

A Semidefinite Programming Approach to Tensegrity Theory and Graph Realization

Anthony So

Department of Computer Science

Manar El-Chammas

Department of Electrical Engineering

Yinyu Ye

Department of Management Science and Engineering and

by courtesy, Electrical Engineering

Stanford University

<http://www.stanford.edu/~yyye>

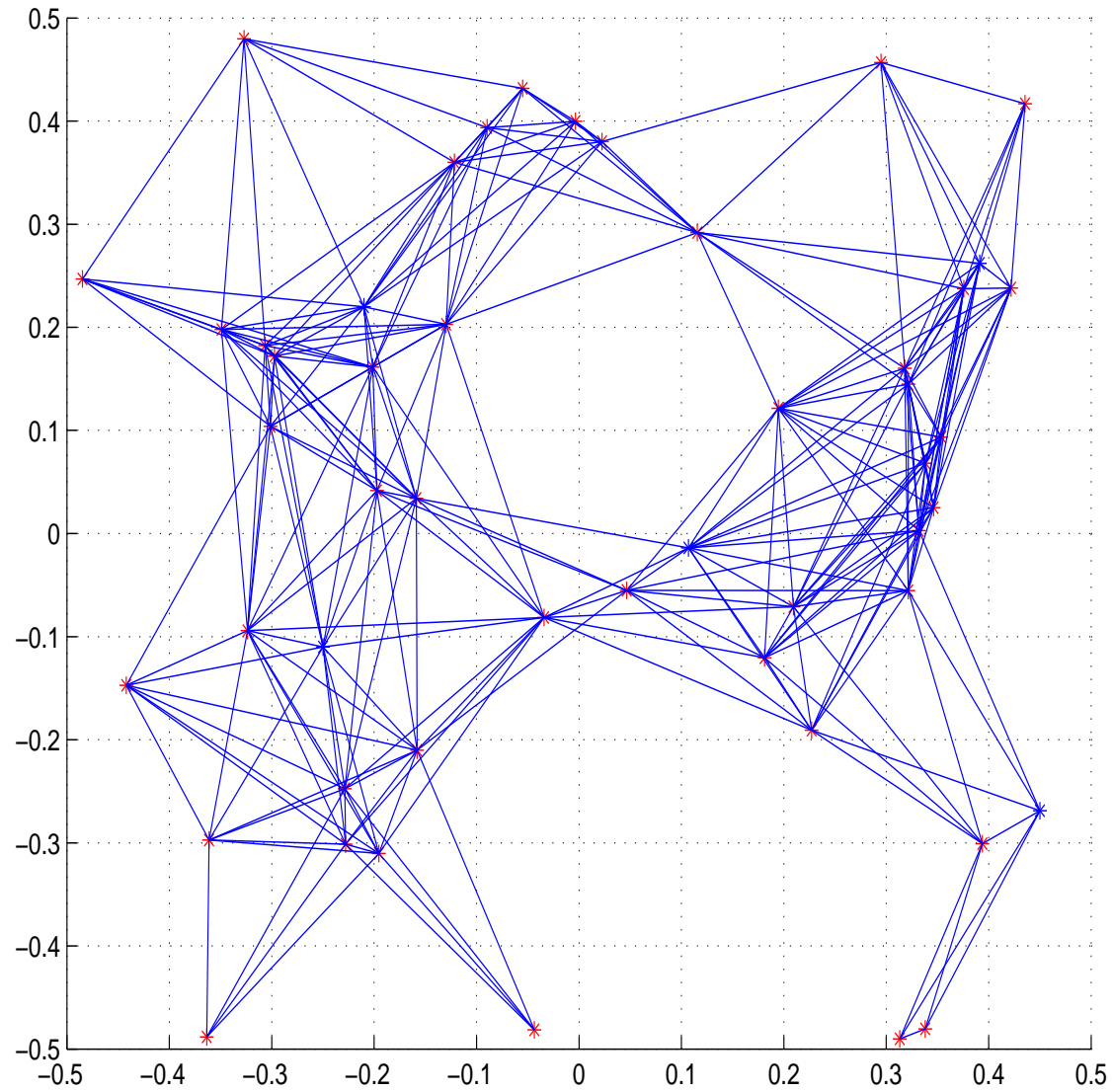
Outlines

- Graph Realization Problem
- d -Realizable Graphs
- SDP Formulation
- Realization Algorithm

The Graph Realization Problem

Given a graph $G = (V, E)$ and a set of non-negative edge weights $\{d_{ij} : (i, j) \in E\}$, the goal is to compute a realization of G in the Euclidean space \mathbf{R}^d for a given dimension d , i.e. to place the vertices of G in \mathbf{R}^d such that the Euclidean distance between every pair of adjacent vertices (i, j) in E equals the prescribed weight d_{ij} .

Figure 1: 50-node Graph Realization in 2D



Applications

- Global Position System (GPS)
- Sensor network localization
- Molecular conformation
- Data dimension reduction
- Euclidean ball packing

Related Work

- Schoenberg and Young/Householder studied the case where all pairwise distances are given. Their theories formed the basis of various **multidimensional scaling** algorithms.
- Barvinok, Pataki, and Alfakih/Wolkowicz used SDP models to show that the problem is solvable in polynomial time if the **dimension** of the realization is not restricted. Moreover, they have given **bounds** on the dimension needed to realize the given distances.
- However, if we require the realization to be in \mathbf{R}^d for some **fixed** d , then the problem becomes **NP-complete** (e.g., Saxe 1979, Aspnes, Goldenberg, and Yang 2004).
- Identify **families of graph instances** that admit polynomial time algorithms for computing a realization in the **required dimension** (Biswas, So, Toh, and Ye 2004-2005, Jin/Saunders 2005; SODA'05, ACM, IEEE, ...).

Today's Talk: d -Realizable Graphs

A graph is d -realizable if it can **always** be realized in \mathbf{R}^d **whenever** it is realizable (the edge weights are Euclidean metric) for every instance of the graph.

