphs via ear decomposition *Preprint* DIMACS
[31] Whiteley W 1998 Private communication. I am making an assumption that a generic mechanism cannot be constructed in three dimensions such that a pair of vertices can define an implied edge throughout the course of motion of the mechanism. In three dimensions there is no known way to construct such a mechanism. A rigorous proof that such a mechanism in three dimensions cannot be made is still lacking.
[32] Franzblau D 1996 Private communication