- Connelly and Sloughter have recently given a complete **characterization** of the class of d -realizable graphs, where $d = 1, 2, 3$
- It is trivial to find a realization of an 1 -realizable graph, since a graph is 1 -realizable iff it is a **forest**.
- A polynomial time algorithm for realizing 2 -realizable graphs exists: **triangulation**.
- Finding a corresponding algorithm for 3 -realizable graphs is posed as an **open question**.

3-realizable graph I

A graph is 3-realizable iff it does not contain K_5 or $K_{2,2,2}$ as a minor (Connelly and Sloughter 2004).

Figure 2: K-5

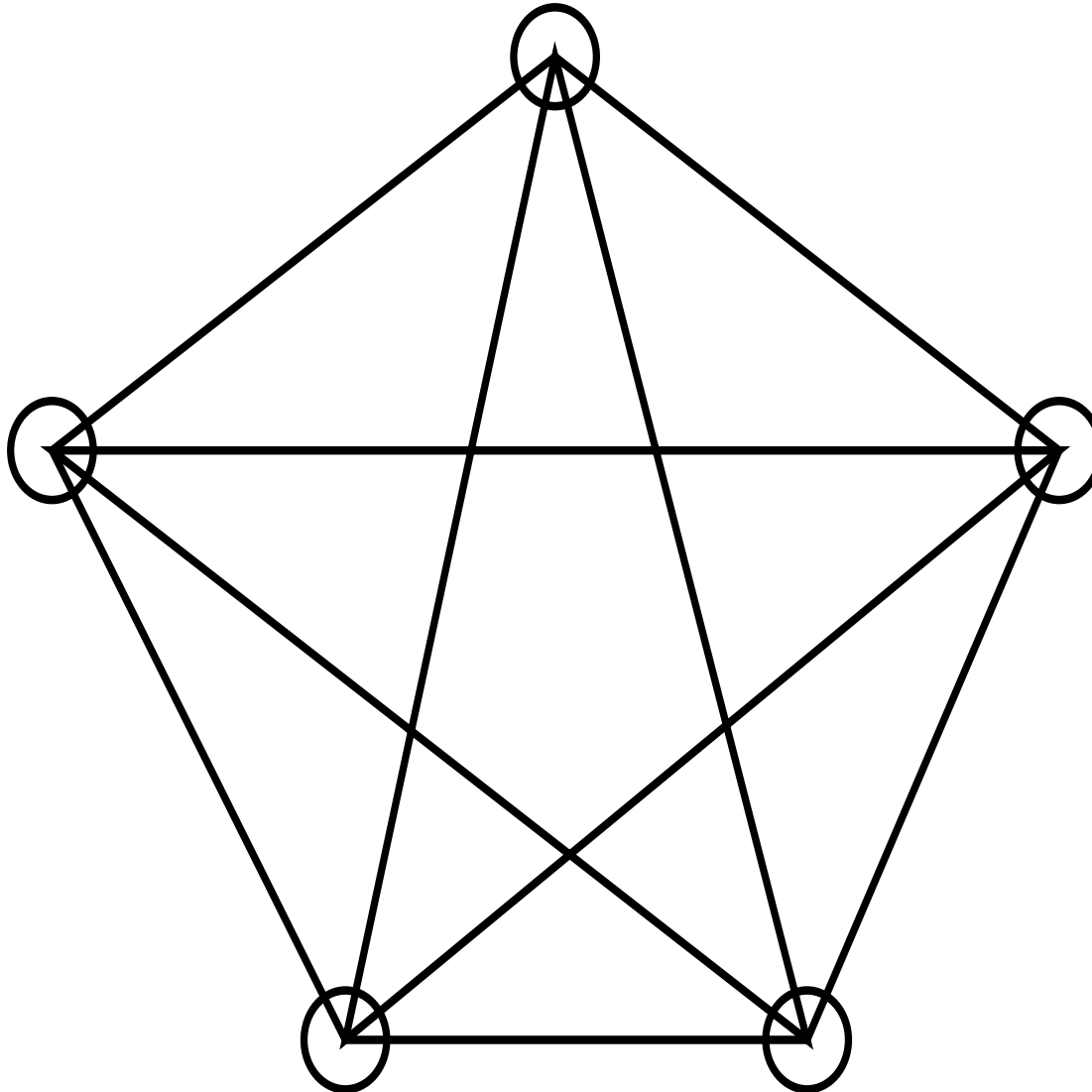
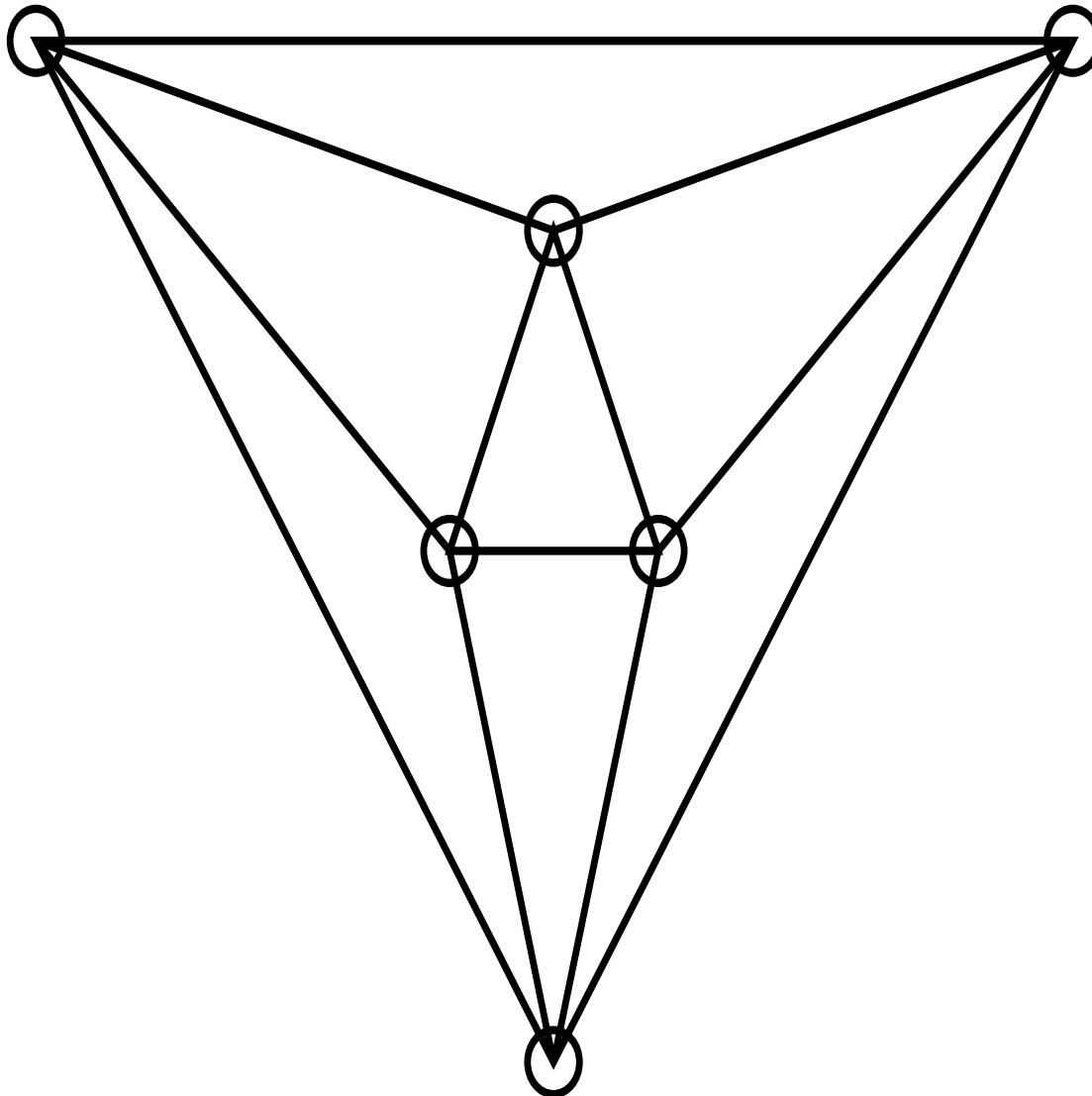


Figure 3: K-2-2-2



3-realizable graph II

Using the forbidden minor characterization of **partial 3-trees**, one can show that a graph is **3-realizable** if it either

- contains an V_8 or an $C_5 \times C_2$ as a **minor**
- or does not contain either graphs as a minor.

Indeed, if it is the latter, then G is a **partial 3-tree**.

An **k -tree** is defined recursively as follows. The complete graph on k vertices is an k -tree. An k -tree with $n + 1$ vertices (where $n \geq k$) can be constructed from an k -tree with n vertices by adding a vertex adjacent to all vertices of one of its k -vertex complete subgraphs, and only to those vertices.

A partial k -tree is a **subgraph** of an k -tree.

Figure 4: V-8

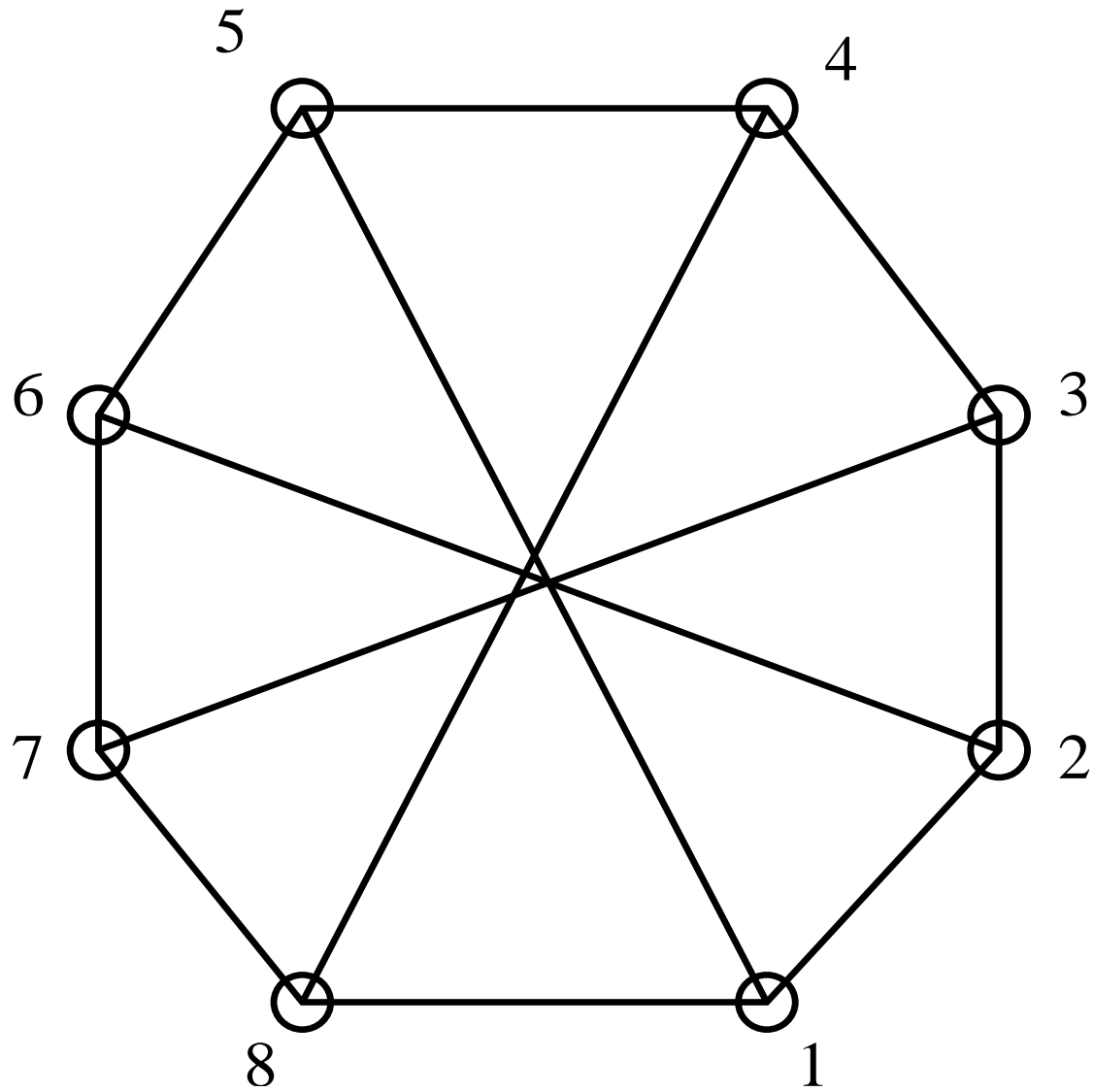
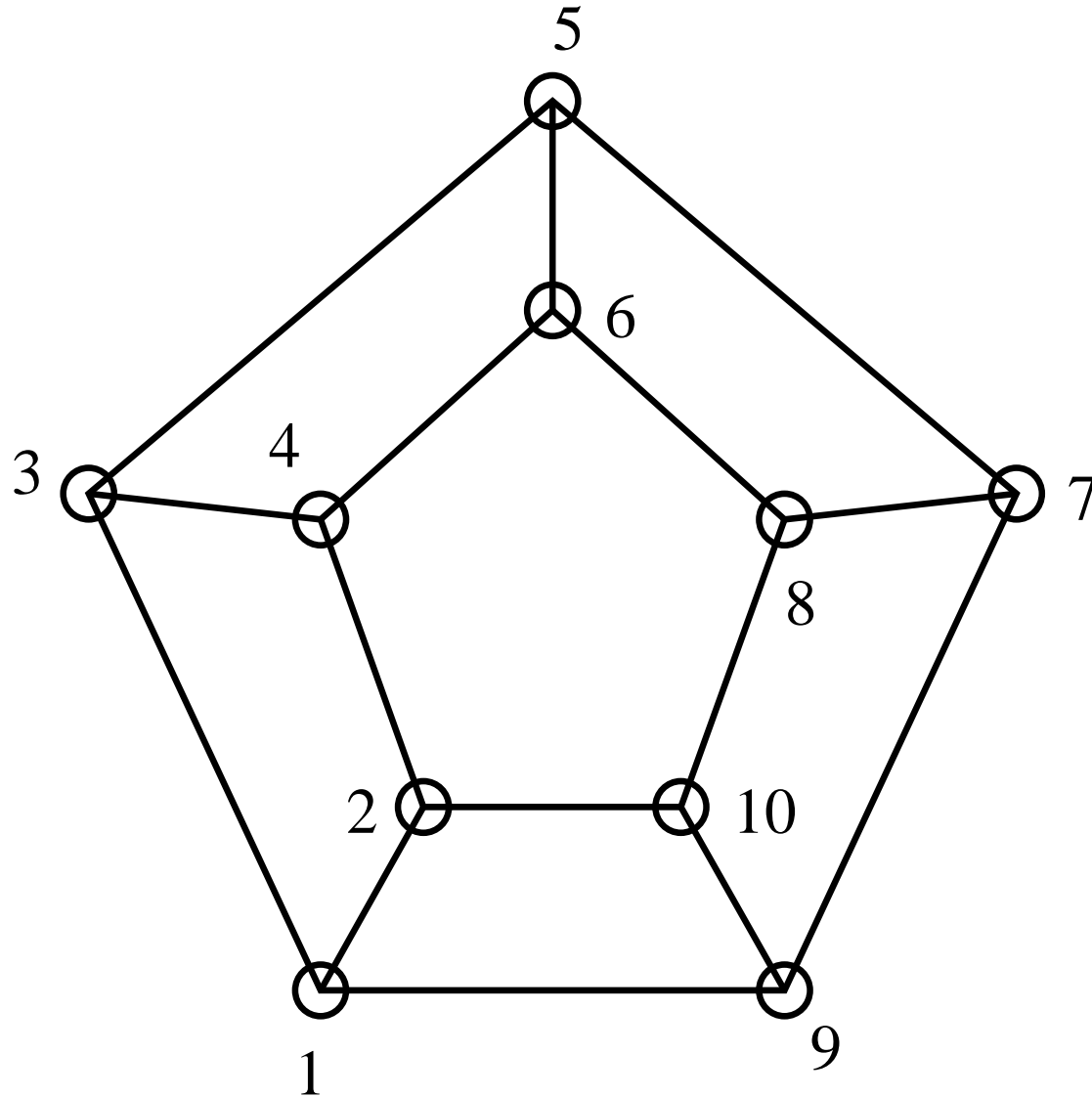


Figure 5: $C_5 \times C_2$ 

Our Result

We resolve the above **open question** by giving a polynomial time algorithm for (approximately) realizing **3-realizable graphs**.

The main bottleneck in the proof is to show that two graphs, V_8 and $C_5 \times C_2$, are **3-realizable**.

There exists a realization \mathbf{p} of $H \in \{V_8, C_5 \times C_2\}$ such that the distance between a certain pair of non-adjacent vertices (i, j) is **maximized**. Such a realization induces a **non-zero equilibrium stress** on the graph H' obtained from H by adding the edge (i, j) . Then use this equilibrium force to prove that H' must be in \mathbf{R}^3 .

We show that the problem of computing the desired \mathbf{p} can be formulated as an **SDP**. More interesting is that the **optimal dual multipliers** of our SDP give rise to a **non-zero equilibrium stress**.

Tensegrity Theory: Realization and Labels

Let $G = (V, E; \mathbf{d})$ be a weighted **connected graph** that contains neither loops nor multiple edges, such that $d_{ij} \geq 0$ for all $(i, j) \in E$.

A **tensegrity** $G(\mathbf{p})$ is a graph $G = (V, E)$ together with a realization $\mathbf{p} = (p_i) \in \mathbf{R}^D \times \dots \times \mathbf{R}^D = \mathbf{R}^{|V|D}$, and a configuration such that each edge is labelled as a **cable, strut, or bar** and each vertex is labelled as **pinned or unpinned**.

The label on each edge is intended to indicate its **functionality**: cables (resp. struts) are allowed to decrease (resp. increase) in length (or stay the same length), but not to increase (resp. decrease) in length; bars are forced to remain the same length.

A **pinned vertex** is forced to remain where it is.

Tensegrity Theory: Equilibrium Stress

An **equilibrium stress** for $G(\mathbf{p})$ is an assignment of real numbers $\omega_{ij} = \omega_{ji}$ to each edge $(i, j) \in E$ such that for each **unpinned vertex** i of G , we have

$$\sum_{j:(i,j) \in E} \omega_{ij} (p_i - p_j) = \mathbf{0}.$$

Furthermore, we say that the equilibrium stress $\omega = \{\omega_{ij}\}$ is **proper** if $\omega_{ij} = \omega_{ji} \geq 0$ (resp. ≤ 0) if (i, j) is a cable (resp. strut).

A Semidefinite Programming (SDP) for Realization

Consider a simple model with C (or S) is a set of cables (or strut):

$$\begin{aligned} \max \quad & \sum_{(i,j) \in S} \|x_i - x_j\|^2 - \sum_{(i,j) \in C} \|x_i - x_j\|^2 \\ \text{s.t.} \quad & \|x_i - x_j\|^2 = d_{ij}^2, \quad \forall (i, j) \in N_x, i < j, \\ & \|a_k - x_j\|^2 = d_{kj}^2, \quad \forall (k, j) \in N_a. \end{aligned}$$

Matrix Representation

Let $X = [x_1 \ x_2 \ \dots \ x_n]$ be the $d \times n$ matrix that needs to be determined. Then

$$\|x_i - x_j\|^2 = (e_i - e_j)^T X^T X (e_i - e_j) \text{ and } \|a_k - x_j\|^2 = (a_k; -e_j)^T [I \ X]^T [I \ X] (a_k; -e_j),$$

where e_j is the vector of all zero except 1 at the j th position.

Matrix Representation

Let $X = [x_1 \ x_2 \ \dots \ x_n]$ be the $d \times n$ matrix that needs to be determined. Then

$$\|x_i - x_j\|^2 = (e_i - e_j)^T X^T X (e_i - e_j) \text{ and } \|a_k - x_j\|^2 = (a_k; -e_j)^T [I \ X]^T [I \ X] (a_k; -e_j),$$

where e_j is the vector of all zero except 1 at the j th position.

$$\begin{aligned} \max \quad & \sum_{(i,j) \in S} (e_i - e_j)^T Y (e_i - e_j) - \sum_{(i,j) \in C} (e_i - e_j)^T Y (e_i - e_j) \\ \text{s.t.} \quad & (e_i - e_j)^T Y (e_i - e_j) = d_{ij}^2, \quad \forall i, j \in N_x, \quad \forall (i, j) \in N_x, \quad i < j, \\ & (a_k; -e_j)^T \begin{pmatrix} I & X \\ X^T & Y \end{pmatrix} (a_k; -e_j) = d_{kj}^2, \quad \forall k, j \in N_a, \\ & Y = X^T X. \end{aligned}$$

where Y denotes the **Gram matrix** $X^T X$.

Semidefinite Programming (SDP)

$$\begin{aligned} (SDP) \quad & \inf \quad C \bullet Z \\ & \text{s.t.} \quad A_i \bullet Z = b_i, i = 1, 2, \dots, m, Z \succeq 0, \end{aligned}$$

where $C, A_i \in \mathcal{M}^n$, the set of n -dimension symmetric matrices.

The dual problem to (SDP) can be written as:

$$\begin{aligned} (SDD) \quad & \sup \quad b^T y \\ & \text{s.t.} \quad \sum_i^m y_i A_i + S = C, S \succeq 0, \end{aligned}$$

where $b = (b_1; \dots; b_m) \in \mathcal{R}^m$, variables $y \in \mathcal{R}^m$ and $S \in \mathcal{M}^n$.

An generalization of linear programming.

SDP Relaxation for Graph Realization

Change

$$Y = X^T X$$

to

$$Y \succeq X^T X.$$

This matrix inequality is equivalent to

$$\begin{pmatrix} I & X \\ X^T & Y \end{pmatrix} \succeq 0.$$

SDP standard form

$$Z = \begin{pmatrix} I & X \\ X^T & Y \end{pmatrix}.$$

Find a symmetric matrix $Z \in \mathbf{R}^{(2+n) \times (2+n)}$ such that

$$\max \left(\sum_{(i,j) \in S} (\mathbf{0}; e_i - e_j)(\mathbf{0}; e_i - e_j)^T - \sum_{(i,j) \in C} (\mathbf{0}; e_i - e_j)(\mathbf{0}; e_i - e_j)^T \right) \bullet Z$$

$$\text{s.t. } Z_{1:d,1:d} = I$$

$$(\mathbf{0}; e_i - e_j)(\mathbf{0}; e_i - e_j)^T \bullet Z = d_{ij}^2, \quad \forall i, j \in N_x, \quad i < j,$$

$$(a_k; -e_j)(a_k; -e_j)^T \bullet Z = d_{kj}^2, \quad \forall k, j \in N_a,$$

$$Z \succeq 0.$$

The Dual of the SDP Relaxation

$$\begin{aligned}
 \min \quad & I \bullet V + \sum_{i < j \in N_x} w_{ij} d_{ij}^2 + \sum_{k, j \in N_a} w_{kj} d_{kj}^2 \\
 \text{s.t.} \quad & \begin{pmatrix} V & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} + \sum_{i < j \in N_x} w_{ij} (\mathbf{0}; e_i - e_j)(\mathbf{0}; e_i - e_j)^T \\
 & + \sum_{k, j \in N_a} w_{kj} (a_k; -e_j)(a_k; -e_j)^T \succeq \\
 & \sum_{(i,j) \in S} (\mathbf{0}; e_i - e_j)(\mathbf{0}; e_i - e_j)^T - \sum_{(i,j) \in C} (\mathbf{0}; e_i - e_j)(\mathbf{0}; e_i - e_j)^T,
 \end{aligned}$$

where variable matrix $V \in \mathcal{M}^d$, variable w_{ij} is the weight on edge from x_i to x_j , and w_{kj} is the weight on edge from a_k to x_j . As we shall see, the optimal w_{ij} are closely related to an **equilibrium stress** for a certain realization of G .

Analysis of the SDP Formulation

Theorem 1. Let $\tilde{X} = [\tilde{x}_1, \dots, \tilde{x}_n]$ be the positions of the unpinned vertices obtained from the optimal primal matrix \bar{Z} , and let $\{\bar{\theta}_{ij}, \bar{w}_{kj}\}$ be a set of optimal dual multipliers. Suppose that we assign the stress $\bar{\theta}_{ij}$ (resp. \bar{w}_{kj}) to the bar $(i, j) \in N_x$ (resp. $(k, j) \in N_a$), a stress of 1 to all the cables, and a stress of -1 to all the struts. Then, the resulting assignment yields a *non-zero proper equilibrium stress* for the realization $\{(a_1; \mathbf{0}), \dots, (a_m; \mathbf{0}), \tilde{x}_1, \dots, \tilde{x}_n\}$.

Proof: The primal is feasible and the dual is strictly feasible. Let \bar{Z} (resp. \bar{U}) be the optimal primal (resp. dual) solution matrix. Then, the absence of a duality gap implies *complementarity*:

$$\bar{Z}\bar{U} = \mathbf{0}.$$

Algorithm Tasks

1. Realizing a partial 3-tree;
2. finding a subdivision of V_8 or $C_5 \times C_2$ in an 3-realizable graph;
3. realizing an V_8 and its subdivisions;
4. realizing an $C_5 \times C_2$ and its subdivisions.

1: Realizing Partial 3-Trees

Suppose that we are given a 3-tree G with feasible edge lengths, and that G is constructed by adding the vertices v_1, v_2, \dots, v_n , in that order. Then, to find a realization of G in \mathbf{R}^3 can be done in **linear time**. A partial 3-tree can be completed into a 3-tree by solving an SDP.

2: Finding a Subdivision of V_8 or $C_5 \times C_2$

Let G be an $\mathfrak{3}$ -realizable graph. We now show how the algorithm of Matoušek and Thomas can be used to obtain a subgraph of G that is a subdivision of V_8 or $C_5 \times C_2$. We shall also use the term “homeomorphic” for subdivision – a graph H_1 is homeomorphic to H_2 if H_1 is a subdivision of H_2 .

1. (Asano) For an $\mathfrak{3}$ -connected graph H , a graph H' has a subgraph homeomorphic to H iff there is an $\mathfrak{3}$ -connected component of H' that has a subgraph homeomorphic to H .
2. (Connelly and Slougher) If an edge is added between a non-adjacent pair of vertices of V_8 (resp. $C_5 \times C_2$), then the resulting graph has K_5 (resp. K_5 or $K_{2,2,2}$) as a minor.
3. (Connelly and Slougher) Let G be an $\mathfrak{3}$ -realizable graph. Suppose that G contains a subdivision of H , where $H \in \{V_8, C_5 \times C_2\}$. Remove the subdivision of H from G and consider the components of the resulting graph.

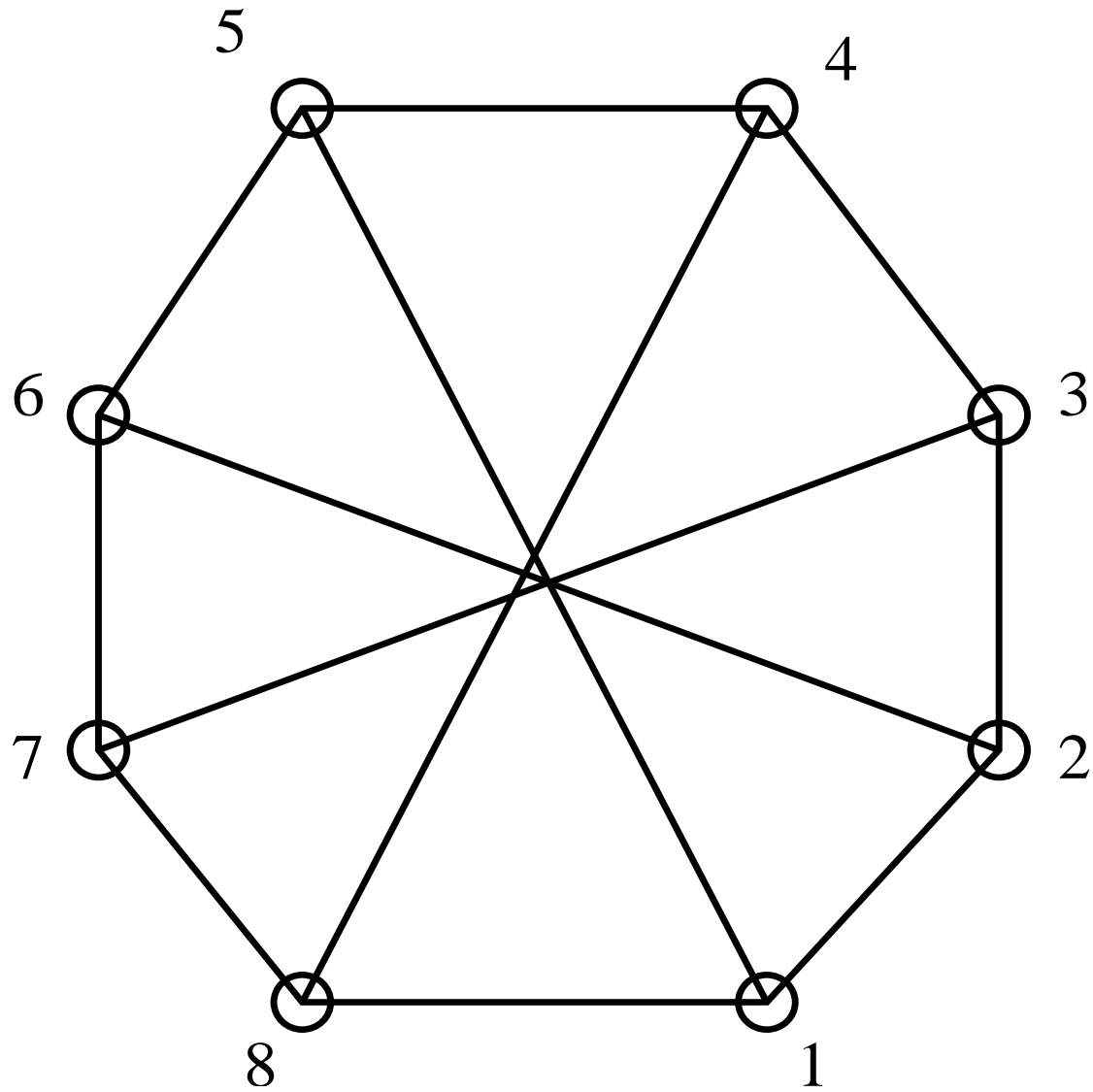
Then, each component is connected in G to exactly one of the subdivided edges of H .

Theorem 2. *Let G be an $\mathfrak{3}$ -realizable graph containing a subgraph homeomorphic to $H \in \{V_8, C_5 \times C_2\}$. Then, one of the **triconnected components** of G is isomorphic to H .*

Algorithm: First, decompose G into triconnected components. Then, we check each of the triconnected components for the presence or absence of V_8 or $C_5 \times C_2$. For this we can run the algorithm on each of those components and see if the component reduces to a **null graph** or not. If the component does not reduce to a null graph, then it is isomorphic to either V_8 or $C_5 \times C_2$, and the number of vertices in the component will determine which one it is. The desired subdivision can then be extracted from G .

Proposition 1. *Let G be an $\mathfrak{3}$ -realizable graph with n vertices. Then, a subdivision of V_8 or $C_5 \times C_2$ in G can be found in $O(n)$ time.*

Figure 6: V-8



Rank of SDP Solutions

We can find an SDP solution whose rank r satisfying

$$\frac{r(r+1)}{2} \leq m,$$

where m is the number of constraints.

This general result falls short what we hope for.

But an optimal stress associated with the chosen SDP objective will help ...

4: Realizing $C_5 \times C_2$ and its Subdivisions

The graph $C_5 \times C_2$ is 3-realizable. We first augment $C_5 \times C_2$ to G by adding a **strut** between vertices **1** and **6**, and we **pin vertex 1** at the **origin**.

Putting Everything Together

Theorem 3. *There is a polynomial time algorithm for (approximately) realizing $\mathfrak{3}$ -realizable graphs.*

The Kissing Problem

- Given a unit sphere at the center, the maximum number of unit spheres, in d dimensions, can touch or **kiss** the center sphere?
- $K(1)=2$, $K(2)=6$; general Solutions does not exist.
- Delsarte Method uses **linear programming** to provide an upper bound on the number of spheres.
- $K(8) = 240$, $K(24) = 196650$.
- $K(4) = 24$: proved using Delsarte Method by Oleg Musin only 3 years ago.
- For other dimensions, lower bounds have been provided by constructing a **lattice structure**. There also exists a bound using the **Riemann zeta** function, but is **non-constructive**.

Our Approach

Given n unit spheres, can they all kiss the center sphere in d dimension space?

- Treat it as a **graph realization**;
- Use **SDP** to provide a lower bound;
- Offer a completely **constructive** approach.

The Kissing Problem as a Graph Realization

- Can be formulated as a SDP feasibility problem; but SDP solution may not provide proper **rank**.

$$\begin{aligned} (e_i - e_j)^T Y (e_i - e_j) &\geq 4, \quad \forall i \neq j, \\ e_i^T Y e_i &= 4, \quad \forall i \\ Y &\succeq \mathbf{0}. \end{aligned}$$

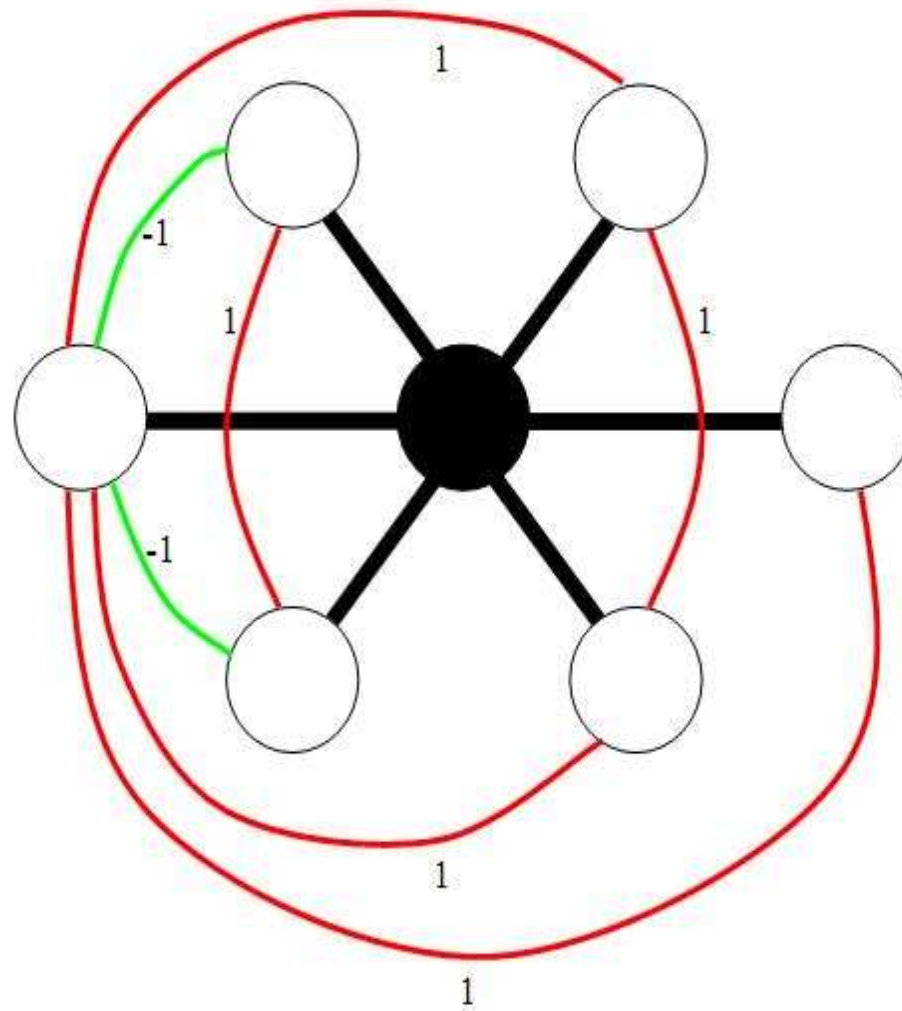
- Construct a **nonzero** SDP objective function to reduce the rank of a solution.

$$\begin{aligned} \min \quad & C \bullet Y, \\ \text{s.t.} \quad & (e_i - e_j)^T Y (e_i - e_j) \geq 4, \quad \forall i \neq j, \\ & e_i^T Y e_i = 4, \quad \forall i \\ & Y \succeq \mathbf{0}. \end{aligned}$$

The objective construction

- Use **pull** some struts and/or **push** some cables in order to force SDP solution into low rank.
- For example, for 2 dimensions, 6 spheres can be connected as follows (thick lines are bars, red lines are struts, green lines are cables).

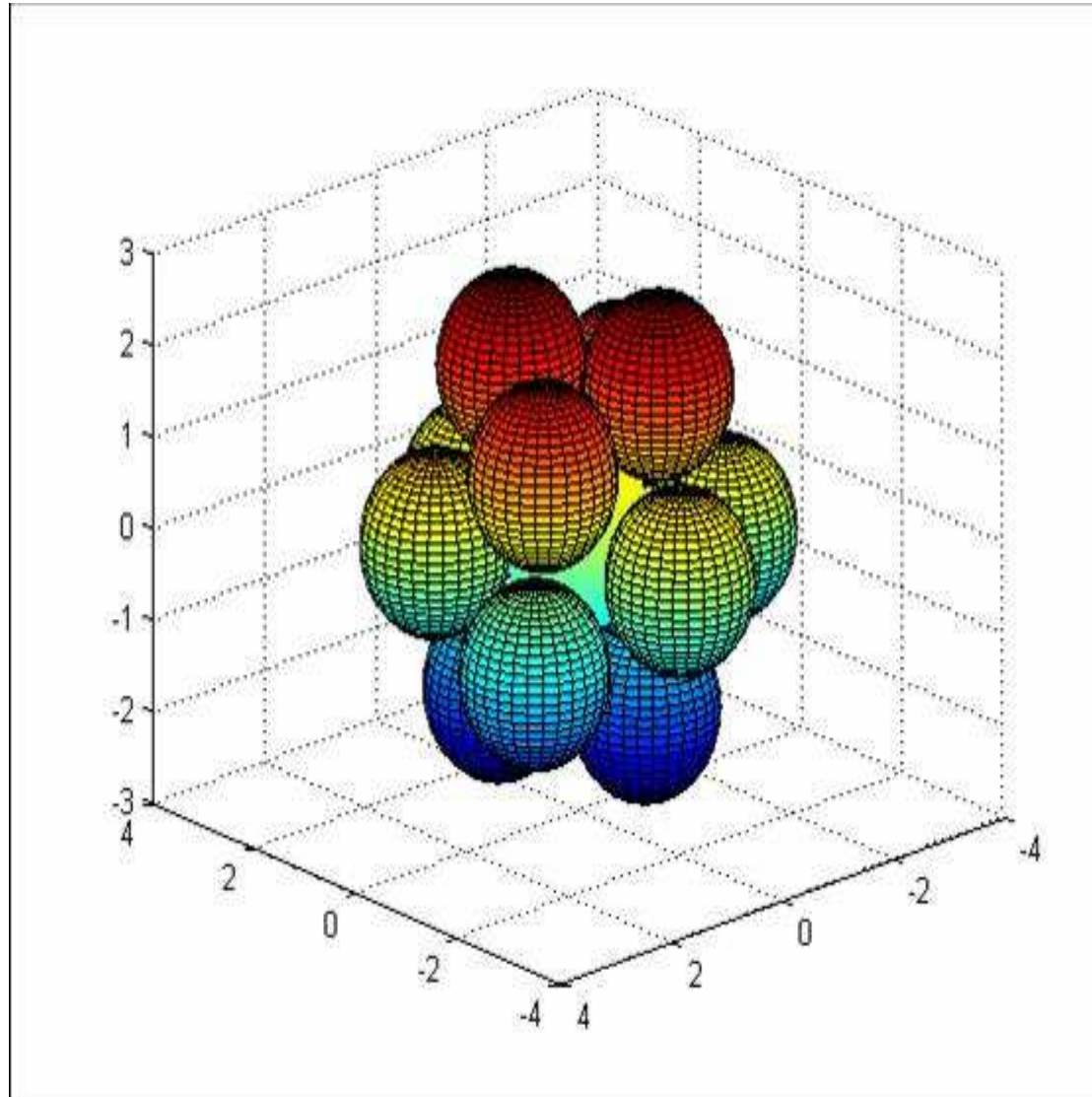
Figure 7: 6 Spheres in 2-D



Realizing the 3-D and 12-Sphere Kissing Problem

This objective structure can be extended to dimension 3. For 12 spheres, SDP method provides the following realization

Figure 8: 12 Spheres in 3-D



Conclusion

- We have studied a connection between SDP and **tensegrity theories**, as well as the notion of **d -realizability of graphs**.
- We have shown that the problem of finding an **equilibrium stress** can be formulated as an SDP. This gives a constructive proof of (a variant of) a result in tensegrity theory that is previously established by **non-constructive means**.
- We then combine this result with other techniques to design an algorithm for realizing **3-realizable graphs**, thus answering an **open question** posed before; and realizing the 3-D Kissing graph.
- We believe that our techniques can be applied to derive some other interesting properties of **tensegrity frameworks**.